National Technical University of Athens
School of Electrical and Computer Engineering
Division of Computer Science

Implementation of Blockchain Application for managing University grades

Diploma Thesis

of

HADJICHRISTOFI CHRISTOS

Supervisor: Vassilios Vescoukis
Professor

Athens, June 2022

National Technical University of Athens
School of Electrical and Computer Engineering
Division of Computer Science

Implementation of Blockchain Application for managing University grades

Diploma Thesis

of

HADJICHRISTOFI CHRISTOS

Supervisor: Vassilios Vescoukis
Professor

Approved by the examination committee on June 21st.

(Signature)	*(Signature)*	*(Signature)*
.....	.....	.....
Vassilios Vescoukis	Fotakis Dimitrios	Aris Pagourtzis
Professor	Associate Professor	Professor

Athens, June 2022

National Technical University of Athens
School of Electrical and Computer Engineering
Division of Computer Science

Copyright © – All rights reserved.
Hadjichristofi Christos, 2022.

The copying, storage and distribution of this diploma thesis, exall or part of it, is prohibited for commercial purposes. Reprinting, storage and distribution for non - profit, educational or of a research nature is allowed, provided that the source is indicated and that this message is retained.

The content of this thesis does not necessarily reflect the views of the Department, the Supervisor, or the committee that approved it.

(Signature)

.....
Hadjichristofi Christos

June

Abstract

A Blockchain is a distributed and decentralized immutable public ledger, that exists across a network. The Blockchain acts as a distributed database that is shared among the nodes of a network. More specifically, it stores information in digital format that can not be edited. Thus, it guarantees the integrity and security of any record of data and therefore, is capable of working without the need of a trusted third party. It is a well-known technology which bloomed over the past years, due to its capability to maintain transactions securely.

This diploma thesis aims to the development of a private Blockchain network with the respective smart contracts and front-end to manage University grades. The problem that this thesis solves, will be addressed and a complete solution will be discussed, among with concepts and basic principles like Blockchain, smart contracts, consensus algorithms but also with the difficulties that were encountered during the implementation of this system.

Finally, it is concluded that a system of such capability can be developed, deployed, exist and co-operate with any other legacy system and offer all the perks of Blockchain technology. In addition, ideas for further expansions of the examined system are discussed, along with the benefits that they may offer.

Keywords

Blockchain, Ethereum, Go Ethereum, Private Blockchain Network, Smart Contracts, Ledger, Chaincode, Decentralized Applications, Integrity, Security

Acknowledgements

After completing my thesis, my five-year trip to the School of Electrical and Computer Engineering at the National Technical University of Athens (NTUA) comes to an end. This would not be possible without my family and friends, which supported me throughout my undergraduate studies.

I would like to express my gratitude to my professor and supervisor Mr Vassilis Vescoukis, for the opportunity to work and get a hands-on experience on this subject, but also for his valuable guidance, feedback and excellent cooperation during my thesis.

In addition, I wish to acknowledge the help provided by the PhD Candidate and Researcher Giannis Tzannetos, who assisted me throughout my thesis, to understand the subject in depth.

Finally, I wish to extend my special thanks to all the people, which were part of this journey at the National Technical University of Athens.

Athens, June 2022

Hadjichristofi Christos

Table of Contents

Abstract 1

Acknowledgements 3

1 Introduction 13
- 1.1 Motivation . . . 13
- 1.2 Thesis structure . . . 14

2 Blockchain Basics 17
- 2.1 What is a Block . . . 17
- 2.2 What is a Node . . . 17
- 2.3 What is a Miner . . . 17
- 2.4 What is a Blockchain . . . 18
- 2.5 Blockchain Network types . . . 18
- 2.6 Consensus algorithms . . . 19
  - 2.6.1 Proof of Work (PoW) . . . 20
  - 2.6.2 Proof of Stake (PoS) . . . 20
  - 2.6.3 Proof of Authority (PoA) . . . 20
  - 2.6.4 Comparison between Consensus Mechanisms . . . 21
- 2.7 Smart contracts . . . 21
- 2.8 Pros and Cons of using Blockchain technology . . . 22

3 Use case study 23
- 3.1 Problem explanation . . . 23
- 3.2 A Blockchain-based approach . . . 25
  - 3.2.1 Privacy . . . 26
  - 3.2.2 Validation and Integrity . . . 26
  - 3.2.3 User Interface . . . 26
- 3.3 A private-permissioned Blockchain approach, based on Ethereum . . . 27
- 3.4 Use Case Model . . . 29
  - 3.4.1 Registrar Users . . . 29
  - 3.4.2 Master Users . . . 29
- 3.5 UML High-level work flow Activity Diagram . . . 30
- 3.6 Wireflow . . . 31
- 3.7 Login and Different Users . . . 32
  - 3.7.1 Description . . . 32

3.7.2 What is MetaMask . . . 32
3.7.3 UML Activity Diagram . . . 32
3.7.4 Login - Metamask Lo-Fi Wireframe . . . 33
3.7.5 Menu - Master Lo-Fi Wireframe . . . 33
3.7.6 Menu - User Lo-Fi Wireframe . . . 34
3.8 Use Case 1: Add Grades . . . 35
3.8.1 Description . . . 35
3.8.2 UML Activity Diagram - Professor generate and distribute file . . . 35
3.8.3 UML Activity Diagram - Add Grades . . . 35
3.8.4 UML Sequence Diagram . . . 36
3.8.5 UML Auxiliary Sequence Diagram . . . 36
3.8.6 Lo-Fi Wireframe . . . 37
3.9 Use Case 2: Show Course Information and Validate . . . 38
3.9.1 Description . . . 38
3.9.2 UML Activity Diagram . . . 38
3.9.3 UML Sequence Diagram . . . 39
3.9.4 Show Courses LoFi Wireframe . . . 39
3.9.5 Show Course Information LoFi Wireframe . . . 40
3.9.6 Show Diff Output LoFi Wireframe . . . 40
3.10 Use Case 3: Vote for new Users . . . 41
3.10.1 Description . . . 41
3.10.2 UML Activity Diagram . . . 41
3.10.3 UML Sequence Diagram . . . 42
3.10.4 Lo-Fi Wireframe . . . 42
3.11 Use Case 4: Start a new Vote . . . 43
3.11.1 Description . . . 43
3.11.2 UML Activity Diagram . . . 43
3.11.3 UML Sequence Diagram . . . 44
3.11.4 Lo-Fi Wireframe . . . 44
3.12 Use Case 5: Show Users . . . 45
3.12.1 Description . . . 45
3.12.2 UML Activity Diagram . . . 45
3.12.3 UML Sequence Diagram . . . 45
3.12.4 Lo-Fi Wireframe . . . 46

4 Implementation: Difficulties encountered and different approaches 47
4.1 Setting up the private Ethereum network . . . 47
4.2 First Implementation: A Hello World dApp . . . 47
4.2.1 UML Component Diagram . . . 48
4.3 Second Implementation: Grades System dApp . . . 48
4.3.1 Technologies Used . . . 48
4.3.2 Issues . . . 48
4.3.3 UML Component Diagram . . . 49

4.4 Third Implementation: Grades System dApp - NextJS 49
4.4.1 Issues 49
4.4.2 UML Component Diagram 50
4.4.3 UML Deployment Diagram 50

5 Implementation: Complete guide of the decentralized application 51
5.1 Create the Ethereum private Blockchain network 51
5.1.1 Install required software 51
5.1.2 Steps - Explanation 52
5.1.3 Steps - Execution 53
5.2 Determine dApp development technology 58
5.2.1 Back-End 58
5.3 Implement the smart contract 59
5.3.1 Smart Contract UML Class Diagram 60
5.3.2 Structures and Variables 61
5.3.3 Functions 62
5.3.4 Compile and Migrate Smart Contracts 62
5.4 Implement the dApp Front-End 65
5.4.1 Structure NextJS project 65
5.4.2 Integrate MetaMask 68
5.5 Dummy .bau files Generator 72

6 Demonstration of the dApp 73
6.1 Login 73
6.1.1 Main Page 73
6.1.2 Select Account 74
6.1.3 Accept or Reject connection with MetaMask 74
6.2 Add Grades 75
6.2.1 Complete form 75
6.2.2 MetaMask prompt 75
6.2.3 Transaction confirmed 76
6.3 Show Courses 77
6.3.1 Registrar User 77
6.3.2 Master User 77
6.3.3 Retrieve information of specific course 78
6.3.4 Validate information of a record 78
6.4 Add a new user 80
6.4.1 Start a new vote - Complete form 80
6.4.2 MetaMask prompt 80
6.4.3 Transaction confirmed 81
6.5 Vote for or against 82
6.5.1 Retrieve pending votes 82
6.5.2 MetaMask prompt 82

