FMH606 Master's Thesis 2024

Industrial IT and Automation

Decentralized application (dApp) on the VechainThor blockchain for business processes

Iñigo Izaguirre Diaz

Faculty of Technology, Natural sciences and Maritime Sciences
Campus Porsgrunn

University of South-Eastern Norway

www.usn.no

Course: FMH606 Master's Thesis, 2023

Title: Decentralized application (dApp) on the VechainThor blockchain for business processes

Number of pages: 55

Keywords: VechainThor Blockchain, Decentralized Application, Traceability, Distributed Ledger Technology, Smart Contract, Consensus mechanism, Authentication, Internet of Things, Authority Masternodes, Blockchain Integration in Business Processes.

Student:	Iñigo Izaguirre Diaz 258803
Supervisor:	Leila Ben Saad
External partner:	**N/A**

Summary:

This master thesis explores the development and implementation of a decentralized application (dApp) on the VechainThor blockchain, with a focus on enhancing traceability and transparency in business processes. Beginning with an overview of distributed ledger technology, including blockchain, IOTA, and Vechain, the report compares these technologies to identify their strengths and limitations. The study delves into the features of the VechainThor blockchain, including its dual token methodology and blockchain architecture. A detailed analysis of the distributed application (dApp) is provided, covering requirements gathering, system architecture, and integration with the VechainThor blockchain. The implementation phase involves smart contract development, front-end development, and testing, with a focus on ensuring security and performance. The discussion section presents the advantages and disadvantages of the implemented solution, particularly in terms of security considerations. The challenges of building such type of applications. Finally, the conclusion summarizes the key findings, highlighting the accomplishments of the project and suggesting future improvements.

Preface

I want to express my gratitude to the Faculty of Technology, Natural Sciences, and Maritime Sciences at Campus Porsgrunn of the University of South-Eastern Norway for providing me with the opportunity to undertake this master's thesis.

This thesis delves into the development of a decentralized application (dApp) on the VechainThor blockchain, aimed at tracking and authenticating product deliveries in business processes.

Special thanks to my supervisor, Leila Ben Saad, for her guidance and support throughout this project.

I also want to express my appreciation to my family and friends for their encouragement and understanding.

Porsgrunn, 2024

Iñigo Izaguirre Diaz

Contents

List of figures

List of tables

Nomenclature

ABI: Application Binary Interface

COO: Coordinator node

CSS: Cascading style sheets

DAG: Directed Acyclic Graphs

dApp: Decentralized Application

DeFi: Decentralized Finance

DoS: Denial of Service

DLT: Distributed Ledger Technology

HTML: Hyper Text Markup Language

IoT: Internet of Things

PoA: Proof of Authority

P2P: Peer to peer

PoS: Proof of Stake

PoW: Proof of Work

SSD: System Sequence Diagram

UI: User Interface

UP Unified Process

UX: User Experience

VET: Vechain token

VTHO: VechainThor token

1 Introduction

In today's business world it is crucial to follow track and ensure security of the supply chain management and business processes. The traditional methods to record all the transactions and processes are often limited in terms of traceability and security with some inefficiencies and vulnerabilities. The emergence of the blockchain technology offers a new way of improving the traditional methods for recording the transactions in a more efficient and secure way.

The aim of this report is to explore a blockchain technology named Vechain [1] and deep into decentralized applications to develop a dApp for business processes. Through an exhaustive analysis, this report aims to identify the benefits of blockchain technology in the context of supply chain management and business processes, with a particular focus on Vechain. Different distributed ledger technologies will be analyzed and compared, and a solution will be proposed based on the Vechain technology.

Considering the existing potential of Vechain and decentralized applications, this report aims to contribute to the advancement of blockchain technology in the business world, offering a solution to the existing challenge of tracking all the deliveries in an increasing globalized world and driving progress in the field of supply chain management and business processes.

1.1 Objectives

The main objective has been defined to get a good scope of the project and from that main objective other sub-objectives have defined.

Main objective:

- Design and develop a decentralized application (dApp) on the VechainThor blockchain oriented towards business processes.

Sub-objectives:

- **Provide a good overview of distributed ledger technology:** This objective will ensure that the reader has a good understanding of the basic concepts necessary for understanding the development of the project.
- **Make an exploration of different distributed ledgers:** The exploration will deep into different technologies and will compare the different technologies.
- **Analyze VechainThor blockchain and its characteristics:** The main technology will be analyzed for understanding the selection of it for developing the decentralized application.
- **Conduct a design process of the decentralized application:** This is a key step in order to get a good development of the application. The definition of requirements and preparation for filling that requirement is a key factor in the development phase.
- **Learn how to develop a decentralized application:** Learning the necessary tools for developing the application will be a good objective to get a better decentralized app.
- **Be able of implementing traceability in the developed application:** Traceability is a critical aspect of supply chain management and business processes. Implementing

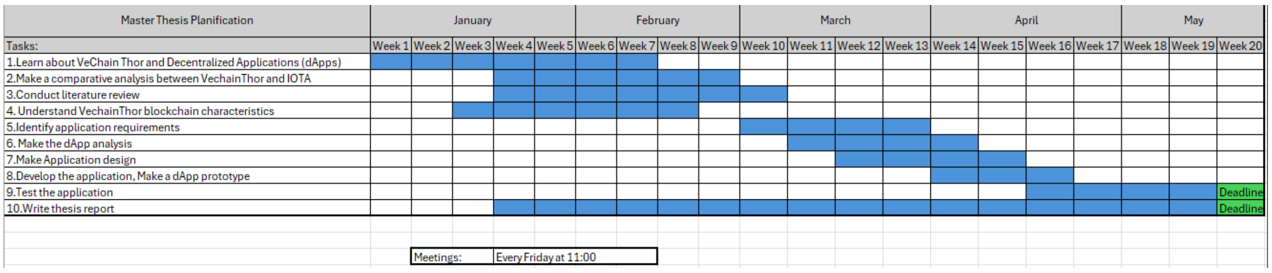
traceability features in your dApp will enhance its utility and value proposition for businesses.

The project description of this master thesis is included in **¡Error! No se encuentra el origen de la referencia.**.

1.2 Planification

A planification of the project has been carried out to define tasks from the previously defined main objective and sub-objectives. With the tasks defined, a Gantt diagram has been created to make a good planning of the master thesis. The time spend for each time has been estimated by the difficulty estimated for each task. In the Figure 1, the developed Gantt diagram of this project is presented.

During the project, the Gantt diagram has been used as reference for checking the process of the master project and define a deadline for all the previously define tasks.



Master Thesis Planification	January					February				March				April				May		
Tasks:	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15	Week 16	Week 17	Week 18	Week 19	Week 20
1.Learn about VeChain Thor and Decentralized Applications (dApps)	🟦	🟦	🟦	🟦	🟦	🟦	🟦													
2.Make a comparative analysis between VechainThor and IOTA				🟦	🟦	🟦	🟦	🟦	🟦											
3.Conduct literature review				🟦	🟦	🟦	🟦	🟦	🟦	🟦										
4. Understand VechainThor blockchain characteristics			🟦	🟦	🟦	🟦	🟦	🟦												
5.Identify application requirements										🟦	🟦	🟦	🟦							
6. Make the dApp analysis											🟦	🟦	🟦	🟦						
7.Make Application design												🟦	🟦	🟦	🟦					
8.Develop the application, Make a dApp prototype														🟦	🟦	🟦				
9.Test the application																🟦	🟦	🟦	🟦	Deadline
10.Write thesis report				🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	Deadline

Meetings:	Every Friday at 11:00

Figure 1: Gantt diagram for the master thesis.

1.3 Report Structure

This report has been structured in different chapters. First, an introduction chapter is given where the project problem and different tasks are described, and objectives are defined from the main task. In the second chapter, a review of distributed ledger technology will be given to get the basic understanding of the project. At the same time different specific DLT technologies will be explained and compared. In the third chapter, VechainThor blockchain will be explained more deeply. After explaining the bases for understanding the thesis, the fourth chapter of the thesis will focus on decentralized applications and the understanding of how decentralized applications work.

With the previous concepts explained and analyzed, the fifth chapter will focus on making the first design and analysis of the software. For that, the requirements first are collected and on this chapter, different diagrams will be created in order to have a good understanding of the project and a good software designing. The sixth chapter will focus on the software development and the tools used during the development phase. In this phase the designed application will be showed, and some results are displayed.

In the seventh chapter a discussion has been made from the results obtained from the project. In the last chapter some conclusions are defined from the previous discussion and the whole project.

2 Distributed Ledger Technology

In this chapter, the basics about the distributed ledgers, blockchain [2] and IOTA [3] protocols will be initially discussed and compared in order to see all the differences and get a good understanding of the concept. Through the exploration of the concepts, the differences and main characteristics of each technology will be described. At the same time, the blockchain used in this project will be described by providing characteristics and concept that allow a good guidance to understand the developed project.

2.1 Distributed Ledger Technology

Distributed Ledger Technology (DLT) can be defined as a digital system or database that registers active transactions where the details are being registered in multiple locations at the same time [4]. The main difference with traditional databases is that in distributed ledgers there is no central data or store functionality so the administration is not centralized.

The name Distributed Ledgers Technology refers to the technological system or protocol that authorize the simultaneous access, validation and update of the registrations that are made in the DLT. This technology works in a computer network with multiple nodes. All the nodes from the computer network receive a copy of the ledgers in order to decentralize the system. At the same time every node can view, modify, and verify the ledger that makes the system more trustable [4]. In Figure 2, a scheme representing all the individuals in the distributed ledger are connected is presented.

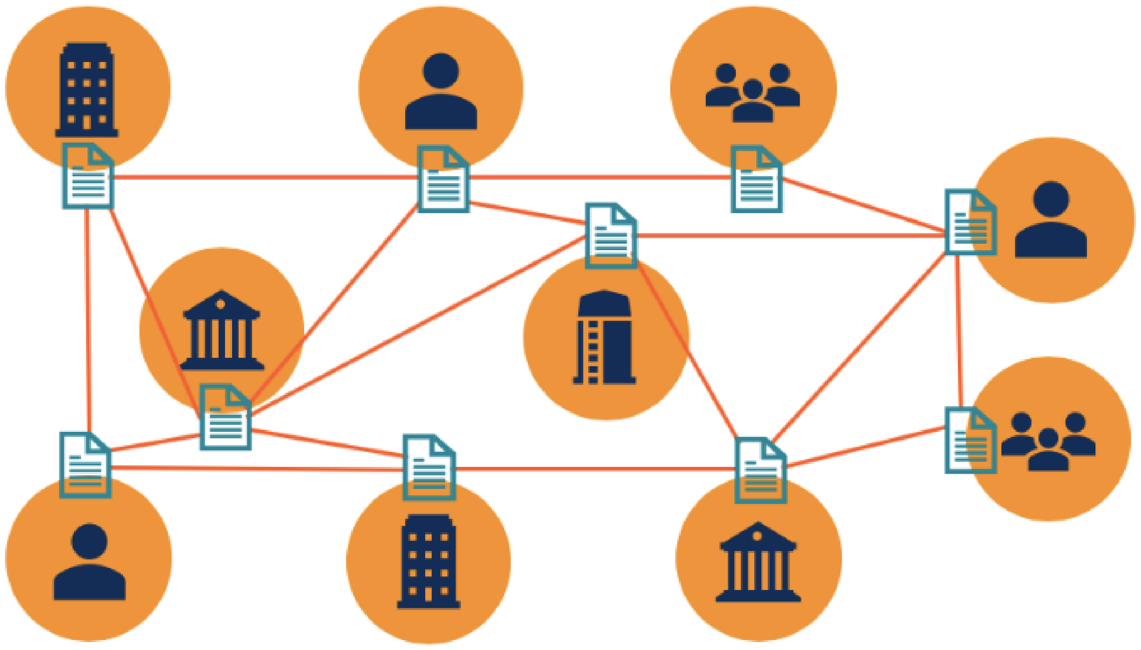


Figure 2: Distributed Ledger Technology scheme [4].

Nodes in distributed ledgers process and verify every item. By doing this, a registration of each item is conducted, and a veracity agreement is being constructed. The data stored in a distributed ledger can vary from static or dynamic data and also can be used as registry or even with financial transactions. Blockchain is using the distributed ledger technology [5].

The increase in the interest of distributed ledgers started with the introduction of cryptocurrencies like Bitcoin in 2009 with the usage of blockchain technology. It has been proved that the technology behaves correctly and is able of scaling while being secure at the same time [6].

Distributed Ledger Technology consists of several features [4]:

- Nodes: Network nodes are individual devices or computer connected needed in order to accept the transactions.
- Consensus mechanism: This can be defined as a protocol or set of rules used in a decentralized system in order to get an agreement on the validity of a transaction between participants without the need of an authority.
- Smart contracts: These contracts are executed when certain requirements are fulfilled without the need of any individual.
- Secret code: It is the way to maintain safety in the DLT. Only the people with the right secret code can access the transaction data.
- Unchangeable notebook: whatever are written in the distributed ledger are not changeable. This avoids any intention of cheating.

2.2 Blockchain

The blockchain [2] technology can be defined as a distributed, shared, and unchangeable database. The main purpose of blockchain is to facilitate the process of saving transactions and tracking records in a business network. The transaction or record can be tangible like vehicle, flat, land, cash or intangible like patents, branding, or intellectual property. Any item with virtual value can be used as record in a blockchain network. Bitcoin in 2009 was the first ever cryptocurrency to use in a blockchain environment [6].

The blockchain network uses the previously explained distributed ledger technology. All the individuals from the network can access to the distributed ledger and all the records saved previously in the network. Transactions are saved just once simultaneously avoiding all the duplication effort required in other technologies. There is no option to modify a transaction once it has been added to the distributed ledger. If an error occurs, a new transaction must be upload to the network, this can be defined as immutable transaction capacity [6].

In all blockchain technologies, an economic model needs to be defined in order to get a good behavior of the system. In this economic model, it is important to define the native coin that will be the primary cryptocurrency. Usually, this one is the fundamental unit of value of the system and serves for various purposes in the blockchain system. The defined native coins are just defined to use in the blockchain network context and commonly. They are used as a way of exchange, store or as a utility token to access or interact with specific blockchain network services [5].

The unit of data in Blockchain network are transactions. The simplest example of a transaction is to transfer tokens from one account to another. Every transaction is added after the previous transaction without the possibility of changing the chain. Each block includes the hash from the previous block [7]. In Figure 3 a visual scheme of the explained structure is presented.

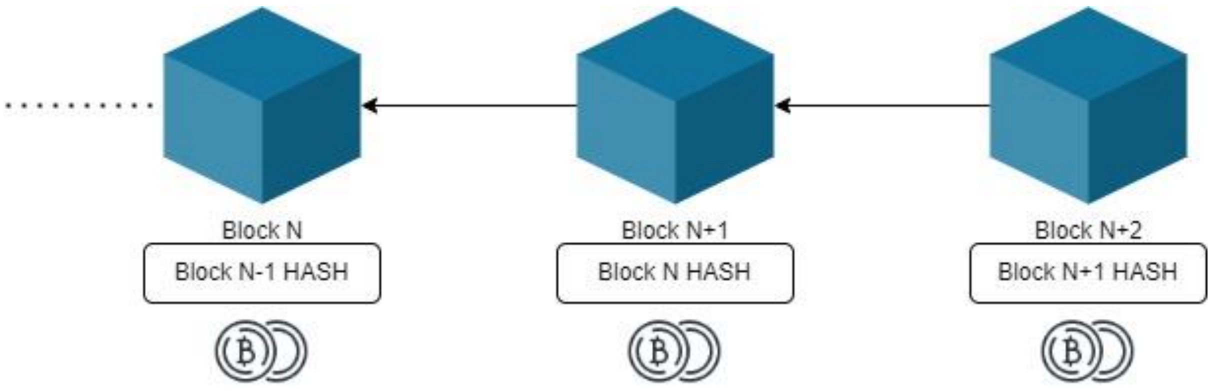


Figure 3: View of Blockchain data structure based [7].

Smart contracts are defined as digital contracts designed to automatically execute when a previously defined conditions are fulfilled. They are used to speed transactions and stored information in the blockchain. This type of contracts avoids the need of intermediaries. The most typical functionality is to automate the execution of an agreement in order to inform all the participants at the same time. Another functionality can be to activate a workflow when a requirement is completes in the smart contract itself [6].

The smart contracts are written in code in the blockchain where simple “if” or “when” statement conditions are defined and utilized. When the conditions are fulfilled, a network of computers execute the action. When the transaction is completed the blockchain, is updated for every individual. The executed transaction is immutable, so it cannot be changed [5]. In Figure 4 a scheme is presented that explains how smart contracts work and the working flow in a general blockchain network.

