Sensur av hovedoppgaver
Høgskolen i Buskerud
Fakultet for Teknologi

Prosjektnummer: 2013-5
For studieåret: 2012/2013
Emnekode: SFHO-3200

Prosjektnavn
LetsDance

Utført i samarbeid med: Sunnaas sykehus og Mektron

Ekstern veileder: Egil Utheim

Sammendrag: Vår oppgave gikk ut på å utvikle et dansespill for Sunnaas sykehus. Spillet skal fungere som rehabilitering for pasienter på Sunnaas. Dette ble gjort med bevegelses-sensor (Kinect) og Unity3D spillmotor. LetsDance tillater brukerne å tilpasse spillet etter sine behov. Brukeren kan også laste opp sin egen musikk som det kan danses til. Prestasjoner og danser som blir utført blir lagret slik at det er mulig å følge progresjonen sin.

Stikkord:
- Rehabilitering
- Dansespill
- Motion tracking

Tilgjengelig: JA

Prosjekt deltagere og karakter:

Navn	Karakter
Dagfinn Kjærnet	
Henrik Olsson	
Jan Erik Helskog	
Marcus Jernberg	
Are Oven	
Zana Ali	

Dato: 12. Juni 2013

Karoline Moholth	Hallstein Asheim Hansen	Bjørn Sundby
Intern Veileder	Intern Sensor	Ekstern Sensor

Dokumentoversikt

Dette er en oversikt over hvor man kan finne de forskjellige dokumentene.

Innbundet vil si at det er tilgjengelig i dokumentasjonen som er å finne på biblioteket.
Sensor perm vil si at det er tilgjengelig i permen som leveres til sensoren(e).
Digitalt vil si at det er tilgjengelig på CD.
Webside vil si at dokumentasjonen ligger tilgjengelig på prosjektets hjemmeside.

Dokumenter	Innbundet	Sensor perm	Digitalt	Webside
Sluttrapport	X	X	X	
Prosjektplan	X	X	X	X
Kravspesifikasjon	X	X	X	
Game Design Dokument	X	X	X	X
Utviklingsmetodikk	X	X	X	
Teknologidokumenter	X	X	X	
Risikodokument	X	X	X	
Analysedokument	X	X	X	
Designdokument	X	X	X	
Testdokumentasjon	X	X	X	
Testcases	X	X	X	
Brukermanual	X	X	X	
Vidreutviklingsdokument	X	X	X	
Kildekodedokumentasjon		X	X	
Kodestandard og review			X	
Sprintdebriefer	X	X	X	
Product backlog		X	X	
Sluttprodukt (Spillet)			X	
Timelister			X	
Budsjett og kjørelister			X	
Dokumenter fra Mektron			X	
Idedokument			X	
Dokumentoversikt			X	
Møtedokumentasjon			X	
Presentasjoner			X	
Prosjektkontrakt			X	

Sluttrapport

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Kontrollert av:	JEH, ZA, AO, DK, HO, MJ

Innholdsfortegnelse

1. Innledning

I prosjektperioden har gruppen utviklet et system for oppdragsgiver, dette dokumentet har som hensikt å kartlegge hvordan prosjektet har fungert, hvilke avgjørelser som har blitt gjort, trekke konklusjoner og greie ut om prosjektmodell.

2. Introduksjon til prosjektet

Vi har jobbet med et prosjekt som vi har fått tildelt av Sunnaas Sykehus i samarbeid med Mektron AS. I prosjektperioden skulle vi utvikle en løsning til pasienter ved Sunnaas Sykehus som sliter med nedsatte motoriske evner. Pasientene trener ofte sammen med ergoterapauter og andre fagfolk for å forbedre bevegelsesmuligheter eller stoppe en negativ utvikling, men denne treningen kan ofte være kjedelig og lite engarsjerende. Derfor er LetsDance en mulig løsning på problemet, ved å lage et dansespill får man kjempet mot begrensningene med fart, takt og moro i en kanskje trist tid av livet. Vi har laget spillet så det er muligheter for å tilpasse det etter brukerens behov for å nå en stor målgruppe. Samtidig vet vi at det er motiverende å se fremgang, derfor fører vi statistikk over progresjonen og gir premier for gjennomførte oppgaver.

3. Prosjektstyring

Vi bestemte oss for Scrum som prosjektmodell da vi ville ha noe som var agilt, riktignok er vi avhengig av at vi har et ferdig produkt i slutten av prosjekt perioden, men fordelen med Scrum er at vi lett kan tilpasse oss nye endringer og ønsker fra oppdragsgiver. Vi har hele tiden hatt tett kontakt med oppdragsgiver for å kartlegge ønskene, vi var flere ganger på Sunnaas sykehus i starten av prosjektperioden for å hente input fra mennesker med forskjellig bakgrunn. I følge Scrum prosjektmodellen er det da opp til produkteier å tolke ønskene til oppdragsgiver for å gjøre det forståelig for utviklingsteamet. Dagfinn har vært den definerte produkteier samtidig som han har stilt som gruppeleder. Da gruppen stort sett har vært samlet i møtene med oppdragsgiver så har alle inntatt rollen i å tolke ønsker. I møtene med Sunnaas sykehus har vi fått muligheten til å være i kontakt med potensielle sluttbrukere, vi var observatører på spillsesjoner som vi dokumenterte. Samtidig har vi fått muligheten til å delta på arrangementer sammen med Mektron, dette har vært veldig inspirerende for hele gruppen.
I tillegg til møter med oppdragsgiver har vi hatt interne møter nesten hver uke med intern veilder som gitt oss god rådgivning i alt fra prosjektstyring til personlig utvikling.

I Scrum har vi det som kalles for sprinter, dette er små iterasjoner hvor gruppen fokuserer på forskjellige oppgaver på et system for å kunne få en forestilling av det fremtidige produktet til et angitt tidspunkt. Etter hver sprint har vi ført rapport kalt en sprint debrief, hvor vi fører opp alle medlemmene i teamet; hva de har jobbet med og hvordan ting har gått.
Samtidig med dette har vi fulgt Scrum ved å holde agile møter hvor vi er effektive, presenterer kjapt hva vi jobber med og hvilke utfordringer vi har kommet over. Disse møtene har ikke blitt dokumentert da vi de har foregått hyppig, siden vi har sittet tett intill hverandre.

Vi har dokumentert arbeid utført på forskjellige krav ved hjelp av at webbassert program som heter OnTime, dette kunne vært gjort noe bedre. Som en backup har vi alltid ført timer vi har jobbet med prosjektet, der har vi som regel oversikt over hvilke prosjektrelaterte oppgaver vi har jobbet med både i felleskap og inviduelt. Innbyrdes har vi brukt OnTime mest som en form for kravspesifikasjon og liste over oppgaver som må bli gjennomført og hvem som tar på seg de forskjellige oppgavene.
I Scrum har vi det som kalles for en productbacklog dette er en agil liste med små fragmenter av krav som er ønsket av både oppdragsgiver og innad i gruppen. Dette er i utgangspunktet scrum sitt svar

på en kravspesifikasjon kjent fra andre prosjektmodeller, men etter oppfordring fra sensor så har vi utformet en kravspesifikasjon, hvor krav har forskjellig gradering etter hvor viktig de er.

Siden vi alle har sittet tett intill hverandre har vi hatt lett tilgang på det å dele arbeidsoppgaver når vi har støtt på utfordringer, alle har vært hjelpsomme og bidratt når oppgaver har vært vanskelig.

4. Avgjørelser

Hvert eneste medlem i gruppen har jobbet med forskjellige i gjøremål igjennom prosjektperioden, gruppen har stått ovenfor en del valg både når det gjelder det administrative og rent tekniske valg. De store valgene som er essensielle i sammenheng med opplevelsen av sluttproduktet har blitt avgjort i plenum, hvor de forskjellige gruppemedlemmene har presentert sitt forslag for resten av gruppen etterfulgt av at vi har blitt enige om hvilken av løsningene som er best. Når det kommer til små tekniske detaljer som for eksempel en sekvens i kildekoden som gjør en intern operasjon har det ikke vært nødvendig å lufte i gruppen.
Det er klart en fordel at vi alle har sittet på et lite rom, da vi har vært i stand til å bare stille et spørsmål ut i lufta og fått respons umiddelbart.

Administrative avgjørelser som valg av teknologier er spesifisert i teknologidokumentene, andre tekniske avgjørelser er dokumentert i designdokumentet. Eventuelle tekniske forbedringer som ikke har blitt implementert i produktet vil være dokumentert i dokumentet for videre utvikling. I dokumentet for videre utvikling legger vi også ut om alle ambisiøse ideer til et mulig fullstendig komersielt produkt. På lik linje med designvalg så er det en del områder innenfor vår oppgave som er aktuelle innenfor forskningen den dag i dag. Valg, forsøk og research i denne kategorien vil havne i research dokumentet.

5. Evaluering av produktet

Vi er godt fornøyd med hva vi har fått til. Vi har en spillbar prototype som oppfyller alle de viktigste kravene og føler vi har tatt mange gode valg og vurderinger underveis. Vi syntes det er litt trist at vi ikke har fått testet prototypen med spillgruppen på Sunnaas, slik at vi kunne gjøre endringer utfra tilbakemeldingene vi da hadde fått. Generelt bare det å få tilbakemeldinger fra målgruppen var noe vi så frem til. Vi har for øvrig fått tilbakemeldinger fra noen av de ansatte på Sunnaas som virket svært fornøyd med hva vi hadde fått til så langt og vi gleder oss til de nå får se den endelige prototypen.

6. Personlige evalueringer

6.1 Zana Ali

I dette prosjektet har jeg opplevd utrolig mye. Kunnskap jeg har dannet meg gjennom tidligere år har blitt brukt og ny kunnskap har blitt tilegnet. Jeg føler at jeg har utviklet meg veldig som ingeniør, og ikke minst som person. Mye av grunnen til dette er prosjektgruppen, oppdragsgiver og involverte veiledere og sensorer. Jeg er veldig takknemlig for å ha jobbet med disse personene. Arbeidet med prosjektet har blitt enklere å jobbe med på grunn av de involverte i prosjektet.

Det å jobbe i en større gruppe på seks og forholde seg en oppdragsgiver i hovedprosjektfaget har vært en ny og spennede ordning jeg har likt å jobbe etter. Vi var så heldige å ha en dyktig veileder i Karoline som kom med gode råd og tilbakemeldinger gjennom hele prosjektet.

Det å jobbe etter en prosjektmodell som Scrum har vært lærerikt, modellen passet vår gruppe og fungert godt. Utover prosjektperioden har jeg blitt bedre og bedre kjent med Scrum modellen. Samarbeidet innad i prosjektgruppa har vært enestående. Gjennom hele perioden har alle vært behjelpelige og samarbeidsvillige. Vi har vært på samme bølgelengde og prioritet prosjektet i stor grad. Derfor har det også blitt en del sene netter med hovedprosjekt når det har vært nødvendig, med hele gruppen tilstede.

Oppgaven vi var så heldige å få har gjort det veldig lett å være engasjert gjennom prosjektet. Den har vært veldig spennende å jobbe med. Jeg har alltid følt meg velkommen alle turene på Sunnaas og ergoterapeutene har vist stor interesse for prosjektet og kommet med nyttige tilbakemeldinger som vi har tatt til oss. Målet med dette prosjektet var å utvikle et dansespill som rehabilitering, underveis skulle konsepter testes ut. Jeg vil si vi har kommet i mål med disse målene og er godt fornøyd med det.

Alt i alt er jeg veldig fornøyd med prosjektet og sluttproduktet.

6.2 Marcus Jernberg

Det var vært en utrolig lærerik prosess å ta del i dette prosjektet. Jeg har lært mye om å jobbe i gruppe på et så omfattende prosjekt. Særlig det å kunne bruke de kunnskapene og erfaringene fra tidligere skoleår, og samtidig lære utrolig mye nytt har vært veldig gøy.

Mitt ansvarsområde i dette prosjektet var i utgangspunktet risikoansvarlig. Siden vi har jobbet med SCRUM som prosjektmodell har vi enkelt kunne ta høyde for dette, siden vi underveis setter opp sprinter med bakgrunn i viss hastighet og gjennomføringsevne. Det har vært veldig lærerikt å jobbe i SCRUM, alle på gruppa har gjennom hele prosjektet vært flinke til å ta til seg arbeidsopggaver, noe som gjør det veldig enkelt å bruke SCRUM.

Jeg føler vi har var veldig heldige som fikk denne oppgaven, dette er et tema som er veldig i vinden for tiden. Vi fikk lov å forme veldig mange av kravene til produktet selv, noe vi har taklet bra. Vi har under hele prosjektet jobbet tett opp mot Sunnaas og fått mange konstruktive og positive tilbakemeldinger fra dem gjennom prosjektet.

Medlemmene på gruppa kjente hverandre godt før vi begynte på prosjektet, så det var en veldig naturlig gruppesammensetning. Alle har vært gode til å ta seg tid til å hjelpe andre på gruppa som har stått fast og motivere hverandre til å stå på. Jeg syntes gruppa har taklet utfordringer utrolig bra og alle har fått vist frem sine styrker, og samtidig har vi styrket hverandres svakheter.

Jeg er stolt av å kunne levere vårt endelige produkt til HIBU og Sunnaas sykehus. Fra idè til ferdig produkt har det vist seg å være en oppgave som har fattet interesse fra flere kanter av Norge, og jeg tror slike produkter som dette er noe vi kommer til å se mye av i fremtiden. Det har vært veldig givende å jobbe på et prosjekt som dette, og det er lærdommen herfra er absolutt noe jeg kommer til å ta med meg videre.

6.3 Henrik Olsson

Jeg synes at det er flott at man blir kastet veldig ut i prosjektperioden ved at man må kontakte bedrifter alene på en proffesjonell måte. Vi tok en oppgave som var anneledes, da vi er opptatt av ny teknologi og forskning. Det skulle vise seg at vi stod veldig fritt til hva vi skulle utvikle, etter å ha fått en del input fra Egil Utheim så bestemte vi for at vi skulle bruke Kinect til å følge pasientenes bevegelser. Jeg likte dette veldig godt da jeg føler at flere fagfelt i Norge burde moderniseres med

nyteknologi for eksempel "gameification for future healthcare". Det var fantastisk å se at ergoterapautene

Prosjektperioden har vært veldig spennende, det har gitt meg et innsyn i hvordan hverdagen er som ingeniør. Vi har vært så heldige å få et eget prosjektrom, her har gruppen oppholdt seg ofte gjennom prosjekt perioden. Vi kan si at vi har nesten vært her like ofte som en vanlig arbeidsuke, noen ganger oftere. Vi har stått på og det er ingen tvil om at vi har jobbet mot toppkarakter hele veien. Fordelen med prosjektperioden fremfor vanlige forelesninger er at vi får gjort veldig mye håndfast. Samtidig tar vi erfaringene vi har fått i alle kursene tidligere på høgskolen og bruker dette i prosjektet, samtidig som vi lærer oss det nye og utvikler oss.

I gruppen er vi en sammensetning av seks forskjellige mennesker, vi har alle våre forskellige kvaliteter så sammen er vi sterke. Om den kunnskapen ikke har strekt til hos noen så har ikke det vært en stopper for å gjennomføre prosjektet da alle har vært hjelpsomme og stilt opp for hverandre. Produktet vi har utviklet sammen har vært veldig interessant, vi fikk kombinert våre tekniske kunnskaper med en utfordring som eksisterte i helse industrien. I tett samarbeid med Sunnaas og Mektron har vi fått viktig innput og lærdom. Den tette kontakten har vært opprettholdt gjennom møter på Sunnaas, møte i oslo, telefonsamtaler og epostutveksling. Jeg er stolt over at vi har vist iver og engasjement ovenfor Sunnaas og Mektron.

Min rolle i prosjektet har vært Testansvarlig, dette ansvaret er ikke alltid like aktuell gjennom hele prosjektperioden. Selvom vi har hatt bestemte roller innefor prosjektstyringen føler jeg at alle har fått muligheten til å stå frem og være med på å lede gruppen fremover. I alle møtene har vi rullert hvem som skal være møteleder og referend, da får alle prøvd å lede ordet og det er viktig. Jeg føler at jeg har vært proaktiv, og at jeg har deltatt aktivt under diskusjoner. Jeg vil si at jeg også har vært opptatt av at alle skal bli hørt.

Vi har alle dårlige dager noen ganger, dette har ikke prosjektet tatt skade av. Vi har tatt vare på hverandre og hvert gode kamerater selvom det til tider har vært høyt temperatur i diskusjonene.

Vi har vært så heldige å blitt lagt merke til av mediene, vi har et spennende prosjekt som er aktuelt for mange. Jeg føler at vi dermed har vært gode ambassadører for skolen utad.

6.4 Jan Erik Heskog

Gjennomføringen av dette prosjektet har vært veldig givende. Igjennom prosjektet har jeg fått veldig mye nyttig erfaring når det kommer til prosjektarbeid. Da spesielt samarbeid og koordinering i gruppe over lengre tid. Samtidig har jeg utviklet meg mye som ingeniør igjennom prosjektet ved å bruke kunnskap jeg har lært tidligere i utdanningen. Selve oppgaven har vært utrolig spennende å arbeide med. Da prosessen fra kun et enkelt konsept til ferdig produkt har vært fylt med mange utfordringer, som jeg synes gruppen har taklet på en god måte. Det har vært utrolig givende å ha en oppgave innen et felt som er veldig i vinden for tiden. Vi var såpass heldige med oppgaven at det var enkelt å være engasjert igjennom hele prosjektet

Det å jobbe etter en annen prosjektmodell en RUP har vært lærerikt. Prosjektmodellen har passet gruppen bra. Da alle har vært flinke til å ta på seg oppgaver underveis i prosjektet. Jeg var veldig klar for å bruke Scrum som prosjektmodell og prøve noe nytt. Det at oppgave var en ren Software oppgave var en av grunnene til at vi ville bruke Scrum. Samtidig som vi ville prøve en modell som var mye brukt i industrien. Vi har klart å bryte oppgavene ned så mye at alle har hatt noe å gjøre hver sprint. Jeg er godt fornøyd med hvordan Scrum har fungert for gruppen, selv om vi noen ganger var på vei inn i RUP tankegang. Da ble vi pisket tilbake i Scrum av veileder.

Rollen jeg har hatt i prosjektet er webansvarlig. Vi valgte å ta i bruke Google Sites til vår hjemme side. Da dette ga oss en god mulighet til å bruke andre Google tools på en enkel måte. Ellers har jeg hatt en rolle der jeg har hjulpet til når det trengtes. Utover dette har jeg arbeidet mye med design og analyse av menyen og har hatt hovedansvaret for det. I implementasjonen har jeg hovedsakelig arbeidet med menyen, menyflyten og funksjoner i menyen, men har hjulpet på andre områder når det var nødvendig.

Jeg synes samarbeidet i gruppen har fungert veldig godt og vi har unngått å bli uvenner. Utfordringer som har dukket opp underveis har blitt løst. Alle har stått på igjennom hele prosjektet, selv om vi har gått hverandre på nervene noen ganger.

Jeg er veldig fornøyd med sluttproduktet vi har klart å utvikle til Sunnaas og Mektron, og det er godt dokumentert slik at videre utvikling av spillet kan skje så smertefritt som mulig. Prosjektgjennomføringen ovenfor skolen er jeg meget godt fornøyd med, vi fikk utdelt et supert prosjektrom, den interne veilederen har vært helt fantastisk igjennom hele prosjektet.

6.5 Are Oven

I løpet av hovedprosjektet har jeg lært veldig mye om det å jobbe sammen i en litt større gruppe. Noe av det viktigste jeg har lært er at godt samarbeid handler veldig mye om å ha god kommunikasjon, med en gang man ikke kommuniserer fungerer ikke samarbeidet. Det har vært noen utfordringer, men dem synes jeg vi har taklet bra. Gjennom hele denne prosessen har jeg lært veldig mye nytt samtidig som jeg har fått god nytte av det jeg har lært gjennom tiden på skolen her.

Vi fikk oppgaven vår av Sunaas Sykehus i samarbeid med Mektron, den gikk ut på å lage et dansespill. Spillet skal være en underholdende måte å trene på, istedenfor å gjøre repeterende standardoppgaver som man vanligvis gjør. Jeg har bare godt å si om både Sunaas, som har vært veldig engasjerte i prosjektet og den eksterne veilederen vår som har vært utrolig hjelpsom og har dratt oss med på mange forskjellige arrangement innen bruken av spill til rehabilitering.

Når vi startet denne oppgaven valgte vi å bruke Scrum som prosjektmodell da det var denne som vi følte ville fungere best med denne oppgaven. Scrum er veldig annerledes enn RUP som jeg er vant til å jobbe med. I Scrum har alle like mye ansvar for at ting blir gjort. Vi var avhengig av at alle på gruppen jobbet godt og det syns jeg har fungert veldig bra, alle har stått på og gjort jobben sin. I starten når Scrum var helt nytt for oss, var det en veldig uvant måte å jobbe på, men det kom seg ganske fort.

Vi var veldig heldige med prosjektrommet vi fikk og ikke minst veilederen vårs. Prosjektrommet var stort og fint slik at vi fikk god plass til å bruke kinecten og ellers gode arbeidsplasser. Karoline har gjort en fantastisk jobb som veileder, hun har alltid vært tilgjengelig og behjelpelig. De ukentlige møtene har vært noe jeg så frem til da vi hadde en god tone og vi fikk nyttige tilbakemeldinger på det vi hadde fått gjort.

Det å ha vært med på hele prosessen fra vi tenkte ut og planla produktet, deretter begynne å jobbe på forskjellige deler av det og se hvordan det vokste seg større og begynte å ta form og til slutt når vi satte alt sammen og fikk se det endelige resultater har vært en utrolig kjekk opplevelse. sluttprodukt vårt er jeg veldig fornøyd med, selvfølgelig er det veldig mye mer som kan legges til spillet, men med tiden vi hadde til rådighet er jeg storfornøyd med resultatet.

6.6 Dagfinn Kjærnet

Min rolle på prosjektgruppen har vært prosjektleder og produkteier. Produkteier vil si at jeg er gruppen og kundens(Sunnaas sykehus) mellomledd, og har som hovedoppgave å tolke de krav og ønsker som stilles til produktet av kunden og formidle dem til gruppen. Som prosjektleder har min rolle vært å holde kontakten med Sunnaas og ekstern sensor og veileder, og å passe på at alle til enhver tid har noe å jobbe med og ikke blir stående fast med noe. Jeg har også hatt som ansvar å passe på at alle gjør tilnærmet like mye arbeid, men dette har i stor grad løst seg da alle har hvert flinke til å trekke til seg passende oppgaver og jobbet hardt gjennom hele prosjektet.

Jeg syntes faget er lagt opp på en interessant måte og liker konseptet med at man selv skal kontakte bedrifter og få tak i en oppgave. På denne måten får vi et lite innblikk i industriens hverdag og får et direkte samarbeid med ressurspersoner med erfaring fra næringslivet.

Vi var svært heldige med oppgaven vår og fikk i stor grad forme den selv. Denne prosessen har vært veldig spennende. Jeg syntes også det har vært veldig lærerikt å være en del av et utviklingsteam. Jeg er veldig fornøyd med både oppdragsgiver, veiledere og sensorer da de har vist stor entusiasme og interesse for prosjektet. Egil som har vært vår eksternveileder har blant annet invitert oss på et møte i Oslo som handlet om spill i helsevesenet som noen av våre gruppemedlemmer deltok på.

Gjennom prosjektet har vi fått bruk for mye av det vi har lært så langt, blant annet har jeg brukt mye av det vi lærte i database faget når jeg lagde dbHandler klassen. Jeg har også lært mye nytt og ikke minst utviklet meg mye som person. Mye takket være vår internveileder Karoline har jeg blant annet blitt mer selvsikker og betraktelig bedre til å holde presentasjoner. Jeg føler også at vi har dratt god nytte av hennes kunnskaper om prosjektstyring og har gjennom hele prosjektet følt at vi har blitt godt ivaretatt.

Jeg syntes gruppen har jobbet godt sammen og vi har stort sett vært enige om hvilke mål vi ønsket å sette for både gruppen og produktet. Selv om vi ikke nådde alle våre mål med produktet føler jeg vi har gjort en god jobb og laget noe vi kan være stolte av. Jeg skulle forøvrig ønske vi fikk utnyttet spillgruppen på Sunnaas litt mer, men dessverre tok det lenger tid enn vi hadde håpet å få en spillbar prototype.

7. Kostnader

Vårt prosjekt har vært ett rent software prosjekt med unntak av kinect. Derfor har vi heller ikke hatt store utgifter når det kommer til innkjøp av komponenter og slik. De fleste utgiftene våre går på innkjøpa av rekvisita og møte/presentasjons godt. Det eneste vi har kjøpt av lisenser/tjenester er tjenesten OnTime som vi bruker opp mot prosjektmodellen Scrum. For å se budsjett se eget dokument.

8. Lærdommer

8.1 Musikkanalysealgoritmer

Vi ønsket i utgangspunktet å ha full støtte for egen musikk i spillet vårt, planen var å analysere lyddataene og benytte algoritmer for å gjenkjenne rytmen i sangen og auto generere en passende dans. Det er tidligere gjort mye forskning på rytmegjenkjenning vi håpet å finne biblioteker som kunne hjelpe oss godt i gang.

Oppgaven virker ved første øyekast triviell; "finn alle 'beats' i en sang", noe som faller nærmest naturlig for de fleste. I praksis har det for øvrig vist seg å være svært problematisk, mye fordi en algoritme som passer for alle musikksjangre vil være svært kompleks. Videre bør man ha kjennskap til signalbehandling, noe dessverre ingen av våre gruppemedlemmer har. Vi har derfor prøvd å finne så mye tilgjengelig informasjon om det som mulig å jobbe ut ifra dette.

Det første vi gjorde var å lage en klasse som kun fikk i oppgave å analysere dataene vi kunne hente ut fra musikkfilen. Deretter lagde vi noen enkle utkast til hvordan vi kunne lokalisere "topp punkter" i sangene, på denne måten følte vi at vi fikk et litt bedre innblikk i problematikken ved rytmedeteksjon. Etter å ha prøvd oss litt frem prøvde vi å finne ett godt bibliotek som kunne gjøre denne analysen for oss og gi oss de dataene vi trengte, som for vår del var en liste med timestamps for hvor i sangen de forskjellige "beatene" befant seg.

Det var mye tilgjengelig forskning på dette[1], men lite konkret kode. Et av de bedre alternativene vi fant er et program som heter BeatRoot[2], dette er skrevet i java og sies å gi svært gode resultater for de fleste musikksjangere. Vi hadde for øvrig dessverre ikke tid til å skrive om denne koden til C# og lette videre etter en litt enklere algoritme, men vi tror dette vil være et godt sted å begynne i en eventuell videreutvikling (se eget videreutviklingsdokument).

Vi fant omsider en relativt kort og grei kode vi tok utgangspunkt i, den er skrevet av forumbrukeren Quincy[3] og er inspirert av en artikkel skrevet av Frédéric Patin[4]. Vi har implementert og testet den en del, den er godt stykke fra å være perfekt men den ser ut til å fungere greit på endel sanger og er derfor tilstrekkelig til vårt bruk.

9. Teknologier

9.1 NGUI: Next-Gen UI kit

Vi er godt fornøyde med hvordan det har vært å jobbe med NGUI. Da vi i stor grad har kunne forandre layouten på spillet slik vi ville. NGUI gjorde jobben vår med å lage knapper, inputfelt, avkrysningsbokser, dynamiske lister og andre GUI elementer på en slik måte vi ønsket det.

9.2 Google Sites

Vi er fornøyde med hvordan det var å jobbe med Google sites. Det å sette opp Google Site siden gikk greit. Men da skolen krever at hjemmesiden skal ligge på skolen sine servere, var vi nødt til å flytte den over og det vi hadde gjort med Google Site virket unødvendig.

9.3 Doxygen

Vi har brukt Doxygen til å dokumentere kildekode. Doxygen har vært et verktøy vi har hatt godt nytte av og er fornøyde med. Dokumentasjon av kildekode uten et verktøy som Doxygen ville gjort det mye tyngre for oss i implementeringsfasen. Vi tror også at de som skal se på dokumentasjon av kildekoden også setter pris på Doxygen, siden leseren vil få en pent og oversiktelig utskrift i blant annet PDF format.

9.4 ZigFu

Zigfu er et rammeverk som brukes for å koble kinecten opp mot Unity, og utvikle applikasjoner som innebærer motion-tracking. Det var svært enkelt å implementere Zigfu i Unity. Sammen med rammeverket, fulgte det med flere eksempler på hvordan Zigfu brukes i Unity. Dette var et godt valg

for oss, siden vi kunne bruke mer tid på å utvikle applikasjonen, og mindre tid på å sette oss inn i oppbyggningen av motion-tracking i Unity.

9.5 Unity3D

Vi er fornøyde med hvordan det har hvert å jobbe i Unity. Vi har i stor grad kunnet jobbe med de separate komponentene hver for oss uten å møte på store problemer når vi skulle sette dem sammen igjen. Vi har dessuten hatt stor nytte av det store nettsamfunnet rundt Unity. Blant annet har vi selv stilt spørsmål[5] og fått svar som har hjulpet oss å lokalisere og løse problemer vi har kommet over. Ofte har det allerede vært noen som har hatt tilsvarende spørsmål så da har vi funnet løsninger i svarene på disse. Til vårt bruk har Unity3D tilfredsstilt alle våre behov.

9.6 C Sharp (C#)

Vi valgte å benytte C# i utviklingen av vårt produkt. Vi er fornøyde med valget og har ikke møtt på noen store språkrelaterte utfordringer.

9.7 SQLite

For å lagre data har vi valgt å bruke SQLite. Dette har til tider vært litt tungvint da syntaksen er noe tyngre og litt mer primitiv enn f.eks mySQL. MySQL ser dessuten ut til å ha flere tilgjengelige guider og tredjepartshjelpeverktøyer enn det vi har funnet til SQLite, noe som gjør design- og feilsøkingsprosessen en del lettere. Ulempen med mySQL kontra SQLite er at man til mySQL må ha en egen uavhengig prosess som kjører databaseserveren, mens SQLite direkte integrerer dette som en del av spillet. Vi mener derfor at til selve utviklingsprosessen ville det ha lønt seg å bruke mySQL, men at SQLite eller en tilsvarende løsning burde blitt implementert senere i utviklingen.

9.8 Microsoft Kinect

Siden vi ønsket at brukeren selv skulle være kontrollen trengte vi en form for motiontracking teknologi. Kinect har fungert veldig bra til vårt bruk, selv om den gjerne skulle vært litt mer presis. Kinect 2 er for øvrig på trappene og er snart tilgjengelig på det offentlige markedet. Den sies å være vesentlig mer presis enn Kinect 1 og vil nok derfor egne seg enda bedre til et slikt prosjekt der man helst vil kunne vurdere utførelse helt ned til den minste detalj. Siden Kinect 2 ikke har hvert tilgjengelig for oss føler vi at vi har gjort et klokt valg ved å bruke Kinect 1 da denne har fungert tilstrekkelig. Forhåpentligvis vil det også la seg gjøre å oppgradere spillet til å støtte Kinect 2 senere, uten å måtte gjøre store endringer.

9.9 Git

Vi brukte Git i starten av implementasjonsfasen, men datt litt av og endte med å bruke dropbox isteden da vi syntes det var relativt tungt å sette seg inn i Git. Det har tidvis vært litt rotete å jobbe med utvikling på denne måten, og vi ser definitivt verdien av å bruke verktøyer for versjonskontroll. Siden vi i stor grad har jobbet med forskjellige "moduler/komponenter" av spillet har det heldigvis gått greit selv om vi kun brukte dropbox. Unity gjør det relativt lett å sette sammen enkeltdelene våre til et helt prosjekt, men vi kunne sikkert kunne spart oss noe tid på dette dersom vi hadde hvert flinkere til å bruke Git.

Vi har i ettertid fått høre at andre versjonskontrollverktøyer ville vært lettere å sette seg inn i, blant annet har vi blitt anbefalt subversion[6]. Vi burde nok derfor ha brukt mer tid på å utforske alternativene når vi gjorde teknologivurderingen.

9.10 Dropbox

Alle på gruppen har mye erfaring med bruken av dropbox fra før som gjør det veldig enkelt for oss å ta teknologien i bruk. Dropbox tar automatisk backup av det som blir lagret, i tillegg tar vi ofte backup manuelt. Siden dropbox lagrer all dataen i en nettsky og automatisk laster ned alle filene som ligger der til alle gruppemedlemmene som gjør det veldig lett å kontrollese dokumenter og sjekke at ting stemmer overens med hverandre.

Dropbox er veldig lett å bruke og jobbe opp mot, dette har gjort det sømløst å jobbe sammen med mange dokumenter, det har vært ganger brukere har lagree over hverandre som gjør at vi får litt ekstra arbeid med å sette ting korrekt sammen, men dette har kun skjedd noen få ganger. Fordelene vi har hatt med dropbox er mye større enn de få ulempene vi har opplevd.

10. Takk Til

Karoline Moholth har vært med oss gjennom hele prosjektet som intern veileder. Hun har vært behjelpelig fra vi begynte til der vi er nå, produktet hadde ikke blitt det samme uten henne.

Sunnaas Sykehus ga oss oppgaven vår og de har vært veldig engasjerte i det vi har gjort og alltid stilt seg til disposisjon hvis det var noe vi trengte hjelp til.

Egil Utheim for en fremragende jobb som ekstern veileder. Han har vært veldig engasjert i oppgaven vår og gitt oss gode tilbakemeldinger gjennom hele prosjektet.

Bjørn Sundby har vært vår eksterne sensor.

Key-Shuh for å ha laget en all lyden i spillet vårt. De gjorde en god jobb med lyden og vi fikk mye bra som vi kunne bruke og har brukt i produktet vårt.

Christoffer Karlsen har laget logoen til LetsDance som har blitt kjennetegnet på produktet vårt. Vi er veldig fornøyd hvordan den ser ut og representerer produktet vårt.

Sondre Adler Guriby har laget plakaten vår som skal henge i gangen på HiBu som en oppsummering av gruppen og hva vi har produsert i løpet av prosjekt tiden.

HiBu som har gitt oss muligheten til å jobbe med slikt spennende prosjekt

Hallstein A. Hansen som har vært vår sensor.

11. Referanser

1. Mario@www.badlogicgames.com. *Beat Detection – The Scientific View*. 2010 24.05.2013]; Available from: http://www.badlogicgames.com/wordpress/?p=99.
2. Cannam, S.D.C. *BeatRoot*. 24.05.2013]; Available from: http://code.soundsoftware.ac.uk/projects/beatroot.
3. Quincy@stackexchange. *Questions - Beat detection and FFT*. Available from: http://gamedev.stackexchange.com/questions/9761/beat-detection-and-fft.
4. Patin, F. *Beat Detection Algorithms*. 2003 26.05.2013]; Available from: http://www.gamedev.net/page/resources/_/technical/math-and-physics/beat-detection-algorithms-r1952.
5. LetsDance. *OnTriggerEnter doesn't work as it should? (c#)*. 2013 25.05.2013]; Available from: http://answers.unity3d.com/questions/452303/ontriggerenter-doesnt-work-as-it-should-c.html.
6. Apache. *Apache™ Subversion®*. 25.05.2013]; Available from: http://subversion.apache.org/.

Prosjektplan

HØGSKOLEN
i Buskerud

Mektron

Sunnaas sykehus HF

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Revisjon	Endringer	Dato	Skribent
1.0	Opprettet	Opprettet	11.12.12	JEH
2.0	Oppdatert	Skrevet mye av innholdet i dokumentet	17.12.12 - 21.12.12	JEH
3.0	Oppdatert	Oppdatert dokumentet	03.01.13	JEH
4.0	Oppdatert	Lagt til avsnitt om GDD	28.02.13	JEH
5.0	Oppdatert	Oppdatert med referanser til analyse og designdokument. Oppdatert milepæler	19.03	JEH
6.0	Oppdatert	Oppdatert kravspesifikasjon, presentasjon 3 tid, sprint plan og dokumentliste	20.05.2013	JEH

Kontrollert av:	ZA,DK

Innholdsfortegnelse

1 Innledning

1.1 Innledning

I forbindelse med vår bacheloroppgave ved Høgskolen i Buskerud har vi fått i oppdrag fra Sunnaas Sykehus å lage et spill som skal benyttes som et tilbud i rehabiliteringsfasen til rullestolbrukere og slagpasienter. Til vanlig er ikke dette noe Sunnaas jobber med, så deres kompetanse innenfor det teknologiske er begrenset. Derfor vil prosjektet bli kjørt i samarbeid med Mektron AS. Mektron As er en bedrift som jobber med å utnytte og knytte teknologi sammen med helse[1].

Spillene som allerede eksisterer og blir benyttet i dag, er enten for lette og barnslige eller så er det alt for vanskelig for pasienten. Derfor er Sunnaas på jakt etter et spill som er skreddersydd deres pasientgruppe.

1.2 Om Dokumentet

Hensikten med dette dokumentet er å gi en beskrivelse av planlegning av prosjektet. Dokumentet inneholder også en beskrivelse av hva oppgaven går ut på, hva som skal leveres når, mål og begrensninger, organisering, estimering av hva som blir gjort når og de overordnende målene med prosjektet.

1.3 Skribent(er)

Jan Erik Helskog(JEH), Marcus Jernberg(MJ)

2 Gruppesammensetning

Navn	Dagfinn Kjærnet
Alder	23 år
Ansvarsområder	Prosjektleder og Produkteier
E-Post	Dagfinn.kjaernet@gmail.com
Telefon	452 92 707
Linje	Data – Virtuelle Systemer

Navn	Henrik Olsson
Alder	22år
Ansvarsområder	Scrum-Master og Test
E-Post	h.h.olsson@gmail.com
Telefon	948 40 828
Linje	Data – Virtuelle Systemer

Navn	Jan Erik Helskog
Alder	21 år
Ansvarsområder	Web
E-Post	janerhel@gmail.com
Telefon	452 11 330
Linje	Data – Virtuelle Systemer

Navn	Marcus Jernberg
Alder	22 år
Ansvarsområder	Implementasjon og risiko
E-Post	Marcus.jernberg@outlook.com
Telefon	934 92 063
Linje	Data – Virtuelle Systemer

Navn	Are Oven
Alder	21 år
Ansvarsområder	Budsjett og innkjøp
E-Post	Are.oven@gmail.com
Telefon	97061360
Linje	Data – Virtuelle Systemer

Navn	Zana Ali
Alder	22 år
Ansvarsområder	Dokumenter
E-Post	Zana_a_ali@hotmail.com
Telefon	994 22 335
Linje	Data – Virtuelle Systemer

3 Oppgaven

Vi skal utvikle en spillapplikasjon. Denne spillapplikasjonen skal være et dansespill og bruke Kinect-teknologi. Spillapplikasjonen skal brukes som et rehabiliteringstilbud til pasientene. Viser til dokumentet om hvorfor bruke spill til rehabilitering[2]. Musikken i spillet skal brukeren selv kunne velge. Dette innebærer at brukeren selv kan legge inn ønsket musikk i spillapplikasjonen. Derfor vil en del av oppgaven være å lage algoritmer som gjenkjenner forskjellige rytmer og deretter genererer en dans.

En stor del av oppgaven vil bestå av å lokalisere krav hos brukerne, dette gjør vi ved å innhente informasjon hos et brukerpanel på Sunnaas.

Spillapplikasjonen skal være veldig enkel å bruke. Systemet skal være enkelt å sette opp og man skal komme raskt i gang.

3.1 Krav til spillapplikasjonen

I forbindelse med utvikling av LetsDance for Sunnaas sykehus er det noen overordnede krav som stilles til hva spillet skal inneholde. Disse listes opp her som krav, men siden vi bruker SCRUM som prosjektmodell vil alle krav komme som user stories i product backloggen.

- Behandlings effekt
- Målgruppe:
  - Slag og rullestolbrukere, lamme på en side, skal kunne tilpasses ut i fra pasientens skade/handicap
- Statistikk og belønningssystem er svært viktig
  - Gjøre det morsomt!
  - Jublende publikum, lyd, score, achievements.
  - Skal kunne se progresjon
- Konkurranseelement
  - Måle seg opp mot andre og seg selv
- Justerbar vanskelighetsgrad
  - Tempo
  - Presisjon, mindre hjelpefunksjoner
- Legge inn egen musikk
- Sensor fanger opp om det er rullestolbruker, og tilpasser seg deretter.
  - Sende en bekreftelses melding på skjermen
  - Begrenser dansebevegelsene til for eksempel kun armer og overkropp
- Profiler
  - For å enkelt kunne se på tidligere resultater og dermed se progresjon.
  - Skal lagres i en database

- Enkelt
    - Skal være enkelt og raskt å komme i gang

Som nevnt over er dette en liste med overordnede krav. Derfor har vi valt å lage en kravspesifikasjon[3], selv om vi benytter oss av prosjektmodellen SCRUM. Dette har vi valgt å gjøre for å få en god oversikt over kravene vi har definert. Kravene har vi definert ut ifra listen med overordnede krav og våre observasjoner ved spillsesjoner. I dette dokumentet vil det være en liste med krav rangert fra A-C. Der A er de viktigste og C er de mindre viktige.

3.2 Krav til dokumentasjon

Høgskolen har en del krav til dokumentasjonen som skal produseres i løpet av prosjektperioden. På grunn av prosjektmodellen SCRUM, som gruppen har valgt å bruke vil navnet på disse dokumentene avvike fra hva man vanligvis ser. De obligatoriske dokumentene vil allikevel bli dekket selv om vi bruker SCRUM.

Kravspesifikasjon er et dokument som kreves av Høgskolen. Dette dokumentet skal inneholde alle krav med tilhørende oversikt over testspesifikasjon. På grunn av at vi bruker SCRUM vil dette dokumentet se litt annerledes ut. I SCRUM blir alle krav definert som user stories i product backlogen. Alle user stories vil også ha en tilhørende test case. Denne testen skal fortelle om det som har blitt gjort tilfredsstiller behovet til brukeren.

Product backlogen som blir produsert i SCRUM er et meget dynamisk dokument. Derfor vil det være viktig å kunne spore endringer som blir gjort i produkt backlogen igjennom prosjektet.

Nedenfor er det en oversikt over planlagte dokumenter og til hvilke presentasjon de eventuelt skal leveres.

Dokument	Klar til presentasjon #
Prosjektplan	1,2,3
Dokumentstandard	1
Product backlog	1,2,3
Teknologidokumenter	1,2,3
Test Case	1,2,3
Møteinnkallelser	1,2,3
Møtereferater	1,2,3
Scrum dokument	1
Kontrakt	1
Kodestandard	1
Sprint debrief	2,3
Risikovurdering	1,2,3
Budsjett(vedlagt kontrakten)	1,3
Timelister	1,2,3
Endelig prosjektplan	3
Designdokument	2,3
Analysedokument	2,3
Sluttrapport	3
Videreutviklingsdokument	3

Doxygen dokumentasjon	2,3
Fremdriftsplan	2,3
Sprint debriefer	2,3
Gamedesign dokument	2,3

3.3 Observerende krav

Prosjektgruppen er invitert til å observere et brukerpanel på Sunnaas sykehus. Brukerpanelet vil bestå av pasienter ved Sunnaas sykehus, i tillegg til en gruppe ergoterapeuter. Dette vil fungere som ukentlige møter for å kartlegge kravene til brukerne. I motsetning til de overordnede vil disse kravene være mer rettet mot hver enkelt bruker.

Vi vil utvikle produktet i samarbeid med brukerpanelet, og når en prototype er klar vil vi også teste produktet i samarbeid med panelet. Testene vil ikke være like hyppige som de ukentlige spillsesjonene, men etter hvert som vi har noen nye endringer som må testes.

Prosjektgruppen vil i starten ta del på disse spillsesjonene, men etter hvert ut i prosjektet flytte fokuset over på utvikling av spillapplikasjonen.

4 Mål

Bakgrunnen for prosjektet er at vi som gruppe skal tilegne oss kunnskap om og vise ferdigheter i grunnleggende prosjektarbeid. Det vil si samarbeid, planlegge, dokumentere, organisere, utvikle og teste prosjektet. Vi skal også utvise følgende:

- Lederskap og benytte relevante tilgjengelige verktøy
- Besluttsomhet knyttet til valg av teknologi
- Erfaringer med bruk av SCRUM som prosjektmodell
- Kunde kommunikasjons erfaring
- Bidra til verdiskapning for vår kunde igjennom et godt produkt
- Bidra til forskning og forsøk innen spill i helsetjenesten

5 Fremdriftsplan

5.1 Sprinter

Dette er en oversikt over de planlagte sprintene og hva som skal bli gjort løpet av sprinten. Sprintene vil ha forskjellig lengde, selv om de i utgangspunktet skal være like lange. Dette grunner i antall dager som er satt av til prosjektet. I høstsemesteret vil en sprint vare i åtte uker, da det kun er satt av en dag til arbeid med prosjekt. I vårsemesteret vil sprintene før påske vare i tre uker, da det er satt av tre dager i uken til prosjekt. Etter påske vil alle sprintene vare i to uker, da det er satt av fem dager i uken.

Sprint #	Dato(Fra og til)	Hovedmål med sprint	Estimert antall timer
1 – Forberedelsesfase	01.11.12 - 08.01.13	Får på plass dokumentasjon og gjøre klar presentasjon 1	300
2 – Innsamling sprint	09.01.13 – 01.02.13	Spillsesjoner ved Sunnaas for å samle inn krav til	300

		spillapplikasjonen, ved å observere og analysere tilbakemeldinger fra brukerpanelet og sette opp user stories, samt estimere tid.	
3 – Design av arkitektur og struktur	04.02.13 – 22.02.13	Lage design av arkitektur og struktur til spillapplikasjonen samt databasen. Gjøres ved hjelp av UML	300
4 – Implantasjon av arkitektur og struktur	25.02.13 - 15.03.13	Implementasjon av arkitekturen og databasen. Jobbe med en enkel prototype	300
5 – Presentasjon 2	18.03.13 - 22.03.13	Lage og holde 2. presentasjon. Ferdigstille design og dokumenter som skal leveres.	300
6 – Implementasjon del1	01.04.13 – 12.04.13	Implementasjon av spillet. Fokusere på de viktigste og vanskeligste kravene til spill applikasjonen. Få det spillbart	300
7 – Implementasjon del2	15.04.13 – 26.04.13	Videre implementasjon av spillapplikasjonen.	300
8 – Utredningsfasen	29.04.2013 - 16.05.2013	Endringer etter bruker tilbakemeldinger. Legge til ny funksjoner hvis mulig.	300
9 - Ferdigstilling	13.05.2013 - 27.05.2013	Lage ferdig dokumentasjon for innlevering. Samt klargjøre applikasjonen for leveranse	300
Sum antall timer i sprinter			2700

5.2 Presentasjoner

Presentasjon	Dato	Tid	Beskrivelse
1	08.01.2013	14.30	Vise hva vi skal gjøre, når vi skal gjøre det og hvordan vi skal løse oppgaven.
2	22.03.2013	12.00	Hva har vi gjort, hva skal vi gjøre videre og en generell oppdatering på fremdrift.
3	03.06.2013	12.00	Vise frem produktet og fortelle om de tekniske løsningene vi har kommet frem til.

6 Milepæler

I løpet av prosjektet har vi valgt å sette oss noen milepæler. Dette er viktige mål i utviklingsprosessen til spillapplikasjonen.

6.1 Prototype

Prosjektet vil benytte seg av prototyper for å få tilbakemeldinger underveis i utviklingsfasen. Den første prototypen vil derfor være en milepæl.

Prototypen er på plass og er klar til å vises frem på andre presentasjon. Denne vil vi også ta med oss til Sunnaas for tilbakemeldinger. Dette er noe vi som gruppe har sett frem til.

6.2 Sprint #2

Dette vil være den første ordentlige sprinten i prosjektet hvor det ikke bare er dokumentasjon og er av den grunn noe gruppen ser frem til.

Sprinten er fullført og gruppen synes det var godt å komme skikkelig i gang med prosjektet.

6.3 Presentasjon 1

Den første presentasjonen i prosjektperioden og er derfor en milepæl gruppen ser frem til.

Presentasjonen gikk over all forventning og tilbakemeldingene var gode. Det ble påpekt noen mangler med tanke på dokumentasjonen, men det var bare små plukk og ble tatt håndom med en eneste gang. Vi fikk også en meget fin artikkel skrevet om prosjektet vårt i Laagendalsposten.

6.4 Presentasjon 2

Den andre presentasjonen i prosjektperioden

Presentasjonen gikk meget bra og bare positive tilbakemeldinger. Hele prosjektgruppen har jobbet over all forventing uken før presentasjonen. Så en meget vellykket milepæl for oss.

6.5 Presentasjon 3

Den aller siste presentasjonen og spillapplikasjonen skal være ferdig. Dette er derfor en naturlig milepæl å se frem til.

6.6 Dokumentasjon ferdig

Det blir mye dokumentasjon i løpet av prosjektperioden og det vil derfor være en stor milepæl når all dokumentasjonen er ferdig slik at det er klart til levering.

7 Utviklingsfaser

Prosjektet består av tre faser, analyse, design og implementasjon.

7.1 Analyse

I denne fasen av prosjektet vil det utredes hva som skal gjøres og hvorfor. På grunn av at prosjektet kun har noen få overordnede krav fra oppdragsgiver vil vi i denne fasen bruke mye tid på å definere krav til spillapplikasjonen. Som det er nevnt tidligere i dokumentet vil kravene bli utviklet ved at gruppen er med på spillsesjoner ved Sunnaas Sykehus. Det er viktig at kravene tilfredsstiller brukerens behov.

I løpet av denne fasen skal det også redegjøres for hvilke teknologier og utviklingsmetoder som skal bli brukt i prosjektet. Dette skal dokumenteres og begrunnes.

Denne fasen vil pågå frem til sprint 2 er avsluttet 01.02.13.
For en gjennomgang av hva som er blitt gjort i denne fasen, se egent analysedokument[4]

7.2 Design

I denne fasen av prosjektet vil gruppen arbeide med den overordnede strukturen til spillapplikasjonen. Gruppen vil her lage UML diagrammer med hjelp av programvaren Visual Paradigm. Dette vil danne grunnlaget for en moderne oppbygning av spillapplikasjonen og vil hjelpe oss videre i implementasjonsfasen.

Det er viktig å merke seg at vi vil i prosjektet bruke Unity og Visual Paradigm har ingen direkte tilknytning til denne spillmotoren. Det at vi bruker Unity vil si at vi kommer til å scripte, altså ikke standard kodeskriving. Det vil derfor være viktig at UML diagrammene blir oversiktlige.

Denne fasen av prosjektet vil pågå fra analysefase slutt. Når designfasen begynner å nærme seg slutten vil materiell som har kommet frem vil bli presentert på Presentasjon 2.
For en gjennomgang av hva som er gjort i design fasen av prosjektet, se egent designdokument[5]

7.3 Implementasjon

Implementasjonsfasen består av programmering. Utvikling og testing av kode vil stå sentralt i denne fasen. Det er spesielt viktig å oppdatere dokumentasjonen som har blitt produsert i tidligere faser av prosjektet dersom planer eller design forandres.

8 Prosjektstyring

8.1 Prosjektmodell

8.1.1 SCRUM

Som prosjektmodell har gruppen bestemt seg for å bruke SCRUM. Valget falt på SCRUM fordi det virker som en spennende og utfordrende prosjektmodell som passer vårt prosjekt godt. Siden vi ser på oss selv som unge, spreke og selvstendige så vil SCRUM være det som passer gruppen best. Vi har tro på at alle individene i utviklerteamet vil ta på seg like omfattende oppgaver, og vil gjøre en god nok dokumentert jobb slik at alt kan Interfaces.

SCRUM er dokumentert nærmere i dokumentet for Utviklingsmetodikk [6].

8.2 Prosjektstyringsverktøy

Som gruppe har vi sett på flere mulige verktøy som kan benyttes til å gi oss en lettere jobb med prosjektstyring. En av de vanligste fallgruvene for SCRUM er at man mister oversikt over hvem som gjør hva. For å unngå dette vil vi trenge et verktøy som gir en enkel og god oversikt over alle user stories. Som gruppe synes vi også at det vil være viktig å kunne dokumentere hva alle har arbeidet og bidratt med til prosjektet på en lett måte.
Som et verktøy har vi bestemt oss for å benytte oss av OnTime.

8.2.1 OnTime

OnTime er utviklet av Axosoft og er et web basert og har et brukergrensesnitt som forenkler SCRUM hverdagen. Alle medlemmene av utviklerteamet har en egen bruker. Dette gir alle muligheten til å ta

på seg user stories fra backlogen. Dersom en user story blir tatt vil statusen endres. OnTime gir oss også muligheten til å skrive ut burndown chart. OnTime er beskrevet i Utviklingsmetodikk[6] dokumentet.

8.3 Game Design Dokument

Game Design Dokumentet[7] er det dokumentet som beskriver hvordan forskjellige elementer i applikasjonen skal være. Dette dokumentet beskriver alt fra hvordan skal gjøres til hvilke ting som skal være med i applikasjonen og hvordan de skal se ut. Dette dokumentet blir skrevet i analyse fasen av prosjektet og vil bli kontinuerlig oppdatert når det blir fattet beslutninger, som påvirker spillet. Kort oppsummert er game design dokumentet en skriftlig beskrivelse av spillapplikasjonen og de tilhørende funksjonene som ligger til grunn.

8.4 Utviklingsmiljø

8.4.1 Spillmotor

Som spillmotor har vi valgt å benytte oss av Unity3D. Dette er et utviklingsverktøy som passer perfekt til vårt formål. Unity3D har tredjeparts støtte for Kinect noe vi er avhengig av.

Unity3D er beskrevet nærmere i et eget teknologidokument[8]. Der står det også mer om hvorfor vi valgte Unity3D.

8.4.2 Revisjonskontroll

På slutten av prosjektet skal vi ha en ferdig applikasjon. Det vil derfor være nødvendig at vi benytter oss av et verktøy for revisjonskontrollering. Ved å bruke dette vil vi enkelt kunne spore endringer som har blitt gjort, samt ha muligheten til å gå tilbake til fungerende versjoner dersom noe skulle gå galt. På grunn av at Unity3D ikke har noe form for revisjonskontroll har vi valgt å bruke Git via Github[9].

8.4.3 UML verktøy

For å få en god struktur på den ferdige applikasjonen er vi nødt til å ha et godt verktøy for UML. Dette vil ikke bare hjelpe på den ferdige applikasjonen, men også i implementasjonsfasen av prosjektet. Til dette har gruppen sagt seg enige i å bruke Visual Paradigm[10]. Dette er et godt verktøy som er kontinuerlig oppdatert. Visual Paradigm er enkelt og oversiktlig å bruke. Derfor har gruppen valgt å bruke Visual Paradigm

8.5 Testspesifikasjon

Som krav fra Høgskolen må vi i prosjektet utføre testing på flere nivåer. Metodene og hvordan vi skal teste er beskrevet i testspesifikasjon[11]dokumentet.

8.6 Andre dokumenter

8.6.1 Kodestandard

Kodestandard[12] er et dokument som definerer hvordan vi som utviklingsgruppe skal skrive kildekoden vi kommer til å produsere. Dette dokumentet er noe vi selv må definere etter våre behov. Kodestandard dokumentet må være på plass før programmeringen kan starte.

8.6.2 Budsjett

Budsjettet[13] er for å enkelt kunne holde oversikt over hvilke utgifter vi har i prosjektet. Dette inkluderer de planlagte utgiftene, som innkjøp av utstyr. Samtidig skal de påløpte uforutsette utgiftene føres opp. Budsjettet ligger som et vedlegg til kontrakten.

8.6.3 Timelister

Timelistene beskriver antall timer vi har brukt på forskjellige aktiviteter og hvilke dato det ble gjort. OnTime vil gjøre det enkelt for oss å knytte timefordeling mot en aktivitet.

9 Risiko

Når man arbeider med prosjektarbeid i gruppe er det nødvendig å ta høyde for at ting ikke alltid går helt etter planen. I de fleste prosjektmodeller er det vanlig å vurdere hvilke risikoer som kan påvirke prosjektet, og deretter sette opp en plan for hvordan man skal håndtere hvert av disse punktene. Siden gruppen jobber med SCRUM som prosjektmodell, bruker vi mindre tid og ressurser på planlegging av risiko. Filosofien bak SCRUM er at man setter opp sprinter med bakgrunn i en viss hastighet og gjennomføringsevne. Dersom det dukker opp uforutsette hendelser må gruppen tilpasse sprinter og forventet hastighet, dette er relativt enkelt å gjøre. Det gjør altså at vi kan bruke mindre ressurser på å planlegge for eventuelle uforutsette hendelser og fokusere på gjennomføring av prosjektet. Så fort gruppen oppdager potensielle problemer skal dette lokaliseres og dokumenteres i risikoanalyse dokumentet [14].

10 Referanser

1. ***Mektron.*** **2012.**
2. ***Hvorfor bruke spill til rehabilitering.*** **2012.**
3. ***Kravspesifikasjon.*** **2013.**
4. ***Analysedokument.*** **2013.**
5. ***Designdokument.*** **2013.**
6. ***Utviklingsmetodikk.*** **2012.**
7. ***Game Design Document.*** **2012.**
8. ***Unity.*** **2012.**
9. ***Git.*** **2012.**
10. ***UnifiedModellingLanguage.*** **2012.**
11. ***Testing.*** **2012.**
12. ***Kodestandard.*** **2012.**
13. ***Budsjett.*** **2012.**
14. ***Risikovurdering.*** **2012.**

Sprintplan for sprint

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
79	UML: Danse	Medium	0	Rejected	8 hrs	0	0
115	Funksjon: Meny - laste bruker liste	Medium	0	Rejected	20 hrs	0	0
168	Funksjon: Dans - Algoritme for "ikke optimal utførelse"	Medium	0	Rejected	6 hrs	0	0
177	Funksjon: Musikk - musikkgrensesnitt databehandling	Medium	0	Rejected	2 hrs	0 hrs	0
1	Sette Kodestandard	High	Marcus Jernberg	Completed	1 hrs	0 hrs	1 hrs
2	User story template	High	Henrik Olsson	Completed	4 hrs	0 hrs	4 hrs
3	Sette opp hjemmeside	High	Jan Erik Helskog	Completed	6 hrs	0 hrs	8 hrs
5	Standard forside	High	Zana Ali	Completed	1 hrs	0 hrs	1 hrs
6	Dokument; standard formatering	High	Zana Ali	Completed	1 hrs	0 hrs	1 hrs
7	Teknologidokument: Googlesites	High	Jan Erik Helskog	Completed	8 hrs	0 hrs	7 hrs
8	Møtereferat [10.12.12]	High	Dagfinn Kjærnet	Completed	1 hrs	0 hrs	1 hrs
9	Risiko	High	Marcus Jernberg	Completed	1 hrs	0 hrs	1 hrs
11	Prosjektplan	High	Jan Erik Helskog	Completed	35 hrs	0 hrs	30 hrs
12	Teknologidokument: RUP utviklingsmodell	High	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	4 hrs
13	Møtereferat [27.11.12]	High	Jan Erik Helskog	Completed	2 hrs	0 hrs	2 hrs
14	Skrive om Kinect	High	Zana Ali	Completed	1 hrs	0 hrs	1.5 hrs
15	Teknologidokument: Dropbox	High	Marcus Jernberg	Completed	1 hrs	0 hrs	1 hrs
16	Dokumentmaler	High	Zana Ali	Completed	8 hrs	0 hrs	10 hrs
17	Kort dokumentasjon av C# og Unity	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
18	Møtereferat [06.11.12]	High	Marcus Jernberg	Completed	1 hrs	0 hrs	1 hrs
19	Møtereferat [23.11.12]	High	Zana Ali	Completed	30 min	0 min	30 min
20	Møteinnkallelse [28.11.12]	High	Zana Ali	Completed	30 min	0 min	30 min
21	Hvorfor vi lager spillet	High	Marcus Jernberg	Completed	2 hrs	0 hrs	1 hrs
22	Lage fem eksempler på userstories	High	Dagfinn Kjærnet	Completed	1 hrs	0 hrs	1 hrs
23	Teknologidokument: C#	High	Dagfinn Kjærnet	Completed	3 hrs	0 hrs	3 hrs
24	Teknologidokument: Unity	High	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	5 hrs
25	Klargjøring av Kontrakt: Endringer, vedlegg m.m.	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
26	Utviklingsmetodikk dokument	High	Henrik Olsson	Completed	8 hrs	0 min	8 hrs
27	Møtereferat [02.11.12]	High	Henrik Olsson	Completed	2 hrs	2	2 hrs
28	Krav	High	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
29	Kontaktinformasjon	Medium	Marcus Jernberg	Completed	1.5 hrs	0 hrs	1.5 hrs
30	Teknologidokument: OnTimeNow	High	Henrik Olsson	Completed	3 hrs	0 hrs	3 hrs
31	TestDokument: Hvorfor vi tester, V&V	Medium	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
32	Skrive om Mektron	High	Zana Ali	Completed	30 min	0 min	30 min
33	Skrive om Prototype, og testing opp mot brukere	High	Zana Ali	Completed	1 hrs	0 hrs	4 hrs
34	Møtereferat 18.12.12	Medium	Marcus Jernberg	Completed	0.5 hrs	0 hrs	0.5 hrs

35	Black-Box & White-Box	Medium	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
36	Test Case	Medium	Marcus Jernberg	Completed	1 hrs	0 hrs	2 hrs
38	spill til rehabilitering	High	Are Oven	Completed	5 hrs	0 hrs	4 hrs
39	Teknologidokument: Unity del 2, Kinect og unity	Medium	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	4 hrs
40	informasjon fremgangsmåte	High	Are Oven	Completed	3 hrs	0 hrs	3 hrs
41	Teknologidokument: Git	High	Are Oven	Completed	5 hrs	0 hrs	4 hrs
42	Test Typer	Medium	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
43	Teknologidokument: C# del 2 om SQLite	Medium	Dagfinn Kjærnet	Completed	1 hrs	0 hrs	1 hrs
44	Skrive om Doxygen	High	Zana Ali	Completed	2 hrs	0 hrs	2 hrs
48	Brukerstatistikk	High	0	Completed	0	0	0
49	Brukerprestasjoner	Low	0	Completed	0	0	0
51	SpillTempo	Medium	0	Completed	0	0	0
53	Lydimport	Low	0	Completed	0	0	0
54	Quick entry	Low	0	Completed	0	0	0
55	Profilering	High	0	Completed	0	0	0
56	Brukergrensesnitt	Medium	0	Completed	0	0	0
57	Møteinnkallelse [18.12.2012]	High	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
58	Debrief doc: Sprint 1	High	Henrik Olsson	Completed	4 hrs	0 hrs	1 hrs
59	GDD: Teknologi	High	Marcus Jernberg	Completed	4 hrs	0 hrs	4 hrs
60	GDD: Dansebevegelser	High	Zana Ali	Completed	6 hrs	0 hrs	8 hrs
61	GDD: Statistikk	High	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	5 hrs
62	GDD: Prestasjoner	High	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	5 hrs
63	GDD: Interface	High	Henrik Olsson	Completed	8 hrs	0 hrs	6 hrs
64	GDD: Innledning	High	Jan Erik Helskog	Completed	3 hrs	0 hrs	3 hrs
65	GDD: GamePlay	High	Jan Erik Helskog	Completed	8 hrs	2 hrs	6 hrs
66	GDD: Dynamisk Vanskelighetsgrad	High	Are Oven	Completed	6 hrs	0	5.5 hrs
67	Sprint-Plan del 1	High	Henrik Olsson	Completed	2 hrs	0 hrs	3 hrs
69	GDD: FAQ	High	Marcus Jernberg	Completed	1.5 hrs	0 min	45 min
70	GDD: Lyd	High	Are Oven	Completed	5 hrs	0	5 hrs
71	GDD: Sette sammen	High	Jan Erik Helskog	Completed	2 hrs	0 hrs	3 hrs
72	Testdokumentasjon: Brukerskjema	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
73	Skjemadesign: Dagfinn	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
74	Skjemadesign: NAVN	High	Jan Erik Helskog	Completed	2 hrs	0 hrs	2 hrs
75	UML Research	High	Marcus Jernberg	Completed	5 hrs	0 hrs	2 hrs
76	UML: Research	High	Henrik Olsson	Completed	5 hrs	0 hrs	5 hrs
77	UML: Research	High	Zana Ali	Completed	5 hrs	0 hrs	5 hrs
78	UML: research	High	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	5 hrs

80	UML: Tilpasse	High	Marcus Jernberg	Completed	8 hrs	0 hrs	3 hrs
81	UML: Vanskelighetsgrad	High	Are Oven	Completed	8 hrs	0 min	4.5 hrs
82	UML: Musikk	High	Henrik Olsson	Completed	8 hrs	0 hrs	10 hrs
83	UML: Prestasjoner	High	Dagfinn Kjærnet	Completed	8 hrs	0 hrs	6 hrs
84	UML: Konto	High	Zana Ali	Completed	8 hrs	0 hrs	6 hrs
85	UML: Statistikk	High	Jan Erik Helskog	Completed	8 hrs	0 hrs	10 hrs
86	Analysedokument: påbegynt	High	Zana Ali	Completed	2 hrs	0 hrs	2 hrs
87	UML: Nullstille passord	High	Zana Ali	Completed	2 hrs	0 hrs	1 hrs
88	UML: Research	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	5 hrs
89	Sprint 2 Debrief	High	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
90	Friske opp Unity-kunnskapene	High	Marcus Jernberg	Completed	8 hrs	0 hrs	8 hrs
91	Unity: Database testing	High	Dagfinn Kjærnet	Completed	8 hrs	0 hrs	8 hrs
92	Audio Import	High	Henrik Olsson	Completed	20 hrs	0	16 hrs
93	Audio Analyzer	High	0	Completed	40 hrs	0 min	0
94	Akritektur: Database - Kontakt med database	High	Dagfinn Kjærnet	Completed	1 hrs	0 hrs	1 hrs
95	Arkitektur: Generator - Legge til bevegelser	High	Are Oven	Completed	20 hrs	0	18 hrs
96	Akritektur: Database - Lese fra database	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
97	Arkitektur: Generator - Hente dans	High	0	Completed	5 hrs	0	0
98	Akritektur: Database - Skrive til database	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
99	Arkitektur: Dansescene - danserom	High	Marcus Jernberg	Completed	10 hrs	0 hrs	6.5 hrs
100	Audio User Interface	High	0	Completed	20 hrs	0	0
101	Funksjon: Generator - Organisere bevegelser	High	0	Completed	8 hrs	0	1 hrs
102	Funksjon: Dancescene - Lagre Statistikk	Medium	0	Completed	10 hrs	0	1 hrs
103	Arkitektur: Dansescene - Kontakt med kinect	High	Marcus Jernberg	Completed	5 hrs	0 min	2 hrs
104	Funksjon: Generator - Generere dans	High	Are Oven	Completed	10 hrs	0 min	3 hrs
105	Arkitektur: Dansescene - Koble avatar mot kinect	High	Marcus Jernberg	Completed	20 hrs	0 hrs	15.5 hrs
106	Arkitektur: Dansescene - Grid	High	Zana Ali	Completed	40 hrs	0 hrs	30 hrs
107	Funksjon: Dynamisk vanskelighetsgrad	Medium	Are Oven	Completed	4 hrs	0 hrs	2 hrs
108	Funksjon: Dansescene - Utføre Achivements	Medium	0	Completed	4 hrs	0	4 hrs
109	Audio Output Interface	High	0	Completed	20 hrs	0	0
110	Funksjon: Dansescene - Instruktør	Medium	Are Oven	Completed	80 hrs	0 hrs	42 hrs
111	Funksjon: Generator - Tilpasse dans	Medium	Are Oven	Completed	20 hrs	0 hrs	12 hrs
112	Funksjon: Dansescene - Grid: Tilpasse	Medium	0	Completed	60 hrs	0	0
113	Arkitektur: Meny	High	Jan Erik Helskog	Completed	20 hrs	0 hrs	20 hrs
114	Funksjon: Meny - Presentere statistikk	Medium	0	Completed	60 hrs	0	0
116	Funksjon: Meny - Velge bruker	Medium	0	Completed	20 hrs	0	0
118	Unity: Friske opp Unity kunskap	High	Jan Erik Helskog	Completed	8 hrs	0 hrs	8 hrs

119	Sette seg inn i Doxygen	Medium	Zana Ali	Completed	4 hrs	0 hrs	1 hrs
121	Arkitektur: Grunnleggende database design	High	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	5 hrs
122	Møtereferat [26.02.2013]	High	Henrik Olsson	Completed	4 hrs	0 hrs	4 hrs
123	Arkitektur: Database, grunnleggende metoder.	High	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	8 hrs
124	Lage en designdokument mal	High	Zana Ali	Completed	4 hrs	0 hrs	3 hrs
125	Møteinnkallelse [05.03.2013]	High	Dagfinn Kjærnet	Completed	30 min	0 min	30 min
126	Testrutine: Databaser	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
127	Arkitektur: Dansescene - Grid (Ressurs#2)	High	Are Oven	Completed	20 hrs	0	7 hrs
129	Dokumentasjon: Database og databasehåndtering	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
130	Dokumentasjon: Audio Import	High	Henrik Olsson	Completed	2 hrs	0 hrs	5 hrs
131	Dokumentasjon: Grid.cs	High	Zana Ali	Completed	3 hrs	0 hrs	4 hrs
132	Dokumentasjon: Menysystemet - teknologi dokument NG	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
133	Arkitektur: Menysystem - New User Scene(3&4)	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
134	Arkitektur: Menysystem - Song Select(5&6&7)	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	8 hrs
135	Arkitektur: Menysystem - Statistikk(8&9)	High	0	Completed	4 hrs	0	0
136	Arkitektur: Menysystem - Achievments(10)	High	0	Completed	2 hrs	0 hrs	2 hrs
138	AudioVisualizer	High	Henrik Olsson	Completed	20 hrs	0 hrs	40 hrs
140	Dokumentasjon: Zigfu	High	Marcus Jernberg	Completed	3 hrs	0 hrs	3 hrs
141	Dokumentmal: Testcaseoversikt	High	Dagfinn Kjærnet	Completed	1.5 hrs	0 hrs	1.5 hrs
142	GDD: Dokumenmtasjon - Danserom	High	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
143	Dokumentasjon: Sprint 4 Debrief	High	Dagfinn Kjærnet	Completed	2 hrs	0	2 hrs
144	Teknologidokument: SQLite	High	Dagfinn Kjærnet	Completed	2 hrs	0 min	2 hrs
146	Prototype: Koble avatar mot grid	High	Marcus Jernberg	Completed	2.5 hrs	0 hrs	2.5 hrs
147	Prototype: Koble avatar mot grid	High	Zana Ali	Completed	2.5 hrs	0 hrs	2.5 hrs
148	DesignDokument: Koble avatar mot kinect	High	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
156	Fullføre dokumentasjon	High	Marcus Jernberg	Completed	14 hrs	0 hrs	14 hrs
157	Fullføre dokumentasjon	High	Zana Ali	Completed	14 hrs	0 hrs	14 hrs
158	Dokumentasjon: Fullføre dokumentasjon	High	Jan Erik Helskog	Completed	14 hrs	0 hrs	18 hrs
159	Fullføre dokumentasjon	High	Dagfinn Kjærnet	Completed	14 hrs	0 hrs	18 hrs
160	Fullføre dokumentasjon	High	Are Oven	Completed	14 hrs	0 hrs	18 hrs
161	Forberede presentasjon 2	High	Dagfinn Kjærnet	Completed	20 hrs	0	20 min
162	Forberede presentasjon	High	Marcus Jernberg	Completed	20 hrs	0 min	20 hrs
163	Forberede presentasjon	High	Zana Ali	Completed	20 hrs	0	20 hrs
164	Forberede presentasjon 2	High	Are Oven	Completed	20 hrs	0	20 hrs
165	Forberede presentasjon2	High	Jan Erik Helskog	Completed	20 hrs	0 min	20 hrs
166	Forberede Presentasjon	High	Henrik Olsson	Completed	20 hrs	0	20 hrs
167	Fullføre dokumentasjonen	High	Henrik Olsson	Completed	14 hrs	0 hrs	18 hrs

169	Funksjon: Dans - Grid - Dybde	Medium	Zana Ali	Completed	1 hrs	0 hrs	30 min
170	Funksjon: Dans - Deteksjon - Ressurs #1	Medium	Zana Ali	Completed	20 hrs	0 hrs	24 hrs
171	Funksjon: Dans - Deteksjon - ressurs #2	Medium	Are Oven	Completed	20 hrs	0 hrs	4 hrs
172	Funksjon: Musikk - Algoritme	Medium	0	Completed	0 hrs	0 hrs	0
173	Funksjon: Musikk - Sync m/musikk	Medium	0	Completed	10 hrs	0	0
174	Funksjon: Statistikk - Storyboard	Medium	Marcus Jernberg	Completed	3 hrs	0 hrs	3.5 hrs
175	Funksjon: Prestasjoner - Implementere rammeverk	Medium	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	5 hrs
176	Funksjon: Statistikk - Lagre	Medium	0	Completed	15 hrs	0	1 hrs
178	Funksjon: Prestasjoner - Legge til pop-up	Medium	Dagfinn Kjærnet	Completed	10 hrs	0 hrs	10 hrs
179	Funksjon: Statistikk - Presentere	Medium	0	Completed	15 hrs	0	0
180	Funksjon: Prestasjoner - legge til databasestøtte	Medium	Dagfinn Kjærnet	Completed	10 hrs	0 hrs	10 hrs
181	Funksjon: Statistikk - Stolpediagram	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	11 hrs
183	Arkitektur: Meny - Storyboard v2 - bruker	High	Jan Erik Helskog	Completed	6 hrs	0 hrs	10 hrs
186	Arkitektur: Meny - Storyboard v2 - InGame	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	6 hrs
187	GDD: Oppdatere - legge til animerte bevegelser	Medium	Henrik Olsson	Completed	2 hrs	0 hrs	2 hrs
188	Planlegging: Dele opp flere user stories	High	0	Completed	10 hrs	0 hrs	10 hrs
189	Funksjon: Dans - Grid - Dybde - Ressurs #2	Medium	Are Oven	Completed	1 hrs	0 min	30 min
190	Funksjon: Statistikk - Research	Medium	Marcus Jernberg	Completed	3 hrs	0 hrs	4 hrs
191	Funksjon: Musikk mappe	High	Jan Erik Helskog	Completed	10 hrs	0 hrs	10 hrs
192	Funksjon: Musikkanalyse, ressurs 2	High	Dagfinn Kjærnet	Completed	15 hrs	0 hrs	15 hrs
193	Funksjon: Dansescene - Instruktør - Ressurs #2	Medium	Zana Ali	Completed	80 hrs	0 hrs	36 hrs
194	Dokumentasjon: Code Review	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	5 hrs
195	Funksjon: DoNotDestroy Unity	Medium	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
196	Support: Dance Generator	Medium	Henrik Olsson	Completed	8 hrs	0 hrs	12 hrs
197	Funksjon: Vanskelighetsgrad - Lengre tid mellom bevegels	Medium	Zana Ali	Completed	20 hrs	0 hrs	1 hrs
198	Funksjon: Instruktør - Visuell	Medium	Marcus Jernberg	Completed	10 hrs	0 hrs	5.5 hrs
199	Testing: Dansegenerator	Medium	Henrik Olsson	Completed	4 hrs	0 hrs	1.5 hrs
200	Merge: Dancegenerator & AudioImport	High	Henrik Olsson	Completed	8 hrs	0 hrs	8 hrs
201	Funksjon: Meny - Change Password scene	High	Jan Erik Helskog	Completed	3 hrs	0 hrs	3 hrs
202	Funksjon: Meny - Startupscreen scene	High	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
203	Funksjon: Meny - Tilpasse scene	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
204	Meny: Oppdatere til metro	High	Jan Erik Helskog	Completed	6 hrs	0 hrs	6 hrs
205	Funksjon: Meny - Statistikkoversikt scene	High	Jan Erik Helskog	Completed	2 hrs	0 hrs	2 hrs
206	Funksjon: Dansegenerator - Opprette bevegelsene	Medium	Are Oven	Completed	10 hrs	0 hrs	6 hrs
207	Funksjon: Database - Flere funksjoner	High	Dagfinn Kjærnet	Completed	15 hrs	0 hrs	15 hrs
208	Ingame pausemeny	Medium	Henrik Olsson	Completed	6 hrs	0 hrs	6 hrs
209	Funksjon: Scoringsystem	Medium	Zana Ali	Completed	20 hrs	0 hrs	14 hrs

210	Funksjon: Scoringsystem - Ressurs#2	Medium	Are Oven	Completed	20 hrs	0 hrs	4 hrs
211	Implementasjon: Merge Statistikk og Database	Medium	Marcus Jernberg	Completed	6 hrs	0 hrs	4.5 hrs
212	Dokumentasjon: Dokumentere kode	Medium	Dagfinn Kjærnet	Completed	10 hrs	0 hrs	10 hrs
213	Implementasjon: Sammensetting av komponenter	High	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	4 hrs
214	Implementasjon: Sammensetting av komponenter	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	5 hrs
215	Dokument: Videre utvikling	High	Henrik Olsson	Completed	12 hrs	0 hrs	10 hrs
216	Dokumentasjon: Dokumentere kode	Medium	Jan Erik Helskog	Completed	8 hrs	0 hrs	8 hrs
217	Funksjon: Visuell ingame danse instruksjon.	Low	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
218	Dokumentasjon: Opprydning i OnTime	Medium	Dagfinn Kjærnet	Completed	2 hrs	0	2 hrs
219	Dokumentasjon: Designdokument	Medium	Zana Ali	Completed	15 hrs	0 hrs	10 hrs
220	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Dagfinn Kjærnet	Completed	16 hrs	0 hrs	16 hrs
221	Dokumentasjon: Oppdatere dokumentasjon	Medium	Jan Erik Helskog	Completed	20 hrs	0	26 hrs
222	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	3 hrs
223	Dokumentasjon: oppdatere dokumentasjon	Medium	Are Oven	Completed	6 hrs	0 hrs	0
225	Dokumentasjon: Sluttrapport	Medium	Henrik Olsson	Completed	20 hrs	0	20 hrs
227	Dokumentasjon: oversiktsdokument	Medium	Marcus Jernberg	Completed	2 hrs	0 hrs	1 hrs
228	Ordne tekst, info, logoer osv til plakat	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	5 hrs
229	Dokumentasjon: Oppdatere	Medium	Henrik Olsson	Completed	8 hrs	0	10 hrs
231	Brukermanual	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	5 hrs
232	Sprint 9 debrief	Medium	Henrik Olsson	Completed	2 hrs	0 hrs	1.5 hrs
226	Dokumentasjon: Oppdatere webside	Medium	0	In Progress	10 hrs	10 hrs	0
50	Konkuranseelement	Low	0	New Request	0	0	0
52	Støtteapparat	Medium	0	New Request	0	0	0
117	Funksjon: Meny - Admin	Low	0	New Request	20 hrs	20 hrs	0
137	Arkitektur: Menysystem - Admin(11)	Low	0	New Request	1 hrs	1 hrs	0
152	Funksjon: Designe egne avatarer	Low	0	New Request	100 hrs	100 hrs	0
153	Funksjon: Manuelt designe danser	Low	0	New Request	200 hrs	200 hrs	0
154	Funksjon: Automatisk deteksjon av funksjonsnivå	Low	0	New Request	100 hrs	100 hrs	0
155	Funksjon: Mulighet for å spille fler samtidig.	Low	0	New Request	50 hrs	50 hrs	0
182	Funksjon: Statistikk - Linjediagram	Medium	Marcus Jernberg	New Request	5 hrs	0 hrs	8 hrs
184	Funksjon: Statistikk - Spiderdiagram	Low	0	New Request	5 hrs	5 hrs	0
185	Arkitektur: Meny - Storyboard v2 - Admin	Low	0	New Request	5 hrs	0 min	0
0	0	0	0	0	0	0	0

Kravspesifikasjon

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	16.03.2013	DK
2.0	Lagt inn krav	19.03.2013	MJ
3.0	Lagt inn flere krav	19.03.2013	DK
4.0	Lagt inn flere krav og fargekodet kravene	24.05.2013	AO
5.0	Oppdatert dokument	24.05.2013	DK
6.0	Oppdatert userstory status.	25.05.2013	DK, AO

Kontrollert av:	HO, MJ

Innholdsfortegnelse

Innledning

Selv om "pure" SCRUM-metodikken ikke benytter seg av en tradisjonell kravspesifikasjon har vi valgt å benytte oss av dette. Det gjør vi først og fremst av personlige preferanser for å få en bedre oversikt over de overordnede kravene og de user-storyene de etter hvert omfatter. Dessuten føler vi det vil være lettere å presentere en slik konkret spesifikasjon til kunden ved produktlevering, da de på denne måten får en konkret oversikt over hvilke av de opprinnelige kravene som er oppfylt.