6.5.3 Transaction confirmed 83
6.6 Show Users 83

7 Epilogue **85**
7.1 Privacy 85
7.2 Integrity and Security 85
7.3 Immediacy of procedures 85
7.4 Further expansions 86

Bibliography **91**

List of Abbreviations **93**

List of Figures

2.1 Blockchain representation [1] . . . 18
2.2 Types of Blockchain Networks [2] . . . 19

3.1 Legacy System work flow . . . 24
3.2 Legacy System work flow along with the Blockchain System . . . 25
3.3 UML Activity Diagram - Validation flow . . . 26
3.4 Block Diagram . . . 28
3.5 UML Use Case Diagram . . . 29
3.6 UML High-level work flow Activity Diagram . . . 30
3.7 Wireflow diagram . . . 31
3.8 UML Activity Diagram: Login Procedure . . . 32
3.9 Login Wireframe . . . 33
3.10 Menu - Master Wireframe . . . 33
3.11 Menu - User Wireframe . . . 34
3.12 UML Activity Diagram - Procedure of Professor creating and distributing file 35
3.13 UML Activity Diagram: Add Grades . . . 35
3.14 UML Sequence Diagram: Add Grades . . . 36
3.15 UML Sequence Diagram: MetaMask prompt - User accept/reject transaction 36
3.16 Add Grades Wireframe . . . 37
3.17 UML Activity Diagram: Show Course Information and Validate . . . 38
3.18 UML Sequence Diagram: Show Course Information and Validate . . . 39
3.19 Show Courses Wireframe . . . 39
3.20 Show Course Information Wireframe . . . 40
3.21 Show Diff Output Wireframe . . . 40
3.22 UML Activity Diagram: Vote for new Users . . . 41
3.23 UML Sequence Diagram: Vote for new Users . . . 42
3.24 Vote for or Against Wireframe . . . 42
3.25 UML Activity Diagram: Start a new Vote . . . 43
3.26 UML Sequence Diagram: Start a new Vote . . . 44
3.27 Start Vote Wireframe . . . 44
3.28 UML Activity Diagram: Show Users . . . 45
3.29 UML Sequence Diagram: Show Users . . . 45
3.30 Show Participants/Users Wireframe . . . 46

4.1 Hello World dApp UML Component Diagram . . . 48

4.2 Grades System dApp (NodeJS) UML Component Diagram 49
4.3 UML Component Diagram 50
4.4 UML Deployment Diagram 50

5.1 Steps followed to develop a private Blockchain Network 51
5.2 Steps followed to develop the Smart Contract 59
5.3 Smart Contract UML Class Diagram 60
5.4 Steps followed to develop the dApp 65
5.5 NextJS Project Structure 65
5.6 MetaMask Add Network 71

6.1 Login Page 73
6.2 Select account to connect with MetaMask 74
6.3 Accept or reject connection with MetaMask 74
6.4 Complete form 75
6.5 MetaMask prompt 75
6.6 Transaction confirmed 76
6.7 Courses a registrar user can retrieve 77
6.8 Courses a master user can retrieve 77
6.9 Information retrieve for a specific course 78
6.10 Validate information - Diff found 78
6.11 Validate information - No Diff found 79
6.12 Complete form 80
6.13 MetaMask prompt 80
6.14 Transaction confirmed 81
6.15 Transaction confirmed 82
6.16 Vote prompt MetaMask 82
6.17 Transaction confirmed 83
6.18 Retrieve Users - Master User 83

7.1 Add grades and validate production workflow 86
7.2 Students requesting doc in Legacy System vs BC System 87

List of Tables

2.1 Consensus Mechanisms comparison [3] 21
2.2 Pros and cons of using Blockchain Technology 22

5.1 Smart Contract Functions 62

Chapter 1

Introduction

Over the past years, Blockchain has become more and more popular and gained lots of recognition. Blockchain has been proposed as a solution to a vast amount and variety of problems such as supply chain management problems, identity theft, digital copyright and piracy, crowdfunding and fundraising and so on. Needless to say, cryptocurrencies have hugely contributed to the development of Blockchain technology, it is clear, however, that Blockchain is not only about cryptocurrencies. On the contrary, cryptocurrencies are based on only one kind of Blockchains, out of several that can enable applications in many domains. The reason this technology can be so generic and offer solutions to different kinds of problems is due to the integrity and security it can offer, which is done by eliminating the dominant "third trusted party", on which most current approaches for information integrity and security are based. As people saw the potential of this technology and the fact that it can be applied in so many use cases, the enthusiasm and hype regarding this technology kept getting bigger. The hype led people to start experimenting with solutions for their problems with the use of Blockchain without taking into consideration that a better and more sustainable solution could be developed without the use of this particular technology. So, they ended up wasting resources, reaching uncoverable costs and producing a negative hype for Blockchain. [4]

On the other hand, when Blockchain technology is used under the right circumstances and in well-suited use cases, its effects can be beneficial. More specifically, as in the case with all technologies, Blockchain needs to be configured to address each specific problem as effectively as possible. Different types of Blockchain networks can be used for solving different types of problems. This thesis focuses on the design and implementation of a private permissioned blockchain in an application domain that demands high levels of privacy and data integrity, namely the management of university exam grades. As such, Blockchain is well suited for this problem. A more detailed discussion on Blockchain network types is available in Section 2.3.

1.1 Motivation

The management of grades in Universities is one of the most important tasks assigned to Registrars. Apart from the obvious functional requirements, such as managing registrations to courses, storing grades in a database, reporting, etc, it is the non-functional

requirements that usually drive the architecture and design of such applications. Among those, "security" is the dominant requirement which can be further analyzed as follows:

- integrity: grades should only be updated by professors who produce them at the end of courses' delivery. Grade management applications should be structured in a way that no other entity, no matter what their role in the management of the information system is, can modify grade data.

- security: grades should only be visible to specific entities, namely the student and the registrar staff strictly for administrative purposes.

- traceability: any valid modification to grades, such as correction of errors, should be done by authorized personnel and should be traceable, meaning that a complete record of the modification should be kept, with the same integrity and security requirements as with the original grade update.

- multiple copies of critical data should be kept and authorized updates should be propagated in real-time across these copies. Access to the digital asset that implements these copies should not allow decryption on any modification of these data.

- It should be possible to automatically verify any digital asset containing decrypted grades, against the trusted distributed grade repository; this verification should be possible only by authorized personnel.

Considering that legacy grade management systems are already in operation, the above non-functional security requirements should be able to be satisfied without disrupting the operation of these systems. Transparent, in-parallel operation of a service that implements the kind of security requirements discussed above, should be possible. Such a system would provide significant improvements to the level of trust in grade reports; it would also minimize the workload needed to verify grades by more traditional methods.

We claim that Blockchain technologies can be used to implement a service that runs in parallel with any legacy grade management system, and enhances its security by implementing the requirements discussed above. This has been the main motivation behind this work.

1.2 Thesis structure

In Chapter 2 the basic principles of a Blockchain network as well as an introduction to concepts such as transactions, smart contracts, consensus algorithms are presented.

In Chapter 3 the user requirements from a University grades management app will be extracted by examining the typical workflow of the current process at NTUA. Use cases will be defined to address this particular case by adding a Blockchain "on the side" of the current practice. This will be done by following a disciplined software engineering

methodology and documentation, including solicitation of requirements, architecture, design and implementation.

In Chapter 4 the architecture and design of a distributed application (dApp) will be discussed, along with the challenges that have been met during the implementation. Also, comparisons for different alternative approaches to the implementation, that has been considered, will be outlined.

In Chapter 5 the implementation from the technical side will be described. How this private blockchain network is set up, what platform was used and why, how these requirements were implemented and what technologies were used to implement the front-end.

In Chapter 6 a simulation of this decentralized application will be presented via screen-shots of different users logging in to the dApp and interacting with the blockchain either by retrieving or by storing information, according to the use cases of the system.

In the Epilogue proposals for the enhancement of system will be presented. How this system can be expanded but also other problems it could potentially solve efficiently and elegantly. Finally, conclusions of this thesis will be drawn.

Chapter 2

Blockchain Basics

This chapter outlines some very basic principles of blockchain with which the reader should have in mind. Thus, the purpose of this chapter is not to acquire extensive knowledge on the blockchain and the related concepts.

2.1 What is a Block

A block is the main element of a blockchain and each block can be decomposed to three components [5]:

1. The data that will be eventually stored

2. The nonce, which is a 32-bit number. It is randomly generated when a new block is created, which then generated a block header hash.

3. The hash, a 256-bit number wedded to the nonce

2.2 What is a Node

A node is a participant of the network, which has its copy of the ledger. Every electronic device that can maintain a copy of the public ledger and keep the network functioning can be a node of a Blockchain network. The number of participants (nodes) in a Blockchain network is very important. Blockchain networks with many participants (nodes) are often considered more secure, as the security of the network is derived from decentralization. Thus, more nodes offer more integrity [5].

2.3 What is a Miner

A miner node is a participant of the network which sacrifices computing power to solve the Proof-of-Work puzzle to maintain the integrity of the network. These nodes help provide security as well as the required decentralization of the network.

2.4 What is a Blockchain

Blockchain is a distributed system which can store information in a certain way that makes it impossible to change or cheat. It is a digital ledger of transactions, which is duplicated and shared across the network. Every time a new transaction occurs in the network, a new record is added to every participant's ledger. Transactions are recorded in the system with an immutable cryptographic signature (hash). This guarantees the immutability of the public ledger across the network because if a block has been changed, it would be immediately apparent. Therefore, if someone would want to change the ledger, every block of the chain across every participant of the network must be changed, which is impossible, thus the integrity of the data shared is guaranteed [6].