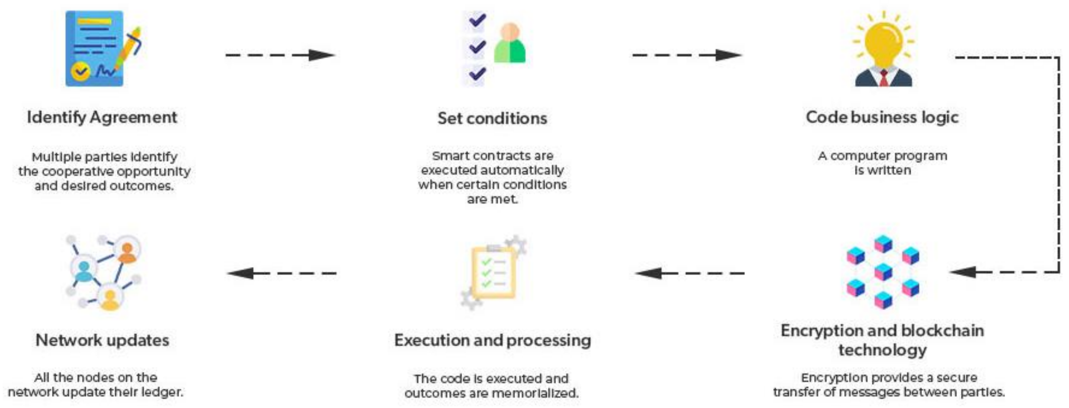

Identify Agreement
Multiple parties identify the cooperative opportunity and desired outcomes.
Set conditions
Smart contracts are executed automatically when certain conditions are met.
Code business logic
A computer program is written
Encryption and blockchain technology
Encryption provides a secure transfer of messages between parties.
Execution and processing
The code is executed and outcomes are memorialized.
Network updates
All the nodes on the network update their ledger.

Figure 4: Smart contracts working scheme [8].

2.3 IOTA

IOTA [3] is an open-source distributed ledger and is designed to enable transferring data and values between humans and machines. Main properties of this distributed ledger are their contactless transactions, safe data, and small number of resources necessary for transactions. This network is able to impulse Internet of Things without requiring a big investment in the infrastructure.

IOTA is not based on blockchain as other cryptocurrencies. In this case, it is based on a data structure named Tangle [9]. Tangle is using a Directed Acyclic Graph (DAG) instead of the blockchain. The DAG can be defined as a graph structure where transactions are linked together in a specific order, without any loops or cycles. The nodes are the transaction executor in the tangle, and they define the ledger for storing transactions in the tangle. The idea of working principle in the tangle is that in order to execute a transaction the user must work to approve other transactions. Then, Tangle uses a proof-of-work (PoW) system for authenticating transactions on a distributed ledger [10]. This way of working ensures that the users that approve one transactions are contributing in the tangle network security. If there is a node that identify that one transaction conflicts with the tangle history the node will not approve the wrong transaction.

2.3.1 Working principle

The steps followed by a node to record a transaction in the tangle can be described as [9]:

1. With the usage of a specific algorithm the node selects two transactions to approve
2. The node checks if the selected transactions have some conflict or no and if they are conflicting the node will not approve the transactions.
3. In order to publish the transaction, the node has to solve a cryptographic puzzle similar to the ones used in other cryptocurrencies. This is accomplished by discovering a nonce, unique number, that, when combined with certain data from the authorized transaction, produces a hash with a specific pattern. In Bitcoin's protocol, for instance, this pattern requires the hash to start with a predetermined number of zeros.

IOTA networks are asynchronous, this means that not all the nodes will see the same set of transactions. At the same time, it is possible in the tangle to have conflicting transactions. But as defined previously, in that situation, the node will decide which one of the conflicting transactions needs to be orphaned.

The nodes in the IOTA network share Tangle and the incoming transactions are validated or added as per the consensus rules. As shown in Figure 5, instead of a group of mining nodes packing transactions inside the block, every new transaction can be a part of the growing tree of transactions in an acyclic way by referencing some existing transaction and thus forming a graph. A new incoming transaction validates the previous ones and helps grow the IOTA network. In this way, many transactions can be validated in parallel providing high throughput and scalability. A special node called Coordinator (COO) helps transactions present in the IOTA ledger attain finality. Since there is no fee for producing a transaction, to protect the ledger from Denial-of-service (DoS) attacks or spamming transactions, a small amount of proof-of-work (PoW) is required which does not take part in the consensus but provides a weight for the transaction to be considered valid [10].

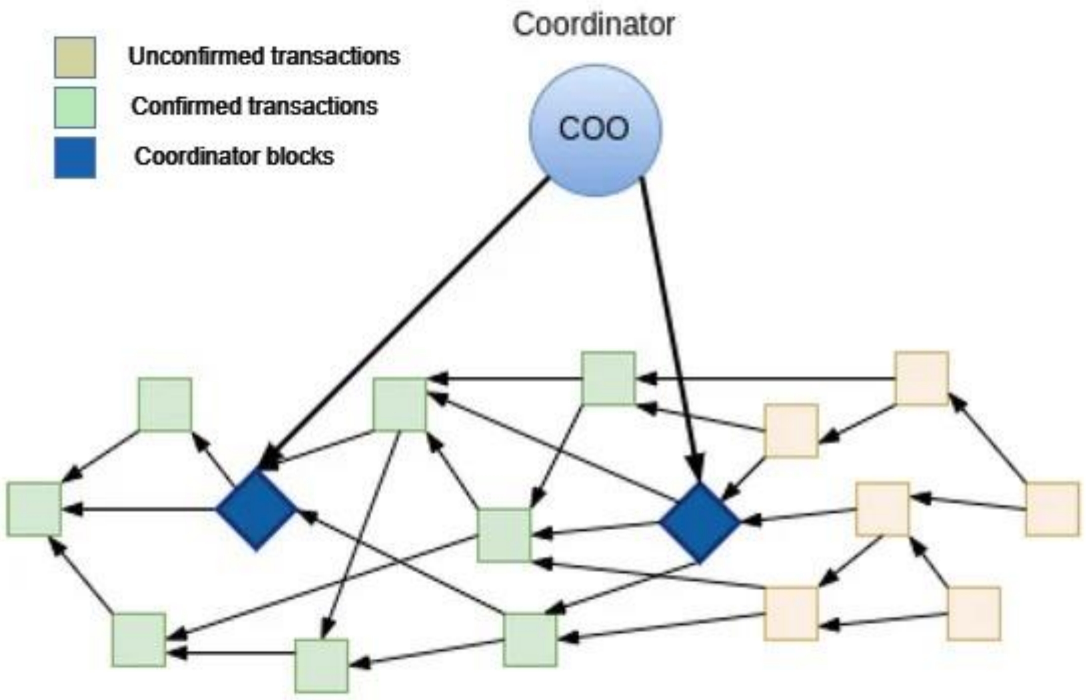


Figure 5: Tangle structure in IOTA [10].

2.4 Vechain

Vechain [1] is a blockchain environment designed for delivering a transparent, efficient, scalable and adaptable blockchain platform. It was created in 2015 by founder Sunny Lu and it is headquartered in San Marino, Europe. VechainThor is considered as a world leading smart contract spearheading the real world adoption of blockchain technology [11]. Vechain is one of the first blockchain projects with the objective to offer blockchain supply chain services to business enterprises [12]. According to Vechain, its goal is to create a trust-free and distributed business ecosystem platform to enable transparent information flow, efficient collaboration, and high-speed value transfers [13].

Vechain has cemented its position among the main blockchain environments in the world. It is in collaboration with many organizations worldwide like Walmart China, BMW, DNV, and the government of San Marino among others. Currently Vechain objective is to join the challenge of building digital ecosystems to get sustainability and digital transformation at global scale [11].

2.4.1 Working principle

Vechain [1] is a decentralized network, so it is not controlled by any individual. It is an open and public network that involve all the users to participate and involved in the community although, it is important to have a direction that will benefit the ecosystem, for that, the directive department and foundation of Vechain is supporting. In Figure 6 a simple scheme is presented to show how Vechain blockchain is designed to work.

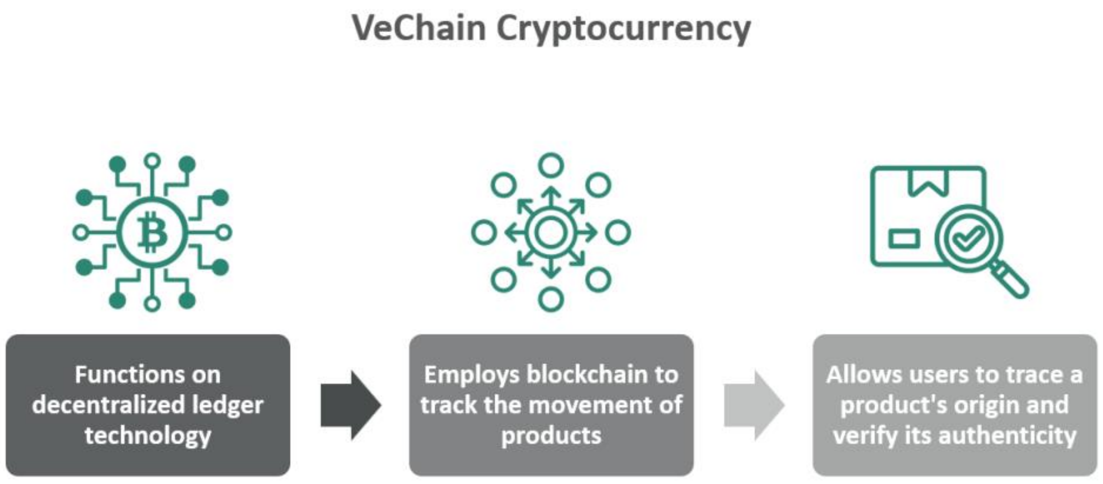

VeChain Cryptocurrency
Functions on decentralized ledger technology
Employs blockchain to track the movement of products
Allows users to trace a product's origin and verify its authenticity

Figure 6: Vechain blockchain working scheme [14].

Vechain offers a range of specific solutions and features to meet the complex requirements of businesses operating in various industries. Here are some key aspects of Vechain [11]:

- Dual-Token System: Vechain operates on a dual-token system consisting of Vechain Tokens (VET) and VeThor Tokens (VTHO). VET serves as a store of value and is used for economic activities on the VechainThor blockchain, while VTHO is used to pay for transaction fees and smart contract execution [11].
- Proof of Authority (PoA) Consensus: VechainThor uses a unique consensus mechanism known as Proof of Authority (PoA), where transaction validation is performed by a selected group of trusted authority nodes. This ensures fast transaction processing and high throughput while maintaining security and reliability.
- Enterprise Solutions: Vechain provides a comprehensive suite of enterprise solutions aimed at addressing various challenges in supply chain management, product verification, and quality assurance. These solutions include tools for product lifecycle management, supply chain traceability, anti-counterfeiting measures, and compliance verification.
- Real-World Applications: Vechain has been implemented in a wide range of industries, including food and beverage, luxury goods, pharmaceuticals, and automotive manufacturing. Real-world applications of Vechain include product traceability, authenticity verification, and quality control, enabling businesses to enhance transparency, reduce costs, and build trust with consumers.
- Integration with IoT Devices: Vechain integrates with Internet of Things (IoT) devices and sensors to capture real-world data and transmit it securely to the blockchain. This enables businesses to monitor and manage their assets more

effectively, ensuring compliance with regulatory standards and quality assurance protocols.

2.5 Comparation of distributed ledger

After describing the different distributed ledger technology, a comparation between them has been realized in order to see the advantages and disadvantages of each specific technology. This comparison will help to identify the most suitable technology for each use case.

Scalability:

- IOTA and Vechain offer innovative approaches to scalability, with IOTA's Tangle architecture enabling parallel transaction processing and VechainThor design optimized for enterprise-scale applications.
- Traditional blockchain platforms such as Ethereum and Bitcoin face scalability challenges due to their linear block structures, leading to network congestion and high transaction fees during peak usage.

Transaction Fees:

- IOTA's feeless transactions make it attractive for microtransactions and IoT applications, eliminating transaction costs for users.
- Vechain transactions can involve nominal fees denominated in VTHO Tokens, while traditional blockchain platforms typically require transaction fees paid to miners or validators and they don’t have a two token dual method.

Consensus Mechanisms:

- IOTA's Tangle consensus mechanism relies on network consensus and cryptographic algorithms, offering scalability and resilience against attacks.
- VechainThor employs a Proof of Authority (PoA) consensus mechanism, ensuring low energy consumption, fast transaction processing and high network reliability through trusted authority nodes.
- Traditional blockchain platforms use consensus mechanisms like Proof of Work (PoW) or Proof of Stake (PoS) to validate transactions, offering security and decentralization but with higher energy consumption.

Use Cases:

- IOTA and Vechain specialize in specific use cases such as IoT applications, micropayments, supply chain management, and product authenticity verification.
- Traditional blockchain platforms find applications in finance, supply chain management, healthcare, and decentralized finance (DeFi), offering a broader range of use cases.

Adoption and Partnerships: IOTA, Vechain, and traditional blockchain platforms have gained significant adoption across industries, with partnerships formed with companies, governments, and organizations worldwide.

Security and Trust: IOTA, Vechain, and traditional blockchain platforms provide strong security and immutability through cryptographic techniques, ensuring tamper-proof record-keeping and transaction verification.

Each distributed ledger technology, whether it's IOTA, Vechain, or traditional blockchain platforms, offers unique properties and capabilities, making it suitable for specific use cases and applications. By understanding the advantages and disadvantages of each technology, businesses can make informed decisions about adopting decentralized solutions to address their unique challenges and requirements. In Table 1 a comparison table is presented in order to summarize the differences between different analyzed technologies.

Aspect	IOTA	Vechain	Traditional Blockchain
Scalability	Tangle enables parallel processing.	VechainThor optimized for enterprise.	Linear blocks pose scalability issues.
Transaction fees	Feeless transactions, ideal for microtransactions.	Nominal fees in VTHO Tokens.	Fees paid to miners/validators.
Consensus Mechanism	Tangle consensus, cryptographic algorithms.	Proof of Authority (PoA) consensus.	Proof of Work (PoW) or Proof of Stake (PoS).
Use Cases	IoT, micropayments, supply chain, authenticity.	IoT, supply chain, product management.	Finance, supply chain, healthcare, DeFi.
Adoption and Partnerships	Significant adoption, global partnerships.	Significant adoption, global partnerships.	Widely adopted, partnerships across sectors.
Security and Trust	Strong security, immutability with cryptography.	Strong security, immutability.	Strong security, immutability.

Table 1: Comparison table of DLT.

3 VechainThor Blockchain

The VechainThor blockchain [15] is the first layer of the system that powers the Vechain ecosystem [16]. This blockchain is established on the Ethereum blockchain and a public history of all transactions is available through the Vechain explorer. This blockchain system is designed for enterprises with scalability [12] in mind.

Some characteristics of VechainThor blockchain are defined [11]:

- VechainThor blockchain is created to be scalable for asynchronous transactions. It scales using clauses and dependencies in the transaction process.
- VechainThor blockchain dual token method makes possible to stabilize the transaction fees during high transaction demand period. In this way, transaction fees can be defined as predictable.
- Vechain is using Proof of Authority (PoA) as a consensus mechanism taking advantage of the assurances, security and scalability and at the same time using low energy consumption, not like other blockchain technologies.
- Security is ensured thanks to the safe consensus mechanism PoA. There are no reported hacks from PoA consensus.
- It can be defined as a robust blockchain due to historical zero downtime from his foundation in 2018.
- VechainThor blockchain is compatible with other type of blockchains.

3.1 Dual token methodology

One of the main characteristics of this specific blockchain technology is the dual token method. An adequate economic model in the system is crucial for the correct behavior of the whole ledger. The reason to set an economic model with two token is to avoid the main problem of the blockchain systems, the unpredictable cost of the system due to token volatility [13].

The main token in VechainThor blockchain system is VET while the token used as transaction fee is the VTHO token. VET token is used as the utility token which means this token is defined by value medium transfer. VTHO is the gas or transaction token in charge to pay fees during a transaction. This unique dual token method makes possible to separate the cost of using the blockchain system from market speculation [11]. Figure 7 presents a scheme that shows both token and the function of each one. VET is used for business and financial activities and VTHO token is used for the transaction payment and execution of smart contracts transactions.