Da vi i stor grad selv har fått forme oppgaven og kun har fått en håndfull svært løst definerte krav fra kunden, har gruppen selv prøvd å klassifisere og definere de forskjellige kravene og produkteier (Dagfinn) har hatt siste ordet.

Tabellforklaring

Kravene klassifiseres i A- B- og C-krav. Et A-krav er et krav som ***skal*** oppfylles, ofte omfatter dette krav som må oppfylles for å få et produkt som fungerer. Et B-krav er et krav som ***bør*** oppfylles, dette vil ofte være krav som består av funksjoner som vil være viktige for et ferdig produkt, men ikke nødvendigvis ting produktet avhenger av. C-krav er krav det ikke er så farlig med, men som ***gjerne*** må legges til dersom vi får tid mot slutten av prosjektet.

Siden det å oppfylle et krav ofte avhenger av at opptil flere småoppgaver må løses har vi valgt å samle disse i dette dokumentet. Videre har vi valgt å dele dem opp i de som er kritiske for å oppfylle kravet, og de vi kan anse som en tilleggsoppgave da de legger til funksjoner som påvirker kravet, men ikke er strengt nødvendige for å oppfylle det. Disse vil så bli fargekodet slik at det blir lettere å se status til de forskjellige oppgavene.

Fargekodene er som følger: fullført, ny forespørsel, avvist.

Vi kommer også til å opprettet tester beregnet på de overordnede kravene. Det er disse testene vi kommer til å bruke for å vurdere om kravet skal anses som godkjent, eller om vi må vurdere om noen av tilleggsoppgavene burde reklassifiseres som kritiske. Testene vil også bli fargekodet utfra resultatet av testen.

Fargekodene er som følger: godkjent, ukjent, feilet.

A-Krav

Krav	Test ID	Omfatter følgende userstories		Kommentar
		Kritiske for kravet	Tillegg utover kravet	
Spillet skal oppfatte brukerbevegelser	27	105, 103, 106, 146, 147, 170, 171	59 169, 189, 127	
Følge statistikk	28	94, 96, 98, 121, 123, 176,179, 181, 207, 209	61, 85, 175, 179, 182, 190, 205	
Spillet skal tilpasses brukerens behov	29	111, 112, 197, 203	80, 52, 51, 107	
Svært enkelt i bruk	34	56, 113	54, 84, 85,87, 114, 116, 117, 133, 134, 135, 136, 137, 183, 185, 186, 204	Menyene skal være lette å forstå og raske å gå igjennom.
Justerbar vanskelighetsgrad	35	197	66, 81, 107	
Hver bruker skal kunne ha sin egen brukerkonto	36	55, 121, 123		

B-krav

Krav	Test ID	Omfatter følgende userstories		Kommentar
		Kritiske for kravet	Tillegg utover kravet	
Legge til egen musikk	37	53, 92	70, 82, 172, 173, 191, 192	
Prestasjoner	38	175	49, 62, 83, 108, 178, 180	
Autogenerere danser til egen musikk	39	101, 104, 206	95, 111, 196, 199, 200,	

C-krav

Krav	Test ID	Omfatter følgende userstories		Kommentar
		Kritiske for kravet	Tillegg utover kravet	
Muligheten til å designe avataren etter brukerens egne begrensninger.	40	152		Slik at brukerne lettere kan identifisere seg med avataren. F.eks avatar som mangler en arm, som sitter i rullestol osv.
Muligheter for manuelt design av danser	41	153		
Automatisk deteksjon av funksjonsnivå	42	154		
Flerspiller funksjon	43	155		

Game Design Document

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBustudentprosjekt 2013

…… ……

…… ……

…… ……

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Første utkast	21.01.2013	ALLE
2.0	Opprettet og satt sammen bidrag til GDD	22.01.2012	JEH
3.0	Lagt til Story Board over Menyene	28.02.13	JEH
4.0	Lagt til avatarer og danserom	14.03.13	MJ
5.0	Lagt til musikk i lyd og gameplay	18.03.13	HO
6.0	Oppdatert dansebevegelser	23.05.13	ZA
7.0	Oppdatert prestasjonsdelen og lagt til noen eksempelprestasjoner.	24.05.2013	DK
8.0	Endret statistikk og instruktør	24.05.2013	MJ
9.0			
10.0			

Kontrollert av:	ZA, HO

Innholdsfortegnelse

1. Innledning

1.1 Om dokumentet

Dette Game Design Dokumentet gir en beskrivelse av utformingen og innholdet til spillet LetsDance. Dette vil være et dokument i utvikling, og vil derfor kunne bli forandret uten advarsler.

1.2 Forfattere

Are Oven (AO), Zana Ali (ZA), Marcus Jernberg (MJ), Henrik Olsson (HO), Dagfinn Kjærnet (DK) og Jan Erik Helskog (JEH).

1.3 Konseptet

LetsDance er et morsomt og spennende danse-musikk spill. I spillet skal spilleren i løpet av en sang gjennomføre forskjellige dansetrinn og blir målt etter hvor godt disse blir utført. Spilleren vil kontrollere en avatar ved hjelp av Kinect, og bruke dette til å utføre gitte danse bevegelser. Det er beregnet at LetsDance er skal brukes som et rehabiliterings tilbud hos Sunnaas sykehus.

1.4 Krav og spesifikasjoner

- Mulighet for login og registrering av brukere
- Enkelt menysystem som fungerer feilfritt
- Lyd 2d og 3d
- Kinect brukes som kontroller i selve spillet
- Touchscreen brukes som kontroller i meny
- Mulighet for å legge inn egne sanger
- Justerbar vanskelighetsgrad
- Dynamisk vanskelighetsgrad
- Bruke Unity3D
- Spilleren representeres som en tegneseriefigurpå skjermen.
- Brukerstatistikk.
    - Antall perfekt, utmerket og ok bevegelser
    - Total poengsum

1.5 Målgruppe

Dette er et spill som er rettet mot personer i rehabiliteringsfasen, men passer for alle aldersgrupper og personer som ikke er i en rehabiliteringsprosess.

1.6 Systemkrav

1. Windows 7 eller nyere operativsystem
2. Kinect

3. Touchscreen

2. Game Play

Spillet vil i utgangspunktet ha en spill modus, men med muligheten til å utvide til flere. Denne spill modusen vil være en slags normal modus. Andre eventuelle moduser kan være multiplayer, "Dance Battle" og "Last One Standing", her vil spillere bli eliminert hvis de ikke får høy nok score. Som det har blitt nevnt vil vi i demoen vi lager kun fokusere på den normale singel-player modusen. I denne spill modusen vil spilleren få poeng etter hvor bra dansetrinnene blir utført. Spillet har tre typer graderinger av treff på dansebevegelsene, perfekt, utmerket og ok. I tillegg til dette er det en ingen dansebevegelsesgradering. Denne vil kun forekomme dersom spilleren ikke engang prøver å utføre dansebevegelsen. For en dypere gjennomgang av poengsystemet og dansebevegelsene, se eget avsnitt om dansebevegelser.

LetsDance bruker Kinect under selve dansingen, spilleren vil derfor ikke være avhengig av å ha noen kontroller. Utenom dansingen vil brukeren ha mulighet til å styre spillet ved å benytte seg av touchscreen. Denne kontroll typen av spillet blir brukt ved login og andre meny valg.

Spilleren har i LetsDance muligheten til å forandre på vanskelighetsgraden. Dette gjøres enkelt i menyen. Da vil man i all hovedsak forandre på hastigheten og antall bevegelser mellom beats. Spillet har en form for dynamisk vanskelighetsgrad, dette er gjort for å kunne minske brukerens behov for å navigere seg rundt i menyer. Ved å ha dette føles det at spillet er utfordrende selv om spilleren blir flinkere. For en bedre beskrivelse av den dynamiske vanskelighetsgraden, se eget avsnitt.

Som en motivasjons faktor er det mulig å samle og oppnå prestasjons poeng, som man senere kan gå tilbake til for å se på. En oppnådd prestasjon kan være å få utmerket på alle dansebevegelsene i løpet av en sang. For en grundigere lesning og gjennomgang av prestasjonssystemet, se eget avsnitt.

Spillet fører en statistikk over spillerens resultater. Disse dataene finner brukeren enkelt igjennom menyen. Resultatene blir fremstilt på en enkel og oversiktelig måte ved hjelp av enkle diagrammer, se eget avsnitt for statistikk.

Før en bruker starter en spillsesjon vil applikasjonen gå til et vindu hvor brukeren har muligheten til å velge en egen lydfil/musikkfil fra et forhåndsbestemt mappeområde. Brukeren vil ha muligheten til å legge til nye filer i denne mappen for å skape variasjon. Applikasjonen vil så scanne igjennom filen for å finne punkter hvor det er et slag/beat, for å gjøre det mulig å bevege seg i takt med musikken. Bevegelsene vil bli fastbundet i en låt for at brukeren skal ha mulighet til å trene på en bestemt låt å bli dyktigere. Statistikken som blir ført vil vise hvilken lydfil som har blitt brukt under sesjonen.

3. Teknologi

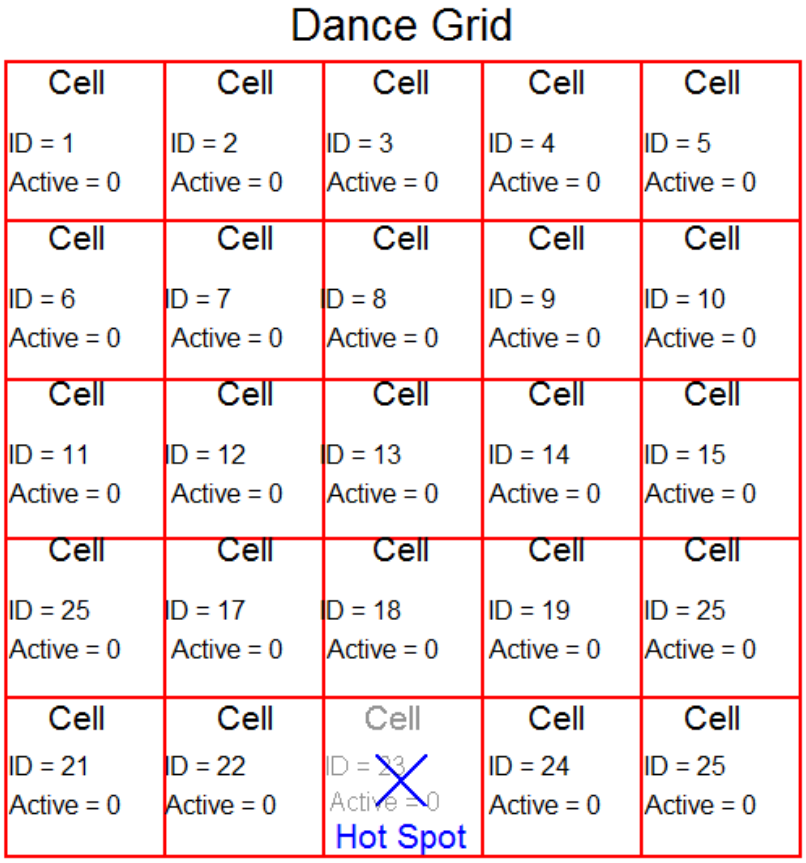
3.1 Hot Spot

"Hot spotten" for spilleren er lokalisert midt på den horisontale aksen av celler og nederst på den vertikale aksen av celler. De andre cellene "tegnes" ut ifra hvor spilleren står. Punktet hvor spilleren står vil alltid være "Hot Spotten"

3.2 Celler

Det er i "Cellene" dansetrinnene blir oppdaget. Alle de separate cellene har en unik Cell ID_ for identifikasjon, og en Active_ verdi for å påvise et dansetrinn. Den Active_ verdien kan være satt til

enten 0 (inaktiv) eller 1 (aktivert). Bare celler med en Active_ verdi på 1 er i stand til å oppdage dansetrinn. (Se figur 1)



Figur 1, Dansecellene, områder for registrering

4. Spilleren

Den fysiske spilleren står foran kinect-kameraet

4.1 Spiller tracking

Kinecten sporer ulike ledd og kroppsdeler (punkter på kroppen) til spilleren som i kinect fungerer som ”kontrollen”.

Et unikt kroppspunkt kalt ”Player Hot Spot” skal spores. Dens X posisjon bør være i sentrum av spilleren (i brystet), mens Y posisjonen bør være på spillerens fot. Den nederste foten skal alltid brukes for å representere spillerens ”Hot Spot”

Kinect sporer X og Y posisjoner i hodet, bryst, lysk, skuldre, hender, føtter, albuer og knær på den fysiske spilleren. Hvert punkt har en unik ID_ verdi tilknyttet punktet. Det vil være en avatar på skjermen som representerer spilleren på skjermen ved å bruke sporingsdata.

5. Avatar

Spilleren vil bli representert på skjermen som en avatar gjennom spillerens kroppspunkter og posisjon (gjennom hot spotten). Spillerens avatar burde gjenspeile en ekte person som spilleren kan identifisere seg med.

Det vil være mulig å velge mellom to forskjellige avatarer, en gutt/mann og en jente/kvinne. Dette er gratismodeller hentet fra Unitys egen butikk.

5.1 Avatarens kroppspunkter

Avatarens kroppspunkter er representert ved en X og en Y posisjon på spillskjermen. Disse punktene knyttes opp mot spillerens kroppspunkter som spores av kinect-kameraet og skal representere samme kroppsdel/ledd.
Med andre ord, hvert kroppspunkt på avataren skal ha samme ID_ verdi som det tilsvarende kroppspunktet på spilleren (avatarens høyre albue skal ha samme ID_ som spillerens høyre albue).

5.2 Avatar hot spot

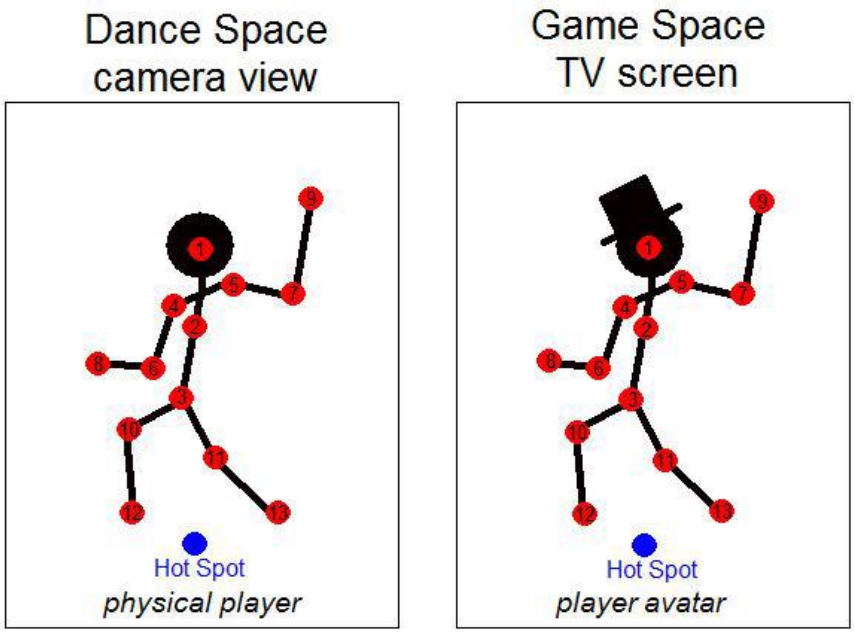
Avatarens hotspot er representert ved en X og en Y posisjon på spill skjermen og brukes som et referansepunkt for posisjonering inne i spillet (på spillskjermen). Plasseringen av avatarens hot spot bestemmes av spillerens hot spot foran kinecten. Hvis spilleren beveger seg utenfor kinectens kamera vil den siste kjente hot spot posisjonen bli brukt for å representere avatarens hot spot.

5.3 Posisjonering

Avatarens hot spot posisjon i ”spill området” på skjermen bør være relativ i forhold spillerens hot spot posisjon på ”spill området” på gulvet foran kinect-kameraet.

5.4 Avatar etterligning

Avatarens kroppspunkter og posisjonering (hot spot) bør være nøyaktig lik spillerens kroppspunkter og posisjonering. Avataren skal bevege sine lemmer (armer, ben, føtter etc) nøyaktig likt som spilleren utfører bevegelsene til enhver tid. Hvis spillerens hot spot er utenfor kinectens kamera, vil avatarens holdning endres til en forhåndsbestemt holdning.



Figur 2, avatar i forhold til spiller

6. Områder

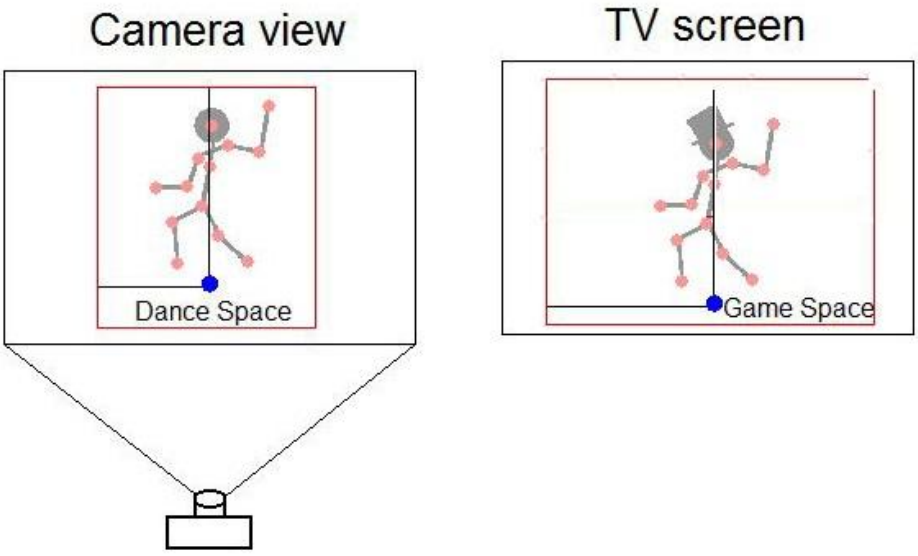
Områder er definert som firkantede statiske felt, og har en størrelse X_ og Y_. Der er to forskjellige områder i spillet, et foran kameraet (danseområdet) og området på skjermen hvor man ser avataren, dansetrinn og selve dansen.

6.1 Danse-området

Danseområdet er et statisk vinkelrett område av bildet kinect-kameraet leser. Danseområdet har en fast X_ og Y_ som leser og lagrer spillerens hot spot-data (X og Y posisjon).

6.2 Spill-området

Spill-området er et statisk firkantet område på skjermen som har fast X og Y posisjon på skjermen, så vel som width_ og height_. Plasseringen av avatarens hot spot på skjermen skal være lik hot spotten til spilleren på danse-området.



Figur3, Områder

6.3 Danserom

I spillsammenheng blir utseende/grafikken på spillet ofte sett på som svært viktig. I LetsDance har vi valgt å gå bort fra å legge for mye vekt på dette, siden målgruppen for spillet vårt er pasienter under rehabilitering kan for mange visuelle effekter virke rotete og bli for mye for enkelte av dem. Så vi velger å lage et enkelt utseende på "Danserommet" for å vektlegge det funksjonelle med spillet.

6.4 Beskrivelse av Danserommet

For å få opplevelsen av å stå i et dansestudio bruker vi parkett-tekstur på gulvet, det blir også et miksebord og noen høytalere i bakgrunnen, sammen med LetsDance-logoen. Vegger og tak lager vi enkelt med LetsDance sin lilla farge.

7. Dansebevegelser

LetsDance baserer seg rundt dansebevegelser, og disse bevegelsene skal demonstreres ved hjelp en instruktør og griden. En dansebevegelse kan være en kombinasjon av flere dansetrinn. De kroppsdelene som skal brukes i en bevegelse blir grønt, med det som skal gjøres først markert i en klar grønnfarge. Framtidige trinn har en grønnfarge med gjennomsiktighet. Etter at en del av et dansetrinn er fullført, vil neste del få en klar grønnfarge. Etter at hele dansetrinnet er helt ferdig vil et nytt dansetrinn hentes.

Flere kroppsdeler kan inngå i en dansebevegelse samtidig. Dansebevegelsene kan bestå av ordinære dansetrinn slik figur 3 viser[1], de kan også være aerobic og chairobic øvelser som er mer relevant for brukergruppen til LetsDance.

figur 3

7.1 Aerobics

Aerobic er en form for fysisk trening som kombinerer rytmisk aerobic trening med styrke og strekke rutiner. Bevegelsene i spillet vil være utført til en musikk med forskjellige BPM(taktslag i minuttet)[2]. Siden aerobicøvelser i stor grad trener mobilitet og balanse er det noe som blir brukt mye i spillet. Formålet med aerobic bevegelser i LetsDance er å bidra til å opprettholde dagliglivets funksjoner og forebygge funksjonssvikt av pasienter med slag. Disse har nedsatt motorikk og svekkende mentale funksjoner, derfor er det en økt risiko for fall og skader som følge av fallene. Dansebevegelsene skal være enkle å huske på, da hukommelsesevnen kan være svekket hos pasientene[3].

7.1.1 Chairobics

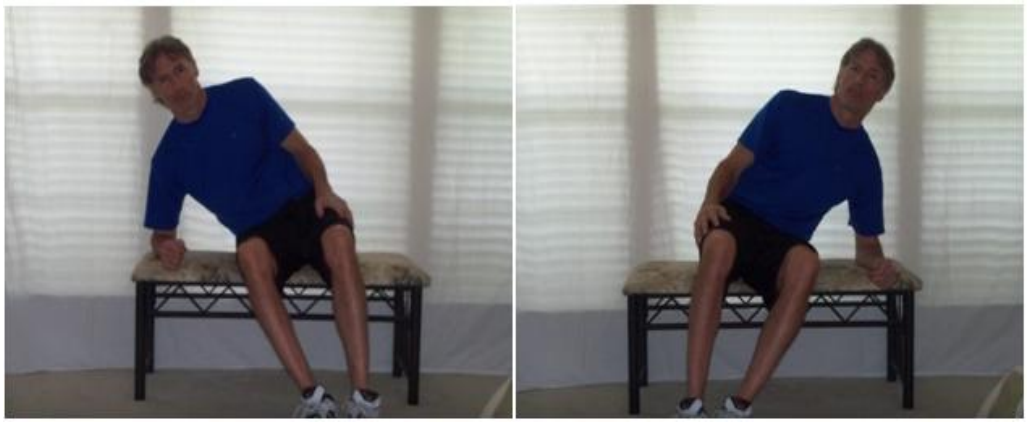
En stor del av brukergruppen til LetsDance er rullestolbrukere. Spillet skal være tilpasset slik at de også kan delta i spillet. Chairobics er en samling aerobic-rutiner som er tilpasset for å dekke behovene til personer som ikke kan bruke beina. Bevegelsene kan være alt fra enkle yogaøvelser til mer akselerende og isometriske bevegelser. Disse utføres alle sittende.

7.2 Liste med chairobics bevegelser

7.2.1 Sideskift

Ha tyngden på høyre hofte og skift så til venstre hofte. Brystkassen flytter fra side til side og hofta bør reise seg fra overflaten. Midjen skal ikke bøyes for å lene seg til sidene. Armene bør ha kontakt med benken gjennom hele øvelsen.

7.2.2 Lene seg ned på albuen

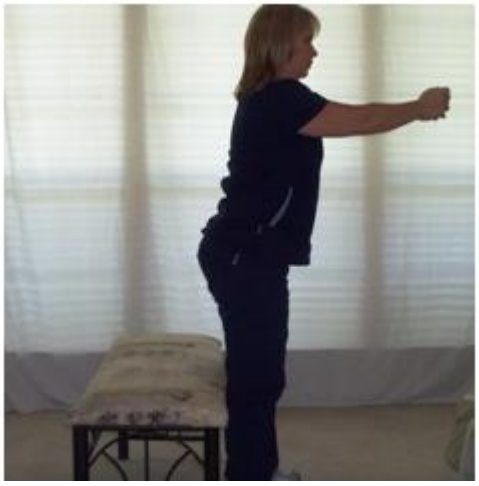


Len deg over på høyre albue, press deg så opp og len deg ned på venstre albue. Gjenta motsatt vei.

7.2.3 Krumme-øvelse

Len deg tilbake, skyv brystkassen frem og trekk skulderbladene sammen. Len deg så fremover og krum ryggen.

7.2.4 Sitte og stå øvelse med samlede hender

Begynn i en sittende stilling. Hold hendene sammen og len deg fremover til rumpa reiser seg, stå så oppreist hvis mulig. Det er også greit å stå i en delvis oppreist stilling. Sett deg ned igjen til startposisjon med samlede hender og gjenta[4].

7.3 Effekten av bevegelser og fysisk aktivitet

Fysiske bevegelser kan ha fysiologiske, mentale og emosjonelle effekter. Fysiske bevegelser som rehabilitering kan føre til økt selvtillit. Aerobic øker koordinasjonen, utholdenheten, styrken og bevegeligheten. Hjertet blir styrket av kardiotrening som aerobics og reduserer risikoen for å utvikle diabetes, hjertesykdommer og fedme.

7.4 Poengsdeling

Rangering av dansebevegelser vurderes utifra vanskelighetsgrad og om en dansebevegelse blir utført på en beat i sangen.

7.4.1 Bevegelse utført på en beat

Vanskelighetsgrad	Poeng
Lett	10
Normal	20
Vanskelig	30

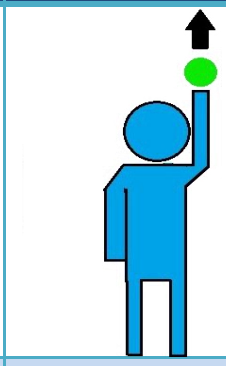
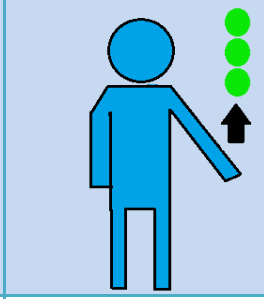
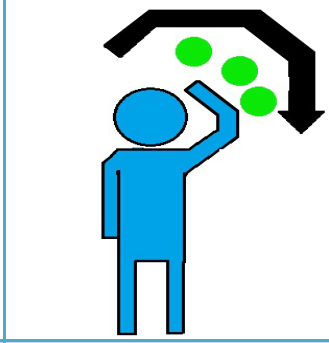
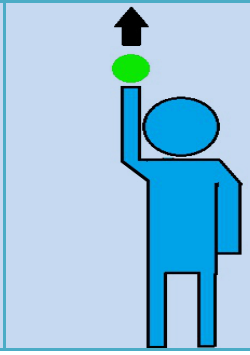
7.4.2 Bevegelse fullført, men ikke på en beat

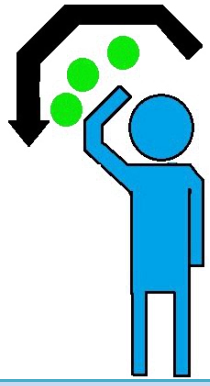
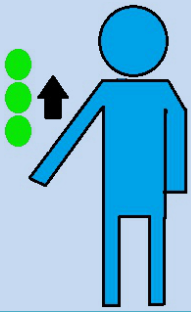
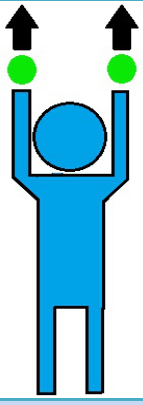
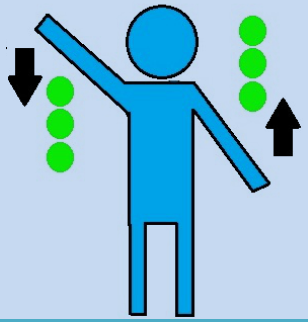
Vanskelighetsgrad	Poeng
Lett	5
Normal	10
Vanskelig	15

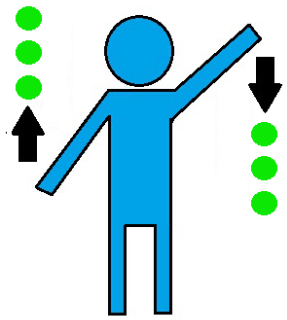
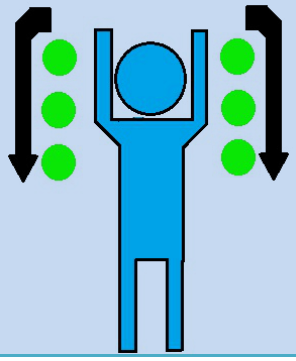
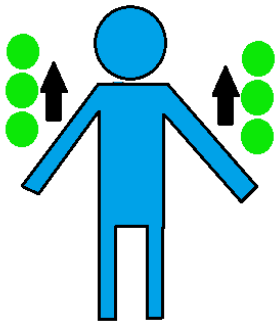
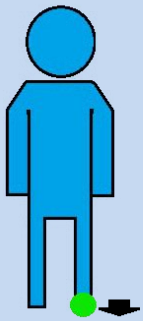
7.4.3 Ingen bevegelse utført

Dersom bevegelsen ikke blir utført blir det ikke gitt noen poeng.

7.5 Liste med dansebevegelser i LetsDance

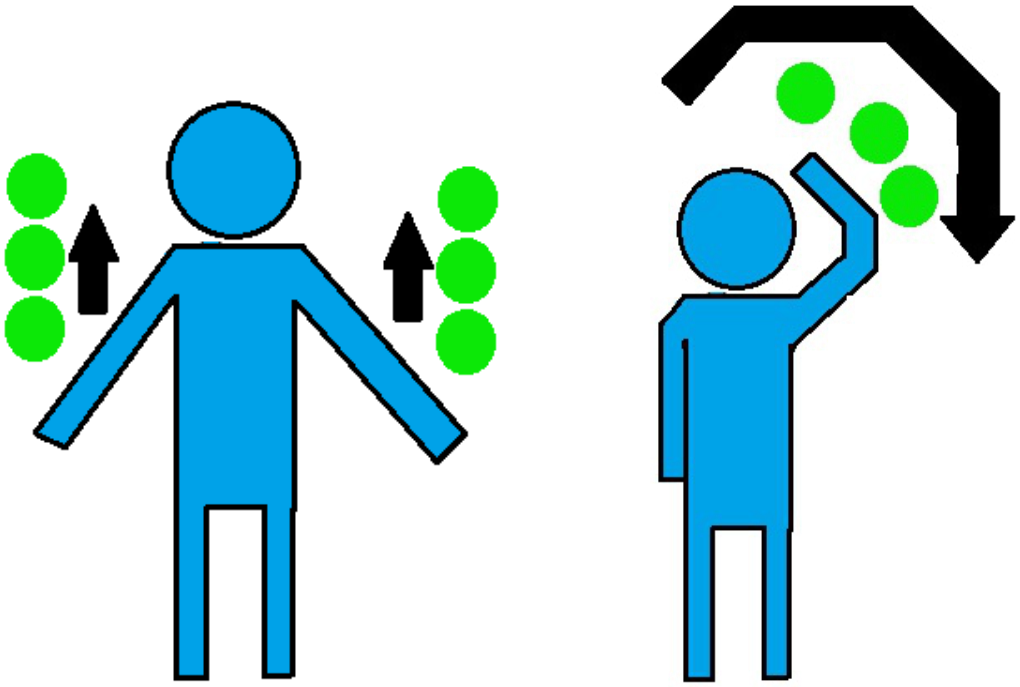
Navn	Instruktør	Beskrivelse
a		Løft venstre hand over hodet og bak mot kulen
b		Løft venstre hand strakt sidelengs
c		Løft venstre hand over hodet og utfør en vinkebevegelse mot venstre
d		Løft høyre hand over hodet og bak mot kulen

e		Løft Høyre hand over hodet og utfør en vinkebevegelse mot venstre
f		Løft høyre hand strakt sidelengs
g		Løft venstre og høyre hand over hodet og bak mot kulene
h		Begynn med høyre arm i skulderhøyde og venstre arm nede langs siden slik at armene er diagonalt. Senk høyre hand sidelengs samtidig som venstre hand løftes strakt opp sidelengs

i		Begynn med venstrehand i skulderhøyde og høyre hand nede langs siden slik at armene er diagonalt. Senk venstre handsidelengs samtidig som høyre hand løftes strakt opp sidelengs
j		Begynn med begge armer over hodet. Senk begge armene strakt ned foran deg
k		Begynn med høyre og venstre arm ned mot siden. Løft så hendene strakt opp
l		Spark med venstre fot. Også kjent som LetsDance varemerke move.

7.5 Instruktør

Når man begynner å spille og skal slå seg løs med dansen. Vil man for hver bevegelse få opp en instruktør som viser hvordan bevegelsen skal utføres. Dette er en stikk-figur som viser hvilke lemmer man skal bevege i hvilken rettning. Vi får opp grønne kuler på skjermen som viser hvordan bevegelsen skal gjøres, men vi ha også instruktøren som viser hvordan man utfører den. Det er lettere å se hvilken arm eller ben man skal bruke ved å se på instruktøren de første gangene man gjør en ny bevegelse.



Figur 4

8. Dynamisk Vanskelighetsgrad

Dynamisk vanskelighetsgrad er et system som forandrer på vanskelighetsgraden automatisk. Det vil si at istedenfor at man stiller den inn selv blir dette gjort automatisk av spillet. Blir dette gjort korrekt vil det gi en veldig bra spillopplevelse for alle spillere, dette fordi alle vil få en passende utfordring. Det som kan være et problem er at mange kanskje føler at de blir holdt i hånden, Men ser vi på fordelene; så kommer man raskere i gang med spillet da man slipper ekstra menyvalg, for dem som ikke er gode på spillet slipper å til å føle seg dårligere enn andre siden de har aldri har satt vanskelighetsgraden til lett.

Målet vårt er å få vanskelighetsgraden til å øke automatisk til et nivå som gjør at spilleren aldri merker noe til at det har blitt lettere eller vanskeligere. Vanskelighetsgraden skal alltid være på et nivå som gjør at spilleren føler seg utfordret uten å være for vanskelig. Når ting blir for vanskelig blir folk frustrert og slutter og spille. Siden dette systemet skal være til mennesker som er under trening er det viktig å øke intensitet (vanskelighetsgraden i vårt tilfelle) i takt med spillernes utvikling, for å ikke stagnere.

Vi kommer til å variere vanskelighetsgraden ved å forandre på reaksjonstiden, man får kortere tid fra man får beskjed om å gjøre noe og til det må være gjort. Totalt antall bevegelser som skal gjøres, mer avanserte bevegelser og presisjon. Presisjon bestemmer vi ved å gjøre punktene som man skal innom større eller mindre alt etter om man skal gjøre det lettere eller vanskeligere.
Ett par ting som vi må være obs på når vi lager kontroller for vanskelighetsgraden er at forskjellige sanger, ut ifra rytmen, kan ha veldig forskjellig vanskelighetsgrad. Dette kan føre til at man får store hopp på grunn av sangene som kan være demotiverende, dette kan også være ødeleggende for statistikken vi fører. For å forhindre dette må vi finne en lur måte slik at alle sanger har ca lik vanskelighetsgrad uansett rytme.

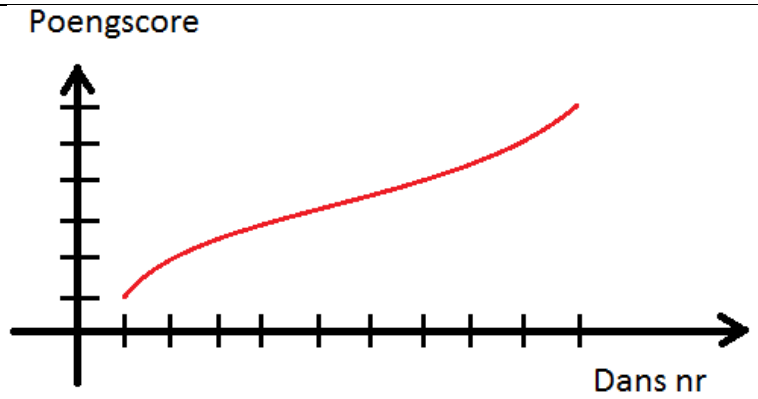
Siden vi vil at vanskelighetsgraden alltid skal følge nivået til personen som spiller spillet bruker vi en AI type som kalles Rubber band AI. Denne metoden ble mye brukt i bilspill og passer spillet vårt veldig bra. I bilspill kjenner man rubber band AI igjen ved at når du kjører bra vil motstanderne dine kjøre ca like bra som deg, ligger de langt bra vil de kjøre bedre for å ta deg igjen. Hvis de ligger langt foran kjører de dårligere eller senere slik at du som spiller skal ha muligheten til å ta dem igjen. Dette fører til at du aldri vil kunne kjøre ifra dem og at du alltid vil ha muligheten til å ta dem igjen da kjørestilen deres bestemmes av avstanden ned til spilleren. Og det er dette som blir målet med LetsDance at nivået alltid skal ligge rett rundt egenskapene til spilleren.

9. Statistikk

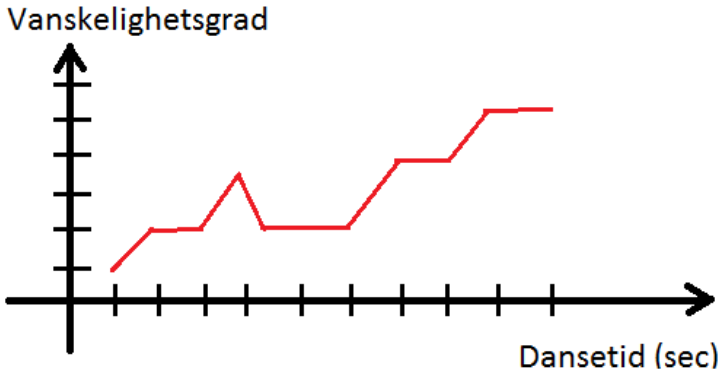
En viktig del av spillet vårt er at brukeren skal kunne følge sin egen progresjon. Vi tror dette vil øke motivasjonen og det vil være et godt hjelpemiddel for brukeren slik at de kan se at treningen de utfører, både i spillet og utenom, faktisk gir uttelling.
Det er spesielt tre elementer vi føler vil være fornuftig å loggføre slik at de kan fremvises i en statistisk oversikt, disse er:

- Poengscore per dans
- Flyten i den dynamiske endringen av vanskelighetsgraden
- Frekvensen av forskjellige typer dansetrinn

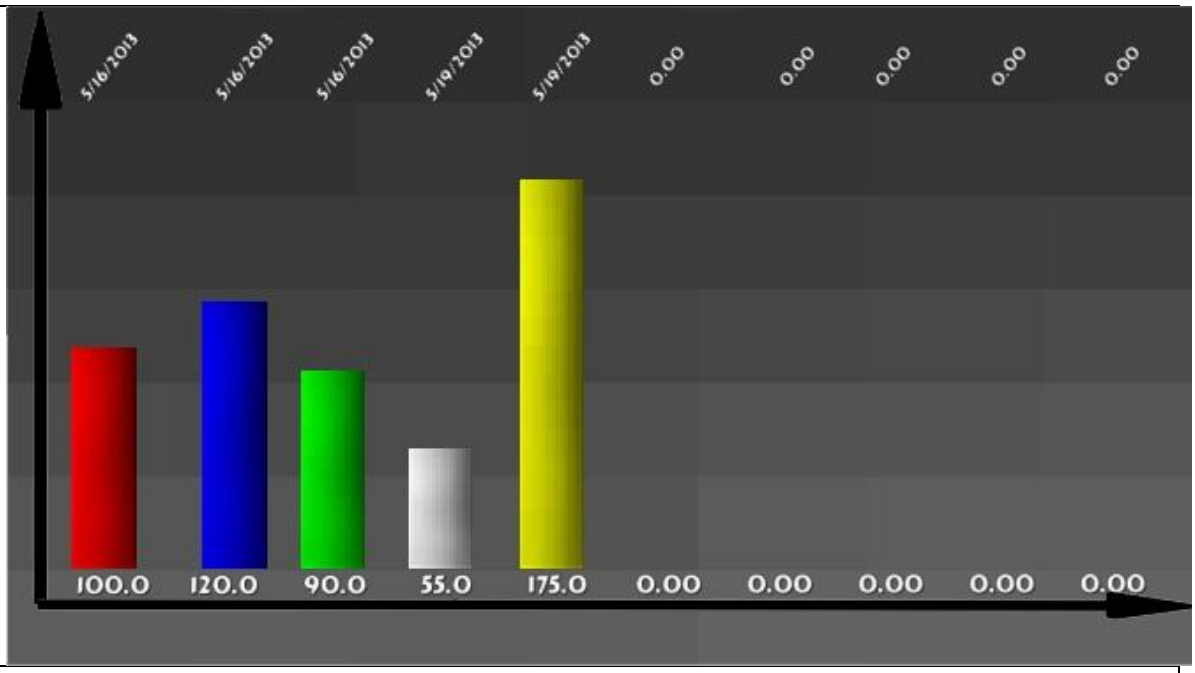
Vi vurderer også å føre statistikk over hvert enkelt dansetrinn/øvelse. Hvis pasienten har høy progresjon i enkeltøvelser kan dette være til stort hjelp.
Vi må finne en oversiktlig måte å fremvise disse statistikkene på, vi kommer til å prøve ut en løsning der vi fremviser de to første statistikkene i et linjediagram, mens den siste kommer vil til å fremvise i et søylediagram.



Poengscore per dans vil defineres av dans nummer og poengscore oppnådd per dans. Denne vil da øke i takt med vanskelighetsgrad fordi vanskelighetsgrad påvirker hvor mange poeng det er mulig å få ila en dans.



Flyt i vanskelighetsgraden defineres av hvordan endringen av vanskelighetsgraden forandrer seg fortløpende, utfra total dansetid og hvilken vanskelighetsgrad spillet er på.



Diagrammet viser poengscore for de 5 første dansene brukeren har danset til den valgte sangen. Her ser vi datoen står skrått over stolpene, og poengscoren er representert ved stolper og poengscoren i tall under stolpene.

10. Prestasjoner

Vi kommer til å benytte oss av prestasjoner i spillet vårt. Prestasjoner er en eller annen form for bekreftelse på at man har utført en oppgave, både store og små oppgaver belønnes. Eksempel på en liten oppgave kan være at man har satt opp profilen sin, eller fullført sin første dans. Eksempler på større oppgaver kan være at man har steget til neste vanskelighetsgrad eller at man har greid ett visst antall "perfekte dansetrinn". Hver prestasjon vil bestå av et representativt ikon, en tittel, en beskrivelse og en poengsum. Poengene vektes utfra hvor vanskelig en prestasjon er, f.eks kan de letteste være verdt 5 poeng og de vanskeligste være verdt 100. Vi vurderer også å dele opp prestasjonene i grupper etter hvor vanskelige de er.

Figur 5 - Eksempel på prestasjons pop-up[5]

Når man har utført en slik oppgave første gang låser man da opp prestasjonen, dette demonstreres av en pop-up som illustrert over, og det blir spilt av en lettgjenkjennelig lyd.
Prestasjoner vil gi brukeren en jevn flyt av positive tilbakemeldinger, vi tror dette vil øke spillerens motivasjon og spillelyst. For noen vil det også øke spillets levetid, da det å samle prestasjoner i seg selv kan være en del av spillopplevelsen.

10.1 Utvikling

For å utvikle prestasjonssystemet vårt kommer vi til å benytte oss av et grunnleggende rammeverk kalt "Progress"[6] som er utviklet av Steve Gargolinski. Dette støtter mange av de grunnleggende funksjonene vi ønsker som ikoner, egen lyd når man oppnår en prestasjon, progresjons prestasjoner, liste over alle prestasjoner osv. Det er også en del funksjoner vi må legge til selv som f.eks pop-up ved oppnåelse[7] og å flytte prestasjonsoversiktslisten til et eget vindu. Vi ønsker også en måte å

lagre hvilke prestasjoner brukeren allerede har låst opp, slik at de ikke låses opp hver gang brukeren begynner en ny sesjon.
Progresjonen vil bli lagret i databasen, antakeligvis kommer vi til å ha en egen tabell knyttet til brukertabellen som lister alle prestasjonene brukeren allerede har låst opp og eventuelt hvor langt man har kommet.

10.2 Prestasjons typer

Vi kommer til å benytte minst tre forskjellige typer prestasjoner som låses opp på forskjellige måter, disse er som følger:

- Prestasjoner som låses opp ved enkelthendelser, f.eks når man har gjennomført sin første dans.
- Prestasjoner som låses opp etter gjentatte hendelser, f.eks en prestasjon som krever at brukeren utfører fem perfekte dansetrinn.
- Prestasjoner som låses opp etter tidsbetingelser, f.eks en prestasjon som låses opp når brukeren har klappet tre ganger i løpet av 10 sekunder.

På denne måten har vi varierte mål for å utføre prestasjonene, istedenfor at man gjør de eksakt samme oppgavene.

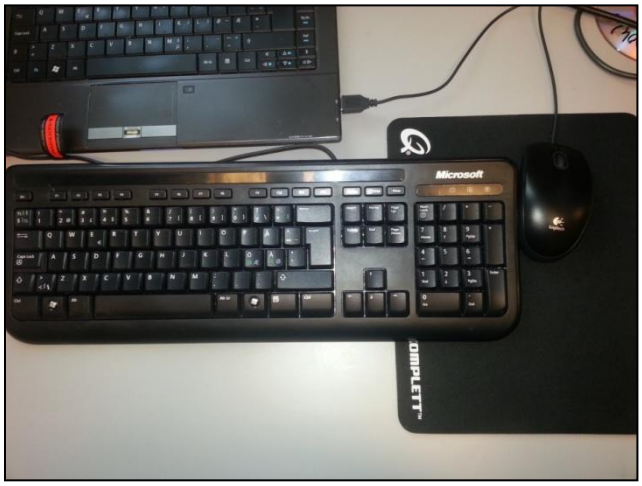
10.3 Forslag til prestasjoner

Her er en liste over noen av prestasjonen vi ønsker å legge til i spillet vårt.

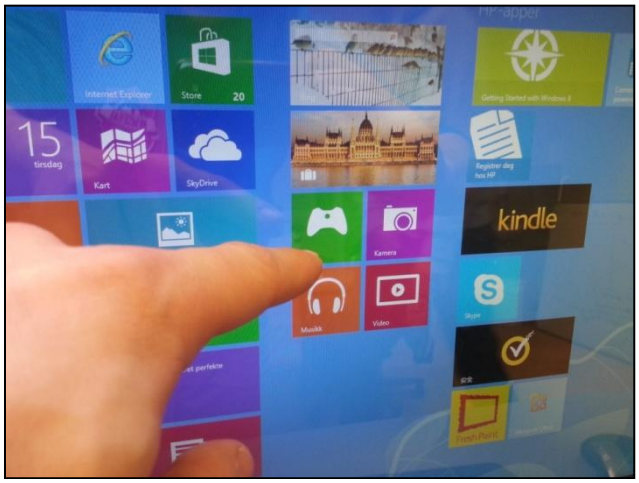
Navn	Poeng	Beskrivelse	Mål verdi
Ny bruker	10	Opprett en ny bruker.	1
Velkommen!	10	Logg inn for første gang.	1
My way or the highway!	20	Importer to sanger.	2
Dansefot	25	Fullfør din første dans.	1
Boogie mester	100	Fullfør 10 bevegelser ila 10 sekunder.	10
Skredderen	10	Endre tilpasnings innstillingen.	1

11. Interface

Grensesnittet mellom mennesket og programvare vil være forskjellige komponenter, først og fremst skal vi benytte oss av en Windows 8 datamaskin med mus og tastatur, samtidig har denne W8 maskinen også trykkfølsomskjerm. ”Touch”- funksjonen vil fungere på lik måte som en musepeker, så vi ønsker å tilpasse applikasjonen vi utvikler slik det vil fungere optimalt med touch funksjonen på pc skjermen.



Input: Keyboard og Mus

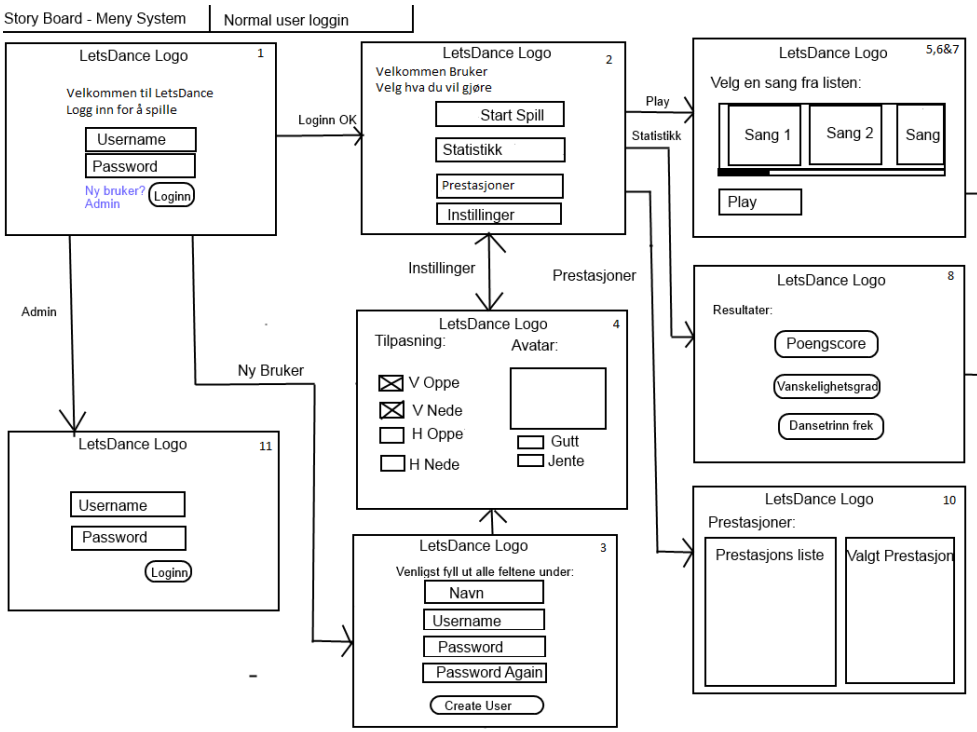


Input: touch skjerm.

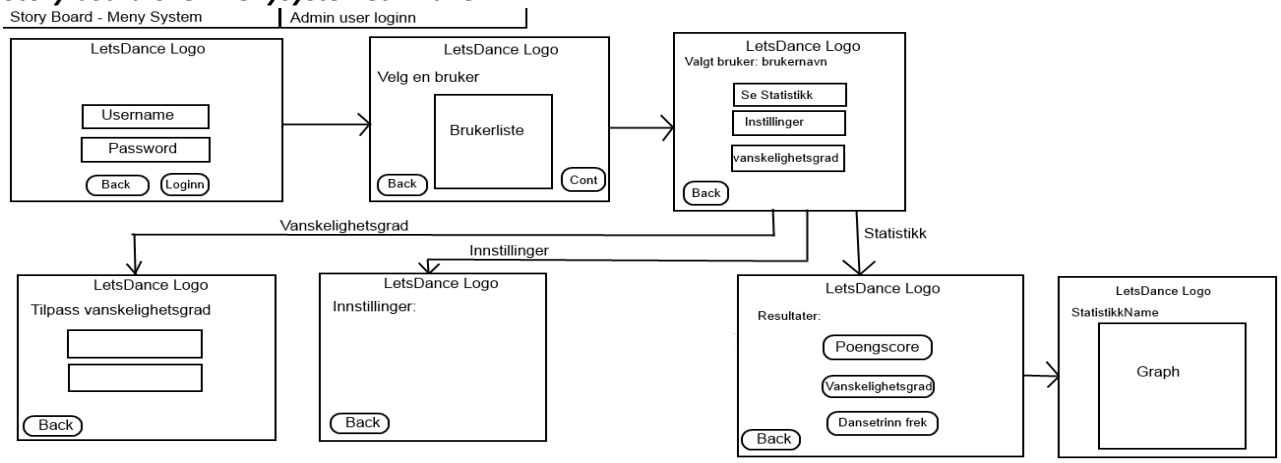
Input: Microsoft Kinect ®

I det vi velger å definere som en spillsesjon skal programmet hente input fra en Microsoft Kinect, denne inputen skal så sammenliknes i programvaren. Når et ”krav” er oppnådd i spillet med tanke på inputen vil man få en form for poeng dette er ytterligere forklart i delen om bevegelse og poeng. Applikasjonen skal ha et ikon på Windows 8 dashbordet som viser tydelig at det er spillet; ”LetsDance”. Når man åpner applikasjonen kommer man til en påloggingsbilde hvor brukeren benytter seg av tastaturet og musen for å fylle inn brukernavn og passord. Om brukeren ikke har noen bruker i systemet enda så vil man måtte trykke på en annen knapp som sier ”ny bruker?”. Her velger brukeren brukernavn og passord som kjent fra mange andre nettsider, samtidig på samtidig vil brukeren velge noen preferanser som vil gjelde for vedkommende profil. Dette blir satt den første gangen slik at man skal slippe dette senere, man skal ha muligheten til å endre dette senere. Når man så går videre skal man kunne velge nytt spill eller brukerprofil. Når man velger nytt spill kan man velge å hente inn egen musikk. Denne musikken skal analyseres for å hente ut ”beat’en” i låten for å kunne bruke dette i spillsesjonen.

På bildene under ser vi to forskjellige story boards. Disse gir en beskrivelse av bruker menyen samt menyen for administratorer.



Story board over menysystemet: Bruker.



Story board over menysystemet: Admin.

12. Lyd

Lyd er et viktig hjelpemiddel til å formidle forskjellige handlinger, for eksempel kan være en bekreftende lyd som kommer hver gang du har trykket på en knapp. Lydene som kommer til å være i spillet kommer til å ha en bestemt funksjon som skal gjøre det lettere å få med seg hva som skjer.

12.1 Meny

Helt i starten, når spilleren starter spillet vil vi spille av LetsDance kjenningsmelodien. Denne kommer til å spille helt til spilleren går videre til menyen, der vil vi ha et roligere tema på musikken. Når man beveger hånden over en knapp eller innstilling vil vi ha ett lite ”pop” som skal gi beskjed til spilleren at han er på ett område der han kan utføre en handling(for eksempel trykke på en knapp). Når spilleren forlater dette området skal en lignende lyd, ikke lik, spilles av som gir beskjed at nå har spilleren forlatt området. Hvis spilleren gjør et valg/trykker på en knapp vil en bekreftende lyd spilles av slik at spilleren er klar over at en handling er utført.

Lyder som skal være i menyen:

- Aktivert knapp/funksjon
- Flytter seg inn i/ut av et område som har en funksjon
- Lets Dance kjenningsmelodi
- Rolig meny musikk
- ”Laster inn” musikk
- Avslutter spillet

12.2 Spillet

Når bildet går over fra lasteskjerm til spillet vil vi ha en ”gjør deg klar” lyd og deretter vil musikken begynne å spille. Når spilleren begynner med dansebevegelsene og spillet registrerer om du gjør en; god, veldig god eller perfekt bevegelse vil det komme en stemme som sier dette i tillegg til en unik lyd. Hvis spilleren setter spillet på pause vil det komme en bekreftende lyd på at spillet er blitt satt på pause, det samme skjer når spilleren forsetter spillet etterpå. Oppnår spilleren en prestasjon under spilløkten vil det en spilles av jubel blandet med applaus. Etter hver sang vil det komme opp en poengsum, når dette skjer vil det komme en fancy lyd/melodi. Så lenge man er på bildet der poengsummen vises, vil det spilles en rolig bakgrunnsmusikk. Til slutt når runden avsluttes vil det være en lyd som bekrefter at du har avsluttet runden. I en "runde" vil brukeren danse/bevege seg etter den musikken han/hun har forhåndsbestemt gjennom et brukergrensesnitt.

Lyder som skal være i spillet:

- Lyd for brautførelse (unik)
- Lyd for veldig brautførelse (unik)
- Lyd for perfekt utførelse(unik)
- Pause/forsett
- Jubel med applaus
- Musikk i en spillsesjon
- Når poengsummen dukker opp
- Poengsummen vises (rolig musikk)
- Avslutter runden

Stemmer som skal være i spillet:

- Bra
- Veldig bra
- Perfekt

13.FAQ:

13.1 Hva slags spill er det?

LetsDance er et dansespill ved bruk av kinect.

13.2 Hvorfor lager vi dette spillet?

LetsDance skal hjelpe pasienter ved komplekse funksjonstap etter sykdom eller skade til å få tilbake bevegelighet i armer og ben. Spillet er først og fremst rettet mot pasienter ved Sunnaas sykehus, men kan være morsomt for funksjonsfriske personer også.

13.3 Hva skjer i spillet?

Spilleren står foran kinecten og får opp dansetrinn på skjermen. Hvis spilleren utfører trinnet riktig får han poeng ut ifra hvor bra utførelsen ble gjort. Det vil bli gitt belønninger til spilleren i form av prestasjoner. Det vil også være mulig å legge inn egen musikk.

13.4 Hvor foregår handlingen i spillet?

Handlingen foregår på et dansegulv.

13.5 Hva kontrollerer karakteren i spillet?

Spilleren kontrollerer karakteren. Karakteren/avataren er en ”kopi/klone” av spilleren. Hvis spilleren beveger på seg vil avataren gjør det samme som om spilleren står foran et speil.

13.6 Hva er poenget med spillet?

Poenget med spillet er å forbedre de motoriske evnene sine. Man kan følge progresjonen sin ved hjelp av statistikken som lagres. Selvfølgelig er også en del av poenget med spillet å ha det morsomt.

13.7 Hva skiller dette spillet fra andre spill?

Det som skiller LetsDance fra andre dansespill er at LetsDance er rettet mot rehabilitering av pasienter med funksjonshemninger.

14. Referanser

1. **BowserBikeJustDance. *[Just Dance 3] Dynamite - Taio Cruz*. 2011 16.01.2013]; Available from: http://www.youtube.com/watch?v=VevE4v065sA.**
2. **Wikipedia. *Aaerobics*. 2012 17.01.2013]; Available from: http://en.wikipedia.org/wiki/Aerobics.**
3. **Helsedirektoratet. *Slag*. 17.01.2013]; Available from: http://www.helsedirektoratet.no/folkehelse/fysisk-aktivitet/aktivitetshandboken/terapikapitler/Documents/kapittel-44-slag.pdf.**
4. **stroke-rehab.com. 2010 17.01.2013]; Available from: http://www.stroke-rehab.com/balance-exercises.html.**
5. **Backas, C. *Achievements System (C#)*. 15.01.2013]; Available from: http://www.gameprefabs.com/products/show/58.**
6. **Gargolinski, S. *Progress - a free achievement framework for unity*. 15.01.2013]; Available from: http://www.stevegargolinski.com/progress-a-free-achievement-framework-for-unity/.**
7. **Unity3D.Community. *Creating an Achievement system*. 15.01.2013]; Available from: http://forum.unity3d.com/threads/115824-Creating-an-Achievement-System.**

Utviklingsmetodikk

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		03.12.12	Henrik Olsson Dagfinn Kjærnet
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	MJ, JEH

Innholdsfortegnelse

1. Innledning

I oppgaven vi har tatt på oss for Sunnaas sykehus skal vi lage et produkt, i sammenheng med dette trenger vi en prosjektmetodikk å forholde oss til. En slik metodikk skal hjelpe oss å følge diverse retningslinjer når det gjelder å utføre arbeidsoppgavene.

2. Om stoffet

For å kunne utvikle et produkt blir vi nødt til å følge en utviklingsmetodikk for å kunne holde orden på prosjektet, i dette dokumentet ser vi nærmere på de to kandidatene som vi anser som egnet til vårt prosjekt.

2.1 Kandidater

2.1.1 Scrum

I Scrum har vi en *product backlog* som er en liste med funksjoner som er ønsket til produktet, de forskjellige funksjonene kaller vi for *user stories*. Denne *product backlog* kan vi se på som en slags kravspesifikasjon. Forskjellige *user stories* blir valgt ut til å bli med i produktet for å tilfredsstille kravspesifikasjonen. *Product backlogen* kan bli delt opp i forskjellige *release backlog* for å sette milepæler. Videre deler man opp *release backlogen* opp i *sprinter*. Sprinter har diverse gjøremål, alle gjøremålene blir brutt ned til "små" oppgaver, hvor de forskjellige kategoriseres inn i for eksempel 1,2,4 eller 8 timer osv. Hver sprint blir overvåket gjennom en *burndown chart*. Oppgavene får forskjellige prioriteter, hver oppgave har også antatt arbeidstid som *burndown charten* viser hvor mange timer arbeid som gjenstår på sprinten, her kan man beregne ca. hastighet prosjektet har i gitte tidspunkt.

Daglig har teamet et 15 minutters Scrum møte hvor alle utviklerne forteller hvordan statusen er med de diverse oppgavene, hva som har blitt gjort og hvilke problemer som har oppstått. I dette møtet skal alle deltagerne stå oppreist, for å holde det effektivt.

Om en *user story* ikke når sprint slutt, må prosjektplanen endres slik at man får tilpasset dette.

Scrum er en prosjektmetodikk som er beregnet for team bestående av selvstendige individer, så fort en *user story* er ferdig av en utvikler vil han plukke selv en ny som passer han/hun. Et Scrum team burde ha en *scrummaster* som tar noe ansvar når det gjelder tildeling av oppgaver.

2.1.2 Fordeler ved Scrum

Scrum er en slik agile eller "smidig" metodikk, dette passer oss i dette prosjektet siden vi skal bruke store dele av prosjekt perioden på å observere og legge til nye "små krav", som da vil inngå under de "store" overordnede kravene.

2.1.3 User Stories

User Stories som da vi ser på en form for krav, hvert krav setter vi på noe som kan minne om en gul postit lapp. Hver User Story skal følge en standard, vi har bestemt oss for standarden hvor vi spesifiserer **hvem** som ønsker **hva**, og **hvorfor** på denne måten;
"*Som en <aktør> ønsker jeg/vi å <handling> så <for å kunne>*".
"As an <actor> I want to <action> so that <achievement>".
Vi kommer til å skrive user storiene i prosjektet på norsk.

Actor: dette er eieren av en user story altså da en bruker.

Action: Dette er en funksjonalitet som er ønsket av eieren/brukeren.

Achievement: Dette er hva eieren/brukeren ønsker å oppnå.

2.1.4 Rational Unified Process

RUP er en inkrementell utviklingsprosess som kan tilpasses prosjektets særegne behov, dette fordi det er iterativt, fokuserer på risiko tidlig i utviklingsfasen og ved å integrere fasene. Utviklerne lærer mens prosessen pågår og prosessen kan forbedres fortløpende.

RUP er delt inn i fire faser, disse er *Inception*, *Elaboration*, *Construction* og *Transition*.

Inception

- Få en omtrentlig oversikt over kostnader.
- Finne grensene for prosjektet.
- Finne og definere krav.
- Diskutere risiko og design samt planlegging.

Elaboration

- Definere den endelige arkitekturen.
- Definere visjoner for prosjektet.
- Lage en detaljert plan for konstruksjonsfasen.
- Demonstrere at den valgte arkitekturen vil virke, gitt tids og kostnads begrensninger.
-

Construction

- Utviklingen av produktet.

Transition

- Testing for å finne feil og mangler som må utbredes.
- Feilretting.
- Forsikre om at produktet er godtatt av alle involverte og at det oppfyller spesifikasjonen.

Fordeler ved Rational Unified Process

- Forhåndsbestemt/Forutsigbart.
- Risiko-drevet.
- Arkitektur-drevet.

2.2 Verktøy

Som et verktøy har vi bestemt oss for å benytte oss av en applikasjon som heter "Ontime" det er utviklet av det Amerikanske selskapet Axosoft. Applikasjonen er webbasert og er utstyrt med et brukergrensesnitt som forenkler vår Scrum hverdag. Hvert medlem av utviklerteamet har en brukerkonto som gir de muligheter til å ta på seg userstories fra backlogen, disse userstoriene får da en status, om de er nye oppgaver, om de er underutvikling eller er ferdig utført. OnTime gir oss også et burndown chart ut i fra timene vi har ført på de forskjellige userstoriene.

3. Konklusjon

Gruppen har bestemt seg for prosjektmodell; de forskjellige prosjektmodellene har hver sine kvaliteter, de vi så som mest aktuelle var Rational Unified Process og Scrum. Disse to er forskjellige når det gjelder lederskap og tildeling av oppgaver. Rational Unified Process er forhåndsbestemt, risikodrevet og arkitekturdrevet, Det krever Rational Unified Process nøye planlegging. Siden vi ser på oss selv som unge, spreke og selvstendige så vil Scrum være det som passer oss best. Vi har tro på at alle individene i utviklerteamet vil ta på seg like omfattende oppgaver, og vil gjøre en god nok dokumentert jobb slik at alt kan interfaces [1, 2].

4. Ordliste

Scrum:
En agil utviklingsmetodikk.

Backlog:
En liste med krav, altså en form for krav spesifikasjon.

User story:
En funksjon som er ønsket til det ferdige produktet.

Burndown chart:
En graf som viser brukt tid i forhold til oppgaver.

Sprint:
En tidsperiode hvor diverse user stories skal bli fullført.

Scrummøte:
Et møte hvor alle står oppreist og som foregår hurtig, her nevner hvert medlem av utvikler teamet hvordan statusen er med de forskjellige oppgavene de jobber med.

Scrum Master:
En som styrer scrum møtene, user stories og burndown chart.

Rational Unified Process:
En utviklingsmetodikk.

Interface:
Grensesnitt.

Actor:
Dette er eieren av en user story altså da en bruker.

Action:
Dette er en funksjonalitet som er ønsket av eieren/brukeren.

Achievement:
Dette er hva eieren/brukeren ønsker å oppnå.

5. Referanser

1. AxoSoft. *Agile Scrum in Under 10 Minutes*. 2012 11.12.2012]; Available from: http://www.youtube.com/watch?v=XU0llRltyFM.
2. breathingtech. *Writing user stories for agile (Scrum) projects*. 2009 11.12.2012]; Available from: http://breathingtech.com/2009/writing-user-stories-for-agile-scrum-projects/.

Teknologidokumenter

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Teknologidokument: ZigFu

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet dokument	05.03.13	Marcus Jernberg
2.0	Info om Scripts	11.03.13	Marcus Jernberg
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	AO og DK

Innholdsfortegnelse

1. Zigfu

"Zigfu Development Kit" er den letteste måten å lage kryssplatform, motion-kontrollerte applikasjoner med Kinect i HTML5/Javascript, Unity3D og Flash. Applikasjoner er laget med ZDK er portable mellom alle operativsystemer, web browsers, computer vision middlewares, og 3D sensorer. [1]

Zigfu har laget en pakke med et installeringsscript som inneholder OpenNI, Sensor Kinect og NITE. Zigfu har den unike posisjonen av å være skrevet av to Eks-Primesense ansatte og to andre utviklere. Primesense var selskapet som skapte NITE. Som et resultat av det er Zigfu det mest stabile scriptet for OpenNI, og har fordelen av å være åpen kildekode.