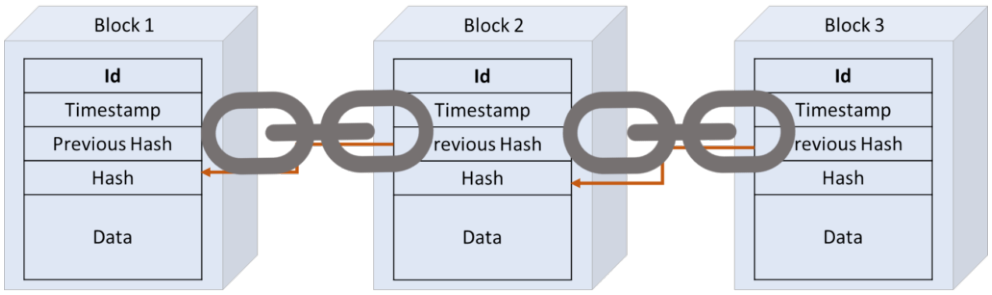


Figure 2.1. *Blockchain representation [1]*

2.5 Blockchain Network types

The first type of Blockchain network is the permissionless network, which allows access to any user to pseudo-anonymously join the network and become a node, without any restrictions. Permissionless networks are also called public networks, and as mentioned above anyone can join. This specific type of network is completely decentralized due to the fact that anyone can become a node of the network. All nodes of a public BC network have equal rigths to access the Blockchain, generate and validate new blocks of data. This particular kind of network is primarily used to exchange and mine cryptocurrencies. Popular public blockchain networks are Bitcoin and Ethereum. On these networks, the nodes mine for cryptocurrencies by creating blocks for the transactions requested on the network by solving cryptographic equations. In return for the resources they offer to the network, miner nodes earn a small amount of cryptocurrencies. The second type of networks, are the permissioned networks, which restrict access to the network to certain nodes. In this type of network there can be rights restrictions. These private blockchain networks are controlled by a single organization that determines who can be a node. Private BC networks are partially decentralized because public access to them is restricted. Examples of private BC networks are Riple and Hyperledger. Due to the drawbacks of the above networks, consortium and hybrid blockchains were developed. Consortium

networks are permissioned blockchain networks governed by a group of organizations. These networks are more decentralized than private blockchains which result in higher levels of security. In contrary, these types of networks can be demanding to set up as it requires cooperation between several organizations. Hybrid blockchain networks are controlled by a single organization but with surveillance of the public blockchain which is required to perform certain transaction validations. An example of hybrid blockchains is the IBM Food Trust which was developed to enchance efficiency throughout the whole food supply chain [2].

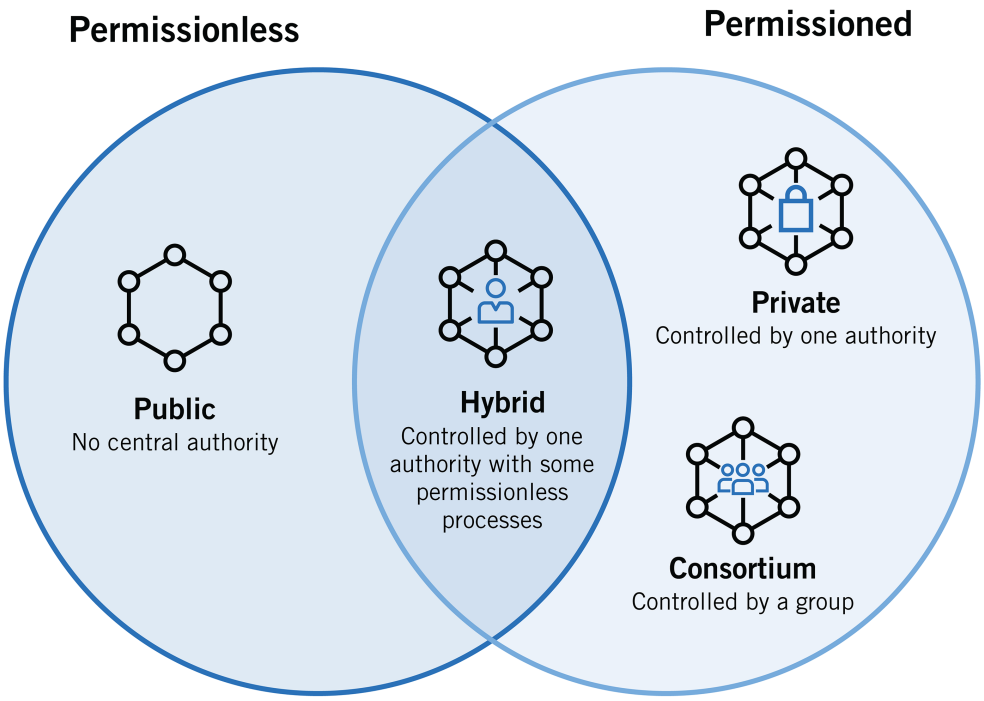


Figure 2.2. *Types of Blockchain Networks [2]*

To sum up, permissionless blockchain networks can be more secure than permissioned blockchains, because in permissionless networks the number of nodes is much greater than in a permissioned blockchain, so it would be more difficult for bad actors to collude on the network. However, in permissionless blockchains, the transaction process time tends to be longer than in a permissioned blockchain network, as more validator nodes exist. Both types of Blockchain networks have their pros and cons, but both are here to offer the same concept, which is the integrity of the public ledger that is shared among the network.

2.6 Consensus algorithms

Consensus algorithms (also known as consensus mechanisms or consensus protocols) provide a secure way for computers that are part of a network (distributed systems) to work together. These kinds of mechanisms have been used over decades to establish

consensus in enterprise infrastructures such as database nodes and servers. In the last few years, new consensus mechanisms have been created and applied to Blockchains to serve cryptoeconomic systems and to agree on the state of the network.

In theory, a malicious user (attacker) can compromise consensus by controlling the 51% of the network, but consensus mechanisms are designed in a way that this attack is practically impossible. There are different kinds of consensus mechanisms that are invented to be used in cryptoeconomic systems [7]. A few typical such mechanisms are discussed in the sequel.

2.6.1 Proof of Work (PoW)

Proof of Work is done by miners, who compete to create new blocks filled with processed transactions. There is a race of who will solve the fastest a mathematical puzzle. This puzzle is the work in proof of work consensus mechanism and produces the cryptographic link between the current block and the previous block. The winner earns some cryptocurrency for the valuable computational power that was offered to the network and the newly mined block is shared to all the participants. The only way to compromise security on a network which uses this consensus algorithm is to own the 51% of the network's computing power, which would led to huge investments in equipment [7].

2.6.2 Proof of Stake (PoS)

Proof of Stake is done by validators who have staked cryptocurrency to participate in the system. For a new block to be created, a random validator node is chosen. When a new block is created, it is shared among the network and the selected validator node earns rewards. This consensus algorithm does not need heavy computational work like the Proof of Work consensus. It simply needs to stake your digital currency in the network. The only way to compromise security on a network is to own the 51% of the total stake cryptocurrency, which would be a great amount of money [7].

2.6.3 Proof of Authority (PoA)

In a permissioned blockchain network all nodes are pre-authenticated which allows to use consensus mechanisms which provide high transaction rate and other benefits. Proof of Authority is one of them and it is an algorithm that provides high performance and fault tolerance through a consensus mechanism based on identity as a stake. So, in PoA only nodes that have proven their identity are awarded to create new blocks and the only way to gain this right is to pass preliminary authentication. This consensus algorithm makes even harder the 51% attack, as the attacker needs to obtain control to 51% of the nodes of the permissioned blockchain network. In addition, PoA mechanism can defend successfully a DoS attack, because all participants that are included to the network are pre-authenticated and so a certain selection can be done. For example only nodes that can withstand a DoS attack could be added as a network participant. Also even if a network node becomes unavailable for a certain time period, it can be removed temporarily as a validator node [8].

2.6.4 Comparison between Consensus Mechanisms

Below, the three consensus mechanisms discussed are compared between their main idea, energy consumption and level of centralization.

Table 2.1. *Consensus Mechanisms comparison [3]*

Consensus Mechanism	**Main Idea**	**Energy Consumption**	**Level of Centralization**
Proof-of-Work (PoW)	Computational power determines the chance to add a new block	High	Low
Proof-Of-Stake (PoS)	Staked wealth determines the chance for adding a new block	Low	Medium
Proof-Of-Authority (PoA)	Only a certain authorized nodes have the ability to add a new block	Low	High

2.7 Smart contracts

A smart contract is an agreement between two parties in the form of computer code. Smart contracts are stored in the blockchain, so they are part of the public ledger and cannot be changed. Smart contracts run on the blockchain and the transactions that happen are processed by it, which means they can be sent automatically without a third party. The transactions only occur when specific conditions are met [9].

Smart contracts first appeared in the Ethereum Blockchain main network. Many other Blockchain networks adopted them and according to the network smart contracts can be written in different languages. For example in Ethereum blockchain networks Solidity the Solidity programming language has been developed to write smart contracts. In Hyperledger Javascript, Go, Java and Solidity.

Furthermore, smart contracts can offer speed and efficiency, as when the conditions are met, the contract will be executed immediately. In addition, they offer trust, transparency and security due to the fact that no third party is involved and the whole process takes place on the blockchain.

2.8 Pros and Cons of using Blockchain technology

Blockchain technology has endless possibilities for use in real-world problems despite its complexity and difficulty to grasp. Of course, getting into details on Blockchain technology, as well as exhaustive information about the benefits and drawbacks of this technology is out of scope in the context of this thesis. More information about Blockchain technology can be found in [10] [11] [12] [13]. A table of some pros and cons of this technology is following.