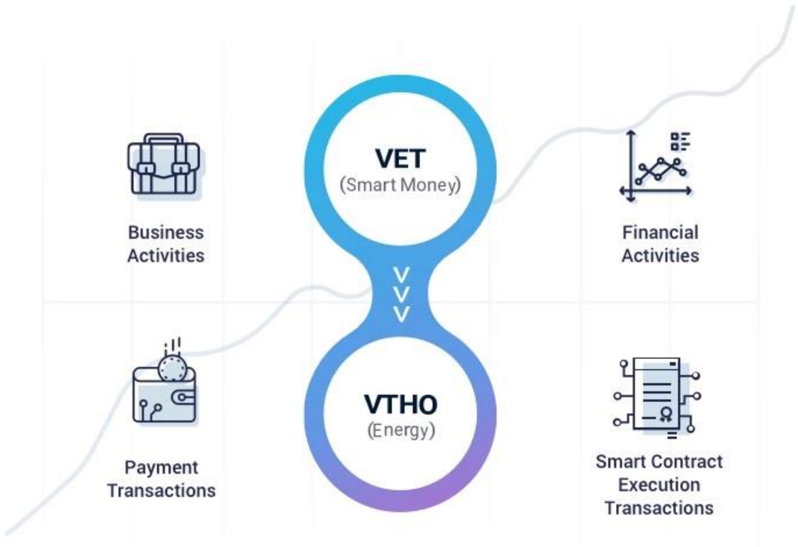


Figure 7: Dual token system sketch in Vechain [12].

In the majority of blockchain systems when the crypto market is chaotic the price of tokens tends to increase and that increases the cost of transacting on the network. This problem happens just in cases where the native token is used to pay the transactions. The main reason to implement the dual token method is to prevent transaction fees from being directly exposed to price changes. From a business view of point is important for the system to be able to predict the future cost of the blockchain system. The addition of a second VTHO token helps in making the system more predictable [6].

As explained previously, the VechainThor utility token is the VET token defined as the native coin of the system. This coin characteristic defined the coin with a precision of 18 decimal places and a total supply of 86.712.634.466 units [11].

An important factor in cryptocurrencies that makes visible the difference between real currencies and cryptocurrencies is the division of the unit. In the VechainThor case VET token can be divided in multiple subunits. The smallest subunit of VET is one wei and there are 1.00.000.000.000.000.0000 wei per VET. The main advantage of this divisibility possibility is that that micropayments can be executed in an easier way than with traditional currencies. In Vechain world micropayments are incentivized and this and other features makes the system a good choice for supporting the advancement of sustainability goals [11].

The total supply of a cryptocurrency is the maximum amount of cryptocurrency that will ever exist in the blockchain lifetime. When this value is defined it remains fixed during the time and should not change during the lifetime of the blockchain. The definition of total supply gives transparency and predictability and also provides clarity to the users of the network. It allows them to see the potential value and long-term option of the cryptocurrency [11].

The second token VTHO can be defined as the energy or the cost of making the blockchain transactions on the VechainThor blockchain. 70% of the amount of VTHO paid in a transaction as transaction fee is burned and the remaining is rewarded to the block creator, in this case the Authority Master node [11].

The VTHO token is defined as VIP180 type token. This token is defined as a fungible token standard in Vechain. VIP180 is a superset of the ERC20 standard on the Ethereum blockchain and is the most used fungible standard used across all blockchains [11]. As explained previously transactions are paid in VTHO and the Authority Master node is paid also in VTHO for creating blocks, securing and maintaining the chain.

In this case, VTHO does not have a maximum supply value like VET. VTHO parameter can only be altered by the transparent governance process that will be configured to increase or reduce the monetary supply. As explained previously the current regulation establish to destroy the 70% of VTHO used in a transaction and to give the other 30% to the Authority Master nodes as reward [11]. This makes possible to control the market value of the cryptocurrency.

The way to earn VTHO coins is to hold VET coins in the wallet. Holding VET generates VTHO automatically. VTHO is generated from holding VET at a constant rate of $\nu = 5 \cdot 10^{-9}$ VTHO per VET per second or 0.000432VTHO per VET every 24 hour [11].

$$VTHO_{gen} = \nu \cdot VET \cdot t \qquad (1)$$

For every transaction in VechainThor a transaction fee must be paid in order to pay for the computation in the network that can be define as:

$$VTHO_{fee} = \rho \cdot G \qquad (2)$$

where G is the amount of gas required to execute the transaction and, ρ is the gas price in VTHO that is a constant value defined as $1 \cdot 10^{-5}$. This value can vary from a ρ_{base} value of $1 \cdot 10^{-5}$ to a $\rho_{base} \cdot 2$. For example if a transaction requires 50.000 gas, then the VTHO fee will be 0.5 VTHO [11].

3.2 Blockchain architecture

In the VechainThor blockchain, the system architecture can be defined by different layers. Each layer will be responsible of diffeerent functionalities and interaction inside the Vechain context. Different layer can be presented as [17]:

- Consensus Layer: This layer is responsible for getting an agreement on the state of the blockchain through the consensus mechanism. In VeChainThor blockchain, the consensus layer will implements the Proof of Authority (PoA) consensus mechanism, where Authority Masternodes, that act as validators, are validating and proposing new blocks in the blockchain.
- Networking Layer: The network layer enables the communication and data exchange between nodes in the VeChainThor network. It includes protocols and mechanisms

for peer-to-peer networking, message propagation, and synchronization of blockchain data across the network.

- Blockchain Layer: At the center of the architecture, there is the blockchain layer, which consists of the general Blockchain data structure and the protocols governing its operation. This layer will store a chronological record of transactions in blocks, linked together to form an immutable and transparent ledger of transactions.
- Smart Contract Layer: The smart contract layer enables the execution of decentralized applications (dApps) and the deployment of smart contracts on the VeChainThor blockchain. Smart contracts are deployed and executed within this layer, providing programmable logic and automation of business processes.
- Application Layer: The application layer consists on the user-facing applications and interfaces built on top of the VeChainThor blockchain. This layer includes decentralized applications (dApps), wallets, and other tools that interact with the blockchain to provide value-added services to users. The developed dApp will be located on this layer.
- Protocol and Governance Layer: This layer manage the rules, protocols, and governance mechanisms that define the operation and evolution of the VeChainThor blockchain. It includes features like token economics, consensus rules, and governance mechanisms.

By organizing the blockchain architecture into distinct layers, the VeChainThor blockchain achieves modularity, scalability, and maintainability, allowing for the efficient development, deployment, and operation of decentralized applications and services. Each layer interacts with adjacent layers in a structured manner, ensuring the integrity and reliability of the blockchain ecosystem. In Figure 8 the system architecture diagram is presented. The blue color layers present the technical functionalities of the system.

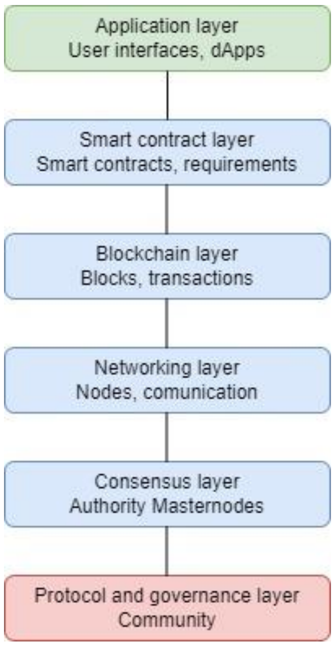


Figure 8: Vechain system architecture [17].

3.3 Cryptocurrency wallet

Cryptocurrency wallets are applications that work as a usual wallet but for digital tokens. In this application you can access to your user adress in the application interface will let the user access to his crypto account. Technically these applications dont't work the same way as a wallet. Instead of holding the phisical money a crypto wallet will store the needed passkeys for making transactions in the blockchain and will provide the user a display of their crypto balance [18].
Inside cryptocurrency wallet, there are different type of wallets [18]:

- Software Wallets: This digital wallets are implemented as software applications, being accessible via desktop, mobile, or web interfaces. Software wallets can be further categorized into:
    - Hot Wallets: They are connected to the internet and suitable for frequent transactions. Examples include mobile wallets and desktop wallets.
    - Cold Wallets: They are offline storage solutions designed for long-term asset storage and enhanced security. Examples include hardware wallets and paper wallets.
- Hardware Wallets: These are physical devices, often USB devices, designed to store cryptocurrency private keys offline. Hardware wallets offer security by keeping the private keys offline, reducing the risk of getting unauthorized access or hacking.
- Paper Wallets: A paper wallet is a physical document containing a cryptocurrency wallet address and private key. It is generated offline and considered one of the most secure forms of cryptocurrency storage, as it is not possible to hack or get online attacks.

Key features of cryptocurrency wallets [18]:

- Security: The safety of the digital tokens of the user is prioritize in a crypto wallet. As safety methods these wallets use encryption of private keys, multi-factor authentication, and secure backup options.
- Easy to use: Wallets are designed to have a user friendly interface allowing the using an easy access to the different capabilities of the wallet.
- Compatibility: It is important for a good crypto wallet to support different ccryptocurrency and be able of working in different blockchain environment.
- Asset control: The user of the wallet will get all the control of their digital assets when using the crypto wallet. The private keys are managed locally giving the user the power in their account.
- Interoperability: This feature allows the user to interact with decentralized applications and decentralized finance platforms from their wallets.

Cryptocurrency wallets are indispensable tool for any blockchain user in order to manage and use their digital assets in a safe and convenient way. A good understanding of the crypto wallets is needed for selecting the correct wallet type. During the project Sync2 [19] wallet has been used.

3.4 Decentralized Applications (dApp)

Decentralized applications also known as (dApps) can be defined as digital applications that run in a blockchain network of computers and not just in a single computer. This software applications will operate in a blockchain network. A decentralized application runs in a peer-to-peer (P2P) blockchain network [20].

Decentralization in these applications means that there is not a single individual authority in the application. Once the developer of the dApp has released, the application will lose the authority of the developer and others are able to build on top of it.

In a centralized application, the owner of the program will have all the control and the authority no matter how many users the program has. This makes the owner to be in charge of the backend of the centralized application.

At the same time, dApps run in a public, open-source, decentralized environment and as mentioned they don't have just an individual control source or owner. In a case where the developed application makes possible to publish or upload messages if someone published a message, there is no way to delete that message. There is no user that has more authority than other even the creator of the application will not be able of making any change in the published message [20].

The main function of developing this kind of application is to decentralized actions that nowadays are centralized. They are many types of scenarios where a decentralized application usage could be suitable like in order to enable secure, blockchain-based voting and governance. Other use of dApps are listed below [21]:

- **Financial services:** dApps are able to make easier financial transactions or exchange of the currencies or assets.
- **Supply chain management:** dApps are able of tracking the movement of products of a company through a supply chain. In this way transparency and accountability will be ensured.
- **Identity verification:** dApps can be used to store and verify identity information, like passport applications.
- **Real estate:** dApps can be used to facilitate the buying and selling of real estate directly between buyer and seller, as well as the tracking of property ownership and related documentation such as deeds.
- **Healthcare:** dApps can be used to store and track healthcare records, as well as to facilitate the communication and collaboration of healthcare professionals.
- **Education:** dApps can be used to create decentralized learning platforms, allowing students and teachers to interact and collaborate directly without the need for intermediaries.
- **Social media:** dApps can be used to create decentralized social media platforms, allowing users to interact and share content without the need for a central authority.
- **Predicting markets:** dApps can be used to create decentralized platforms for predicting all kind of markets.

The decentralized applications are stored and executed on blockchain networks.

4 Analysis of distributed application dApp

In this chapter, the decentralized application will be analyzed and designed. For a good development of the decentralized application, the design and structure are crucial. In the first section, application requirements are collected and then from the defined requirements use-cases are extracted and a use-case diagram and system sequence diagram has been constructed. A good design process is crucial for a proper implementation later in the project. The Unified Process UP has been followed during the analysis phase of the project [22].

4.1 Collecting the Requirements

First a requirement list has been developed in order to organize the requirements of the developed application. Requirement list can be seen in Table 2. All the requirements are classified depending on their functionality using FURPS+ method as defined in the Unified Process [22]. This process helps to identify the challenges that will be founded during the implementation process.

Table 2: Requirements for the developed application.

Functional	• Authenticate and authorize the user • Store Data on Vechain [1] Blockchain • Read Data from Vechain Blockchain • Display data from the Vechain Blockchain for customer and provider.
Usability	Make a user-friendly interface for easy data input and read.
Reliability	• Ensure data integrity and confidentiality through encryption techniques in the dApp. Employ secure protocols for transmitting and storing data on the blockchain to prevent unauthorized access or tampering. • Implement error handling mechanisms to handle exceptions gracefully. Log system activities and errors to facilitate troubleshooting and auditing.
Performance	Design the application to handle a potentially large volume of data transactions efficiently. Optimize performance to minimize latency and ensure responsiveness, even during peak usage
Supportability	Provide documentation and help including user guides.
+	-

4.2 Use case diagram

After defining the requirements, use cases are obtained and a use case diagram has been developed based on the functionality requirements of the software. Actors and use cases are defined in Figure 9. In the developed software, there will be three main use cases. Initially the software needs to have an authentication phase where the user will be authenticated through the signing of a virtual certificate. After that, the software will provide actors the option to store the quantity of products delivered from the provider to the customer and at the same time read option is displayed in order to give the consumer the information saved on the blockchain. Errors will be handled, and problems or wrong information will be displayed in the screen.

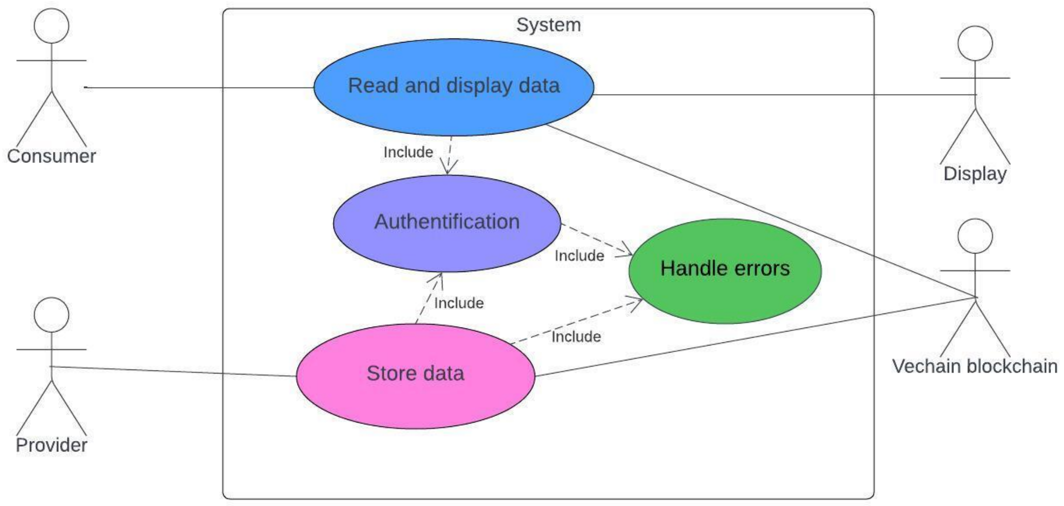


Figure 9: Use case diagram.

4.3 Fully dressed use case document

Fully dressed use case document of the most important use case has been studied based on the Unified process [22]. The most important use case for a good implementation of the software is storing the number of products delivered from the provider. The developed document is shown in Table 3. This document contains the most important factors to consider during the implementation of the most important use case.

Use case name	Store data
Scope	Vechain blockchain.
Level	User goal
Primary actor	Provider

Stakeholders	The product provider wants to record the delivered quantity. The customer wants to be able of checking the product quantity.
Preconditions	Authentication must be completed. Number of products must be defined. Smart contract must be deployed.
Success guarantee	Value is stored in the VechainThor blockchain. Reading option is available on the dApp.
Main success scenario	1. Enter delivered product quantity. 2. Press store button. 3. Request transaction call. 4. Execute smart contract. 5. Sign transaction
Extensions	2.1. User authentication failed. 2.2. Error message. 4.1 Transaction error. 4.2 Error message transaction.
Special requirements	None

Table 3: Fully dressed use case document for storing data.

The fully dressed use case document for the second use case has also been developed and is presented on Table 4.

Use case name	Read and display data
Scope	Vechain blockchain.
Level	User goal
Primary actor	Consumer
Stakeholders	The product provider wants to record the delivered quantity. The customer wants to be able of checking the product quantity.

Preconditions	Authentication must be completed. Number of products must be defined and registered. Smart contract must be deployed.
Success guarantee	Previously registered value is showed in the dApp. Reading table is updated.
Main success scenario	1. Press reading button. 2. Wait connection with the blockchain. 3. Get the last registered value on Vechain. 4. Update the application table.
Extensions	2.1 Reading error. 2.2 Error message transaction.
Special requirements	None

Table 4: Fully dressed use case document for read and display data.

4.4 System Sequence diagram

A system sequence diagram has been developed in order to see the designed structure of the decentralized application with all the functionalities previously described. In Figure 10, you can see the designed system sequence diagram, where first the authentication is executed. The authentication phase is crucial in order to give access to the user for saving the quantity of products delivered from provider to customer. In the authentication, first the system checks the user using an authentication certificate in the Vechain blockchain. If the authentication phase is failed and the user is not found, the error will be displayed in the software. If everything is correct and the certificate is signed by the user, the user address will be displayed and the store and read options will be displayed giving full access to the user for all utilities in the dApp.