1.1 ZDK for Unity3D

ZDK støtter både Unity native og WebPlayer systemer for motion-kontrollerte applikasjoner. I Unity pakken som kommer med ZDK'en inneholder flere fullt funksjonelle eksempler, slik at man slipper å måtte starte fra scratch når man skal utvikle en nytt spill. ZDK'en for Unity3D fungerer både på PC og MAC med OpenNI / NITE og Microsoft Kinect SDK. [2]

1.2 Hvordan vi bruker Zigfu

I forbindelse med utviklingen av spillet ønsker vi å koble Kinecten opp mot en avatar som "kopierer" bevegelsene utført av brukeren. For å gjøre dette kreves det en hel del avanserte scripts. Så for å spare oss for unødvendig tid og hodebry tar vi disse scriptene som utgangspunkt for å løse utfordringen med å koble kinecten opp mot avataren.

Zigfu har en trial versjon som er gratis og har de samme mulighetene som fullversjonen. Trial versjonen er ikke for kommersielt bruk, men vi annser heller ikke spillet som kommersielt i det stadiet vi befinner oss på. Zigfu har også en logo som kommer opp på spillskjermen vår når spillet kjøres.

Vi kommer til å bruke 4 scripts fra Zigfu, disse er enkelt forklart i punktet under.

1.3 De forskjellige scriptene

Zig – Overordnet script
ZigDepthViewer – Dette er scriptet som oppfatter spilleren, og viser bilde av spilleren i høyre hjørne av skjermen
ZigEngageSingleUser – Kobler spilleren opp mot avataren
ZigSkeleton – Binder leddene(armer, ben osv) til modellen opp mot spillerens

2. Ordliste

OpenNI – "er en bransje-ledende, non-profit organisasjon fokusert på sertifisering og bedre interoperabilitet av naturlige brukergrensesnitt og organsisk brukergrensesnitt for naturlige samspill enheter, programmer som bruker disse enhetene og mellomvare som muliggjør tilgang og bruk av slike enheter" – *Wikipedia*
OpenNI er hele rammeverket for utvikling av applikasjoner ved hjelp av kropps interaksjon. Det inneholder en åpen kildekode API som kan brukes for å støtte forskjellige kameraer (som Kinect og Xtion) og algoritmer (for kropps deteksjon osv).
ZDK – Zigfu Development Kit
NITE - Er et biblotek på toppen av OpenNI rammeverket. Det fokuserer mer på funksjoner som "Push", sveip, vinke, sirkel osv. NITE's eneste funksjon er skeleton tracking.

3. Referanser

[1] Zigfu - http://zigfu.com/en/zdk/overview/
[2] Zigfu for Unity - http://zigfu.com/en/zdk/unity3d/

Teknologidokument: Unity

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		17.12.12	Dagfinn Kjærnet
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	JEH, MJ

Innholdsfortegnelse

1. Innledning

Siden vi skal utvikle et spill så trenger vi et rammeverk for grafikk og fysikk siden dette er ting vi ikke ønsker å bruke tid på selv og det finnes mange gode alternativer for dette allerede, både i form av grafikkmotorer og fysikkmotorer og ikke minst spillmotorer som er en komplett løsning med både grafikk og fysikkløsninger. Det var også viktig å ha mulighet for støtte av Kinect. Valget falt på Unity.

2. Om Unity

Unity er en komplett spillmotor som inneholder både grafikk/rendering motor og støtter Nvidia's fysikkmotor kalt PhysX[1]. Den støtter også scripting via Mono[2] som er en åpen kildekode implementasjon av .NET rammeverket og vi kan dermed benytte oss av språket C#(se eget teknologi dokument).

Unity har ikke det man kan kalle "state of the art" grafikk, men er fremdeles svært populært blant uavhengige utviklerne. Mye av grunnen til dette er nok den rimelige prisen på lisenser, og ikke minst det at man får en tilnærmet fullverdig versjon av Unity gratis så lenge man ikke skal bruke det til

kommersielt bruk[3]. På denne måten blir det mye bruk blant personer som først og fremst ønsker å lære seg spill programmering, og når disse da ønsker å selge produkter velger de å bruke den spillmotoren de allerede kjenner godt til.

Av eksempler på spill utviklet i Unity har vi blant annet "Bad Piggies" og "Dead Trigger"[4].

2.1 Unity og enhetstesting

Unity har et eget innebygget rammeverk for enhetstesting, dette kalles UUnit og er inspirert av NUnit som er et populært enhetstest rammeverk for .NET språk[5].

2.2 Fordeler og ulemper ved Unity

Av fordeler kan man nevne at siden Unity blir svært mye brukt av uavhengige utviklere har det dannet seg et stort utviklermiljø i form av forumer og nettsamfunn der man deler kodeeksempler, guider og hjelper hverandre. Dette er noe vi tidligere har hatt et stort utbytte av og vi ønsker å benytte oss av dette også i forbindelse med hovedprosjektet. Unity har ikke innebygd støtte for Kinect, men det finnes en del informasjon tilgjengelig om hvordan man bruker Kinect i Unity ved hjelp av tredjepartsverktøyer[6].

Til vårt formål er dessuten gratis versjonen av Unity tilstrekkelig og vi sparer potensielt svært mye på dette. Vi har heller ikke noe stort behov når det kommer til grafikk og Unity yter mer enn bra nok på dette feltet.

Unity har også noe de kaller "Asset Store", dette er en markedsplass der man enkelt kan hente ned diverse resurser, som for eksempel 3D modeller, lyder, musikk og brukergrensesnitt. Mange av disse er også gratis.

Av ulemper så er det som tidligere nevnt en litt utdatert grafikkmotor, og litt begrenset hva gjelder valg av språk å scripte i. Unity er også en noe mer lukket løsning enn noen av alternativene, så man har liten frihetsgrad til å velge alternative løsninger enn det som tilbys i standardpakken og man må derfor godta de begrensninger som finnes.

En av de større ulempene er mangel på kildekontroll i gratisutgaven[7], men dette kan man skaffe ved hjelp av andre programmer. Da blir ikke dette integrert i Unity men heller en egen tredjepartsløsning, eksempler på dette vil være SVN eller Git.

3. Konklusjon

Vi velger Unity fordi det tilfredsstiller våre krav og det er dessuten designet for å gjøre utviklingsprosessen lettere for utviklerne. For å løse problemet med kildekontroll tenker vi å benytte Git(Se egen dokumentasjon). Dette er forøvrig ikke en perfekt løsning da det ikke er tilpasset Unity sin mappestruktur og den sliter med å synkronisere alt av biblioteker og lignende, se risikodokument for risikovurdering.

Dessuten har alle studentene på gruppen samt ekstern veileder også erfaring med Unity fra tidligere og dette var også noe vi tok med i vurderingen vår.

3.1 Ordliste

Åpen kildekode – Kildekoden er tilgjengelig for allmenheten.
Spillmotor – Et rammeverk for å lage spill, tar seg av det meste av kjernefunksjoner som fysikk, grafikk, lyd og lignende[8].
Kildekontroll (også kjent som revisjonskontroll) – Administrasjon og kontrollering av endringer I kildekoden[9].

3.2 Referanser

1. **Wikipedia. *Unity (game engine)*. 19.12.2012]; Available from: http://en.wikipedia.org/wiki/Unity_(game_engine).**
2. **Unity. *Unity scripting*. 19.12.2012]; Available from: http://unity3d.com/unity/workflow/scripting.**
3. **Unity. *License Comparisons*. 19.12.2012]; Available from: http://unity3d.com/unity/licenses.**
4. **Unity. *Game List*. 19.12.2012]; Available from: http://unity3d.com/gallery/made-with-unity/game-list.**
5. **Unity3D.Wiki. *UUnit*. 20.12.2012]; Available from: http://wiki.unity3d.com/index.php?title=UUnit.**
6. **ETC.Internal.Unity3D.Wiki. *Microsoft Kinect - Microsoft SDK*. 20.12.2012]; Available from: http://wiki.etc.cmu.edu/unity3d/index.php/Microsoft_Kinect_-_Microsoft_SDK.**
7. **StackExchangeCommunity. *What are the pro/cons of Unity3D as a choice to make games?* 19.12.2012]; Available from: http://gamedev.stackexchange.com/questions/8118/what-are-the-pro-cons-of-unity3d-as-a-choice-to-make-games.**
8. **Wikipedia. *Game engine*. 19.12.2012]; Available from: http://en.wikipedia.org/wiki/Game_engine.**
9. **Wikipedia. *Revision control*. 19.12.2012]; Available from: http://en.wikipedia.org/wiki/Source_control.**

Teknologidokument: SQLite

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		14.03.13	Dagfinn Kjærnet
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	AO og JEH

Innholdsfortegnelse

1. Innledning

Dette dokumentet er en utredning av dokumentasjonen om SQLite som vi i utgangspunktet skrev om i dokumentasjonen for C#. Siterer derfor her originalbeskrivelsen:

"Vi ønsker å benytte en database for lagring av brukerdata, vi kommer til å benytte oss av SQLite. SQLite er et lite C-bibliotek som implementerer en frittstående SQL-databasemotor. I motsetning til andre database håndteringssystemer er SQLite ikke en separat prosess som er tilgjengelig fra klientprogrammet, men en integrert del av det."[1]

2. Om SQLite

SQLite er et software bibliotek som implementerer en selvstendig og server løs SQL-databasemotor. Det er den mest utbredte SQL-databasemotoren i verden, og kildekoden er offentlig tilgjengelig.

SQLite er et kompakt bibliotek. Med alle funksjoner aktivert, kan bibliotekstørrelsen være mindre enn 350KiB. SQLite kan også settes opp for å kjøres i minimal stakkplass (4KiB) og svært liten heap (100KiB), noe som gjør den til en populær databasemotor på enheter med begrenset minne som mobiltelefoner, PDA-er og MP3-spillere. SQLite går vanligvis raskere jo mer minne du gir den, likevel er ytelsen vanligvis ganske bra selv i miljøer med lite minne[2].

2.1 Ulemper ved SQLite

SQLite støtter ikke kryptering "rett ut av boksen", dette krever tilleggs implementasjoner. Flere av disse er forøvrig enten svært dyre å anskaffe lisens til mens andre er relativt ustabile[3].

Syntaxen er litt forskjellig fra hva vi er vant med fra mySQL, og den har ikke fullt så mange funksjoner.

Kan være tungvint å bruke og vanskelig å feilsøke, og det finnes få gode databasehåndteringsverktøyer som forenkler prosessen med å opprette og vedlikeholde databasen. Til sammenligning har mySQL et svært godt verktøy for dette som kalles dbDesigner[4]. Til SQLite har vi valgt å bruke en plugin til firefox kalt "SQLite Manager" for å generere databasen slik vi ønsker å ha den, mens vi har brukt dbDesigner i designprosessen.

2.2 Fordeler ved SQLite

For brukeren vil det være lettere å installere og bruke spillet da databasen blir en innebygget funksjonalitet i spillet, og ikke avhenger av en ekstern server.

3. Konklusjon

Vi velger å bruke SQLite på tross av ulempene fordi vi ønsker å etterfølge spillets krav om å være lett å ta i bruk. Det vil nok på mange måter være en fordel å bruke f.eks mySQL slik at databasen blir lettere å designe, opprette og vedlikeholde, men dette vil innebære at brukeren må installere mySQL og sette opp dette selv.

Vi kommer forøvrig til å ta opp med Sunnaas hvorvidt databasen bør krypteres og jobbe mot en løsning som tilfredsstiller de kravene Sunnaas måtte ha mtp personvern, men for vår første prototype trenger vi bare en enkel databaseimplementasjon og siden vi allerede har brukt en del tid på å sette opp SQLite velger vi å fortsette med denne enn så lenge. Alternativene vi i så fall kommer til å se på er å implementere en av krypteringstilleggene til SQLite som er tilgjengelige, eller å bytte til en løsning som benytter seg av MySQL. Det skal også være innebygget støtte for kryptering i .net rammeverket[5, 6], men denne har foreløpig vist seg å være litt trøblete å aktivere i Unity.

4. Ordliste

Syntax – et sett med regler som definerer de kombinasjonene av symboler som blir betraktet som korrekt strukturerte programmer i det gjeldende programmeringsspråket[7].
Database – En strukturert samling av data.

5. Referanser

1. **Kjærnet, D., *Teknologidokumentasjon: C#.* LetsDance, 2012.**
2. **SQLite. *About SQLite*. 14.03.2013]; Available from: http://www.sqlite.org/about.html.**
3. **StackOverflow. *SQLite with encryption/password protection*. 14.03.2013]; Available from: http://stackoverflow.com/questions/5669905/sqlite-with-encryption-password-protection.**
4. **fabforce. *General Information - What is DBDesigner 4?* 14.03.13]; Available from: http://www.fabforce.net/dbdesigner4/.**
5. **Simpson, R. *Encrypting, decrypting and attaching to encrypted databases*. 14.03.2013]; Available from: http://sqlite.phxsoftware.com/forums/t/130.aspx.**
6. **StackOverflow. *Password Protect a SQLite DB. Is it possible?* 14.03.2013]; Available from: http://stackoverflow.com/questions/1381264/password-protect-a-sqlite-db-is-it-possible.**
7. **Wikipedia. *Syntax (programming languages)*. 13.12.2012]; Available from: http://en.wikipedia.org/wiki/Syntax_(programming_languages).**

Teknologidokument: NGUI (Next-Gen UI kit)

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

....

....

....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		20.02.2013	Jan Erik
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	AO og DK

Innholdsfortegnelse

1. Innledning

Siden vi skal utvikle et spill hvor det skal være et enkelt menysystem, samtidig som den skal være ren designmessig er vi nødt til å se på mulige GUI kit siden standard GUI kit i Unity er noe vrient å bruke, samtidig som den ikke kan forandres på i like stor grad. I den forbindelse ønsker vi å se nærmere på NGUI: Next-Gen UI kit for å kunne ha et godt grunnlag til å ta en beslutning på om det skal brukes. NGUI er en GUI plugin til Unity som er utviklet av Tasharen Entertainment.

2. Skribent

Jan Erik Helskog(JEH)

3. Om NGUI: Next-Gen UI kit

NGUI[1] er et veldig stødig og kraftig UI og varslings rammeverk for Unity, da både for gratis og pro versjonen. NGUI er skrevet i C# noe som passer prosjektet bra. Dessuten følger den KISS prinsippet, dette er med på å gjøre at NGUI er enkelt å bruke for utviklere.

3.1. Andre ting som er verdt å nevne

Det å jobbe med NGUI er akkurat som å jobbe med Unity. Ved å bruke Widget Tool kan man kjapt lage en template widget, eller man kan lage sin egen fra enkle komponenter. Full støtte for copy/paste slik at man unngår å lage det samme om igjen. Det å lagre sin egendefinerte widget som en prefab er gjort på få klikk. Scripting av forskjellige GUI elementer, altså kode som kan gjøre at knapper kan bevege seg og bytte farge når de blir trykket. Til dette følger det med scripts som gjør at widgetene blir knapper, input field, bytte farge, spille av lyd, starte animasjoner og mye mer.

4. Fordeler og ulemper

4.1. Fordeler

- Trenger ikke å trykke på play for å se resultater
- Det du ser i sceen view er det du ser i spillet
- Fleksibelt event system
- Kan lage complex UIs med kun ett draw kall
- Komponent basert, alle widget kan få den oppførselen du selv vil
- Støtte for avskårede paneler, da med hard eller myke kanter
- Støtte for forskjellig lyssetning, refleksjon, mapping og mye mer
- Bygd inn støtte for keyboard og kontroller støtte
- Enkel, kort og optimalisert C# kode
- Ikke noen DLLs eller eksterne resurser
- Gratis til utdanning bruk, så lenge man ikke modifiserer NGUI mappen utover å fjerne overflødige filer og gjør det klart at det er laget av Tasharen Entertainment[2]
- Gode guider[3] fra brukere og Tasharen Entertainment, samt vel dokumentert wiki

4.2. Ytelse

NGUI er en veldig lettvekter av en plugin. Ved å bruke NGUI vil man ikke oppleve noe større krav til ytelse fra utviklermaskin eller spill maskin. Dette kommer mye av at scriptene som brukes er godt skrevet og optimaliserte for å gi en god ytelse.

4.3. Ulemper

- Gratis versjonen gjelder kun for ikke kommersielt bruk, ved kommersielt bruk kreves det kjøp av lisens til 95 dollar

5. Konklusjon

Etter nøye vurdering og testing av NGUI har vi kommet frem til at vi vil bruke denne pluginen som verktøy til utvikling av meny systemet i spillet. Da det bare er fordeler å bruke denne pluginen.

6. Ordliste

- **GUI kit –** Graphical User Interface verktøy
- **KISS prinsippet** - Keep it simple, stupid.

7. Referanser

1. ArenMook. *NGUI: Next-Gen UI kit*. 2011 21.02.2013]; Available from: http://www.tasharen.com/?page_id=140.
2. ArenMook. *About*. 2011; Available from: http://www.tasharen.com/?page_id=8.
3. ArenMook. *NGUI: Documentation*. 2011; Available from: http://www.tasharen.com/?page_id=197.

Teknologidokument: Kinect

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		11.12.12	Zana Ali
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	JEH, MJ

Innholdsfortegnelse

1. Kinect

Kinect er en bevegelsessenor enhet laget av Microsoft for Xbox 360 spillkonsoll og Windows PC'er. Den baserer seg rundt et webkamera som gjør det mulig for brukerne å styre og samhandle med Xbox'en uten bruken av noen fysiske enheter som en spillkontroll for kommunikasjon. Det er en kombinasjon av kameraer, mikrofoner og programvare som gjør deg til spillkontrolleren. Dette skjer gjennom et brukergrensesnitt med sensorer som fanger opp talekommandoer og bevegelser. Kinect "ser" hver bevegelse av kroppen din og reproduserer det på skjermen mens du spiller. Den kjenner også igjen ansiktet, og stemmen din slik at den kan gjenkjenne deg i rommet og vite hvem du er selv om det skulle være flere personer foran Kinect sensorene. Kinect ble lansert i november 2012 og solgte svært mange enheter ved lanseringen [1, 2].

1.1 Kinect vs Playstation Move og Wii Motion-Control Showdown

Alle de tre store spill-konsollene tilbyr en form for motion-control.
Kinect har en av fordelene at den kan fange opp bevegelsene dine uten en fysisk kontroller. Siden systemet er basert på "kun" kamera", krever det en del plass.

Playstation move kombinerer et videokamera med en fysisk kontroll og krysser teknologien mellom Kinect og Ninendo Wii. Denne er billigst av de tre motion-control enhetene. Ulempen er at den krever en lesepenn per spiller.
Med **Nintedo Wii** skjer motion-control rundt en fjernkontroll. Et akselerometer sporer bevegelse, mens en IR-sensor overvåker posisjonering av lysene som slippes ut av sensorbaren.
Nintendo Wii er billig og har et stort spillbibliotek. Den er dog den minst nøyaktige motion-sensoren av de tre systemene og er heller ikke high-def [3].

1.2 Kinect Software Development Kit (SDK)

Kinect Windows SDK åpner muligheten for ubegrensede muligheter tilbydd av Kinect-teknologi. SDK'en har blitt oppdatert til å støtte enda flere funksjoner og gir utviklere mer fleksibilitet og bedre kontroll over utviklingen av hver applikasjon. Den støtter applikasjoner bygget med C++, C# eller Visual Basic. De siste oppdateringene av Kinect for Windows SDK og Developer Toolkit eksponerer flere funksjoner i Kinect-sensoren, forbedrer utviklings-effektivitet, og legger til operativsystem og annen støtte verktøy.

De nye oppdateringene gir støtte for virtuelle maskiner, støtte for Windows 8 og Visual Studio 2012 blant annet [4, 5].

1.3 Hvorfor Kinect?

En av de definitivt største fordelene med Kinect er den fysiske aktiviteten og ubegrensede bevegelsene du kan gjøre med kroppen din. Det gir en mer virkelighetsfølelse å gjøre bevegelser som å sparke eller hoppe istedenfor å trykke på noen knapper på håndkontrollen. Navnet Kinect er inspirert av ordene *kinetic* som betyr å være i bevgelse og "koble", som betyr at den kobler deg til dine venner og underholdningen du liker [1, 2].

Siden alle spillene baserer seg på dine egne kroppsbevegelser føles alt en gjør med Kinect veldig intuitivt. Det er lett å forstå hvordan ting fungerer umiddelbart. Kinect byr også på underholdning som har en lav terskel for hvem som kan delta [6].

2. Referanser

1. Wikipedia. *Kinect*. 2012 02/12/2012]; Available from: http://en.wikipedia.org/wiki/Kinect.
2. electronics.howstuffworks. *How Microsoft Kinect Works*. 02/12/2012]; Available from: http://electronics.howstuffworks.com/microsoft-kinect.htm.
3. pcmag. *Kinect vs. PlayStation Move vs. Wii: Motion-Control Showdown*. 2010 13/12/2012]; Available from: http://www.pcmag.com/article2/0,2817,2372244,00.asp.
4. Microsoft. *What's new*. 2012 20.12.2012]; Available from: http://www.microsoft.com/en-us/kinectforwindows/develop/new.aspx.
5. Microsoft. *Kinect for Windows Features*. 2012 20.12.2012]; Available from: http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx.
6. Filmpolitiet. *Kinect er i hus!* 2011 10/12/2012]; Available from: http://p3.no/filmpolitiet/2010/11/kinect-er-i-hus/.

Teknologidokument: Google Sites

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	03.12.2012	JEH
2.0	Finpuss	10.12.2012	MJ
3.0	Finpuss	03.01.2013	ZA
4.0			
5.0			
6.0			

Kontrollert av:	MJ, ZA

Innholdsfortegnelse

1. Innledning

Vi ønsker å se nærmere på denne teknologien. Dette fordi vi ønsker å benytte oss av Google Sites til vår webside.

2. Skribent

Jan Erik Helskog(JEH).

3. Hva er Google Sites

Google Sites er et strukturert websideverktøy som er levert av Google, som er en del av deres produktivitetsserie. Google Sites er spesiallaget for alle som vil lage en teamorientert side, der alle har muligheten til å samarbeide og dele filer.

4. Eksempel på bruk av Google Sites

Det å bruke Google Sites er enkelt. Det eneste man trenger er å logge seg inn på https://sites.google.com/ men en googlekonto. Deretter kan man enten gå inn på en allerede opprettet side, eller man kan trykke på opprett og starte en ny side. Da vil man være nødt til å gi nettstedet et navn og velge om man vil bruke en mal, eller starte med en blank side. GUIen for å redigere på nettstedet er meget enkelt og alle funksjonene er lette å sette opp.

5. Andre ting som er verdt å nevne

Google Sites er gratis å bruke. Alt på nettstedet lagres i nettskyen og det vil derfor være mulig å nå nettstedet på alle mulige enheter som kan være koblet til internett. Andre Google Tools er integrert, så det vil være enkelte å dele kalendere, videoer, bilder, presentasjoner og dokumenter. Alle endringer på nettstedet blir lagret så det er enkelt å ha en oversiktlig revisjons kontroll.

6. Fordeler og ulemper

6.1 Fordeler

- Det er gratis å bruke
- Lett å lage. Klarer man å legge inn tekst er man på god vei
- Det er lett å samarbeide med andre om å redigere og legge ut nytt innhold
- Integrert med andre verktøy fra Google
- Nettstedet kan bli funnet ved googlesøk dersom man tillater det
- God revisjon historikk. Hva er blitt endret, av hvem og når
- Man bestemmer selv hva slags rettigheter brukere skal ha på nettstedet
- Nettstedet er lagret på Google's trygge servere
- 100MB gratis online lagringsplass

6.2 Ytelse

Dette er en veldig lettvektig måte å jobbe med et nettsted. Dette fordi alt foregår som en webapplikasjon i nettleseren. Derfor vil man ikke ha noen store problemer med å bruke verktøyet uansett enhet.

6.3 Ulemper

For å kunne jobbe med siden er man nødt til å ha tilgang til internett, da Google Sites ikke støtter lokalt arbeid [1, 2].

7. Referanser

1. Fran, M. *Mary Fran's Getting Started with Google Sites*. 2012; Available from: https://sites.google.com/site/mflynchsites/.
2. Google. *Google Sites Overview*. 2011; Available from: http://www.google.com/sites/help/intl/en/overview.html.

Teknologidokument: Git

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		21.12.2012	Are Oven
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA, MJ

Innholdsfortegnelse

1. Innledning

Siden vi er seks personer på gruppa og skal jobbe med ett prosjekt, er vi nødt til å ha en måte å sette koden sammen uten at det skaper problemer. Det er dette vi skal bruke Git til. Det er også en gratis åpen kildekode.

2. Git

Vi skal bruke Git til å sjekke kodebitene til hverandre. En av de viktigste egenskapene til Git er at når vi lager eller forandrer på ett stykke kode blir ikke den gamle koden slettet, men istedenfor blir den lagret som en eldre versjon. Dette gir oss muligheten til å forandre på ting uten at kode går tapt i prosessen. Git bruker en lagringsmetode som gjør at man ikke kan forandre på filer, datoer osv. pga dette vil vi alltid ha full kontroll over hvem som har gjort hva og når ting ble gjort. Det er også et distribuert system som gjør at man alltid har tilgang til dataene. På grunn av dette systemet, jobber man ikke med filene direkte, men man jobber med en kopi. En av de største fordelene med dette er at man i praksis har sikkerhetskopi av alle dataene på alle maskinene som er koblet til systemet. Git er et lite og raskt program sammenlignet med lignende programmer. I motsetning til lignede

programmer jobber Git lokalt og kan derfor utføre oppgaver mye raskere enn hvis all dataen må sendes over internett.[1]

3. Konklusjon

Git er et lite og veldig rask, mye på grunn av at det jobber lokalt. Som en følge av dette får vi praksis en backup av all dataen hvis noe skulle krasje. Det er også gratis for oss å bruke som kommer godt med.

4. Referanser

1. Git. *about - Git*. 21.12.2012]; Available from: http://git-scm.com/about/small-and-fast.

Teknologidokument: Dropbox

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		03.01.13	Marcus Jernberg
2.0	Korektur	03.01.13	JEH
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA, JEH

Innholdsfortegnelse

1. Innledning

Siden vi kommer til å ha mange dokumenter, bilder, filer etc i prosjektet er det en fordel å ha et sted å lagre disse hvor gruppen, veiledere og sensorer har tilgang. Vi har valg å bruke Dropbox til dette siden vi har god kjenskap til dette programmet fra før.

2. Om Dropbox

Dropbox er et program som gir deg mulighet til å lagre dokumenter og filer i en såkalt nettsky. Dropbox gir brukerne mulighet til å opprette spesielle mapper på pcen sin, som blir synkronisert slik at det ser ut som man har den samme mappen på alle PCene sine. Disse mappene kan deles med andre brukere. Noe som gjør det enkelt å dele dokumenter og filer med hverandre. [1]

2.1 Fordeler og ulemper ved Dropbox

Fordeler med Dropbox er at det er lett å sette opp på PCen, det er enkelt å invitere hverandre til nye mapper, og det er enkelt å bruke. Som studenter har vi få utvidet lagringsplassen med 15GB på grunn av SpaceRace. [2]

Ulemper med Dropbox er at det har en relativ høy pris pr GB lagringsplass. Det kan også være problematisk når flere jobber i samme dokument.

3 Konklusjon

Vi bruker dropbox fordi alle har tidligere erfaring med det. Det blir mange dokumenter etterhvert, og da tilbyr dropbox en oversiktelig måte å lagre dokumentene på.

4 Ordliste

GB – Gigabyte

5 Referanser

1. **Wikipedia. *Dropbox*. 2013 03.01.13]; Available from: http://en.wikipedia.org/wiki/Dropbox_(service).**
2. **SpaceRace, D. *Dropbox*. 2013 03.01.13]; Available from: https://www.dropbox.com/spacerace.**

Teknologidokument: Doxygen

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		21.12.12	Zana Ali
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	MJ, AO, JEH

Innholdsfortegnelse

1. Doxygen

Dokumentasjon er essensielt i systemutvikling. Kildekode skal også bli dokumentert. Doxygen er et dokumentasjonsverktøy for en rekke programmeringsspråk som C++, Java, C#, PHP m.fl. Doxygen genererer programvare-referanse dokumentasjon. Dokumentasjonen er skrevet i koden, dermed er det relativt enkelt å holde seg oppdatert. Doxygen kan krysse referansedokumentasjon og kode, slik at leseren av dokumentet lett kan se selve koden[1, 2].

1.1 Design

I likhet med Javadoc, trekker Doxygen dokumentasjon fra kildefil- kommentarer. I tillegg til Javadoc syntaks, støtter Doxygen dokumentasjonstag brukt i Qt toolkit og kan generere output i HTML, samt i Portable Document Format(PDF) og noen flere formater[2].

1.2 Funksjoner

- Doxygen kan generere et online dokumentbrowser(i HTML) og offline referanse manuel fra et sett med dokumenterte kildefiler. Det er også støtte for genrerering av output i RTF(MS-Word). Dokumentasjonen er hentet direkte fra kildene, noe som gjør det lettere å holde den i samsvar med kildekoden

- Du kan konfiguere Doxygen til å trekke kodestrukturen fra udokumenterte kildefiler. Dette er svært nyttig for å raskt finne veien i store kilde distribusjoner. Du kan også visualisere forholdet mellom de ulike elementene ved hjelp av blant annet avhengighetsgrafer, arv diagrammer og samarbeids diagrammer, som alle kan genereres automatisk
- Du kan også bruke Doxygen til å lage normal dokumentasjon[1]

1.3 Eksempler

```
/**
<A short one line description>

<Longer description>
<May span multiple lines or paragraphs as needed>

@param  Description of method's or function's input parameter
@param  ...
@return Description of the return value
*/
```

```
/**
 * Constructor that sets the time to a given value.
 */
Time (int timemillis ///< Number of milliseconds passed since Jan 1, 1970.
      )
{
    // the code
}
```

[2]

1.4 Fordeler med Doxygen

Doxygen er gratis som genererer en dokumentasjon av god kvalitet. Den har utvidelser som tillater deg å enkelt integrere det inn i Visual Studio. Man kan oppleve at det tar veldig lang tid å dokumentere egne kilder, andres dokumentasjon er ikke god, eller oppdaterte. Doxygen er løsningen på de nevnte problemene, og gjør livet enklere når det gjelder dokumentasjon. Dette er grunnene til at vi har valgt å bruke Doxygen[3].

2. Referanser

1. **Stack.nl. *Doxygen manual*. 2012 19.12.12]; Available from: http://www.stack.nl/~dimitri/doxygen/.**
2. **Wikipedia. *Doxygen*. 2012 19.12.12]; Available from: http://en.wikipedia.org/wiki/Doxygen.**

3. drdobbs. *Better Docs with Doxygen*. 2003 03.01.13]; Available from: http://www.drdobbs.com/better-docs-with-doxygen/184416678.

Teknologidokument: C#

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		11.12.12	Dagfinn Kjærnet
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA, MJ

Innholdsfortegnelse

1. Innledning

Vi må velge et utviklingsspråk vi kan bruke for å lage de funksjonene vi ønsker at spillet skal ha. Siden vi allerede har bestemt oss for å bruke spillmotoren Unity så er valgalternativene noe begrenset fordi den kun støtter en håndfull språk og de tilgjengelige alternativene som var aktuelle for oss var JavaScript, C++ og C#. Valget falt da på C#.

2. Om C#

C# er et objekt orientert programmeringsspråk som utvikles av Microsoft. Språket er basert på C++ og Java og er designet for å balansere styrke (fra C++) og rask utvikling (Java)[1, 2].
C# kode kompileres ikke direkte til maskinkode men heller til et mellom format som kan kjøres på rammeverket ".NET". Det vil si at det benytter seg av et mellomlag mellom applikasjon og operativsystem i likhet med det Java gjør.

SQLite

Vi ønsker å benytte en database for lagring av brukerdata, vi kommer til å benytte oss av SQLite. SQLite er et lite C-bibliotek som implementerer en frittstående SQL-databasemotor. I motsetning til andre database håndteringssystemer er SQLite ikke en separat prosess som er tilgjengelig fra klientprogrammet, men en integrert del av det[3, 4]

2.1 Ulemper ved C#

Selv om måten C# opererer på kan minne mye om Java så har ikke C# samme kryssplattformstøtte[3]. Kun plattformer som støtter ".Net" rammeverket kan kjøre programmer som er utviklet i C#, dette er i stor grad kun Microsofts egne plattformer, først og fremst Microsofts Windows operativsystem.

2.2 Fordeler ved C#

C# ligner mye på Java hva gjelder syntax og tilgjengelige biblioteker. Vi syntes det er lettere å kode i enn f.eks C++ og de fleste på gruppen har dessuten erfaring med C# fra tidligere prosjekter.

C# har i likhet med Java en egen garbage collector, som minker sjansen for minnelekkasjer[1].

3. Konklusjon

Vi velger å bruke C# fordi det er et språk vi tror vil passe prosjektet bra, dessuten har de fleste av studentene på gruppen brukt C# i andre sammenhenger, og det ligner også mye på Java som er det språket vi har brukt mest på høgskolen. Eksternveileder har også mest erfaring med C# og foretrakk at vi bruker det.

4. Ordliste

Syntax – et sett med regler som definerer de kombinasjonene av symboler som blir betraktet som korrekt strukturerte programmer i det gjeldende programmeringsspråket[6].
Garbagecollector – Betegnelse for en prosess som automatiserer frigjøringen av minneresurser etter bruk[7].
Java – Programmeringsspråk utviklet av Sun.
Rammeverk – En abstraksjon i programvare som tilbyr generiske funksjoner[8].
Database – En strukturert samling av data.

5. Referanser

1. **Wikpedia. *C# (Progamming language)*. 13.12.2012]; Available from: http://en.wikipedia.org/wiki/C_Sharp_(programming_language).**
2. **Wikipedia. *C#*. Available from: http://no.wikipedia.org/wiki/C_Sharp.**
3. **Community, D.I.C. *Pros/Cons of the C# Language*. 13.12.2012]; Available from: http://www.dreamincode.net/forums/topic/29142-proscons-of-the-c%23-language/.**

4. **SQLite.org. *About SQLite*. 20.12.2012]; Available from: http://www.sqlite.org/about.html.**
5. **Richter, J. *Garbage Collection: Automatic Memory Management in the Microsoft .NET Framework*. 13.12.2012]; Available from: http://msdn.microsoft.com/en-us/magazine/bb985010.aspx.**
6. **Wikipedia. *Software framework*. 13.12.2012]; Available from: http://en.wikipedia.org/wiki/Software_framework.**

Risikovurdering

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0		03.01.13	Marcus Jernberg
2.0	Lagt til C# - SQLite	05.02.13	Marcus Jernberg
3.0	Lagt til et punkt "Løst" + oppdatert C#/SQLite under løst	12.03.13	Marcus Jernberg
4.0	Lagt til risiko ang kryptering.	14.03.2013	Marcus Jernberg
5.0	Lagt til PC-kræsj	18.03.2013	Marcus Jernberg
6.0	Lagt til risiko mtp musikk	30.04.2013	Henrik/Marcus
7.0	Lagt til løsning på musikk	24.05.2013	Marcus

Kontrollert av:	HO, ZA

Innholdsfortengelse

1. Risikoanalyse

Når man jobber med prosjektarbeid i gruppe er det nødvendig å ta høyde for at ting ikke alltid går helt etter planen. I de fleste prosjektmodeller er det vanlig å vurdere hvilke risikoer som kan påvirke prosjektet, og deretter sette opp en plan for hvordan man skal håndtere hvert av disse punktene. Siden gruppen jobber med SCRUM som prosjektmodell, bruker vi mindre tid og ressurser på planlegging av risiko. Filosofien bak SCRUM er at man setter opp sprinter med bakgrunn i en viss hastighet og gjennomføringsevne. Dersom det dukker opp uforutsette hendelser må gruppen tilpasse sprinter og forventet hastighet, dette er relativt enkelt å gjøre. Det gjør altså at vi kan bruke mindre ressurser på å planlegge for eventuelle uforutsette hendelser og fokusere på gjennomføring av prosjektet. Så fort gruppen oppdager potensielle problemer skal dette lokaliseres og dokumenteres i risikoanalyse dokumentet.

2. Mulige forutsette problemer

- Det kan være et problem å eksportere Unity prosjekter fra en PC til en annen fordi det ikke finnes noen innebygget kildekontroll som er tilpasset filstrukturen til Unity. Vi har lest om forskjellige metoder å gjøre dette på, som blandt annet innebærer å ZIPe alle biblioteker når man har gjort endringer i prosjektet. Men vi vet ikke helt hvordan dette vil funke i realiteten. Vi er også usikre på hvor stort problem dette blir.
- Det kan være problematisk å sette opp SQLite når vi programmerer i C#. Det kan være enklere om vi bruker Javascript. Dette er noe vi må undersøke ytterligere, og deretter bestemme oss om vi blir nødt til å gå over til javascript. Om dette skjer blir det lite problematisk fordi vi undersøker dette før vi begynner å programmere noe annet.
- Databasen er foreløpig ikke kryptert. Hvis det i senere samtaler med Sunnaas viser seg at databasen må krypteres av hensyn til personvern, så har vi følgende alternativer: Enten implementerer vi en løsning som benytter seg av MySQL, evt prøver vi å legge til en av krypteringsimplementasjonene til SQLite.
- Det har vist seg å være vanskelig å ha en metode for automatisk rytmedeteksjon i musikken har vist seg å være veldig krevende og muligens lar det seg ikke gjør i det heletatt. Ved å importere en tilfeldig lydfil, er det vanskelig å forutse hvordan man kan hente ut beats. Da det er vanvittig variasjon i samplene hentet ut fra lydfilen. Det som gjenstår er å designe en algoritme som vil være tilpasningsdyktig i forskjellige områder, imidlertid er dette noe som viser seg å være veldig krevende. Da vi skal ha en spillbar prototyp kan vi prøve å få tilpasset denne algoritmen til en spesifikk lydfil.

3. Løst

- Det viste seg å være uproblematisk å bruke SQLite og C# sammen.
- Vi har klart å hente ut beats fra lydfiler på en annen måte en vi først antok. Det fungerer og vi får frem rytmen i musikken, men ikke like bra som vi håpet på.

4. Uforutsette problemer som oppstod

- Hovedkortet på den ene stasjonære PCen som står i grupperommet sluttet å fungere. Heldigvis hadde vi backup av alt som var på den. Hard disken fungerer fortsatt.

5. Ordliste

SCRUM – Prosjektmodell
Unity – Utviklingsverktøy
ZIP – Filformat for å komprimere data
SQLite – Hjelpemiddel for å lage databaser
C# - Programmeringsspråk

Analysedokument

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet innledende dokument	04.02.13	Zana Ali
2.0	Oppdatert dokumentet	17.03.13	Zana Ali
3.0	Satt sammen analysedokumenter	19.03.13	Zana Ali
4.0			
5.0			
6.0			

Kontrollert av:	

Innholdsfortegnelse

1. Innledning

I vårt prosjekt har vi fått en del krav til systemet som skal utvikles. Disse skal analyseres for å bedre forståelse av kravene og hvordan systemet skal bygges opp. Analyse betyr å bryte ned, og det er det vi gjør med de kravene vi har. UML har blitt brukt til dette.

2. UML

UML er et standardisert modellereringsspråk. UML gir et sett med grafiske notasjonsteknikker for å lage en visuell modell av objektorienterte software systemer. Verktøyet som har blitt brukt til å tegne diagrammene er Visual Paradigm.

2.1 Diagrammer

UML består av 9 ulike diagrammer, kun de sentrale som vi har brukt er blitt beskrevet.

2.1.1 Use Case diagram

Et use case diagram er en simpel representasjon av brukerens interaksjoner med systemet, og oppførselen til systemet. Use case diagrammer beskriver hva systemet skal gjøre, men ikke hvordan det skal gjøres. Brukeren blir sett på som en aktør. Et system består gjerne av flere aktører som er involvert i systemet. Use case er det samme som en funksjon. Hver ting en funksjon skal gjøre for oss kalles derfor en use case.

2.1.2 Aktivitetsdiagram

Aktivitetsdiagram er en grafisk representasjon av scenariobeskrivelsen. Den viser steg for steg aktiviteter og handlinger, med støtte for valgalternativer. Hver use case har gjerne et tilhørende aktivitetsdiagram.

2.1.3 Klassediagram

Klassediagram viser hvilke objekter programmet består av. Klassediagrammet viser klassene og samhandlingene som inngår i modellen og relasjonene mellom dem. Struktur og ansvarsområdet blir beskrevet her. Hver klasse har attributer og funksjoner. Attributer er variablene mens funksjoner er metodene. Funksjonene får ting til å skje, mens attributene blir satt og kan brukes og modifieres av funksjonene. Navnet på et klassediagram bør gjenspeile hva slags arbeidsoppgave objektet har. I UML brukes symbolene (+) og (-) for å representere public og private attributter og funksjoner.

2.1.4 Sekvensdiagram

Sekvensdiagram beskriver metodekall og tidsrekkefølgen for disse. Sekvensdiagrammet har to dimmensjoner:
1. Vertikal som representerer tiden.
2. Horisontal som representerer de forskjellige objektene[1]

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Lagt inn diagrammene i analysedokumentet	04.02	Zana Ali
2.0	Oppdatert	14.03	Zana Ali
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	MJ, JEH

3. Konto

Her beskrives hvordan konto fungerer og hvilke muligheter en bruker og admin har i systemet.

4. UML

4.1 Use case diagram

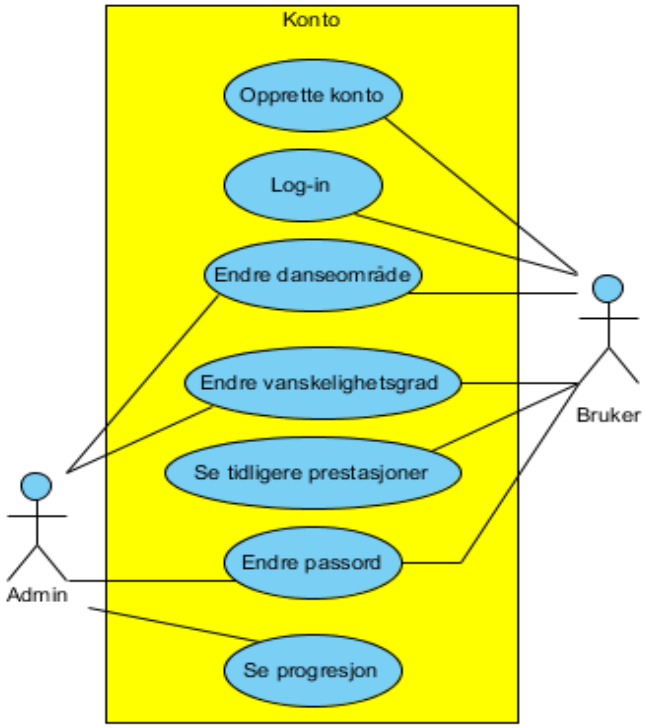
En bruker skal kunne:

- Opprette konto
- Logge inn på sin konto
- Kunne velge og endre danseområde
- Kunne velge og endre vanskelighetsgrad
- Se tidligere prestasjoner
- Endre passord

Administrator skal kunne:

- Logge inn som administrator
- Endre brukernes danseområde
- Endre brukernes sin vanskelighetsgrad
- Endre brukernes passord
- Se progresjonen til brukere

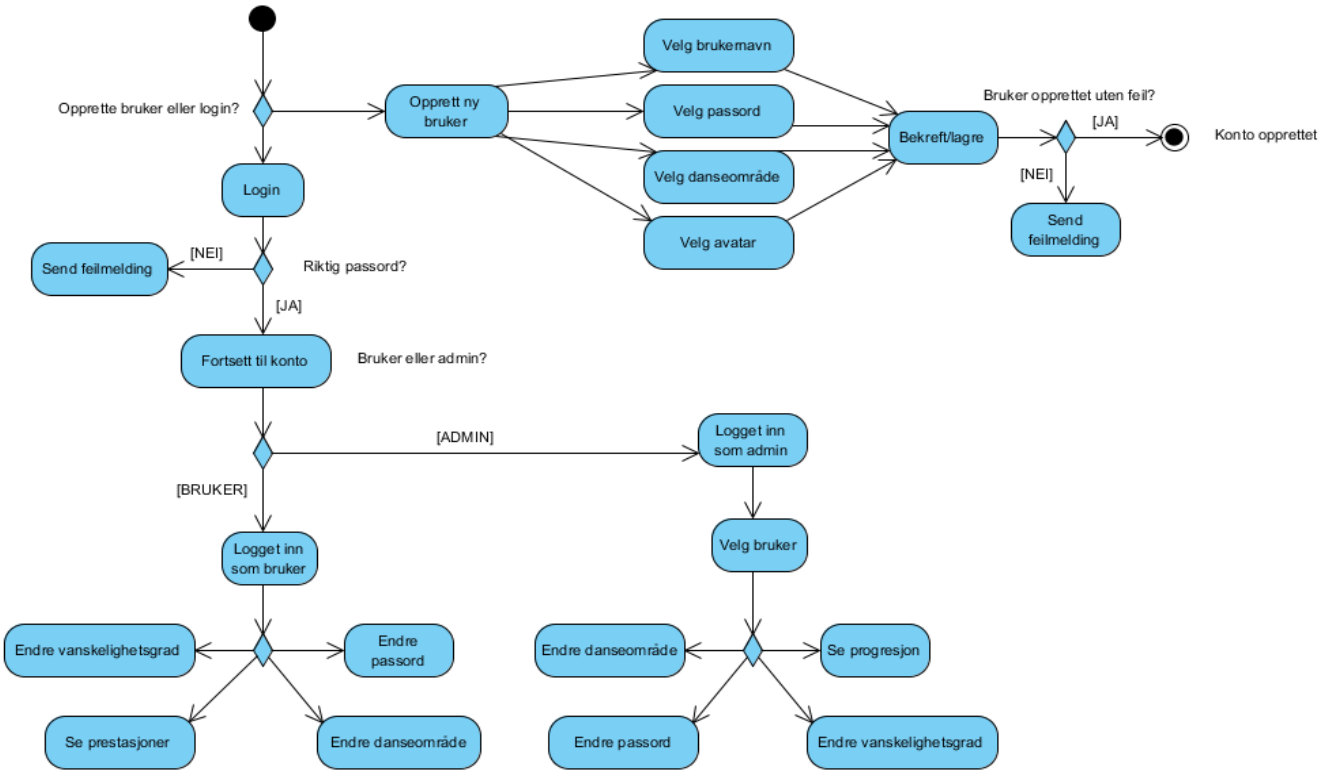
Ergoterapeutene vil ha mer ut av å se på progresjonen til brukerne enn noe annet, mens brukerne sannsynligvis er mer opptatt av å se prestasjoner og mer spillrelatert data. Derfor er de ikke en use-case for begge aktørene.



Figur1: Use case diagram som beskriver bruker og admin i systemet og interaksjonene mellom aktørene og use-casene.

4.2 Aktivitetsdiagram for konto

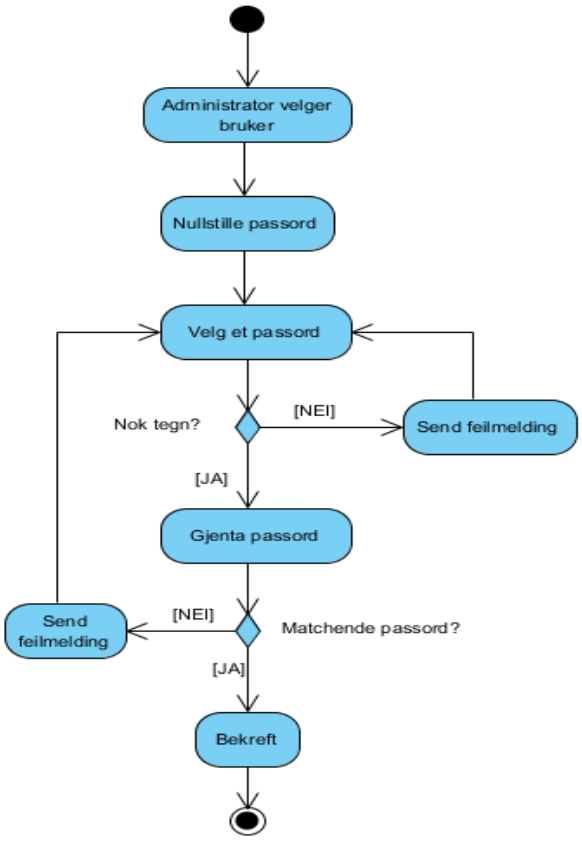
I figur 2 ser man et aktivitetsdiagram som gir oversikten over aktivitetene i konto. Aktivitetsdiagrammet baserer seg rundt use-case diagrammet for konto. Aktivitetsdigrammet viser flyten fra en aktivitet til en annen og de mulighetene en bruker og en admin skal ha i systemet.



Figur2: Aktivitetsdiagram over konto.

4.2.1 Nullstilling av passord som admin

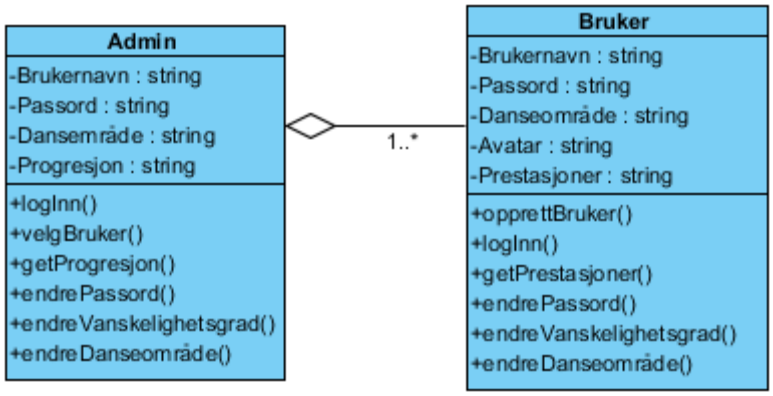
Som administrator skal man kunne nullstille passord for en bruker. Diagrammet under viser hendesesforløpet for dette. Brukeren kan også nullsstille passord, med nærmest identisk oppsett bortsett fra at en bruker kun kan nullstille eget passord, og ikke gå inn på andre brukere slik admin har rettigheter til.



Figur3: Aktivitetsdiagram for nullstilling av passord

4.3 Klassediagram

Klassediagrammet viser metodene og attributene som inngår i konto. Attributene står først med datatypen ved siden av. Relasjonsnotasjonen representerer at admin har tilgang til flere brukere.



Figur4: Klassediagram som viser relasjonen mellom admin og bruker

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Første utkast	14.02.2013	Dagfinn Kjærnet
2.0	Oppdatert	18.03.2013	Dagfinn Kjærnet
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	HO, JEH

5. Database

For å kunne lagre data mellom spillsesjoner benytter vi oss av en database. På denne måten kan brukerne fortsette der de gav seg og de slipper å fullføre prestasjoner og lignende på nytt. Dessuten fortsetter man rett på den vanskelighetsgraden man nådde, og kan følge statistikken sin for å se hvordan brukeren har utviklet seg.

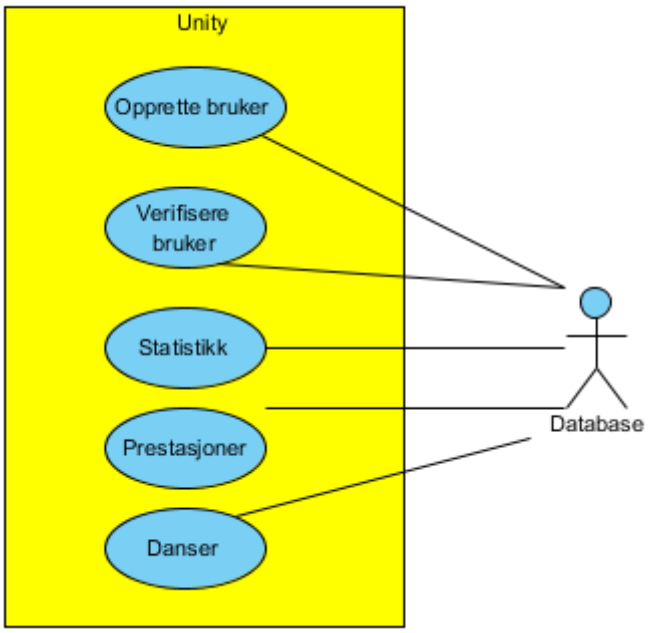
For å kunne koble til en database benytter vi oss av SQLite, dette er en kompakt selvstendig databasemotor som kan integreres direkte inn i spillet så vi slipper å sette opp en egen databaseserver. Vi trenger da et script som kan håndtere all database interaksjon og konstruere gyldige SQL spørringer og kommandoer.

6. UML

6.1 Use case diagram

Databasen skal kunne utføre følgende oppgaver:

- Lagre og legge til nye brukere.
- Sammenligne inntastet data med infoen som ligger lagret i databasen, slik at brukere kan verifiseres.
- Lagre og hente statistikken til brukerne.
- Lagre prestasjonsprogresjonen til brukerne.
- Lagre dansebevegelsene som genereres til en sang, slik at det ikke genereres nye danser hver gang.

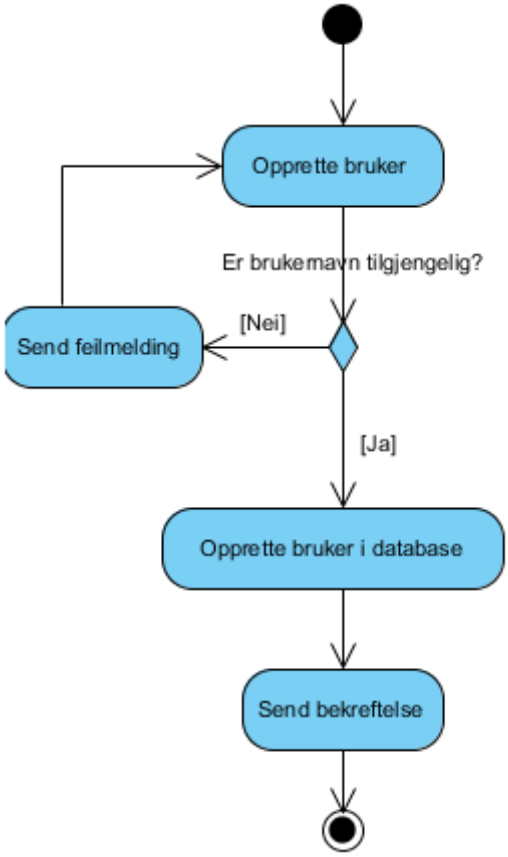


Figur5: Use case diagram som beskriver databasens relasjon til Unity.

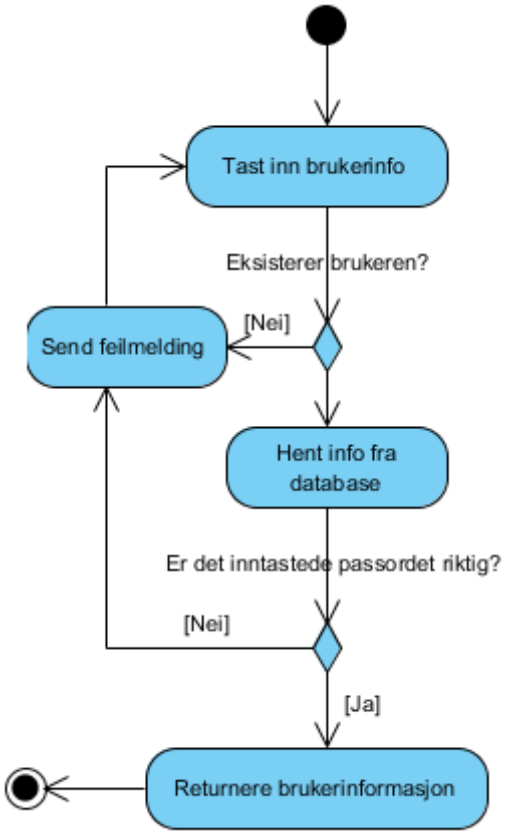
6.2 Aktivitetsdiagram for bruker

I **figur6** ser vi aktivitetsdiagrammet for å opprette en ny bruker. Vi ønsker å sjekke om brukernavnet er tilgjengelig og å gi brukeren beskjed dersom brukernavnet allerede er tatt. Hvis brukernavnet er gyldig skal det genereres og kjøres en SQL querry som legger til denne brukeren i databasen, deretter skal det sendes en bekreftelse på om opprettingen ble fullført.

I **figur7** ser vi diagrammet for når en bruker prøver å logge inn, infoen blir sendt til databasehåndteringsklassen som sjekker om brukeren eksisterer. Hvis ikke skal det sendes en feilmelding om dette, hvis brukeren finnes så hentes infoen til denne brukeren ut. Deretter sammenlignes passordet i databasen med det inntastede passordet, hvis de ikke er like skal det returneres en feilmelding om at passordet er feil og vi går tilbake og venter på å bli matet med ny innloggingsinfo. Hvis passordet er riktig skal all brukerinformasjonen returneres samt en beskjed om at innloggingen var vellykket.



Figur6: Aktivitetsdiagram for å opprette bruker.

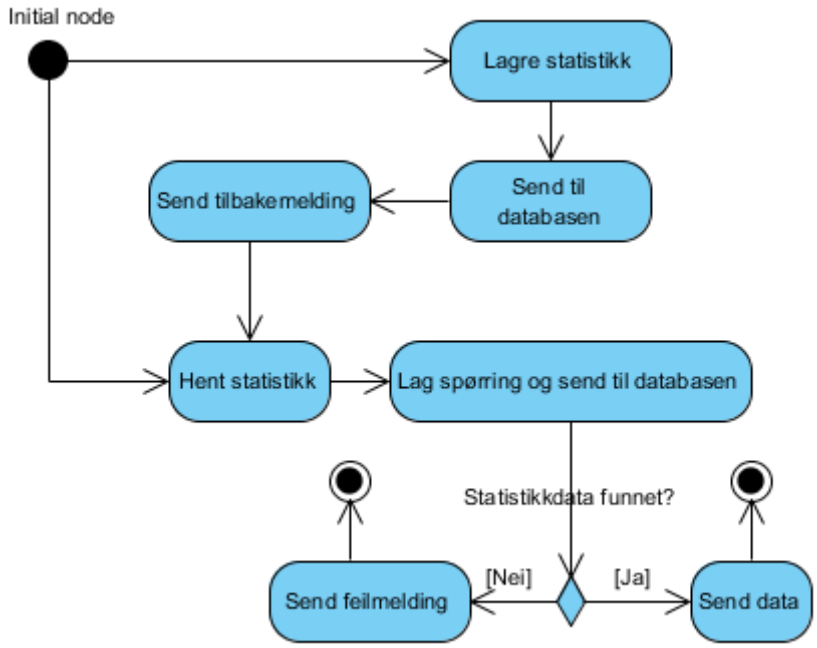


Figur7: Aktivitetsdiagram for å logge inn en bruker.

6.2.1 Lagring og lasting av brukerstatistikk

Vi ønsker å lagre brukernes resultater etter hver dans slik at de kan følge statistikken og se sin egen progresjon, **figur8** viser aktivitetsdiagrammet for hvordan vi ser for oss at databaseklassen skal håndtere dette.

Det blir sendt en forespørsel om å lagre statistikken og det genereres en SQL-querry og statistikken lagres i databasen, deretter skal det returneres en tilbakemelding om lagringen var vellykket eller ei. Deretter kan brukeren be om å hente inn statistikken, spørring lages for å hente ut brukerens statistikk. Dersom den ikke fant noen statistikk for brukeren skal det sendes en feilmelding til brukeren om dette, hvis den finner data skal denne returneres. Brukeren kan også velge å se på sin statistikk uten å utføre en dans først, i så fall går man direkte til aktiviteten "Hent statistikk" og følger flyten videre derfra.

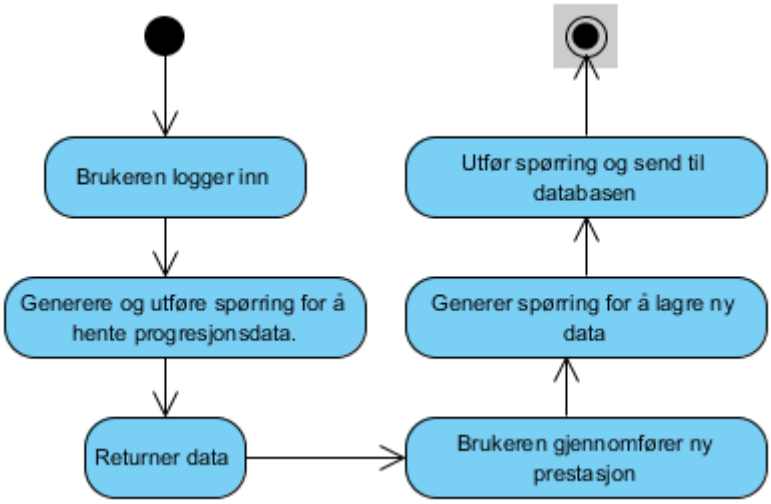


Figur8: Aktivitetsdiagram for statistikk.

6.2.2 Lagring og lasting av prestasjoer

Vi ønsker å lagre en liste over hvilke prestasjoner hver bruker har utført, slik at man ikke må samle alle prestasjonene pånytt hver gang man spiller.

Brukeren logger inn og det genereres og utføres en spørring som henter brukerens progresjon og returnerer resultatet. Når brukeren utfører nye prestasjoner genereres det en spørring som kan lagre dette som deretter kjøres og lagrer statusen i databasen.

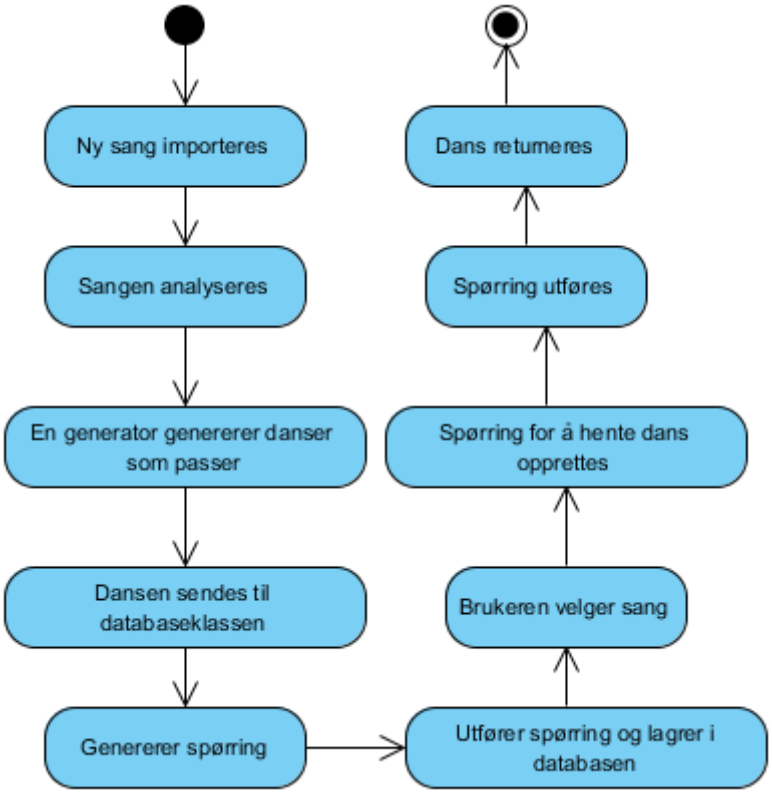


Figur9: Aktivitetsdiagram for prestasjoner.

6.2.3 Lagring av danser

Det kan være frustrerende for brukeren om dansen han/hun har trent på til en spesifikk sang endrer seg mellom hver gang vedkommende ønsker å danse til denne sangen, derfor ønsker vi å lagre alle danser som genereres slik at de forblir like hver gang, i det minste frem til vanskelighetsgraden økes.

Figur10 viser hvordan vi ser for oss denne prosessen. Brukeren henter inn en ny sang, sangen analyseres av en annen klasse og resultatet sendes til en dansegenerator som genererer en sammenhengende dans utfra hvilke dansetrinn som passer sammen. Denne dansen, eventuelt dansene sendes til databaseklassen som genererer og utfører en spørring for å lagre det i databasen. Deretter kan brukeren velge sangen, det opprettes en spørring for å hente tilhørende dans. Spørringen utføres og dansen returneres.



Figur10: Aktivitetsdiagram for danser.

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	30.01.13	JEH
2.0	Oppdatert	20.03.13	JEH
3.0	Oppdatert	19.03.13	JEH
4.0			
5.0			
6.0			

Kontrollert av:	MJ og AO

7. Song Select

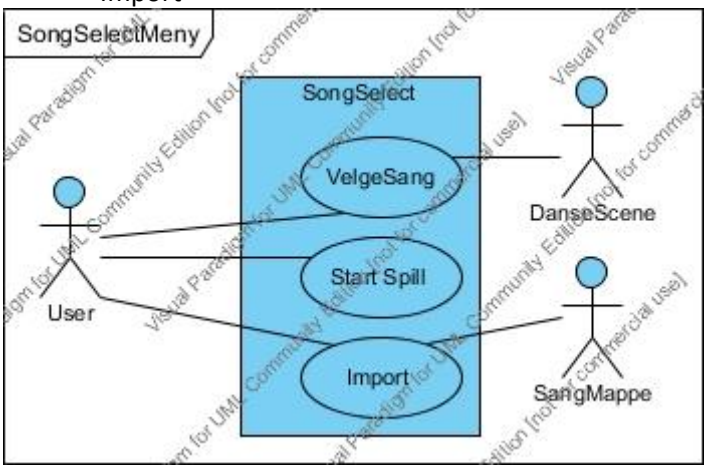
Her vil den delen av menysystemet der man velger sang bli beskrevet og hvilke muligheter brukeren har i denne prosessen.

8. UML

8.1 Use case diagram

Brukeren skal kunne gjøre følgende når det skal velges sang:

- Velge Sang
- Starte spill
- Import



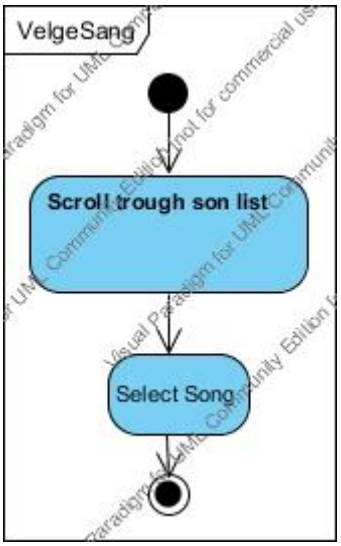
Figur11: Use case Diagram som beskriver SongSelect i systemet og interaksjonen mellom brukeren, danserommet og sangmappen.

Sangmappen inneholder sangene i spillet og disse vil bli importert. DanseScene er nødt til å vite hvilke sang som blir valgt av brukeren.

8.2 Aktivitetsdiagram for SongSelect

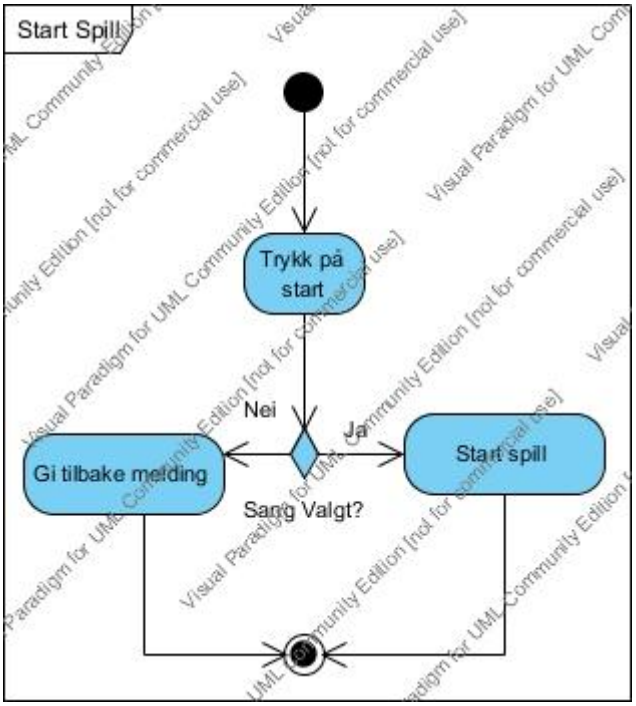
I figurene under vil man se aktivitetsdiagrammer som vil gi en oversikt over aktivitetene som SongSelect systemet inneholder. Aktivitetsdiagrammene baserer seg på use case diagrammet over SongSelect delen. De gir også en beskrivelse av flyten mellom aktivitetene.

8.2.1 VelgeSang



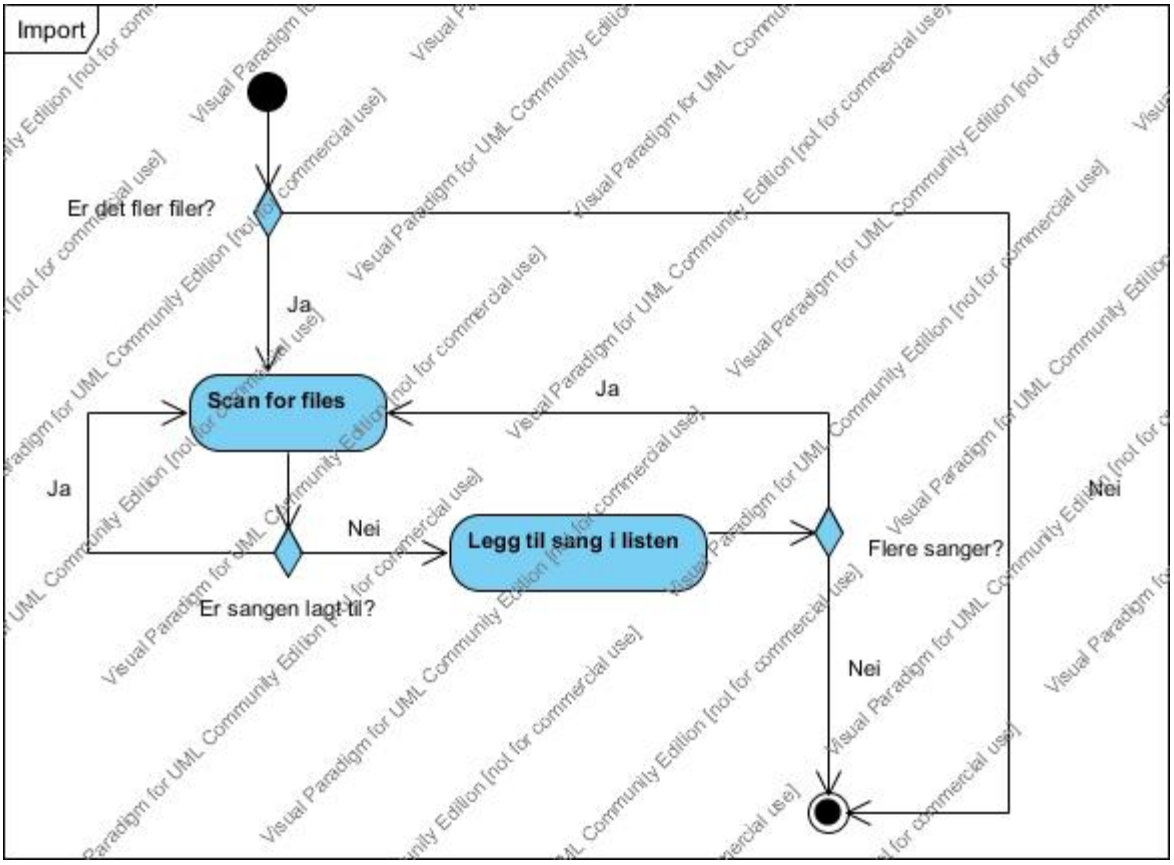
Figur12: Aktivitetsdiagram over VelgeSang

8.2.2 Start Spill



Figur13: Aktivitetsdiagram over Start Spill.

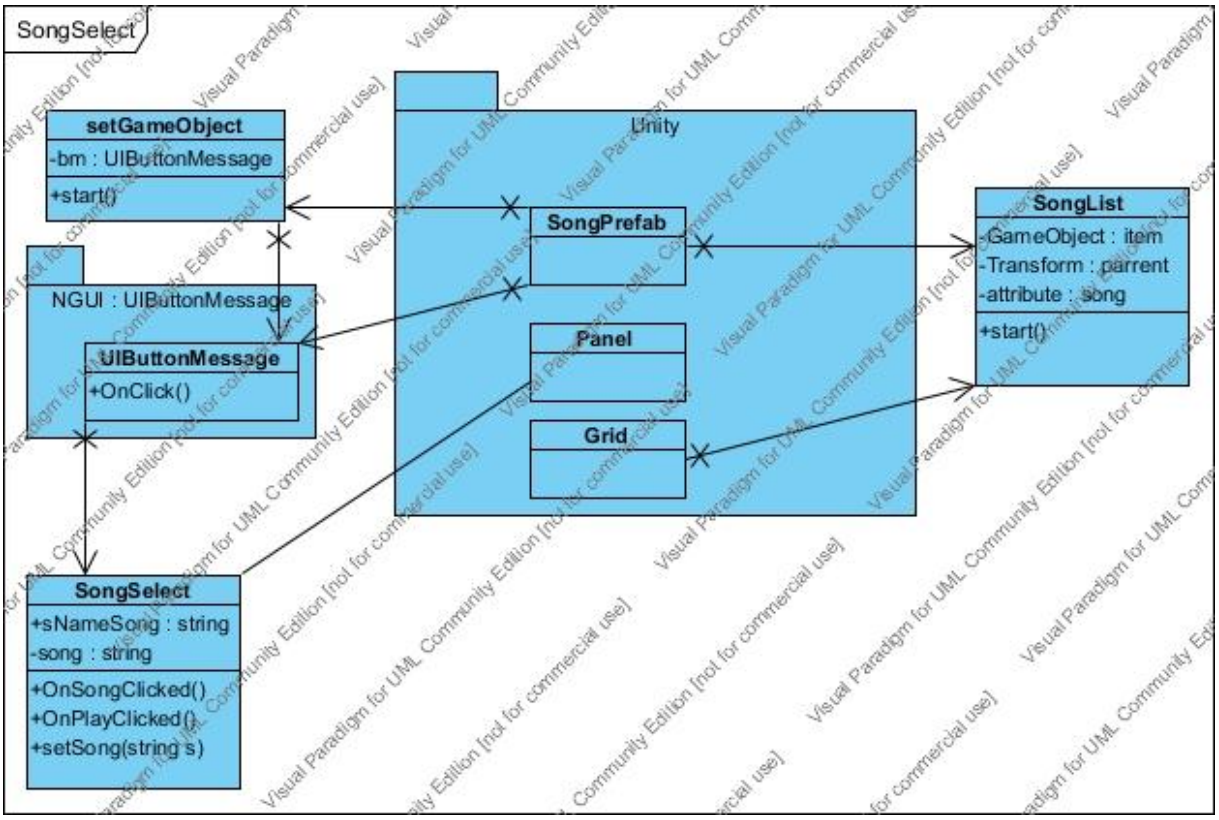
8.2.3 Import



Figur14: Aktivitetsdiagram over Import

8.3 Klassediagram

Klassediagrammet viser metodene og attributtene som inngår i SongSelect delen ved menyen. Attributtene står først og viser hva slags datatype det er. På grunn av at vi bruker unity3D blir klassediagrammet noe spesielt. Klassene inne i Unity packagen vil være GameObjects og klassene uten for vil være scripts som må lages og legges på de forskjellige GameObjectene. Toolkittet vi har valgt å bruke heter NGUI inneholder mange ferdige scripts. Til disse scriptene vil vi kun feste på objekter, slik at objektene kan snakke sammen. I de fleste scriptene vil alt foregå i start(). Der det forekommer noe annet står det beskrevet.