Pros	Cons
Transparent technology	Limitations on data storage
Harder to tamper data due to decentralization/immutability	Limited number of transactions per second
Accuracy improvement due to the removal of human involvement in verification	Technology cost
Efficient, secure and private transactions	
Stability	
Reduce costs by eliminating third-party verification	
Offers automation (smart contracts)	
Visibility and traceability	

Table 2.2. *Pros and cons of using Blockchain Technology*

Chapter 3

Use case study

3.1 Problem explanation

The present process that manages the grades, which is followed by each school in the university is quite cumbersome and prone to errors. More specifically, the procedure that a registrar follows to eventually add grades to the present legacy system can be described with the following steps:

1. Start enrollment depending on the academic period
2. Students enroll in term and courses
3. As soon as the enrollment system close and no students omitted to enroll an individual from the Computers Center visits the registrar to complete the enrollment in the system
    - In case of any omissions occur during the enrollment are reported in school's general meeting
    - School's general meeting decide for the omissions
    - Registrar manually inserts omitted enrollments
4. BAU files are issued for each course. BAU files are a special file format that eventually holds the students' grades when they are graded by the professor
5. BAU files are sent to each professor
6. Professors fill the respective BAU file with the students' grades and send the file back encrypted to the registrar
7. Registrar uses certain software to register grades to the present system
8. Students are able to see the grade from the respective front-end which retrieves their data
    - In case student's grade needs update due to any error professor needs to contact the registrar
    - Registrar sends the document

- Professor fills it with changes, signs the document and sends it back to the registrar
- Registrar updates manually the respective grades
- Student information updates

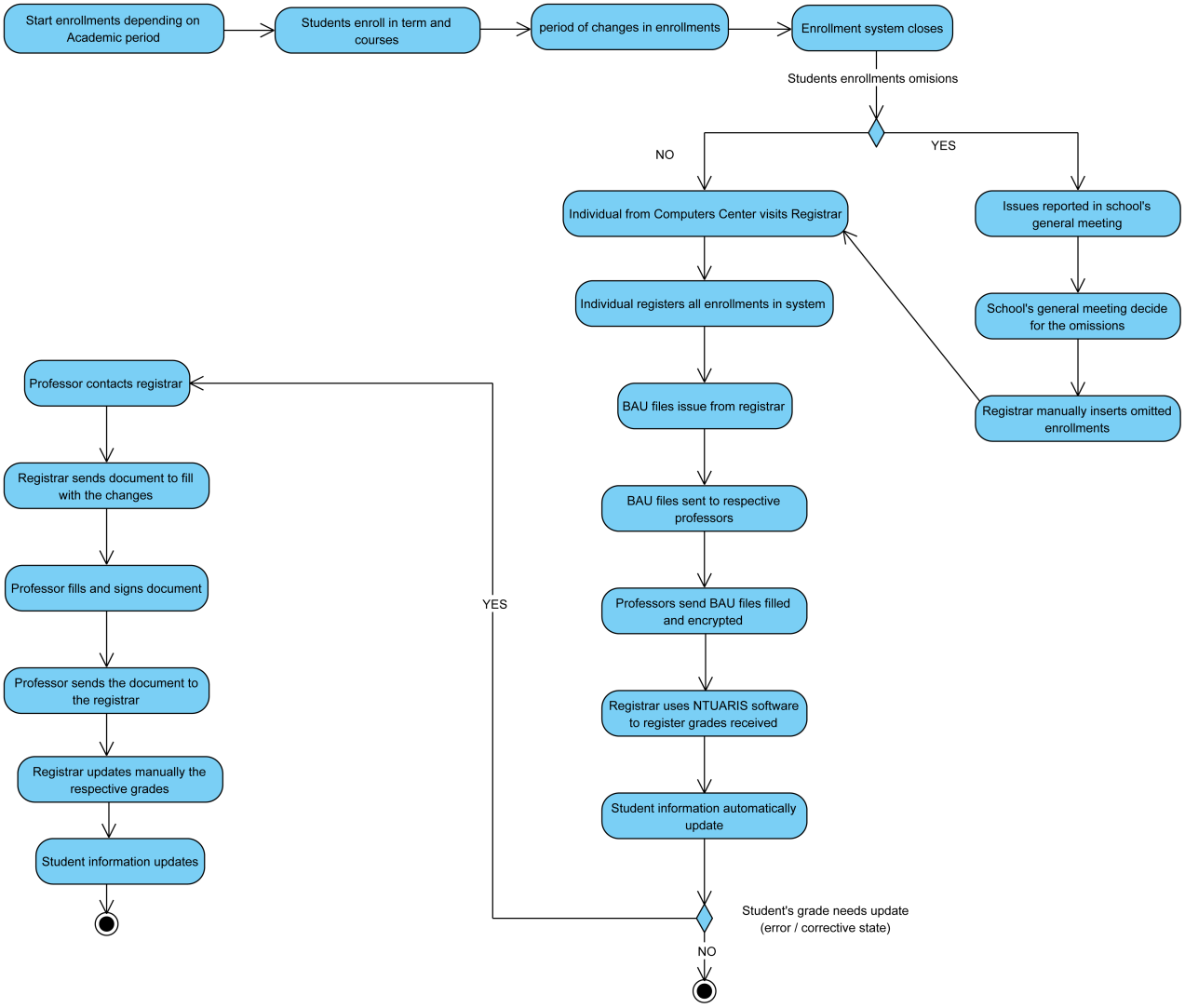


Figure 3.1. *Legacy System work flow*

The problem that this thesis addresses is the integrity and security of the grades that are eventually stored in the system. Grades are sent from the professor to the registrar inside a file and this file might be tampered with, even though is encrypted. Additionally, when a student's grade needs to be updated after the grades have been sent by the professor and inserted to the system by the registrar, no new file is sent. Instead a plain document is sent which mentions what updates must be done. Current non-blockchain-based grade management applications take several measures to ensure both integrity and confidentiality of data. These measures are usually based on trusting some third party, such as a CA, the database administrators, database cryptography, etc. Blockchain takes a different approach to this and introduces a zero-trust, fully distributed model among

peers. As mentioned previously, all nodes of the Blockchain network have an exact copy of the ledger that is distributed among the network. Thus, Blockchain technology adds one more layer of security, since the ledger exists in different computers/nodes that participate in the network and it is practically impossible to compromise every one of them. Finally, a system that works in parallel with the legacy system is proposed, to store the same data that the legacy system stores, but also to have the capability to validate if data that eventually ended up in the legacy system's database has been altered.

3.2 A Blockchain-based approach

It is important to note that this proposal does not alter the legacy system, but in fact it exists along with it.

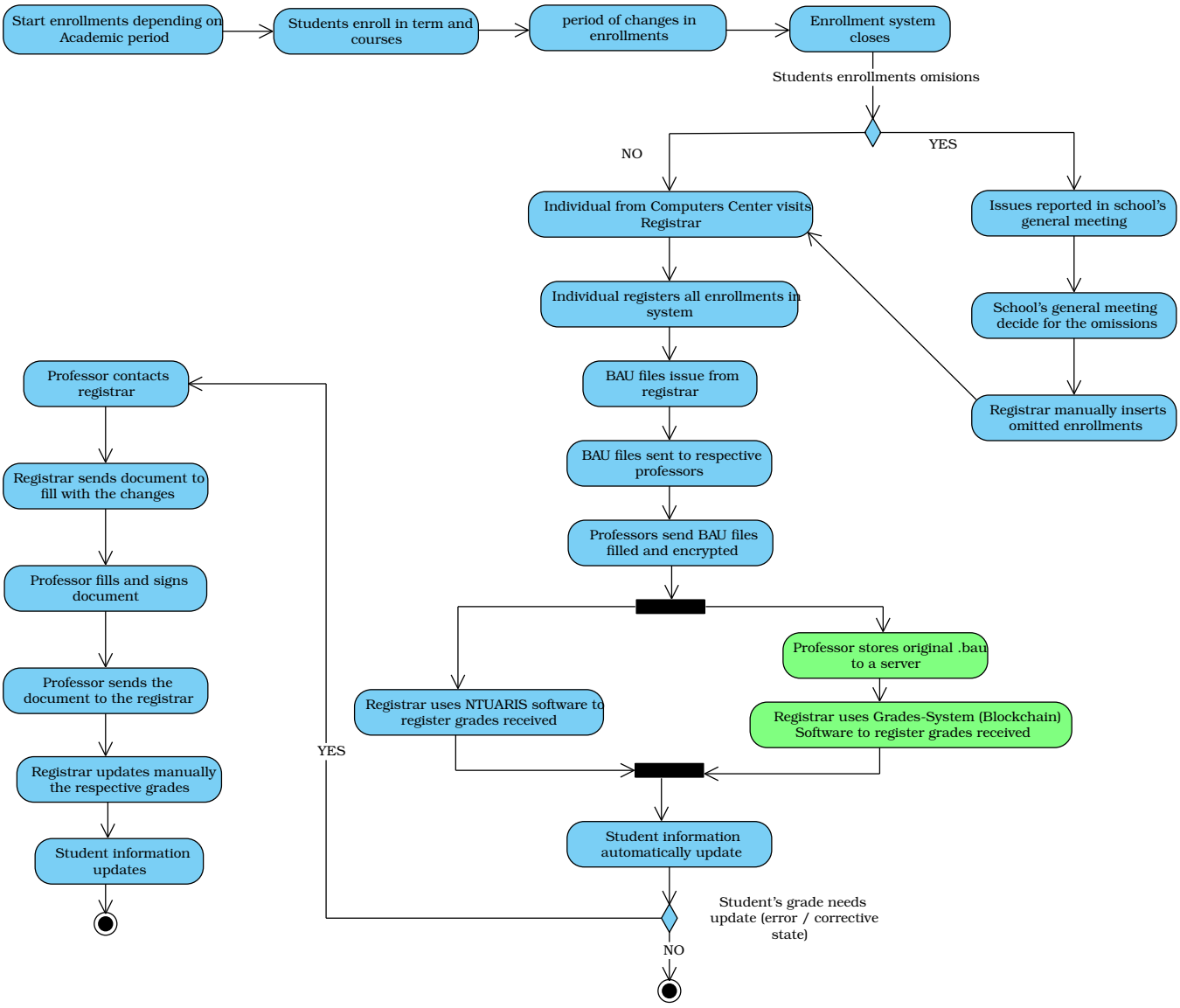


Figure 3.2. *Legacy System work flow along with the Blockchain System*

3.2.1 Privacy

Privacy is an important concept in this system as only specific users must be able to store, retrieve and complete operations with these data. The reason is that these data are personal information of students and so they must not be exposed to the public. On contrary, a certain mechanism must be implemented to keep the privacy of the data.