After getting the full access to the dApp, the application gives the option to store a value in the blockchain. The software is designed to store the quantity of products delivered. For that, transaction details need first to be defined in the user interface and then store button is pressed. To complete the store transaction in the Vechain blockchain, the transaction needs to be signed by the user using his crypto wallet password. If everything has been completed correctly the value will be stored in the blockchain.

After storing the quantity of products, the stored quantity can be checked for confirming the correct functionality of the software. The Read button is in charge of this use case, when the customer presses the button, the system will request the contract to Vechain blockchain and the value will be extracted and sent to the system. Then the value will be displayed in the dApp software. The System Sequence diagram in Figure 10 shows all the working utilities in the designed decentralized application.

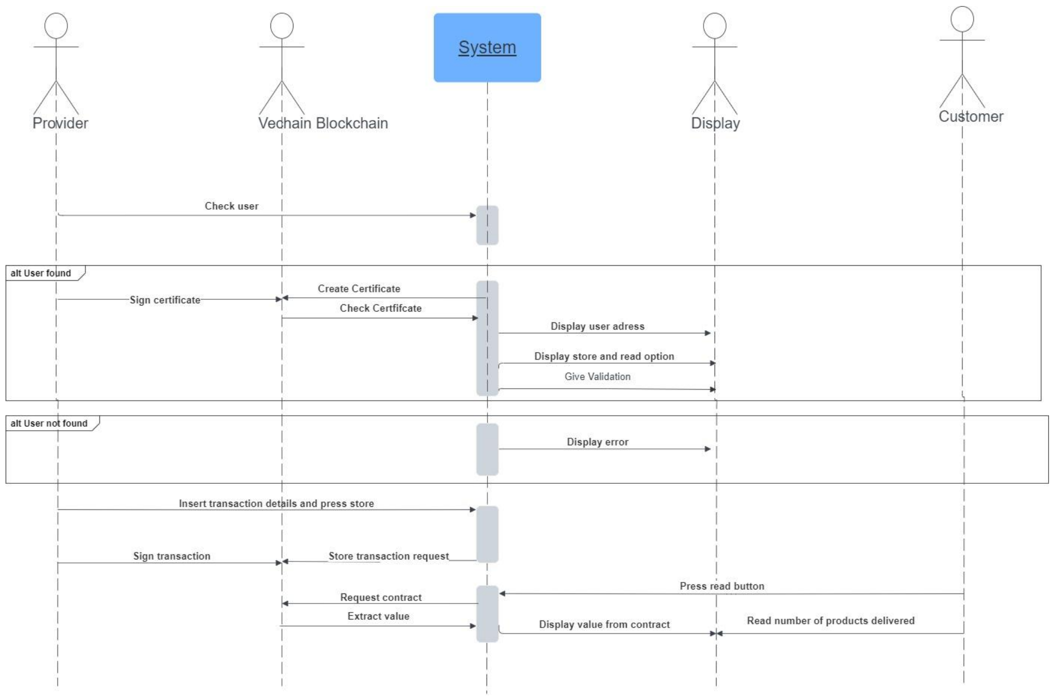


Figure 10: System Sequence Diagram.

4.5 System Architecture

The system architecture of the decentralized application (dApp) encompasses several layers and components to facilitate secure user authentication, data storage, and retrieval on the VechainThor blockchain. The architecture is designed to ensure robustness, scalability, and efficiency in handling user interactions and blockchain transactions. Below is a detailed overview of each component:

- User Interface (UI) Layer: The UI layer serves as the interface for users to interact with our dApp. It includes intuitive user interfaces and forms for authentication, data input, and display.
- Application Layer: At the core of the designed dApp, the application layer contains all the logic and functionality. This layer consists of:

  - Authentication Module: Responsible for verifying the user identity.
  - Data Storage Module: Handles the storage and retrieval of data connecting with the Vechain blockchain.
  - Business Logic: Processes user requests, validates input data, and manages interactions between different layers.
- Blockchain Layer: Interfacing directly with the Vechain blockchain, this layer manages smart contracts and transaction execution. Key components include:

    - Smart Contracts: These contracts define the rules for storing and accessing data securely on the blockchain.
    - Transaction Management: Manages the creation and execution of transactions, ensuring data integrity and consistency.
- Network Layer: The network layer facilitates communication between the dApp components and the Vechain blockchain network. It ensures secure and efficient data transmission.

- Security Layer: Ensuring the integrity and confidentiality of user data and transactions, the security layer implements robust security measures, including access control mechanisms and authorization protocols.

This designed system architecture provides a solid foundation for the development of the dApp, providing friendly user experience and secure interactions with the Vechain blockchain. Through the effective integration of these layers and components, the objective is to develop a reliable and efficient decentralized solution to the users.

5 Implementation

In this chapter, the technical details of the developed decentralized application (dApp) in the VechainThor blockchain are explained. The chapter begins with an introduction of the used tools for developing the dApp, this will include the tools, libraries and frameworks utilized during the process. Next sections in the chapter explore smart contract development, covering design principles and implementation strategies for enabling the storage and authentication of product deliveries. Additionally, the chapter discusses frontend development, focusing on user interface design and functionalities into the dApp. Furthermore, it examines the integration of the dApp with the VechainThor blockchain, showing mechanisms for deploying smart contracts, executing transactions, and requesting blockchain data. The chapter also contains testing, debugging, security considerations, optimization techniques, user documentation, and support mechanisms to provide a comprehensive overview of the implementation process.

5.1 Introduction to the development environment

In developing the decentralized application (dApp) on the VechainThor blockchain, a robust and efficient development environment is essential to build the implementation process. The chosen tools and technologies were meticulously selected to get a good integration, robust smart contract development, and user-friendly interface design.

The learning process of the different tools has been an important aspect of the project's journey. During the development process, getting knowledge in each tool and technology has been essential for achieving project objectives effectively. The experience obtained during this phase has not only contributed to the successful implementation of the dApp but has also provided valuable insights and skills for future projects.

- Smart contract development was facilitated by the Remix IDE [23] (Integrated Development Environment), a versatile platform offering comprehensive tools for writing, testing, and debugging smart contracts directly in the browser. Remix intuitive interface and built-in compiler simplified the process of deploying contracts to the VechainThor blockchain, ensuring efficiency and reliability in the development workflow.
- The frontend interface of the dApp was created using Vite [24], a next-generation frontend tooling framework known for its blazing-fast development experience. Leveraging HTML [25] and JavaScript [26] as primary coding languages, Vite facilitated the creation of dynamic and interactive web interfaces, enabling seamless interaction with the underlying blockchain functionalities.
- Visual Studio Code (VS Code) [27] served as the preferred Integrated Development Environment (IDE) for programming the dApp, providing a powerful and customizable environment for coding, debugging, and version control. With its extensive ecosystem of extensions and plugins, VS Code offered a rich set of features together with the specific requirements of blockchain development, giving productivity and code quality to the software development.

To summarize the Table 5 presents the used tools and a short description of each software.

Development tool	Description
Remix IDE [23]	Programming environment for smart contract develop.
Vite [24]	The development server used to create the web application.
HTML [25]	Programming language used for frontend design of the dApp.
JavaScript [26]	Programming language used for backend design of the dApp and connection with Vechain.
Visual Studio Code (VS Code) [27]	Programming environment used to code and developed the designed application in the project.

Table 5: Summary of the used tools in the project.

In addition to selecting the appropriate tools and technologies, careful consideration has been given to maintain compatibility and interoperability between different components of the development environment. This included the integration between smart contract development in Remix IDE and frontend development in Vite, allowing for efficient communication and data exchange between the backend and frontend layers of the dApp. Furthermore, Visual Studio Code (VS Code) [27] served as a central environment for the development workflow. In Figure 11 ,the used tools system sketch can be seen, where the previously described tools are displayed for a good understanding of how each tool has helped to the application development.

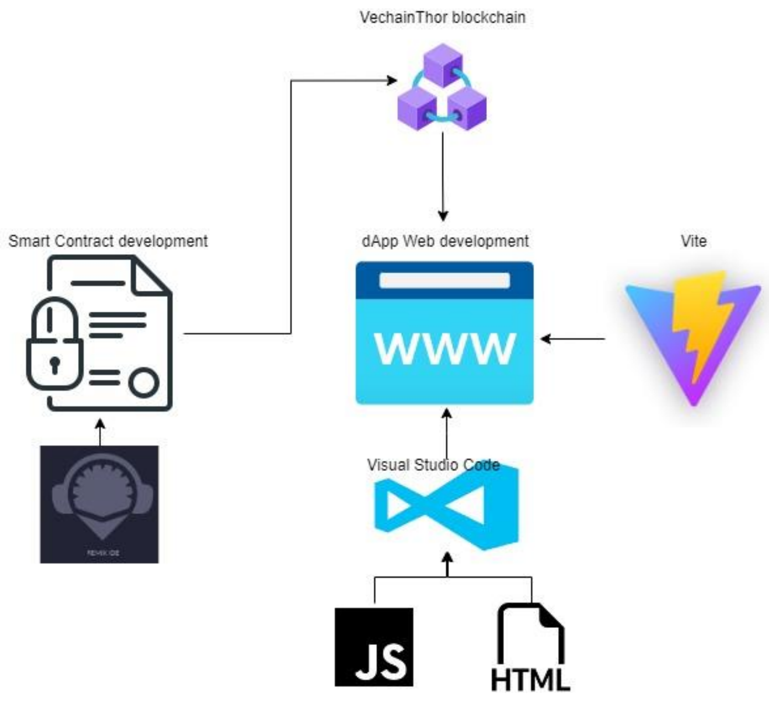


Figure 11: Development tools system sketch.

5.2 Smart contract development

Smart contract development [6] is a crucial aspect of building decentralized applications (dApps) on blockchain platforms like VechainThor. In this subsection, the main concepts about smart contracts will be explained, from designing, implementing, and deploying smart contracts to facilitate the main functionalities of the designed dApp. For the development of the smart contract as explained previously Remix IDE [23] has been used.

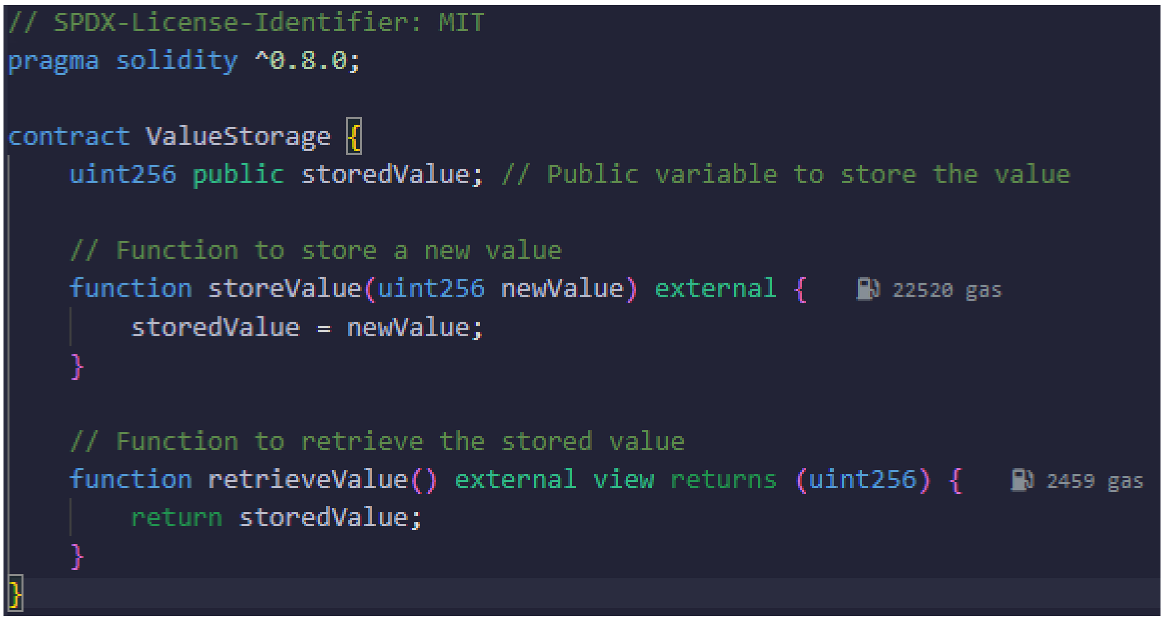
Before starting to code and build the smart contract, it is very important to have a good understanding of what type of contract is needed for this application and how to build it. Before using a specific smart contract, it is important to understand the smart contract requirements:

Smart Contract Requirements are:

- Register product information.
- Update product authenticity status.
- Record delivery date.
- Verify delivery authenticity.
- Access Control
- Restrict access based on user roles.
- Gas Optimization:
- Optimize gas usage for efficiency.
- Security Considerations:

These requirements focus on the core functionalities and security considerations of the smart contracts, providing a clear outline for their development on the VechainThor blockchain.

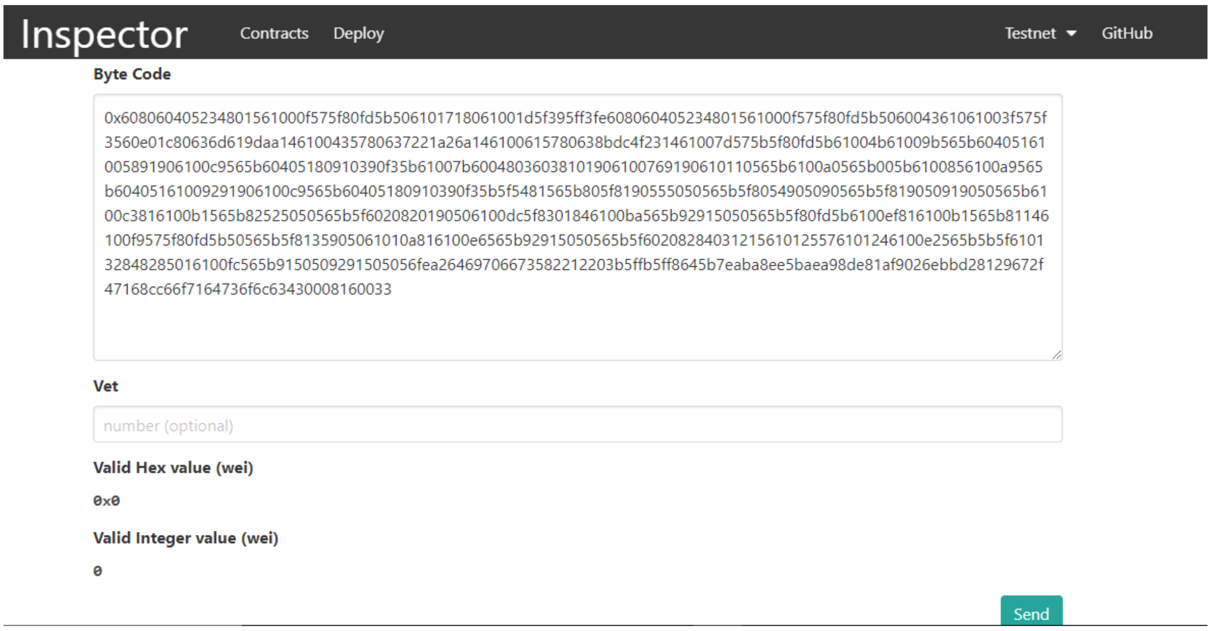
After defining smart contract requirement, the coding part has been executed and a smart contract code has been constructed for saving the quantity of products delivered from a provider to a customer. In Figure 12, the developed smart contract can be seen.



```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract ValueStorage {
    uint256 public storedValue; // Public variable to store the value

    // Function to store a new value
    function storeValue(uint256 newValue) external {    22520 gas
        storedValue = newValue;
    }

    // Function to retrieve the stored value
    function retrieveValue() external view returns (uint256) {    2459 gas
        return storedValue;
    }
}
```

Figure 12: Smart contract developed in Remix IDE.

After constructing and building the smart contract, the contract must be deployed in the Vechain blockchain. To deploy the contract on the blockchain environment the Inspector [28] tool has been used. In Figure 13, it can be seen the used tool where the bytecode of the smart contract must be extracted from the Remix IDE solidity code in order to deploy the contract in Vechain.