Figur15: Klassediagram over SongSelect, som viser attributer og metoder.

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	06.02.2013	JEH
2.0	Oppdatert	21.02.2013	JEH
3.0	Oppdatert	19.03.2013	JEH
4.0			
5.0			
6.0			

Kontrollert av:	AO, MJ

9. Resultater/Statistikk

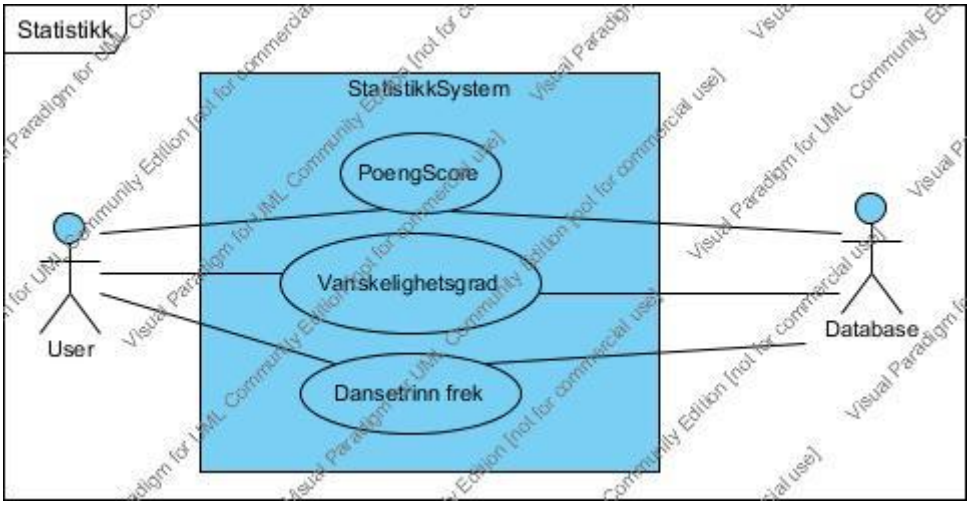
Her beskrives hvordan statistikk delen av systemet vil være og hvilke type statistikk brukeren kan velge mellom å få vist frem.

10. UML

10.1 Use case diagram

Brukeren og admin skal kunne se følgende statistikker:

- Poengscore
- Vanskelighetsgrad
- Dansetrinn frekvens



Figur16: Use Case diagram som beskriver statistikk i systemet og interaksjonen mellom brukeren og databasen som inneholder dataen brukeren trenger for å få vist frem ønsket statistikk.

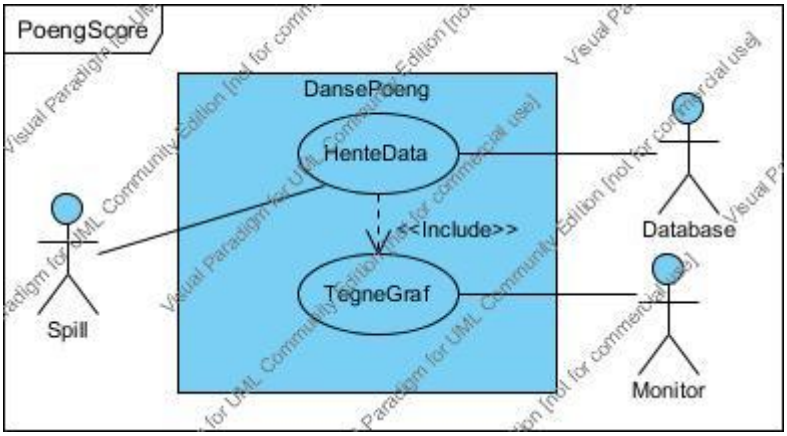
10.1.1 Use case diagram for PoengScore, Vanskelighetsgrad og Dansetrinn frekvens

For å vise frem statistikkene er vi nødt til å ha følgende funksjoner:

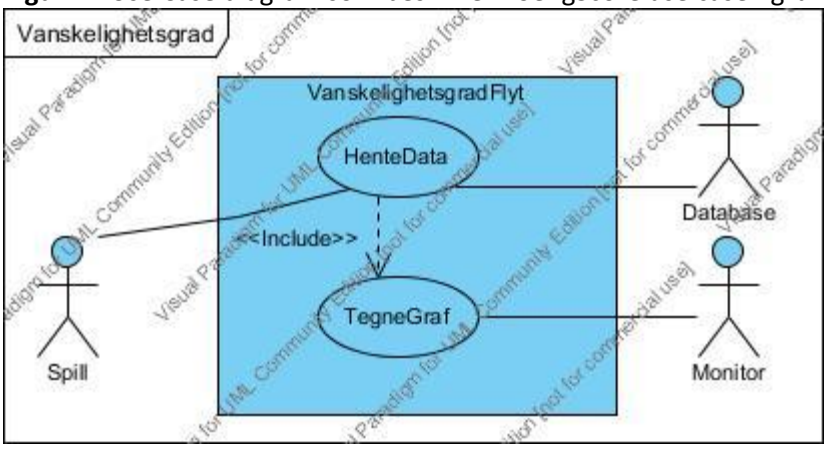
- Hente data
- Tegne graf

Disse funksjonene er avhengige av følgende tre aktører:

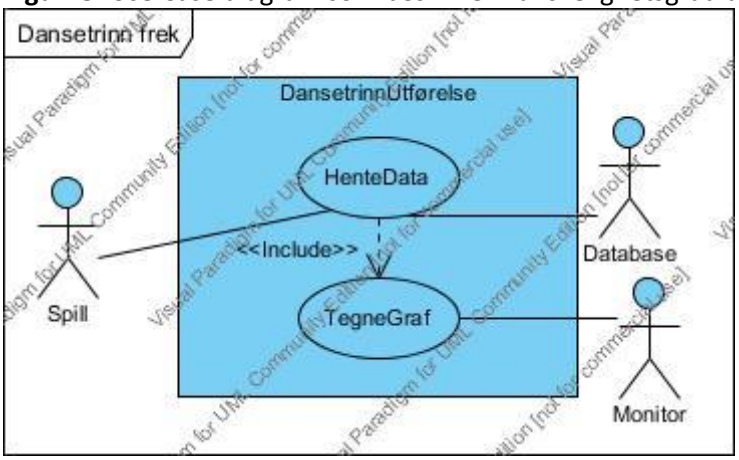
- Spillet/User
- Database
- Monitor



Figur17: Use Case diagram som beskriver PoengScore use casen grundigere.



Figur18: Use Case diagram som beskriver Vanskelighetsgrad use casen grundigere.

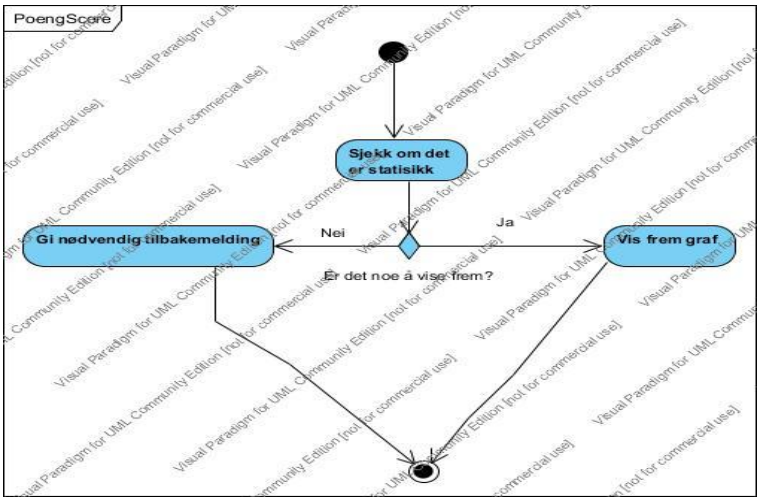


Figur19: Use Case diagram som beskriver Dansetrinn frekvens use casen grundigere.

10.2 Aktivitetsdiagram for statistikker

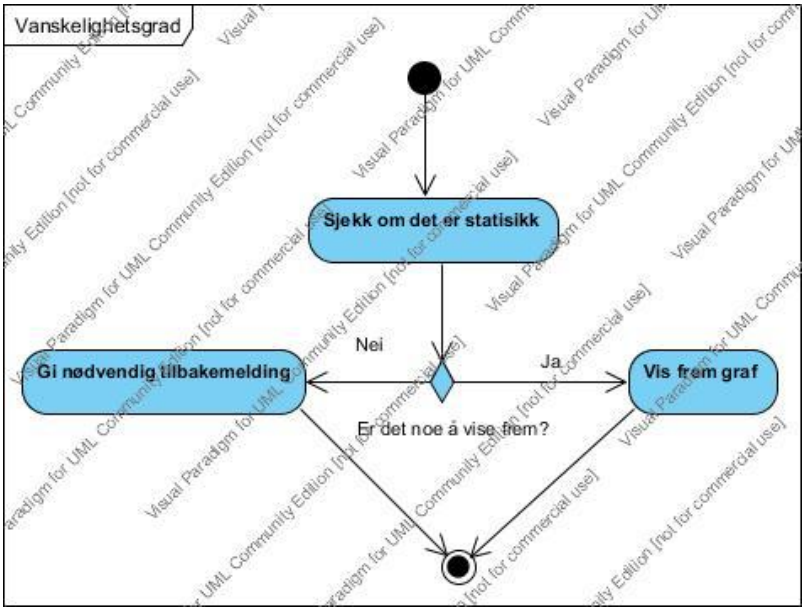
I figurene under vil man se aktivitetsdiagrammer som gir en oversikt over aktivitetene som statistikk systemer vil inneholde. Aktivitetsdiagrammene baserer seg rundt use-case diagrammet for statistikk. Det gir også en beskrivelse av flyten fra en aktivitet til en annen og de valgene systemet vil møte.

10.2.1 PoengScore



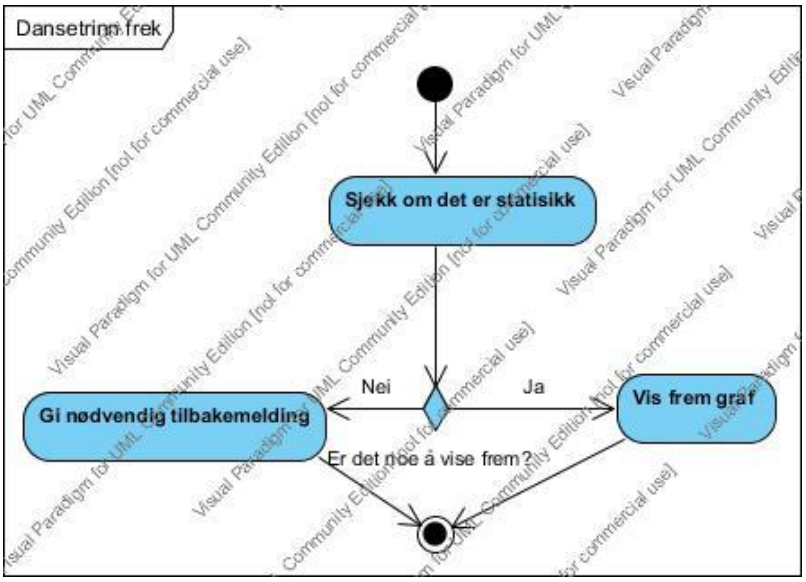
Figur20: Aktivitetsdiagram over PoengScore.

10.2.2 Vanskelighetsgrad



Figur21: Aktivitetsdiagram over vanskelighetsgrad.

10.2.3 Dansetrinn frekvens



Figur22: Aktivitetsdiagram over dansetrinn frekvens.

10.3 Klassediagram

Klassediagrammet viser metodene og attributtene som inngår i statistikk fremvisningen. Attributtene står først og viser hva slags datatype det er. I dette tilfellet vil vi ha float array som vil bli gitt verdier ved hjelp av get-metodene.

Figur23: Klassediagram som viser metoder og attributter.

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet dokumentet	30.01.13	Are Oven
2.0	Oppdatert	26.02.13	Are Oven
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	MJ og DK

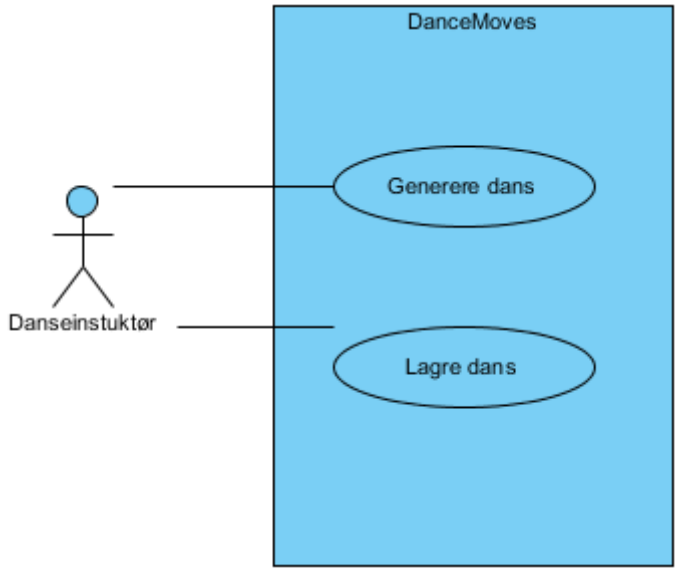
11. Dansegenerator

Dansegeneratoren skal ta å generere dans og sende den videre til danseinstruktøren

12. UML

12.1 Use case diagram

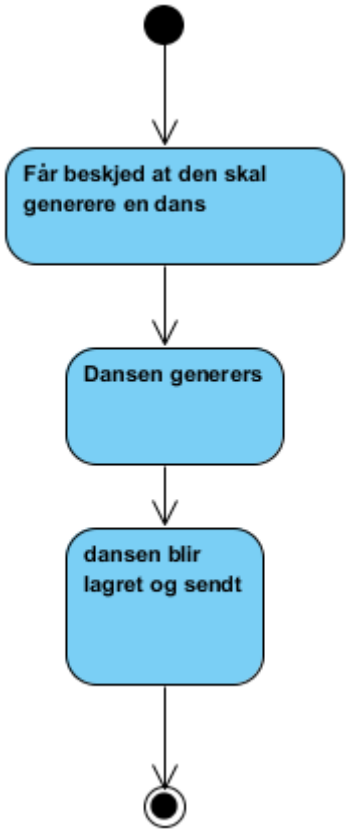
Use case diagrammet viser hva som har kontakt med dancemoves. Det er bare danseinstruktøren som skal ha noe med denne klassen, som i hovedsak skal generere bevegelsene.



Figur24: Use case diagram

12.2 Aktivitetsdiagram

Dette akrivitetsdiagrammet viser aktivitetene som skjer inni danceMoves. Den får beskjed om å lage en dans, deretter blir dansen generert. Når dansen har blitt generert blir den så sendt tilbake til danseinstruktøren.



Figur25: Aktivitetsdiagram for Dansegeneratoren.

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet dokumentet	30.01.2013	Zana Ali
2.0	Oppdatert	21.02.2013	Zana Ali
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	MJ og DK

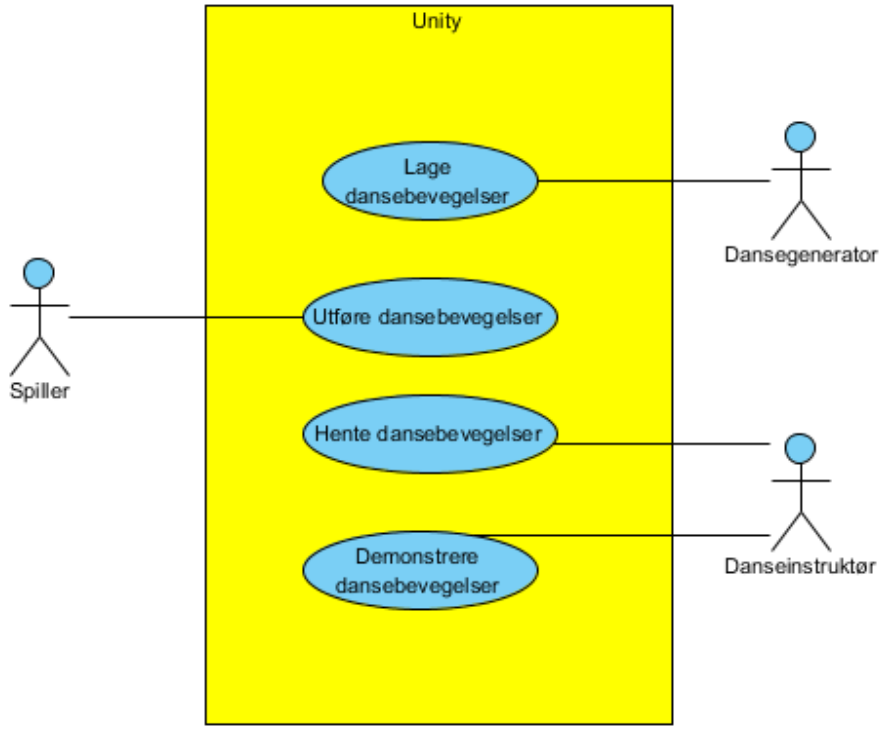
13. Danseinstruktør

Her beskrives danseinstruktøren. Danseinstruktøren skal grafisk demonstrere hvordan en dansebevegelse skal utføres for spilleren.

14. UML

14.1 Use case diagram

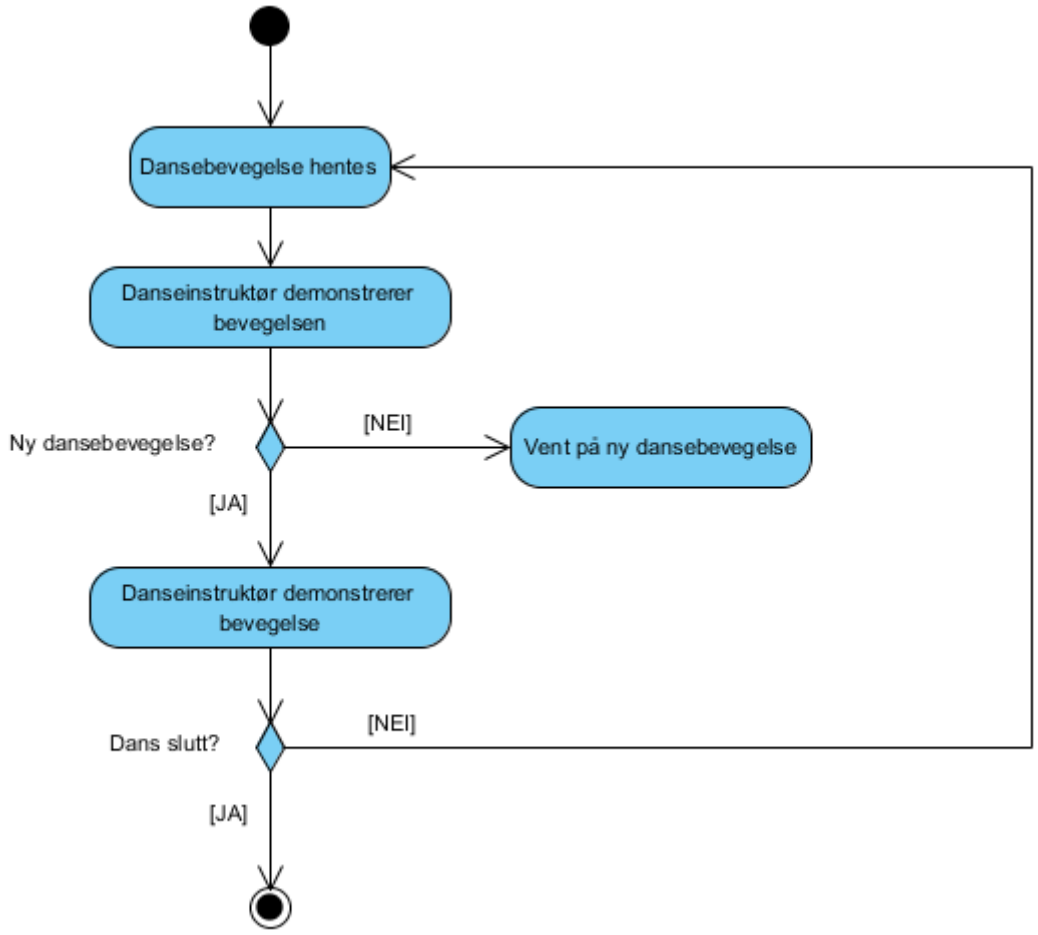
Use case diagrammet viser systemet og hvilke aktører og use caser som er involvert i danseinstruktør.



Figur26: Use case diagram

14.2 Aktivitetsdiagram

Et enkelt aktivitetsdiagram viser oversikten over aktivitene i danseinstruktør. Danseinstruktør henter en bevegelse og demonstrere den. Deretter venter den på ny bevegelse. Ny bevegelse blir generert enten etter en gitt tid, eller når brukeren fullfører bevegelsen. Dette gjentar seg til en dans er slutt.



Figur27: Aktivitetsdiagram for danseinstruktør

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Første utkast	30.01.2013	Marcus Jernberg
2.0	Endret	19.03.2013	Marcus Jernberg
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	DK, JEH

15. Kontakt med kinect

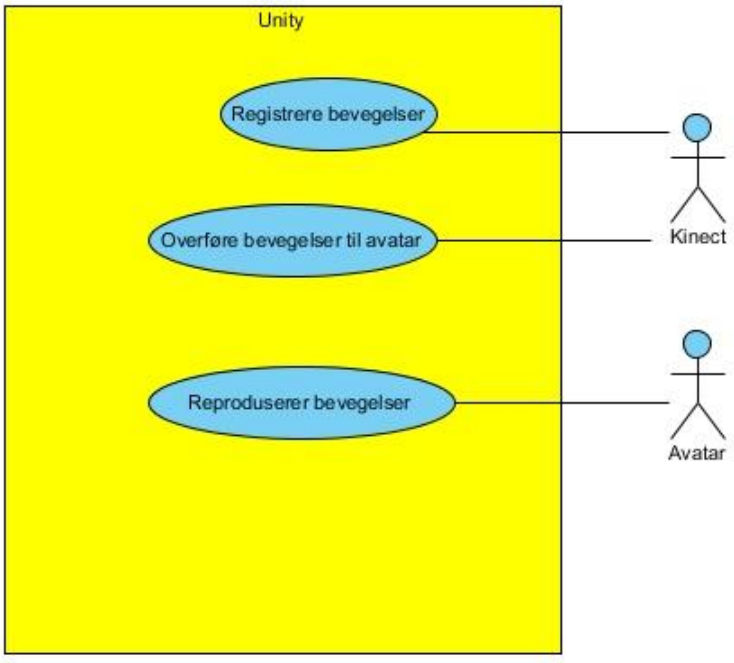
For at spillet skal kunne fungerere som planlagt må vi ha kontakt mellom Unity og kinecten. Unity skal også reprodusere bevegelsene til brukeren gjennom en avatar på skjermen.

16. UML

16.1 Use case diagram

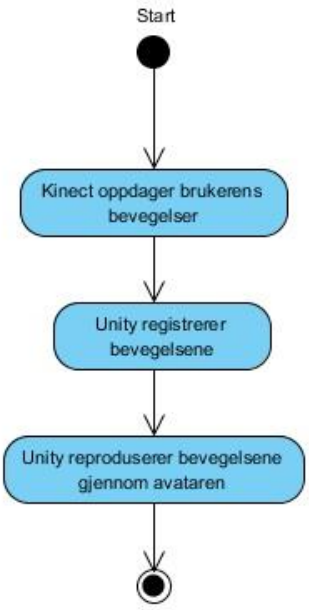
Use Case diagrammet må inneholde:

- To aktører: Kinect og Avatar
- Kinecten skal registrere bevegelser
- Bevegelsene skal hentes inn i unity
- Avataren skal gjenspeile bevegelsene



Figur28: Use Case Diagram

16.2 Aktivitetsdiagram



Figur 19: Aktivitetsdiagram

Revisjonshistorikk

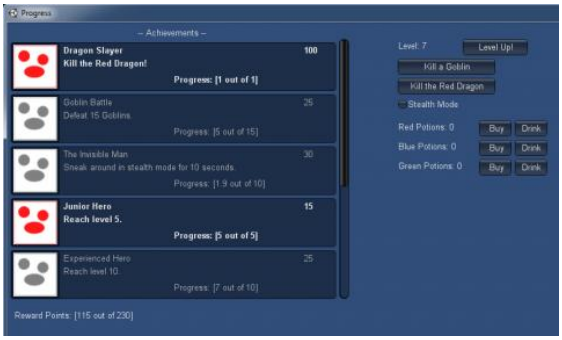
Revisjon #	Endringer	Dato	Skribent
1.0	Første utkast	30.01.2013	Dagfinn Kjærnet
2.0	Oppdatert	19.03.2013	Dagfinn Kjærnet
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	AO, ZA

17. Prestasjoner

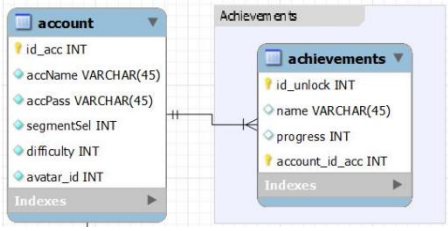
Vi ønsker å benytte oss av prestasjoner som en av motiveringsfaktorene våre. Prestasjoner er små enkle oppgaver en bruker kan løse og blir belønnet for i form av prestasjonspoeng og en beskjed som gratulerer spilleren med gjennomføringen av prestasjonen.

Vi kommer til å benytte oss av rammeverket "Progress"[2] som er utviklet av Steve Gargolinski. Dette støtter mange av de grunnleggende funksjonene vi ønsker som ikoner, egen lyd når man oppnår en prestasjon, liste over alle prestasjoner osv. Det er også en del funksjoner vi må legge til selv som f.eks pop-up ved oppnåelse[3], flytte prestasjonslisten til et eget vindu. Vi ønsker også en måte å lagre hvilke prestasjoner brukeren allerede har låst opp, slik at de ikke låses opp hver gang brukeren begynner en ny sesjon.



Progress - Prestasjonsrammeverk

For å lagre progresjon kommer vi til å benytte oss av databasen, vi lager en tabell som lagrer en liste over progresjonen de forskjellige brukerne har på de forskjellige prestasjonene.



Lagring av progresjon i databasen

Progress inneholder et prestasjonshåndterings script(Achievement Manager), hvor vi lett kan opprette og håndtere prestasjoner. For å endre progresjonen på en prestasjon kalles en av metodene i håndteringsscriptet, f.eks AddProgressToAchievement(). Vi må videreutvikle disse metodene slik at de også aktiverer en pop-up beskjed og lagrer progresjonen i databasen.

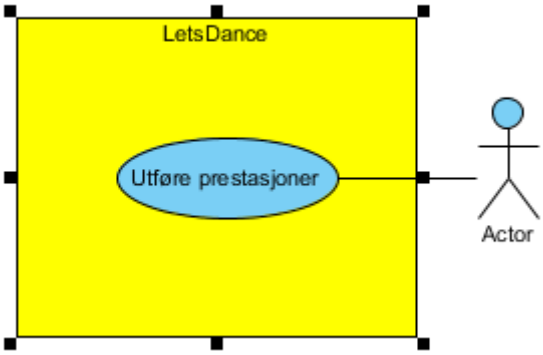
18. UML

Denne seksjonen inneholder de nødvendige UML diagrammer.

18.1 Use case diagram

Prestasjonshåndteringsscriptet skal holde styr på:

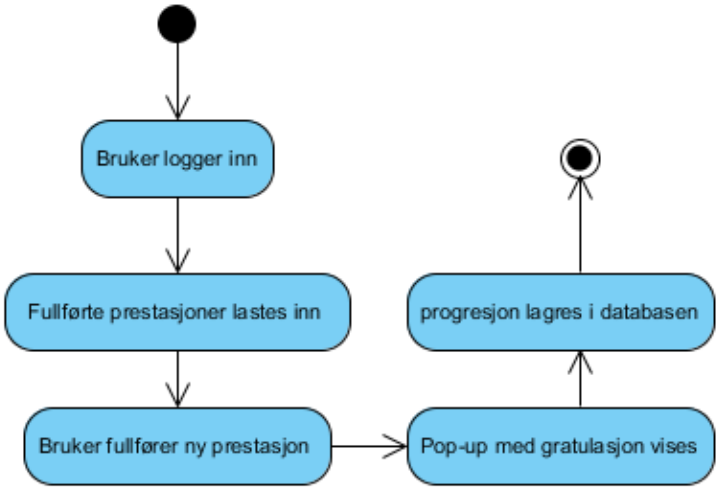
- Hvilke prestasjoner brukeren allerede har låst opp.
- Progresjonen hver bruker har på prestasjoner.
- Vise pop-up ved utførelse av ny prestasjon.



Figur30: Use case diagram som beskriver databasens relasjon til unity.

18.2 Aktivitetsdiagram for prestasjoner

I **figur31** ser vi aktivitetsdiagrammet for prestasjonssystemet. Brukeren logger inn og det utføres en sjekk for å hente en liste fra databasen over utførte prestasjoner. Brukeren fullfører en ny prestasjon, det skal da vises en pop-up for å gratulere brukeren, så skal progresjonen lagres i databasen.



Figur31: Aktivitetsdiagram.

Revisjonshistorikk

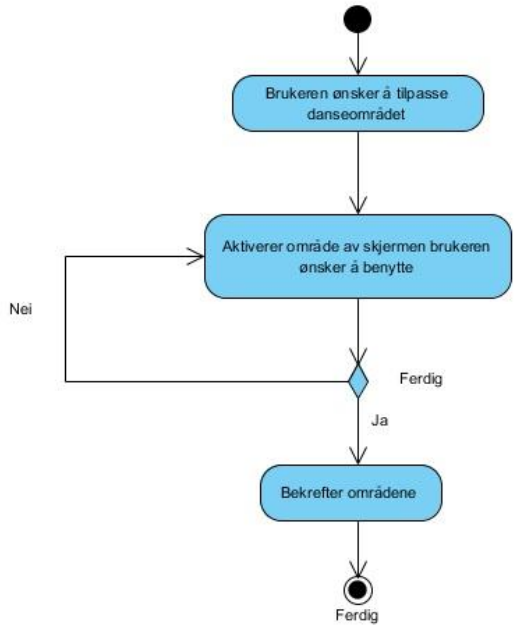
Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet	14.03.13	MJ
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	AO, JEH

18.2.1 Use Case: Tilpasse

For at brukere som ikke kan bevege alle deler av kroppen skal kunne få mest ut av LetsDance, ønsker vi at de skal kunne tilpasse spillet etter hvilke deler av kroppen de ønsker å bruke. Dette gjøres ved at de beveger f.eks høyre arm på høyre side av skjermen, høyre fot på høyre side av skjermen og venstre fot på venstre side av skjermen. Da vil bevegelser som inkluderer bruk av venstre arm falle fra, og brukeren vil kunne utføre bevegelser tilpasset hans eller hennes behov.

18.2.2 Aktivitetsdiagram



19. Ordliste

UML - Modellereringsspråk (Unified Modelling Language)
Script - I Programmering: Sekvens med instruksjoner som blir tolket og utført
Database - Elektronisk arkiv som lagrer data

20. Referanser

1. home.online. 1999 04.02.2013]; Available from: http://home.online.no/~moestboe/uml_oversikt.htm.
2. Gargolinski, S. *Progress - a free achievement framework for unity*. 15.01.2013]; Available from: http://www.stevegargolinski.com/progress-a-free-achievement-framework-for-unity/.
3. Unity3D.Community. *Creating an Achievement system*. 15.01.2013]; Available from: http://forum.unity3d.com/threads/115824-Creating-an-Achievement-System.

Designdokument

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Satt sammen	25.05.13	JEH
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA, AO, DK

Innholdsfortegnelse

Meny

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	20.02.13	JEH
2.0	Oppdatert	19.03.13	JEH
3.0	Oppdatert, lagt til nye userstories og ført aktuelle endringer	21.05.13	JEH
4.0	Oppdatert, ført aktuelle endringer	23.05.13	JEH
5.0	Skrevet statistikk, ID 181	23.05.13	MJ
6.0	Lagt til klassediagram	24.05.13	JEH
7.0			

Kontrollert av:	MJ, ZA

1. Userstories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
113	Arkitektur: Meny	JEH
181	Funksjon: Statistikk - Stolpediagram	MJ
183	Arkitektur: Meny – Storyboard v2 - Bruker	JEH

1.1 ID 113 - Meny

Userstoryen er blitt delt opp i flere mindre userstories, da det kom frem at det ville ta lenger tid en antatt. Dette dokumentet vil derfor inneholde følgende userstories:

ID 113 – Meny
- Anslått tid: 20 timer
- Tid Bruk: 20 timer

ID 133 – New User
- Anslått tid: 4 timer
- Tid Bruk: 4 timer

ID 134 – Song Select
- Anslått tid: 5 timer
- Tid Bruk: 8 timer

ID 136 – Achievments
- Anslått tid: 2 timer
- Tid Brukt: 3 timer

ID 191 - Musikk mappe
- Anslått tid: 10 timer

- Brukt tid: 10 timer

ID 201 – Change Password scene
- Anslått tid: 3 timer
- Brukt tid: 3 timer

ID 202 – Startupscreen scene
- Anslått tid: 1 time
- Brukt tid: 2 timer

ID 203 – Tilpasse scene
- Anslått tid: 4 timer
- Brukt tid: 4 timer

ID 205 – Statistikkoversikt scene
- Anslått tid: 2 timer
- Brukt tid: 2 timer

ID 181 – Funksjon: Statistikk – Stolpediagram
- Anslått tid: 5 timer
- Brukt tid: 5 timer

ID 195 – DoNotDestroy
- Anslått tid: 1 time
- Brukt tid: 2 timer

1.2 ID 183 - Storyboard v2 – Bruker

Disse userstoryene er kommet med fordi vi ønsket å oppgradere layouten og flyten i menyen. Vi ønsket å få oppgradert til en meny som tar utgangspunkt i metrodesign. Da dette vil føre til at vi får en meny som er meget enkel å benytte seg av.

ID 183 – Storyboard v2 – bruker
- Anslått tid: 10 timer
- Brukt tid: 10 timer

ID 204 – Oppdatere til metrodesign
- Anslått tid: 6 timer
- Brukt tid 6 timer

2. Innledning

Alle userstories denne delen av designdokumentet omhandler, er arkitektur deler av menysystemet. Menyen blir designet og laget ved hjelp av NGUI. For mer informasjon om NGUI se eget teknologidokument[1]. I userstory ID: 113 blir det opprettet prefabs som gjør arbeidet i de andre userstories enklere. Da det bare er å dra innpå prefabene for å få lik layout. Vi har valgt å opprette en egen scene for hver userstory. Dette kan vi gjøre fordi de er forskjellige deler av menyen.

Mange deler av menysystemet er nært knyttet til andre deler av systemet. Menyen har en nær tilknytning til databasen, import av sang, generering av dans og danserom. Disse vil derfor bli inkludert i noen av diagrammene.

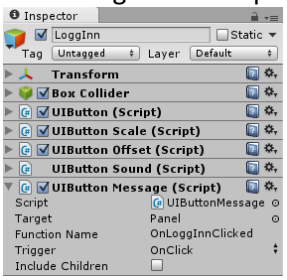
I userstory ID: 183 blir det tegnet et nytt og forbedret versjon av storyboardet som omhandler brukeren sin meny. I forbindelse med å oppgradere layoutene til menyen var vi nødt til å redesigne noen av prefabene som tidligere har blitt laget.

3. Implementasjon

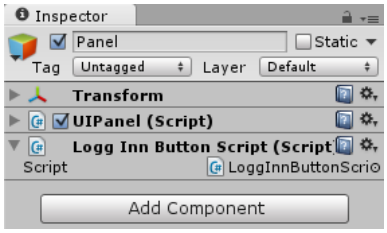
Implementasjonen av hver userstory varierer en del. Det som er felles er hvordan de skal opprettes og hvordan utformingen skal være. Når den tilhørende scenen blir opprettet i unity3d skal kameraet som blir automatisk opprettet slettes. Deretter skal NGUI-UITool benyttes for å opprette rammeverket i scenen. I user story 113 skal det lages prefaber som vil gjøre jobben med lik layout lett. En prefab er et objekt i Unity som er forhåndsdefinert og er bare å dra inn i scenen.

For knapper, tekstbokser, labels, textfield, lister osv. Benytter vi oss også av NGUI. Dette innebærer at vi vil bruke ferdig laget komponenter som allerede har scripts lagt til. Dette er scripts som gjør at en knapp vil fungere og at det blir sendt en beskjed når den blir trykket. Denne beskjeden vil bli sendt til et script vi bestemmer og kaller på en bestemt metode vi selv velger. Dette kan være alt fra last inn neste scene til å vise og forandre forskjellige ting.

Felles for alle scenene er at knappene skal ha en UIButtonMessage script. Det dette skripet skal gjøre er å sende en beskjed til et script som kontrollerer alle knappene i scenen. Dette kontroller scriptet skal ligge på objektet Panel, som blir automatisk generert ved å bruke NGUI UI tool. Det er viktig å merke seg at navnet på dette scriptet kan variere fra scene til scene.



Figur1: Button Message



Figur2: Panel med kontroller script

Det er slik de aller fleste scenene vil være bygget opp. Der det vil vike fra dette vil det kommenteres og forklares i eget avsnitt under.

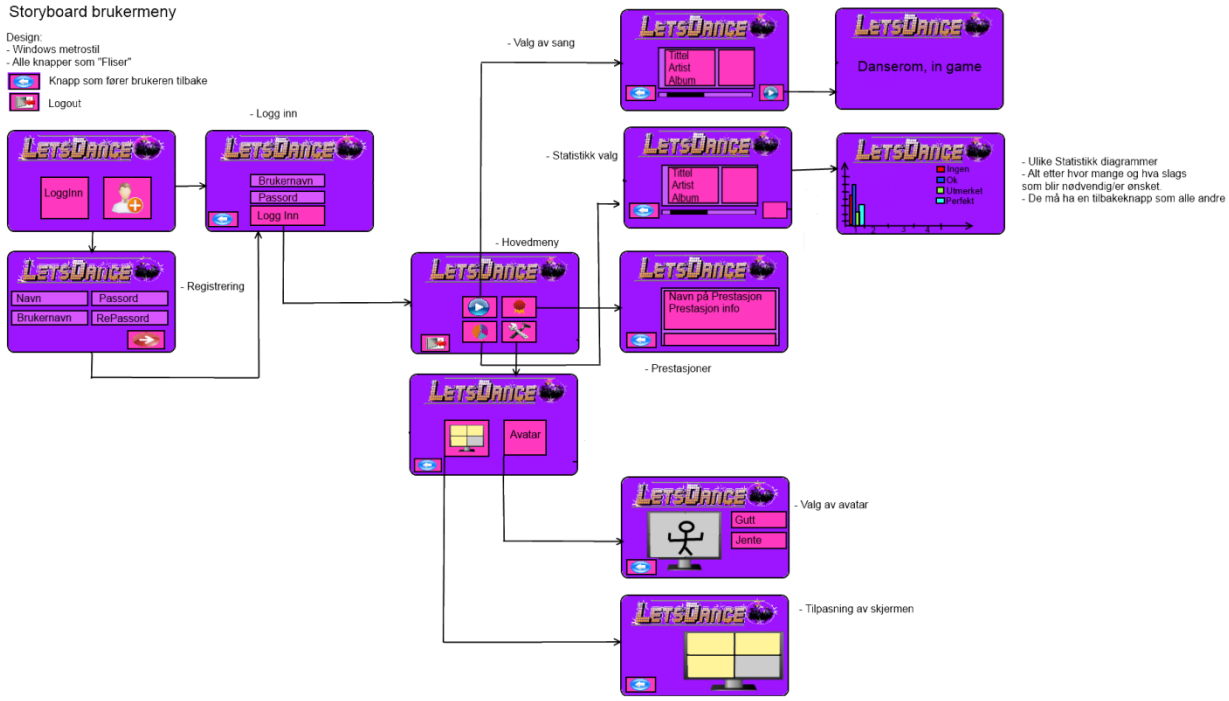
3.1 Storyboard

For å få en bedre oversikt over hvordan menyen kom til å se ut og hvordan flyten i menyen vil være, besluttet vi å benytte oss av storyboard.

3.1.1 ID 183: Storyboard v2 – bruker

I denne userstoryen er målet at vi skal få en god oversikt over flyten i menyen. User story ID: 204 Oppdatere til metro har gått inn under denne delen.

3.1.1.1 Implementasjon

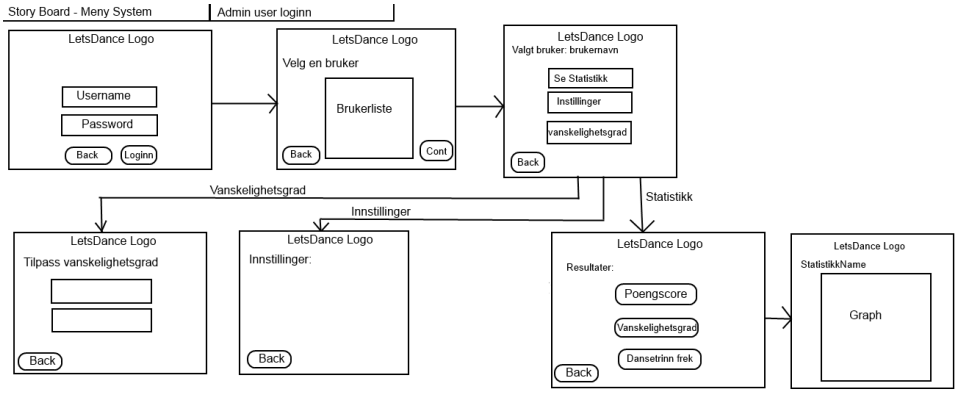


Figur3: Storyboard – user, med metro stil.

3.1.1.2 Vurdering

Som layout i brukeren sin meny har vi valgt å gå for å bruke store "fliser" slik at det skal bli oversiktlig. Ikonene på disse "flisene" vil være ikoner som er lett for brukeren å forstå. Slik at man ikke blir i tvil om hvor knappene fører og hva de gjør.

3.1.2 Storyboard administrator



Figur4: Storyboard – admin

Admin skal ha samme layout som brukeren sin meny, da med store "fliser" og ikoner. Det kommer kun til å være noen få forskjeller fra bruker og administrator menyene.

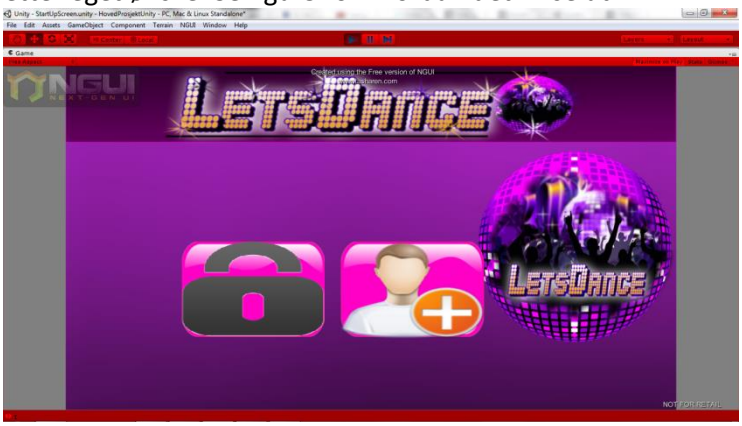
3.2 ID: 113 – Meny

Denne userstoryen er stor og inneholder flere ting, men de har fremdeles mye å gjøre med hverandre. Layout prefabs og knappe prefabs blir definert her.

3.2.1 Implementasjon

Først lages alt av layout i en tom scene. Dette gjøres ved hjelp av NGUI widget tool. Det vi kom frem til er at bakgrunn, header og body skal benytte sliced sprite templaten, med atlaset fantasy. Dette er et atlas og template som følger med NGUI. For å få inn vår logo i header og body benyttes NGUI atlas tool. Det samme gjelder de ikonene vi skal ha på knappene, for å gjøre det mer ryddig har vi puttet flere ikoner inn i samme atlas. Knappene skal benytte standard NGUI buttons oppsett med rosa bakgrunn. Når dette er på plass opprettes det prefabs og vi drar så de forskjellige delene over på disse. Alt skal være skalert etter størrelsen vi har valgt, 1920x1080.

Når prefabene er laget er det kun å dra de inn på GUI panelet som er generert. Knappene kan flyttes etter eget ønske. Se figur5 for hvordan det vil se ut.



Figur5: StartScreen laget med drag and dropp prefabs.

3.2.2 Vurderinger

3.2.2.1 Type startscreen

For å kunne gi brukeren en god opplevelse og samtidig ha en enkel meny. Har vi vært inne på flere løsninger om hvordan vi vil at brukeren skal bli møtt når spillet startes opp. Det er hovedsakelig tre typer som har vært diskutert. Møter logg inn, registrering og en startscreen som binder de to sammen. Alle tre har fordeler og ulemper. Det vi kom frem til er at enten rett til logg inn eller en scene som binder de sammen er mest aktuelt.

For å få en enklest mulig meny tok vi en beslutning på å ha en startscreen der brukeren selv kan velge om man vil registrere eller logge inn. Dette vil gi brukeren en god og oversiktlig møte med menyen i spillet.

3.3 ID: 133 - New User

Denne userstoryen er mindre da alt av prefabs er laget fra før, slik at jobben med layout blir mindre. I Denne skal det legges til mulighet for å registrere en ny bruker. Dette innebærer å legge til funksjon på ny bruker knappen i startscreen. Deretter skal det lages en ny scene hvor brukeren er nødt til å fylle ut nødvendig informasjon.

3.3.1 Implementasjon

Opprett ny scene og slett main camera. Bruker deretter NGUI tool for å opprette nytt 2d UI. Deretter dras prefabene som har blitt laget i en tidligere user story inn på GameObjectet som heter Panel. Da har vi fått den layouten vi ønsker.

Da layouten er på plass oppretter vi alle widget som må på plass ved hjelp av widget toolen. Ut ifra vurderinger som har blitt tatt skal det opprettes fire inputfelt og to knapper. Dette vil være navn, brukernavn, passord og passord igjen som inputfelt. Knappene vil være til å lage brukeren og eventuelt gå tilbake. Alle inputfeltene har et script lagt til som holder styr på hva som blir skrevet der. Panelet skal ha et UserReg script som styrer alle knappene og henter inn det som står i tekstfeltene. Dette scriptet skal sjekke om alle verdiene er gyldige. Dette gjøres ved hjelp av if-setninger. Det skal også sjekke hvilke tilbakemelding man får fra databasen før man sender brukeren videre. Dette gjøres ved å sjekke hva som blir returnert fra databasen når man prøver å legge inn brukeren.

Figur6: Registrer ny bruker scene.

Når brukeren sendes videre vil brukeren sendes til scenen der man skal logge seg inn. Denne scenen er beskrevet der vi skriver om ID 202 – Startupscreen scene userstoryen.

3.3.2 Vurdering

3.3.2.1 Data

En viktig vurderings sak er, hva skal registreres og hvordan skal det registreres. På grunn av sikkerhet har vi valgt å gå for en modell der vi registrerer minst mulig om brukeren. Dette er med tanke på personvern. Ved å gjøre dette slipper vi også unna å være nødt til å kryptere databasen. Det vi kom frem til som absolutt nødvendig informasjon var navn, brukernavn og passord. Navn har vi valgt å ta med for at administrator lett skal kunne knytte bruker mot person. Navnet til brukeren vil aldri forekomme i spillet.

Det skal også registreres hvilke deler av skjermen brukeren vil benytte seg av. Samt hvilke avatar som blir brukt. Vanskelighetsgraden spilleren velger å ha lagres også i databasen.

3.3.2.2 Skal scenen deles opp?

For å få en ryddig registrerings prosess var vi nødt til å vurdere hva og hvor mye som skal vises om gangen. Har vi alt på en scene blir det fort trangt og uoversiktlig. Dermed ville et av kravene til spillapplikasjonen feile. Ut ifra dette fattet vi en beslutning at vi vil i første omgang kun registrere navn, brukernavn og passord. Deretter vil vi sende brukeren til logg inn.

Vi fattet enda en beslutning at tilpasningen kan brukeren selv gå inn og stille på dersom det skulle være ønsket. Dette vil da være en funksjon i hovedmenyen.

3.4 ID:134 – Song Select

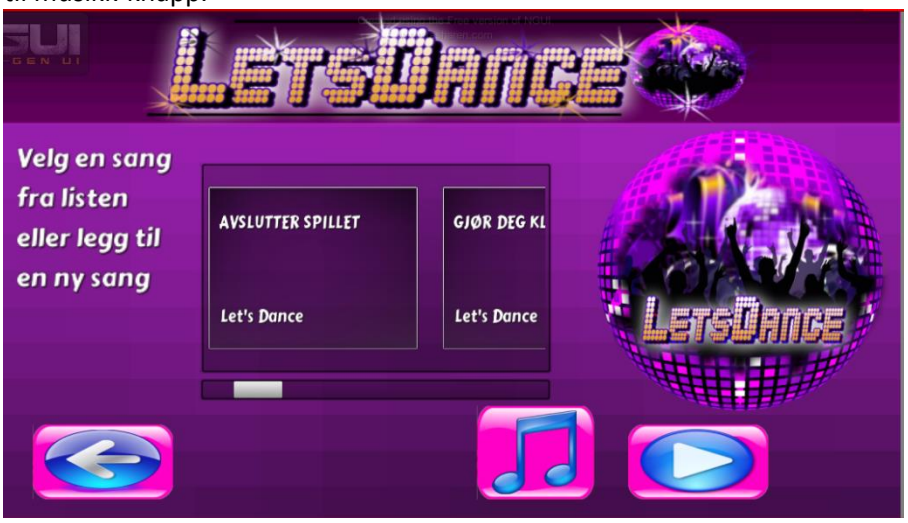
I userstory 134 skal det legges til rette for at brukeren skal kunne velge sang. I tillegg til dette skal det legges til rette for at brukeren skal ha mulighet til å kunne legge til egne sanger.

3.4.1 Implementasjon

Følger standard oppsett for å få standard layout. I denne scenen er det en noe komplisert GUI widget. Dette er en dynamisk scrollist der det blir lagt til knapper. Disse knappene kan brukeren trykke på og de representerer sanger som er lagt til i systemet.

Scrollbaren lages ved at vi lager to nye paneler. Det ene panelet skal inneholde to boxcollidere, disse skal stå på hver sin side av panelet. Dette panelet skal ha en tilhørende ScrollBar, denne lages ved NGUI widget tool. Panelet skal også ha en outline som har UIDragablePanelContent script lagt til. Det andre panelet skal ha en Grid som child, med scriptet UIGrid dette scriptet vil legge alle elementene etter hverandre i en liste. I tillegg til dette legger vi på et SongList script som skanner igjennom alle sangene og legger de til som et child av griden. Dermed blir de lagt til i grid listen.

SongList scriptet skal "spawne" prefabs som har blitt laget. Denne prefaben har alle nødvendige scripts og relasjoner forhånds definert. Prefaben er en modifisert knapp som inneholder to labels, for artist og sang navn. I tillegg til dette skal scenen inneholde en play knapp, en tilbakeknapp og en legg til musikk-knapp.



Figur7: Song Select scene.

3.4.2 Vurdering

3.4.2.1 Import

For å importere sangene var vi nødt til å ta en vurdering av hvordan vi skulle løse det. Da det er vanskelig å få til at brukeren navigerer seg fram i runtime. Dette viste seg å være en såpass stor oppgave at vi besluttet å legge den til en egen user story. Løsningen vi ser for oss er at vi lager en mappe hvor brukeren kan legge sangene sine.

3.4.2.2 Generering av dans

Spillet skal ha en dansegenerator. Vi har besluttet at det er i song select vi er nødt til å generere dansene. Så en vurdering vi var nødt til å ta i forbindelse med det er hvor vi skal generere dansene. Vi

har to valg vi så for oss. Når listen lastes inn kan vi sjekke om sangen har fått en dans, hvis ikke så generer vi. Eller vi kan sjekke når brukeren trykker play for også generere. De genererte dansene lagres i databasen med tilhørende sang navn, mer om det der databasen er beskrevet.

Vi ser for oss at den sist løsningen er den mest optimale. Det er fordi vi vil minske trafikken til databasen og vi vil slippe å ha mye trafikk på den.

3.5 ID: 191 – Musikk mappe

I userstory 191 har vi gjort slik at brukeren kan legge til egne sanger og de vil dukke opp i listen over sanger som er tilgjengelige. Vi har da en mappe alle sangene blir lagt til og scriptet som laster inn sanglisten legger de til i listen inne i spillet.

3.5.1 Implementasjon

For denne userstoryen blir det ikke laget noen ny scene. Vi benytter scenen som ble laget til Song Select. I første omgang laget vi en mappe som brukeren kunne legge sangene i. Sangene lastes inn ved at når Song Select scenen lastes vil songlist scriptet gå igjennom mappen og legge til filene. Dette gjøres ved at spillapplikasjonen finner filbanen til mappen uansett hvor spillapplikasjonen måtte ligge. Dette gjøres med Application.dataPath kallet som er innebygget i unity. Deretter finner vi informasjonen til alle filene i mappen og behandler de som nødvendig. Det vil si fjerne blant annet filendelsen. For hver fil vil en prefab til lagt til og vi får listen beskrevet i user story ID: 134 – Song Select.

For å gjøre jobben med å legge til musikk for brukeren enklere, har vi implementert muligheten til å bruke en filutforsker som vil komme opp dersom legge til musikk knappen blir trykket. Dette er gjort ved å bruke EditorUtility.OpenFilePanel. Når filen er valgt kaller vi på SongListUpdate(sSongName) for å oppdatere sanglisten.

3.5.2 Vurderinger

3.5.2.1 Mappestruktur

Hvordan mappe strukturen skal bygges opp og fungere er noe vi måtte ta en grundig vurdering av. Mappen må være enkel å komme frem til for brukeren. Det vi kom frem til er at vi har en mappe som heter Sanger der brukeren kan legge til sine sanger. Denne mappen har vi valgt å legge helt ytterst i mappestrukturen til applikasjonen.

3.5.2.2 Filutforsker

For å gjøre spillapplikasjonen enda enklere for brukeren har vi hat en vurdering om vi skal implementere en filutforsker, slik at brukeren kan bla seg frem til sangene sine. Det er ingen filutforsker som er innebygget som kan brukes når vi bygger spillapplikasjonen. Ved å kjøre det i unity editoren kan vi velge å bruke EditorUtility.OpenfilePanel. Dette kallet går ikke å bruke i en standalone spillapplikasjon.

For å kunne få en filutforsker i en standalone spillapplikasjon er man nødt til å lage den selv, eller gå til innkjøp av en asset som har implementert dette. En asset man kan kjøpe og gjør det man trenger er UniFileBrowser[2].

Derfor har vi for vår prototype valgt å gå for en løsning der man i editoren vil kunne benytte seg av filutforskeren, men i en standalone spillapplikasjon må man legge sangene inn i mappen selv. Det vil fremdeles være det samme scriptet.

3.6 ID: 201 – Change Password Scene

I userstory ID: 201 – Change Password scene har vi lagt til rette for at brukeren kan bytte passord på brukeren sin. Denne scenen knyttes sammen med databasen igjennom gameobjektet som skal gå igjennom hele spillet.

3.6.1 Implementasjon

I denne userstoryen opprettes det en scene. Denne scenen lages ved å følge standard oppsett, da layouten skal være lik. Av GUI widgets som er med i denne scenen er det, to knapper, tre inputfelt og to labels. Knappene er for å kunne gå tilbake i menyen og bytte passord. De tre inputfeltene skal være for gammelt passord, nyttpassord og nyttpassord igjen. Som de andre inputfeltene vil det være auto genererte NGUI script knyttet til dem. Disse scriptene vil holde styr på hva som har blitt skrevet i dem.

Panel gameobjektet i denne scenen vil ha ChangePasswordController scriptet knyttet til seg. Dette scriptet vil holde styr på hva som skal utføres når tilbake eller bytte passord knappene blir trykket. Tilbakeknappen skal rett og slett bare gå tilbake ved å laste inn forrige scene, som er scenen der brukeren vil ha muligheten til å velge mellom en rekke andre innstillinger. Denne scenen er beskrevet under ID 203 – Tilpasse scene.

Når bytte passord knappen blir trykket vil metoden OnChangePasswordClicked bli kjørt, denne vil igjen sette i gang ChangePassword(). Denne metoden henter inn informasjonen som er skrevet i inputfeltene. Deretter vil denne informasjonens sjekkes, er nytt passord likt gammelt, er nytt passord likt nytt passord på nytt og er passordet kortere en 6 langt. Er alt greit så bytter vi passordet i userID objektet som er på DoNotDestroy, gameobjektet som går igjennom hele spillet. Mer om DoNotDestroy der ID: 195 – DoNotDestroy er dokumentert.

Figur8: Password change scene.

3.6.2 Vurderinger

3.6.2.1 Kriterier for å bytte passord

For å kunne bytte passord var vi nødt til å ta en vurdering av hva som skulle til for at et passord bytte var greit. Hvor langt må passordet være, kan det være likt som det gamle, en bekreftelse på om brukeren kan sitt nye passord? Alle disse spørsmålene var vi nødt til å vurdere. Det vi kom frem til var at det nye passordet ikke kan være likt det gamle. Denne beslutningen tok vi fordi vi ikke ser noen hensikt i at en bruker skal bytte passordet sitt til det samme. På andre spørsmålene bestemte vi oss for at vi må ha de samme kriteriene som når en bruker blir opprettet.

3.6.2.2 Hvordan skal det nye passordet lagres

Når vi skal lagre det nye passordet til brukeren er vi nødt til å ta en vurdering om hvordan vi skal lagre det. Vi har her to muligheter, den ene er å lagre det direkte i databasen. Den andre løsningen er å bytte det i userID objektet som er plassert på DoNotDestroy.

Begge løsningene ville fungert, men for å slippe mye trafikk på databasen kom vi frem til at å forandre det i userID objektet ville være det beste.

3.7 ID:202 – Startupscreen scene

Denne userstoryen omhandler scenen som brukeren vil møte når spillapplikasjonen blir startet. Logg inn scenen blir også laget her.

3.7.1 Implementasjon

Både StartUpScreen og logg inn skal ha den standard scene oppbygningen og layouten. Det vil ikke være noen kompliserte GUIwidgets i noen av disse scenene. Alle knappene som skal være med opprettes ved hjelp av knappe prefabene som er laget. Inputfeltene som skal være i logg inn scenen opprettes ved hjelp av NGUI widget tool.

I StartUpScreen er det kun ett script som blir brukt og det er et kontroller skript som sender brukeren til riktig scene når knappene blir trykket. Knappene som er i StartUpScreen scenen er logg inn og registrer ny bruker. De skal ta brukeren til henholdsvis logg inn og registrer ny bruker scene.

Figur9: Startupscreen scene.

Logg inn scenen består av to inputfelt og to knapper. Inputfeltene er for at brukeren skal kunne skrive inn brukernavn og passord. Disse har NGUI script knyttet til seg slik at LoggInnButtonScript som holder styr på hva som skjer når kan hente inn hva brukeren har skrevet. Scriptet sender brukeren tilbake til tidligere scene dersom tilbake blir trykket. Dersom brukeren trykker på logg inn vil vi sjekke med loginUser metoden i dbHandler. Denne metoden sjekker om brukeren eksisterer, om passordet er riktig. Dersom brukernavn og passord stemmer vil userData bli bunnet til DoNotDestroy. Brukeren vil deretter bli sendt til hovedmenyen. Dersom brukernavn og passord ikke stemmer vil brukeren få utskrevet en feilmelding.

Figur10: Logg inn scene.

3.7.2 Vurderinger

Vurderingen av hvilke type startscreen spillapplikasjonen har er beskrevet under ID 113: Meny. Den vurderingen blir derfor ikke beskrevet her.

3.7.2.1 Hvordan sjekke brukernavn og passord

For å kunne sjekke passord og brukernavn var vi nødt til å ta en vurdering av hvordan vi skulle gjøre det. Det finnes mange metoder vi kunne brukt for å sjekke brukernavn og passord med. Spesielt med tanke på hva som skal bli returnert fra databasen. Løsningen vi kom frem til er at vi sender brukernavn og passord til databasen. Den vil da returnere en id dersom alt stemmer og laster inn data på userData objektet. For en grundig forklaring på hvordan den fungerer se dbHandler.

3.8 ID: 203 – Tilpasse scene

I denne userstoryen laget vi scenen som tillater brukeren å bestemme hvilke deler av skjermen som kan brukes til dansingen. Slik at brukeren kan selv tilpasse spillet til sitt behov. I tillegg til denne scenen har vi laget en innstilling scene. På denne måten får vi en scene der brukeren får en fin oversikt over innstillingene som kan forandres på.

3.8.1 Implementasjon

I begge scenene bruker vi standard layout og oppbygning. Tilpasse scenen består av en tilbakeknapp og fire avkryssingsbokser. For tilbakeknappen benytters tilbake prefab og til avkryssingsboksene benytter vi oss av NGUI sine avkrysningsbokser. Disse boksene vil ha et NGUI checkbox script tilknyttet. Dette scriptet gir oss muligheten til å sjekke om boksene er avkrysset eller ikke. Som kontroller script i denne scenen har vi et script SettingScript. Scriptet tar seg av hva som skjer når knappene blir trykket. Tilbakeknappen i denne scenen tar seg av å lagre hva brukeren har valgt før brukeren blir sendt tilbake. Først blir eksisterende innstilinger hentet. Vi plaserer deretter hver char i et array. Deretter sjekker vi hver enkelt avkryssingsboks om de er avkrysset, dersom de er det forandres tilhørende verdi til 0. Stringen settes sammen igjen og lagres. Vurderingene og en grundig beskrivelse av prosessen finnes der dbHandler blir forklart.

For å kunne bestemme hvilke bokser som skal være avkrysset når scenen lastes bruker vi et CheckBoxDecider script. Dette scriptet henter inn nåværende innstillinger og sjekker hver boks om de skal være avkrysset eller ikke.

Figur11: Tilpasse scene.

Innstilling scenen er meget enkel og består kun av knapper og et script som holder styr på hvor brukeren skal sendes når de trykkes.

Figur12: Settings

3.8.2 Vurderinger

3.8.2.1 Lagre tilpasning

Hvordan vi skal lagre tilpasningen til brukeren er noe vi vurderte nøye. Skal det lagres som int, string, bool. Vi endte da opp med å lagre innstillingene i en string bestående av 1 og 0. slik at vi enkelt kunne sjekke hvilke del av skjermen som er tillat å bruke. Selve lagringsprosessen blir beskrevet der vi beskriver dbHandler.

3.8.2.2 Tilpasnings metode

Hvilke type tilpasning skal det være, og hvordan skal vi kunne registrere det på en enkel måte. Her måtte vi ta en vurdering på hvor mange deler vi ville dele opp skjermen. For å dekke kravene våre fant vi det hensiktsmessig å dele skjermen inn i fire. Slik at en bruker som kun kan bruke overkroppen kan velge det om de vil.

Deretter måtte vi vurdere hvordan vi ville registrere hvilke tilpasning som var ønsket. Det beste hadde vært en automatisk deteksjon av dette, men slik teknologien er i dag så vi ikke dette som en reel løsning. Det vi kom frem til da er at vi bruker avkryssingsbokser, slik at brukeren selv enkelt kan

se hva som er valgt. Jobben med å registrere hva brukeren ønsker blir også enklest mulig med å benytte avkryssingsbokser.

3.9 ID:205 – Statistikkoversikt scene

Denne userstoryen omhandler scenen der brukeren har en oversikt over hvilke statistikker han kan velge mellom.

3.9.1 Implementasjon

Denne scenen er bygget opp på samme måte som ID:134 – Song Select. Hvor brukeren får en dynamisk liste over sangene. Brukeren kan ut ifra den velge hvilke sang han vil se på og deretter trykke på statistikknappen og se progresjonen på den dansen.

For en bedre beskrivelse om oppbygning av scenen se ID:134 – Song Select avsnittet. Den eneste forskjellen er at det er et annet kontroller script. I denne scenen er det StatistikkController som styrer hva som skjer. OnSongClicked() metoden er helt lik den i Song Select og statistikknappen fører brukeren til en scene der statistikken blir fremvist.

Figur13: Statistikk select scene.

3.9.2 Vurderinger

3.9.2.1 Hva slags statistikk skjerm

For å kunne gi brukeren en god måte å velge hva slags statistikk som skal vises er vi nødt til å ha en scene der brukeren lett kan få en oversikt. Etter mye vurdering kom vi frem til at vi skal ha en scene der brukeren kan velge hvilke sang man vil se statistikk over. Dette er en fin måte å sørge for at brukeren på en god og oversiktlig måte kan se den statistikken han vil.

3.10 ID: 181 – Funksjon – Stolpediagram

3.10.1 Implementasjon

User Story 181 omhandler statistikk i form a stolpediagrammer. Stolpediagrammets utseende ble laget ved å bruke en plugin til Unity3D kalt NGUI, som inneholder en hel del grafiske stolper, bakgrunner, labels, knapper osv.

For å lage scenen for statistikk tok vi utgangspunkt i en av meny-scenene som allerede var laget ved å bruke NGUI, så designet på scenen var allerede ferdig.
Det ble lagt til en "SlicedSprite" som bakgrunn for stolpene. Stolpene ble laget ved å bruke "sliders". For å skille søylene fra hverandre, brukte vi forskjellige farger. Vi lagde to piler, en i X-retning og Y-retning, for å vise maksimal høyde og hvor langt bort i x-retning stolpene strekker seg.

Poengscore går fra 0 til 200. Og blir presentert i form av stolpenes høyde. Poengscoren står også som en label under hver stolpe.

Over stolpene er det en label for hver stolpe, som viser hvilken dato dansen er utført.

Vi la også til en label som viser navnet på sangen.

3.10.1.1 RandomVal.cs

I RandomVal hentes poengscoren fra databasen og "settes" inn i stolpene. Det er en for-løkke som oppretter og setter inn verdien for stolpene. Hvis brukeren kun har danset to ganger, tegnes du kun to stolper. Dette er også gjort for labelene i form av poengscore og dato.

3.10.2 Vurderinger

3.10.2.1 Antall stolper

Siden dette er en prototype, valgte vi å ha 10 stolper. Siden statistikken vises etter navnet på sangen, tar vi i betrakting at samme sang ikke vil bli danset til mer en 10 ganger.

3.10.2.2 Poengscore i tall

Vi fant det fornuftig å vise poengscore i form av tall under stolpene for å vise scoren mer konkret. Stolpene fungerer som en grafisk formening om hvordan utviklingen har endret seg fra gang til gang.

Figur14: Statistikk scene.

3.10.2.3 Dato

Det var utfordrende å finne en måte å presentere hvilken dato dansen er utført på. Dette løste vi ved å kutte bort klokkeslett for utførelsen, og kun vise dag/måned/år.

3.11 ID: 195 – DoNotDestroy

I denne userstoryen skal det ikke lages noen scene. Dermed er det ingen layout arbeid i denne storyen. Hensikten med denne er å lage et gameobjekt som ikke blir ødelagt når man går igjennom de forskjellige stadiene i menyen.

3.11.1 Implementasjon

Først oppretter vi et gameobjekt som er tomt. Oppgaven til dette objektet er å ta vare på data igjennom spillet. Dette er all data som vi på et eller annet tidspunkt vil få bruk for, da spesielt om brukeren sine innstillinger. For å få gameobjektet til og ikke bli slettet når vi bytter scene laget vi et

script vi festet på gameobjektet. Scriptet DoNotDestroy kaller DontDestroyOnLoad() dersom det er første gang objektet blir laget. Dette metodekallet gjør at gameobjektet ikke blir slettet ved Application.LoadNextLevel().

Variabler som vi legger til gameobjektet er iDificulity, iTilpassing, userData, sUserName, sSongName, sPassword, bCreated. Disse variablene vil bli tatt vare på helt til spilleren logger ut. userData er data som er hentet fra databasen hvordan det foregår er beskrevet der UserData er beskrevet.

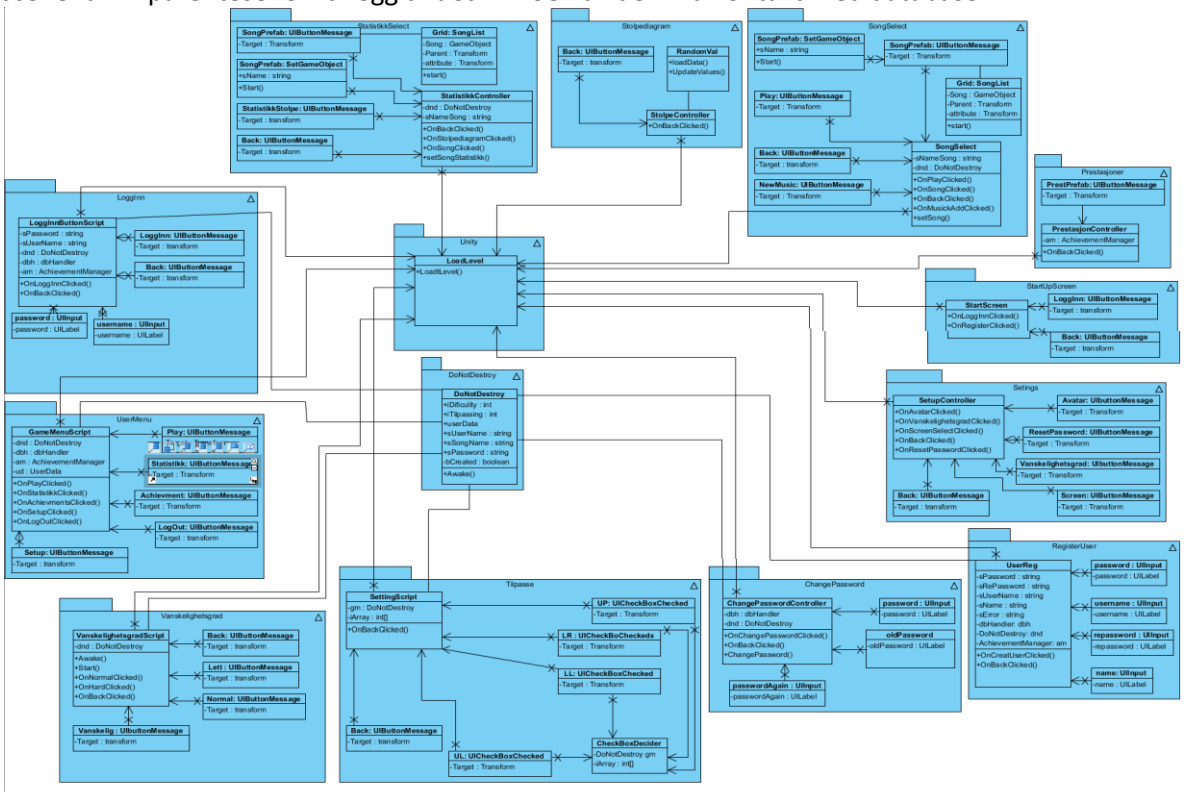
3.11.2 Vurderinger

3.11.2.1 Hvilke data skal være med

For å slippe for mye trafikk på databasen har vi sett oss nødt til å ta en vurdering av hva slags data kommer vi til å bruke igjennom hele spillet. På denne måten vil vi kunne knytte de til gameobjektet når brukeren logger inn. De variablene vi trenger å vite er vanskelighetsgrad, tilpasning, brukernavn, passord altså generell bruker data.

4. Klassediagram

Dette klassediagrammet viser oversikten over klassene som er involvert i menyen. Alle pakkene i diagrammet representerer en scene i menyen. Alle klassene som har navn: UIButtonMessage skal være knappet som vil kalle på kontroller scriptet i tilhørende scene. Det som er felles for alle kontrollerscriptene er at de på ett eller annen tidspunkt kaller på LoadLevel(). Der det skal stå scenenavn i parentesene. I tillegg til det vil noen av dem ha kontakt med databasen.



Figur14: Klassediagram.