Thus, privacy of the network can be achieved by setting up a private Blockchain network as well as with the respective smart contracts. As smart contracts are the logic of the decentralized application, a permission system can be implemented, where existing users can vote for new users if they can use the dApp or not. In case a user is not accepted by others, will not be able to perform any actions in the dApp and eventually interact with the smart contracts. By implementing a voting mechanism, malicious users that try to interact with the smart contracts can be excluded.

3.2.2 Validation and Integrity

Since the proposed solution must solve the problem of the integrity of students' grades, there must be a corresponding mechanism that can validate if something has changed in the BAU file that has eventually ended up in the legacy system.

For this reason, these files should be stored in the blockchain and there should be a feature for validating the file sent by the professor to the registrar, at any time. As mentioned above the registrar receives and stores the file in the legacy system with the help of specific software. The content of the file exists in a database and the file is also stored on a server. Validation can be done by comparing the file on the server with the corresponding data on the blockchain and if any differences are found, a visualization of them can be generated.

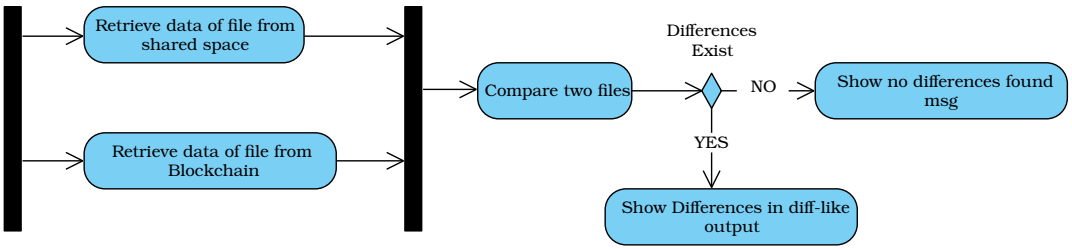


Figure 3.3. *UML Activity Diagram - Validation flow*

3.2.3 User Interface

A decentralized application (dApp) must be developed for the users of this system. Through this dApp, users can login with their wallet and perform all necessary actions in the system by interacting with the smart contracts that are deployed. The process for users should be simple without complicated steps and the data should be presented accordingly.

3.3 A private-permissioned Blockchain approach, based on Ethereum

The described solution is crucial to be implemented with a private or permissioned blockchain network as the only actors that must interact with the system are the entities of a university involved in the workflow of grade management, such as registrars, students and professors.

To implement a permissioned network different alternatives exist, some of which are briefly discussed in the sequel [14].

1. R3 Corda [15] is an innovative blockchain platform for businesses, which aims to reduce the cost of business transactions and increase their speed. This project was originally designed for the financial sector, however, it can be applied to other uses, such as healthcare, supply chain, government and public services, and trade finance.

2. Quorum platform was created by the company JP Morgan and is the corporate version of the Ethereum blockchain [16]. By modifying the Ethereum kernel, the Quorum platform can quickly integrate Ethereum updates. It is an open source blockchain platform that uses vote based algorithms to execute hundreds of trades per second. As it is a private blockchain platform, it only allows authorized participants to take part in transactions.

3. Ripple platform aims to connect financial operators, trading companies, banks and payment service providers [17]. Ripple allows international payments through a digital asset called Ripple or XRP. Using a probabilistic voting method, the Ripple platform achieves consensus between nodes on the network. Several large companies such as American Express, SBI Holdings and Deloitte are experimenting with Ripple's blockchain capabilities to transform payment processes.

4. Hyperledger [18] Foundation offers an open source blockchain system portfolio which consists of distributed ledger technologies, libraries, domain specific and tools.

   The DLTs that Hyperledger Foundation offers are Fabric, Sawtooth, Iroha, Besu and Indy. Hyperledger Fabric is designed for permissioned networks and allows only entities with known identities to participate in the system. Only authorized participants can participate in the transactions made on the Hyperledger Fabric platform. Thus, Hyperledger Fabric [19] was considered as an option for the implementation of the blockchain network.

Despite the different options that exist to implement the private blockchain network, Ethereum was chosen for several reasons. First and foremost is that most of the mentioned options were too specific for this particular project and the only alternative could be the Hyperledger Fabric. The second reason is that Ethereum private network was fairly straightforward to set up compared to the Hyperledger Fabric, which can be backed

up from Chapter 7.1 of [14]. Finally, Ethereum has rich documentation and a strong community that is a great resource for any problems that may occur.

Below, a simple block diagram is shown with the nodes of the permissioned blockchain and how users can attach to a node and perform operations on the blockchain through the smart contracts.

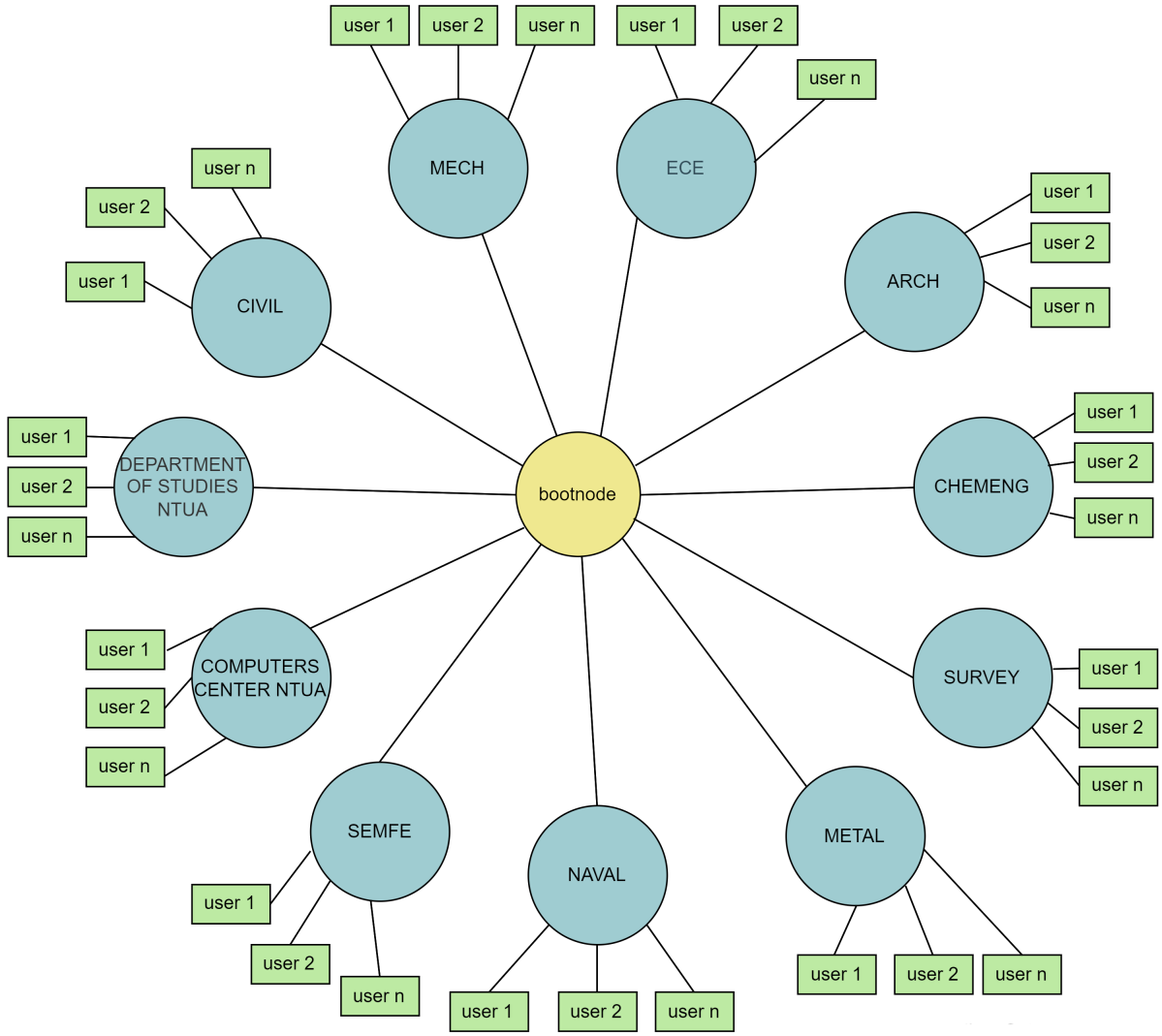


Figure 3.4. *Block Diagram*

3.4 Use Case Model

The requirements explained previously, are presented below in a UML use case diagram. The actors involved in these use cases are the master user(s) and the registrar users.