Byte Code

```
0x6080604052348015610000f575f80fd5b5061017180610001d5f395ff3fe6080604052348015610000f575f80fd5b506004361061003f575f
3560e01c80636d619daa146100435780637221a26a146100615780638bdc4f231461007d575b5f80fd5b61004b61009b565b60405161
005891906100c9565b60405180910390f35b61007b60048036038101906100769190610110565b6100a0565b005b6100856100a9565
b604051610092919061006c9565b60405180910390f35b5f5481565b805f8190555050565b5f8054905090565b5f819050919050565b61
00c3816100b1565b82525050565b5f6020820190506100dc5f8301846100ba565b92915050565b5f80fd5b6100ef816100b1565b81146
100f9575f80fd5b50565b5f8135905061010a816100e6565b92915050565b5f60208284031215610125576101246100e2565b5b5f6101
32848285016100fc565b9150509291505056fea26469706673582212203b5ffb5ff8645b7eaba8ee5baea98de81af9026ebbd28129672f
47168cc66f7164736f6c63430008160033
```

Vet

number (optional)

Valid Hex value (wei)

0x0

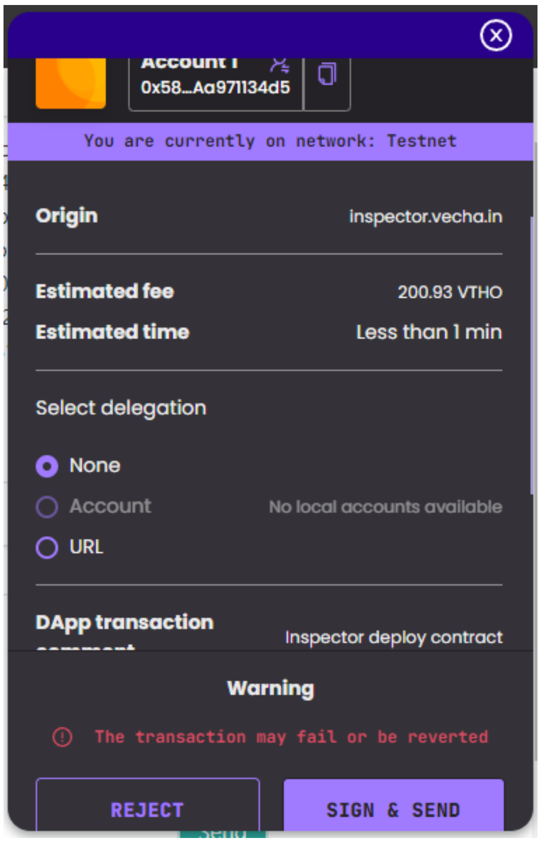
Valid Integer value (wei)

0

Send

Figure 13: Contract deployment in Inspector tool.

After deploying the smart contract in Vechain, the transaction needs to be completed by paying the required gas token. In this deployment, VTHO tokens are required. In Figure 14, the transaction can be seen and is defined as Inspector deploy contract.



You are currently on network: Testnet

Origin	inspector.vecha.in
Estimated fee	200.93 VTHO
Estimated time	Less than 1 min

Select delegation

[● None]
○ Account — No local accounts available
○ URL

DApp transaction comment	Inspector deploy contract

Warning

The transaction may fail or be reverted

Figure 14: Smart Contract deployment.

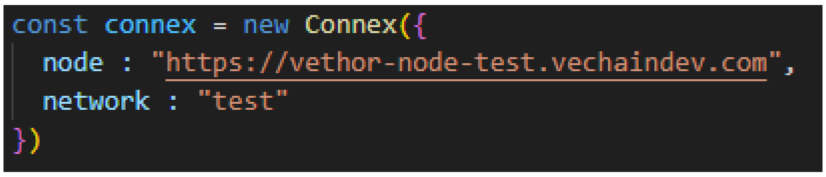
5.3 Integration with VechainThor blockchain

The main functionality of the dApp is to be connected with the previously explained Vechain environment. This can be reduced in being able of storing values in the blockchain and retrieving the values from it. To develop the connection needed between the dApp and the blockchain, JavaScript [26] language has been used.

The connection between blockchain and user in dApp is developed by using the Connex library [29] in JavaScript. Connex library is used as a communication bridge between the developed dApp and the VechainThor blockchain. The library provides a set of tools that allow the developers to perform different tasks like sending transactions, requesting data and executing smart contracts. As example, online course on developing dApp on Vechain has been taken [30]. Code [31] from that course has been used as reference to develop the specific dApp. All the developed code files for this project can be seen in a Github public repository shared in Appendix B: Github repository.

5.3.1 Initial steps for network connection

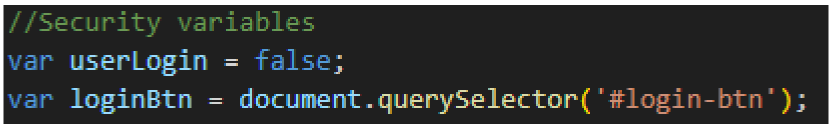
After importing the library in the main.js file of the developed dApp. The initial step in establishing the connection is to define the network to which the dApp will connect. As presented in Figure 15, in this scenario, the test network is designated as the connected network to provide a testing environment for development purposes. The defined constant 'connex' network will be utilized for connecting to the blockchain.



```
const connex = new Connex({
  node : "https://vethor-node-test.vechaindev.com",
  network : "test"
})
```

Figure 15: Connection of dApp and test network.

Some variables are also defined in order to maintain a secure and fast connection process. In Figure 16, the defined variables can be seen, one variable is used for checking the user login status and another for login button.

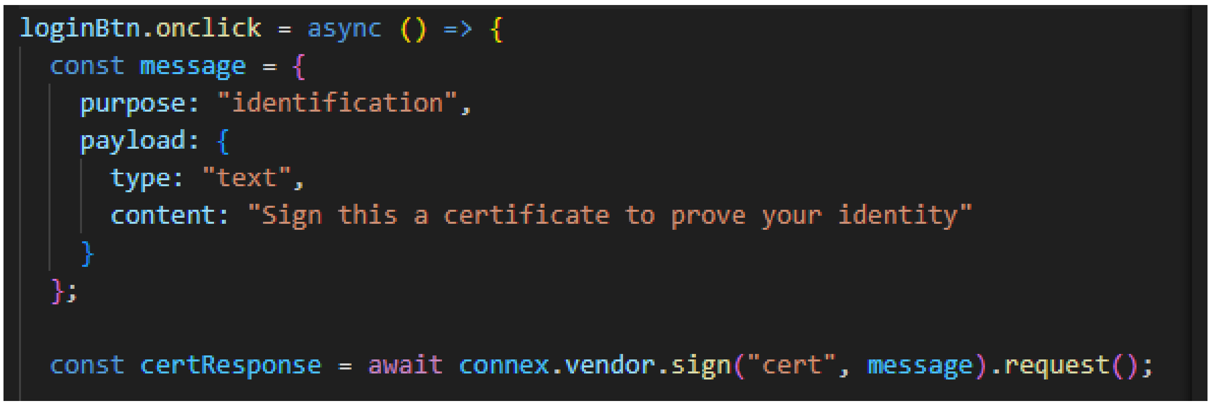


```
//Security variables
var userLogin = false;
var loginBtn = document.querySelector('#login-btn');
```

Figure 16:Variables definition.

5.3.2 Authentication certificate

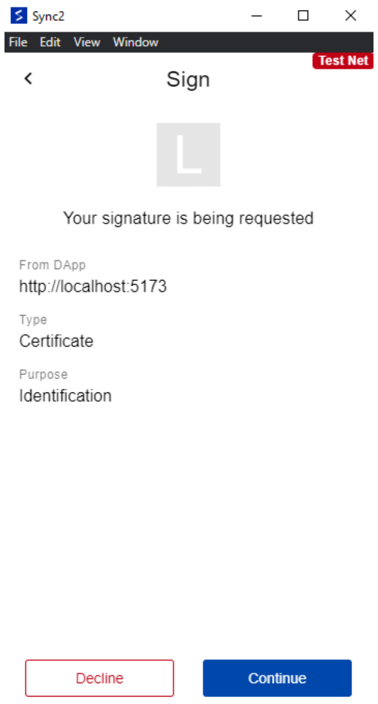
The first step in the development of the dApp is to authenticate the user. For that, in Figure 17, the developed code for user authentication can be seen. When the login button from the user interface is pressed, the code is executed. First, an identification message is defined with an appropriate title and content. Then, with the message and the use of a vendor module from the previously defined connex network, the request of a certificate signing is executed.



```
loginBtn.onclick = async () => {
  const message = {
    purpose: "identification",
    payload: {
      type: "text",
      content: "Sign this a certificate to prove your identity"
    }
  };

  const certResponse = await connex.vendor.sign("cert", message).request();
```

Figure 17: User login in the decentralized application.

The user login procedure requires the user to sign a transaction in the used wallet, in this case Sync2 wallet has been used. In Figure 18, the proposed authentication certificate with the defined message can be seen. This certificate must be signed in order to use the dApp and the other functionalities.



Your signature is being requested

From DApp
http://localhost:5173

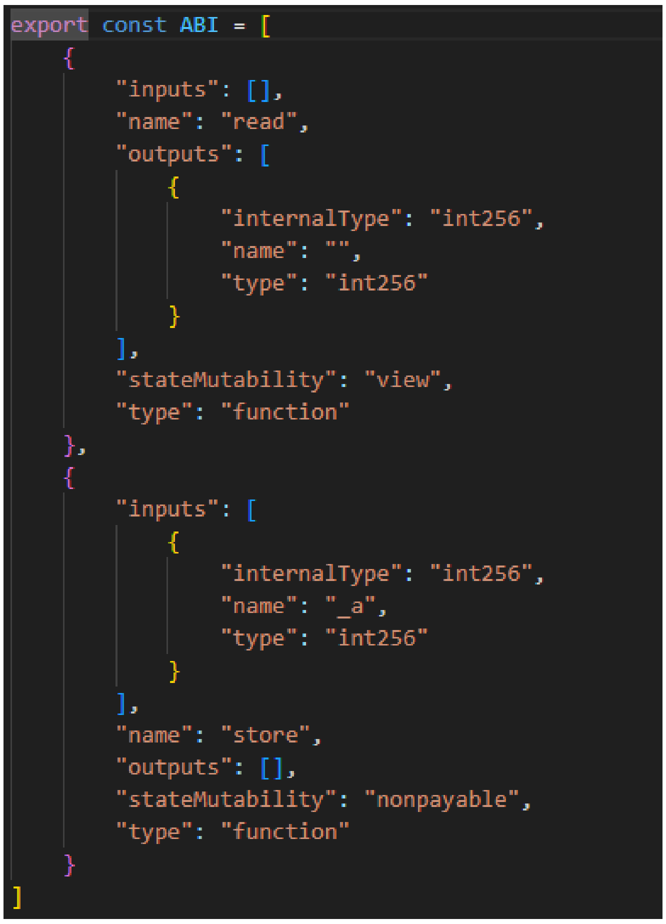
Type
Certificate

Purpose
Identification

Figure 18: Authentication certificate.

5.3.3 Storing data on Vechain

Once the authentication process is complete, the application will give the user, the opportunity to store the number of products delivered from the provider to the customer. To start with the store procedure, the ABI from the deployed smart contract must be first extracted and imported in an abi.js file. In Figure 19 this file, the storing variables in blockchain are defined as the Application Binary Interface. This file defines how to interact with the already deployed smart contract. The file contains function signatures, event definitions and the contract deployment information.

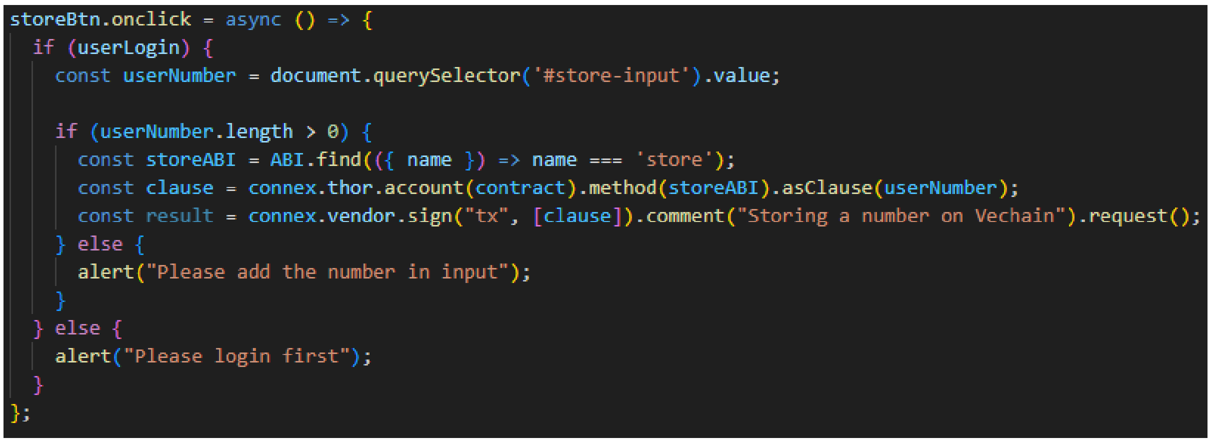


```
export const ABI = [
    {
        "inputs": [],
        "name": "read",
        "outputs": [
            {
                "internalType": "int256",
                "name": "",
                "type": "int256"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    },
    {
        "inputs": [
            {
                "internalType": "int256",
                "name": "_a",
                "type": "int256"
            }
        ],
        "name": "store",
        "outputs": [],
        "stateMutability": "nonpayable",
        "type": "function"
    }
]
```

Figure 19: ABI for storing and reading data.

After doing that, when the store button is pressed, the next code piece is executed as shown in Figure 20. In this event, first the user login variable is checked to know if authentication has been correctly completed or not. If authentication has not been completed an error message will appear telling the user that login is needed in order to continue. In the case that authentication was correctly executed, the application will look for the ABI defined store function and will save it in a constant storeABI.

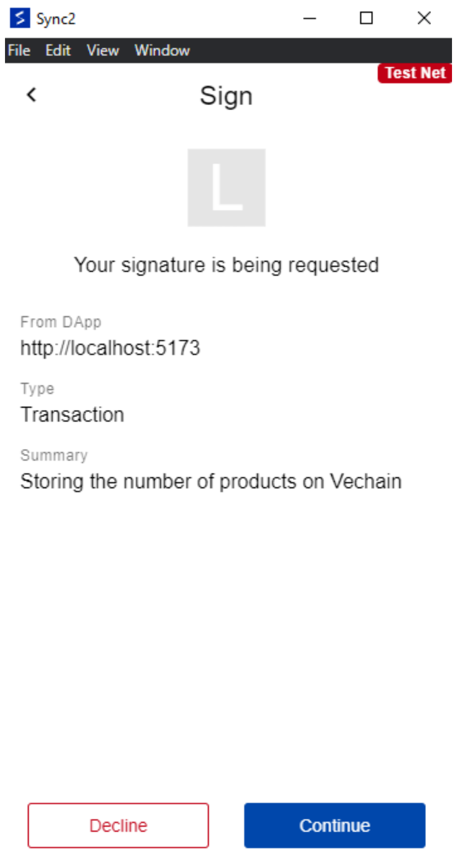
Previous step to execute the smart contract is to define the clause or condition. In the clause, the 'id' is defined by introducing the deployed smart contract address as the account, 'storeABI' as the method, and 'usernumber,' which represents the number of products delivered, as a parameter. Then the transaction can be executed by requesting to the vendor module. In case there is no number defined as input, an error message will appear.



```
storeBtn.onclick = async () => {
  if (userLogin) {
    const userNumber = document.querySelector('#store-input').value;

    if (userNumber.length > 0) {
      const storeABI = ABI.find(({ name }) => name === 'store');
      const clause = connex.thor.account(contract).method(storeABI).asClause(userNumber);
      const result = connex.vendor.sign("tx", [clause]).comment("Storing a number on Vechain").request();
    } else {
      alert("Please add the number in input");
    }
  } else {
    alert("Please login first");
  }
};
```

Figure 20: Store procedure code .

When a transaction is requested, it will appear in the wallet that is being used, which in this case is Sync2, as shown in Figure 21 when the execution ‘const result’ line in Figure 20 is triggered. This transaction must be signed to store the number in the blockchain. Once the transaction is signed, the smart contract will be executed in VechainThor test net blockchain. In Figure 22, transaction fee can be seen in VTHO token.



Sync2
File Edit View Window
Test Net
Sign

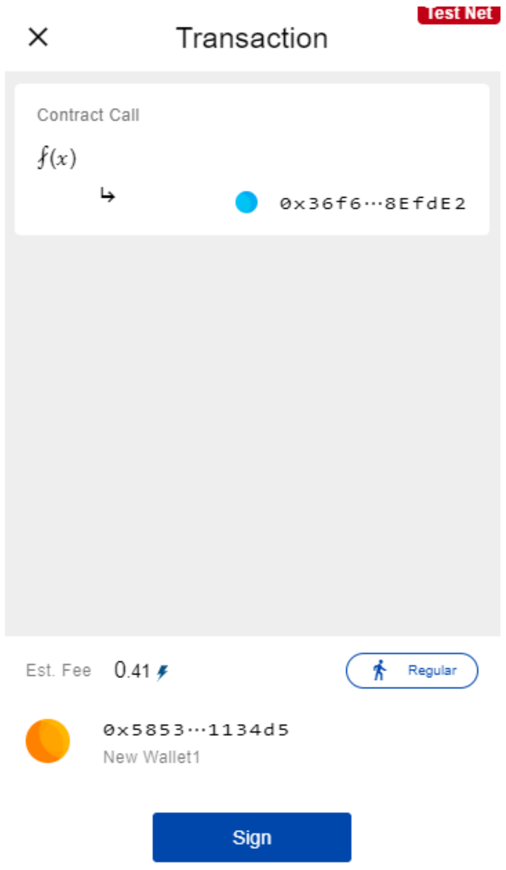
Your signature is being requested

From DApp
http://localhost:5173

Type
Transaction

Summary
Storing the number of products on Vechain

Figure 21:Transaction request in Sync2.