5. Referanser

1. ***NGUI: Next-Gen UI kit.*** **2013.**
2. ***UniFileBrowser*****. 2013 [cited 2013 21.05.2013]; Available from: https://www.assetstore.unity3d.com/#/content/151.**

dbHandler

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Første utkast	14.02.2013	Dagfinn Kjærnet
2.0	Oppdatert	19.03.2013	Dagfinn Kjærnet
3.0	Oppdatert tekst, og nytt DB design.	10.05.2013	Dagfinn Kjærnet
4.0	Oppdatert diagrammer og tekst	22.05.2013	Dagfinn Kjærnet
5.0			
6.0			

Kontrollert av:	MJ, JEH

1. Userstories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
94	Akritektur: Database - Kontakt med database	Dagfinn
96	Akritektur: Database - Lese fra database	Dagfinn
98	Akritektur: Database - Skrive til database	Dagfinn
121	Arkitektur: Grunnleggende database design	Dagfinn
123	Arkitektur: Database, grunnleggende metoder.	Dagfinn
207	Funksjon: Database - Flere funksjoner	Dagfinn

2. Innledning

Vi trenger en database til spillet vårt, slik at brukere kan følge med på sin egen progresjon og ikke mister denne mellom spillsesjoner. Vi benytter også databasen til å lagre musikken vi legger til og dansene som genereres, slik at de samme dansene spilles hver gang og det ikke må genereres nye danser hver gang en sang lastes.

3. Implementasjon

Vi trenger et script som kan håndtere kontakten med databasen. Vi kommer til å feste dette scriptet til et permanent GameObject slik at det kan benyttes gjennom hele spillet og man slipper å opprette nye instanser av dette scriptet.

Vi har valgt å kalle scriptet dbHandler.cs og de fleste metodene vil ha i oppgave å generere forskjellige SQL spørringer som kjøres mot databasen. Vi må selvfølgelig også ha en database og starter med å designe denne. Da må vi først analysere hva vi ønsker å lagre i databasen.

3.1 Vurderinger

Vi må gjennomgå en del vurderinger, både i forhold til scriptet og i forhold til designen av databasen. Vi mener det vil være lønnsomt å designe databasen først, siden vi da kan rette oss etter dette når vi skal designe scriptet.

3.1.1 Vurderinger i implementasjonen av databasen

Vi ønsker å lagre følgende:

- Kontoinformasjon; brukernavn, passord
- Kontoinnstillinger; hvilken avatar har bruker valgt, hvilke tilpasningsinnstillinger har vedkommende valgt(i vårt tilfelle, hvilke deler av danse-griden skal aktiveres) og hvilken vanskelighetsgrad har spilleren nådd.
- Hvilke prestasjoner har spilleren utført.
- Statistikk; ønsker å lagre highscore og diverse andre data som kan hentes ut og presenteres for å vise progresjon.
- Musikk og dans; Vi ønsker en liste over sanger brukerne har importert og de tilhørende dansene slik at det ikke genereres ny dans mellom hver gang brukeren spiller.

Vi har bestemt oss for at vi kun benytter databasen til ting som endres i runtime, selv om vi på noen av områder kunne benyttet oss av databaser for å gjøre ting mer oversiktlig og fremtidsrettet.

Vi har valgt å hardcode en del data i forhold til prestasjoner og dansebevegelser. Vi kunne godt lagret all data om prestasjoner i databasen, men da vi ikke kommer til å ha så veldig mange av dem i vår prototype velger vi å følge rammeverket[1] vi skal benytte oss av som har hardcodet dem. På denne måten kan prestasjonene enkelt redigeres direkte i unity istedenfor at de må legges til i databasen. Det er for øvrig relativt lett å tilpasse dette senere slik at prestasjonene også hentes ut fra databasen. Rammeverket støtter ikke lagring av progresjon og vi har derfor lagt til metoder som muliggjør dette og denne dataen lagres så i databasen.

Siden vår prototyp ikke kommer til å ha svært mange dansetrinn velger vi å hardcode disse, slik at vi lett kan gjøre endringer direkte i unity eller mono istedenfor å måtte legge til og endre ting i databasen. Vi ønsker for øvrig å lagre de sammensatte dansene til hver sang slik at brukeren får servert den samme dansen hver gang, og ikke må lære seg en ny dans mellom hver sesjon.

3.1.2 Valg og vurderinger i implementasjonen av databasehåndteringsscriptet.

Vi har valgt å lage egne variable for alle kolonne- og tabellnavn, selv om dette i praksis betyr at man blir sittende svært lenge å "kopiere" databasestrukturen vil det på sikt gjøre det lettere å tilpasse scriptet dersom vi gjør endringer i databasestrukturen.

Vi ønsker å ha en egen metode som kun har som oppgave å skrive til databasen, det vil si at alle metoder som skal skrive noe til databasen. I praksis kun har som oppgave å generere SQL spørringene og deretter sende disse til skrivemetoden. Vi valgte denne løsningen fordi en skrive-til-db spørring ikke trenger å gi oss noe data tilbake, utover en eventuell feilmelding dersom skrivingen ikke var vellykket.

Når vi skal lese fra databasen har vi derimot valgt å gjøre dette direkte i metodene da vi forventer en del forskjellig informasjon og hver return må tilpasses, og vi ønsker i noen tilfeller detaljerte feilmeldinger.

3.2 Utfordringer og løsninger

Her beskrives utfordringer vi møter i forbindelse med implementasjonen av databasen og databasehåndteringsscriptet.

3.2.1 Utfordringer under implementasjonen av databasen

Implementeringen av databasen har vært litt tungvinn da vi ikke har funnet noen gode design-verktøyer tilsvarende dbDesigner[2]. Når vi genererer SQL scriptet for automatisk oppretting av databasen og prøver å kjøre disse mot SQLite får man en feilmelding fordi mySQL syntaxen er litt annerledes enn SQLite syntaksen.

3.2.2 Løsninger

For å lage selve designet har vi benyttet oss av dbDesigner, deretter bruker vi tabellene dbDesigner genererer for å manuelt opprette databasen via en SQLite behandler kalt SQLite manager[3].

3.2.3 Utfordringer under implementasjonen av databasehåndteringsscriptet

I starten var det svært vanskelig å feilsøke da SQLite sjeldent gir tilbake feilmeldinger og hele systemet var veldig nytt og fremmed. Ofte var det vanskelig å vite om problemet lå i bruken og rekkefølgen av metodene man kjører eller om det rett og slett var pga SQL-syntax feil, men etter hvert som man jobber mer med det får man heldigvis erfaring og man ser lettere hvor feilen kan ligge. Vi møtte også på utfordringer da vi skulle bygge prosjektet da vi ikke lenger fikk kontakt med databasen, men dette løste seg selv da vi bygget på en annen pc og var nok relatert til PCen og ikke spillet.

3.2.4 Løsninger

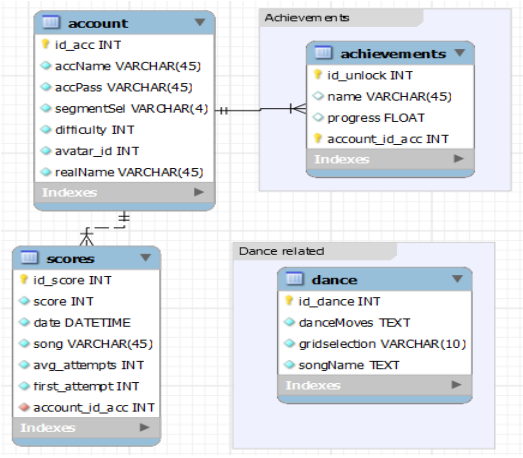
Man må skrive gode rutiner for debugging slik at man lettere kan følge og lokalisere feilen. Prøv å bygge spillet på en annen maskin dersom det oppstår problemer, pass på at man har oppdatert maskinens programvare(windows update, .NET updates osv).

4. UML og Database struktur

Denne delen av dokumentet inneholder UML diagrammer og tabeller for å beskrive implementasjonsløsningene.

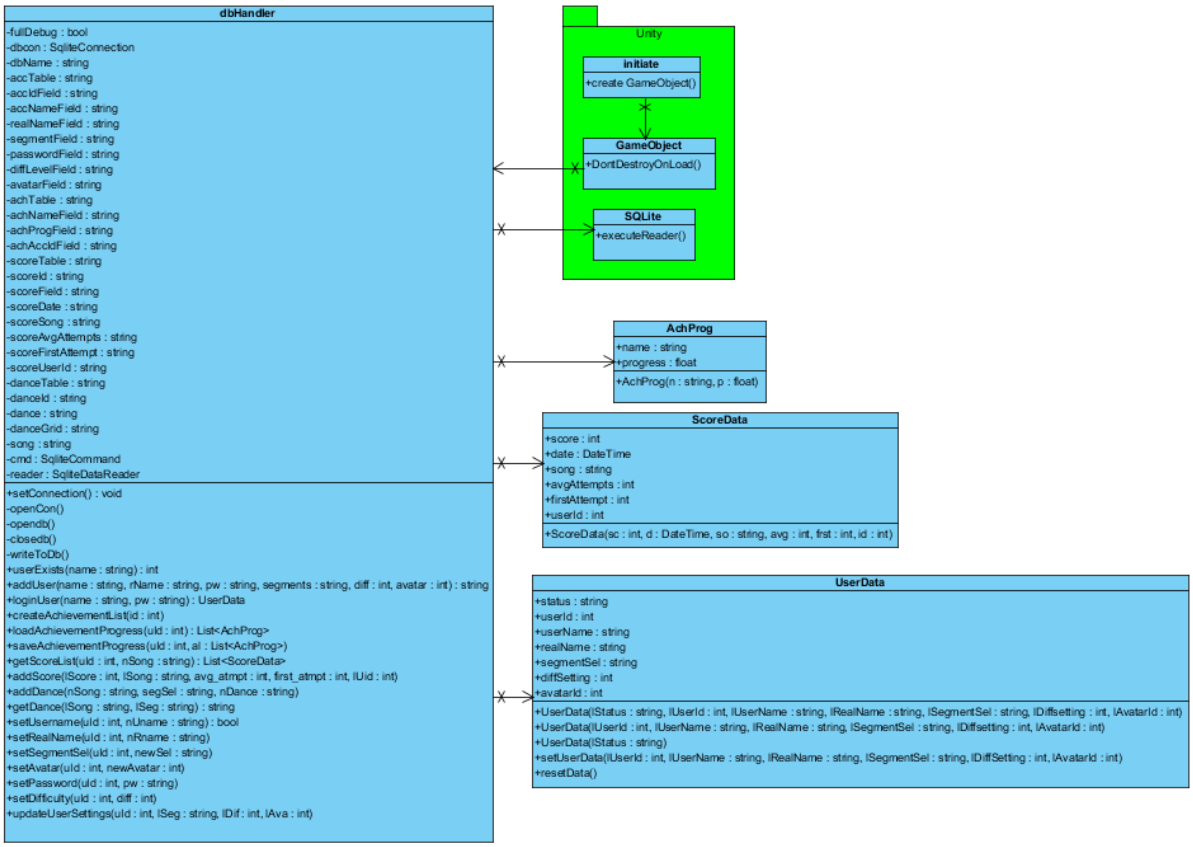
4.1 Database struktur

Vi benytter oss av følgende databasestruktur.



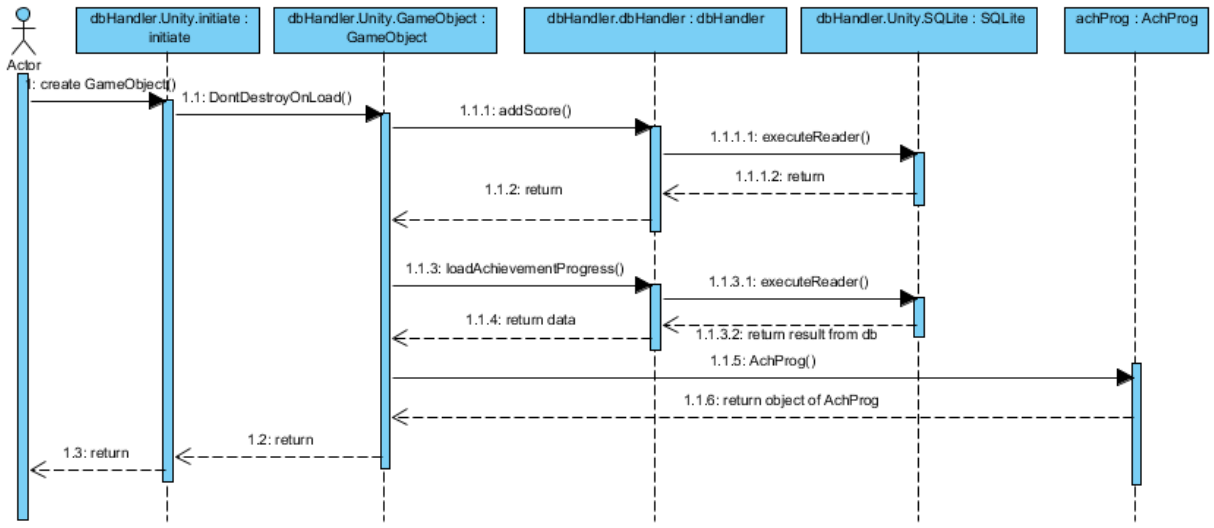
4.2 Klassediagram for dbHandler.cs

Vi fester databasehåndteringsscriptet til et GameObject i unity, dette objektet eksisterer mellom alle scener og er alltid tilgjengelig. Andre script får derfor tilgang til dbHandler gjennom dette objektet og kan på denne måten skrive til og lese fra databasen. Dbhandler benytter seg av SQLite til å skrive til og lese fra databasefilen.



4.3 Sekvensdiagram

Sekvensdiagrammet beskriver prosessflyten ved bruk av dbHandler, jeg har kun laget dette diagrammet for to av prosessene(addScore og loadAchievementProgress) da de andre er svært like.



5. Design prosess

Til vår første prototype har vi implementert metodene som er nødvendige for å legge til og logge inn brukere.

Etter dette har vi implementert de resterende metodene fortløpende og alt som er beskrevet her i designdokumentasjonen er på dette tidspunktet implementert og testet.

6. Ordliste

UML - Modellereringsspråk (Unified Modelling Language)
GUI - Graphical User Interface
SCRUM - Prosjektmodell
Script - I Programmering: Sekvens med instruksjoner som blir tolket og utført

7. Referanser

1. Gargolinski, S. *Progress - a free achievement framework for unity*. 15.01.2013]; Available from: http://www.stevegargolinski.com/progress-a-free-achievement-framework-for-unity/.
2. fabforce. *General Information - What is DBDesigner 4?* 14.03.13]; Available from: http://www.fabforce.net/dbdesigner4/.
3. *sqlite-manager*. 19.03.2013]; Available from: http://code.google.com/p/sqlite-manager/.

Prestasjoner

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Første utkast	04.04.2013	Dagfinn Kjærnet
2.0	Oppdatert	30.04.2013	Dagfinn Kjærnet
3.0	Oppdatert tekst, og nytt design	22.05.2013	Dagfinn Kjærnet
4.0			
5.0			
6.0			

Kontrollert av:	ZA, JEH

Userstories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
175	Funksjon: Prestasjoner - Implementere rammeverk	Dagfinn
178	Funksjon: Prestasjoner - Legge til pop-up	Dagfinn
180	Funksjon: Prestasjoner - legge til databasestøtte	Dagfinn
207	Database - Flere funksjoner	Dagfinn

1. Innledning

For å gi brukerne ekstra motivasjon og for å øke levetiden til spillet vårt ønsker vi å legge til prestasjoner. Vi vet selv fra egen erfaring at prestasjoner øker både motivasjon, levetid og konkurranselysten. Prestasjoner er for tiden svært populært og er å finne i de fleste Playstation 3 og xbox360 spill samt de fleste nye PC spill. Det er rett og slett en veldig god og enkel måte å tilføre spillet veldig mye på en gang.

Vi kommer til å ta utgangspunkt i et gratis prestasjonsrammeverk som heter "Progress"[1], og legger selv til de metodene vi trenger utover dette.

2. Implementasjon

Vi begynner med å legge inn det originale rammeverket og sørger for å forstå hvordan det er bygget opp og fungerer, deretter bestemmer vi oss for hva vi ønsker å legge til.

2.1 Vurderinger

Vi må vurdere hvilke ekstra funksjoner vi ønsker å legge til i prestasjonssystemet. Blant annet støtter ikke rammeverket muligheten for å lagre progresjonen mellom spillsesjoner, og har heller ikke støtte for flere brukere. Vi ønsker også en visuell bekreftelse på at en prestasjon har blitt fullført, helst i form av en informativ pop-up beskjed som forsvinner automatisk etter en viss tid. Vi trenger altså følgende tilleggs funksjoner:

- Støtte for lagring av progresjon.
- Støtte for innlasting av tidligere progresjon.
- Støtte for å nullstille progresjonen når man bytter bruker.
- Støtte for popup beskjed når brukeren fullfører nye prestasjoner.

2.2 Utfordringer og løsninger

Her beskrives utfordringer vi møter i forbindelse med implementasjonen.

2.2.1 Utfordringer

Rammeverket er foreløpig kun lagt opp til å fungere i runtime, det vil si at all informasjon om progresjon forsvinner når spilleren avslutter spillet.

2.2.2 Løsninger

For å løse problemet med "runtime only" lagring benytter vi oss av databasen, og for å gjøre det lettere å håndtere dataene oppretter vi en ny klasse kalt "AchProg" som kun har to datafelt, ett for navnet på prestasjonen og ett for progresjonen. På denne måten kan vi sette inn all progresjonsdataene til en bruker i en ArrayList av AchProg objekter og sende denne til database scriptet som sørger for å få dataene lagret og knyttet til brukeren som sendte dem.

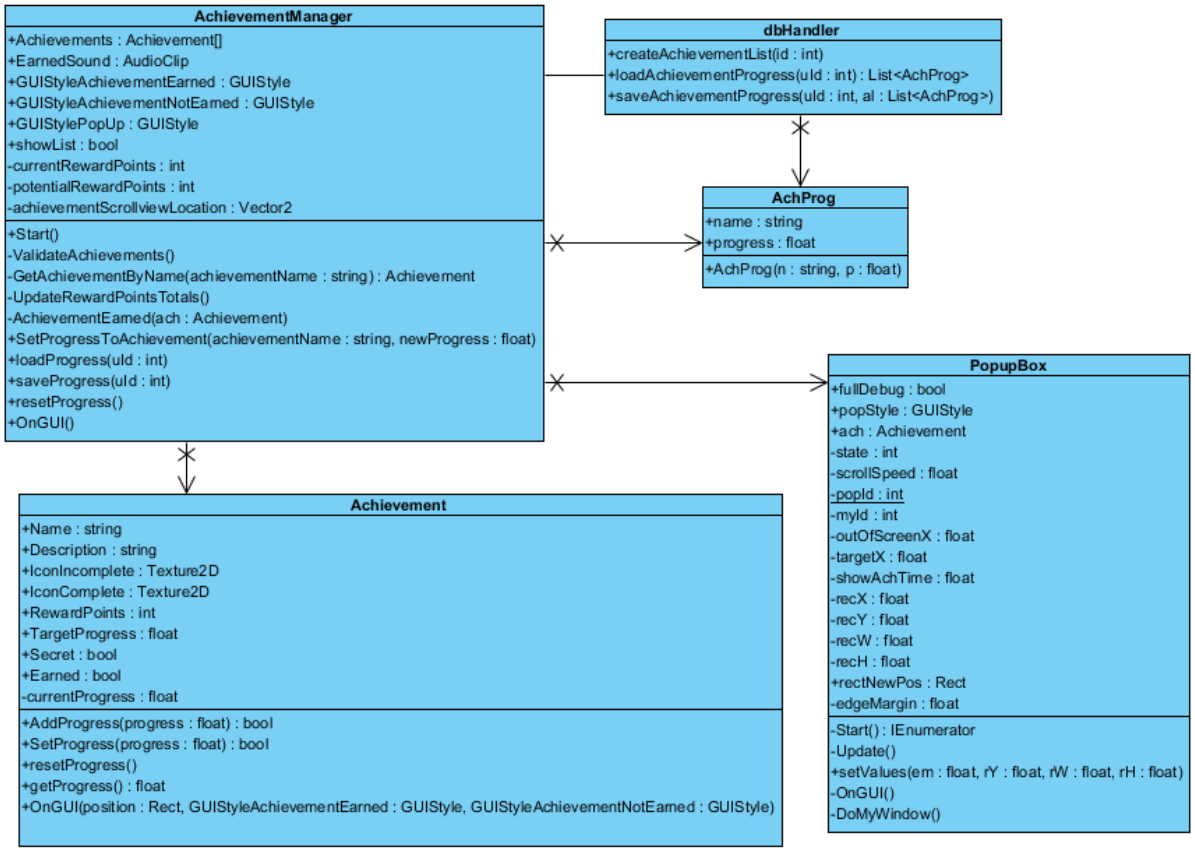
3. UML

Denne delen av dokumentet inneholder UML diagrammer og tabeller for å beskrive implementasjonsløsningene.

3.1 Klassediagrammer

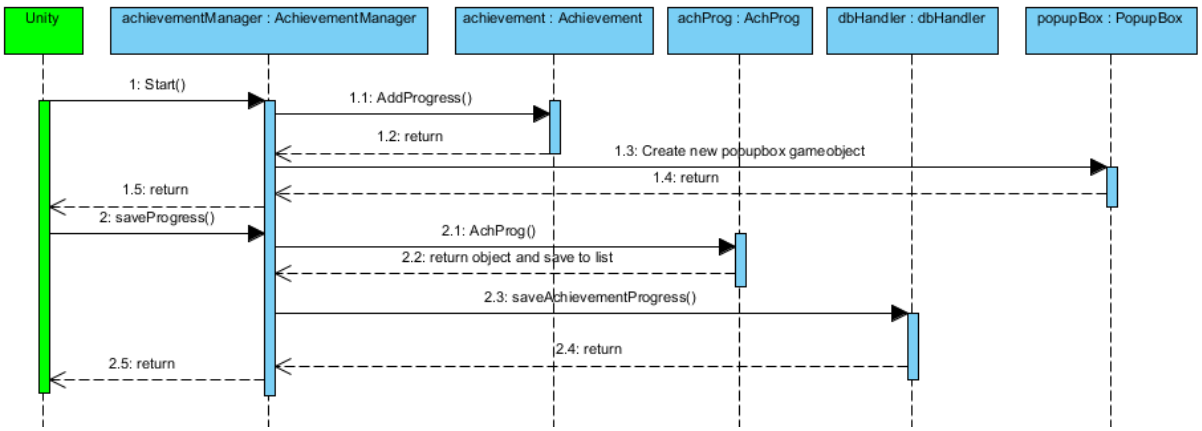
Achievement-klassen inneholder alle data for hver enkelt prestasjon, mens AchievementManager-klassen håndterer alle achievement objektene. Sistnevnte plasseres på et gameobject som ikke ødelegges mellom scenene. Ved innlogging kalles tilhørende metode fra dbHandleren for å laste inn progresjonen til brukeren som logger inn og realtime progresjonslisten oppdateres. Når brukeren så logger ut sendes progresjonsdataene til databasen for å lagres, deretter nullstilles runtime-progresjonslisten.

Når en bruker oppnår en ny prestasjon så opprettes et nytt gameobject som vi så fester PopupBox scriptet på, dette objektet ødelegges når popup boksen forsvinner ut av skjermen igjen.



3.2 Sekvensdiagram

Sekvensdiagrammet beskriver prosessflyten til prestasjonssystemet.



4. Design prosess

Vi satte først opp rammeverket i et eget isolert prosjekt. Vi la så til muligheten for popups, for å få til dette fulgte vi en guide av Patrick Boelens[2].

Da popupboksene var implementert la vi til metoder som ville støtte databasen. Først laget vi en klasse som kunne holde de nødvendige dataene(navn og progresjon), så lagde vi dummydata vi kunne bruke for å unit-teste de nye metodene uten å involvere databasen.

Når vi var fornøyd med testene fjernet vi dummy dataene og skrev om koden slik at den ville fungere med databasescriptet. Deretter implementerte vi databasescriptet og testet at alt fungerte som det skulle.

5. Referanser

1. Gargolinski, S. *Progress - a free achievement framework for unity*. 15.01.2013]; Available from: http://www.stevegargolinski.com/progress-a-free-achievement-framework-for-unity/.
2. Boelens, P. *Creating an Achievement System*. 23.05.2013]; Available from: http://cgcookie.com/unity/2011/12/16/creating-an-achievement-system/.

Visuellinstruktør

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	21.05.13	DK
2.0			
3.0			

Kontrollert av:	ZA, AO

Userstories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
217	Funksjon: Visuell ingame danse instruksjon	Dagfinn

1. Innledning

For å gi brukerne bedre forståelse av hvordan man skulle utføre dansebevegelsene bestemte vi oss for å lage en ingame instruktør.

2. Implementasjon

Ingame instruktøren er rett og slett bare en liten boks som vises på UI'en som inneholder en tegning som visualiserer utførelsen av en bevegelse. Dette var fort gjort å legge til da vi i stor grad kunne bruke samme fremgangsmåte som med prestasjonssystemet.

2.1 Vurderinger

Vi vurderte lenge om vi skulle ha en "flytende instruktørlinje" som inneholdt bildene for X antall av de neste bevegelsene, men konkluderte med at siden vi var så sent i prosessen ville det bli for omfattende å begynne på dette nå. Vi konkluderte da med at det for vår del ville holde med et lite vindu som viser en og en bevegelse.

2.2 Utfordringer og løsninger

Her beskrives utfordringer vi møter i forbindelse med implementasjonen.

2.2.1 Utfordringer

Vi hadde litt problemer med å finne en god løsning på hvor vi skulle sette hvilken bevegelse som skulle vises til enhver tid.

2.2.2 Løsninger

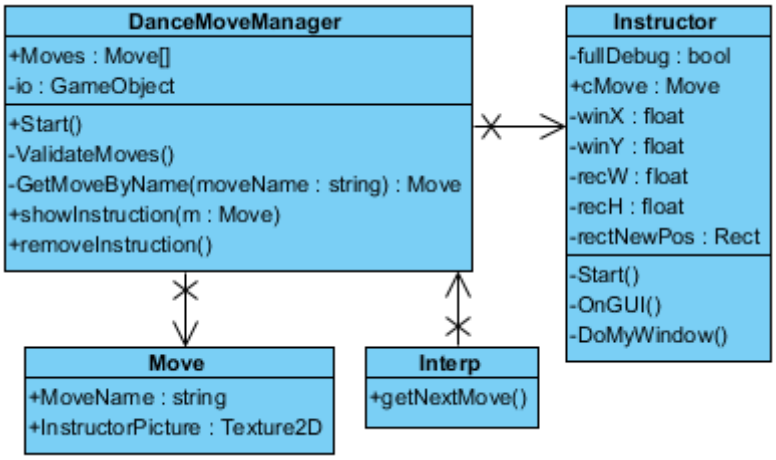
Vi besluttet at interp-klassen skulle bestemme hvilken bevegelse som skal vises.

3. UML

Denne delen av dokumentet inneholder UML diagrammer og tabeller for å beskrive implementasjonsløsningene.

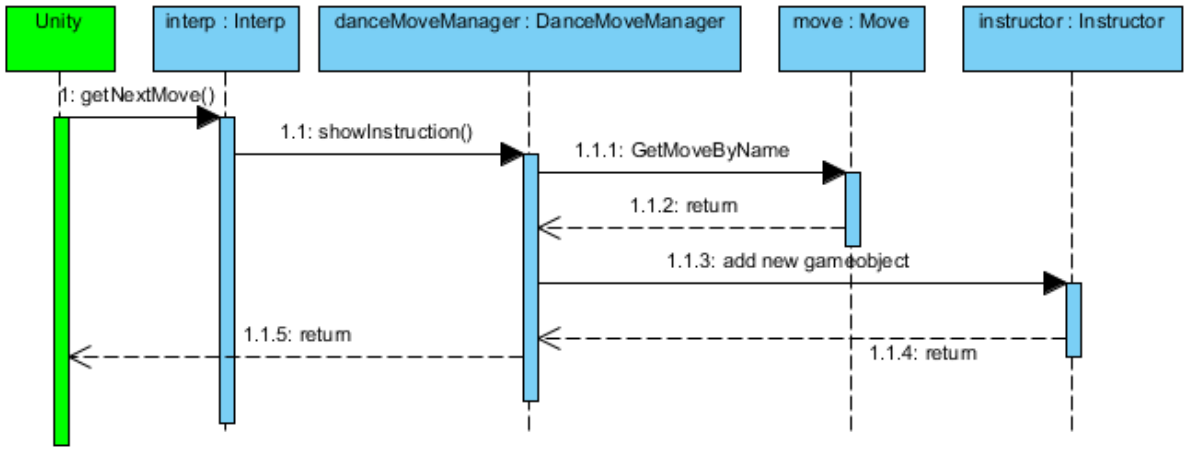
3.1 Klassediagrammer

DanceMoveManager lager et nytt GameObject og plasserer Instructor scriptet på dette. Instructor klassen bestemmer hvordan vinduet skal se ut og hvor på skjermen det skal være. Instructor har også en public Move variabel som definerer hvilken bevegelse som skal vises, DanceMoveManager endrer denne direkte når man kaller på showInstruction og bildet som vises oppdateres fortløpende.



3.2 Sekvensdiagram

Sekvensdiagrammet beskriver prosessflyten til instruktørsystemet.



4. Design prosess

Vi tok utgangspunkt i prestasjonssystemet og kuttet det ned til det som var nødvendig for instruktørsystemet.

Kollisjon deteksjon

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	20.02	Zana Ali
2.0	Oppdatert versjon	17.03	Zana Ali
3.0	Oppdatert	21.05	Zana Ali
4.0	Lagt inn om vanskelighetsgrad og score	21.05	Zana Ali
5.0			
6.0			

Kontrollert av:	MJ, JEH

1. User stories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
106	Arkitektur: Dansescene - Grid	Zana
169	Funksjon: Dans - Grid - Dybde	Zana
189	Funksjon: Dans - Grid - Dybde - Ressurs #2	Are
170	Funksjon: Dans - Deteksjon - Ressurs #1	Zana
171	Funksjon: Dans - Deteksjon - Ressurs #2	Are
110	Funksjon: Dansescene - Instruktør	Are
193	Funksjon: Dansescene - Instruktør - Ressurs #2	Zana

2. Innledning

User story 106 er implementasjon av et gridsystem. Dette gridsystemet består av et 6x8x3 cellesystem , eller gameobjects som er sfærer(kuler som er strekt ut) med identiske størrelser. Størrelsen til griden kan justeres i "inspector" menyen i Unity, der xRad, yRad og zRad er variablene. Griden sin oppgave er å gjøre det mulig å oppdage bevegelser som spilleren gjør, ved å detektere kollisjon som oppstår mellom avataren og cellene . Selve kollisjon deteksjonen er det et eget script kalt DanceDetector.cs som tar seg av. Kun de cellene i griden som er i en dansebevegelse skal være synlig for brukeren. Når en celle har blitt kollidert med vil den usynliggjøres igjen.

Griden gjør det mulig å lage de bevegelsene man måtte ønske i et 3D plan, slik at bevegelser i dybden også er mulig. Gridskriptet er nært knyttet DanceDetector og FollowBody skriptene.

3. Implementasjon

3.1 Grid.cs

Gridscriptet oppretter et objekt kalt cell, som blir satt til å være en kube i dette tilfellet. Deretter klones cell ved å settes inn i et array antall ganger som det er celler(xRad*yRad*zRad) ved en for-løkke. Her får også objektene(cellene) også navn som korresponderer med deres ID. Alt dette skjer i en metode kalt initializeCells som kjører en gang i start(). Metoden som også blir kalt i Start() er PositionCells(). Denne sørger for å posisjonere spillobjektene i spillverden rundt avataren. Det er en nested for-løkke som iterer gjennom xRad, yRad og zRad bruker xPos, yPos og zPos som er variabler som kan endres i inspektør-vinduet. Dette er mulig siden disse variablene er satt til public.

3.2 FollowBody.cs

FollowBody fester objekter til forskjellige deler av avataren(hender, bein osv). Disse objektene er usynlige og har fått ulike navn som gjør det mulig å sjekke etter at det er riktig kroppsdel som treffer en aktiv celle. Disse objektene har DanceDetectorskriptet knyttet til seg, for å kunne detektere kollisjon mellom disse kubene og cellene i griden.

3.3 DanceDetector.cs

DanceDector detekterer kollisjon, henter dansebevegelser og tegner de i spillverden. OnTriggerEnter() er metoden som blir kalt hver gang objektet med dette skriptet knyttet til seg, kolliderer med et annet objekt som er trigger, i dette tilfelle cellene. I dette skriptet blir aktive kroppsdeler koblet opp mot aktive celler slik at det er mulig å detektere at riktig bevegelse blir utført. Det er mulig å detektere fire forskjellige kroppsdeler samtidig opp mot fire aktive celler.

3.4 Vurderinger

3.4.1 Antall celler

Det var ønskelig å ha en stor grid med mange celler(eksempelvis 60x80*30). Tatt i betraktning om det å ha så mange objekter inne i spillverdenen vil påvirke ytelsen har vi valgt å lage en mindre grid enn så lenge. Slik løsningen har blitt, fungerer også griden som en instruktør. Fordelen med en større grid vil gjøre det mer detaljert og lettere å skille mellom om spilleren gjør en utmerket eller en perfekt bevegelse eksempelvis. Ulempen er at dersom antallet celler øker, blir størrelsen til cellene også satt til å være mindre. Noe som vil gi en mindre instruktør for brukeren og vanskeligere å se og treffe.

3.4.2 GUIGrid vs spillobjekter

En annen løsning som har blitt vurdert er å lage en grid som er posisjonert på selve skjermen, og ikke i en spillverden, en GUIGrid. Siden det har vært lite informasjon om dette har valget falt på å posisjonere griden i spillverdenen. Fordelen med en grid i en spillverden er at vi vet hvordan deteksjon av forskjellige kroppsdeler kan skilles fra hverandre ved hjelp av tags som kan legges til og vi er kjent med kollisjon deteksjons metoder som Unity tilbyr.

3.4.3 Aktivering og deaktivering av celler

Hver celle har en unik ID for å kunne skille mellom cellene. Dette er nødvendig dersom man vil vite hvilke celler som skal være aktive, og når de skal aktiveres eller deaktiveres. Den første aktive cellen får en klar og tydelig grønnfarge slik at spilleren lett kan se hva det er som skal treffes, er det flere celler i en dansebevegelse blir de neste cellene også tegnet grønne, men med høy gjennomsiktighet.

En dansebevegelse kan se slik ut:
a[][] = {{a, b},
{1,2,3},
{4,5,6}
};

Dette er et 2D array med kroppsdel og celler. Her ser vi at det er to kroppsdeler involvert i dansebevegelsen, noe a og b representerer. De to neste arrayene 1,2,3 og 4,5,6 er cellene for de tilhørende aktive kroppsdelene. Celle 1 og celle 4 vil få en klar grønnfarge mens cellene 2,3 og 5,6 vil få grønnfarge med gjennomsiktighet. Dersom celle 1 som er nåværende aktive celle blir truffet av kroppsdel a, vil neste element i det arrayet være nåværende aktive celle og få en klar grønnfarge.

Når en dansebevegelse er utført hentes ny bevegelse, aktiv celle og kroppsdel endres. Merk at en dansebevegelse kan bestå av flere kroppsdeler og aktive celler, en dansebevegelse blir derfor ikke ansett som utført før alle cellene for kroppsdelene har blitt truffet.

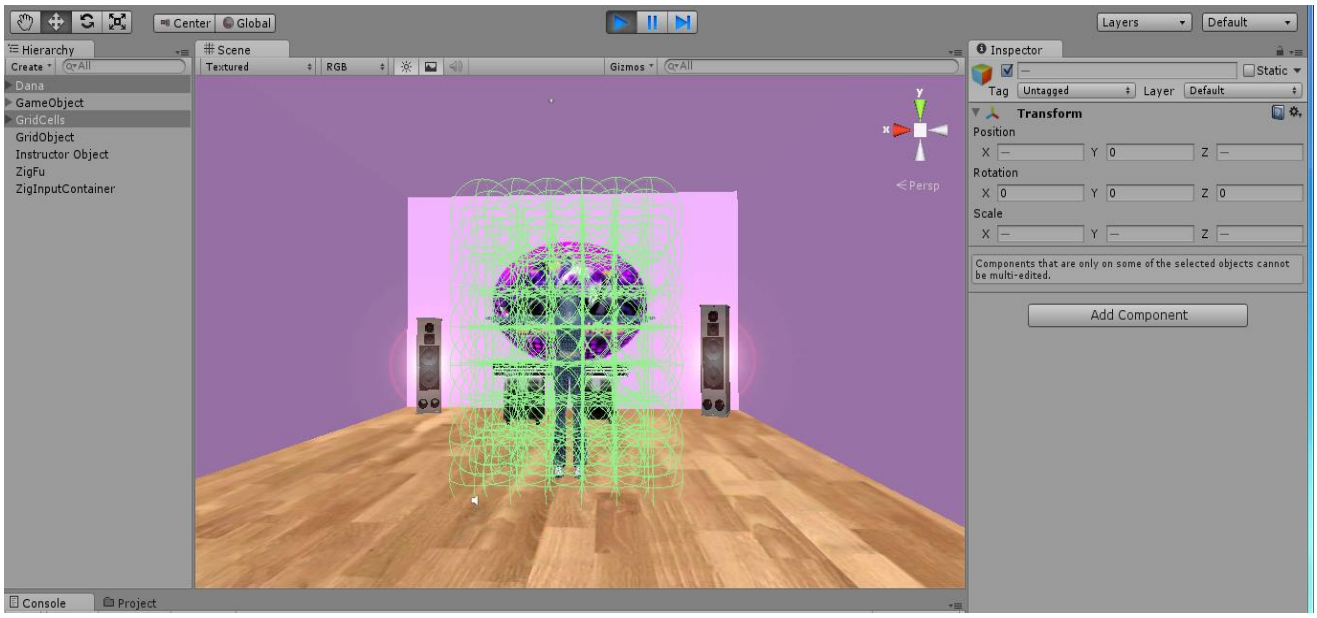
3.4.4 Skille mellom kroppsdeler

I Unity er det mulig å bruke tags og navnet til avataren som parameter i triggers. I metoden OnTriggerEnter() som blir kalt når en collider(objekt) entrer triggeren er det mulig å spesifisere hva betingelsene er for å trigge en ønsket handling. Her kommer tags og inn i bildet, slik at det er mulig å skille mellom når en arm treffer en celle fra et bein f.eks.

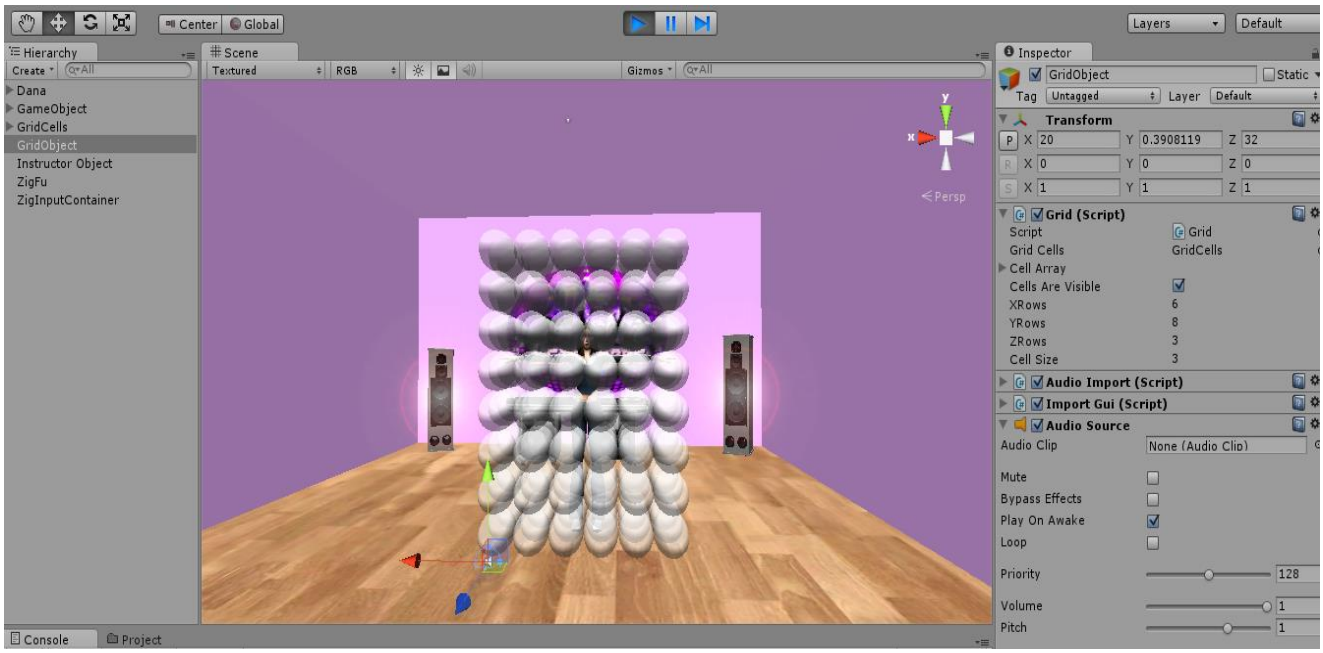
Cellene i griden er triggerne våre mens alt som kolliderer med den er colliders. Man kan bruke en kroppsdel som har en unik tag som gir utslag for at noe kan skje. Dette er en eventdrevet metode som kalles hver gang et objekt er i kontakt med et annet objekt. Variable blir brukt til å holde aktive celler og kroppsdeler som endrer seg i henhold til spillets flyt. Om en bevegelse er fullført, vil en ny bevegelse hentes og nye celler og kroppsdeler aktiveres.

3.4.5 Modifikasjon av synliggjøring

Scriptet er lagt opp slik at cellene kan synliggjøres om ønskelig. Dette er enkelt å gjøre i Unity, man setter renderer til objektet til å være false dersom man ønsker å gjøre objektet "usynlig". Grunnen til dette er først og fremst for testing formål. I selve applikasjonen vil ikke cellene, bortsett fra de aktive, være synlige for brukeren av åpenbare årsaker.



Figur 1: Cellene er ikke synlige, men siden de er merket i prosjektet, synes de.



Figur 2: Cellene er synlige og objektene vises med de egenskapene de har.

3.5 Utfordringer og løsninger

3.5.1 Mesh renderer

Med mesh renderer vil man kunne bruke hele modellen, eller avataren i kollisjon deteksjon. I motsetning til box collider, vil en mesh collider forme seg etter modellen og ikke andre uønskede områder. Utfordringen med mesh renderer har vært å oppdatere mesh rendereren etter avatarens posisjon. Mesh rendereren viste seg som ved initialiseringen slik modellen er når den står stille, men oppdaterer seg altså ikke etter modellen dersom den beveger seg. Mesh renderer viste seg også som krevende ved kjøring i Unity da det påvirker ytelsen. Se testcase for user-story ID 147 for testen som ble gjort av dette.

3.5.2 Løsninger

Vi har valgt å løse dette ved å lage et skript (FollowBody.cs) som fester kuber til modellen. Slik at det kun sjekkes etter kollisjon på de delene vi selv ønsker. Disse kubene har box colliders knyttet til seg, noe som gir mindre overhead sammenlignet med mesh collider. Dette funket i motsetning til mesh renderer(les: testcase for user-story ID 146).

3.5.3 Passing parameters

I C# kan parametere bli sendt enten ved value eller reference. Default er det by value, noe som ga oss en utfordring siden vi ønsket å ha by reference. Vi opplevde at variablene ikke endret seg slik vi ønsket, så vi opprettet en tråd på Unity-forumet. Link:
http://answers.unity3d.com/questions/452303/ontriggerenter-doesnt-work-as-it-should-c.html

Vi fikk gode inputs og brainstormet litt rundt det selv.

3.5.4 Løsninger

Tilbakemeldingene vi fikk testet vi for å lokalisere hva det var som var galt. Når vi oppdaget at parameter passing blir gjort by value, fant vi ut at det å bruke statiske variable kunne være en løsning på dette, siden vi ønsket globale variable. Derfor er de fleste variablene i DanceDetector.cs statiske.

4. UML

Her vises UML diagrammer over hvordan systemet er implementert og fungerer for kollisjon deteksjon.

4.1 Klassediagram

Dette klassediagrammet viser oversikten over klassene som er involvert i kollisjon deteksjon. DanceDetector detekterer når noe er i kontakt med cellene og trigger ønskede handlinger. Nye bevegelser hentes, slik at griden aktiverer de områdene som er spesifisert. For at skriptene skal kunne kjøre, må de være knyttet til Unity på en eller annen måte. Grid.cs er knyttet til et empty gameobject i Unity kalt grid, og kjørernår Unity starter danserom scenen. FollowBody er også knyttet til et vilkårlig objekt i Unity. De andre scriptene er relaterte til Grid ved å lage instans av klassen og ved hjelp av metodekall manipulere metodene og objektene i klassen.

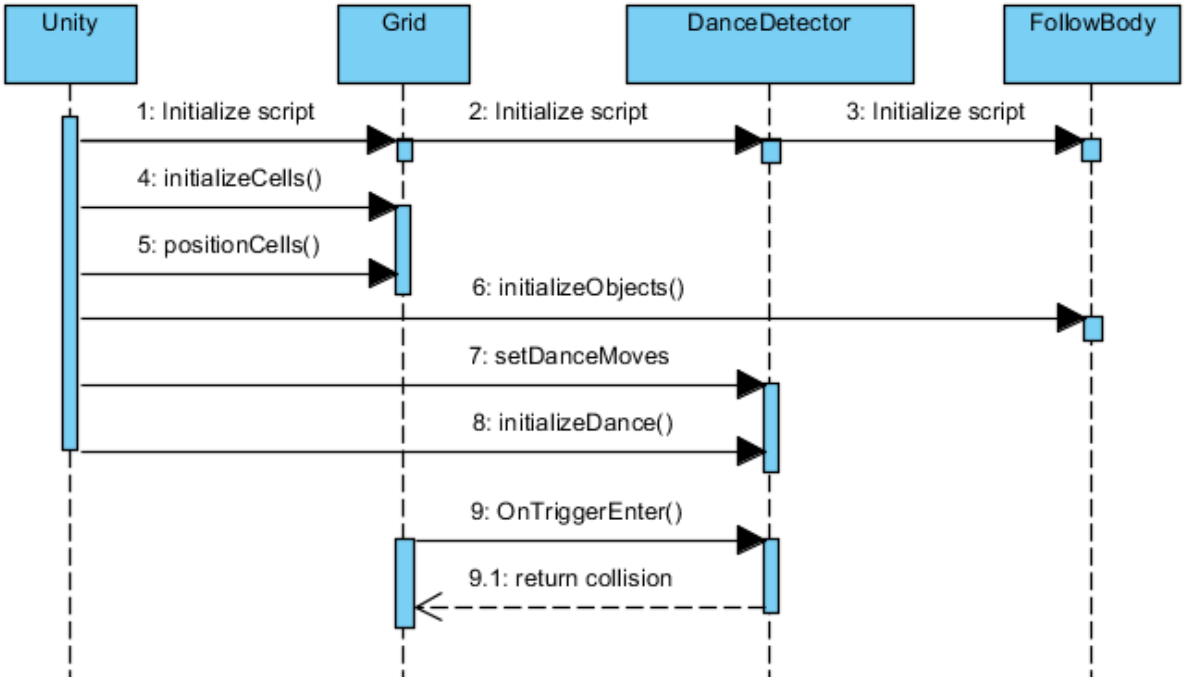
Figur 3: Klassediagram som viser klassene involvert og relasjonene mellom dem

4.2 Sekvensdiagram over involverte klasser

Griden vil samhandle med flere klasser, sekvensdiagrammet viser metodekallene disse gjør seg i mellom og tidsrekkefølgen det skjer.

Det første som skjer er at Unity initialiserer skriptene knyttet til objekter i scenen.InitializeCells() blir kalt fra Grid.cs, deretter positionCells(). Det som skjer er at cellene blir skapt og plassert ute i spillverdenen. Dansebevegelser skal deretter bli generert, FollowBody.cs fester kubene til avataren.

DanceDetector.cs skal bestemme hva som det skal sjekkes kollisjon etter, altså dansebevegelsen med aktive celler og tilhørende aktive kroppsdeler. Betingelsen blir satt i OnTriggerEnter(). Kun betingenelsene som stemmer overens med det som er spesifisert trigger en ønsket handling, selv om kollisjon skjer mellom avataren og griden hele tiden.



Figur 4: Sekvensdiagram

Vanskelighetsgrad og scoringsystem

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	22.05	Zana Ali
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	MJ, AO

1. Userstories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
209	Funksjon: Scoringsystem	Zana
210	Funksjon: Scoringsystem - Ressurs #2	Are
197	Funksjon: Vanskelighetsgrad - Lengre tid mellom bevegelser	Zana
107	Funksjon: Dynamisk vanskelighetsgrad	Zana og Are

2. Innledning

Måten vi har løst vanskelighetsgrad på, er å justere på tiden spilleren har på å utføre en bevegelse. I LetsDance kan du velge mellom vanskelig, normal og lett. Hvis du velger lett, vil du få mer tid på å utføre en bevegelse enn om du ville valgt normal eller vanskelig. Det er en stoppeklokke som teller ned hver gang en bevegelse er generert. Når stoppeklokka når null, eller dansebevegelsen er utført vil den da resettes slik at spilleren får den samme tiden til å utføre neste bevegelse.

Det er også implementert en dynamisk vanskelighetsgrad som justerer vanskelighetsgraden i runtime etter hvordan brukeren gjør det.

Scoringsystemet er basert på hvor godt du utfører en bevegelse i forhold til beats i sangen som blir spilt. Hvis en bevegelse blir fullført på en beat, blir spilleren belønnet med ekstra poeng.

3. Implementasjon

3.1 CheckTime.cs

CheckTime sjekker hvor lang tid det har gått fra en bevegelse har blitt generert ved hjelp av en timer som teller ned. Når tiden er ute kaller den på DanceDetector som henter ny bevegelse og timeren blir resatt.

3.2 GiveScore.cs

GiveScore vurderer hvor bra en dansebevegelse er utført. Variabelen msRange bestemmer hvor lang tid til og fra en beat i det en dansebevegelse er utført for at det skal anses som truffet på beat. Denne variabelen er større for en lett vanskelighetsgrad, enn en vanskelig en. Slik at marginene spilleren har på å treffe en beat varierer ut ifra vanskelighetsgraden.

Høytalerne i danserommet blir større på en beat for å vise spilleren når det er en beat i sangen. Denne visuelle effekten kan få det til å se ut som en animasjon. Det skjer i dette skriptet fordi det spesifiseres her når stoppeklokka passerer en beat.

3.3 DanceDetector.cs

Dette scriptet sender beskjed til GiveScore og CheckTime hver gang en bevegelse er utført. Vanskelighetsgrad som brukeren har valgt blir satt her, sammen med tiden de får på å utføre en bevelse.

Metoden checkPerformance blir kalt hver gang en bevegelse er over. For hver tiende bevegelse sjekkes det etter hvor godt brukeren har gjort det, og en beslutning blir tatt på om vanskelighetsgraden skal endres eller ikke

3.4 Utfordringer

3.4.1 Sette vanskelighetsgrad

Det å bestemme hva som er lett, vanskelig eller middels er ingen lett oppgave. Det er ingen fasit på hva som definerer noe som er lett. Ofte blir det en subjektiv vurdering basert på egen erfaring. Det er heller ikke enkelt å finne riktig balansegang mellom de tre gradene av vanskelighetsgrad.

3.5 Løsninger

Måten vi har valgt å løse vanskelighetsgrad på er å justere på tiden en bruker har på å utføre en bevegelse. Vi føler det er en god måte å gi brukeren mulighet til å fullføre en bevegelse som kan være en prestasjon i seg selv for målgruppen til LetsDance. Det å ha muligheten til å endre vanskelighetsgrad selv gir også brukeren bedre mulighet til å tilpasse vanskelighetsgraden til sitt eget nivå. Noe annet som vanskelighetsgraden også bestemmer er antallet millisekunder fra en bevegelse er utført og en beat i musikken blir spilt. Variasjonen i millisekunder er mindre dersom vanskelighetsgraden er høy.

Etter flere tester satte vi verdier vi tror er gunstige for målgruppen til LetsDance.

I tillegg har vi implementert en dynamisk vanskelighetsgrad. Den dynamiske vanskelighetsgraden justeres hvis spillet blir for lett eller for vanskelig for brukeren. For hver tiende fullførte dansebevegelse sjekkes det etter hvordan brukeren gjør det. Dersom ikke minst 6 av 10 bevegelser er fullførte justeres vanskelighetsgraden ned. Dersom minst 9 av 10 bevegelser er fullførte vil vanskelighetsgraden justeres opp.

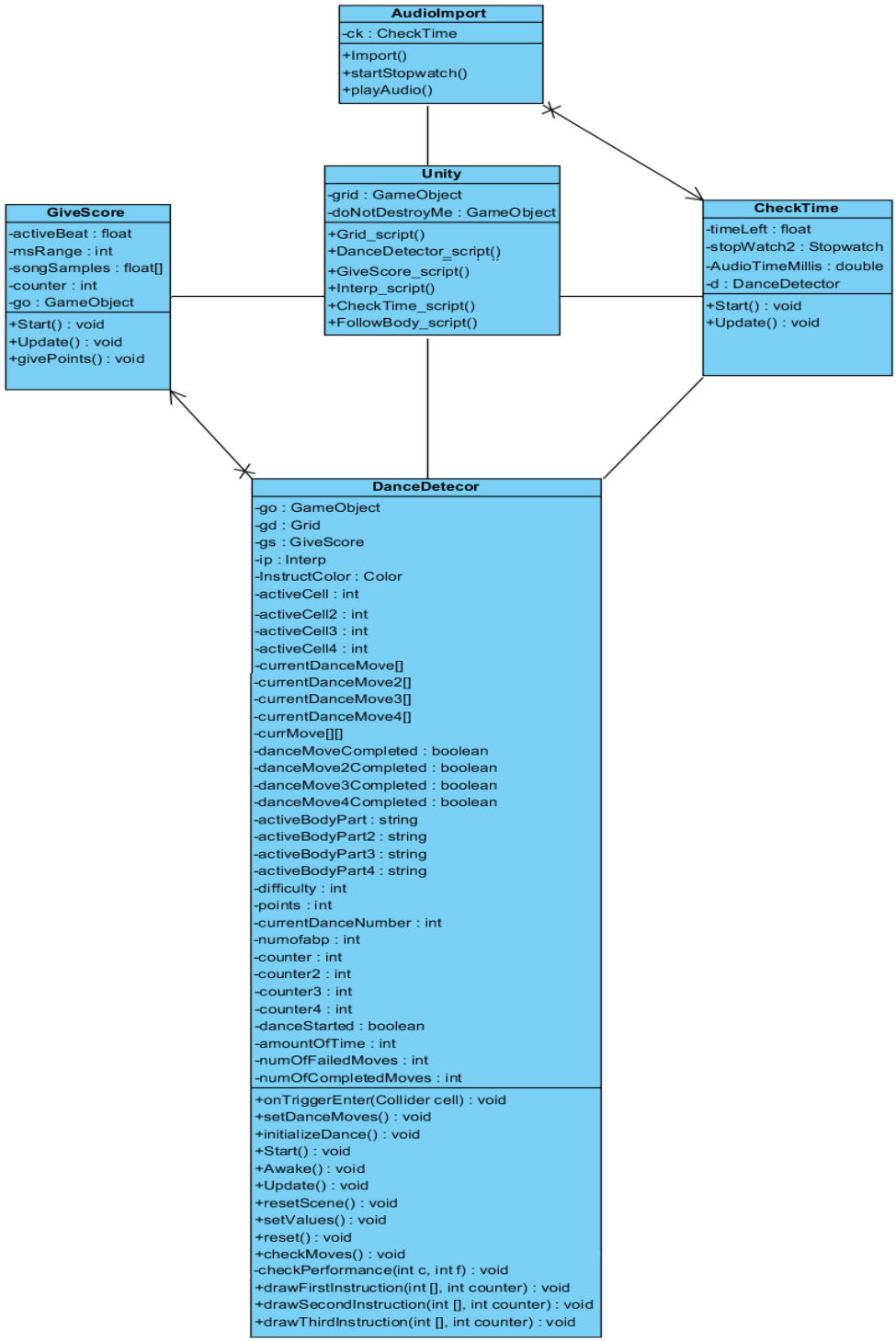
3.5.1 Poeng

Når det gjelder poeng gjelder de samme type utfordringene som med vanskelighetsgrad. Det er en nær tilknytning mellom poeng og vanskelighetsgrad. Desto vanskeligere en oppgave er, desto høyere skal også belønningen være. I dansespill vil man gjerne danse i takt med musikken, og det er det vi oppfordrer spilleren til å gjøre. Dersom en bevegelse blir utført på en beat blir spilleren ekstra

belønnet. Dersom en bevegelse blir utført blir også spilleren belønnet, men kun med halvparten så mye poeng som det å treffe på en beat.

4. UML

4.1 Klassediagram

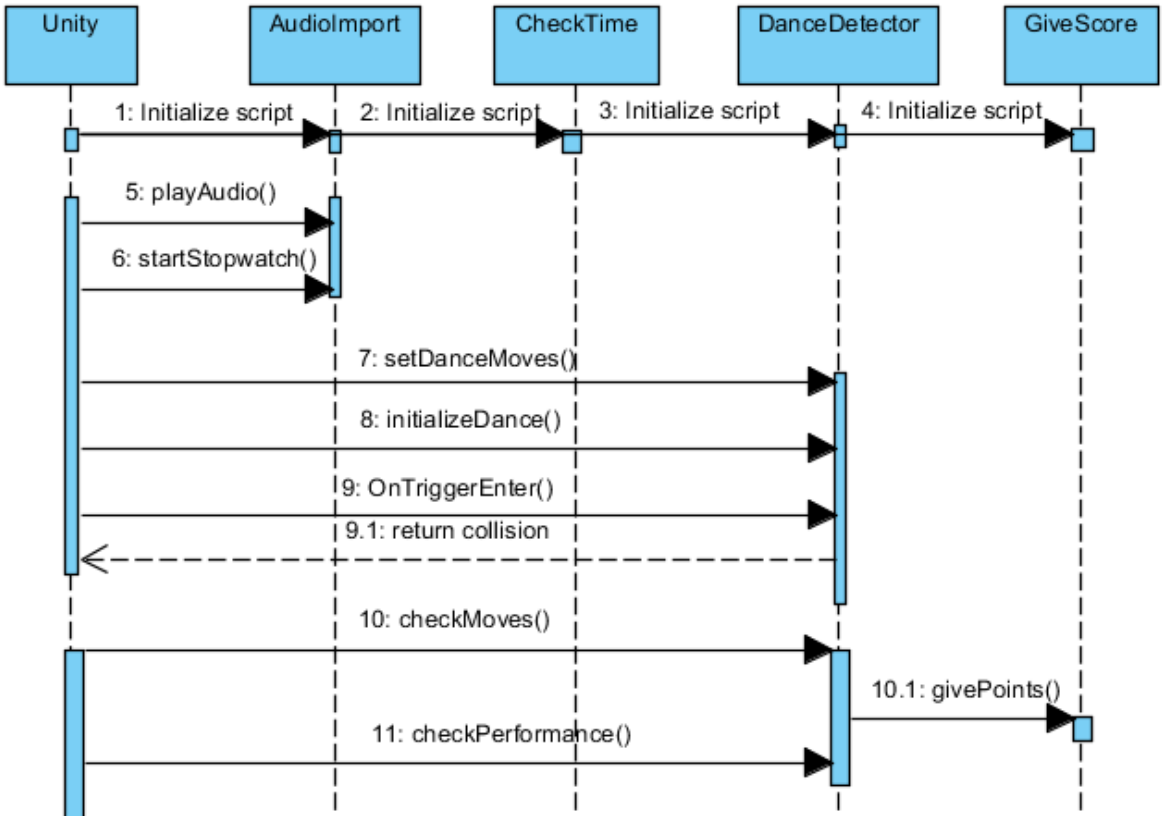


Figur 3: Klassediagram som viser klassene involvert og relasjonene mellom de

4.2 Sekvensdiagram over involverte klasser

AudioImport skriptet starter stoppeklokka i CheckTime. Musikk blir spilt, og dans generert. Deretter sjekkes det etter kollisjon deteksjon i DanceDetector. checkMoves blir så kalt for å sjekke om dansebevegelsen er blitt fullført. Dersom den er fullført blir givePoints i GiveScore kalt. givePoints sjekker vanskelighetsgraden og om en bevegelse blir utført på en beat. Poeng blir deretter gitt basert på disse to parameterne.

Etter at en bevegelse er utført, blir checkPerformance kalt for å se hvordan brukeren gjør det. Basert på vil vanskelighetgraden bli justert opp, ned eller forbli den samme.



Figur 4: Sekvensdiagram

Audio Import

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	05.02.13	HO
2.0	Redigert	19.03.13	HO
3.0	Redigert	21.05.13	HO
4.0			
5.0			
6.0			

Kontrollert av:	ZA, AO

1. User stories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
138	Musikk - AudioVisualizer	Henrik Olsson
92	AudioImport	Henrik Olsson
192	Funksjon: Musikkanalyse, ressurs 2	Dagfinn Kjærnet

2. Innledning

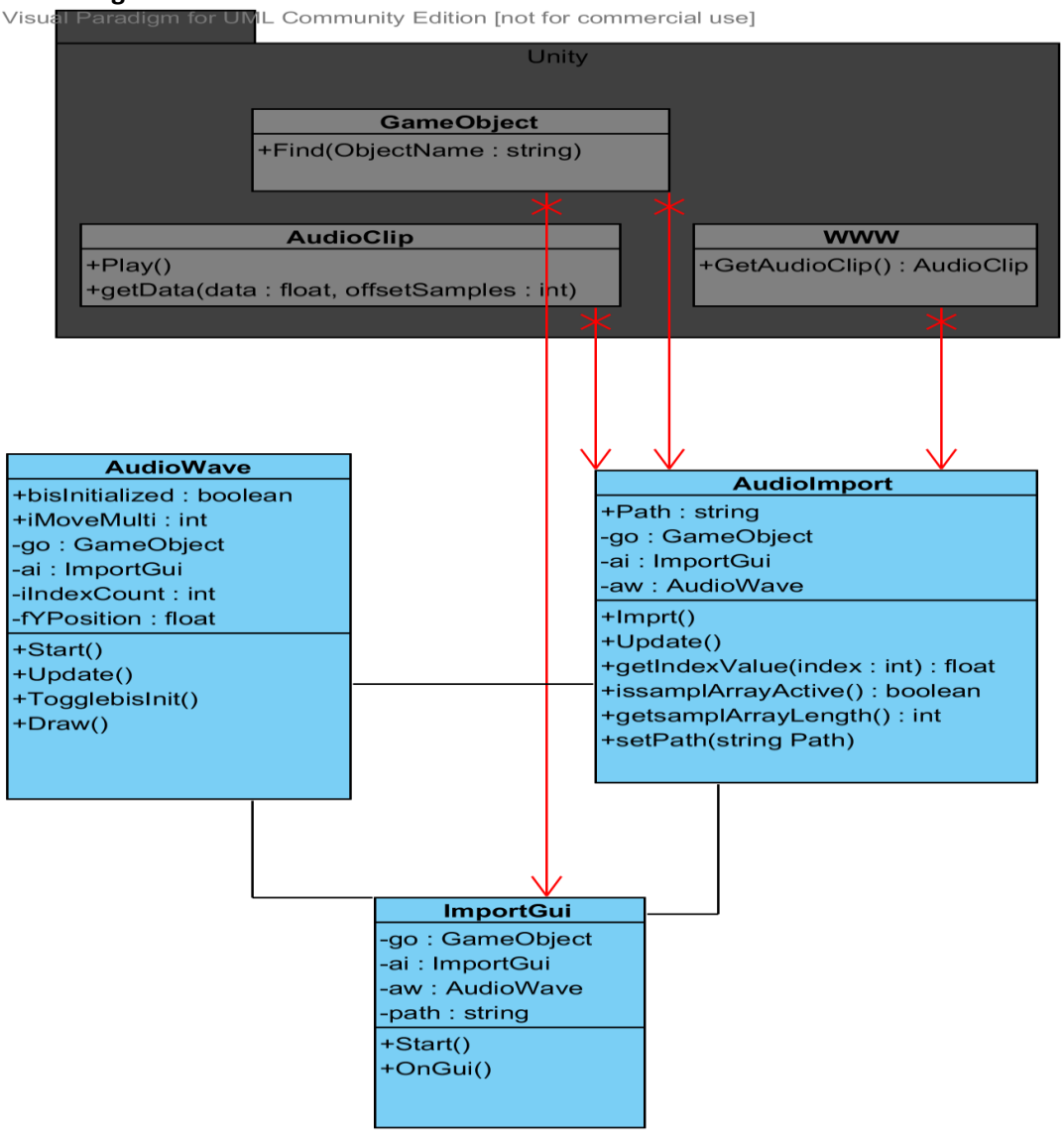
Et av de definerte kravene til produktet er at brukeren skal ha mulighet til å kunne legge til egendefinert musikk. Fordelen med å kunne ha egendefinert musikk i spillet er at man vil få stor variasjon og et mer personlig forhold. Samtidig vet vi at bevegelse i takt er et viktig prinsipp når det kommer til å trene motorikk derfor ønsker vi at bevegelser skal bli utført i takt med musikken. Videre blir designprossesen og implementasjonen forklart ved hjelp av klassediagrammer og sekvensdiagrammer.

3. Designprossess - AudioVisualizer

AudioVisualizer er et lite subsystem av produktet som foreløpig har som oppgave å visualisere lyd til utvikling av produktet. Slik jeg forstår det vil det bli lettere å forstå hvor en "beat" er i en lydfil hvis jeg lager en visuell representasjon av hver eneste sample i filen. Dette er ikke noe som vil være en funksjon i Lets Dance, men kun et verktøy for analyse og utvikling.

Følgende vises det til to diagrammer som går innen for UML standaren, siden vi ikke bygger applikasjonen fra scratch men ut fra Unity motoren vil vi måtte forholde oss til de funksjonene unity har innebygd. Unity har et stort bibliotek som vi finner dokumentert på utviklerens hjemme side. http*://docs.unity3d.com/Documentation/ScriptReference/*
I denne lille delen av applikasjonen vil det kun bli utnyttet noen av funksjonene i Unity derfor er kun disse klassene representert i diagrammene.

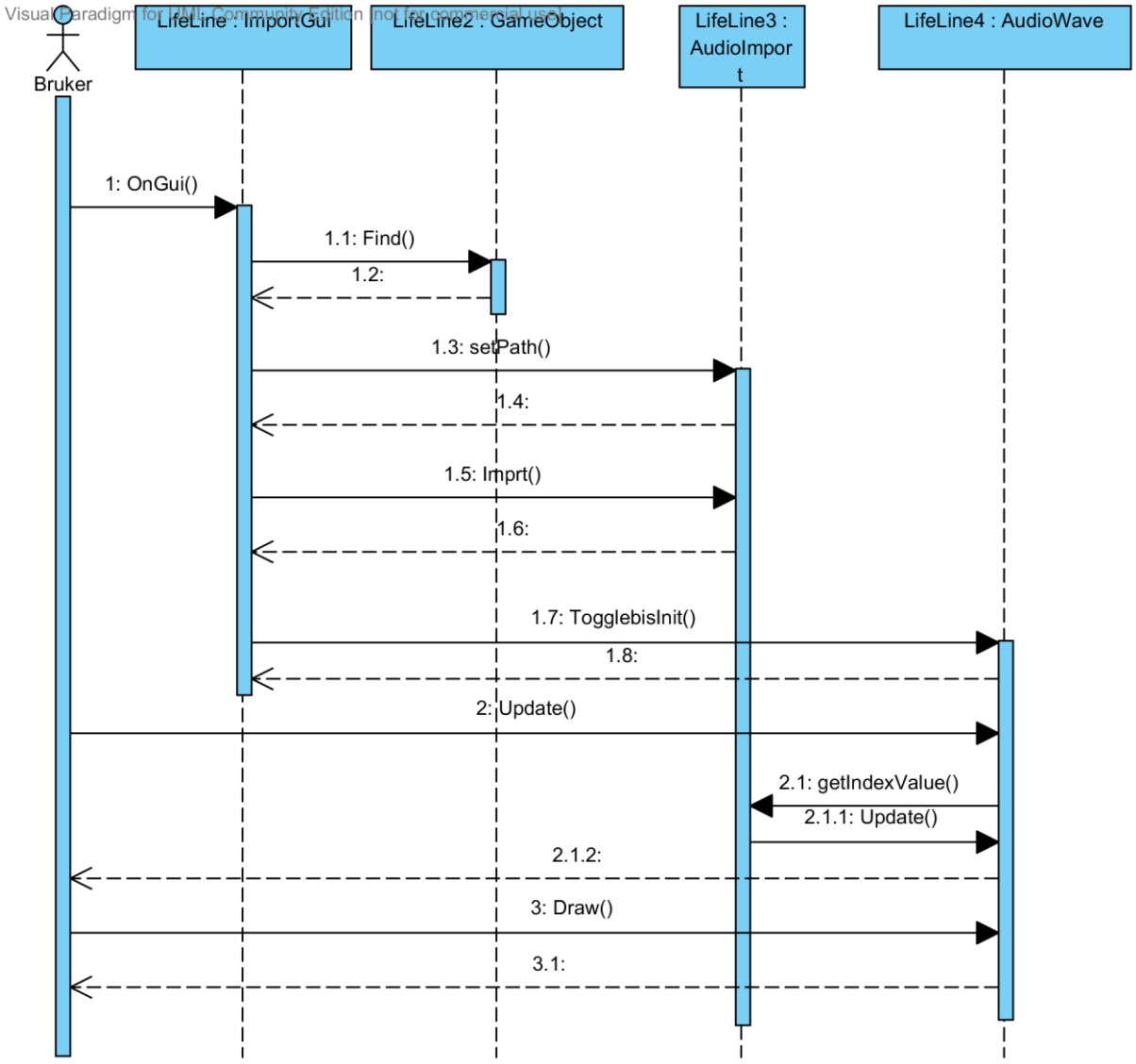
Klassediagram:



Klasser og funksjoner som er bakt inn i Unity har ikke vi muligheten til å gjøre endringer på, bare utnytte ved å trykke inn data i funksjoner og hente ut annen data.
I dette systemet har jeg satt opp tre Klasser; AudioImport, AudioWave og ImportGui. Disse har som roller å lage et enkelt brukergrensesnitt hvor brukeren kan taste inn banen til en lydfil av typen .ogg format. Deretter trykke OK, etter det blir det kalt på en Imprt() funksjonen i AudioImport. Imprt funksjonen er av typen IEnumerator kan man "pause" ved hjelp av yield kommandoen. I dette tilfellet vil vi være avhengig av å vente på at importeringen av lydfilen skal bli gjennomført før den brukes til noe. Klassen AudioImport vil også ta seg av å gjøre en numerisk fremstilling av lydfilen ved å konvertere det om til et flyttalls array. Dette blir så hentet inn i klassen AudioWave, denne klassen har som oppgave å gjøre noe visuelt. foreløpig er dette festet til en kube som vil bevege seg etter langs en vertikal linje etter verdiene som blir hentet ut fra flyttals arrayet.

Det skal også nevnes at det finnes funksjoner i klassene som ikke er representert i diagrammene, dette er funksjoner som ikke spiller noen viktige roller.

I denne delen av systemet har vi utnyttet klassene GameObject, WWW og AudioClip som unity tilbyr. GameObject er et objekt som ofte vil gjenta seg i utviklingen av applikasjoner i Unity da man må benytte seg av dette for å kunne refere til for eksempel en spesifik instans av en klasse. WWW objektet vil bli brukt for å kunne hente en fil fra mappestrukturen i Runtime. AudioClip objektet er rett og slett bare for å kunne trykke spill av for å spille lydfilen i runtime.

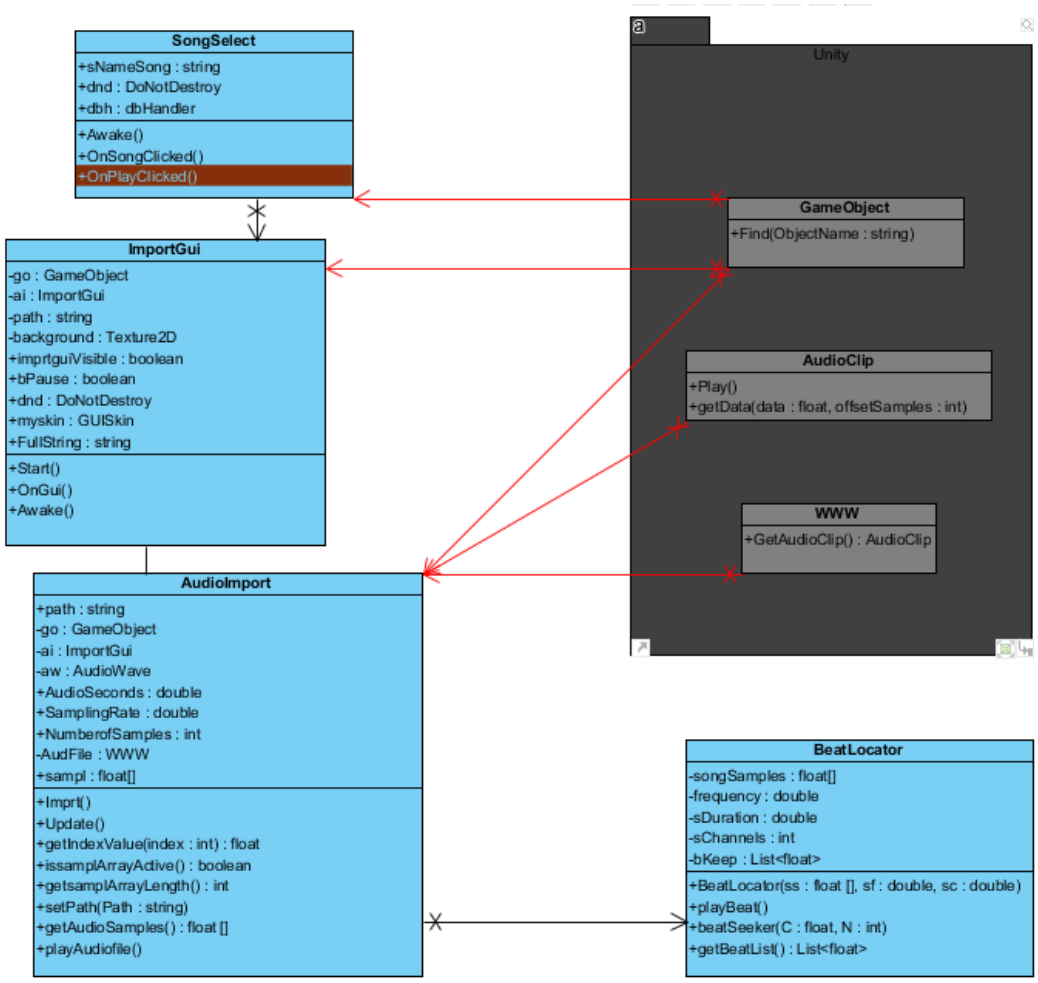


I denne figuren er det vist hvordan en bruker av applikasjonen får de forskjellige klassene til "å prate sammen" ved hjelp av funksjoner og returmelding.