3.4.1 Registrar Users

Registrar users are able to do the following operations in the system:

1. submit a new grades file in the blockchain
2. see the respective courses of the school they belong
3. vote for a new user, if a new user is to be accepted as a new user in the system

3.4.2 Master Users

Master users can perform every operation registrar users can as well as:

- start a new voting process for a new user
- retrieve all participants of the system

Figure 3.5. *UML Use Case Diagram*

3.5 UML High-level work flow Activity Diagram

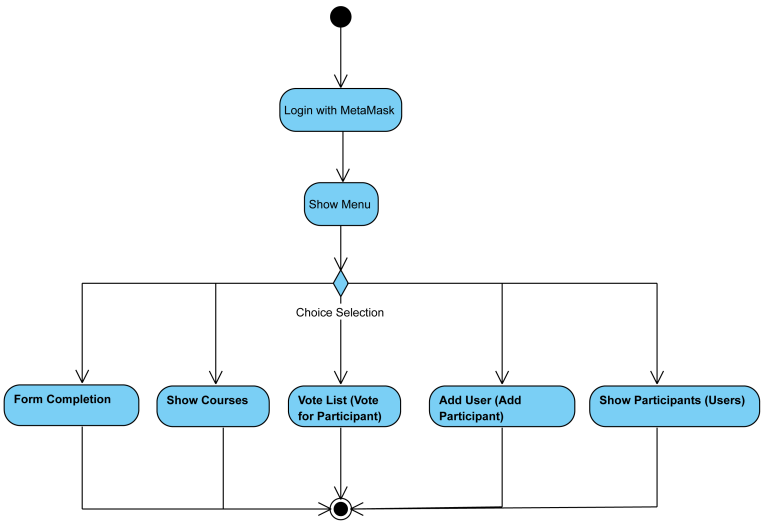


Figure 3.6. *UML High-level work flow Activity Diagram*

The activity diagram above highlights the high-level work flow of the decentralized application. The first step is that the user must log in to the dApp using MetaMask. When the log in is completed, user's information are retrieved from the Blockchain. Using these information a permissions check takes place. In case the user has permissions, the main menu of the dApp is shown an can perform any of the following actions.

- Form Completion: Both master users and registrar users can perform this action. By completing this form and adding all the necessary information, new course grades are registered to the ledger.
- Show Courses: This action can be performed by both users, with a major difference. Master users can retrieve all courses and eventually see the ledger stored in the Blockchain, while registrar users can only retrieve course information regarding the school they are assigned to. In addition, another functionality of this action is that users can validate retrieved course information and in case of any difference found in the stored data and the original data, a visual representation of it will appear.
- Vote List (Vote for Participant): Both users can use this functionality. When a new user wants to have access to the shared ledger, a voting process starts from a master user and every participant with permissions in the system can vote for or against this new user. For the new user to be added, there must be unanimity.
- Add User: This action can only be performed by a master user, where a form is completed with the necessary information of the applicant.
- Show Participants: Every user that has permissions to the system can be listed by performing this action. Only a master user can retrieve all participants.

A more detailed description of the above together with the corresponding figures is available below.

3.6 Wireflow

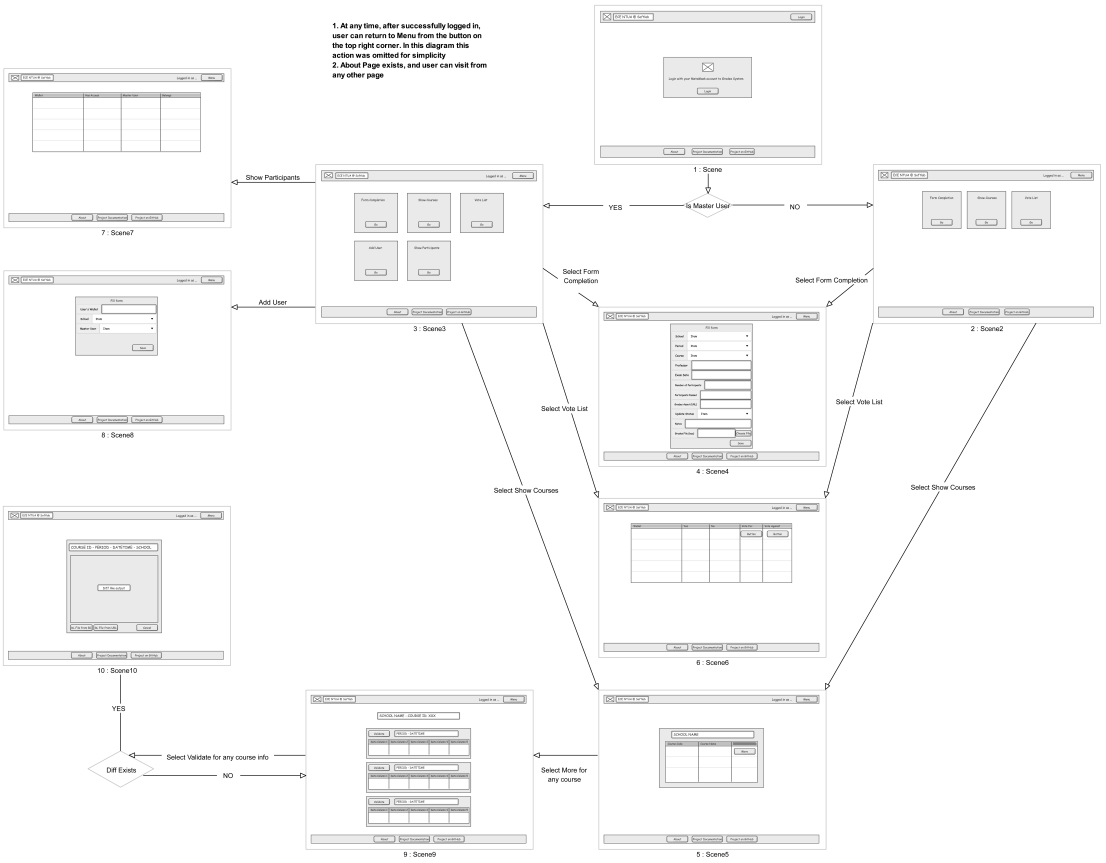


Figure 3.7. *Wireflow diagram*

3.7 Login and Different Users

3.7.1 Description

As mentioned above two different kinds of users exist in the proposed system with different permissions. Therefore their user interface should differ. Below the Activity Diagram of the login procedure is shown. In addition, login page, and the menu page for the respective users are presented in Lo-Fi wireframes.

3.7.2 What is MetaMask

MetaMask is a service offered as a browser extension and a mobile application which was created by ConsenSys and allows users to store and manage account keys, broadcast transactions, send and receive Ethereum-based cryptocurrencies and tokens, and securely connect to decentralized applications through a compatible web browser or the mobile app's built-in browser. Developers achieve a connection between MetaMask and their decentralized applications by using a JavaScript plugin such as Web3js or Ethers to define interactions between MetaMask and Smart Contracts [20].

So MetaMask will enable the users to interact with the dApp in an easy convenient and transparent way.