Test Net
Transaction

Contract Call
$f(x)$
0x36f6⋯8EfdE2

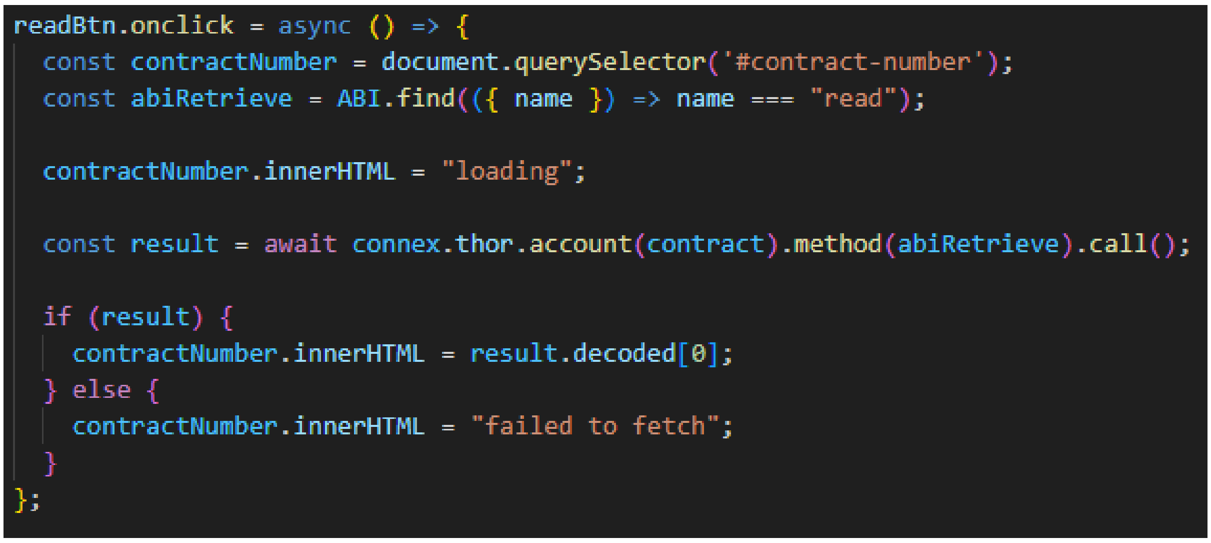
Est. Fee 0.41
Regular

0x5853⋯1134d5
New Wallet1

Figure 22: Transaction cost in Sync2.

5.3.4 Reading data from Vechain

The last functionality of the decentralized application is to read the stored value in the VechainThor blockchain. This application will be able to read the previously stored value in the blockchain. To achieve that, use case the code in Figure 23 has been developed. This code is only executed when the read button is pressed. In this event, the looked function in the ABI will be read and will be stored in abiRetrieve constant. Then during the process loading will be plotted in the user interface in order to notify the user that reading process is correctly executing. For getting the saved value in the blockchain a call is executed with the deployed smart contract address and the abiRetrieve constant. This call will be decoded, and its value will be displayed in the user interface.



```
readBtn.onclick = async () => {
  const contractNumber = document.querySelector('#contract-number');
  const abiRetrieve = ABI.find(({ name }) => name === "read");

  contractNumber.innerHTML = "loading";

  const result = await connex.thor.account(contract).method(abiRetrieve).call();

  if (result) {
    contractNumber.innerHTML = result.decoded[0];
  } else {
    contractNumber.innerHTML = "failed to fetch";
  }
};
```

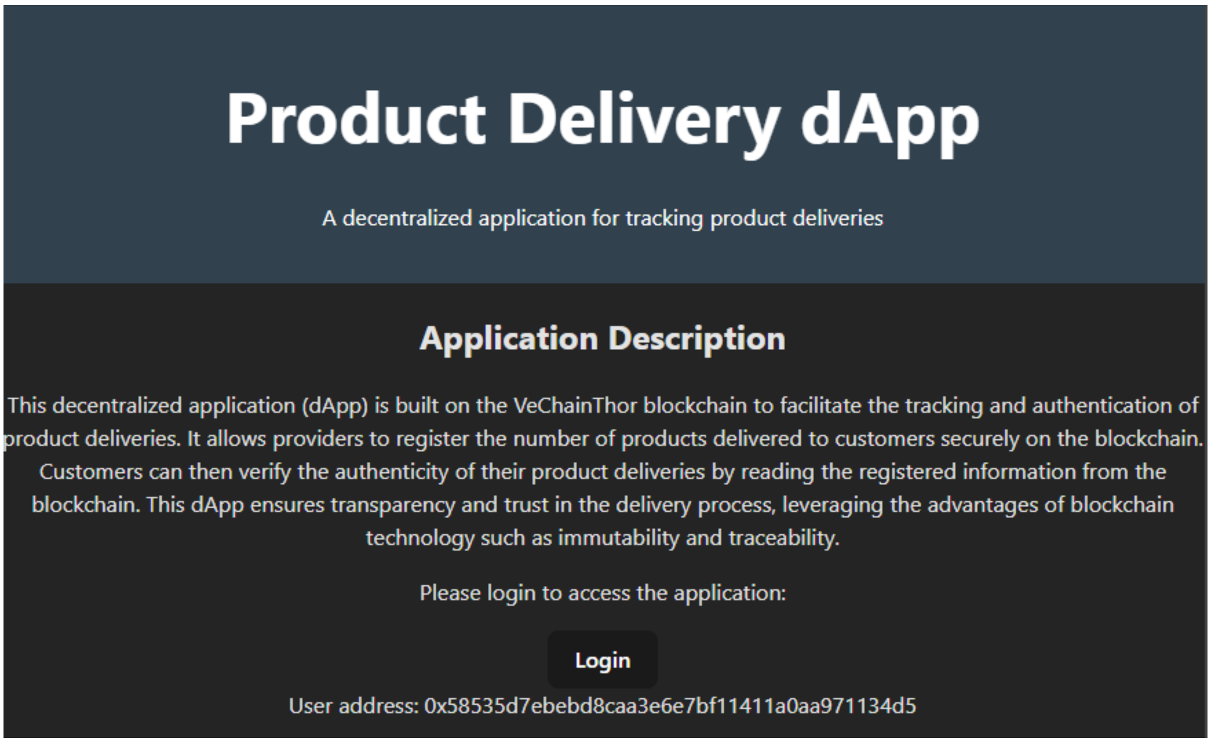
Figure 23: Code for reading data from Vechain.

5.4 Front-end development

The frontend development of the decentralized application (dApp) on the VeChainThor blockchain serves as the primary tool for users to interact with the created solution in this project. This section explains the process of creating the user interface (UI) and user experience (UX) that define the face of the dApp.

The front-end development has been completed with the use of tools such as Vite [24] and a vanilla JavaScript [26] template, prioritizing simplicity, performance, and scalability in the frontend development. Complemented by HTML [25] for structuring the user interface. The main focus was on the development process, while also making sure to prioritize the best possible user experience.

At the beginning of the dApp, a small description of the application is presented as it can be seen in Figure 24. In this starting page, the application will ask the user to please login using the defined Login button. Once the user is logged the user address of the specific user will be given as shown in Figure 24.


Product Delivery dApp

A decentralized application for tracking product deliveries

Application Description

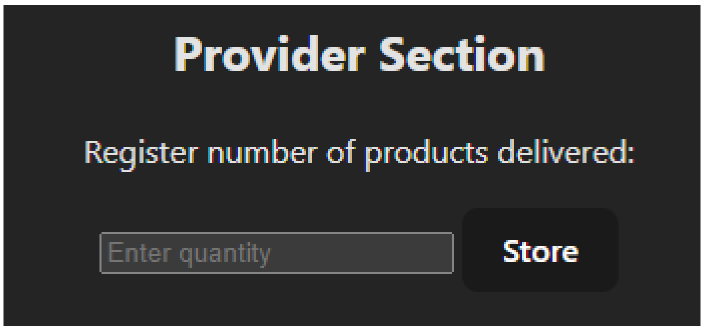
This decentralized application (dApp) is built on the VeChainThor blockchain to facilitate the tracking and authentication of product deliveries. It allows providers to register the number of products delivered to customers securely on the blockchain. Customers can then verify the authenticity of their product deliveries by reading the registered information from the blockchain. This dApp ensures transparency and trust in the delivery process, leveraging the advantages of blockchain technology such as immutability and traceability.

Please login to access the application:

User address: 0x58535d7ebebd8caa3e6e7bf11411a0aa971134d5

Figure 24: Authentication and dApp description section.

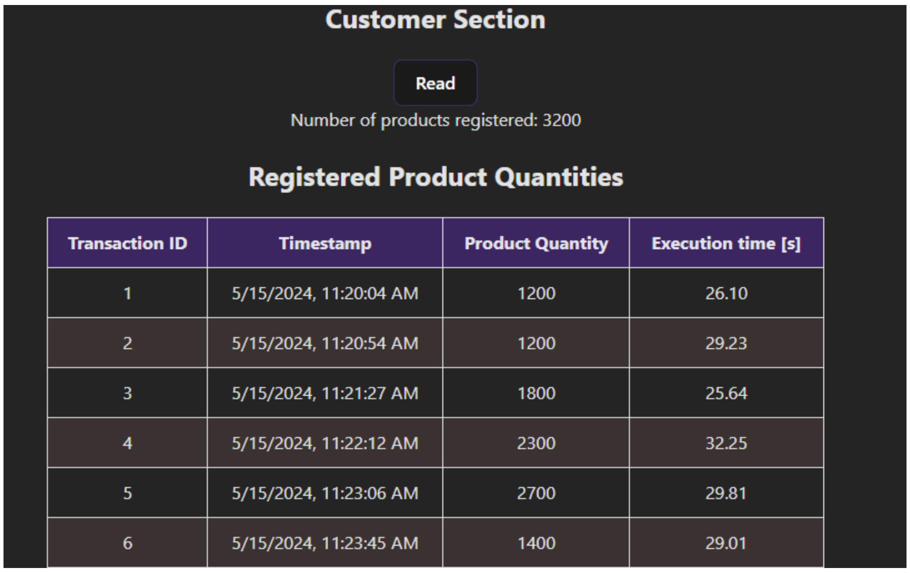
After description and authentication section a simple register section has been added for the delivered product provider. The provider will enter a number in the box in order to be able to continue with the store procedure in the application as shown in Figure 25.


Provider Section

Register number of products delivered:

Figure 25:Provider section in the dApp.

The customer section in the application is focused on reading the previously stored data. At the same time, every time read option is selected the data will be displayed both in the page and in table as it can be seen in Figure 26. Each row will be added when a new transaction is executed.


Customer Section

Read

Number of products registered: 3200

Registered Product Quantities

Transaction ID	Timestamp	Product Quantity	Execution time [s]
1	5/15/2024, 11:20:04 AM	1200	26.10
2	5/15/2024, 11:20:54 AM	1200	29.23
3	5/15/2024, 11:21:27 AM	1800	25.64
4	5/15/2024, 11:22:12 AM	2300	32.25
5	5/15/2024, 11:23:06 AM	2700	29.81
6	5/15/2024, 11:23:45 AM	1400	29.01

Figure 26: Customer section of the dApp.

In Figure 27 the graph visualization of stored transactions is presented. A bar chart has been used in order to get a visual view of the delivery product quantity in each transaction.

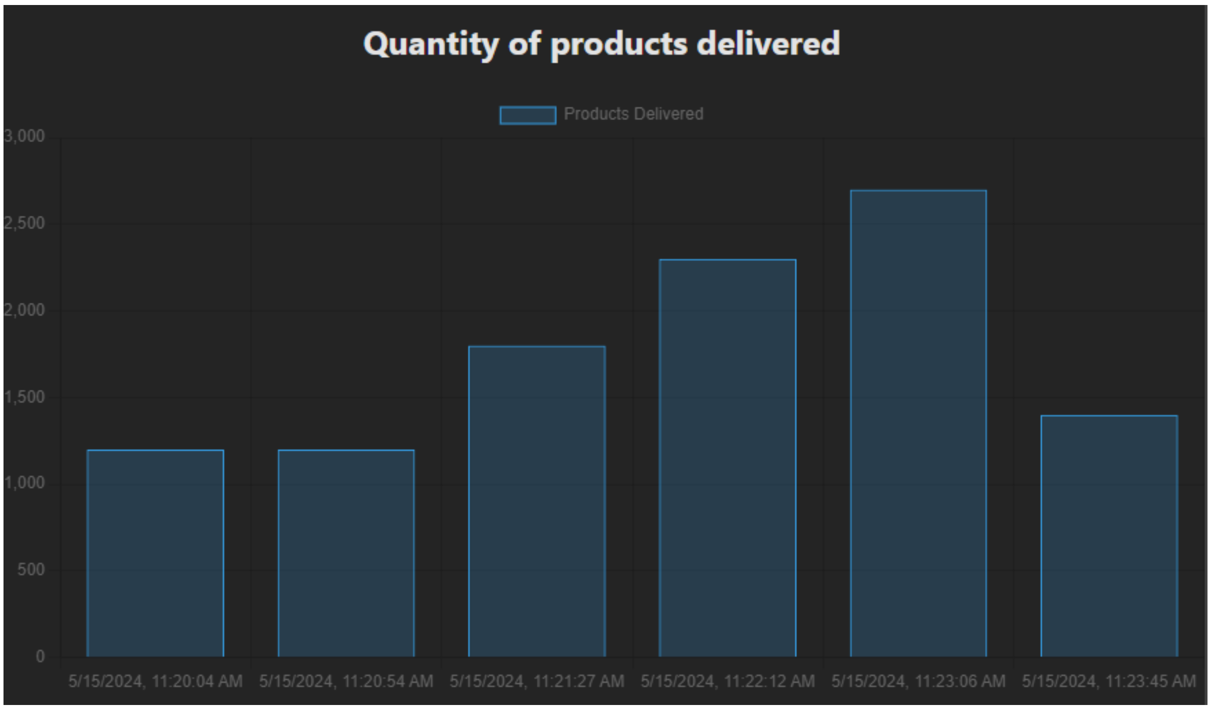


Figure 27: Plotting of the stored transactions on Vechain.

5.5 Testing of the dApp

The testing and debugging phase of the developed decentralized application on the VechainThor blockchain has been crucial for ensuring its reliability and functionality. This chapter will focus on how this testing procedure was executed.

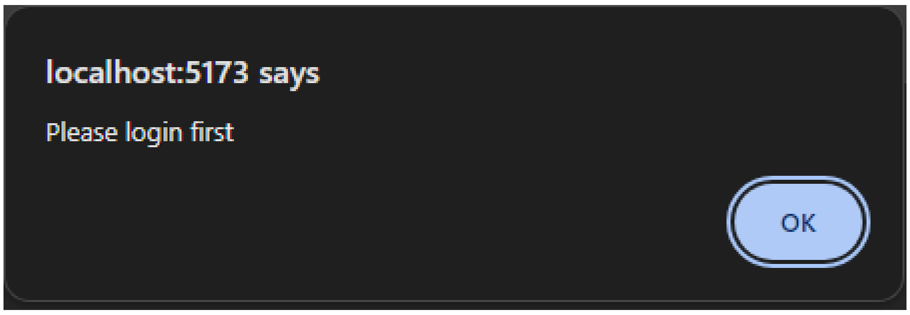
Various types of tests have been carried out to check the designed dApp in an appropriate way. Carried tests can be listed as:

1. Unit Tests: Small parts of the dApp have been tested to make sure they worked correctly on their own. Each use case has been tested first independently before configuring them and making the proper connections between the different use cases.
2. Integration Testing: We checked how different parts of the dApp work together to ensure they operate well as a whole.
3. System Testing: The entire dApp has been tested to see if it meets all the requirements and work smoothly.

Each test helps during the development process to analyze and examine different aspects of the dApp.

5.5.1 Warning security messages

During the testing process, the functionalities of the dApp has been tested one by one to see if they fill the use case requirements. Some improvement has been added to the application in form of warning messages when the user login is still not completed. In Figure 28 is shown how the warning error is displayed in the application.