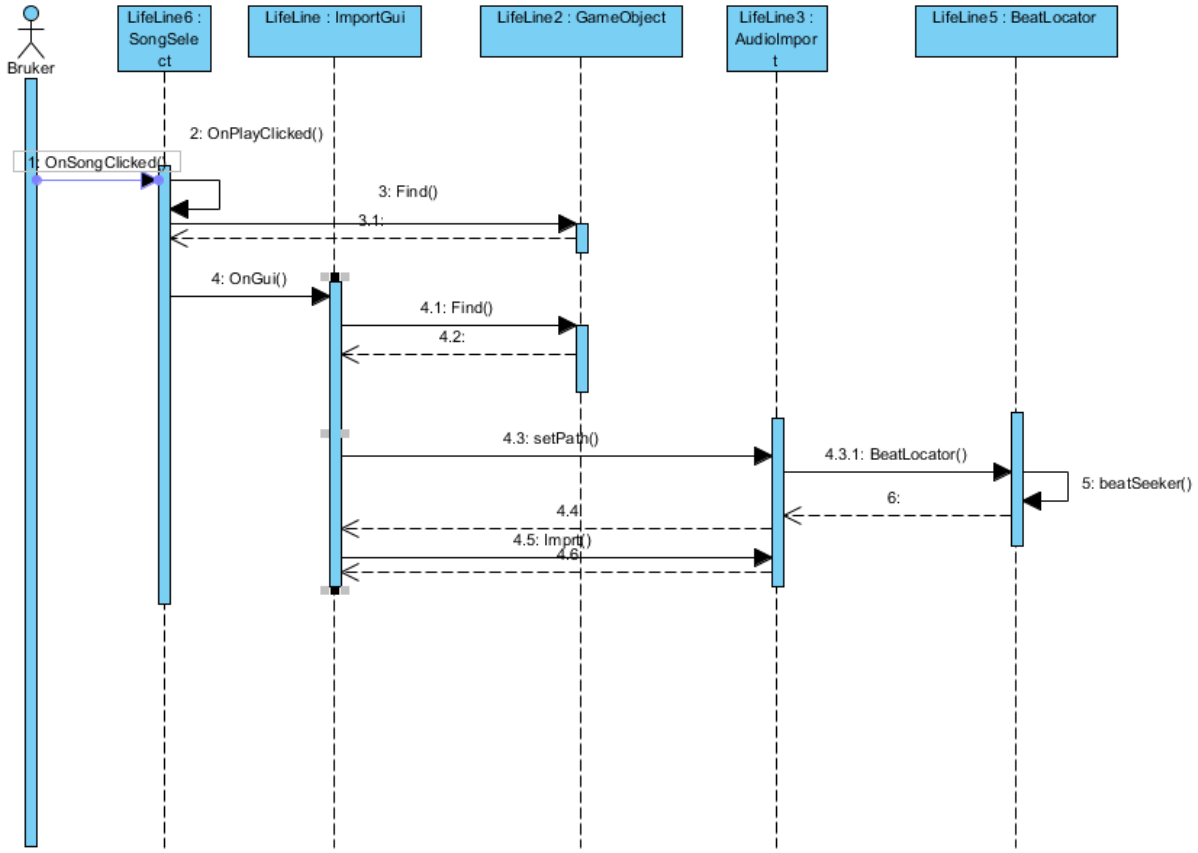
Som nevnt ble AudioVisualizer bare brukt under utviklingsprossesen med de fungerende klassene; ImportGUI, AudioImport og AudioWave, dette ga en visuel presentasjon av lydsignalet imporert. AudioWave er den klassen som faller bort fra produktet. Den klassen erstates av BeatDetector som er skrevet av Dagfinn.

4. Implementasjon

For å kunne utnytte dataene funnet i sample arrayet, er vi avhenig av å prossesere dette. Dette er en stor mengde data, standard samplingsrate er 44100 hz. Vi konverterer dette til flyttals array for å ha en numerisk representasjon av dataene. Denne prosseseringen blir gjort av AudioImport klassen, videre ønsker vi en egen klasse for å kunne hente takten i låten. Så vi velger å lage en klasse BeatLocator som tar hvor constructeren krever tre parametre; sample array, frekvens og antall kanaler i lydfilen.
AudioImport vil fortsatt bli kalt av ImportGUI, men riktignok tar Import klassen lenger for seg å presentasjonen av brukergrensesnittet for valg av fil i mappestrukturen. Nå har det rollen som et usynelig grensesnitt mellom SongSelect og AudioImport for import av lydfil. ImportGUI har også fått rollen som in-game meny, altså da to knapper som gir brukeren muligheten til å pause og å avbryte spillet.



Som vi ser i denne figuren er at en klasse SongSelect referer til ImportGUI, SongSelect har pathen til hver låt lagret i minnet. Denne pathen blir avgjort etter hvilke filer brukeren har "lagt til" i spillet tidligere. Filene blir lastet inn under runtime, egentlig er det bare pathen/banen som blir lagret når brukeren finner fram filen i mappestrukturen.



I denne figuren ser vi hvordan sekvensen blir etter brukeren har valgt lydfil og startet et spill. Spilleren starter et spill, så trykker "GO!" for å sette i gang dansen og systemklokken. GameObject er et objekt i Unity som lager et interface mellom allerede instanserte klasser som arver fra monobehaviour, dette er noe vi må benytte oss av for å få kontakt med script som er knyttet til andre fysiske objekter i rommet. ImportGui finner videre AudioImport objektet for å sette den bestemte banen til lydfilen. Videre vil Imprt funksjonen i AudioImport hente inn lydfilen for å sende de angitte parametrene i det nye objektet av BeatLocator. Beatlocator kaller på den interne funksjonen beatSeeker som gjør diverse operasjoner for å hente ut alle "beatene" i lydfilen, dette er dokumentert i kildekoden. Disse beatene er blir lagret i en datastrukturen List. BeatLocator har også en public funksjon playBeat, denne vil bli kalt på fra utsiden av dette subsystemet, dette er for å få beaten synkron med musikken og DanceDetector. Til slutt er det en funksjon som returnerer hele listen med beats.

Interp

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	20.05.13	AO
2.0	Oppdatert	22.05.13	AO
3.0			

Kontrollert av:	ZA, JEH

1. User stories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
206	Funksjon: Dansegenerator - Opprette bevegelsene	Are Oven

2. Innledning

Interp-klassen tar seg av tolkningen av dansebevegelsene, det vil si at den får inn data og tolker den og returnerer verdier som brukes videre til å aktivere celler.

3. Implementasjon

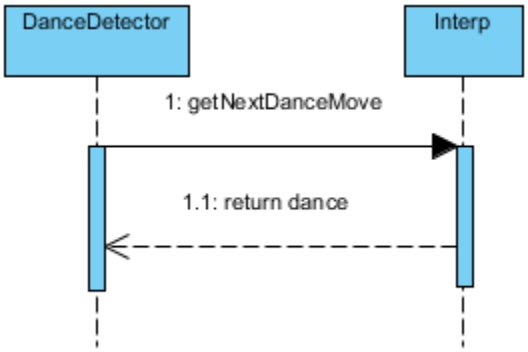
Interp-klassen er den som tolker alle bevegelsene som er i en dans. Den henter den valgte dansen(en string) fra DoNotDestroy objektet og venter på å motta en int-verdi som sier ifra hvor i stringen man er slik at den kan returnere den korrekte bevegelsen.

Vi har valgt å lagre en dans som en string som inneholder mange bokstaver, for seg selv er det bare en string uten noe videre betydning. Det interp-klassen gjør er å ta en bokstav og konvertere den til et 2D string-array. Dette arrayet inneholder opp til fem string-array. Det første av disse inneholder bokstaver som representerer kroppsdeler, for eksempel ”a” representerer venstre hånd. Arrayene som kommer etter det første inneholder tallverdier. Disse verdiene representerer id-en til cellene som brukes i griden til å lage en bevegelse. Det første arrayet etter det som inneholder bokstavene vil bli koblet mot den første bokstaven i det første arrayet, det andre vil bli koblet mot den andre bokstaven og så videre. Det vil alltid være slik at den bokstaven som kommer først i alfabetet vil stå først i arrayet. Det vil si at "a" vil alltid stå først, hvis det ikke eksisterer en "a" så vil "b" stå først og så videre. Dette gjør at vi kan ha kontroll på hvilke celler som skal kobles til hvilke kroppsdeler.

4. UML

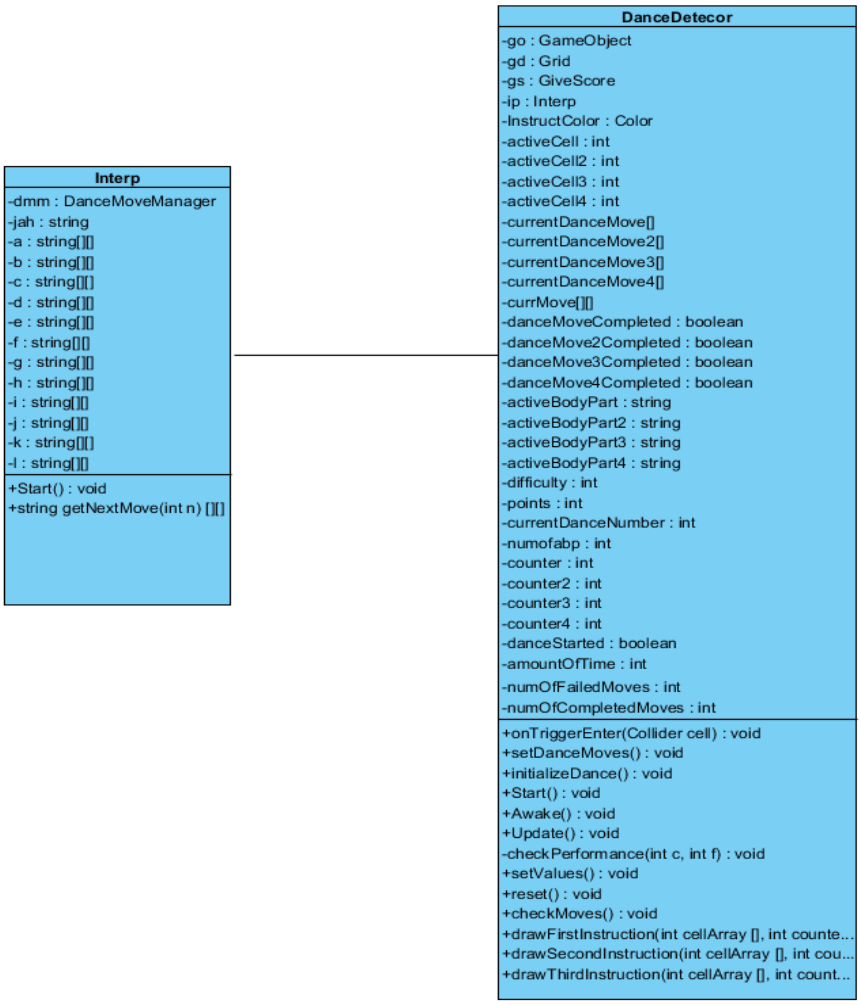
4.1 Sekvensdiagram

Sekvensdiagrammet viser hendelsesforløpet til interp. Interp venter til DanceDetector ber om en ny bevegelse fra Interp.



4.2 klassediagram

klassediagrammet viser klassen



Dansegenerator

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	22.05	Are Oven
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA, JEH

1. User stories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
104	Funksjon: Generator - Generere dans	Are Oven
111	Funksjon: Generator - Tilpasse dans	Are Oven
95	Arkitektur: Generator - Legge til bevegelser	Are Oven

2. Innledning

Det er viktig for oss å lage en dansegenerator som kan generere tilfeldige danser ut ifra hvilke utfordringer spilleren har. Dansegeneratoren får inn informasjonen den trenger når spilleren importerer en dans. Informasjonen den får inn, vil være basert på hvilke deler av skjermen spilleren har valgt som aktiv.

3. Implementasjon

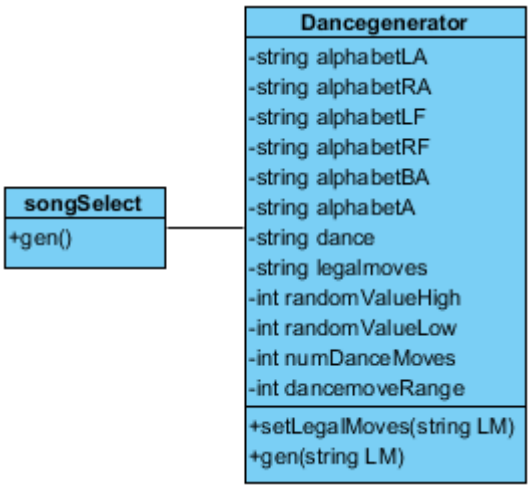
Når dansegeneratoren blir opprettet får den tilsendt en string-verdi som består av 0-ere og 1-ere som bestemmer hvilke kroppsdeler som skal brukes. Dette blir brukt til å hente frem hvilke bevegelser som er lovelige å bruke i den spesifikke dansen for den aktive brukeren. Generatoren har også en funksjon som skal hjelpe til med å få lignende bevegelser etter hverandre. På grunn av denne funksjonen vil det være fare for å få verdier som er utfor string-en som inneholder lovlige bevegelsene. For å motvirke dette har det blitt laget en løsning som setter variabelen til enten null eller lengden på stringen med lovlige bevegelser. Dette gjør at hvis man får verdier som er utenfor string-en eller havner i en av endene vil sannsynligheten være femti prosent for at neste bevegelse som kommer er den samme. Dette gjelder kun bevegelsene som ligger i enden.

Dansegeneratoren tar seg av det å tilpasse dansene. Dansebevegelsene har egne bokstavkoder, en bevegelse tilhører en bokstav. For eksempel vil abc tilhøre venstre hånd og def vil tilhøre høyre hånd. Hvis en spiller velger og bare å bruke den høyre hånd vil kun def bli lagt til i listen over lovlige bevegelser dansegeneratoren kan bruke til å generere dans. Når dansen er ferdig generert vil den bli returnert og lagret i databasen for videre bruk.

4. UML

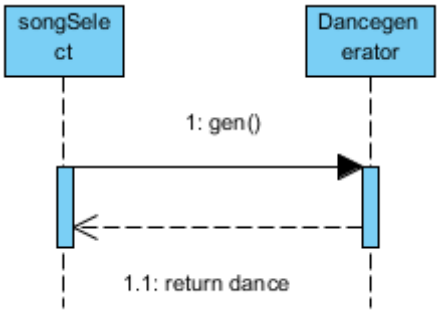
4.1 Klassediagram - Dancegenerator

Her vises klassen Dancegenerator, den blir opprettet av songSelect. Den får tilsendt en string og genererer en dans ut ifra dataen som ligger på den. Etter dette er gjort returnerer den en string som inneholder hele dansen.



4.2 Sekvensdiagram - Dancegenerator

Dette sekvensdiagrammet viser at songSelect kaller på gen() og får returnert en ferdig dans.



Avatar mot Kinect

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet	25.02.2013	Marcus Jernberg
2.0	oppdatert	18.03.2013	Marcus Jernberg
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	DK, ZA

1. User stories

Dokumentasjonen omhandler følgende userstories.

ID	Tittel	Ressurs
105	Dansescene – koble avatar mot kinect	Marcus

2. Innledning

User story 105 går ut på å få kontakt mellom kinecten og avataren. Det er flere måter dette kan løses på, jeg valgte å bruke Zigfu sin ZDK. Zigfu inneholder OpenNI og NITE (Beskrevet i TeknologiDokument: Zigfu).

3. Implementasjon

Med Zigfu fulgte det med en rekke "sample scenes". Fra den ene scenen hentet jeg de scriptene jeg trengte for å få kontakt mellom kinecten og avataren. Scriptene jeg hentet var:

- Zig
- ZigDepthViewer
- ZigEngageSingleUser
- ZigSkeleton

Hva scriptene gjør er forklart i teknologidokumentet for Zigfu.

ZigSkeleton er koblet til avataren.
Zig, ZigDepthViewer og ZigEngageSingleUser er koblet til et "gameobject"

nested for-løkke som iterer gjennom xRad, yRad og zRad bruker xPos, yPos og zPos som er variabler som kan endres i inspektør-vinduet. Dette er mulig siden disse variablene er satt til public.

3.1 Vurderinger

3.1.1 Hvorfor velge Zigfu

Grunnen til at vi endte opp med å bruke Zigfu var at det var disse som fungerte best opp mot avataren som er hentet fra Unity sin egen butikk. Zigfu hadde også gjort noen endringer i modellen slik at leddene kan kobles opp mot ZigSkeleton, slik at når brukeren beveger høyre underarm så beveger avatarens høyre underarm seg. Da slapp vi å endre modellene i 3D-modelleringsprogrammer først.

Testdokument

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Revisjon	Endringer	Dato	Skribent
1.0		Om testing	20.12.12	Marcus Jernberg
2.0		Prototyping, brukertesting	20.12.12	Zana Ali
3.0		Korrektur, Scrum,	21.12.12	Henrik Olsson
4.0		Sjekkliste	21.12.12	Are Oven
5.0		Lagt til test-case mal	20.03.13	Henrik Olsson
6.0				

Kontrollert av:	JEH, AO

Innholdsfortegnelse

1. Generelt om testing

Med testing ønsker vi å oppnå bedre kvalitet på produktet vi lager, færre bugs samt mindre backtracking. Det er veldig lett å overse feil på det vi jobber med, og ved å teste kan vi oppdage disse. Etter at man har funnet feil i et system må denne feilen lokaliseres og rettes, deretter må systemet testes på nytt. Det å påvise feil, og det og rette dem er to forskjellige aktiviteter. Tester benyttes for å påvise feil, mens det å lokalisere og rette opp feilen er en annen oppgave. Det kan lønne seg å lokalisere flere feil før man retter dem opp. Dette betyr at når vi finner en feil, lager vi en feilrapport. Denne feilrapporten skal inneholde hva feilen er, hvordan den oppstod og hvem oppdaget den.

Vi bruker verifisering og validering for å vurdere hva vi gjør og hvorfor vi gjør det. Behovet for verifisering øker ved størrelsen på prosjektet, og ca 1/3 av utviklingstiden brukes på testing. Softwaretesting er en del av " Verification and validation", V&V.

V&V er det generelle navnet som gies til all sjekking av produktet slik at vi vet at det følger spesifikasjonen og at kunden blir fornøyd. Systemet bør gå igjennom en V&V fase etter hvert utviklingstrinn.

Verifisering: Bygger vi produktet riktig? Verifisering involverer at vi sjekker at systemet virker etter spesifikasjonen. Vi må snakke med brukerne, i vårt tilfelle har vi et brukerpanel, for å avdekke om vi bygger produktet riktig ut fra kravene deres. Gjennom verifisering forsikrer vi oss om at produktet oppfører seg slik vi vil at det skal oppføre seg.

Validering: Bygger vi det riktige produktet? Validering er at vi sjekker at systemet virker slik kunden hadde tenkt at det skulle. Man evaluerer et system eller en komponent under, eller på slutten av en utviklingsfase for å bestemme om det tilfredstiller spesifiserte krav. Gjennom validering forsikrer vi oss om at det ikke har blitt gjort feil under utviklingen, slik at produktet vi leverer er det kunden ønsker.

1.1 Statisk og dynamisk testing

For å kunne gjennomføre V&V på et system inkluderer dette både statiske og dynamiske tester.

Statisk testing: Her vil man gjøre en kodegjennomlesning for å se om man har skrevet kildekode som vil fungere og ikke minst er riktig i henhold til implementasjonsdokumentet, dette kan bli gjort av utvikleren selv eller en annen i teamet for å få et annet synspunkt. Det som blir sjekket er; hvor lesbar koden er, stemmer alle syntakser og algoritmer. Statisk testing kan stort sett gjennomføres i alle fasene i et prosjekt. Og er en formell verifikasjon eller analyse.

Kode gjennomlesning ser vi på en form for statisk testing da vi vil se om programkoden vil stemme med det vi har satt som kodestandard, det vil bli utført av forfatteren selv. Når forfatteren mener koden er fullstendig vil den bli levert videre slik at en skal bli sett vurdert av en annen i utviklerteamet. Begge vil da følge en sjekk liste, samtidig som den andre personen vil vurdere lesbarheten i kildekoden.

Dynamisk testing: Skjer når vi simulerer/tester/bruker selve produktet som er produsert.
Det involverer å kompilere og kjøre kode for å teste. Finner oftest sted mot slutten av prosjektet. Og foregår ved hjelp av to typer test data:

- Statisk testing – Dette brukes for å kunne gi et tall på driftssikkerheten til systemet. Inndata til systemet må være lik de dataene som kommer til systemet når det er i virkelig drift-
- Feilfinning – Denne testtypen brukes for å påvise feil i systemet. Her er testene skrevet spesifikt for å fremprovosere feil i systemet. (Få systemet til å oppføre seg på en ulovlig måte)

2. Typer testing

Det er forskjellige måter å teste av programvare og programvarekode her blir det nevnt noen kjente typer; enhetstesting, integrasjonstesting, systemtesting, aksepttesting, regresjonstesting. Vi kan se på disseforskjellige testene som forskjellige nivåer i et hierarki, hvor vi begynner på bunn og går opp.

2.1 Enhetstesting

Enhetstesting eller unittest er det laveste nivået vi tester. Her tester vi funksjoner og klasser ved å sette opp en unittest som vil gjøre funksjonskall mange ganger hurtig, om funksjonene trenger parametre og returnerer verdier, så kan vi sette en forventet verdi og sammenlikne det med den faktiske verdien som blir returnert. Enhetstesting er det laveste nivået i testhierarkiet. Målet med enhetstesting blir å isolere en hver del av programmet og teste at kildekoden vil levere riktige verdier [1].

2.2 Integrasjonstesting

Integrasjonstesting kombinerer man forskjellige kildekodekomponenter, og tester for å se hvordan de samhandler. Man bruker gjerne både white-box og black-box testing til dette. Den som tester verifiserer at enheter og klasserelasjoner fungerer sammen når de kobles sammen. Selv om alle komponentene fungerer indiviuelt, er det ikke sikkert at de fungerer slik vi ønsker når de settes sammen/integreres. [2]

2.3 Systemtesting

Systemtesting utføres på et fullstendig større system, vi tester dette for å sjekke om det samsvarer med de spesifiserte kravene. Med systemtesting ønsker man å kontrollere at alle subsystemene fra integrasjonstesten fungerer sammen. Systemtesting har som mål å eksponere defekter i både de indre sammensetningene og i systemet som en helhet[3]. Her tester man gjerne ikke-funksjonelle egenskaper i programmet, som for eksempel:

- **Ytelsestesting** - Tester ytelsen i form av respons og stabilitet under en bestemt arbeidsbelastning.
- **Stresstesting** – Tester stabiliteten. Det involverer å teste forbi normal kapasitet.
- **Usabilitytesting** – Tester systemet opp mot brukeren, for å få en reell forståelse av hvordan systemet fungerer. [3]

2.4 Aksepttesting

Aksepttest er det høyeste nivået. Man tester systemet for å beslutte om det tilfredstiller kravene spesifisert av kunden. I SCRUM blir aksepttesting en slags funksjonell testing av en user story. En story kan ha flere aksepttester, de kan ha så mange som trengs for å sikre at funksjonaliteten fungerer. En user story er aldri ferdig før den har bestått alle aksepttester.

Aksepttesting er en formell test og er veldig likt som systemtesting. Forskjellen kommer av at det testes av "kunden", ofte i form av at noen tar på seg rollen som kunden. Vi skal også benytte oss av et brukerpanel som skal bruke programvaren dette kan vi se på som en del av aksepttesting. [4]

2.5 Regresjonstesting

Regresjonstesting gjøres ved å repetere tidligere kjørte test caser for å finne eventuelle nye feil i systemet som følge av ny kode eller andre endringer. Intensjonen med regresjonstesting er å forsikre seg om at de endringene som gjøres ikke fører til endringer i andre deler av kildekoden.

Vanlige metoder for regresjonstesting innebærer å repetere tidligere godkjente tester og sjekke om oppførselen til systemet har endret seg eller om gamle feil har gjenoppstått.

Siden vi benytter oss av SCRUM vil alle nivåene besøkes i hver sprint. På slutten av en sprint vil vi gjennomgå en aksepttesting, dvs. At en og en user story testes sammen om noe da ikke fungerer som det skal må vi gå tilbake. [5]

3. Teknikker for testing

Det er hovedsaklig to typer testing som gjelder når man tester. Disse teknikkene er black-box-testing og white-box-testing. Disse varierer mellom testerens ståsted når han ser på den ferdige programvaren eller programvarekoden.

3.1 Black-Box-Testing (Funksjonell-test)

Black-Box-testing baserer seg på at tester skal lages med utgangspunkt i kravspesifikasjonen. Vi kan se på systemet som en svart boks med en rekke innganger og et sett med utganger. Man kan teste ved bruk av black-box i alle de forskjellige testnivåene (enhet, integrasjon, system og aksept).
Det er ikke nødvendig for testeren å ha kunnskap om programmets kode/intern struktur. Testeren trenger bare å vite hva programvaren er ment til å gjøre, men ikke hvordan den gjør det. Når en viss input er angitt, blir en viss output returnert. Hvordan dette blir produsert skal ikke testeren ta hensyn til, bare at input og output stemmer overens.

3.2 White/Glass-Box-Testing(Strukturell-testing)

Denne metoden brukes i tillegg til black-box testing. Den bygger på kjennskap til den interne oppbygningen av systemet. Fordelen med dette er at den som tester kan analysere oppbyggningen og finne hvor mange testscenarioer som er nødvendig for å sikre en viss sikkerhet for å oppdage de feilene som finnes. En vanlig regel i denne sammenhengen er at alle kodelinjer skal kjøres minst en gang. White-box testing/analyse kan også av og til hjelpe de som skal identifisere klassen i black-box testing med jobben. Et eksempel på white-box testing blir å følge koden med et debug verktøy og hele tiden holde oversikt over hvordan variabler oppfører seg i henhold til koden.

3.3 Prototyping

En prototype er en foreløpig utgave eller et utkast til hvordan et system kan se ut. Det kan skje ved blant annet storyboarding med tegninger, eller skjermbilder uten funksjonaliteter. Prototypen lages før en starter produksjon av en vare. Formålet med en prototype er å demonstrere og teste funksjon og design. Det er for å validere systemet i krav og analysefasen[6, 7].

3.3.1 Software prototyper

Software prototyper operer ganske annerledes fra standard prototyper, da de ikke er fysiske modeller, men heller en alfa-versjon av et program. Begrepet alfa refererer til det faktum at prototypen er den første versjonen av programmet som skal kjøres, med etterfølgende prototyper nevnt i den rekkefølgen de er utviklet (beta, gamma osv). I alfa-versjonen skal som regel bare grunnleggende funksjoner være til stedet, for å ha noe som skal bygges på videre. Etter at flere funksjoner er lagt til alfa-versjonen, beveger prototypen seg inn mot beta, fordi det i hovedsak fungerer son en mer avansert prototype .

3.3.2 Andre prototyper

- Skissemodeller: Viser designideer
- Funksjonsmodeller: Demonstrer funksjoner
- Utseendemodeller: Viser ytre design slik produktet skal se ut, uten at modellen fungerer som produktet.
- Førserie prototype: Alle komponenter er lik produktet, men forskjellen er at komponenter er laget under industrielle produksjonsverktøy

3.3.3 Fordeler og ulemper

Det er enkelt for brukere å forholde seg til, sammenlignet med modeller og dokumenter. Det avdekker effektivt manglende eller feil krav og forretningsregler. I tillegg kan det være sparsomt økonomisk. Noen av ulempene er at gode prototyper kan gi falske forventninger i forhold til hvor langt prosjektet har kommet. Det kan også lede til at brukere blir opphengt i detaljer som er irrelevante [6, 7].

4. Test Case

En test case i software-sammenheng er et sett med test-inputs, betingelser, variabler og forventede resulater, for å hjelpe testpersonen til å avgjøre om applikasjonen eller systemet fungerer korrekt eller ikke. Man bør lage test caser så tidlig som mulig. Hvis man venter for lenge vil test casene vanligvis bli dårligere, og man vil begynne med ad-hoc testing, som betyr at man vil teste det man kommer på etterhvert.

I en test case skal man følgende punkter:

- Test-ID
- Test Case-beskrivelse
- Testtype
- Forventet resultat
- Faktisk resultat

Siden vi benytter SCRUM som prosjektmodell tar vi med story-ID , dette for å kunne se hvilken userstory som testes.

Under beskrivelse, beskriver man spesifikt et sett med steg og/eller inputs for det du vil teste, dette inkluderer eventuelle forberedelser. I faktisk resultat skriver man PASS eller FAIL. Hvis testen FAILer er det greit å ha med beskrivelse av hva som gikk galt. Det er viktig å ha spesifikke beskrivelser for å kunne gjenskape eventuelle feil. Etter gjennomført test kan det inkluderes flere felter, som f.eks:

- Hvem som utførte testen og når det ble utført
- Kommentarer

Vi lagerer test caser og tillhørende user story i produkt backloggen.

Test case – ID #			
Story ID		Story tittel	
Hva testes			
Type test			
Story beskrivelse			
Utførelse			
Kriterier			
Kommentar			
Test nr		Resultat	Ukjent
Dato utført		Utført av	

Test-Case mal

5. Hvordan vi skal teste

Basert på dette dokumentet har gruppa bestemt seg for hvordan og hva vi skal teste.

5.1 Brukertesting

Brukertesting er en evaluering der man observerer og analyserer hvordan funksjoner i en løsning blir brukt av faktiske brukere. Brukertesting er den mest valide metoden for å dekke svakheter ved brukervennligheten til en programvare. Hensikten er å få en reell tilbakemelding fra brukere av programvaren, og bruke disse til å forbedre den.

I forbindelse med testing av programvare opp mot brukeren, skal vi ha ukentlige møter på Sunnass der et brukerpanel bestående av ergoterapeuter. Vi skal observere flere spillsesjoner, mest for å kartlegge. Vi vil se pasientene utføre oppgaver I en tidlig fase har vi behov for en del informasjon og svar på spørsmål om behovsdrevet design, disse ergoterapeutene tilbyr oss dette.

5.2 Metode og prosedyre for testing

En måte er å liste opp en sjekkliste over ting som skal gjøres for brukerne. Deretter får man en tilbakemelding på hvor bra disse funksjonene er, og hvor god brukervennligheten er. En rapport kan lages om hva de synes, dette er en del av kvalitetskontroll.
En annen måte er å la brukerne få teste og svare på et spørreskjema de får utdelt. Det kan inneholde vurderinger som en rangering av brukeropplevelsen fra 1-10 [9].

5.3 Brukervennlighet og brukeropplevelse

En definisjon på brukervennlighet er et produkt eller en løsning er inuitiv for en person å bruke, men det kan også være mye mer enn det. Ved vurdering av brukervennlighet ser man på hvordan ting henger sammen og hvordan man kan manøvrere seg rundt. Navigasjon og interaksjonsdesign er de to tingene som blir vurdert her. En brukervennlig programvare vil ha hjelpefunksjoner dersom brukeren trenger assistanse.Grafiske elementer som ikoner, symboler, piler knapper og så videre må være lett forståelige. For å få en god brukervennlighet kan det knyttes til relevans og behov.

Brukeropplevelsen er det i en veldig stor grad kun brukeren av systemet som kan svare på hvordan er. Som utvikler av programvaren, er det elementer som påvirker brukeropplevelser som blir forsket og testet på. Det er ikke lett å beskrive brukeropplevelse, for det er mer kunst enn vitenskap siden opplevelsene er individuelle og påvirket av hvem vi er [10].

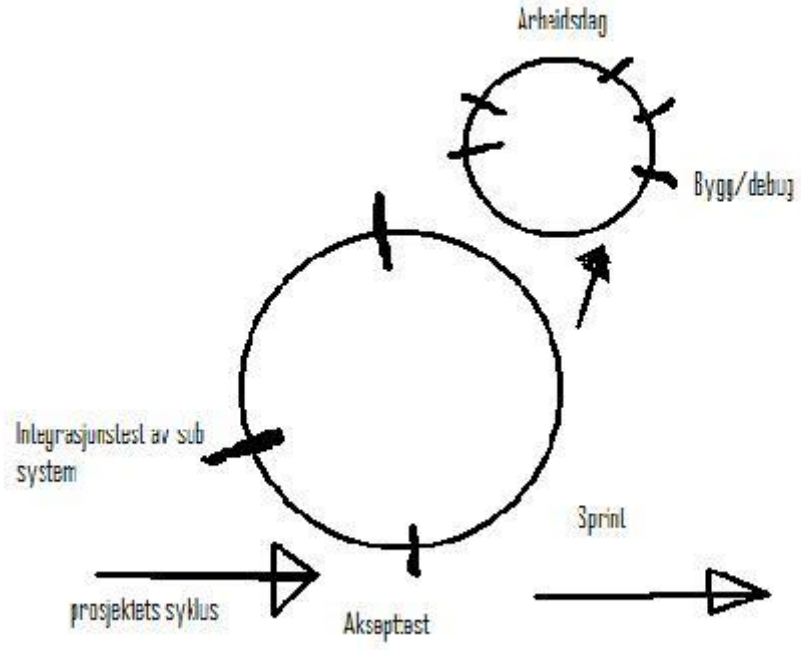
5.4 Styrker og svakheter med brukertesting

Ved brukertesting får man en reell tilbakemelding av en reell bruker, som vi ikke kan få ved å teste programvaren selv som utviklere av det. Det koster heller ingenting ikke noe å få tilbakmelding av brukere. Det er som nevnt den mest valide metoden for å dekke svakehter ved brukervennligheten til en programvare.

Man får kun testet et representativt utvalg av testpersoner, og resultatene er basert på deres subjektive preferanser og ferdigheter. Brukerne kan bli hemmet av at de blir observert. Testen vil også kun avdekke et utvalg av scenarier, og ikke dekke alle oppgaver [9].

6. Testing og Scrum

I Scrum er vi alltid inne i en sprint, sprintene har alltid fast tidsintervall. I hver sprint utfører vi bestemte userstories som koblet sammen vil danne et slags sub system. Hver arbeidsdag hvor vi implementerer faller det naturlig at vi bygger, kompilerer og debugger hver gang vi har skrivet en bit med programkode. Dette er også en form for testing, dette ligger på nivå med Enhet/Unit – test, dette vil forekomme oftest. Når vi så kobler sammen forskjellige sett med arbeid må dette *merges* så testes dette er en integrasjonstest av et sub system, dette vil foregå noen ganger i en sprint. Når en sprint slutt er nådd skal ferdig utviklet arbeid testet ved hjelp av systemtest, regresjonstest eller integrasjonstest. Mellom sprinter vil vi ha møter med oppdragsgiver da vil vi vise deler av produktet og gjør så en aksepttest.



Figur: Her ser vi prosjektets syklus mtp. Scrum og Testing

7. Sjekkliste for Code review:

Når vi er ferdige med koden skal vi gjennomgå sjekklisten som er vist nedenfor. For at koden skal bli godkjent må den bestå alle kravene som listen under inneholder, hvis koden feiler på ett av punktene må dette fikses før man går gjennom listen på ny og sjekker om koden består alle kravene. Når alle kravene på listen er bestått er den klar til å bli sendt videre.

Sjekklisten:

Tegnsetting	Sjekke at det er brukt korrekte tegn, står krokodilletegnet rett vei osv.
Grenseverdier	Sjekke at grenseverdiene er korrekte, f.eks. når man lager løkker og slik må man passe på at det er rette verdier som brukes.
Deadlocks	Se om det kan bli laget deadlocks når man driver med trådprogrammering.
Uendelige løkker	Sjekke at ingen av løkkene har mulighet for å være en evig løkke. Se at det er korrekte verdier som er gitt.
Kodestandard	Sjekke at koden følger den oppsatte kodestandarden for prosjektet.
Divided by zero	Sjekke at det ikke er mulighet for å komme i situasjonen der noe blir delt på null.

8. Ordliste

Ergoterepeut: Helsearbeider som jobber med mennesker som har fysiske og/eller psykiske lidelser, skader og sykdommer.

Programvare: Dataprogrammer.

Interaksjonsdesign: Definerer atferden til produkter og systemer som mennesker kan samhandle med.
Unittest: Enhetstest.

Backtracking: Generell metode for å finne noen, eller alle løsninger på beregninger.

9. Referanser

1. Wikipedia. *Enhetstesting*. 2013 03.01.2013]; Available from: http://en.wikipedia.org/wiki/Unit_testing.
2. Wikipedia. *Integrasjonstesting*. 2013 03.01.2013]; Available from: http://en.wikipedia.org/wiki/Integration_testing.
3. Wikipedia. *Systemtesting*. 2013 03.01.2013]; Available from: http://en.wikipedia.org/wiki/System_testing.
4. Wikipedia. *Aksepttesting*. 2013 03.01.2013]; Available from: http://en.wikipedia.org/wiki/Acceptance_testing.
5. Wikipedia. *Regresjonstesting*. 2013 03.01.2013]; Available from: http://en.wikipedia.org/wiki/Regression_testing.
6. Wikipedia. *Prototype*. 2012 18.12.12]; Available from: http://no.wikipedia.org/wiki/Prototype.
7. Ribu, K. *Validering og verifisering*. 2005 18.12.12]; Available from: http://www.iu.hio.no/~kirstenr/systemutvikling/slides_host2005/7_Validering_verifisering.ppt.
8. thomasnet. *Prototypes in Electronics, Computer Software, and Computer Engineering*. 2012 18.12.12]; Available from: http://www.thomasnet.com/articles/engineering-consulting/computer-electronic-prototypes.
9. Wikipedia. *Brukertesting*. 2012 19.12.12]; Available from: http://no.wikipedia.org/wiki/Brukertesting.
10. Tronerud, J. *Usability og mennesker*. 2012 20.12.12]; Available from: http://brukertest.com/.

Test Cases

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Userstory ID	Testcase ID	Kort beskrivelse av test
48	1	Black-Box, systemtest: Logg inn
50	2	Black-Box, systemtest: Logg inn
49	3	Black-Box: Prestasjoner/Achivements
95	4	Black-Box: Generator - Legge til bevegelser
	6	White-Box: Generator - Legge til bevegelser
133	5	Black-Box: Menysystem - New user
	7	White-Box: Menysystem - New User
106	8	Black-Box: Dansescene - Grid
123	9	White-Box: Database - Grunnleggende metoder
	10	White-Box: Database - Grunnleggende metoder
147	11	Black-Box: Koble avatar mot grid
	12	Black-Box: Koble avatar mot grid
105	13	Black-Box: Koble avatar mot kinect
103	14	Black-Box: Kontakt med kinect
113	15	White-Box: Meny
92	16	Black-Box: Audio Import
92	17	Stress-test: Audio Import
170	18	White-box: Aktivkroppsdel
170	19	White-box: Flere kroppsdeler
110, 193	20	Black-box: Hente dansebevegelser
170	21	Stress-test: Dans - Deteksjon
192	22	Unit-test: Musikkanalyse
181	23	Unit-test: Statistikk-Stolpediagram
211	24	Black-box: Statistikk-Database
107	25	Black-box: Dynamisk vanskelighetsgrad
197	26	White and Black - box: Dynamisk vanskelighetsgrad
105, 103, 106, 59, 169, 189, 112, 127, 146, 147, 170, 171	27	Black-box, Systemtest: Oppfattebruker bevegelser
61, 70, 85, 94, 96, 98, 121, 123, 175, 176, 179, 181, 190, 209, 205	28	Black-box, Systemtest: Følge statistikk
111, 80, 52, 51, 197, 203, 107	29	Black-box, Systemtest: Brukers behov
180	30	Unit-test: Prestasjoner - database
180	31	Unit-test: Prestasjoner - database
180	32	Black-box: Prestasjoner - database
180	33	Black-box: Prestasjoner- database
54, 56, 53, 84, 85, 87, 113, 114, 116, 117, 133, 134, 135, 136, 137, 204	34	Brukertest, systemtest: Enkelt brukergrensesnitt
66, 81, 107, 197	35	Black-box, systemtest: Justerbar vanskelighetsgrad
55,121	36	Black-box, systemtest: Brukerkonto

92, 82, 172, 173, 191, 192	37	Black-box, systemtest: egen musikk
49, 62, 83, 108, 175, 178, 189	38	Black-box, systemtest: Prestasjoner
104, 200, 206, 111, 199, 95, 101, 196	39	Black-box, systemtest: Autogen musikk til egen musikk
152	40	Black-box, systemtest: Design av egen avatar
153	41	Black-box, systemtest: Manuelt design av dans
154	42	Black-box, systemtest: Automatisk deteksjon av funksjonsnivå
155	43	Black-box, systemtest: Flerspiller
56	44	Black-box: Brukergrensesnitt
54	45	Black-box: Quick Entry
189	46	White-box: Dans - Grid- Dybde
111	47	White-box: Tilpasse dans
111	48	Black-box: Tilpasse dans
206	49	White-box: Opprette bevegelser
206	50	Black-box: Opprette bevegelser
191	51	Black-box: Musikkmappe
203	52	Black-box: Tilpass scene
202	53	Black-box: Startup scene
	54	Systemtest: flyt i meny, knapper og input

Test case – ID 1			
Story ID	48	Story tittel	
Hva testes	Logg inn		
Type test	Blackbox, systemtest		
Story beskrivelse	Som bruker ønsker jeg å ha muligheten til få oppført statistikk over mine prestasjoner for å kunne holde oversikt over fremgangen.		
Utførelse	Vi tester dette ved å lagre og hente ut data fra databasen.		
Kriterier	Vi får lagret og hentet ut data		
Kommentar	Tilbakemeldingene ved feil brukernavn/passord er korrekte. Vi blir logget inn og sendt videre i menyen.		
Test nr			
Dato utført		Utført av	

Test case – ID 2			
Story ID	50	Story tittel	
Hva testes	Logg inn		
Type test	BlackBox, Systemtest		
Story beskrivelse	Som en bruker ønsker jeg å ha muligheten til å se min fremgang i forhold til andre for å kunne skape konkurranse.		
Utførelse	Det skal spilles, og brukerne skal få riktig tilbakemelding avhengig av oppnådd poengscore.		
Kriterier	Det skal være mulig å konkurrere med andre ved hjelp av oppnådde poengsummer.		
Kommentar	Tilbakemeldingene ved feil brukernavn/passord er korrekte. Vi blir logget inn og sendt videre i menyen.		
Test nr			
Dato utført		Utført av	

Test case – ID 3			
Story ID	49	Story tittel	
Type test	Blackbox		
Story beskrivelse	Som en bruker ønsker jeg å få prestasjoner/achivements for å motivere til å fortsette å spille spillet.		
Utførelse	Vi lager en prestasjon som kan oppnås på kommando.		
Kriterier	Når kriteriet for å låse opp prestasjonen er fullført(i dette tilfellet når kommandoen utføres) skal prestasjonen låses opp, men den skal kun låses opp en gang per bruker.		
Type test	Blackbox		
Kommentar	Tilbakemeldingene ved feil brukernavn/passord er korrekte. Vi blir logget inn og sendt videre i menyen.		
Test nr			
Dato utført		Utført av	

Test case – ID 4			
Story ID	95	Story tittel	Arkitektur: Generator - Legge til bevegelser
Hva testes	Generering av dans		
Type test	Blackbox		
Story beskrivelse	Som en utvikler ønsker jeg en generator som legger til bevegelser som kan brukes i en dans		
Utførelse	Vi bruker en random-generator som velger ut hvilke dansebevegelser som skal brukes		
Kriterier	For at den kan regnes som fullført må generatoren velge ut tilfeldige dansebevegelser som passer sammen og lagre dem		
Kommentar	Vi får ut den infoen vi vil ha ut når den blir kjørt		
Test nr	1	Resultat	Godkjent
Dato utført	17.03.2013	Utført av	Are Oven

Test case – ID 5			
Story ID	133	Story tittel	Arkitektur: Menysystem – New User
Hva testes	Tilbakemeldinger ved mangler av utfylt informasjon		
Type test	Blackbox		
Story beskrivelse	Som en utvikler ønsker jeg å lage arkitekturen for ny bruker til vårt system.		
Utførelse	Bevist å legge inn informasjon med mangler for å trigge feilmeldingene i systemet. Sjekker også om brukeren eksisterer. Dette gjøres ved at jeg har sagt at det er en bruker "bruker" som eksisterer. Følgende mangler blir testet: • For kort passord, ikke passord • Passord som ikke matcher • Ikke noe informasjon • Ikke noe brukernavn • Ikke noe navn		
Kriterier	Skal gi korrekt tilbakemelding på mangler. Brukeren skal ikke sendes videre uten at alt er riktig fylt ut.		
Kommentar	Alle feilmeldingene kom frem når de skulle og brukeren får beskjed om hva som er feil. Brukeren blir sendt videre når alt er fylt ut korrekt.		
Test nr	1		Godkjent
Dato utført	17.03.13	Utført av	Jan Erik Helskog

Test case – ID 6			
Story ID	95	Story tittel	Arkitektur: Generator - Legge til bevegelser
Hva testes			
Type test	Whitebox		
Story beskrivelse	Som en utvikler ønsker jeg en generator som legger til bevegelser som kan brukes i en dans		
Utførelse	Vi bruker en random-generator som velger ut hvilke dansebevegelser som skal brukes		
Kriterier	For at den kan regnes som fullført må generatoren velge ut tilfeldige dansebevegelser som passer sammen og lagre dem		
Kommentar	For å sjekke at man får rett verdier skrives det ut verdiene før de blir forandret før og etter at de har blitt forandret til nye verdier. Disse verdiene har deretter blitt sjekket at de stemmer med det som skal komme ut		
Test nr	2	Resultat	Godkjent
Dato utført	17.03.2013	Utført av	Are Oven

Test case – ID 7			
Story ID	133	Story tittel	Arkitektur: Menysystem – New User
Hva testes	Registrering av hvilke del av skjermen som skal bli brukt.		
Type test	Whitebox		
Story beskrivelse	Som en utvikler ønsker jeg å lage arkitekturen for ny bruker til vårt system.		
Utførelse	Huker av forskjellige alternativer for å sjekke om verdien til variablene blir satt riktig.		
Kriterier	Variabelverdiene skal ha riktig verdi i henhold til hvilke bokser som er huket av		
Kommentar	Variabelverdiene er korrekte i alle tilfellene som er testet.		
Test nr	2		Godkjent
Dato utført	17.03.13	Utført av	Jan Erik Helskog

Test case – ID 8			
Story ID	106	Story tittel	Arkitektur: Dansescene - Grid
Hva testes	Kollisjon deteksjon med ulike celler		
Type test	Black-Box		
Story beskrivelse	Teste om kollisjon mellom et objekt og cellene stemmer overens med cellen som detekterer kollisjonen.		
Utførelse	En debug som skriver ut hvilke celler som har blitt kollidert med, og endre farge på cellen som kollideres med.		
Kriterier	Detektere at riktig celle har blitt kollidert med, etter at objektet ikke er i kontakt med cellen lenger skal kollisjonen opphøre		
Kommentar	Dette fungerte slik det skulle, noe utskriften i debugloggen viste og fargeendringene ved kollisjon.		
Test nr	1	Resultat	Godkjent
Dato utført	07.03.13	Utført av	Zana Ali

Test case – ID 9			
Story ID	123	Story tittel	Arkitektur: Database, grunnleggende metoder.
Hva testes	Hvor robust er genereringen av SQL setninger		
Type test	Whitebox		
Story beskrivelse	Som utvikler ønsker jeg metoder som kan sette sammen setninger med spørringer som kjøres mot databasen slik at spørringene ikke må hardcodes inn i spillet.		
Utførelse	Jeg bruker dbTester klassen for å kalle på metoden som oppretter nye brukere og velger et navn navn som inneholder tegn som benyttes i SQL syntaxen for å se om dette ødelegger spørringen slik at den ikke kan utføres. Jeg velger å opprette en bruker med navnet " ' Test1 ".		
Kriterier	Spørringene godtas og riktig brukernavn opprettes.		
Kommentar	Testen feilet da spørringen for å sjekke om brukernavnet allerede eksisterer ble kjørt. '-tegnet jeg bruker i navnet benyttes i SQL syntaxen og når jeg setter dette inn i navnet lages en ugyldig spørring. Dette kan løses ved å utføre en sjekk på all input, slik at man fjerner alle tegn som kan bety trøbbel. Det tryggeste vil være å begrense antall gyldige tegn direkte. Jeg foreslår at dette gjøres direkte i GUIen der inputen kommer fra.		
Test nr	1	Resultat	Feilet
Dato utført	13.03.2013	Utført av	Dagfinn Kjærnet

Test case – ID 10			
Story ID	123	Story tittel	Arkitektur: Database, grunnleggende metoder.
Hva testes	Kontakten mot databasen, kan vi skrive til og lese fra databasen?		
Type test	Whitebox		
Story beskrivelse	Som utvikler ønsker jeg metoder som kan sette sammen setninger med spørringer som kjøres mot databasen slik at spørringene ikke må hardcodes inn i spillet.		
Utførelse	Jeg oppretter en egen testklasse som kaller på de nødvendige metodene i klassen dbHandler, jeg ønsker å teste kommunikasjonen mot databasen, og at jeg får både skrevet til og hentet fra databasen. Jeg ønsker også å finne ut om SQL setningene konstrueres riktig. Dette skal testes ved å legge til en bruker, for deretter å prøve å "logge inn" brukeren både med feil passord og riktig passord.		
Kriterier	Testen er vellykket hvis man får opprettet en ny bruker i databasen og får hentet ut igjen den lagrede informasjonen så man får sammenlignet inntastet passord og passordet som er lagret i databasen.		
Kommentar	Jeg hadde endel problemer med å få metodene til å fungere. Metoden for å opprette brukeren skyldtes det at jeg hadde brukt feil syntax i SQLen og spørringene ble derfor feil. Den andre feilen var i metoden for å logge inn, det viste seg at jeg hadde glemt en while-løkke for å lese SQL-readeren. Etter å ha rettet opp i disse feilene fungerte database-scriptet som det skulle, både skriving til og avlesing fra databasen fungerte.		
Test nr	1	Resultat	Godkjent
Dato utført	13.03.2013	Utført av	Dagfinn Kjærnet

Test case – ID 11			
Story ID	147	Story tittel	Prototype: Koble avatar mot grid
Hva testes	Kollisjon deteksjon mellom avatar og cellene når avataren er i bevegelse. Med mesh renderer knyttet til avataren		
Type test	Black-Box		
Story beskrivelse	Teste om cellene detekterer kollisjon dersom avataren beveger seg i griden ved å knytte mesh renderer til avataren.		
Utførelse	Skrive betingelser for å sjekke om kollisjon oppstår mellom avataren og griden, farge de cellene avataren er i kontakt med som rød.		
Kriterier	Detektere at avataren beveger seg i griden		
Kommentar	Denne testen feielt da mesh renderer ikke fungerte slik den skulle. Når avataren stod i ro var ting slik det skulle være og de cellene som var i kontakt med avataren ble rød. Men når avataren var i bevegelse stod fortsatt mesh renderer stille og oppdaterte seg ikke etter avatarens posisjon.		
Test nr	1	Resultat	Feilet
Dato utført	11.03.13	Utført av	Zana Ali, Marcus Jernberg

Test case – ID 12			
Story ID	147	Story tittel	Prototype: Koble avatar mot grid
Hva testes	Kollisjon deteksjon mellom avatar og cellene når avataren er i bevegelse. Med "cubes" festet til armene		
Type test	Black-Box		
Story beskrivelse	Teste om cellene detekterer kollisjon dersom avataren beveger seg i griden ved å feste "cubes" til hendene med box colliders		
Utførelse	Skrive betingelser for å sjekke om kollisjon oppstår mellom cubene festet til hendene på avataren og griden, farge de cellene cubene er i kontakt med som rød.		
Kriterier	Detektere at områdene i griden cubene er i kontakt med blir røde.		
Kommentar	Denne testen ble godkjent. Vi prøvde først med mesh renderer for hele avataren. Da dette ikke fungerte prøvde vi cube og box colliders, noe som fungerte utmerket. Vi gjorde cubene relativt små og usynelige, slik at det ser ut som det er armene som blir detektert. Vi kan ved senere tidspunkt feste cuber til alle leddene som skal detekteres, på samme måte som vi gjorde her.		
Test nr	2	Resultat	Godkjent
Dato utført	11.03.13	Utført av	Zana Ali, Marcus Jernberg

Test case – ID 13			
Story ID	105	Story tittel	Dancescene – Koble avatar mot kinect
Hva testes	Gjennspeiler brukerens bevegelser seg i avataren ved bruk av kinect		
Type test	Black-box		
Story beskrivelse	Som utviklere ønsker vi å ha en arkitektur for å koble avatar mot kinect		
Utførelse	Jeg tester selv ved å bruke kinecten og avataren i unity for å teste at avatarens bevegelser gjennspeiler de bevegelsene jeg utfører		
Kriterier	Avatarens bevegelser skal gjennspeile brukerens		
Kommentar	Det fungerer bra. Kinecten plukker opp noen objekter i bakgrunnen som kan føre til bugs(f.eks venstrefoten til avataren flyttes ut av posisjon), men det skyldes at det er dårlig plass der testen ble utført.		
Test nr	1	Resultat	Godkjent
Dato utført	11.03.2013	Utført av	Marcus Jernberg

Test case – ID 14			
Story ID	103	Story tittel	Dancescene – Kontakt med kinect
Hva testes	Får vi kontakt mellom Unity og kinect		
Type test	Black-Box		
Story beskrivelse	Som utviklere ønsker vi å ha kontakt mellom Unity3D og kinect		
Utførelse	Jeg testet ved å laste ned Microsoft Kinect SDK, OpenNI og Zigfu ZDK. Jeg koblet kinecten til Pcen og prøvde å kjøre "sample scenes" som fulgte med Zigfu ZDK.		
Kriterier	Ser at Unity3D har kontakt med kinecten og at modellen i scenene som fulgte med Zigfu følger mine bevegelser.		
Kommentar	Det fungerer som forventet. Unity og Kinecten fikk kontakt, og scenene kjørte som forventet.		
Test nr	1		Godkjent
Dato utført	11.03.2013	Utført av	Marcus Jernberg

Test case – ID 15				
Story ID	113	Story tittel	Arkitektur: Meny	
Hva testes	Logg inn			
Type test	Whitebox			
Story beskrivelse	Som en utvikler ønsker jeg å lage arkitekturen for menysystemet Kommentar: Mye tid ble brukt til å sette seg inn i NGUI og lage prefabs til videre arbeid. User Story delt opp til mindre biter.			
Utførelse	Lager en bruker "bruker" med tilhørende passord "passord". Skal så teste først med feil brukernavn/passord for og deretter prøve med korrekt informasjon			
Kriterier	Korrekte feilmeldinger skal komme ved feil brukernavn/passord. Skal bli logget inn når informasjonen som blir tastet inn er korrekt.			
Kommentar	Tilbakemeldingene ved feil brukernavn/passord er korrekte. Vi blir logget inn og sendt videre i menyen.			
Test nr	1		Godkjent	
Dato utført	18.03.13	Utført av	JEH	

Test case – ID 16				
Story ID	92	Story tittel	Audio Import	
Hva testes	Importering av lydfil			
Type test	Blackbox			
Story beskrivelse	Brukeren av applikasjonen skal ha mulighet til å velge en lyd fil fra directory for så å importere og spille av denne.			
Utførelse	Taste inn pathen til filen i tekstfeltet, så trykke på import.			
Kriterier	Jeg som utviklet applikasjonen, vet at unity har kun støtte for å importere .ogg format i runtime. Nå er det også slik at banen/pathen må være korrekt for å kunne hente filen.			
Kommentar	Dette er en veldig simpelt brukergrensesnitt, det burde vært muligheter for å kunne få opp en gui filstruktur for å kunne hente ut filer fra forskjellige steder på maskinen, eventuelt så kan vi designe applikasjonen slik at man kan ha en spesifikk mappe på pcen hvor brukeren legger sine musikkfiler.			
Test nr	1	Resultat	Godkjent	
Dato utført	07.03.13	Utført av	Henrik Olsson	

Test case – ID 17			
Story ID	92	Story tittel	Audio Import
Hva testes	Stabiliteten til applikasjonen		
Type test	Stresstest		
Story beskrivelse	Brukeren av applikasjonen skal ha mulighet til å velge en lyd fil fra directory for så å importere og spille av denne.		
Utførelse	Jeg tester stressnivået til applikasjonen ved å hente inn lydfilene om og om igjen rett etter hverandre, ved å trykke barnslig fort på importer knappen.		
Kriterier	Banen/Pathen må være riktig, filen må være .ogg format		
Kommentar	Alt fungerte etter planen, jeg rakk å klikke 10-15 ganger før programvaren hang seg. Programmet kræsjet ikke, men funksjonsnivået ble satt ned.		
Test nr	2	Resultat	Godkjent
Dato utført	07.03.13	Utført av	Henrik Olsson

Test case – ID 18			
Story ID	170	Story tittel	Funksjon: Dans – Deteksjon – ressurs #1
Hva testes	Sjekking av aktiv kroppsdel mot den angitte aktive kroppsdel		
Type test	White		
Story beskrivelse	Brukeren skal følge angitte bevegelsesmønsteret, programvaren skal sjekke at den rette kroppsdelen treffer det rette punktet.		
Utførelse	Vi tester rett og slett om programvaren gjør som vi ønsker, vi starter et spill og beveger avataren til de gitte posisjonene i danserommet mens vi føler med i debug loggen. Dette blir gjort uten kinect. Vi looper dansen til "evig tid".		
Kriterier	Dansebevegelsene må bli detektert med riktig kroppsdel.		
Kommentar	Programmet gjør det, det skal ved vanlig bruk. Testen ble bare utført ved å gjøre to dansebevegelser.		
Test nr	2	Resultat	Godkjent
Dato utført	09.05.13	Utført av	HO & ZA

Test case – ID 19			
Story ID	170	Story tittel	Funksjon: Dans – Deteksjon – ressurs #1
Hva testes	Flere kroppsdeler mot bestemt celler.		
Type test	Whitebox		
Story beskrivelse	Brukeren skal kunne bruke flere kroppsdeler til å fullføre en dansebevegelse.		
Utførelse	Vi setter ut flere aktive celler som er knyttet opp mot bestemte kroppsdeler.		
Kriterier	Det er tillatt at de forskjellige kroppsdelene når punktene sekvensielt.		
Kommentar	Testen ble gjennomført, ingen problemer ble lagt merke til.		
Test nr	3	Resultat	Godkjent
Dato utført	09.05.13	Utført av	HO & ZA

Test case – ID 20			
Story ID	110, 193	Story tittel	Funksjon: Dansescene - Instruktør
Hva testes	Hente dansebevegelser		
Type test	Black-Box		
Story beskrivelse	Denne testen går ut på å hente danser fra en annen klasse og bruke verdiene til å aktivere dansen		
Utførelse	Debugging i form av utskrift til skjerm og grafisk visning av det som skjer.		
Kriterier	Hente inn en dansebevegelse og bruke denne som aktive dansebevegelse		
Kommentar	Verdiene ble hentet og kunne brukes til å lage en dans.		
Test nr	1	Resultat	Godkjent
Dato utført	03.05.13	Utført av	Zana Ali, Are Oven

Test case – ID 21			
Story ID	170	Story tittel	Funksjon: Dans – Deteksjon – ressurs #1
Hva testes	Om systemet tåler stress påført av brukeren.		
Type test	Stresstest		
Story beskrivelse	Brukeren skal følge angitte bevegelsesmønsteret, programvaren skal sjekke at den rette kroppsdelen treffer det rette punktet.		
Utførelse	Vi beveger avataren til målpunktene så fort som over hodet mulig for å stresse systemet.		
Kriterier	Programvaren skal være solid nok til å ståle stresset påført av input dataen fra kinect.		
Kommentar			
Test nr	4	Resultat	Godkjent
Dato utført	09.05.13	Utført av	HO & ZA

Test case – ID 22			
Story ID	192	Story tittel	Funksjon: Musikkanalyse, resurs 2
Hva testes	Vi tester om verdiene BeatLocator stemmer overens med sangen.		
Type test	Unit test		
Story beskrivelse	Som utvikler ønsker vi en måte å analysere musikken på slik at vi kan finne passende steder å plassere dansebevegelser.		
Utførelse	Jeg har laget en egen test-metode som printer en beskjed i debug-loggen etter de tidsverdiene analyse-metoden har funnet, og spiller av musikk filen samtidig. For at dette skal gå må analyse objektet kjøres i en egen tråd.		
Kriterier	Det skal være mulig å se og høre en sammenheng mellom de beatene vi hører i sangen, og når debug-loggen sier det skal være en beat. Da dette er en unøyaktig test vil det være opp til testeren å bedømme om den skal anses som godkjent		
Kommentar	Siden algoritmen vi bruker er relativt enkel regner vi ikke med å få et veldig godt resultat, men håper å få et som er bra nok. Hvor bra algoritmen fungerer avhenger i stor grad av hvilken type musikk vi spiller av, men for sangen "Mr Scruff – Kalimba" ser algoritmen ut til å ha funnet en ok rytme.		
Test nr	3	Resultat	Godkjent
Dato utført	09.05.13	Utført av	DK

Test case – ID 23			
Story ID	181	Story tittel	Funksjon: Statistikk - Stolpediagram
Hva testes	Vi tester om stolpediagrammet tegnes etter gitte verdier		
Type test	Unit test		
Story beskrivelse	Som utviklere ønsker vi å presentere statistikken i form av et stolpediagram.		
Utførelse	Jeg har laget en randomgenerator som genererer int verdier mellom 0 og 200 når jeg trykker på en knapp i scenen. Deretter setter jeg verdiene i X-vector3'en, og sjekker at stolpene tegnes etter verdiene.		
Kriterier	Det skal være mulig å se at stolpene tegnes riktig etter de genererte verdiene når man trykker på knappen.		
Kommentar	Stolpene tegnes som forventet. Og testen er godkjent.		
Test nr	1	Resultat	Godkjent
Dato utført	20.05.13	Utført av	MJ

Test case – ID 24			
Story ID	211	Story tittel	Implementasjon: Merge statistikk og database
Hva testes	Vi tester om vi får hentet verdier fra databasen inn som statistikk		
Type test	Black-Box		
Story beskrivelse	Vi ønsker å hente data fra databasen og presentere det i stolpediagrammene		
Utførelse	La inn noen verdier i databasen for poengscore, dans og dato. Prøvde så å hente ut de verdiene og presentere det i statistikken.		
Kriterier	Det skal være mulig å se at stolpene tegnes riktig etter de verdiene som er i databasen.		
Kommentar	Stolpene tegnes som forventet etter de gitteverdiene i databasen.		
Test nr	1	Resultat	Godkjent
Dato utført	20.05.13	Utført av	MJ

Test case – ID 25				
Story ID	107	Story tittel	Funksjon: Dynamisk vanskelighetsgrad	
Hva testes	Vi tester den dynamiske vanskelighetsgraden opptrer slik den skal			
Type test	Black-Box			
Story beskrivelse	Som utviklere ønsker vi å lage en dynamisk vanskelighetsgrad som justerer på vanskelighetsgraden etter hvor mange bevegelser brukeren klare å fullføre.			
Utførelse	Begynner først med en høy vanskelighetsgrad, prøver å feile ti bevegelser på rad for å se om vanskelighetsgraden forandret seg. Deretter begynne med en lav vanskelighetsgrad og fullføre minst ni av ti bevegelser for å se om vanskelighetsgraden endret seg			
Kriterier	Vanskelighetsgraden endrer seg slik det er spesifisert, og ikke går utenfor de tre vanskelighetsgradene			
Kommentar	Vanskelighetsgraden endret seg, noe vi så ved både utskrift og tiden brukeren hadde på å utføre en bevegelse endret seg.			
Test nr	1		Resultat	Godkjent
Dato utført	23.05.13		Utført av	ZA, AO

Test case – ID 26				
Story ID	197	Story tittel	Funksjon: Dynamisk vanskelighetsgrad	
Hva testes	Vi tester verdier som settes for de tre vanskelighetsgradene. Verdiene er: Tiden brukeren har på å utføre en bevegelse, poengscore og tiden fra og til en beat en bevegelse blir utført på.			
Type test	White and Black-Box			
Story beskrivelse	Ønsker å endre vanskelighetsgraden ved å gi brukerne lengre tid mellom hvert dansetrinn, og ved å gi dem lengre tid på å fullføre bevegelsene(treffe boksen).			
Utførelse	Setter verdier og tester om disse er gunstige ved å spille gjennom danser.			
Kriterier	Basert på våre erfaringer, sjekker vi om de gitte verdiene er passende for brukerne av LetsDance			
Kommentar	Verdiene vi har valgt føler vi er passende under de tre vanskelighetsgradene Merk: Dette er en subjektiv test			
Test nr	1		Resultat	Godkjent
Dato utført	14.05.13		Utført av	ZA, AO

Test case – ID 27			
Story ID	105, 103, 106, 59, 169, 189, 112, 127, 146, 147,170,171	Story tittel	Krav: Spillet skal oppfatte brukerbevegelser.
Hva testes	Om spillet oppfatter brukerens bevegelser.		
Type test	Blackbox, Systemtest.		
Story beskrivelse			
Utførelse	Vi starter opp spillet for å se om avataren beveger seg etter brukerens bevegelser.		
Kriterier	Avataren må kunne gi en brukbar oppfatning av brukeren for det skal kunne være spillbart.		
Kommentar	Spillet oppfatter brukerens bevegelser, siden Kinect sensoren og programvaren ikke er helt nøyaktig med tanke på dybde hender det at det er noe upressist.		
Test nr	1	Resultat	Godkjent
Dato utført	24.05.13	Utført av	Henrik Olsson, TestAnsvarlig

Test case – ID 28			
Story ID	61,70,85, 94,96,98, 121,123,175,176,179,181,190,209,205	Story tittel	Krav: Følge statistikk
Hva testes	Om spillet fører statistikk over brukerens progresjon og alt rundt.		
Type test	Blackbox, Systemtest		
Story beskrivelse			
Utførelse	Vi fullfører tre danser i tre forskjellige sanger, sjekker om det stemmer overens. Så starter vi spillet på nytt for å se om vi fortsatt har oversikt over progresjonen til brukeren.		
Kriterier	Hver eneste dans skal ha statistikk lagret, dette skal være uavhengig om man går ut av spillet eller ikke.		
Kommentar	Etter tre danser på samme låt (lets dance, david bowie). statistikken ble ført i stolpediagram, dette var fortsatt tilgjengelig etter å ha startet spillet på nytt. Etter å ha danset seks danser til sjekket jeg statistikken på alle sammen. Det stemte, riktignok plukket jeg opp en liten bug da jeg danset. En ny dansebevegelse ble hentet før en annen hadde blitt fjernet.		
Test nr	1	Resultat	Godkjent
Dato utført	24.05.13	Utført av	Henrik Olsson, Test Ansvarlig

Test case – ID 29			
Story ID	111,80,52,51,197,203,107	Story tittel	Krav: Spillet skal tilpasses brukerens behov
Hva testes	Om det er mulighet for å tilpasse spillet etter brukerens behov, da tenker vi på man skal kunne begrense seg til for eksempel begrense seg til bare bruk av armer i spillet.		
Type test	Blackbox, Systemtest		
Story beskrivelse			
Utførelse	Vi starter opp spillet, går så inn i innstillinger for å bestemme hvilke deler av kroppen som er aktiv. Jeg byttet forskjellige kroppsdeler etter flere danser.		
Kriterier	Spillet skal forholde seg til kun de brukerbestemte kroppsdelene.		
Kommentar	Spillet fungerte akkurat som det står i spesifikasjonen.		
Test nr	1	Resultat	Godkjent
Dato utført	24.05.13	Utført av	Henrik Olsson, Test Ansvarlig

Test case – ID 30			
Story ID	180	Story tittel	Funksjon: Prestasjoner - legge til databasestøtte
Hva testes	Tester om metoden for å laste inn prestasjonsprogresjon fra databasen fungerer.		
Type test	Unit test		
Story beskrivelse	Som utvikler ønsker jeg å bygge videre på prestasjonsrammeverket og legge til støtte for lagring i database slik at progresjonen kan lagres mellom sesjoner.		
Utførelse	Jeg lager en lokal liste som jeg ønsker å hente progresjonen fra og sette denne progresjonen inn i listen som er i spillet.		
Kriterier	Progresjons verdiene fra listen skal vises i spillet.		
Kommentar	Verdiene ble hentet frem på korrekt måte og vises i spillet.		
Test nr	1	Resultat	Godkjent
Dato utført	08.05.13	Utført av	DK

Test case – ID 31			
Story ID	180	Story tittel	Funksjon: Prestasjoner - legge til databasestøtte
Hva testes	Tester om metoden for å lagre prestasjonsprogresjon fra databasen fungerer.		
Type test	Unit test		
Story beskrivelse	Som utvikler ønsker jeg å bygge videre på prestasjonsrammeverket og legge til støtte for lagring i database slik at progresjonen kan lagres mellom sesjoner.		
Utførelse	Jeg lager en kaller på løkken som skal lagre verdiene, og lager debug statements so skriver ut innholdet i listen som genereres.		
Kriterier	Progresjons fra spillet skal printes ut i debug listen.		
Kommentar	Verdiene ble riktig printet, jeg kan nå bytte ut debug statementen med et kall som sender listen til databaseskriptet for lagring.		
Test nr	1	Resultat	Godkjent
Dato utført	08.05.13	Utført av	DK

Test case – ID 32			
Story ID	180	Story tittel	Funksjon: Prestasjoner - legge til databasestøtte
Hva testes	Tester om metoden for å laste inn prestasjonsprogresjon fra databasen fungerer.		
Type test	Blackbox testing		
Story beskrivelse	Som utvikler ønsker jeg å bygge videre på prestasjonsrammeverket og legge til støtte for lagring i database slik at progresjonen kan lagres mellom sesjoner.		
Utførelse	Jeg lager manuelt progresjonsdata i databasen som jeg ønsker å få hentet ut, og kaller på metoden for å laste inn data.		
Kriterier	Progresjons verdiene fra databasen skal vises i spillet.		
Kommentar	Verdiene ble hentet frem på korrekt måte og vises i spillet.		
Test nr	1	Resultat	Godkjent
Dato utført	08.05.13	Utført av	DK

Test case – ID 33				
Story ID	180	Story tittel	Funksjon: Prestasjoner - legge til databasestøtte	
Hva testes	Tester om metoden for å lagre prestasjonsprogresjon fra databasen fungerer.			
Type test	blackbox test			
Story beskrivelse	Som utvikler ønsker jeg å bygge videre på prestasjonsrammeverket og legge til støtte for lagring i database slik at progresjonen kan lagres mellom sesjoner.			
Utførelse	Jeg kaller på metoden som lagrer progresjonen i en liste, som igjen kaller på metoden i databasescriptet for å få lagret denne listen.			
Kriterier	Progresjonen fra spillsesjonen skal være opprettet i databasen, jeg sjekker dette via tredjepartsverktøyet "SQLite manager" for å få direkte innsyn i database filen.			
Kommentar	Verdiene ble opprettet i databasen			
Test nr	1		Resultat	Godkjent
Dato utført	08.05.13		Utført av	DK

Test case – ID 34				
Story ID	54,56,53, 84,85,87, 113,114,1 16,117,13 3,134,135 ,136,137, 204	Story tittel	Krav: Svært enkelt brukergrensesnitt	
Hva testes	I hviken grad brukergrensesnittet er enkelt			
Type test	Brukertest, Systemtest			
Story beskrivelse				
Utførelse	Spillet blir startet opp, vi tester så ut hvor mange taste trykk det tar før vi danser.			
Kriterier	Det skal ta svært kort tid før man begynner å danse. enkle menyer osv. Riktignok er det vanskelig å sette noe tall for dette.			
Kommentar	Spill menyen var enkel og intuitiv, jeg brukte 6 museklikk for å komme i gang med å danse. Jeg føler at alle menyen jeg gikk igjennom var hensiktsmessig plassert. Store knapper og tydelig skrift gjorde brukeropplevelsen enkel. Dette ble utført av en person med teknisk innsikt så brukeropplevelsen kan variere fra person til person.			
Test nr	1		Resultat	GodKjent
Dato utført	25.05.13		Utført av	Henrik Olsson, Test Ansvarlig