3.7.3 UML Activity Diagram

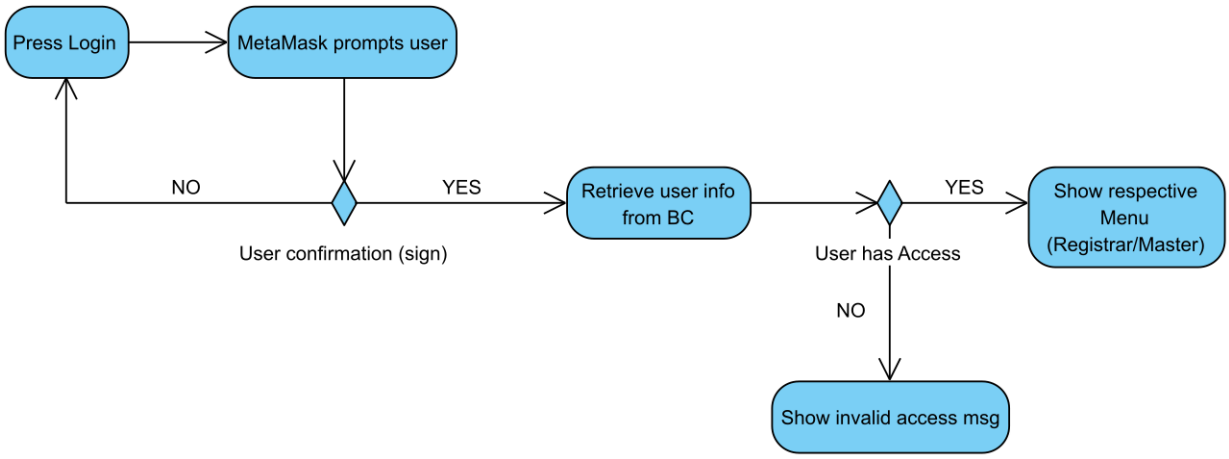


Figure 3.8. *UML Activity Diagram: Login Procedure*

3.7.4 Login - Metamask Lo-Fi Wireframe

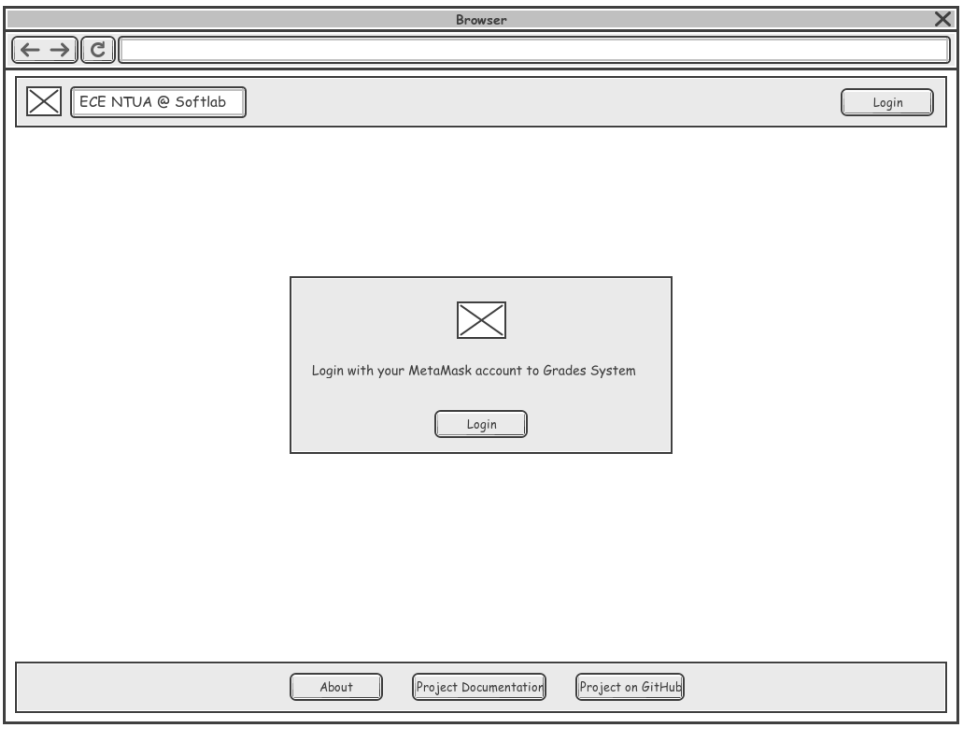


Figure 3.9. *Login Wireframe*

3.7.5 Menu - Master Lo-Fi Wireframe

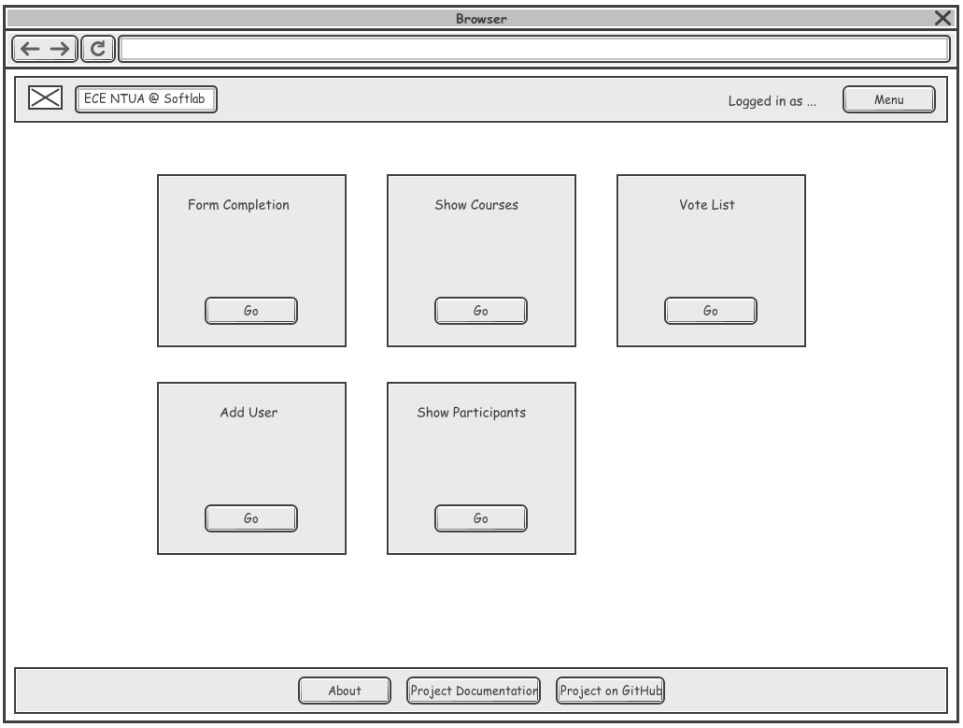


Figure 3.10. *Menu - Master Wireframe*

3.7.6 Menu - User Lo-Fi Wireframe

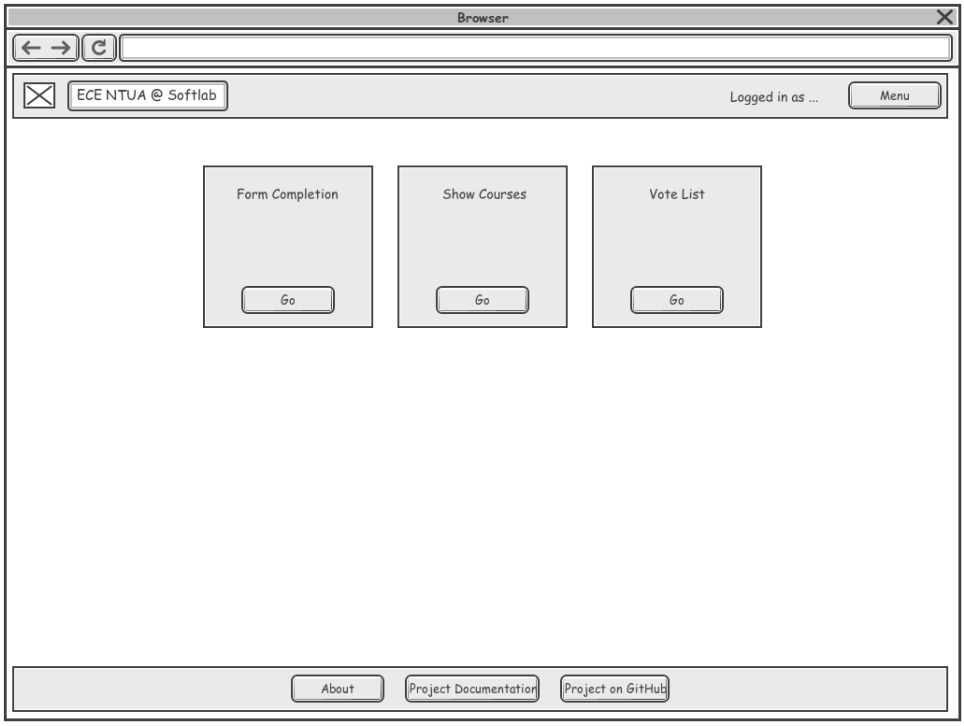


Figure 3.11. *Menu - User Wireframe*

3.8 Use Case 1: Add Grades

3.8.1 Description

This feature can be used both from master users and registrar users. Users must complete a form with all the information and when save is pressed the form is validated. If any validation errors are found, the action is cancelled.

Master users are able to add grades for every single school in contrast with the registrar user who can add grades only for the school that belongs.

3.8.2 UML Activity Diagram - Professor generate and distribute file

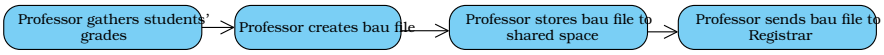


Figure 3.12. *UML Activity Diagram - Procedure of Professor creating and distributing file*