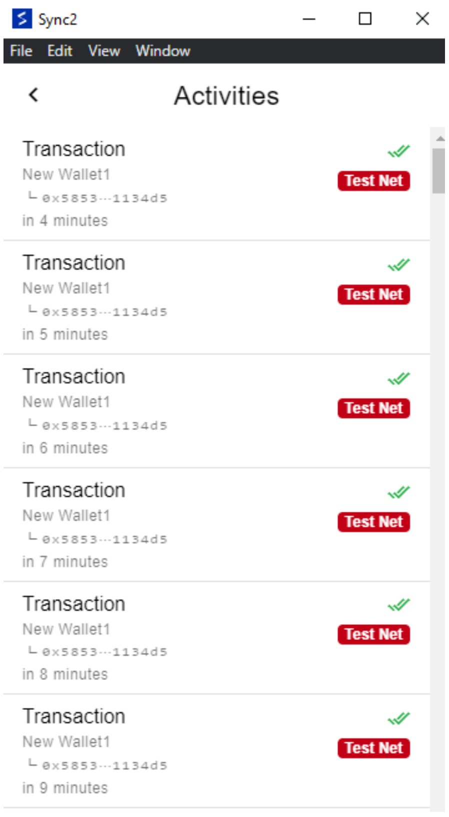

localhost:5173 says

Please login first

Figure 28: Warning message user login is not completed.

5.6 Traceability in Vechain

All the transactions that executed in the designed dApps, are stored in the VechainThor blockchain. In Figure 29, it is showed how all the previously executed transactions in Figure 26 are registered. In Figure 30, the transaction name can be seen, and it is the same as was configured on the storing process.



Transaction
New Wallet1
└ 0x5853…1134d5
in 4 minutes
Test Net

Transaction
New Wallet1
└ 0x5853…1134d5
in 5 minutes
Test Net

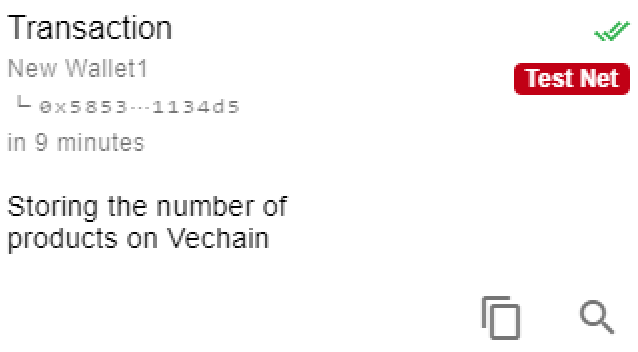
Transaction
New Wallet1
└ 0x5853…1134d5
in 6 minutes
Test Net

Transaction
New Wallet1
└ 0x5853…1134d5
in 7 minutes
Test Net

Transaction
New Wallet1
└ 0x5853…1134d5
in 8 minutes
Test Net

Transaction
New Wallet1
└ 0x5853…1134d5
in 9 minutes
Test Net

Figure 29: Registered transactions on the VechainThor blockchain.

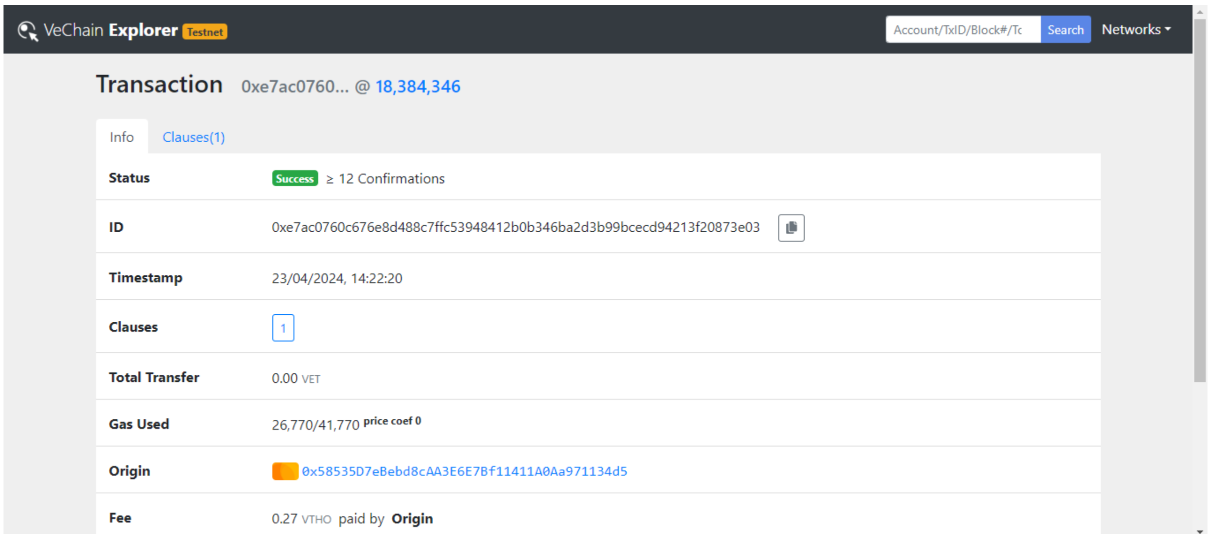


Transaction
New Wallet1
└ 0x5853…1134d5
in 9 minutes
Test Net

Storing the number of products on Vechain

Figure 30: Stored transaction description.

From the same Sync2 wallet, transaction details can be requested and consulted in the Vechain explorer as shown in Figure 31. The Vechain explorer gives relevant details like status, Id, timestamp, used gas or transaction fee.

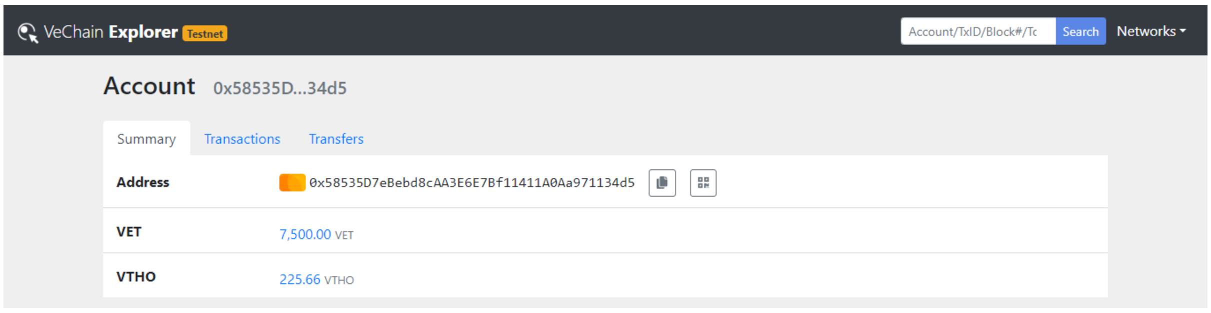


Transaction 0xe7ac0760... @ 18,384,346

Status	Success ≥ 12 Confirmations
ID	0xe7ac0760c676e8d488c7ffc53948412b0b346ba2d3b99bcecd94213f20873e03
Timestamp	23/04/2024, 14:22:20
Clauses	1
Total Transfer	0.00 VET
Gas Used	26,770/41,770 price coef 0
Origin	0x58535D7eBebd8cAA3E6E7Bf11411A0Aa971134d5
Fee	0.27 VTHO paid by Origin

Figure 31: Stored transaction in Vechain explorer.

By selecting the origin of the transaction, the Vechain explorer sends the user to the account page, where the user address and tokens can be checked as shown in Figure 32.

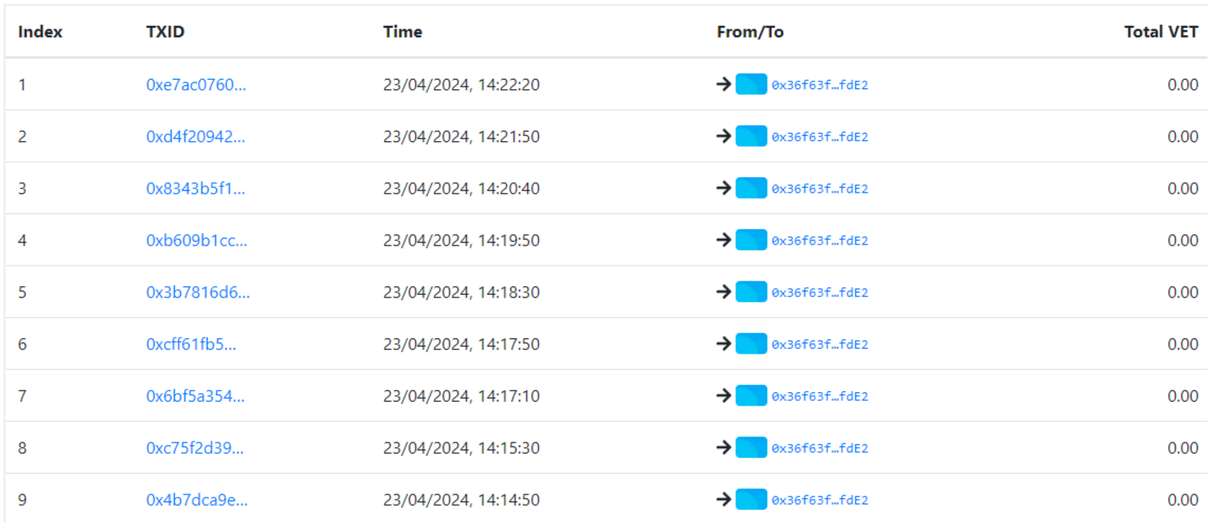


Account 0x58535D...34d5

Address	0x58535D7eBebd8cAA3E6E7Bf11411A0Aa971134d5
VET	7,500.00 VET
VTHO	225.66 VTHO

Figure 32: Transaction origin.

To see all the transactions made by this account transaction, the button can be selected and there the executed transaction list can be seen as presented in Figure 33.



Index	TXID	Time	From/To	Total VET
1	0xe7ac0760...	23/04/2024, 14:22:20	→ 0x36f63f…fdE2	0.00
2	0xd4f20942...	23/04/2024, 14:21:50	→ 0x36f63f…fdE2	0.00
3	0x8343b5f1...	23/04/2024, 14:20:40	→ 0x36f63f…fdE2	0.00
4	0xb609b1cc...	23/04/2024, 14:19:50	→ 0x36f63f…fdE2	0.00
5	0x3b7816d6...	23/04/2024, 14:18:30	→ 0x36f63f…fdE2	0.00
6	0xcff61fb5...	23/04/2024, 14:17:50	→ 0x36f63f…fdE2	0.00
7	0x6bf5a354...	23/04/2024, 14:17:10	→ 0x36f63f…fdE2	0.00
8	0xc75f2d39...	23/04/2024, 14:15:30	→ 0x36f63f…fdE2	0.00
9	0x4b7dca9e...	23/04/2024, 14:14:50	→ 0x36f63f…fdE2	0.00

Figure 33: Executed transaction list in Vechain explorer.

5.7 Performance of dApp

The performance of the dApp has been analyzed during the project. To get a good scope of the performance of the application, the execution time of every transaction with Vechain blockchain is measured using “performance.now()” function. The Figure 34 shows the stored transactions during the performance test.

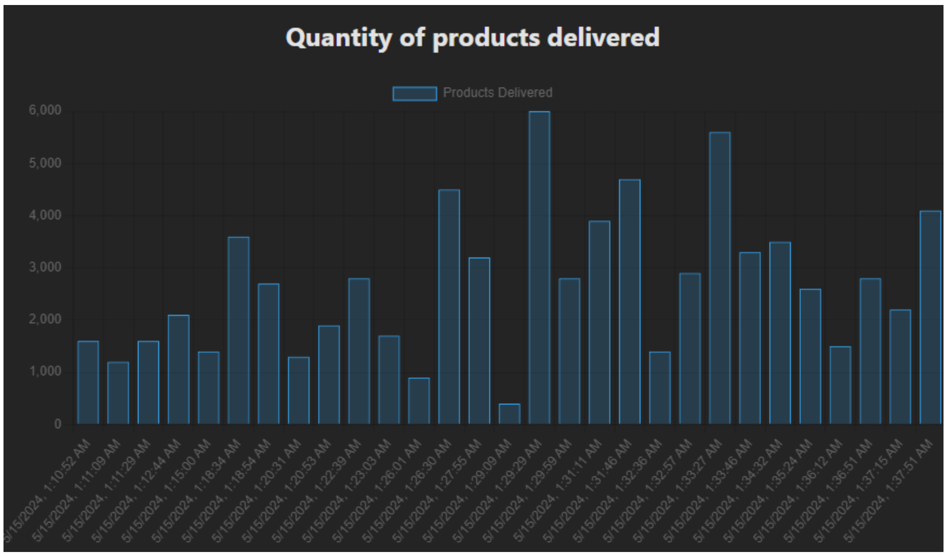


Figure 34:Performance test transactions registered in the dApp.

In Figure 35 a graph can be seen with the different execution times for every transaction. The execution time for each transaction usually varies with the speed that user uses for completing the transaction. During the performance test the time spent by the user has been fixed to get a more realistic results on the execution time of transactions. The execution time has been measured in 30 different transactions. The average execution time has been estimated in 11.56 seconds.

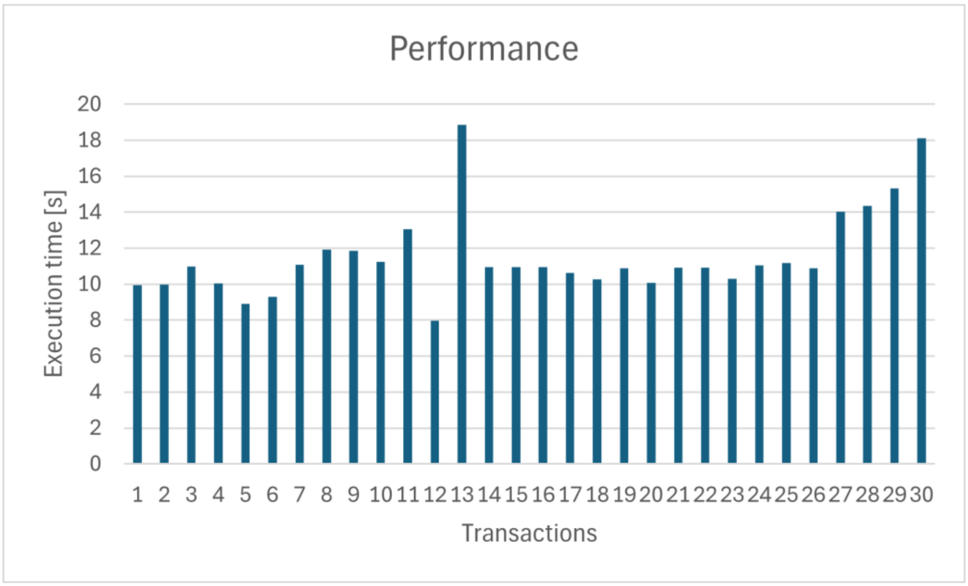


Figure 35: Performance graph of the dApp.

6 Discussion

In this report, the main features of distributed ledger technologies have been explained and an overview of the existing different technologies has been given. After focusing on Vechain blockchain and describing its features, the development of a decentralized application (dApp) has been given with a main focus on the connection between the user and the distributed ledger technology. While decentralized applications (dApps) on the VechainThor blockchain offer numerous benefits, they are also subject to certain safety disadvantages and limitations that required consideration.

One of the primary safety concerns associated with dApps is the presence of smart contract vulnerabilities. Smart contracts deployed on the blockchain are susceptible to coding errors, bugs, and vulnerabilities that could be exploited by malicious actors. These vulnerabilities can end in financial losses, data breaches, and system fracture if not properly addressed. Developers must conduct thorough code audits, testing, and security assessments to identify and mitigate potential vulnerabilities in smart contracts.

During the development phase of the dApp, technical challenges were faced with the deployment of the smart contract. It was important to design a good smart contract and to deploy it correctly. This involved testing it thoroughly to avoid any issues. Making sure it worked well with the blockchain network and following good coding practices were essential. Additionally, ensuring the dApp had a user-friendly interface, managing gas fees effectively and giving a good visualization for customer were significant challenges. Integration with other decentralized services and ensuring data privacy and security also required careful consideration.

Privacy is another area of concern for dApps on the VechainThor blockchain. While blockchain technology offers transparency and immutability, it also increases privacy concerns due to the public nature of transaction data, transaction details are available for everyone. Public blockchains like VechainThor store transaction data openly on the network, exposing sensitive information to unauthorized users. This lack of privacy can be problematic when the application where to record deliveries contain sensitive data like personal information, financial transactions, or proprietary business data.

Scalability remains a significant challenge for dApps on the VechainThor blockchain, impacting their performance and user experience. As the number of users and transactions increases, the network may experience congestion, delays, and higher transaction fees. Scalability limitations can affect the adoption and usability of dApps, particularly for applications requiring high transaction throughput and real-time responsiveness. Implementing scalable is crucial to address these scalability challenges and ensure the long-term viability of dApps on the VechainThor blockchain.