Test case – ID 35			
Story ID	66,81,107 ,197	Story tittel	Krav: Justerbar vanskelighetsgrad
Hva testes	Er det mulig å justere vanskelighetsgraden? Hvilke følger får det når jeg gjør det.		
Type test	Blackbox, systemtest		
Story beskrivelse			
Utførelse	Starter opp spillet, følger menyen slik at jeg kommer inn i instillinger. I instillingene er det en mulighet til å justere mellom tre nivåer.		
Kriterier	Etter å ha endret vanskelighetsgrad i menyen, så skal nivået i spillet endre seg. Det skal bli vanskeligere på en aller annen måte.		
Kommentar	Vanskelighetsgraden hadde tre nivåer; lett, normal og vanskelig. Jeg begynte på lett og utførte alle oppgavene, vanskelighetsgraden økte etersom jeg klarte alle dansetrinnene. Dansetrinnfrekvensen økte med vanskelighetsgraden. Etter jeg var ferdig med dansen gikk jeg tilbake og sjekket hvilken vanskelighetsgrad jeg var på. Nå hadde det blitt endret til vanskelig, dette samsvarer med designdokumentasjonen.		
Test nr	1	Resultat	Godkjent
Dato utført	25.05.13	Utført av	Henrik Olsson, Testansvarlig

Test case – ID 36			
Story ID	55,121	Story tittel	Krav: Hver bruker skal kunne ha sin egen brukerkonto.
Hva testes	Om mine egne data på min brukerkonto vil være tilgjengelig for bare meg, om det er lett for andre å hente ut mine data, og om mine data blir lagret etter jeg er ferdig med å spille.		
Type test	Blackbox, systemtest		
Story beskrivelse			
Utførelse	Etter å ha startet opp spillet, følger jeg instruksene og oppretter en bruker. Med brukernavn og passord, etter det får jeg logget inn på min bruker. Jeg velger så å gjennomføre en dans for å få opp min statistikk, logger så ut og inn for å se om statistikken fortsatt er tilstede. Samtidig prøver jeg å logge inn på brukeren min med feil passord.		
Kriterier	Jeg skal ikke få tilgang til en annen konto, statistikken og andre valg skal være synlig etter å ha logget ut.		
Kommentar	Jeg måtte taste inn riktig passord for å få tilgang til min bruker, min statistikk var også lagret etter å ha avsluttet spillet og logget ut. Slik som spillet er i Unity uten å ha blitt gjort om til en enkelt exe fil så kan man hoppe inn i scener uten å logge inn, selvom dette er mulig så vil ingen brukerdata bli eksponert da vi vil få nullpointer exception.		
Test nr	1	Resultat	Godkjent
Dato utført	25.05.13	Utført av	Henrik Olsson, Testansvarlig

Test case – ID 37			
Story ID	92,82,172 ,173,191, 192	Story tittel	Krav: Legge til egen musikk
Hva testes	Om det er mulig å legge til egen musikk i spillet for så å kunne danse i takt med dette.		
Type test	Blackbox, Systemtest		
Story beskrivelse			
Utførelse	Vi starter opp spillet, og går frem til musikk menyen. Jeg velger så note knappen for å hente inn egen musikk, da får jeg opp file exploreren, jeg velger å legge til en kortversjon av Lets Dance - David Bowie. Så starter jeg dansen for å se om jeg kan danse i takt til denne låta.		
Kriterier	Jeg skal kunne være istand til å legge til min egen musikk, for så å kunne danse til dette.		
Kommentar	Filene må være .ogg format for å kunne bli importert i spillet. dette er pga lisensiering. .Ogg er et åpent format.		
Test nr	1	Resultat	Godkjent
Dato utført	25.05.13	Utført av	Henrik Olsson, Testansvarlig

Test case – ID 38			
Story ID	49,62,83, 108,175,1 78,180	Story tittel	Krav: Prestasjoner
Hva testes	Om brukeren får belønninger ettersom bestemte oppgaver blir gjennomført. Og om det er mulig å se de belønningene jeg har fått i ettertid.		
Type test	Blackbox, Systemtest		
Story beskrivelse			
Utførelse	Jeg spiller gjennom spillet på vanlig måte for å se om får belønninger. Samtidig sjekker jeg om jeg kan gå å se tilbake på mine prestasjoner		
Kriterier	Jeg må få minst en belønning for at testen skal bli godkjent.		
Kommentar	Jeg fikk en achievement allerede når jeg hadde lagd en bruker, så kan man i hovedmenyen mulighet til å gå inn i listen over prestasjoner/achiements.		
Test nr	1	Resultat	Godkjent
Dato utført	25.05.13	Utført av	Henrik Olsson, Test Ansvarlig

Test case – ID 39			
Story ID	104,200,206,111,199,95,101,196	Story tittel	Krav: Autogenerere danser til egen musikk
Hva testes	Om dansen blir autogenerert etter egen musikk		
Type test	Blackbox, Systemtest		
Story beskrivelse			
Utførelse	Etter å ha hentet inn egen musikk for å danse til, så starter jeg opp en dans. Deretter ser jeg om spillet beveger seg i takt med musikken.		
Kriterier	Det skal være mulig å se visuelt at spillet går i takt med musikken.		
Kommentar	Høytalerene viser at spillet er i takt med musikken. Jeg får også bra poeng om jeg treffer de bestemte punktene i takt med musikken.		
Test nr	1	Resultat	Godkjent
Dato utført	25.05.13	Utført av	Henrik Olsson, Testansvarlig

Test case – ID 40			
Story ID	152	Story tittel	Krav: Mulighet til å designe egen avatar
Hva testes	Om det er mulig å egendefinere avataren som blir brukt i danserommet		
Type test	Blackbox, Systemtest		
Story beskrivelse	Brukere skal ha muligheten til å identifisere seg med avataren		
Utførelse	Etter å ha logget inn velger jeg instillinger for å kunne bestemme min egen avatar, deretter gå inn i danserommet for å benytte meg av denne.		
Kriterier	Jeg skal ha muligheten til å bestemme utseendet på avataren i danserommet		
Kommentar	Dette er ikke ferdig implementert, dette er også et lavt prioritert krav.		
Test nr	1	Resultat	Feilet
Dato utført	25.05.13	Utført av	Henrik Olsson, Testansvarlig

Test case – ID 41				
Story ID	153	Story tittel	Krav: mulighet for manuelt å designe danser	
Hva testes	Om det er mulig å lage en fullstendig egendefinert dans.			
Type test	Blackbox, Systemtest			
Story beskrivelse	Muligheter for manuelt design av danser.			
Utførelse	Etter å ha logget inn velger jeg å gå inn i verktøyet for å designe egne danser.			
Kriterier	Det skal være mulighet for å designe en egen fullstendig dans.			
Kommentar	Dette har ikke blitt implementert i denne versjonen av spillet.			
Test nr	1		Resultat	Feilet
Dato utført	25.05.13		Utført av	Henrik Olsson, Testansvarlig

Test case – ID 42				
Story ID	154	Story tittel	Automatisk deteksjon av funksjonsnivå	
Hva testes	Om det er mulig å oppdage funksjonsnivået automatisk.			
Type test	Blackbox, Systemtest			
Story beskrivelse				
Utførelse	Etter å ha logget inn prøver jeg å finne verktøyet for automatisk deteksjon av funksjonsnivå.			
Kriterier	Spillet må ha mulighet til å gi en oppfatning av hvilke deler av kroppen jeg er i stand til å bruke.			
Kommentar	Det er ikke implementert et verktøy for dette enda.			
Test nr	1		Resultat	Feilet
Dato utført	25.05.13		Utført av	Henrik Olsson, Testansvarlig

Test case – ID 43				
Story ID	155	Story tittel	Muligheter for flerspiller	
Hva testes	Om det er mulig å være to samtidig i danserommet.			
Type test	Blackbox, Systemtest			
Story beskrivelse	Flerspiller modus			
Utførelse	Starte opp dansespillet i flerspiller modus for å se om det fungerer.			
Kriterier	Dansespillet skal ha en viss underholdningsverdi for to mennesker i danserommet(oppfattet av kinectsensoren) samtidig.			
Kommentar	Dette er ikke implementert			
Test nr	1		Resultat	Feilet
Dato utført	25.05.13		Utført av	Henrik Olsson, Testansvarlig

Test case – ID 44				
Story ID	56	Story tittel	Brukergrensesnitt	
Hva testes	Vi sjekker om brukergrensesnittet er enkelt å bruke og komme i gang med spillet.			
Type test	Blackbox			
Story beskrivelse	Som en bruker ønsker jeg et brukergrensesnitt for å enkelt komme i gang.			
Utførelse	Vi går gjennom spillet og går utifra de tekniske delene som er der for eksempel er det mye rot i menyene, står ting tydelig osv.			
Kriterier	Menysystemet skal være lett å forstå og enkelt å bruke.			
Kommentar	Vi gikk gjennom alle scenene i spillet og menyen har ikke noe overflødige knapper, alle fører til bestemte og viktige funksjoner. Knappene er store og tydelige slik at det skal være enkelt å navigere seg gjennom menyen.			
Test nr	1		Resultat	Godkjent
Dato utført	24.05.2013		Utført av	Are Oven

Test case – ID 45			
Story ID	54	Story tittel	Quick entry
Hva testes	Her sjekkes antall scenebytter som skal til før man er i gang med spillet		
Type test	blackbox		
Story beskrivelse	Som en bruker ønsker jeg å komme fort i gang så det ikke blir kjedelig.		
Utførelse	Vi går gjennom spillet fra man logger inn til man er inne å spiller og teller antall ting man må gjøre.		
Kriterier	for at vi kan regne spille vårt som enkelt og at man kommer raskt i gang må vi være inne å spille etter te sceneskift		
Kommentar	Denne regnes som godkjent, du må gjennom tre ting før du er inne å spiller, hvis du har bruker må du: logge in, velge at man skal spille og velge sang.		
Test nr	1	Resultat	Godkjent
Dato utført	24.05.2013	Utført av	Are Oven

Test case – ID 46			
Story ID	189	Story tittel	Dans - Grid - Dybde
Hva testes	Vi sjekker om vi har klart å lage en 3d-grid		
Type test	whitebox		
Story beskrivelse	Som en utviker ønsker vi å ha dybde i griden slik at vi kan legge til bevegelser i 3 dimensjoner.		
Utførelse	Vi starter spillet og sjekker om alle cellene er på plass, har rett navn og om de virker slik de skal.		
Kriterier	For at testen skal være godkjent må alle cellene være på plass, ha rett navn og virke korrekt.		
Kommentar	3D-gridden kjører korrekt, cellene er der de skal og gjør det de skal.		
Test nr	1	Resultat	Godkjent
Dato utført	16.04.2013	Utført av	Are Oven

Test case – ID 47			
Story ID	111	Story tittel	Funksjon: Generator - Tilpasse dans
Hva testes	Det testes om generatoren holder seg innenfor størrelsen på stringen		
Type test	Whitebox		
Story beskrivelse	Som utvikler ønsker jeg å kunne tilpasse dansen ut ifra hvilke utfordringer spilleren har, slik at det passer for en større brukergruppe		
Utførelse	Blackbox		
Kriterier	Generatoren må generere bevegelser uten å be om verdier utenfor string, da dette vil føre til en feil		
Kommentar	På grunn av funksjonen i generatoren som skal gjøre at det komme lignende bevegelser etter hverandre ville dette ført til ganger der generatoren ville havnet utenfor den gitte stingen, men dette klarte vi å unngå. Testen blir godkjent da generatoren ikke prøve å hente verdier på utsiden av stringen		
Test nr	2	Resultat	godkjent
Dato utført	21.05.2013	Utført av	Are Oven

Test case – ID 48			
Story ID	111	Story tittel	Funksjon: Generator - Tilpasse dans
Hva testes	Det testes om dansegeneratoren er i stand til å tilpasse danser.		
Type test	blackbox		
Story beskrivelse	Som utvikler ønsker jeg å kunne tilpasse dansen ut ifra hvilke utfordringer spilleren har, slik at det passer for en større brukergruppe.		
Utførelse	Velger bestemte kroppsdeler og sjekker om bevegelsene passer til de definerte kroppsdelene.		
Kriterier	For at testen skal regnes som godkjent må kun bevegelser som passer til de definerte kroppsdelene være valgt.		
Kommentar	Dansegeneratoren klarer å generere bevegelser ut ifra hvilke kroppsdeler som er valgt.		
Test nr	1	Resultat	Ukjent
Dato utført	21.05.2013	Utført av	Are Oven

Test case – ID 49			
Story ID	206	Story tittel	Funksjon: Dansegenerator - Opprette bevegelsene
Hva testes	Blir bevegelsene tolket korrekt		
Type test	whitebox		
Story beskrivelse	Som en utvikler ønsker jeg å opprette bevegelsene slik at det eksisterer bevegelser som kan brukes til dansene.		
Utførelse	Vi sender inn mange bevegelser og sjekker at verdiene som skal komme ut kommer ut.		
Kriterier	For at testen kan regnes som godkjent må verdiene som vi får ut være korrekte.		
Kommentar	Vi får rett verdier tilbake på alle bevegelsene		
Test nr	2	Resultat	Godkjent
Dato utført	22.05.2013	Utført av	Are Oven

Test case – ID 50			
Story ID	206	Story tittel	Funksjon: Dansegenerator - Opprette bevegelsene
Hva testes	Blir bevegelsene laget korrekt		
Type test	blackbox		
Story beskrivelse	Som en utvikler ønsker jeg å opprette bevegelsene slik at det eksisterer bevegelser som kan brukes til dansene.		
Utførelse	Vi sjekker om bevegelsene som er lagret blir korrekte når vi kjører spillet		
Kriterier	For at den kan regnes som fullført må bevegelsene eksistere og fungere korrekt		
Kommentar	Bevegelsene blir laget korrekt og må utføres slik det var tenkt.		
Test nr	1	Resultat	Godkjent
Dato utført	20.05.2013	Utført av	Are Oven

Test case – ID 51			
Story ID	191	Story tittel	Funksjon: Musikk mappe
Hva testes	Musikk fil flyttes til korrekt mappe og listen med sanger in game blir oppdatert		
Type test	Blackbox		
Story beskrivelse	Som bruker ønsker jeg muligheten til å legge egen musikk i en mappe på pcen slik at jeg enkelt kan hente fram egen musikk i spillet.		
Utførelse	Testes i runtime. Skal finne en sang på PCen ved hjelp av file exploreren og sjekke om den blir kopiert til riktig mappe når man klikker velg. Denne sangen skal også legges til i listen med sanger som brukeren kan velge av in game.		
Kriterier	Filen blir flyttet, file-extensions blir klippet vekk og sangen blir lagt til i listen med sanger.		
Kommentar	Fileexploreren åpnet seg og valg sang ble kopiert til riktigmappe og hadde riktig navn uten file-extension. Sangen ble korrekt lagt til i listen over sanger, slik at brukeren kan velge den.		
Test nr	1	Resultat	Godkjent
Dato utført	13.05.2013	Utført av	JEH

Test case – ID 52			
Story ID	203	Story tittel	Funksjon: Meny – Tilpasse scene
Hva testes	Brukeren sin ønsket tilpasning		
Type test	Blackbox		
Story beskrivelse	Som en utvikler ønsker jeg å lage en scene der brukeren kan tilpasse hvilke deler av skjermen som skal brukes.		
Utførelse	Testes i runtime ved at vi krysser av på forskjellige valg av tilpasning og går frem og tilbake mellom scenene for å se om valgene blir bevart.		
Kriterier	Verdiene blir tatt vare på og checkboxene er riktigavkrysset etter dem.		
Kommentar	Verdiene blir tatt vare på mellom scenene og checkboxene blir korrekt avhaket når man går tilbake til tilpasning scenen		
Test nr	1	Resultat	Godkjent
Dato utført	13.05.13	Utført av	JEH

Test case – ID 53			
Story ID	202	Story tittel	Funksjon: Meny – Startup scene
Hva testes	Funksjonalitet av ikonene		
Type test	Blackbox		
Story beskrivelse	Som en utvikler ønsker jeg å lage en startupscreen til spillet		
Utførelse	Tester om ikonene faktisk fører til riktig sted ved å klikke på de. • Logginn ikonet skal føre brukeren til logg inn scenen • Add User ikonet skal før brukeren til registrerings scenen		
Kriterier	Metro ikonene fører brukeren til riktig scene		
Kommentar	Begge ikonene gjør det de skal og fører brukeren til riktig scene		
Test nr	1	Resultat	Godkjent
Dato utført	13.05.13	Utført av	JEH

<table>
<tr><th colspan="5">Test case – ID 54</th></tr>
<tr><td>Story ID</td><td></td><td>Story tittel</td><td colspan="2"></td></tr>
<tr><td>Hva testes</td><td colspan="4">Flyten i menyen, alle knapper og input.</td></tr>
<tr><td>Type test</td><td colspan="4">Systemtest</td></tr>
<tr><td>Story beskrivelse</td><td colspan="4"></td></tr>
<tr><td>Utførelse</td><td colspan="4">Tester alle knapper i menyen og ser om de fører til riktig sted ut ifra hva som er dokumentert i storyboardet over menyen. Vi tester også om funksjonene på knappene gjør det de skal.</td></tr>
<tr><td>Kriterier</td><td colspan="4">Alle ikonene/knapper skal føre til riktig sted.</td></tr>
<tr><td>Kommentar</td><td colspan="4">
StartupScreen - Alle knapper fører til riktig sted
LoggInn – All input og knapper gjør som de skal, logg inn knappen logger inn brukeren.
RegistrerUser – All input og knapper gjør som de skal, registrerbruker får registrert brukeren, slik at brukeren får logget inn.
Setings – Alle knappene fører til riktig sted, utenom avatar knappen som ikke er implementert i vår prototype
UserMenu – Alle knappene i hovedmenyen fører til riktig sted. Log ut knappen lagrer all brukerdata og nullstiller alle variabler.
SongSelect – Alle knappene fører til riktig sted, sangnavn blir registrert når en sang er klikket og play sender brukeren til danserommet.
StatistikkSelect – Tilbakeknappen fører dit den skal, valget av sang fungerer. Se statistikk-knappen fører til statistikk scenen.
ChangePassword – All input og knapper gjør som de skal. Passordet blir byttet når bytt passord er trykket.
Vanskelighetsgrad – Alle knappene gjør som de skal og lagrer ønsket vanskelighetsgrad.
Prestasjoner – Tilbakeknappen fører tilbake til hovedmenyen
Stolpediagram – Knappene fører dit de skal.
Tilpasse – Alle avkrysningsboksene fungerer, tilbakeknappen gjør det den skal. Valgt tilpasning lagres.
</td></tr>
<tr><td>Test nr</td><td>1</td><td>Resultat</td><td colspan="2">Godkjent</td></tr>
<tr><td>Dato utført</td><td>24.05.13</td><td>Utført av</td><td colspan="2">JEH</td></tr>
</table>

Brukermanual

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokument	23.05.2013	MJ
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	HO, AO

Innholdsfortegnelse

1. Innledning

Dette er en brukermanual for spillet LetsDance. I denne manualen får du en rask gjennomgang av spillet, med valgene man kan ta og hvordan man kommer i gang.

2. Brukermanual

2.1 Logge inn/ Ny bruker

Når man har startet spillet er dette skjermbildet som møter deg. Her har man to valg. Hvis man allerede har laget seg en bruker, kan man trykke på hengelåsen for å logge inn. Hvis det er første gangen man spiller, kan man trykke på avataren med pluss, og dette bildet kommer opp:

Her velger man et **navn**, navnet brukes ikke til noe annet enn at administratorer kan se hvem brukeren tilhører. **Brukernavn** er navnet man bruker når man skal logge inn, og **passord** er som kjent koden man bruker for å logge inn. Man kan bruke både store og små bokstaver i alle feltene.

2.2 Hovedmeny

Når man har laget en ny brukerkonto(I dette tilfellet kalt MarcusJ) og logget inn, kommer man inn på hovedmenyen:

Dette er hovedmenyen, hvis man trykker på knappen oppe til venstre kommer man igang med spillet, man kommer først til en meny hvor man kan velge sang:

Her kan man velge en av sangene i listen, hvis man vil danse til en annen sang er det ikke verre enn at man legger til sangen man vil danse til i mappen ”sanger” og du trykker på note-ikonet, dermed vil man få opp filutforskeren. Når man har valgt sang og trykket play kommer man ut i danserommet, man trykker go! for å sette i gang:

Oppe til venstre ser man instruktøren som viser hvordan bevegelsene utføres. Ønsker man å avslutte eller pause før dansen er ferdig kan man enkelt trykke på enten ”Abort” eller ”Pause” til venstre på skjermen. Når man har danset ferdig blir man sendt rett til oversikten over statistikken for den sangen:

2.4 Statistikk

Her ser man statistikken til dansen (dette er bare et illustrasjonsbilde, så poengscore i poeng og dato vises ikke) under søylene står poengscoren i tall, og over søylene står datoen for når dansen ble utført. Ønsker man å sjekke statistikken ved en senere anledning kan man enkelt gå tilbake til hovedmenyen og trykke på denne knappen:

Etter det kan man velge hvilken sang man skal se statistikk for, også kommer man inn på samme bilde som vist ovenfor.

2.3 Prestasjoner

I LetsDance har man noe som heter prestasjoner. Dette er små belønninger man får underveis når man utfører oppgaver. Hvis man ønsker å se hvilke prestasjoner man har utført trykker man på denne knappen i hovedmenyen.

Da blir man sendt videre til en oversikt over de prestasjonene man har utført til nå:

Her ser man at den innloggede brukeren har utført to prestasjoner. ”Ny bruker” og ”Logg inn”. Etterhvert som man spiller vil man se at lister fyller seg opp ytterligere.

2.5 Instillinger

I LetsDance har man forskjellige innstillinger man kan endre på. Hvis man trykker på denne knappen i hovedmenyen:

Kommer man inn på innstillingsmenyen:

Oppe til venstre har man et ikon for avatartilpasning (dette er ikke implementert i prototypen vår).

Oppe til høyre ser man et ikon for skjermtilpasning. Trykker man på den kan man tilpasse hvilke deler av skjermen man ønsker å benytte for dansen:

2.6 Tilpasning

Hvis man for eksempel fjerner krysset på venstre nede og høyre nede, vil man ikke få bevegelser som omfatter beina når man danser. Hvis man for eksempel sitter i rullestol er dette aktuelt.

2.7 Vanskelighetsgrad

Går man tilbake til innstillingsmenyen og trykker på denne knappen:

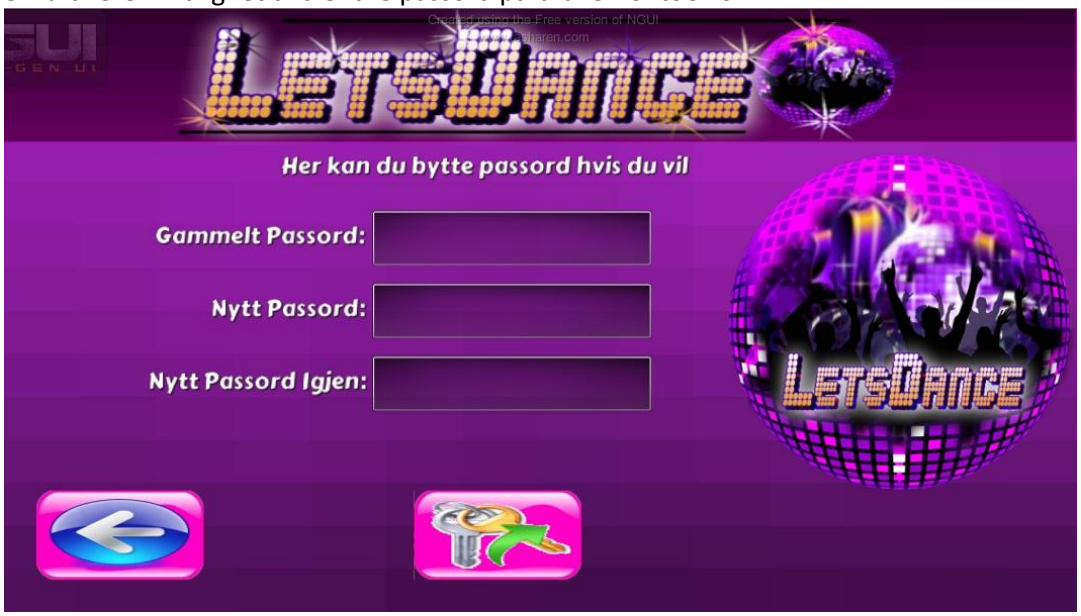
Her kan man endre vanskelighetsgrad på dansen. Vanskelighetsgraden omfatter hvor lang tid man har på seg til å utføre bevegelsen. Vanskelighetsgraden er også dynamisk, det vil si at man har begrenset tid på å utføre en bevegelse, bruker man kort eller lang tid endres vanskelighetsgraden deretter.

Man kan velge mellom tre vanskelighetsgrader: **Lett, Normal** og **Vanskelig.**

2.8 Endre passord

Den siste knappen for innstillinger:

Gir brukeren mulighet til å endre passord på brukerkontoen sin:

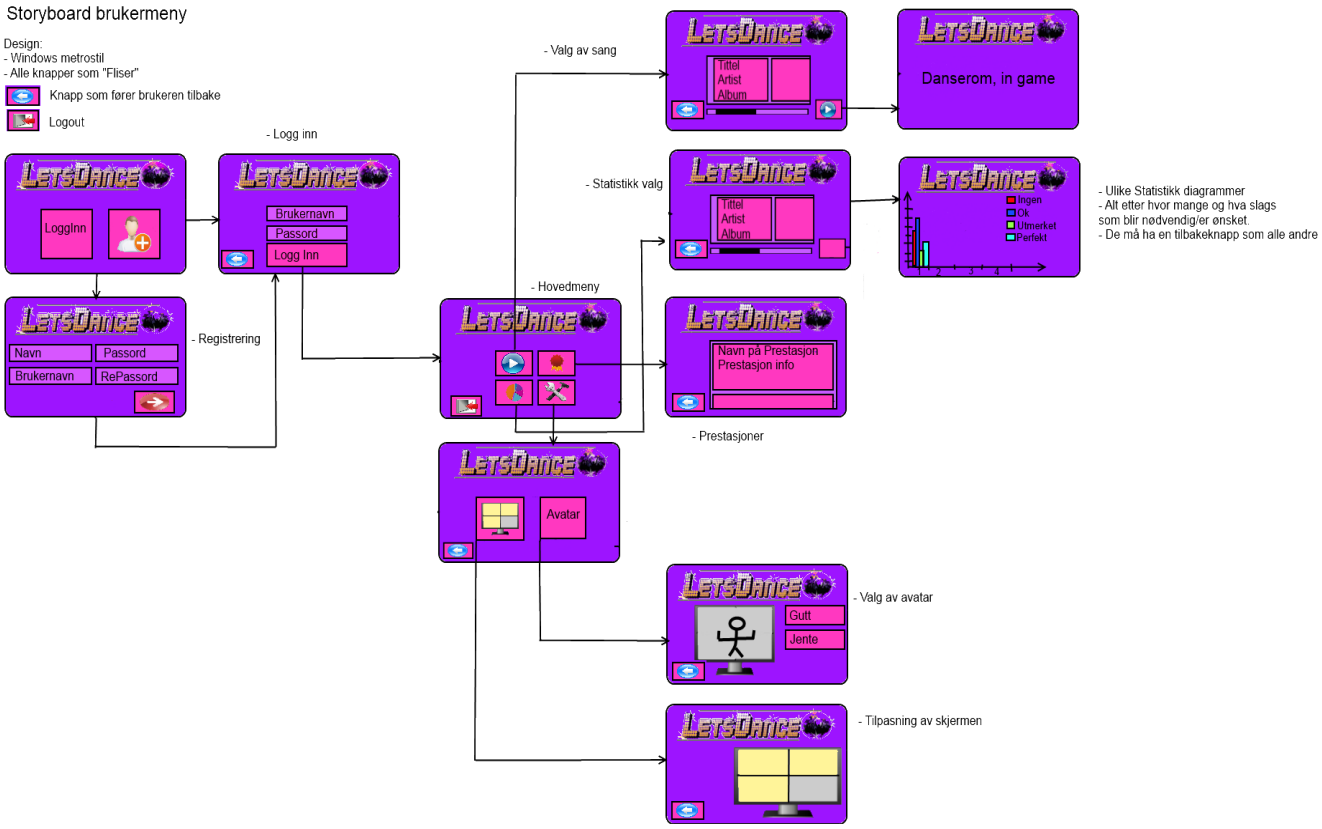


Her kan man fylle inn gammelt passord, og velge seg et nytt. Man trykker på knappen på nede midt på skjermen for å bekrefte.

2.10 Tilbake

Gjennom de fleste scenene vil man finne en knapp som tar brukeren tilbake til forrige scene. Denne knappen ser slik ut:

3 Storyboard



Videre utvikling

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet, Innledning, lets dance, videre utvikling	19.05.13	HO
2.0	Skrevet om vanskelighetsgrad	24.05.13	ZA
3.0	Skrevet om musikkanalyse	24.05.13	DK
4.0	Skrevet om statistikk, instruktør og grafikk	23.05.13	MJ
5.0			
6.0			

Kontrollert av:	ZA, MJ

Innholdsfortegnelse

1. Innledning

LetsDance-prototypen ble utviklet av et team studenter innenfor en begrenset tidsperiode, likevel er dette et spennende konsept som fortjener fokus. Derfor har dette dokumentet som hensikt å legge frem mulighetene for konseptet har ved en kommersiell satsning.

2. LetsDance

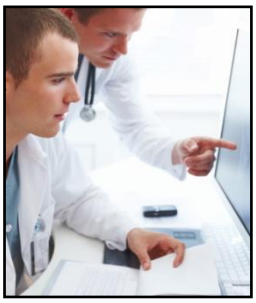
LetsDance er et dansespill utviklet for Sunnaas Sykehus i samarbeid med Mektron, dette prosjektet har som hovedfokus å forenkle treningen til pasientene ved Sunnaas. Dette er pasienter som sliter med nedsatte motoriske evner som følge av ulykker eller sykdom, under behandling jobber de med å trene opp bevegelsesevnen. Tradisjonell behandling kan ofte være kjedelig og trist, derfor skal LetsDance være en alternativ og underholdende rehabilitering.

Ved bruk av programvare og en Xbox Kinect sensor har vi utviklet et dansespill, det er mye potensiale i dette konseptet som vi ikke kan ta oss tid til å utnytte. Uansett så fanger programvaren opp brukerens bevegelser og gir poeng når en bestemt bevegelse blir gjennomført. Vi vet som erfarne gamere at noe så enkelt som poeng er motiverende for å prøve å oppnå nye resultater. Dette er grunnlaget for at det er en underholdende treningsform. I tillegg har programvaren diverse andre funksjoner som det å føre statistikk over brukerens progresjon og muligheten til å bevege seg i takt med musikk for å gjøre det hele mer underholdende.

3. Videre utvikling

Kinect slik som den er i dag kan bli brukt som et verktøy til å oppfatte store bevegelser på en menneskekropp, vi vet at pasienter som sliter med nedsatt bevegelsesmuligheter ikke nødvendigvis har noen fordeler ved å trene på store bevegelser når problemet kanskje ligger på mer nøyaktig nivå. Da vil man være avhengig av et mer nøyaktig verktøy og programvare for opptrening på finbeveglser. Vi vet at Microsoft utvikler en ny sensor Kinect 2.0 som skal være en oppgradering med mer nøyaktig evne til å fange opp brukernes bevegelser. Vi kan da ta i betraktning at utviklere vil skape mer nøyaktig programvare skreddersydd for denne nye teknologien. Da kan vi også håpe at mulighetene ligger mer tilrette for å utvike lignende rehabiliteringssystemer, men med fokus på nøyaktighet og rette bevegelser som er tilpasset for den spesifikke brukeren.

Samfunnet kan ha behov for et full automatisk system som vil være i stand til å avlaste terapeutene hos helseinstitusjonene, hvor brukeren er i stand til å følge interaktive instruksjoner gitt fra en PC skjerm, lerret eller liknende. La oss si at dette var et masseprodusert verktøy som brukere hadde muligheten til å ta med seg hjem, hvor senere en observatør kunne følge brukerens progresjon fra et annet en annen lokasjon.



En annen utfordring ved å automatisere dette er at man vil ha et behov for å kunne gjøre en automatisk gjenkjenning av funksjonsnivå. Med ny teknologi og smarte hoder innenfor det kombinerte fagfeltet medisin, helse og teknologi vil det være gode muligheter for å utvikle algoritmer og teknikker som gjør det mulig for å gjenkjenne funksjonsnivået til en bruker. Deretter hadde det vært ønskelig med en mulighet for å opparbeide vanskelighetsgraden, slik at oppgavene ble mer krevende på områdene; Tempo, funksjonsnivå, frekvens og rytme.

Under prosjektperioden valgte vi å lage et dansespill som kombinerte bruk av flere deler av kroppen i samspill. Vi vet at samspillet mellom forskjellige kroppsdeler er en viktig del av treningen, for å kunne trene opp grunnmotorikken. For eksempel hvis man skal gjøre en enkel oppgave i hverdagslivet som å plukke opp gjenstander, og la oss si at man har nedsatt funksjonsevne i en av hendene. Da vil man ofte velge den bevegelsen som er lettest. Derfor ønsker man å trene opp motorikken i diverse kroppsdeler i samspill med andre kroppsdeler. Med en mer nøyaktig sensor kan man trene opp rytmikk og samspill mellom begge hendene ved hjelp av instruksjoner på en PC skjerm og for eksempel musikk som går i takt med de bestemte oppgavene.

Et fremtidig produkt vil også ha behov for å kunne konfigurere nye dansebevegelser som fagfolk mener er viktig for progresjonen. For å gjøre denne konfigurasjonen uten å ha kunnskap om det som skjer i kildekoden, vil produktet være avhengig av et brukergrensesnitt hvor man kan definere dansebevegelser. Dette kan da løses ved hjelp av at en person som vil være en form for instruktør som utfører den ønskede dansebevegelsen fremfor sensoren, programvaren vil da holde oversikt over rekkefølgen og hvilke deler av griden som er aktiv under "innspillingen" av en ny bevegelse. Dette vil så bli lagret i en database, hvor man senere kan hente ut disse bevegelsene til forskjellige brukere[1].

3.1 Animasjon

Avataren i LetsDance gjenspeiler brukeren i virkeligheten, da Kinecten og programvaren ikke er nøyaktig vil ikke animasjonen gjengi ett skikkelig bilde av brukeren. Fra kommersielle spill hvor man er veldig opptatt av de visuelle elementene i spillet, har man skreddersydd programvaren for å få det "pent". Da vi har valgt å gjengi all input som blir hentet fra Kinecten, så har man i kommersielle spill laget skinnebasert opplegg hvor bevegelsene er på forhånd animert og guider avataren etter bevegelsene som Kinecten plukker opp.

3.2 Musikk

I LetsDance var musikk et viktig krav, for å få et mer personlig forhold til spillet ble ønsket å kunne hente inn egendefinert musikk.

Som vi nevnte i sluttrapporten vår tror vi at å ta utgangspunkt i BeatRoot[2] vil være en god start for en eventuell videreutvikling av denne løsningen.

BeatRoot er et interaktivt rytmedeteksjon og visualiseringssystem, prosjektet har åpen kildekode og er skrevet i Java. Prosjektet er satt i gang på initiativ fra Dr. Simon Dixon som jobber ved Queen Mary universitetet i London. Dixon har skrevet mye detaljert dokumentasjon som beskriver de algoritmene programmet bruker[3].

Forøvrig sies det at selv de beste rytmegjenkjenningsprogrammene på markedet i dag har et stykke å gå før de er perfekte[4], og det vil derfor neppe være gunstig å bruke i et kommersielt produkt. Selv om man finner en god algoritme å bruke for rytmegjenkjenning vil det være begrenset hvor gode dansene man greier å generere er, og vi tror derfor at i en eventuell videreutvikling bør man utforske andre alternativer. Vi mener fremdeles at muligheten for støtte med egen musikk vil være svært gunstig, men at det bør legges opp til dette på en annen måte, for eksempel som en danceEditor i form av et eget vedleggsprogram hvor man selv analyserer sanger og designer danser. Egne editor programmer er brukt i mange spill, spesielt i strategispill er dette populært slik at man kan designe

egne nivåer. Dette kan også være med på å øke aktiviteten i nettsamfunn dedikert til spillet hvor brukerne selv kan hjelpe hverandre og dele sine danser.

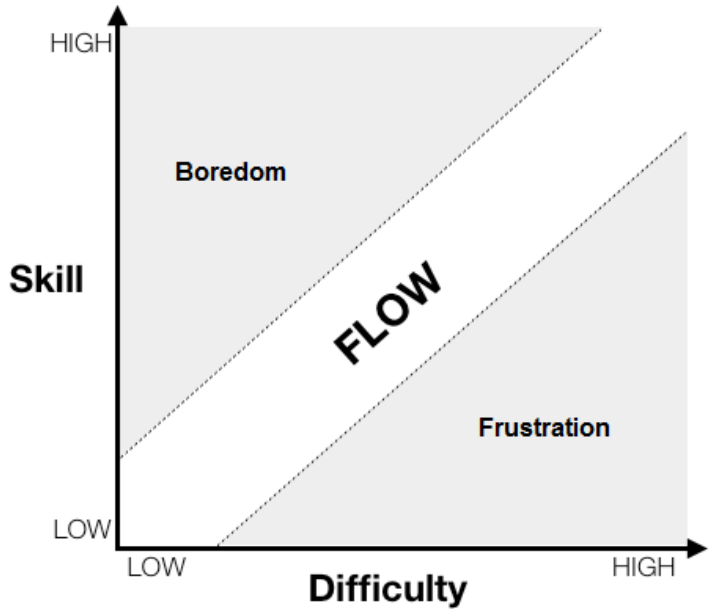
3.3 Vanskelighetsgrad

Vanskelighetsgraden i spillet har et stort videreutviklings potensial. Det finnes mange ulike egenskaper ved spillet som gjør det mulig å definere vanskelighetsgraden på flere måter. I LetsDance finnes det en statisk og en dynamisk vanskelighetsgrad. Den statiske gir brukeren mulighet til å velge mellom tre nivåer av vanskelighetsgrad i menyen før dansen begynner. Den dynamiske justerer på disse i sanntid etter hvordan brukeren presterer.

Noen aspekter ved spillet som kan justeres og bli brukt som en del av vanskelighetsgrad:

- Tiden spilleren har på å utføre en bevegelse
- Kompleksiteten til en dansebevegelse
- Hvor nærme en beat dansebevegelsen blir utført på
- Størrelsen på området som det skal danses etter
- Danse med eller uten instruktør
- Antallet ulike bevegelser i en dans

I en videreutvikling kan det jobbes med å skape en balansert vanskelighetsgrad som er tilpasset brukeren. Dette er en stor utfordring siden en tilpasset vanskelighetsgrad for en spesifikk bruker kan være opp mot antallet brukere av spillet siden hver spiller har unike behov. Det er her dynamisk vanskelighetsgrad kan brukes som et middel for å tilpasse brukeren et passende nivå etter hvordan han eller hun presterer. En utfordring med vanskelighetsgrad er å få det vel balansert og følge en jevn kurve i vanskelighetsgraden slik figuren under viser.



Blir vanskelighetsgraden for enkel for brukeren, kan spillet fort bli kjedelig. På den andre skalaen, dersom vanskelighetsgraden for høy slik at brukeren bommer på mange bevegelser kan det lede til frustrasjon. LetsDance målgruppe er spesielt sårbar mot sistnevnte, derfor er det viktig å passe på at brukeren ikke blir frustrert og demotivert av å spille LetsDance[5].

3.4 Dynamisk vanskelighetsgrad

Målet med dynamisk vanskelighetsgrad er å tilpasse vanskelighetsgraden etter brukerens behov og nivå. På den måten kan brukeren bli engasjert og ha det morsomt over lengre tid over å spille LetsDance. Tradisjonell rehabilitering er ofte kjedelige. Det ønsker vi ikke at spillet skal bli, derfor vil en dynamisk vanskelighetsgrad som tilpasses etter brukerens prestasjoner være et godt verktøy for å hindre dette.

Den dynamiske vanskelighetsgraden som er implementert i LetsDance sjekker hvordan brukeren har gjort det hver tiende bevegelse. Dersom seks av ti, eller færre bevegelser har blitt utført, blir vanskelighetsgraden justert ned. Dersom ni eller ti av de ti siste bevegelsene har blitt utført, justeres vanskelighetsgraden opp. I en videreutvikling kan det også sjekkes etter hvilke bevegelser som brukeren sliter med, og gjøre denne enklere for brukeren å gjennomføre. En årsak til at brukeren kan slite med en bevegelse kan være nedsatt funksjonalitet i en eller flere kroppsdeler som er involvert i den dansebevegelsen. Kanskje når ikke brukeren fram til de områdene som skal treffes. En måte å løse det på er å flytte disse nærmere avataren slik at det blir mulig for brukeren å gjennomføre den bevegelsen.

4. Statistikk

4.1 Prototypen

I prototypen er statistikk presentert i form av stolpediagrammer. Vi har gitt brukerne muligheten til å se poengscore fra de ti siste dansene sine. Poengscoren går fra 0 til 200, og man kan se hvilken dato dansen er utført.

4.2 Videreutvikling

Det finnes utrolig mange muligheter å presentere statistikk på. Man kan presentere poengscore alene med mange forskjellige typer diagrammer (linjediagram, kakediagram, etc.).

I videre utvikling kan man også legge inn flere ting å presentere. For eksempel antall treff per dans, antall bom per dans, hvilken sang man danser best til, treff/bom totalt osv. Det kan være nyttig for ergoterapeutene å få oversikt over hvilke kroppsdeler eller bevegelser man er best/dårligst på, for å kunne hjelpe pasientene med mest effektiv opptrening av de kroppsdelene de er dårligst med.

5. Instruktør

5.1 Prototypen

Slik instruktøren er satt nå er det et enkelt bilde av en ”pinne-figur” som viser hvordan bevegelsene skal utføres. Bilde popper opp i høyre hjerne for hver bevegelse mens sangen spilles.

5.2 Videre utvikling

For å få spillet til å se bedre ut kan det være en fordel å tegne instruktøren i 3D. Men det enkleste er ofte det beste, så lenge det er lett å forstå hvordan bevegelsen skal utføres kan instruktøren være relativt enkel. Kan som sagt se litt ”proffere” ut.

Man kan også legge opp instruktøren slik at den sklir inn fra siden og inn på skjermen, også kan neste bevegele skli inn rett bak men være halvveis transparent. Da kan man forberede seg mer på den kommende bevegelsen.

6. Grafikk

Det er mye som kan gjøres med det visuelle utseendet til spillet. Slik prototypen er, består "danserommet" kun av gratismodeller som er av ganske dårlig kvalitet. Avataren er hentet fra unitys asset store, og er relativt dårlig modellert og har dårlige teksturer. Grunnen til at vi brukte den er pga den var lett å koble opp mot kinecten og brukeren.
Det kan være en fordel å få en 3D designer til å lage nye modeller og danserom. Det kan være en mulighet å la brukeren velge hvilke omgivelser han eller hun vil danse i, for å skape mer variasjon.

7. Referanser

1. *Ann Christin Eliasson, PhD Karolinska Institutet & Egil Utheim, Mektron AS*: Møte 28.02.2013, Oslo.
2. Cannam, S.D.C. *BeatRoot*. 24.05.2013]; Available from: http://code.soundsoftware.ac.uk/projects/beatroot.
3. Dixon, S. *BeatRoot related publications*. 2001-2007 24.05.2013]; Available from: http://code.soundsoftware.ac.uk/publications?project_id=beatroot.
4. jsipic@unity3dForums. *Analyze Music-frequency?* 2012; Available from: http://forum.unity3d.com/threads/125046-Analyze-Music-frequency.
5. Baron, S. *Cognitive Flow: The Psychology of Great Game Design*. 2012 [cited 2013 24.05.2013]; Available from: http://www.gamasutra.com/view/feature/166972/cognitive_flow_the_psychology_of_.php?print=1.

Debrief: Sprint 1

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	09.01.13	HO
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA, DK

Innholdsfortegnelse

1. Innledning

I Sprint 1 brukte vi tiden på dokumentasjonen til prosjektet. Her brukte vi mye tid på å bestemme oss for prosjektmodell, utviklingsmiljø og hvilke verktøy vi skal benytte oss av. Vi førte opp alle mulige oppgaver som ble fordelt opp i Scrumverktøyet slik at vi hadde oversikt over hvem som hadde gjort hva, og hvor lang tid vedkommende hadde brukt eller skulle ha brukt på oppgaven. Slik holder vi oversikt over hvem som jobber med hvilke oppgaver.

2. Det som gikk bra

Gruppen jobbet bra med oppgavene som ble tildelt, alle møtte opp på skolen til avtalt tid. Og vi kom alltid til enighet når vi måtte jobbe "overtid". Dokumentene som ble skrevet av de forskjellige i gruppen, ble senere kontrollert av to andre medlemmer av gruppen. Dette førte med seg at budskapet i dokumentene ble godt formidlet. Vi bestemte oss også for standarder slik at punkter i product backloggen (Kravspesifikasjon), test spesifikasjonen, og andre dokumenter blir likt. Dette innebærer da forside, overskrifter, tekststørrelse, linjeavstand altså generell formatering. I en agile metodikk er man avhengig av tett kontakt med oppdragsgiver for hele tiden ha mulighet til endre funksjoner etter hva oppdragsgiver ønsker, foreløpig er ikke det så interessant siden vi ikke har begynt med implementasjonen av programvaren. Men vi har lagt grunnlaget for fremtidig kommunikasjon, dette har Dagfinn gjort. Dagfinn som er gruppeleder og det som scrum omtaler som

"produkt-eier". I denne rollen er han hovedkontakten med Sunnaas Sykehus (oppdragsgiver) og Mektron (samarbeidspartner). Verktøyet vi benytter oss av til å føre timer i henhold til scrum er veldig brukervennlig. Alle medlemmene i gruppen bidro til presentasjonen og var godt forberedt. Vi fikk gode tilbakemeldinger om at vi viste stor entusiasme og at "prosjektet var i trygge hender". Gruppen svarte bra på spørsmål om mangler til prosjektet under første presentasjon.

Dagfinn: Jeg syntes vi har jobbet godt sammen i prosjektet, vi har jobbet veldig mye sammen på skolen spesielt rett før jul og arbeidsflyten vår i disse sesjonene har hvert god. Jeg syntes alle har hvert flinke til å ta til seg oppgaver, og det har sjeldent hvert behov for å mase på noen for å få ting gjort da alle så langt har jobbet godt selvstendig.

Marcus: Samarbeidet fungerte godt. Syntes vi jobbet jevnt og jeg er fornøyd med dokumentene vi leverte. Vi har også lært mye vi kan forbedre til de neste sprintene. Alle påtok seg oppgaver og ellers var fordelingen av arbeidsoppgavene fair.

Zana: Samarbeidet i gruppa fungerte godt. Mye på grunn av at vi består av en gruppe på seks kamerater som kommer godt overens og har god kontakt. Kontakten var hyppig og alle viste stor interesse og arbeidsvilje for prosjektet. Vi fikk på plass gode dokumenter som ble kontrollest av flere på gruppa.

Jan Erik: Jeg syntes gruppen har fungert fint sammen. Sammarbeidet har vært godt og alle har kommet med innspill og tanker i diskusjoner. I fellesøktene har arbeidsmoralen vært god. Alle i gruppen var flinke til å på ta på seg oppgaver. Totalt sett vil jeg si at dokumentene som har blitt produsert i denne perioden er gode. I løpet av denne sprinten har vi også lært oss mye som vi må forbedre til neste sprint.

Are: I denne perioden har det vært veldig godt arbeid fra alle i gruppen. Gruppa veldig godt samkjørt og alt som blir sagt blir vurdert, dette gjør at vi får veldig gode diskusjoner rundt problemer som må løses.

Henrik: Gruppen fungerer fint, alle bidrar og er aktive. Vi har vært flinke til å kontroll lese og se på produktet til andre og gitt konstruktiv tilbakemelding.

3. Det som ikke gikk bra

Sensor følte at vi manglet dokumentasjon på kravspesifikasjon og en mer detaljert prosjektplan;

- Hva er det egentlig som skal stå ferdig i Juni.
- Når skal forskjellige oppgaver være ferdige.
- Hvordan skal de overordnede kravene testes.
- Det trengs et game design dokument.

Userstories burde vært bedre planlagt, slik at de ikke hele tiden ble lagt til noe nytt når vi kom på noe selv om dette er et vanlig punkt i scrum.
Standarene ble satt litt sent i forhold til når vi begynte å skrive dokumentasjonen.

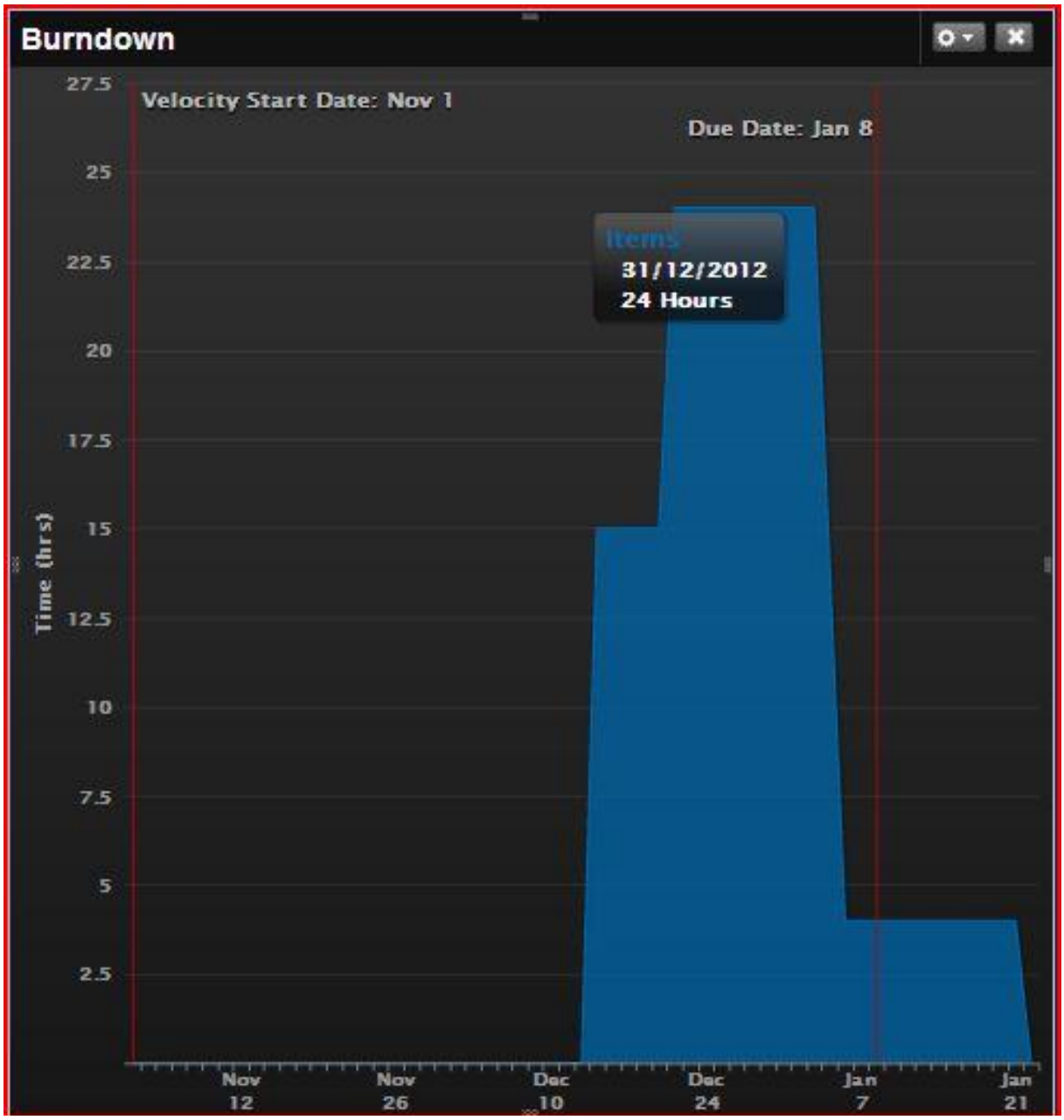
Dagfinn: Alle har så langt jobbet godt selvstendig men en del av oss bør nok bli litt flinkere til å spørre hverandre om ting vi lurer på. Vår hovedutfordring denne sprinten har i stor grad hvert å tilpasse oss SCRUM og verktøyet OnTime, og det har også tidvis hvert vanskelig å vite konkret hva vi bør jobbe

med videre. Vi har nå fått en del tips til å hjelpe oss med planleggingen videre. Blant annet skal vi nå lage et GDD dokument og en overordnet prosjektplan.

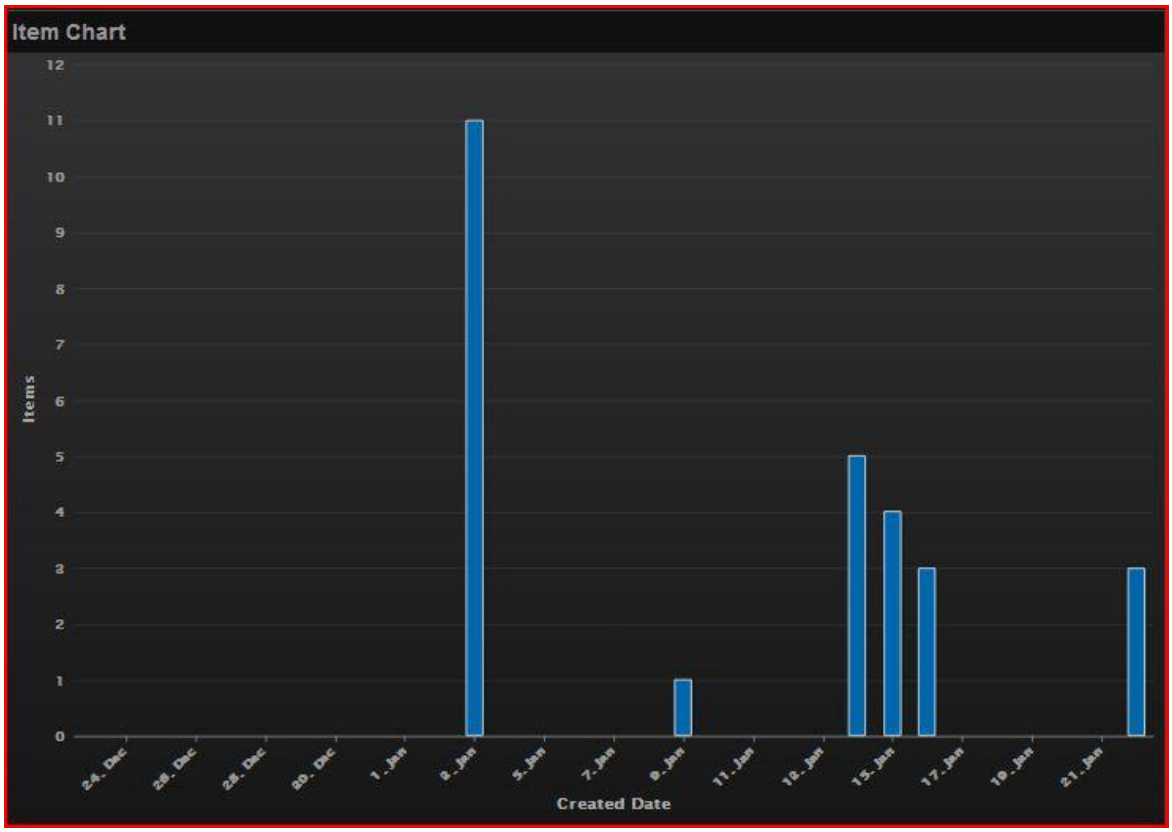
Marcus: Burde kanskje satt oss ned og opprettet user storys med en gang og ikke etterhvert som vi kom på nye. Glemte innholdsfortegnelse og burde levert dokumentene i en perm slik at sensor lettere finner frem i dokumentene. Fikk oppgaven i boks litt sent, så fikk ikke skrevet en skikkelig forstudie.

Zana: Vi burde planlagt videre arbeid mer grundig og tatt større høyde for utfordringer vi kan møte på i prosjektet. Slik spørsmålene vi fikk på 1. presentasjonen antydet.

Jan Erik: Bedre planlegging av sprinten, slik at alle userstories er på plass og ført inn i OnTime ved sprint start. Ved å gjøre dette vil vi slippe å føre opp userstories etter hvert som vi kommer på dem i sprinten og får dermed en mer effektiv og produktiv sprint. Vi burde alle i gruppen bli flinkere til å bruke verktøyet OnTime på en effektiv og riktig måte, noe som til tider var dårlig i denne sprinten.
Are:
Henrik: Vi kunne planlagt sprintene bedre, gitt et bedre bilde om hva som skal være ferdig til en hver tid, jeg synes også at vi kunne vært flinkere til å føre timer i ontime.



Her ser vi at trenden får en brå slutt, det kommer av at en oppgave ble liggende ontime alt for lenge og så ble den slettet.



Her ser vi hvilke oppgaver som blir sluttført forskjellige datoer, det er et problem når gruppen plutselig sluttfører oppgaver som har vært ferdige den dagen de har jobbet.

4. Konklusjon

Vi må forholde oss til spørsmålene angående mangler på dokumentasjonen, lage et game design document, lage en prosjektplan hvor vi har setter opp hva som skal være gjennomført i hver sprint. Få delt opp alle user storiene i mindre krav slik at det blir lettere å fordele og at det fortsatt ikke er så små oppgaver at det hemmer effektiviteten i prosjektet. Vi har hatt tett kontakt med intern veileder som har gitt mye bra input gjennom hele perioden hittil.

Debrief: Sprint 2

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	05.02.13	MJ
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA og DK

Innholdsfortegnelse

1. Innledning

Sprint 2 var en innsamlingssprint. Vi har vært to ganger på Sunnaas sykehus på spillsesjoner med pasienter og ergoterapeuter. Der har vi sett etter hvilke utfordringer spillerne har, samt hentet inn ønsker/krav vi må ha med i spillet. Vi har også brukt en del tid på å spille SCRUM-poker, for å sette opp user stories og kartlegge hvor lang tid vi tror de tar. Mot slutten av sprinten har vi jobbet litt i UML for å få konkret oversikt over hvordan spillet fungerer.

2. Det som gikk bra

Vi fikk et innblikk i hvordan det er for pasienter med diverse utfordringer å spille spill på Kinect. Vi fikk en del input både fra observasjoner og tilbakemeldinger fra pasienter om hva de ønsket/ser for seg at vi har med i spillet. Blant annet et alternativ til hvordan man starter spillet. Ofte er det standard at man holder hendene i været for å starte.

Vi fikk også tid til å planlegge neste sprint, og vi føler det gikk bra å spille SCRUM-poker til å fordele antall timer til hver user story.

3. Det som ikke gikk bra

Vi føler at vi ikke fikk like mye ut av innsamlingsfasen og møtene på Sunnaas som vi på forhånd hadde håpet på. Vi er også litt skuffet over oppmøtet av pasienter, vi hadde kanskje fått mer ut av spillsesjonene om det var en større andel pasienter som møtte opp.

4. Evaluering

Dagfinn: Denne sprinten har vi jobbet mye med videre planlegging av prosjektet. Vi har blant annet spilt SCRUM poker for å dele opp oppgavene i mindre deler, vi har dessuten jobbet mye for å få på plass en overordnet plan over utviklingen og videre fremdrift. Det siste vi jobbet med denne sprinten har hvert å lage UML diagrammer for utviklingen for å få på plass grunnarkitekturen. Jeg syntes samarbeidet vårt så langt har hvert bra og er veldig spent på hvordan det blir å jobbe sammen med implementeringen og ikke bare dokumentering.

Henrik: Her jobbet vi mye med videre planlegging av prosjektet, vi ønsket å legge alle mulige funksjoner i produktet på bordet og dele det opp for å kunne få bedre flyt i prosjektet. Vi har prøvd å sette sammen en overordnet prosjektplan for å prøve å vise hva som vil være ferdig i slutten av Mai. Vi ønsket også å sette opp noen uml diagrammer over hvordan arkitekturen skal bli, problemet med det er at Unity som er en lukket kildekode vil være vanskelig å presentere i diagrammene. Så langt fungerer samarbeidet brukbart, jeg har følt at jeg har vært veldig ivrig til å komme i gang med implementasjonen.

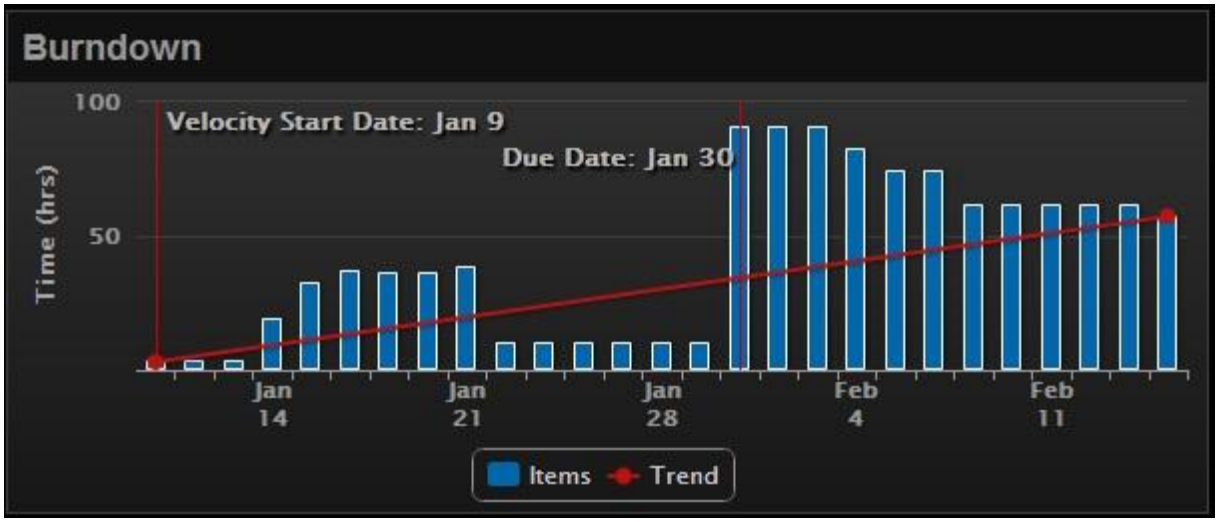
Are: I løpet av denne sprinten har vi jobbet med game design dokumentet og fått delt opp og laget overordene userstorys for spillet. Vi har også laget uml diagram av de viktigste userstoryene, Disse skal deles mer opp etter hvert som vi begynner å arbeide med dem, akkurat nå er de veldig uoverkommelige. Vi har også brukt scrum-poker som fungerte overraskende bra, så vi har fått angitt hvor lang tid vi tror er nødvendig for å gjennomføre de forskjellige oppgavene. Videre nå kommer vi til å jobbe med arkitekturen til spillet.

Zana: Vi har fått delt opp de overordna funksjonene spillet skal ha inn i mindre user-stories og delt de ut i sprintene. Dette er veldig viktig for å fokusere på det som er vanskelig og viktig i systemet først, og det har vi gjort ved å spille scrum-poker. Det gjør det også enklere å ha et utgangspunkt å jobbe ut ifra. VI har også drevet en del med modellering (UML) for å kartlegge mer konkret hvordan systemet fungerer og henger sammen. Jeg har også skrevet innledende analysedokument som skal bestå av UML diagrammene.

Vi hadde håpet på å få mer ut av spillsesjonene, selv om vi fikk en god del nyttig informasjon og tilbakemelding. Oppmøte var ikke like bra som vi hadde håpet på, men det er noe vi satser på endrer seg når vi har noe spillbart å vise frem.

Jan Erik: Tiden i denne sprinten har for det meste gått med til å planlegge veien videre i prosjektet. Vi har delt inn oppgavene i mindre deler. For å estimere antall timer arbeid på hver oppgave har vi spilt SCRUM-poker for å kunne gi et estimat. Estimatene vi har kommet frem til mener jeg på noen områder er noe forsiktige. Mot slutten av sprinten ble det fokusert mye på UML diagrammer til applikasjonen. Spillsesjonene vi har deltatt har gitt oss innsikt over hvilke utfordringer vi står ovenfor når det kommer til vår pasient målgruppe. Samarbeidet innad i gruppen har så langt vært bra og det ser lovende ut for videre arbeid i prosjektet med tanke på implementeringen.

Marcus: Vi fikk delt inn de overordna user storyene i hvor mange timer vi tror hver av dem tar, noe som er bra. Vi har også fått planlagt sprint 3 en del. Hvor vi nå skal begynne å implementere. Jeg syntes det er ganske vanskelig å modellere i UML, fordi det er mye som er veldig abstrakt. SCRUM-poker'n fungerte bra. Jeg syntes ikke vi fikk samla inn like mye som jeg håpte på, med tanke på dårlig oppmøte på Sunnaas.



Oppsummering: Fullførte userstories

ID	Title	Workflow Step	Priority	Assigned To	Original Estim
79	UML: Danse	New Request	Medium		8 hrs
82	UML: Musikk	New Request	Medium	Henrik Olsson	8 hrs
83	**UML: Prestasjoner**	**New Request**	**Medium**	**Dagfinn Kjærne**	**8 hrs**
86	UML: Danseinstruktør	New Request	Medium		8 hrs
	25 Items, 91.8 Hours Worked, 24 Hours Remaining, 79.3 % Complete				
58	Debrief doc: Sprint 1	Completed	High	Henrik Olsson	4 hrs
59	GDD: Teknologi	Completed	High	Marcus Jernberg	4 hrs
60	GDD: Dansebevegelser	Completed	High	Zana Ali	6 hrs
61	**GDD: Statistikk**	**Completed**	**High**	**Dagfinn Kjærne**	**5 hrs**
62	**GDD: Prestasjoner**	**Completed**	**High**	**Dagfinn Kjærne**	**5 hrs**
63	GDD: Interface	Completed	High	Henrik Olsson	8 hrs
64	GDD: Innledning	Completed	High	Jan Erik Helskog	3 hrs
65	GDD: GamePlay	Completed	High	Jan Erik Helskog	8 hrs
66	GDD: Dynamisk Vanskelighetsgrad	Completed	High	Are Oven	0
67	Sprint-Plan del 1	Completed	High	Henrik Olsson	2 hrs
69	GDD: FAQ	Completed	High	Marcus Jernberg	1.5 hrs
70	GDD: Lyd	Completed	High	Are Oven	0
71	GDD: Sette sammen	Completed	High	Jan Erik Helskog	2 hrs
72	**Testdokumentasjon: Brukerskjema**	**Completed**	**High**	**Dagfinn Kjærne**	**2 hrs**
73	**Skjemadesign: Dagfinn**	**Completed**	**High**	**Dagfinn Kjærne**	**2 hrs**
74	Skjemadesign: NAVN	Completed	High	Jan Erik Helskog	2 hrs
75	UML Research	New Request	High	Marcus Jernberg	5 hrs
76	UML: Research	Completed	High	Henrik Olsson	5 hrs
77	UML: Research	Completed	High	Zana Ali	5 hrs
78	**UML: research**	**Completed**	**High**	**Dagfinn Kjærne**	**5 hrs**
80	UML: Tilpasse	New Request	High	Marcus Jernberg	8 hrs
81	UML: Vanskelighetsgrad	In Progress	High	Are Oven	8 hrs
84	UML: Konto	Completed	High	Zana Ali	8 hrs
87	UML: Nullstille passord	Completed	High	Zana Ali	2 hrs
88	UML: Research	Completed	High	Jan Erik Helskog	5 hrs
29 User Stories • 91.8 Hours Worked • 52 Hours Remaining					

4. Konklusjon

Samarbeidet innad i gruppen er fortsatt like bra. Vi fikk planlagt og laget en overordnet plan som skal hjelpe oss fremover. Mesteparten av tiden ble brukt til planlegging og estimering, noe som fungerte bra. Besøket hos Sunnaas gav oss en indikasjon på hvilke utfordringer vi står ovenfor med tanke på utviklingen. Å jobbe med design og arkitektur i UML har vist seg å være ganske abstrakt og vanskelig. Og blir noe vi må jobbe mer med i sprintene frem mot 2. Pres.

Som man ser av oppsummeringsbildet ble vi ferdige med GDDet, men det meste vi gjorde på UML oppgavene ble vi ikke ferdige med da vi ikke var helt fornøyd med resultatet. Vi har nå fått frisket opp UML kunnskapene og kommer til å jobbe videre med en måte å integrere dette i designdokumentasjonen vår.

Sprintplan for sprint #2

ID	Title	Workflow Step	Priority	Assigned To	Original Estim
79	UML: Danse	New Request	Medium		8 hrs
82	UML: Musikk	New Request	Medium	Henrik Olsson	8 hrs
83	**UML: Prestasjoner**	**New Request**	**Medium**	**Dagfinn Kjærne**	**8 hrs**
86	UML: Danseinstruktør	New Request	Medium		8 hrs
	25 Items, 91.8 Hours Worked, 24 Hours Remaining, 79.3% Complete				
58	Debrief doc: Sprint 1	Completed	High	Henrik Olsson	4 hrs
59	GDD: Teknologi	Completed	High	Marcus Jernberg	4 hrs
60	GDD: Dansebevegelser	Completed	High	Zana Ali	6 hrs
61	**GDD: Statistikk**	**Completed**	**High**	**Dagfinn Kjærne**	**5 hrs**
62	**GDD: Prestasjoner**	**Completed**	**High**	**Dagfinn Kjærne**	**5 hrs**
63	GDD: Interface	Completed	High	Henrik Olsson	8 hrs
64	GDD: Innledning	Completed	High	Jan Erik Helskog	3 hrs
65	GDD: GamePlay	Completed	High	Jan Erik Helskog	8 hrs
66	GDD: Dynamisk Vanskelighetsgrad	Completed	High	Are Oven	0
67	Sprint-Plan del 1	Completed	High	Henrik Olsson	2 hrs
69	GDD: FAQ	Completed	High	Marcus Jernberg	1.5 hrs
70	GDD: Lyd	Completed	High	Are Oven	0
71	GDD: Sette sammen	Completed	High	Jan Erik Helskog	2 hrs
72	**Testdokumentasjon: Brukerskjema**	**Completed**	**High**	**Dagfinn Kjærne**	**2 hrs**
73	**Skjemadesign: Dagfinn**	**Completed**	**High**	**Dagfinn Kjærne**	**2 hrs**
74	Skjemadesign: NAVN	Completed	High	Jan Erik Helskog	2 hrs
75	UML Research	New Request	High	Marcus Jernberg	5 hrs
76	UML: Research	Completed	High	Henrik Olsson	5 hrs
77	UML: Research	Completed	High	Zana Ali	5 hrs
78	**UML: research**	**Completed**	**High**	**Dagfinn Kjærne**	**5 hrs**
80	UML: Tilpasse	New Request	High	Marcus Jernberg	8 hrs
81	UML: Vanskelighetsgrad	In Progress	High	Are Oven	8 hrs
84	UML: Konto	Completed	High	Zana Ali	8 hrs
87	UML: Nullstille passord	Completed	High	Zana Ali	2 hrs
88	UML: Research	Completed	High	Jan Erik Helskog	5 hrs

29 User Stories • 91.8 Hours Worked • 52 Hours Remaining

Debrief: Sprint 3

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	27.02.13	JEH
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	AO, DK

Innholdsfortegnelse

1. Innledning

I sprint 3 har vi brukt tiden på å planlegge arkitektur og struktur til applikasjonen. Det har blitt brukt mye tid på å få fordelt arkitekturen opp i mindre biter, slik at hver enkelt del ikke ble for stor. Disse oppgavene har så blitt fordelt og ført opp i OnTime. Tidsforbruk på forskjellige arkitektur oppgaver har blitt bestemt ved å spille scrum-poker. Mot slutten av sprinten begynte hver enkelt å jobbe med sine oppgaver. I tillegg har vi i løpet av denne sprinten vært på Sunnaas å holdt presentasjon for dem.

2. Det som gikk bra

Gruppen har jobbet bra og alle er ivrige etter å komme igang med implementeringen av arkitekturen. Denne sprinten ble mye bedre planlagt enn tidligere sprinter. Dette førte til at utførte arbeidsmengde og effektiviteten ble bedre. Når vi skulle planlegge hvilke deler som var viktige og tilhørte arkitekturen til spillet deltok alle aktivt og la frem sine meninger. Dette gjelder også under scrum-pokeren, alle deltok og argumenterte for sine valg der det forekom forskjeller på ønsket tids estimat.

Etter arkitektur planlegningen, har alle tatt til seg en eller fler userstories. Dette er bra da vi bruker scrum og er avhengige av at gruppe medlemmer tar initiativ selv. Alle medlemmene har kommet godt i gang på hvert sitt område på arkitekturen og er klare for en ny sprint med fokus på implementering av arkitekturen.

Foredraget på Sunnaas Sykehus gikk bra og alle som deltok var fornøyd med hvordan det gikk. Tilbakemeldingene etter foredraget bare positive og vi fikk mange forslag til hvordan ting kunne se ut og burde være.

3. Det som ikke gikk bra

Vi føler at UML delen av arkitektur planlegningen blir veldig abstrakt. Dette kommer av at vi bruker unity som kommer til å gjøre masse kalkulasjoner for oss og vi vil i all hovedsak bruke enkeltstående scripts. Klasse diagrammene vil dermed bli veldig tynne og vi vil derfor for det meste få skikkelig bruk for aktivitets diagram og sekvensdiagrammer. Så dette er noe vi må se nærmere på i løpet av neste sprint slik at vi får en god design dokumentasjon på plass.

4. Evaluering

Dagfinn: Vi hadde en real runde med scrum poker for å anslå timer og dele opp oppgaver, spesielt i forbindelse med arkitekturen, men en del av oppgavene var vanskelige å kutte ned i små konkrete oppgaver da vi foreløpig ikke har så mye kunnskap om dem enda. Jeg håper vi får dette bedre til etter hvert som vi jobber med prosjektet slik at vi mot slutten kun har korte konkrete user stories, istedenfor de store stygge ulvene vi har nå med 20 timer eller mer.

Henrik: I denne sprinten brukte vi mye tid på å planlegge hvor lang tid diverse oppgaver, hvordan de skulle gjøres og hvor lang tid de kom til å ta. Det var vanskelig å annslå hvor lang tid diverse oppgaver kom til å ta siden vi alle er generelt dårlige på å se for oss hvor lang tid oppgaver kan ta. Jeg og Are var også i Oslo på møte med Oslo Medtech her fikk vi høre en del om bruk av spill til rehabilitering vi var også i et møte hvor vi fikk en del input på hvordan systemet vårt kan bli i fremtiden. Arkitekturen var vanskelig å designe da vi benytter oss av Unity, unity har diverse klasser og funksjonskall som er skjult for utviklerne så det blir en annen workflow en om man skal bygge noe fra scratch.

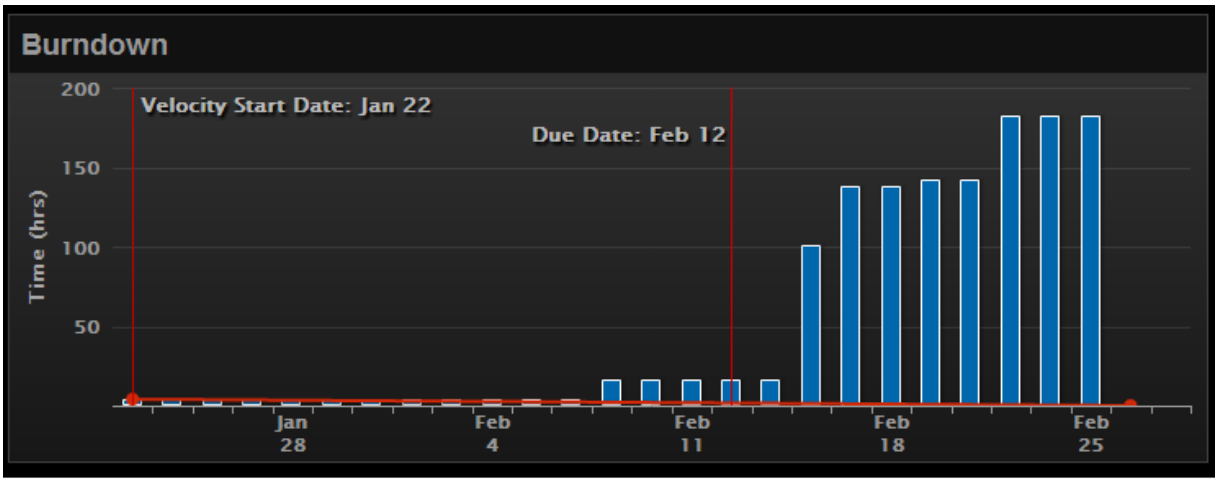
Are: I løpet av denne sprinten har vi vært gjennom ett par runder med scrum-poker der vi alle sa hvor lang tid vi tror vi kommer til å bruke på hver userstory. Jeg og Henrik var også på en samling i Oslo. Der ble det tatt opp spill i bruk til rehabilitering, etter samlingen var vi i et møte der vi det ble snakket om videreutvikling av spillet for oss og etter endt prosjekt.
Det blir jobbet godt fra alle parter i gruppa.

Marcus: Det fungerte bra å spille Scrum poker for å anslå timer. Vi hadde en presentasjon på Sunnaas som gav oss mange gode tilbakemelidinger. Vi har nå jobbet litt med arkitekturen. Det er utfordrene å få til mye av det vi jobber med. Men jeg føler i hvertfall at jeg gjør fremskritt for hver dag. Jeg syntes det er vanskelig å planlegge i UML mtp at vi bruker Unity. Men håper å få det på plass i løpet av neste sprint. Gruppesamarbeidet fungerer fortsatt godt.

Zana: I denne sprinten var det planlegging som var felles fokus. Som en del av dette har vi spilt SCRUM-poker for å anslå tiden en user-story tar og argumentere for det. Jeg har også sammen med nesten hele gruppa vært på Sunnaas og holdt 1. presentasjon for ergoterapeuter og andre på

Sunnasas. En liten user-story jeg har jobbet med har vært å sette seg inn i Doxygen siden vi skal bruke Doxygen til å dokumentere kildekode.

Jan Erik: Nok en gang var det en sprint med mye planlegging som sto i fokus. Det ble spilt ett par runder med poker for å få delt inn oppgavene og gitt dem et tidsestimat. Videre som nevnt i debriefen var vi og hadde en presentasjon på Sunnaas. Denne presentasjonen ga oss mange gode tilbakemeldinger. I tillegg til dette har vi påbegynt arbeidet med arkitekturen. Det er mye å sette seg inn i, men jeg føler vi er på riktig vei og ting begynner å ta form. Det er utfordrende å planlegge med UML da vi bruker Unity. En av user-storyene mine ble flyttet over i neste sprint.



Oppsummering: Userstories som ble fullført ila sprint 3

ID	Title	Workflow Step	Priority	Assigned To	Original Estima
edium	1 Items, 1 Hours Worked, **0 Hours Remaining**, 100% Complete				
119	Sette seg inn i Doxygen	Completed	Medium	Zana Ali	4 hrs
gh	9 Items, 49 Hours Worked, **0 Hours Remaining**, 100% Complete				
89	Sprint 2 Debrief	Completed	High	Marcus Jernberg	2 hrs
90	Friske opp Unity-kunnskapene	Completed	High	Marcus Jernberg	8 hrs
91	**Unity: Database testing**	**Completed**	**High**	**Dagfinn Kjærnet**	**8 hrs**
92	Audio Import	Completed	High	Henrik Olsson	20 hrs
94	**Akritektur: Database - Kontakt med database**	**Completed**	**High**	**Dagfinn Kjærnet**	**1 hrs**
96	**Akritektur: Database - Lese fra database**	**Completed**	**High**	**Dagfinn Kjærnet**	**2 hrs**
98	**Akritektur: Database - Skrive til database**	**Completed**	**High**	**Dagfinn Kjærnet**	**2 hrs**
103	Arkitektur: Dansescene - Kontakt med kinect	Completed	High	Marcus Jernberg	5 hrs
118	Unity: Friske opp Unity kunskap	Completed	High	Jan Erik Helskog	8 hrs

5. Konklusjon

Vi er nødt til å finne ut hvordan vi vil gjøre det med UML og designdokumentajson slik at vi får en god standard på hvordan den skal skrives. Videre må vi fortsette å jobbe på og ta initiativ når noe må gjøres og beslutninger må tas.

Som man ser av oppsummeringen ble en stor del av userstoriene for store for denne sprinten. De fleste av disse vil vi derfor fortsette med neste sprint. Fremover kommer vi til å fokusere på å dele opp userstoriene litt mer enn det vi har gjort til nå, men da en del av disse oppgavene har hvert veldig fremmede for oss har det hvert vanskelig i denne omgang.

Sprintplan for sprint #3

ID	Title	Workflow Step	Priority	Assigned To
Medium	2 Items, 0 Hours Worked, **8 Hours Remaining**, 0% Complete			
68	GDD: Modeller	New Request	Medium	Henrik Olsson
119	Sette seg inn i Doxygen	New Request	Medium	Zana Ali
High	18 Items, 43.5 Hours Worked, **203.5 Hours Remaining**, 17.6% Complete			
85	UML: Statistikk	In Progress	High	Jan Erik Helskog
89	Sprint 2 Debrief	In Progress	High	Marcus Jernberg
90	Friske opp Unity-kunnskapene	In Progress	High	Marcus Jernberg
91	**Unity: Database testing**	In Progress	**High**	**Dagfinn Kjærnet**
92	Audio Import	In Progress	High	Henrik Olsson
93	Audio Analyzer	New Request	High	
94	**Akritektur: Database - Kontakt med database**	In Progress	**High**	**Dagfinn Kjærnet**
95	Arkitektur: Generator - Legge til bevegelser	In Progress	High	Are Oven
96	**Akritektur: Database - Lese fra database**	In Progress	**High**	**Dagfinn Kjærnet**
97	Arkitektur: Generator - Hente dans	New Request	High	
98	**Akritektur: Database - Skrive til database**	In Progress	**High**	**Dagfinn Kjærnet**
99	Arkitektur: Dansescene - danserom	In Progress	High	Marcus Jernberg
103	Arkitektur: Dansescene - Kontakt med kinect	New Request	High	
105	Arkitektur: Dansescene - Koble avatar mot kinect	In Progress	High	Marcus Jernberg
106	Arkitektur: Dansescene - Grid	In Progress	High	Zana Ali
109	Audio Output Interface	New Request	High	
113	Arkitektur: Meny	In Progress	High	Jan Erik Helskog
118	Unity: Friske opp Unity kunskap	Completed	High	Jan Erik Helskog

Debrief: Sprint 4

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	16.03.2013	DK
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	HO, AO

Innholdsfortegnelse

1. Innledning

I sprint 4 har vi jobbet med implementasjonen av arkitekturen og målet har hvert å ha klar en prototype vi kan vise frem på andre presentasjon. Vi har blant annet jobbet med å få på plass et menysystem, database, musikkbehandling(import og avspilling), en måte å håndtere dansebevegelser på(f.eks hvordan lagre en dansebevegelse), en måte å registrere dansebevegelser på(hvordan registrere at bevegelsen spilleren utfører er den riktige dansebevegelsen) og ikke minst en måte å oppfatte og registrere spillerens bevegelser. Vi har dessuten begynt å se litt på muligheten for å analysere musikken slik at vi kan auto generere danser for å prøve å få et bedre overblikk over hvor omfattende denne delen av oppgaven vil bli og om den i det hele tatt vil la seg gjennomføre ila tidsplanen.

2. Det som gikk bra

Gruppen arbeider godt sammen og når noen står fast er vi flinke til å spørre hverandre om hjelp. Vi kom nesten helt i mål med prototypen vi ønsket å ha klar og kommer til å sette av litt tid i neste sprint for å få den klar til presentasjon 2.

3. Det som ikke gikk bra

Vi sliter fremdeles en del med beregning av tid og oppdeling av arbeidsoppgaver, men føler det har kommet seg en del siden sist.

4. Evaluering

Dagfinn: Jeg har i denne sprinten jobbet med å utvikle scriptet som tar seg av databasehåndteringen, i utgangspunktet var det satt av liten tid til dette og planen var at jeg skulle hjelpe til på den av de større oppgavene vi regnet med ville kreve mer tid og resurser. Det viste seg forøvrig å bli en større oppgave enn antatt og endte opp med å oppta det meste av tiden min, men jeg har prøvd å bidra litt overalt når noen trengte hjelp. Heldigvis har godt samarbeid på gruppen ført til at vi på tross av dette har kommet svært nær målet vi satt oss med å ha en prototyp klar til presentasjon 2.

Henrik: Jeg har i denne perioden jobbet med å utvikle muligheten til å velge mellom forskjellige lydfiler når spillet er i runtime, og da laste dette inn i spillet for så å spille det av. Videre ønsket jeg å kunne visualisere lydfilen samtidig som den spilles av, dette har jeg måtte jobbe videre med siden det var tidkrevende. Når dette er klart ønsker vi å få til en mulighet til å plukke ut slike beats i en låt for å sette dansebevegelsene i takt med musikken.

Are: I sprint fire begynte vi å programmere arkitekturen. Jeg har jobbet med å lage en dansegenerator. Denne skulle ha dansebevegelsene lagret og kunne generere et utvalg av dem. Jobbingen med dette har gått veldig bra. Det ble brukt ganske lang tid på å finne ut hvordan jeg skulle gjøre det, og kom frem til en enkel måte å gjøre dette på. Når løsning først kom ble ting veldig lett. Jeg har også jobbet litt sammen med Zana på griden.
Samarbeid i gruppen fungerer veldig bra og alle hjelper til når det trengs.

Marcus: Jeg har i denne sprinten jobbet med å få kontakt mellom kinecten og avataren. Denne userstoryen hadde i utgangspunktet blitt estimert til 20 timer. Det stemte ganske bra, jeg brukte ca 16.5 timer på å få laget noe som er godt nok til at vi kan bruke det videre. Det er veldig avansert scripting for å gjøre det mulig å koble avataren mot kinect, så jeg brukte Zigfu og OpenNI for å få det til.

Jeg har også designet danserommet, det gikk bra. Brukte litt mindre enn forhåndsestimert tid her også. Resten av sprinten har jeg brukt på dokumentasjon, og så har jeg og Zana jobbet med å implementere Griden i danserommet sammen med avataren.

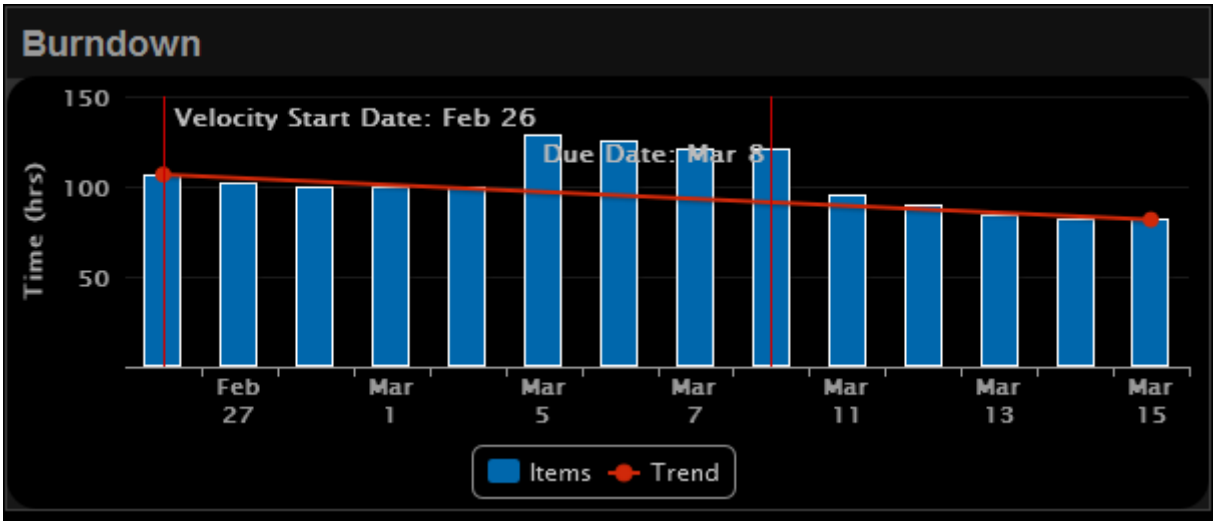
Det er vanskelig å estimere tid, men til neste gang så vet vi litt bedre hvor lang tid ting tar.

Zana: Jeg har i hovedsak jobbet med å implementere griden i denne sprinten. Tiden brukt på denne user-storyen var på 30 timer. Grunnen til at de forventede 40 timene ikke har blitt brukt, er at arkitekturen var på plass tidligere enn forventet, samtidig som det har dukket opp andre prioriteringer som dokumentasjon før innlevering. Sammen med Are har vi også jobbet med kommunikasjonen mellom dansebevegelser og griden. Det har gått en del tid til å dokumentere kildekode og lage innledende designdokument og designdokument mal for user-stories som danner arkitekturen for applikasjonen. Jeg har også forklart for gruppemedlemmer hvordan doxygen kan brukes.

Gruppemedlemmene har vært samarbeidsvillige og samarbeidet fungerer godt.

Jan Erik: I denne sprinten har jeg brukt mye tid på å lære meg NGUI, som er et Toolkit vi har valgt å bruke for å lage vår GUI. Dette kittet var mye å sette seg inn i, men svært effektivt når jeg fikk det inn i fingrene og prefabene laget. User-storyen jeg har jobbet mest med er arkitekturen til menysystemet. Denne var originalt 40 timer, noe som i utgangspunktet er for stort. Det ble derfor besluttet at vi deler den opp i mindre deler ut ifra storyboard bildet som jeg laget over menysystemet. Sammen med Dagfinn har vi fått til et logg-inn system som fungerer og samarbeidet om å legge til rette for at koblingen mot databasen skal bli så godt som mulig. I løpet av sprinten har jeg rukket å lage arkitekturen til logg-inn, brukerregistrering, innstillinger, mainmenu og songselect.

Vi ser nå at tidsestimering er vanskelig og at vi på noen user-storyer har bommet totalt.



Oppsummering: Userstories som ble fullført ila sprint 4

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
95	Arkitektur: Generator - Legge til bevegelser	High	Are Oven	Ready For Testing	20 hrs	0	18 hrs
99	Arkitektur: Dansescene - danserom	High	Marcus Jernberg	Completed	10 hrs	0 hrs	6.5 hrs
105	Arkitektur: Dansescene - Koble avatar mot kinect	High	Marcus Jernberg	Completed	20 hrs	0 hrs	15.5 hrs
106	Arkitektur: Dansescene - Grid	High	Zana Ali	Completed	40 hrs	0 hrs	30 hrs
113	Arkitektur: Meny	High	Jan Erik Helskog	Completed	20 hrs	0 hrs	20 hrs
121	Arkitektur: Grunnleggende database design	High	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	5 hrs
122	Møtereferat [26.02.2013]	High	Henrik Olsson	Completed	4 hrs	0 hrs	4 hrs
123	Arkitektur: Database, grunnleggende metoder.	High	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	8 hrs
124	Lage en designdokument mal	High	Zana Ali	Completed	4 hrs	0 hrs	3 hrs
125	Møteinnkallelse [05.03.2013]	High	Dagfinn Kjærnet	Completed	30 min	0 min	30 min
126	Testrutine: Databaser	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
127	Arkitektur: Dansescene - Grid (Ressurs#2)	High	Are Oven	Completed	20 hrs	0	7 hrs
130	Dokumentasjon: Audio Import	High	Henrik Olsson	Completed	2 hrs	0	2 hrs
131	Dokumentasjon: Grid.cs	High	Zana Ali	Completed	3 hrs	0 hrs	4 hrs
132	Dokumentasjon: Menysystemet - teknologi dokument NGUI	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
133	Arkitektur: Menysystem - New User Scene(3&4)	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
138	AudioVisualizer	High	Henrik Olsson	In Progress	20 hrs	4 hrs	16 hrs
140	Dokumentasjon: Zigfu	High	Marcus Jernberg	Completed	3 hrs	0 hrs	3 hrs
141	Dokumentmal: Testcaseoversikt	High	Dagfinn Kjærnet	Completed	1.5 hrs	0	0
143	Dokumentasjon: Sprint 4 Debrief	High	Dagfinn Kjærnet	Completed	2 hrs	0	2 hrs
144	Teknologidokument: SQLite	High	Dagfinn Kjærnet	Completed	2 hrs	0 min	2 hrs

5. Konklusjon

Når vi sammenligner sprintplanen og de utførte oppgavene for denne sprinten ser vi at vi har hvert flinkere til å planlegge oppgavene som skal utføres og de fleste oppgavene som ble lagt til greide vi å fullføre. En del av dem har vi forøvrig stykket opp i mindre biter og noen av disse har vi måttet utsette til senere sprinter, og vi kommer fremdeles på nye oppgaver underveis som vi legger til og utfører fortløpende.
Vi er dessuten svært nær å fullføre vår første prototype og regner med å få denne på plass ila neste sprint, som er en mini sprint før 2 presentasjon. I denne prototypen håper vi å kunne demonstrere litt av menysystemet og det at vi kan utføre bevegelser.

Vi har også gjort noen endringer i dokumentasjonen underveis. Siden de overordnede kravene nå er kuttet ned i opptil flere små userstories har vi valgt å lage en egen kravspesifikasjon slik at vi lettere kan holde øye med hvilke userstories som må fullføres for å kunne anse et av de overordnede kravene som fullført.

Sprintplan for sprint #4

ID	Title	Workflow Step	Priority	Assigned To	Original Estima
	21 Items, 62 Hours Worked, 124.5 Hours Remaining, 33.2% Complete				
95	Arkitektur: Generator - Legge til bevegelser	In Progress	High	Are Oven	20 hrs
99	Arkitektur: Dansescene - danserom	In Progress	High	Marcus Jernberg	10 hrs
105	Arkitektur: Dansescene - Koble avatar mot kinect	In Progress	High	Marcus Jernberg	20 hrs
106	Arkitektur: Dansescene - Grid	In Progress	High	Zana Ali	40 hrs
113	Arkitektur: Meny	In Progress	High	Jan Erik Helskog	20 hrs
121	**Arkitektur: Grunnleggende database design**	**Completed**	**High**	**Dagfinn Kjærnet**	**4 hrs**
122	Møtereferat [26.02.2013]	Completed	High	Henrik Olsson	4 hrs
123	**Arkitektur: Database, grunnleggende metoder.**	**In Progress**	**High**	**Dagfinn Kjærnet**	**5 hrs**
124	Lage en designdokument mal	In Progress	High	Zana Ali	4 hrs
125	**Møteinnkallelse [05.03.2013]**	**Completed**	**High**	**Dagfinn Kjærnet**	**0.5 hrs**
126	**Testrutine: Databaser**	**In Progress**	**High**	**Dagfinn Kjærnet**	**5 hrs**
127	Arkitektur: Dansescene - Grid (Ressurs#2)	New Request	High		20 hrs
129	**Dokumentasjon: DbHandler.cs (Databaseklassen)**	**New Request**	**High**	**Dagfinn Kjærnet**	**2 hrs**
130	Dokumentasjon: Audio Import	New Request	High	Henrik Olsson	2 hrs
131	Dokumentasjon: Grid.cs	In Progress	High	Zana Ali	3 hrs
132	Dokumentasjon: Menysystemet - teknologi dokument NGUI	Ready For Testing	High	Jan Erik Helskog	4 hrs
133	Arkitektur: Menysystem - New User Scene(3&4)	New Request	High		2 hrs
134	Arkitektur: Menysystem - Song Select(5&6&7)	New Request	High		2 hrs
135	Arkitektur: Menysystem - Statistikk(8&9)	New Request	High		4 hrs
136	Arkitektur: Menysystem - Achievments(10)	New Request	High		2 hrs
137	Arkitektur: Menysystem - Admin(11)	New Request	High		1 hrs

Debrief: Sprint 5

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	23.03.13	ZA
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	

Innholdsfortegnelse

1. Innledning

Sprint 5 har vært en kort sprint som i hovedsak dreiet seg om å fullføre dokumentasjon og klargjøre dokumenter som skal leveres inn før vår 2. presentasjon. Gruppen har også brukt en del tid på forberedelse av presentasjonen.

2. Det som gikk bra

Gruppen har stått på iherdig i denne sprinten og jobbet både effektive og lenge sammen. Vi fikk på plass dokumentasjon som vi var fornøyde med, noe vi også fikk tilbakemelding på var bra etter 2. presentasjon. Vi satte stor pris på all den gode tilbakemeldingen vi fikk og var godt fornøyde etter presentasjonen. I denne sprinten har gruppen vist en god arbeidsmoral.

3. Det som ikke gikk bra

Det er oftest veldig hektisk i perioden rett før en innlevering. En ting vi kunne gjort annerledes var å fullføre dokumentasjonen på et tidligere stadium slik at de siste dagene før innlevering ble mindre hektiske. Det er noe vi skal huske på til neste innlevering.

4. Evaluering

Dagfinn: Sprint 5 har jeg i stor grad jobbet for å få ferdig en del påbegynt dokumentasjon, samt forberede presentasjon 2. Jeg er svært fornøyd med denne sprinten da alle har jobbet utrolig hardt for å få ferdig dokumentasjonen i tide og sørget for at den er på et nivå vi er tilfreds med. Vi jublet høyt og var svært fornøyde da vi omsider kunne levere fra oss dokumentasjonen og begynne på forberedelsene til 2. presentasjon. Vi jobbet effektivt også med dette og fikk svært gode tilbakemeldinger, som igjen gir oss god motivasjon videre. Til neste og siste presentasjon bør vi nok sette av litt mer tid til forberedelsen av både dokumenter og presentasjon, sånn at vi får mulighet til å slappe av litt innimellom, selv om det har hvert svært givende å virkelig jobbe under press og fremdeles føle at man oppnår det man ønsker.

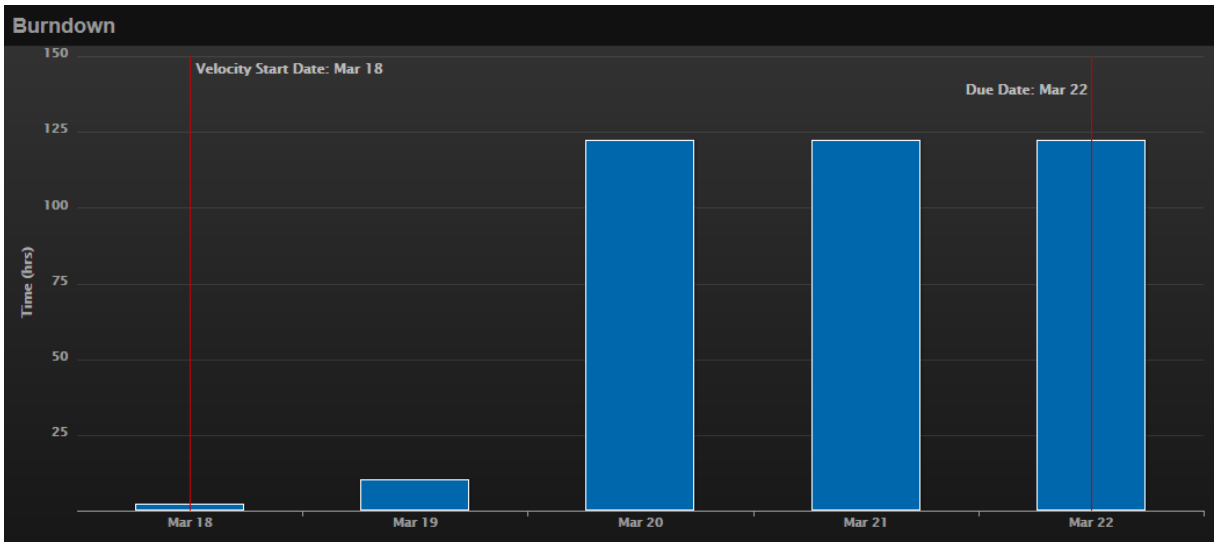
Henrik: Det var en anstrengende sprint, vi bestemte oss for å få ferdig alt av design og analyse dokumentasjon til levering. Dette ble tidkrevende, men gruppen stod på og leverte en solid samling dokumenter. Noe som jeg mener at vi har all grunn til å være stolte av. Vi hadde kun en liten periode på å øve til presentasjonen, hvor gruppen gjorde en flott innsats. Presentasjonen ble bra utført, vi fikk mye god tilbake melding som vil være motivasjon for innspurten til siste presentasjon og levering av produktet.

Are: Zana og jeg har jobbet mye sammen med denne sprinten, med i hovedsak danseinstruktøren og få dybde i gridden. Dette har for det meste gått veldig bra. Jeg har også gjort en del forandringer på dansegeneratoren.

Marcus: Mestparten av denne sprinten gikk ut på å få ferdig dokumentasjon til andre presentasjon. Det var tidskrevende, men gruppa stod på, og vi klarte å få på plass alt vi trengte. Etter at dokumentasjonen var levert, ble det noen intense timer med øving før presentasjonen. Noen av oss jobbet også med å få på plass en liten demonstrasjon av det vi hadde fått på plass til nå. Presentasjonen gikk kjempebra, og alle kan være stolte av innsatsen vi la den i denne sprinten.

Zana: Jeg er stolt over gruppa og måten vi har stått på. Alle har bidratt til å gjøre resultatet best mulig i form av dokumentasjon og presentasjon. Etter denne sprinten er motivasjonen på topp for å jobbe videre med prosjektet og det er noe jeg ser frem til.

Jan Erik: I sprint 5 har jeg i all hovedsak jobbet med å forberede presentasjon 2, samt gjøre ferdig dokumentasjonen som måtte på plass. Arbeidsinnsatsen til gruppen i denne sprinten er jeg svært fornøyd med. Alle jobbet hardt for å få ferdig dokumentasjonen i tide med et kvalitetsnivå vi selv kan si oss fornøyd med. Etter at all dokumentasjonen var levert fortsatte gruppen med samme arbeidsinnsats for å få ferdig presentasjon 2, samt intens øving. Presentasjonen ble bra utført, med mye god tilbakemelding fra sensorer og veiledere. Det samme gjaldt kvaliteten på dokumentasjonen som ble levert. Kort summert mener jeg at alle kan være stolte av innsatsen de har lagt ned i denne sprinten.



Oppsummering: Userstories som ble fullført ila sprint 5

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
82	UML: Musikk	High	Henrik Olsson	Completed	8 hrs	0 hrs	10 hrs
83	UML: Prestasjoner	High	Dagfinn Kjærnet	Completed	8 hrs	0 hrs	6 hrs
85	UML: Statistikk	High	Jan Erik Helskog	Completed	8 hrs	0 hrs	10 hrs
104	Funksjon: Generator - Generere dans	High	Are Oven	Completed	10 hrs	0 min	3 hrs
129	Dokumentasjon: Database og databasehåndtering	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
134	Arkitektur: Menysystem - Song Select(5&6&7)	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	8 hrs
146	Prototype: Koble avatar mot grid	High	Marcus Jernberg	Completed	2.5 hrs	0 hrs	2.5 hrs
147	Prototype: Koble avatar mot grid	High	Zana Ali	Completed	2.5 hrs	0 hrs	2.5 hrs
148	DesignDokument: Koble avatar mot kinect	High	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
156	Fullføre dokumentasjon	High	Marcus Jernberg	Completed	14 hrs	0 hrs	14 hrs
157	Fullføre dokumentasjon	High	Zana Ali	Completed	14 hrs	0 hrs	14 hrs
158	Dokumentasjon: Fullføre dokumentasjon	High	Jan Erik Helskog	Completed	14 hrs	0 hrs	18 hrs
159	Fullføre dokumentasjon	High	Dagfinn Kjærnet	Completed	14 hrs	0 hrs	18 hrs
160	Fullføre dokumentasjon	High	Are Oven	Completed	14 hrs	0 hrs	18 hrs
161	Forberede presentasjon 2	High	Dagfinn Kjærnet	Completed	20 hrs	0	20 min
162	Forberede presentasjon	High	Marcus Jernberg	Completed	20 hrs	0 min	20 hrs
163	Forberede presentasjon	High	Zana Ali	Completed	20 hrs	0	20 hrs
164	Forberede presentasjon 2	High	Are Oven	Completed	20 hrs	0	20 hrs
165	Forberede presentasjon2	High	Jan Erik Helskog	Completed	20 hrs	0 min	20 hrs
166	Forberede Presentasjon	High	Henrik Olsson	Completed	20 hrs	0	20 hrs
167	Fullføre dokumentasjonen	High	Henrik Olsson	Completed	14 hrs	0 hrs	18 hrs

5. Konklusjon

Vi må jobbe videre på samme måte som før og ta til oss de tilbakemeldingene vi får. Vi bør også sette av mer tid til presentasjonsforberedelser til neste presentasjon.

Sprintplan for sprint #5

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
146	Prototype: Koble avatar mot grid	High	Marcus Jernberg	Completed	2.5 hrs	0 hrs	2.5 hrs
147	Prototype: Koble avatar mot grid	High	Zana Ali	Completed	2.5 hrs	0 hrs	0 hrs
148	DesignDokument: Koble avatar mot kinect	High	Marcus Jernberg	Completed	2 hrs	0 hrs	2 hrs
82	UML: Musikk	High	Henrik Olsson	Completed	8 hrs	0 hrs	10 hrs
83	UML: Prestasjoner	High	Dagfinn Kjærnet	Completed	8 hrs	0 hrs	6 hrs
85	UML: Statistikk	High	Jan Erik Helskog	Completed	8 hrs	0 hrs	10 hrs
104	Funksjon: Generator - Generere dans	High	Are Oven	Completed	10 hrs	0 min	3 hrs
129	Dokumentasjon: Database og databasehåndtering	High	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
134	Arkitektur: Menysystem - Song Select(5&6&7)	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	8 hrs
156	Fullføre dokumentasjon	High	Marcus Jernberg	In Progress	14 hrs	2 hrs	12 hrs
157	Fullføre dokumentasjon	High	Zana Ali	In Progress	14 hrs	4 hrs	10 hrs
158	Dokumentasjon: Fullføre dokumentasjon	High	Jan Erik Helskog	In Progress	14 hrs	4 hrs	14 hrs
159	Fullføre dokumentasjon	High	Dagfinn Kjærnet	In Progress	14 hrs	4 hrs	14 hrs
160	Fullføre dokumentasjon	High	Are Oven	In Progress	14 hrs	4 hrs	14 hrs
138	AudioVisualizer	High	Henrik Olsson	In Progress	20 hrs	4 hrs	16 hrs
167	Fullføre dokumentasjonen	High	Henrik Olsson	In Progress	14 hrs	4 hrs	0
162	Forberede presentasjon	High	Marcus Jernberg	In Progress	20 hrs	18 hrs	2 hrs
163	Forberede presentasjon	High	Zana Ali	In Progress	20 hrs	20 hrs	0
164	Forberede presentasjon 2	High	Are Oven	In Progress	20 hrs	20 hrs	0
165	Forberede presentasjon2	High	Jan Erik Helskog	In Progress	20 hrs	20 hrs	0
166	Forberede Presentasjon	High	Henrik Olsson	New Request	20 hrs	20 hrs	0
161	Forberede presentasjon 2	High	Dagfinn Kjærnet	In Progress	20 hrs	20 hrs	0

Debrief: Sprint 6

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	18.04.13	ZA
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	

Innholdsfortegnelse

1. Innledning

Sprint 6 har i likhet med sprint 5 vært en kort sprint, siden denne falt under eksamensperioden. I denne sprinten skulle det fokuseres på planlegging av implementasjon, slik at vi hadde user-stories delt opp og arbeid klare til å påbegynnes etter eksamensperioden.

2. Det som gikk bra

Gruppen møttes på skolen to dager for å planlegge videre arbeid. Til tross for at det var under eksamensperioden, og flere gruppemedlemmer hadde andre valgfag de også måtte jobbe med i denne perioden. Vi diskuterte og delte opp user-storiene som skal jobbes videre med. Noen user-stories ble også lagt til.

3. Det som ikke gikk bra

Denne sprinten ble kort og burde muligens vært noe lengre, men eksamener måtte prioriteres i tillegg.

4. Evaluering

Dagfinn: Det var veldig ålreit å samle gruppen igjen etter ferien, og vi fikk planlagt hva vi skulle gjøre etter eksamenstiden.

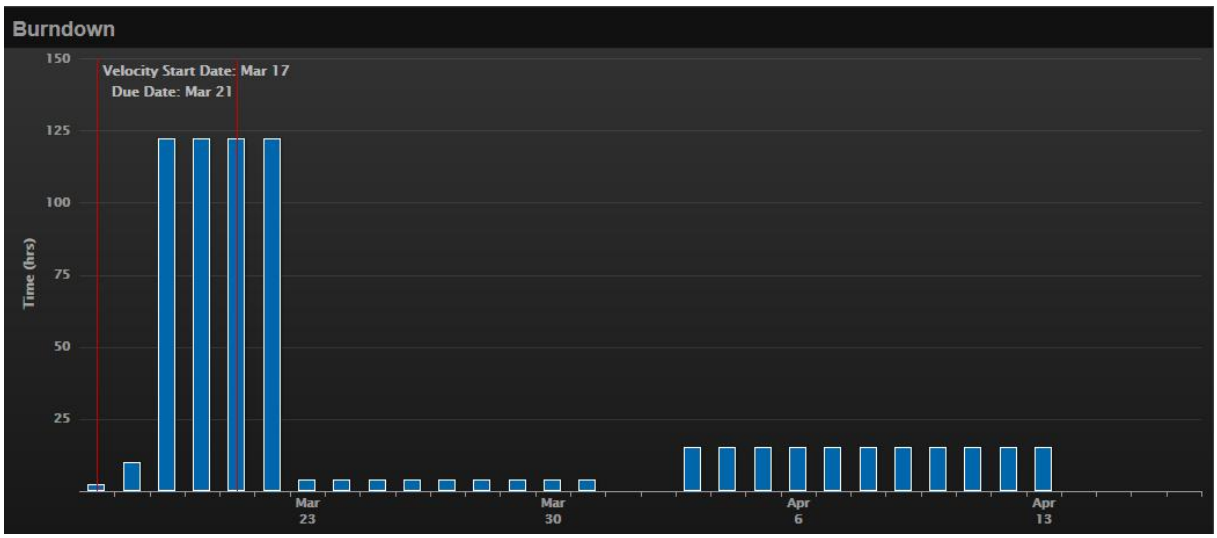
Henrik: Gruppen kom seg på plass etter påsken, vi startet med et gruppemøte og begynte planleggingen av nye krav og ideer. Vi brøt dette ned i mindre biter på en strukturert måte og gjorde anslag på tid det kom til å ta.

Are:

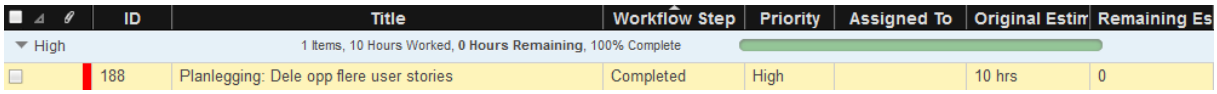
Marcus: Vi fikk gjort det vi skulle. Var også ålreit å ha et gruppemøte etter påskeferien, for å planlegge oppstart av neste sprint, samt dele opp/legge til nye user stories.

Zana: Ikke så mye å si om denne sprinten, annet enn at jeg synes det var bra vi møttes i en periode der vi hadde andre ting å tenke på også, slik at fremdriften i prosjektet ikke ble satt på hold.

Jan Erik: Ikke så mye å si om denne sprinten. Vi fikk gjort det vi skulle, nemlig å planlegge sprintene videre i prosjektet. Denne sprinten ble kort, da det var eksamen og noen hadde andre innleveringer. Samtidig som jeg skulle ønske vi hadde fått gjort mer i denne sprinten.



Oppsummering: Userstories som ble fullført ila sprint 6



ID	Title	Workflow Step	Priority	Assigned To	Original Estim	Remaining Es
High	1 Items, 10 Hours Worked, **0 Hours Remaining**, 100% Complete					
188	Planlegging: Dele opp flere user stories	Completed	High		10 hrs	0

5. Konklusjon

Dette var en minisprint som var nødvendig ble gjennomført for fremdrift i prosjektet.

Debrief: Sprint 7

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	01.04.2013	AO
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	

Innholdsfortegnelse

1. Innledning

I sprint 7 har vi hovedsakelig jobbet med danseinstruktøren, bevegelsesdeteksjonssystem, prestasjoner og musikk analysering. Det har vært viktig for oss å få på plass noe av de viktige tingene som statistikk og musikk. Vi har satt flere folk på musikk biten for å komme i mål med noen som vi kan bruke. Det har blitt arbeidet mye med å få på plass instruktøren slik at spillet kan hente bevegelser ut ifra hvilken dans som spiller. Meny-systemet har blitt bearbeidet slik at det skal være enklere å forstå og bruke.

2. Det som gikk bra

Gruppen jobber generelt godt sammen, trenger man hjelp er det ikke noe problem å få hjelp. Vi begynner å få kontroll på de tingene vi har igjen å gjøre før innlevering.

3. Det som ikke gikk bra

Det har vært noen problemer med oppgaver som trenger en del samarbeid, da oppgaver må bli satt på vent for å kunne fokusere mer på dem.

4. Evaluering

Dagfinn: Denne sprinten har jeg jobbet med å få på plass prestasjonssystemet, jeg tok utgangspunkt i rammeverket "Progress" og la til støtte for pop-ups. Når jeg var ferdig med dette kom det fram at Henrik hadde kjørt seg fast på musikk analyse delen og ønsket å jobbe med noe annet en stund, så jeg hoppet inn som resurs 2 på musikk-delen. Dette viste seg å være en svært omfattende oppgave som blant annet krevde en del kunnskaper om signalbehandling. Jeg prøvde å finne ut om noen hadde laget og gitt ut noe som gjorde nettopp dette, og fant også et svært lovende prosjekt som heter BeatRoot som er programmert i Java, men det ble for mye å sette seg inn i på for kort tid.

Jeg kommer til å jobbe videre med dette i neste sprint.

Henrik:
I denne sprinten jobbet jeg lenge med å få hentet ut beats i musikkfilen jeg jobbet med, jeg fikk til å lese av spekteret. men det å designe et intervall med å stadig hente ut den sterkeste samplen slet jeg med å få til, så jeg stanget hodet i veggen lenge. Da kom Dagfinn til unnsetning og hjalp meg, han tok delvis over oppgaven. Både han og jeg lese mange forskningsdokumenter på område beat lokasjon, så gjorde han et gjennombrudd som holder til prosjektet. Jeg hoppet så over til Zana og hjalp han med dansegeneratoren og deteksjon av bevegelser.

Are: Zana og jeg har jobbet mye sammen med denne sprinten, med i hovedsak danseinstruktøren og få dybde i gridden. Dette har for det meste gått veldig bra. Jeg har også gjort en del forandringer på dansegeneratoren. denne sprinten har vært viktig for måten dansebevegelsene blir tolket på, jeg klarte å kutte en hel klasse med litt smartere lagring

Marcus: I denne sprinten har mesteparten av tiden min blitt brukt til statistikk. Research og storyboarding tok en del timer. Jeg brukte mye tid på å sette meg inn i NGUI og etterhvert fikk jeg laget stolpediagrammer. Det er nå mulig å presentere statistikk fra dansene i form a stolpediagrammer. Jeg brukte mye tid på å få laget linjediagrammer for å presentere statistikk. Men fant etterhvert ut at dette kom til å ta veldig lang tid, og det kom ikke til og verken se eller bli sånn som planlagt. Dette ble lagt på is, ettersom det er viktigere ting å bruke tiden på frem mot tredje presentasjon.

Samarbeidet innad i gruppen har vært upåklagelig, vi har begynt å jobbe flere timer på skolen, ettersom det bare er noen uker igjen. Det virker som de viktigste funksjonene kommer på plass i prosjektet.

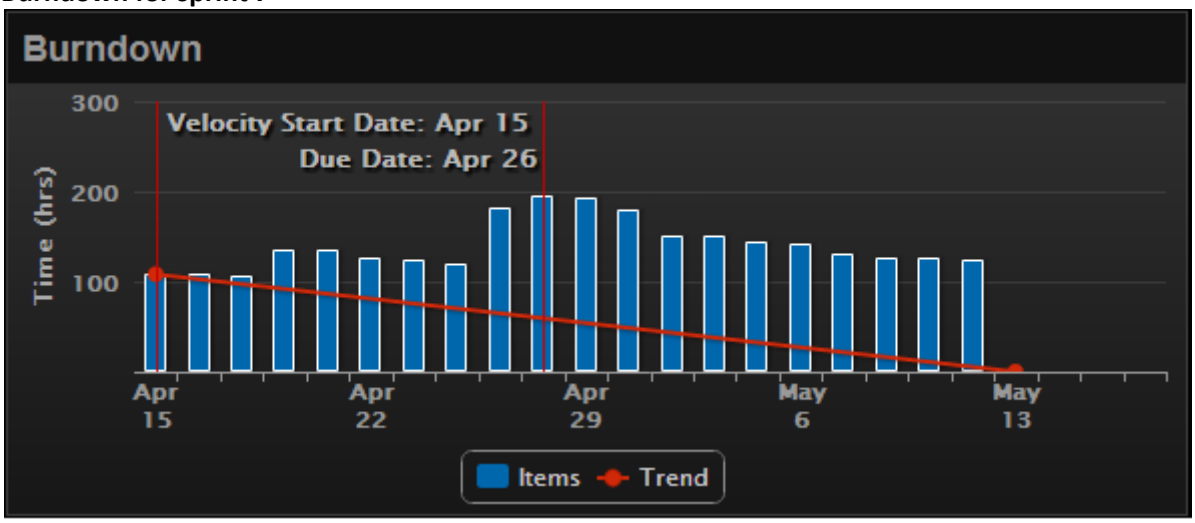
Zana: Jeg og Are fikk på plass mye i løpet av denne sprinten, slik at vi fikk en kommunikasjon mellom klassene våre slik at bevegelser kunne hentes og initialiseres i danserommet. Jeg synes gruppen har jobbet godt i denne sprinten og har vært behjelpelige. Som et resultat av dette har også samarbeidet vært ganske bra og vi har et godt miljø innad i gruppa.

Jan Erik: I denne sprinten har mesteparten av min tid gått med på å oppdatere layouten i menyen til metrostyle. I forbindelse med dette gikk det mye tid på å finne gode ikoner som kunne bli brukt på "flisene". Samtidig har jeg brukt en del tid på å få til et gameobjekt som ikke blir ødelagt mellom scenene i spillet. Samarbeidet har vært bra og folk har hjulpet der det har vært nødvendig.

Oppsummering: Userstories som ble fullført ila sprinten

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
182	Funksjon: Statistikk - Linjediagram	Medium	Marcus Jernberg	Rejected	5 hrs	0 hrs	8 hrs
138	AudioVisualizer	High	Henrik Olsson	Completed	20 hrs	0 hrs	40 hrs
169	Funksjon: Dans - Grid - Dybde	Medium	Zana Ali	Completed	1 hrs	0 hrs	30 min
174	Funksjon: Statistikk - Storyboard	Medium	Marcus Jernberg	Completed	3 hrs	0 hrs	3.5 hrs
175	Funksjon: Prestasjoner - Implementere rammeverk	Medium	Dagfinn Kjærnet	Completed	5 hrs	0 hrs	5 hrs
178	Funksjon: Prestasjoner - Legge til pop-up	Medium	Dagfinn Kjærnet	Completed	10 hrs	0 hrs	10 hrs
181	Funksjon: Statistikk - Stolpediagram	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	11 hrs
183	Arkitektur: Meny - Storyboard v2 - bruker	High	Jan Erik Helskog	Completed	6 hrs	0 hrs	10 hrs
186	Arkitektur: Meny - Storyboard v2 - InGame	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	6 hrs
189	Dans - Grid - Dybde - Ressurs #2	Medium	Are Oven	Completed	1 hrs	0 min	30 min
190	Funksjon: Statistikk - Research	Medium	Marcus Jernberg	Completed	3 hrs	0 hrs	4 hrs
194	Dokumentasjon: Code Review	High	Jan Erik Helskog	Completed	5 hrs	0 hrs	5 hrs
110	Funksjon: Dansescene - Instruktør	Medium	Are Oven	In Progress	80 hrs	52 hrs	28 hrs
192	Funksjon: Musikkanalyse, resurs 2	High	Dagfinn Kjærnet	In Progress	15 hrs	5 hrs	0
193	Funksjon: Dansescene - Instruktør - Ressurs #2	Medium	Zana Ali	In Progress	80 hrs	69 hrs	11 hrs
145	Funksjon: Audio play 'n show	High	Henrik Olsson	New Request	20 hrs	20 hrs	0 hrs
185	Arkitektur: Meny - Storyboard v2 - Admin	High	0	New Request	5 hrs	5 hrs	0
191	Funksjon: Musikk mappe	High	Henrik Olsson	New Request	10 hrs	10 hrs	0

Burndown for sprint 7



5. Konklusjon

Det er fremdeles mye igjen som må gjøres før vi er helt i mål, vi har jobbet godt og det må vi må forsette med den siste tiden som er igjen av hovedprosjektet.

Sprintplan for sprint #7

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
110	Funksjon: Dansescene - Instruktør	Medium	Are Oven	In Progress	80 hrs	60 hrs	0
138	AudioVisualizer	High	Henrik Olsson	In Progress	20 hrs	15 hrs	0
145	Funksjon: Audio play 'n show	High	Henrik Olsson	New Request	20 hrs	20 hrs	0
169	Funksjon: Dans - Grid - Dybde	Medium	Zana Ali	New Request	1 hrs	1 hrs	0
174	Funksjon: Statistikk - Storyboard	Medium	Marcus Jernberg	New Request	3 hrs	3 hrs	0
175	Funksjon: Prestasjoner - Implementere rammeverk	Medium	Dagfinn Kjærnet	New Request	5 hrs	5 hrs	0
178	Funksjon: Prestasjoner - Legge til pop-up	Medium	Dagfinn Kjærnet	New Request	10 hrs	10 hrs	0
181	Funksjon: Statistikk - Stolpediagram	Medium	Marcus Jernberg	New Request	5 hrs	5 hrs	0
182	Funksjon: Statistikk - Linjediagram	Medium	Marcus Jernberg	New Request	5 hrs	5 hrs	0
183	Arkitektur: Meny - Storyboard v2 - bruker	High	Jan Erik Helskog	New Request	6 hrs	6 hrs	0
185	Arkitektur: Meny - Storyboard v2 - Admin	High	0	New Request	5 hrs	5 hrs	0
186	Arkitektur: Meny - Storyboard v2 - InGame	High	Jan Erik Helskog	New Request	5 hrs	5 hrs	0
189	Dans - Grid - Dybde - Ressurs #2	Medium	Are Oven	New Request	1 hrs	1 hrs	0
190	Funksjon: Statistikk - Research	Medium	Marcus Jernberg	New Request	3 hrs	3 hrs	0
191	Funksjon: Musikk mappe	High	Henrik Olsson	New Request	10 hrs	10 hrs	0
192	Funksjon: Musikkanalyse, resurs 2	High	Dagfinn Kjærnet	New Request	15 hrs	15 hrs	0
193	Funksjon: Dansescene - Instruktør - Ressurs #2	Medium	Zana Ali	New Request	80 hrs	80 hrs	0
194	Dokumentasjon: Code Review	High	Jan Erik Helskog	New Request	5 hrs	5 hrs	0

Debrief: Sprint 8

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Skrevet dokumentet	21.05.2013	MJ
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	

Innholdsfortegnelse

1. Innledning

I sprint 8 har vi hovedsakelig jobbet med å gjøre ferdig de funksjonene som lar seg gjøre før prosjektet skal leveres. Dette innebærer ting som menyen, legge til musikk, statistikk, prestasjoner, instruktør, vanskelighetsgrad, scoringsystem, legge til dansebevegelser. Vi har også brukt mye tid merge danserommet, menyen og databasen, for å få en ferdig prototype av spillet som vi kan vise frem på siste presentasjon.

2. Det som gikk bra

Gruppen jobber generelt godt sammen, trenger man hjelp er det ikke noe problem å få hjelp. Det har blitt noen seine kvelder, men alle har stått på og hjulpet hverandre til å bli ferdig med det de har drevet med.

3. Det som ikke gikk bra

Skulle gjerne hatt litt bedre tid til å utvikle videre, siden vi allerede ser mye potensiale vi gjerne skulle jobbet mer med. Noen problemer med databasen når vi bygger spillet, mister noen rettigheter til å lese og skrive til databasen.

4. Evaluering

Dagfinn: I denne sprinten har jeg laget de siste metodene vi trengte i databasehåndteringsklassen(dbHandler) og lagt til rette for database støtte i prestasjonssystemet. Ettersom Henrik ønsket å prøve seg på andre deler av prosjektet en stund så jobbet jeg med musikkanalyse delen og fikk etter hvert på plass en klasse som kunne analysere musikkdataene. Siste innspurten av sprinten gikk med på å sette sammen de forskjellige komponentene vi har jobbet på for å få en gjennomspillbar prototype. Det var veldig moro å se hvordan de forskjellige komponentene jobbet sammen og utførte de oppgavene de var satt til å gjøre og at de faktisk fungerte som deler i det komplette systemet. På tampen av sprinten bestemte vi oss også for å legge til et liten ingame danseinstruktør som skulle vise hvordan man utførte bevegelsene som vistes på skjermen, jeg tok på meg denne oppgaven da jeg hadde en god plan for hvordan vi kunne få til dette ved å bruke deler av prestasjonssystemet.

Dette har definitivt hvert en av de morsomste, men også mest travle sprintene så langt. Jeg er veldig fornøyd med å se at det gode samarbeidet har gitt resultater.

Henrik: I denne sprinten jobbet vi med siste del av implementasjonen, her stod vi på for å få satt sammen alle modulene som vi hadde utviklet hver for oss. Dagfinn leverte sin del av lydsystemet som skulle hente ut beats, så jeg satt dette subsystemet sammen med dancedetector sammen med Zana, vi måtte bruke en del tid på dette. Samtidig måtte vi lage et system for å synkronisere prossesene mellom griden og beats. Samtidig skjønte vi at vi måtte ha en mulighet for å pause spillet ingame og mulighet til å avbryte. For å kunne sette spillet på pause, dette ble en liten utfordring.

Det var tilfredstillende å se at ting fungerte som det skulle når vi satt det sammen, gruppen jobbet bra sammen.

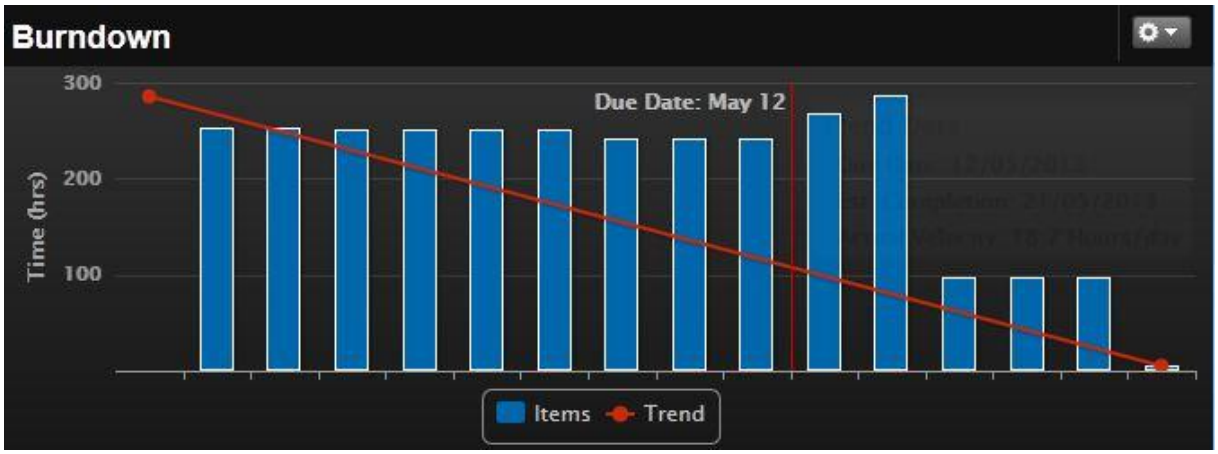
Are: også denne sprinten har Zana og jeg jobbet en del sammen. Jeg jobbet med å legge til bevegelser og få korrekt tolkning av dem. Dansegeneratoren ble helt ferdig slik at den kan tilpasse bevegelsene slik at de passer til det spilleren har valgt i forhold til aktive deler av skjermen.

Marcus: Denne sprinten har gått bra. Jeg har blitt ferdig med user storyene mine. Har også gjort meg behjelpelig for andre der det har vært behov for hjelp. Sammarbeidet fungerer fortsatt bra. Det er fortsatt litt ståpå-vilje igjen hos gruppen, men alle kjenner det skal bli godt å bli ferdig.

Zana: I denne sprinten gjorde jeg meg ferdig med user-stories som omhandler selve implementasjon av spillet, i likhet med resten av gruppa. Instruktør som jeg har jobbet en god del med, sammen med Are ble tidlig ferdig i denne sprinten. Scoringsystem og vanskelighetsgrad er to andre user-stories som ble påbegynt og fullført i denne sprinten noe jeg er fornøyd med. Det var også morsomt å se hvordan spillet så ut, etter at vi satte det sammen. Jeg ble positivt overrasket over resultatet.

Gruppen er klar over at det ikke er lenge igjen før innlevering og presentasjon. Vi er derfor innstilte på å legge inn en formidabel innsats de siste ukene for å oppnå et best mulig resultat.

Jan Erik: I denne sprinter har vi jobbet med de siste funksjonene i spillet. Mot slutten av sprinten jobbet vi med å slå sammen de forskjellige delene. Samarbeidet i gruppen har vært bra, dersom det har oppstått et problem har det ikke vært noe problem å få hjelp fra andre. I løpet av denne sprinten har jeg lagt til muligheten til å byttepassord, stille på vanskelighetsgraden og velge hvilke sang man vil se statistikk over. Sammenslåingen av de forskjellige komponentene gikk overraskende bra og det var gøy å se spillet begynne å ta form.



Oppsummering: Userstories som ble fullført ila sprinten

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
107	Funksjon: Generator - Vanskelighetsgrad	Medium	0	Rejected	0	0 hrs	0
185	Arkitektur: Meny - Storyboard v2 - Admin	High	0	Rejected	5 hrs	0 min	0
110	Funksjon: Dansescene - Instruktør	Medium	Are Oven	Completed	80 hrs	0 hrs	42 hrs
111	Funksjon: Generator - Tilpasse dans	Medium	Are Oven	Completed	20 hrs	0 hrs	12 hrs
136	Arkitektur: Menysystem - Achievments(10)	High	0	Completed	2 hrs	0 hrs	2 hrs
180	Funksjon: Prestasjoner - legge til databasestøtte	Medium	Dagfinn Kjærnet	Completed	10 hrs	0 hrs	10 hrs
191	Funksjon: Musikk mappe	High	Jan Erik Helskog	Completed	10 hrs	0 hrs	10 hrs
192	Funksjon: Musikkanalyse, resurs 2	High	Dagfinn Kjærnet	Completed	15 hrs	0 hrs	15 hrs
193	Funksjon: Dansescene - Instruktør - Ressurs #2	Medium	Zana Ali	Completed	80 hrs	0 hrs	36 hrs
195	Funksjon: DoNotDestroy Unity	Medium	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
196	Support: Dance Generator	Medium	Henrik Olsson	Completed	8 hrs	0 hrs	12 hrs
197	Funksjon: Vanskelighetsgrad - Lengre tid mellom beveg	Medium	Zana Ali	Completed	20 hrs	0 hrs	1 hrs
198	Funksjon: Instruktør - Visuell	Medium	Marcus Jernberg	Completed	10 hrs	0 hrs	5.5 hrs
199	Testing: Dansegenerator	Medium	Henrik Olsson	Completed	4 hrs	0 hrs	0
200	Merge: Dancegenerator & AudioImport	High	Henrik Olsson	Completed	8 hrs	0 hrs	8 hrs
201	Funksjon: Meny - Change Password scene	High	Jan Erik Helskog	Completed	3 hrs	0 hrs	3 hrs
202	Funksjon: Meny - Startupscreen scene	High	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
203	Funksjon: Meny - Tilpasse scene	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
204	Meny: Oppdatere til metro	High	Jan Erik Helskog	Completed	6 hrs	0 hrs	6 hrs
205	Funksjon: Meny - Statistikkoversikt scene	High	Jan Erik Helskog	Completed	2 hrs	0 hrs	2 hrs
206	Funksjon: Dansegenerator - Opprette bevegelsene	Medium	Are Oven	Completed	10 hrs	0 hrs	6 hrs
207	Funksjon: Database - Flere funksjoner	High	Dagfinn Kjærnet	Completed	15 hrs	0 hrs	15 hrs
208	Ingame pausemeny	Medium	Henrik Olsson	Completed	6 hrs	0 hrs	6 hrs
209	Funksjon: Scoringsystem	Medium	Zana Ali	Completed	20 hrs	0 hrs	14 hrs
210	Funksjon: Scoringsystem - Ressurs#2	Medium	Are Oven	Completed	20 hrs	0 hrs	4 hrs
211	Implementasjon: Merge Statistikk og Database	Medium	Marcus Jernberg	Completed	6 hrs	0 hrs	4.5 hrs
212	Dokumentasjon: Dokumentere kode	Medium	Dagfinn Kjærnet	Completed	10 hrs	0 hrs	10 hrs
213	Implementasjon: Sammensetting av komponenter	High	Dagfinn Kjærnet	Completed	4 hrs	0 hrs	4 hrs
214	Implementasjon: Sammensetting av komponenter	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	5 hrs
216	Dokumentasjon: Dokumentere kode	Medium	Jan Erik Helskog	Completed	8 hrs	0 hrs	8 hrs
217	Funksjon: Visuell ingame danse instruksjon.	Low	Dagfinn Kjærnet	Completed	2 hrs	0 hrs	2 hrs
0	0	0	0	0	0	0	0

5. Konklusjon

Prosjektet begynner å nærme seg slutten. Nå gjenstår det mye dokumentasjon, og finpussing. Det er viktig å holde tunga rett i munnen når vi skal levere dokumentene. Vi må nok også bruke mye tid på å lage presentasjonen og øving.

Sprintplan for sprint #8

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
110	Funksjon: Dansescene - Instruktør	Medium	Are Oven	Completed	80 hrs	0 hrs	42 hrs
191	Funksjon: Musikk mappe	High	Jan Erik Helskog	Completed	10 hrs	0 hrs	10 hrs
192	Funksjon: Musikkanalyse, resurs 2	High	Dagfinn Kjærnet	Completed	15 hrs	0 hrs	15 hrs
193	Funksjon: Dansescene - Instruktør - Ressurs #2	Medium	Zana Ali	Completed	80 hrs	44 hrs	36 hrs
195	Funksjon: DoNotDestroy Unity	Medium	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
196	Support: Dance Generator	Medium	Henrik Olsson	Completed	8 hrs	0 hrs	12 hrs
199	Testing: Dansegenerator	Medium	Henrik Olsson	Completed	4 hrs	0 hrs	0
202	Funksjon: Meny - Startupscreen scene	High	Jan Erik Helskog	Completed	1 hrs	0 hrs	1 hrs
203	Funksjon: Meny - Tilpasse scene	High	Jan Erik Helskog	Completed	4 hrs	0 hrs	4 hrs
204	Meny: Oppdatere til metro	High	Jan Erik Helskog	Completed	6 hrs	0 hrs	6 hrs
180	Funksjon: Prestasjoner - legge til databasestøtte	Medium	Dagfinn Kjærnet	In Progress	10 hrs	5 hrs	5 hrs
198	Funksjon: Instruktør - Visuell	Medium	Marcus Jernberg	In Progress	10 hrs	7.5 hrs	2.5 hrs
200	Merge: Dancegenerator & AudioImport	High	Henrik Olsson	In Progress	8 hrs	8 hrs	0
201	Funksjon: Meny - Change Password scene	High	Jan Erik Helskog	In Progress	3 hrs	1 hrs	2 hrs
205	Funksjon: Meny - Statistikkoversikt scene	High	Jan Erik Helskog	In Progress	2 hrs	1 hrs	1 hrs
107	Funksjon: Generator - Vanskelighetsgrad	Medium	0	New Request	200 hrs	200 hrs	0
185	Arkitektur: Meny - Storyboard v2 - Admin	High	0	New Request	5 hrs	5 hrs	0
197	Funksjon: Vanskelighetsgrad - Lengre tid mellom be	Medium	0	New Request	20 hrs	20 hrs	0

Sprint 9 debrief

Sunnaas sykehus HF

Mektron

HØGSKOLEN
i Buskerud

HiBu studentprosjekt 2013

.....	.....
.....	.....
.....	.....

Revisjonshistorikk

Revisjon #	Endringer	Dato	Skribent
1.0	Opprettet	24.05.13	HO
2.0			
3.0			
4.0			
5.0			
6.0			

Kontrollert av:	ZA

Innholdsfortegnelse

1. Innledning

I Spring 9 har vi i hovedsak jobbet med å få på plass dokumentasjonen, da dette er ganske mye så har vi satt av hele sprinten til det. Vi har også tatt og gjort siste finpuss på produktet samtidig som vi har kjørt tester. Denne sprint debriefen blir skrevet tidlig i sprinten da den skal være en del av dokumentasjonen som leveres om kort tid. I skrivende stund jobber gruppen hardt med å bli ferdig.

2. Det som gikk bra

Alle står på og ofrer mye fritid for å komme i mål med prosjektet. Prosjektperioden har vært krevende og sene kvelder ville det ha blitt uansett da vi ønsker å få produktet så pent som overhodet mulig. Det er ingen tvil om at alle er like ivrige på å komme i mål på best mulig måte. Nå gjelder det å holde koken frem til dokumentasjonen skal leveres, samtidig som gjør oss klare til siste presentasjon.

3. Det som ikke gikk bra

Det er tydelig at gruppemedlemene kan bli litt lei av hverandre etter å ha sittet mange timer i strekk i et trangt rom, men dette er noe vi hadde regnet med.

4. Evaluering

Dagfinn: Denne sprinten har i stor grad bestått i ren dokumentering og småfiksing på prototypen. Det skal bli godt å kunne levere fra seg dokumentasjonen og kunne fokusere på presentasjon 3. Denne sprinten har vi hatt noen svært krevende dager hvor vi har jobbet i mange timer, men gruppen har holdt motet oppe og hjulpet hverandre godt slik at vi kom i mål.

Henrik: Det er tilfredstillende å se hva vi har fått til, jeg ser frem til å bli ferdige med prosjektet og ikke minst presentere arbeidet vi er veldig stolte av. Jeg føler at jeg har fått skrevet mye og hvert utrolig effektiv de siste dagene. Godt å se at alle er iverige på å gjøre dokumentasjonen fyldig. Det er så klart mye mer man skulle hatt tid til, men det er problemet med ingeniøryrket. En gang må man bestemme seg for at produktet er godt nok.

Are: Vi har brukt denne sprinten til å skrive og oppdatere alle dokumentene vi skal levere inn. Det har vært mange lange og slitsomme dager. Zana og jeg jobbet også med å få på plass dynamiske vanskelighetsgrad, det gikk veldig greit. Dette har vært siste innspurt før presentasjonen og selv med lange og slitsomme dager har gruppen jobbet godt. Nå skal det bli godt å få skiftet fokus over til presentasjonen.

Marcus: I denne sprinten har timene gått med til dokumentering og bug-fixes. Jeg har personlig brukt litt tid på å få i stand plakaten til LetsDance, men harogså brukt tid på printing og dokumentering. Alle på gruppen kjenner at det skal bli godt å bli ferdig nå, og endelig kunne levere LetsDance.

Zana: Vi har jobbet veldig hardt nå mot slutten av prosjektet. Vi er alle klare over hva som står på spill og derfor har vi også tilbrakt mye tid til prosjektet. Det har blitt endel sene og intense kvelder. Men gruppen står sammen og støtter hverandre og står på, moralen bra er fortsatt bra til tross for mye arbeid i løpet av kort tid. Jeg er fornøyd med at jeg og Are fikk på plass en enkel dynamisk vanskelighetsgrad i denne sprinten. Det gjelder å fortsette å stå på mot slutten slik at vi kommer i mål med et resultat vi vil være fornøyde med.

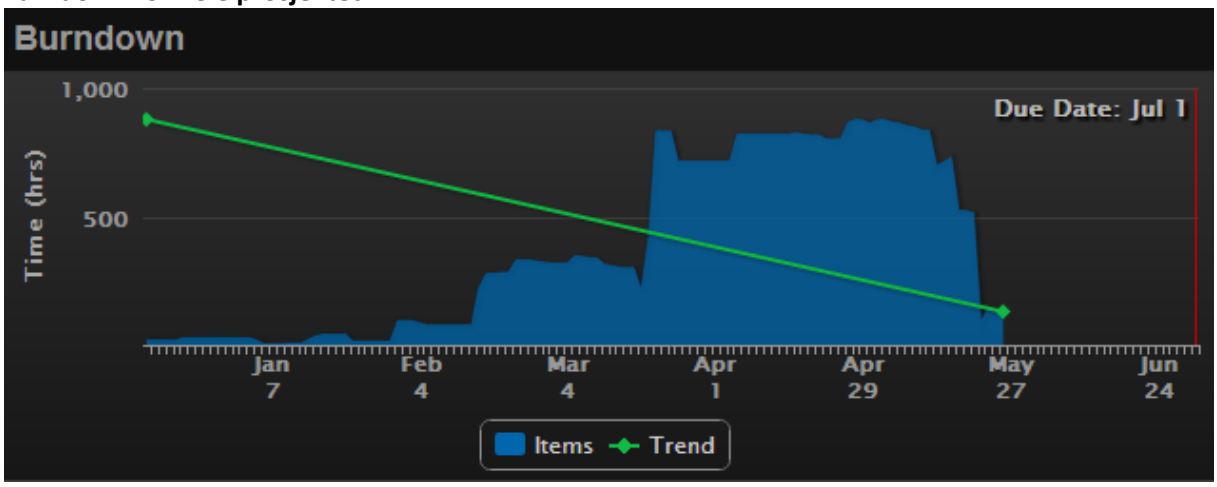
Jan Erik: I denne sprinten har vi jobbet med dokumentasjon og bug fiksing på prototypen. Det skal bli godt å få levert dokumentasjonen. Det har blitt mange lange og slitsomme dager denne sprinten. I denne sprinten har jeg skrevet mye på designdokumentasjonen og satt det sammen. Jeg har også fikset mange små bugs i menyen. Gruppen har jobbet bra i denne sprinten og det skal bli godt å legge fra seg dokumentasjonen. Slik at vi kan konsentrere oss om presentasjon 3.

Oppsummering: Userstories som ble fullført ila sprinten

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
107	Funksjon: Dynamisk vanskelighetsgrad	Medium	Are Oven	Completed	4 hrs	0 hrs	2 hrs
108	Funksjon: Dansescene - Utføre Achivements	Medium	0	Completed	4 hrs	0	4 hrs
130	Dokumentasjon: Audio Import	High	Henrik Olsson	Completed	2 hrs	0 hrs	5 hrs
187	GDD: Oppdatere - legge til animerte bevegelser	Medium	Henrik Olsson	Completed	2 hrs	0 hrs	2 hrs
215	Dokument: Videre utvikling	High	Henrik Olsson	Completed	12 hrs	0 hrs	10 hrs
218	Dokumentasjon: Opprydning i OnTime	Medium	Dagfinn Kjærnet	Completed	2 hrs	0	2 hrs
219	Dokumentasjon: Designdokument	Medium	Zana Ali	Completed	15 hrs	0 hrs	10 hrs
220	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Dagfinn Kjærnet	Completed	16 hrs	0 hrs	16 hrs
221	Dokumentasjon: Oppdatere dokumentasjon	Medium	Jan Erik Helskog	Completed	20 hrs	0	26 hrs
222	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	3 hrs
223	Dokumentasjon: oppdatere dokumentasjon	Medium	Are Oven	Completed	6 hrs	0 hrs	0
225	Dokumentasjon: Sluttrapport	Medium	Henrik Olsson	Completed	20 hrs	0	20 hrs
227	Dokumentasjon: oversiktsdokument	Medium	Marcus Jernberg	Completed	2 hrs	0 hrs	1 hrs
228	Ordne tekst, info, logoer osv til plakat	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	5 hrs
229	Dokumentasjon: Oppdatere	Medium	Henrik Olsson	Completed	8 hrs	0	10 hrs
231	Brukermanual	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	5 hrs
232	Sprint 9 debrief	Medium	Henrik Olsson	Completed	2 hrs	0 hrs	1.5 hrs

Burndown for sprint 9

Burndown for hele prosjektet



Figuren viser trenden over fullførte arbeidsoppgaver og bruk av tid. Vi ser at trenden er grønn, riktignok gir ikke burndown charten en nøyaktig presentasjon av antall timer vi har stått på. Vi ser uansett at prosjektet nermer seg slutten og vi har oversikt over alle timene folk har stått på og jobbet.

5. Konklusjon

Det er ingen tvil om at vi er lei av sene kvelder og hardt arbeid, men vi håper at dette vil lønne oss godt senere. Alle er flinke til å ta på seg oppgaver og lite tid blir surret bort. Alle står på og er flinke.

Sprintplan for sprint #9

ID	Tittel	Prioritet	Ressurs	Status	Estimert tid	Gjenstående tid	Reel tid
107	Funksjon: Dynamisk vanskelighetsgrad	Medium	Are Oven	Completed	2 hrs	0 hrs	2 hrs
130	Dokumentasjon: Audio Import	High	Henrik Olsson	Completed	2 hrs	0 hrs	5 hrs
215	Dokument: Videre utvikling	High	Henrik Olsson	Completed	12 hrs	0 hrs	8 hrs
219	Dokumentasjon: Designdokument	Medium	Zana Ali	Completed	15 hrs	0 hrs	10 hrs
228	Ordne tekst, info, logoer osv til plakat	Medium	Marcus Jernberg	Completed	5 hrs	0 hrs	5 hrs
218	Dokumentasjon: Opprydning i OnTime	Medium	Dagfinn Kjærnet	In Progress	3 hrs	1 hrs	2 hrs
220	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Dagfinn Kjærnet	In Progress	16 hrs	4 hrs	12 hrs
221	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Jan Erik Helskog	In Progress	12 hrs	3 hrs	13 hrs
222	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Marcus Jernberg	In Progress	12 hrs	9 hrs	3 hrs
223	Dokumentasjon: Oppdatere designdokumentasjon	Medium	Are Oven	In Progress	12 hrs	12 hrs	0
229	Dokumentasjon: Oppdatere	Medium	Henrik Olsson	In Progress	8 hrs	4 hrs	0 hrs
231	Brukermanual	Medium	Marcus Jernberg	In Progress	5 hrs	0 hrs	5 hrs
108	Funksjon: Dansescene - Utføre Achivements	Medium	0	New Request	4 hrs	4 hrs	0
187	GDD: Oppdatere - legge til animerte bevegelser	Medium	0	New Request	2 hrs	2 hrs	0
225	Dokumentasjon: Sluttrapport	Medium	Henrik Olsson	New Request	20 hrs	20 hrs	0
227	Dokumentasjon: oversiktsdokument	Medium	0	New Request	2 hrs	2 hrs	0