7 Conclusion and future work

The aim of this master thesis is to explore the use of distributed ledger technology (DLT), specifically the Vechain blockchain, for improving the tracking and authentication of product deliveries within supply chain management. The goal was to design and develop a decentralized application (dApp) capable of securely recording the quantity of products delivered from a provider to a customer. The objective has been achieved by creating a decentralized application dApp that enables transparent recording of delivery data. Moreover, a good overview of some of the existing distributed ledger technologies has provided.

The application has been successfully designed identifying all the necessary requirements in the analysis section. It has developed by implementing smart contracts to manage delivery number of products data securely on the VechainThor blockchain. Integrating the frontend interface with the blockchain allowed to exchange the product quantity of a provider-customer delivery.

The results demonstrate the practicability and effectiveness of blockchain in improving delivery tracking and authentication. The dApp provides a secure and decentralized solution, reducing risks of fraud and data manipulation. Moving forward, improving security measures, addressing scalability challenges, and incorporating user feedback will be key priorities. This iterative approach will ensure continuous improvement of the functionality and user experience inside the application.

As future work, different distributed ledgers can be implemented and compared in terms of performance, security and scalability. The obtained results in performance with the designed dApp of 11.56 seconds of average execution time is expected compared with confirmation time in IOTA is between 10-12 seconds [32]. However, they are different technologies, and a better performance test should be executed in order to get a better scope of the whole performance of the application.

In summary, this thesis helps in the foundation for blockchain-based solutions in supply chain management. By analyzing blockchain technology, the potential to increase transparency and efficiency has been shown in delivery processes, with opportunities for further advancements in the future. The developed application can be improved in designing a optimize smart contract with more information regarding the delivery.

References

[1] “Vechain - Blockchain for a Better World.” Accessed: Apr. 18, 2024. [Online]. Available: https://www.vechain.org/

[2] “Blockchain.” Accessed: Apr. 18, 2024. [Online]. Available: https://www.blockchain.com

[3] “IOTA.” Accessed: Apr. 18, 2024. [Online]. Available: https://www.iota.org

[4] “History of Distributed Ledger Technology: The DLT Evolution.” Accessed: Mar. 28, 2024. [Online]. Available: https://imiblockchain.com/history-of-distributed-ledger-technology/

[5] “Overview of a Prototype for Application of Blockchain Supply Chain for Traceability using VeChain, IoT, Sigfox as LPWAN.” Accessed: Apr. 19, 2024. [Online]. Available: https://tinyurl.com/ubv8u3yn

[6] “What is Blockchain? Definition, Examples and How it Works | TechTarget,” CIO. Accessed: Apr. 16, 2024. [Online]. Available: https://www.techtarget.com/searchcio/definition/blockchain

[7] I. Weber, “Introduction and Background: Blockchain and Smart Contracts,” 2021, pp. 1–11. doi: 10.1007/978-3-030-81409-0_1.

[8] “Smart Contracts in Blockchain,” GeeksforGeeks. Accessed: Apr. 24, 2024. [Online]. Available: https://www.geeksforgeeks.org/smart-contracts-in-blockchain/

[9] H. Hellani, L. Sliman, M. B. Hassine, A. E. Samhat, E. Exposito, and M. Kmimech, “Tangle the Blockchain: Toward IOTA and Blockchain Integration for IoT Environment,” in *Hybrid Intelligent Systems*, vol. 1179, A. Abraham, S. K. Shandilya, L. Garcia-Hernandez, and M. L. Varela, Eds., in Advances in Intelligent Systems and Computing, vol. 1179. , Cham: Springer International Publishing, 2021, pp. 429–440. doi: 10.1007/978-3-030-49336-3_42.

[10]A. Rawat, V. Daza, and M. Signorini, “Offline Scaling of IoT Devices in IOTA Blockchain,” *Sensors*, vol. 22, no. 4, Art. no. 4, Jan. 2022, doi: 10.3390/s22041411.

[11]“Introduction to vechain.” Accessed: Mar. 11, 2024. [Online]. Available: https://docs.vechain.org/introduction-to-vechain

[12]IvanOnTech, “What is VeChain, VeChainThor, the VET Token and VTHO Token?,” Moralis Academy. Accessed: Apr. 01, 2024. [Online]. Available: https://academy.moralis.io/blog/what-is-vechain-vechainthor-the-vet-token-and-vtho-token

[13]“Vechain - Definition, What is Vechain, Advantages of Vechain, and Latest News,” cleartax. Accessed: Apr. 15, 2024. [Online]. Available: https://cleartax.in/glossary/vechain

[14]S. Ghosal, “VeChain Cryptocurrency - What It Is, Pros, How To Mine & Trade?,” WallStreetMojo. Accessed: Apr. 24, 2024. [Online]. Available: https://www.wallstreetmojo.com/vechain-cryptocurrency/

[15]“Vechain | VechainThor, the Low-carbon, Highly Scalable Smart Contract Platform,” vechain. Accessed: Apr. 02, 2024. [Online]. Available: https://www.vechain.org/vechainthor/

[16]“About the vechain blockchain | VECHAIN Docs.” Accessed: Apr. 01, 2024. [Online]. Available: https://docs.vechain.org/introduction-to-vechain/about-the-vechain-blockchain

[17] “VeChain Development Plan and Whitepaper (Medium Format),” Medium. Accessed: Apr. 25, 2024. [Online]. Available: https://medium.com/@bsc44/vechain-development-plan-whitepaper-bccc3c255082

[18]“Cryptocurrency Wallet: What It Is, How It Works, Types, Security,” Investopedia. Accessed: Apr. 25, 2024. [Online]. Available: https://www.investopedia.com/terms/b/bitcoin-wallet.asp

[19]“Sync Outlook and Google Calendar and Contacts. Sync2 - Synchronize Microsoft Outlook on multiple PCs without a server.” Accessed: Apr. 25, 2024. [Online]. Available: https://www.sync2.com/

[20]Z. Chen, “Design, development, and deployment of decentralized applications,” *Appl. Comput. Eng.*, vol. 48, no. 1, pp. 46–52, Mar. 2024, doi: 10.54254/2755-2721/48/20241132.

[21]“What Is a Decentralized App? | Definition from TechTarget,” IoT Agenda. Accessed: Apr. 22, 2024. [Online]. Available: https://www.techtarget.com/iotagenda/definition/blockchain-dApp

[22]C. Larman, *An Introduction to Object-Oriented Analysis and Design and the Unified Process*, 2nd ed. 2005.

[23]“Remix - Ethereum IDE.” Accessed: Apr. 08, 2024. [Online]. Available: https://remix.ethereum.org/#lang=en&optimize=false&runs=200&evmVersion=null&version=soljson-v0.8.22+commit.4fc1097e.js

[24]“Vite.” Accessed: Apr. 23, 2024. [Online]. Available: https://vitejs.dev

[25]“HTML For Beginners The Easy Way: Start Learning HTML & CSS Today ».” Accessed: Apr. 15, 2024. [Online]. Available: https://html.com/

[26]“Learn JavaScript Online - Courses for Beginners - javascript.com.” Accessed: Apr. 15, 2024. [Online]. Available: https://www.javascript.com/

[27]“Visual Studio Code - Code Editing. Redefined.” Accessed: Apr. 08, 2024. [Online]. Available: https://code.visualstudio.com/

[28]“Inspector.” Accessed: Apr. 11, 2024. [Online]. Available: https://inspector.vecha.in/#/contracts

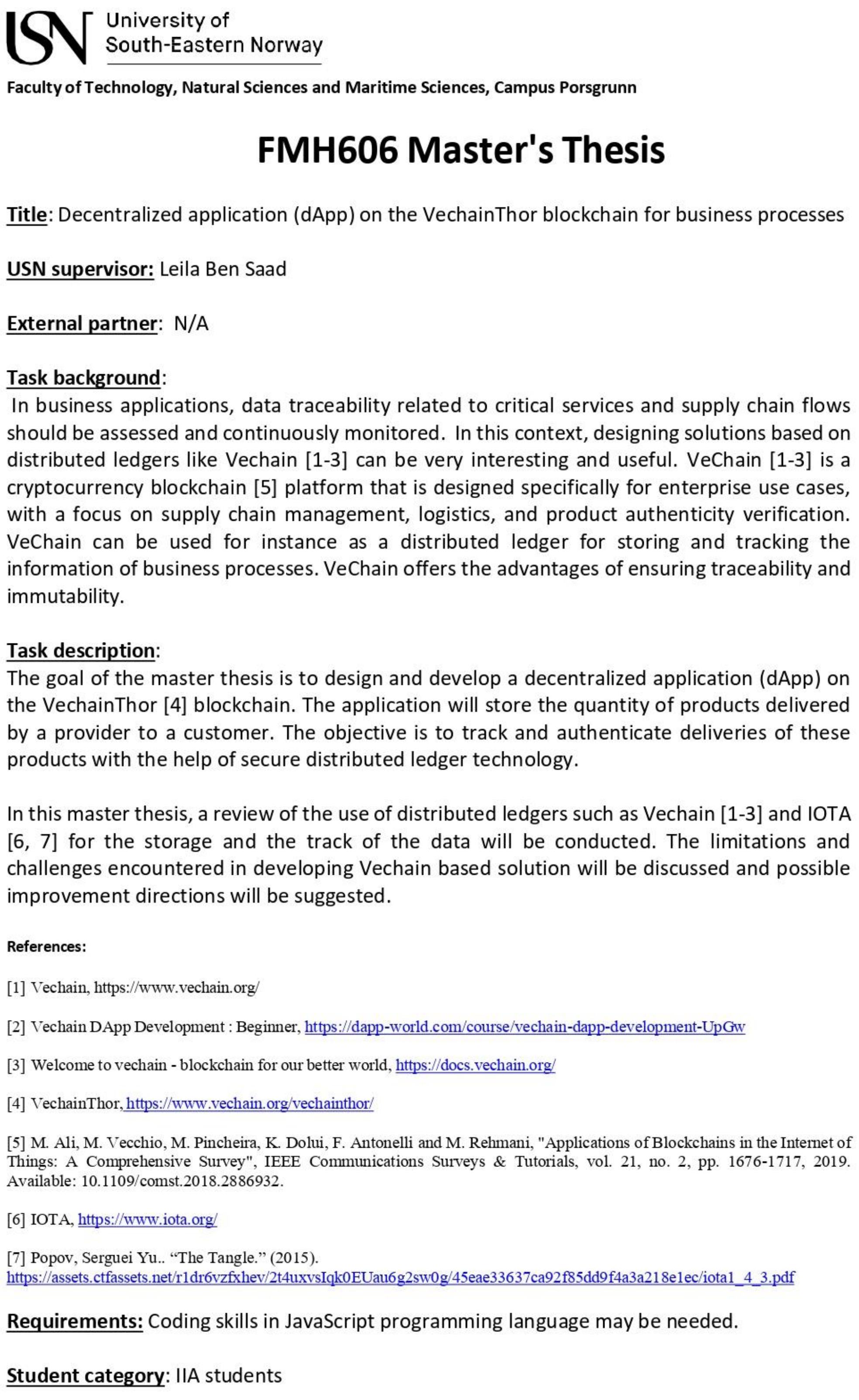
[29]“Connex | VECHAIN Docs.” Accessed: Apr. 15, 2024. [Online]. Available: https://docs.vechain.org/developer-resources/sdks-and-providers/connex

[30]“DApp World | Master Blockchain Programming,” DApp World. Accessed: Apr. 23, 2024. [Online]. Available: https://dapp-world.com/images/og/index.png

[31]DApp-World, “DApp-World/DApp-development-on-VeChain-Beginner.” Mar. 14, 2024. Accessed: Apr. 23, 2024. [Online]. Available: https://github.com/DApp-World/DApp-development-on-VeChain-Beginner

[32]“What is IOTA? How does it differ from Blockchain?,” https://www.iota-services.com/. Accessed: May 15, 2024. [Online]. Available: https://www.iota-services.com/what-is-iota/

Appendix A: Master thesis desccription



University of South-Eastern Norway

Faculty of Technology, Natural Sciences and Maritime Sciences, Campus Porsgrunn

FMH606 Master's Thesis

Title: Decentralized application (dApp) on the VechainThor blockchain for business processes

USN supervisor: Leila Ben Saad

External partner: N/A

Task background:
In business applications, data traceability related to critical services and supply chain flows should be assessed and continuously monitored. In this context, designing solutions based on distributed ledgers like Vechain [1-3] can be very interesting and useful. VeChain [1-3] is a cryptocurrency blockchain [5] platform that is designed specifically for enterprise use cases, with a focus on supply chain management, logistics, and product authenticity verification. VeChain can be used for instance as a distributed ledger for storing and tracking the information of business processes. VeChain offers the advantages of ensuring traceability and immutability.

Task description:
The goal of the master thesis is to design and develop a decentralized application (dApp) on the VechainThor [4] blockchain. The application will store the quantity of products delivered by a provider to a customer. The objective is to track and authenticate deliveries of these products with the help of secure distributed ledger technology.

In this master thesis, a review of the use of distributed ledgers such as Vechain [1-3] and IOTA [6, 7] for the storage and the track of the data will be conducted. The limitations and challenges encountered in developing Vechain based solution will be discussed and possible improvement directions will be suggested.

References:

[1] Vechain, https://www.vechain.org/

[2] Vechain DApp Development : Beginner, https://dapp-world.com/course/vechain-dapp-development-UpGw

[3] Welcome to vechain - blockchain for our better world, https://docs.vechain.org/

[4] VechainThor, https://www.vechain.org/vechainthor/

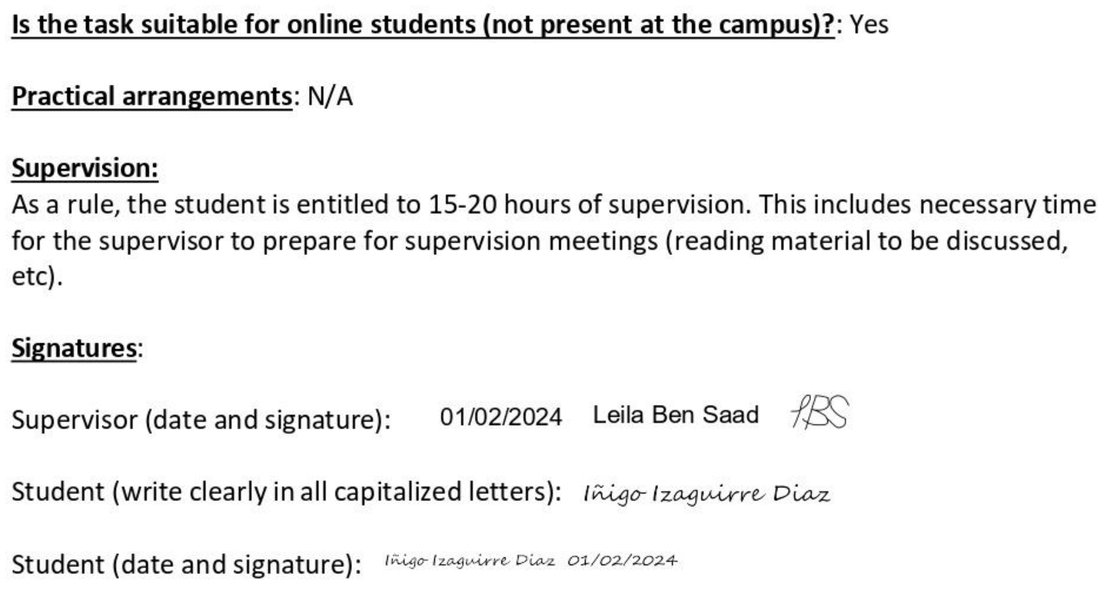
[5] M. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli and M. Rehmani, "Applications of Blockchains in the Internet of Things: A Comprehensive Survey", IEEE Communications Surveys & Tutorials, vol. 21, no. 2, pp. 1676-1717, 2019. Available: 10.1109/comst.2018.2886932.

[6] IOTA, https://www.iota.org/

[7] Popov, Serguei Yu.. “The Tangle.” (2015). https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf

Requirements: Coding skills in JavaScript programming language may be needed.

Student category: IIA students



Is the task suitable for online students (not present at the campus)?: Yes

Practical arrangements: N/A

Supervision:
As a rule, the student is entitled to 15-20 hours of supervision. This includes necessary time for the supervisor to prepare for supervision meetings (reading material to be discussed, etc).

Signatures:

Supervisor (date and signature): 01/02/2024 Leila Ben Saad [illegible]

Student (write clearly in all capitalized letters): Iñigo Izaguirre Diaz

Student (date and signature): Iñigo Izaguirre Diaz 01/02/2024

Appendix B: Github repository

Link to the repository: https://github.com/inigoIzaguirre21/Vechain_dApp_project.