3.8.3 UML Activity Diagram - Add Grades

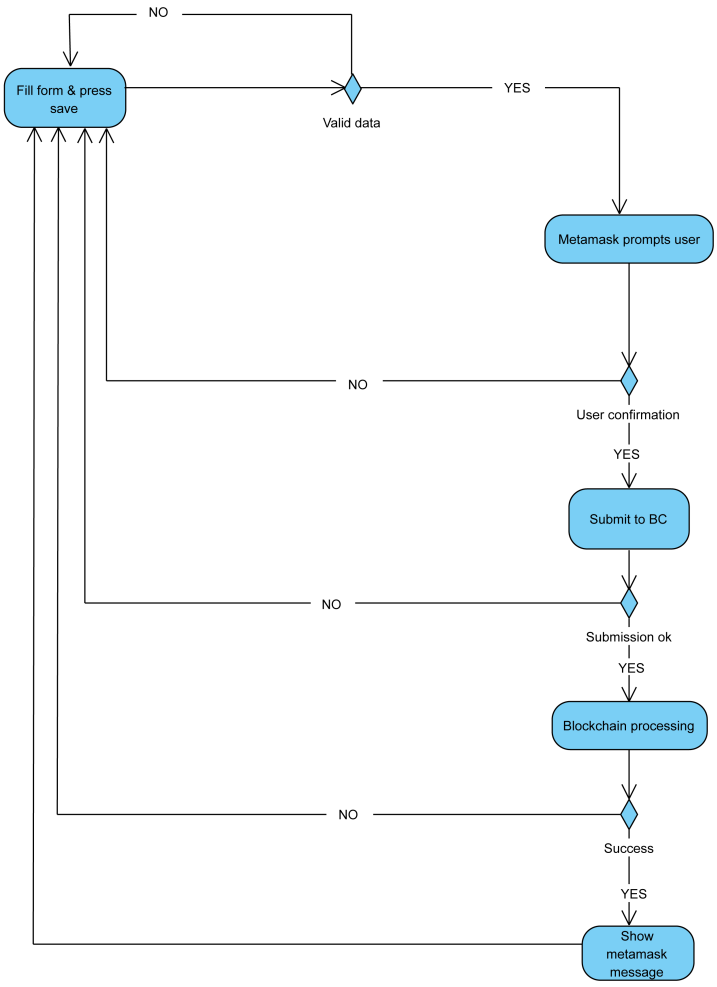


Figure 3.13. *UML Activity Diagram: Add Grades*

3.8.4 UML Sequence Diagram

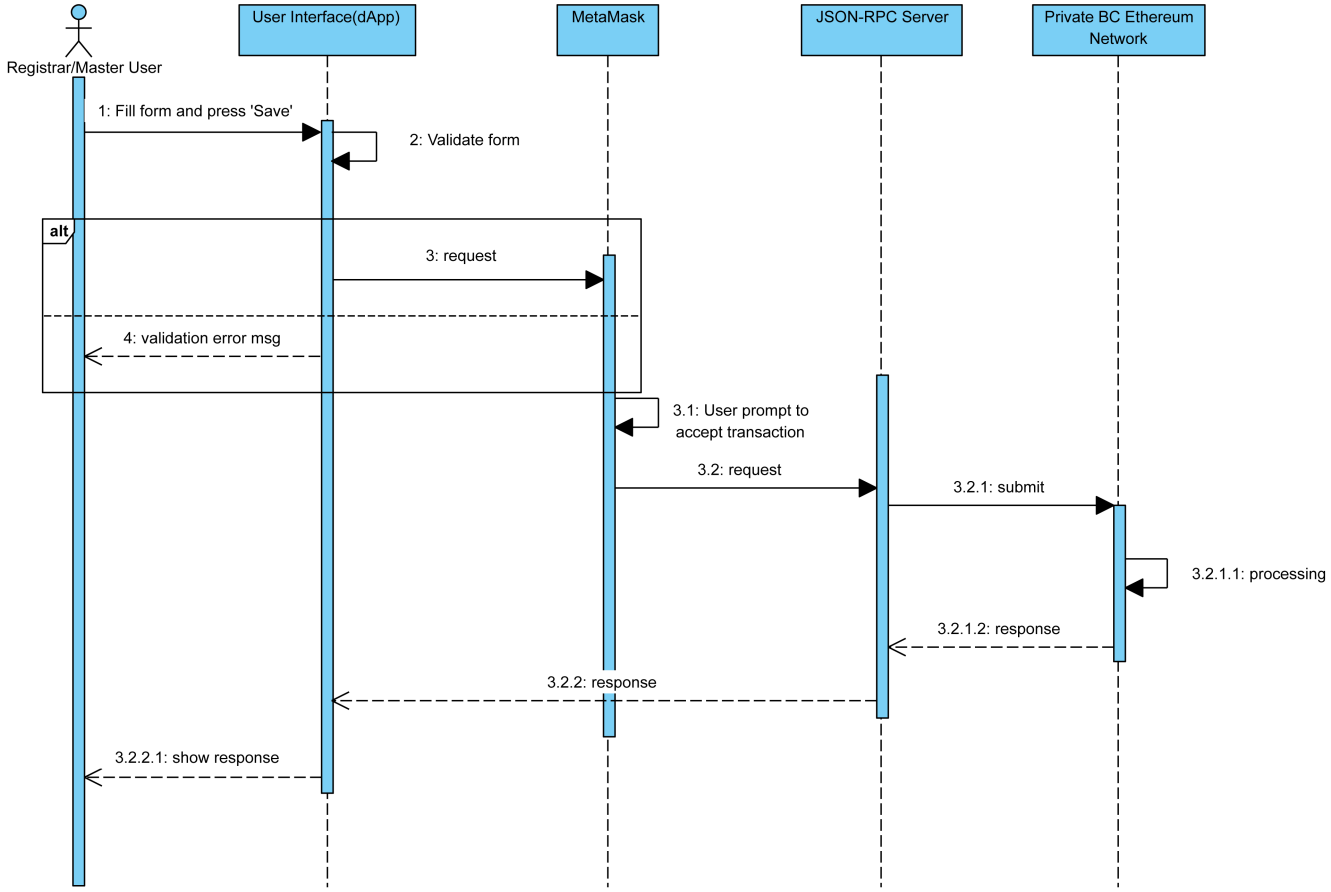


Figure 3.14. *UML Sequence Diagram: Add Grades*

3.8.5 UML Auxiliary Sequence Diagram

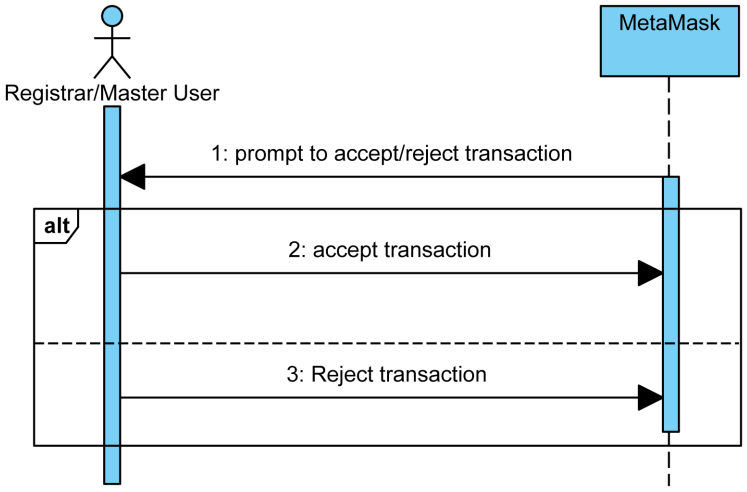


Figure 3.15. *UML Sequence Diagram: MetaMask prompt - User accept/reject transaction*

3.8.6 Lo-Fi Wireframe

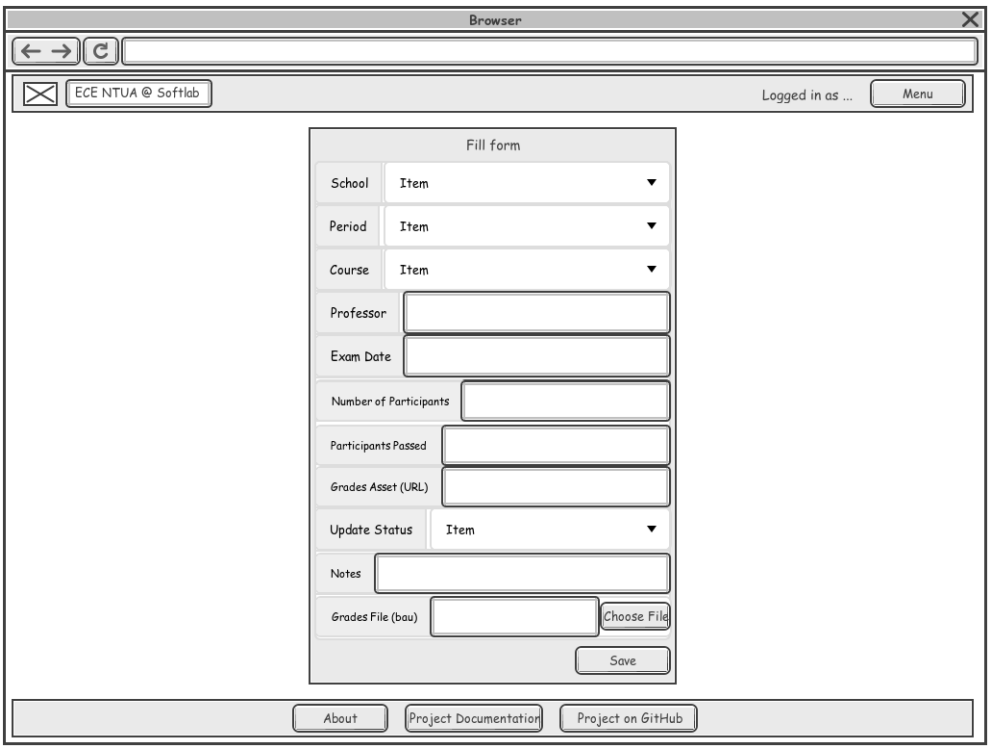


Figure 3.16. *Add Grades Wireframe*

3.9 Use Case 2: Show Course Information and Validate

3.9.1 Description

Both user categories can use this functionality, which retrieves information of a specific course. A master user is able to retrieve every course's information of any school, while a registrar user can only retrieve course's information for the school that belongs.

Users can also validate the course's information by pressing the 'Validate' button. If any differences are found between the data retrieved from the blockchain and the data of the file that was uploaded during the insertion, then a modal with the differences is loaded, and users can download both files.

3.9.2 UML Activity Diagram

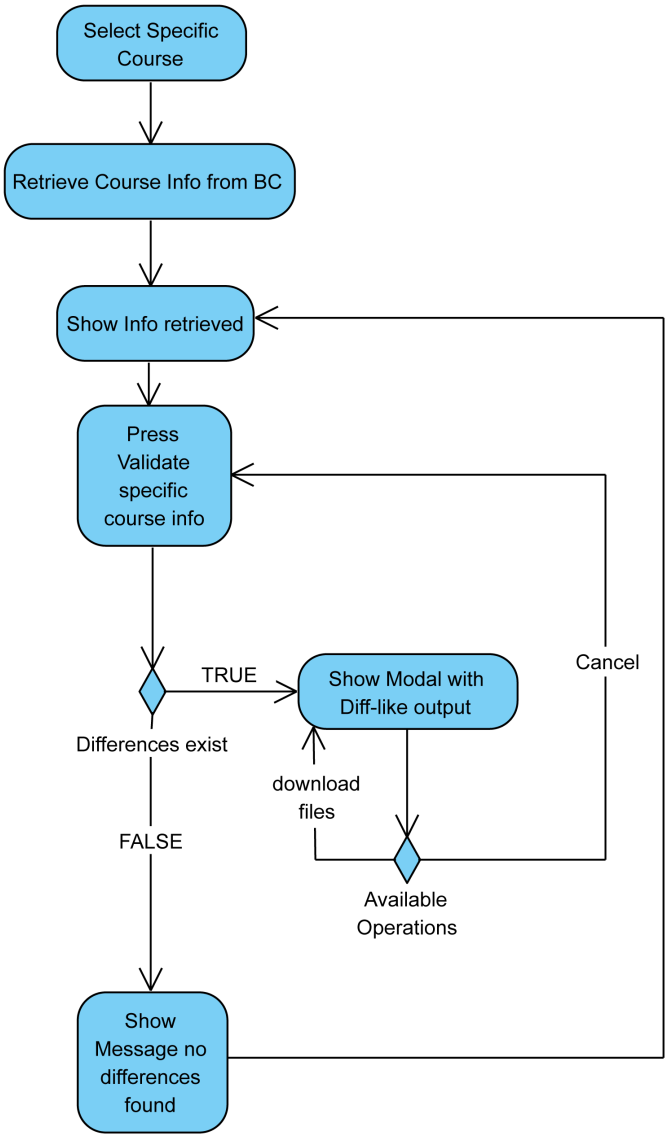


Figure 3.17. *UML Activity Diagram: Show Course Information and Validate*

3.9.3 UML Sequence Diagram

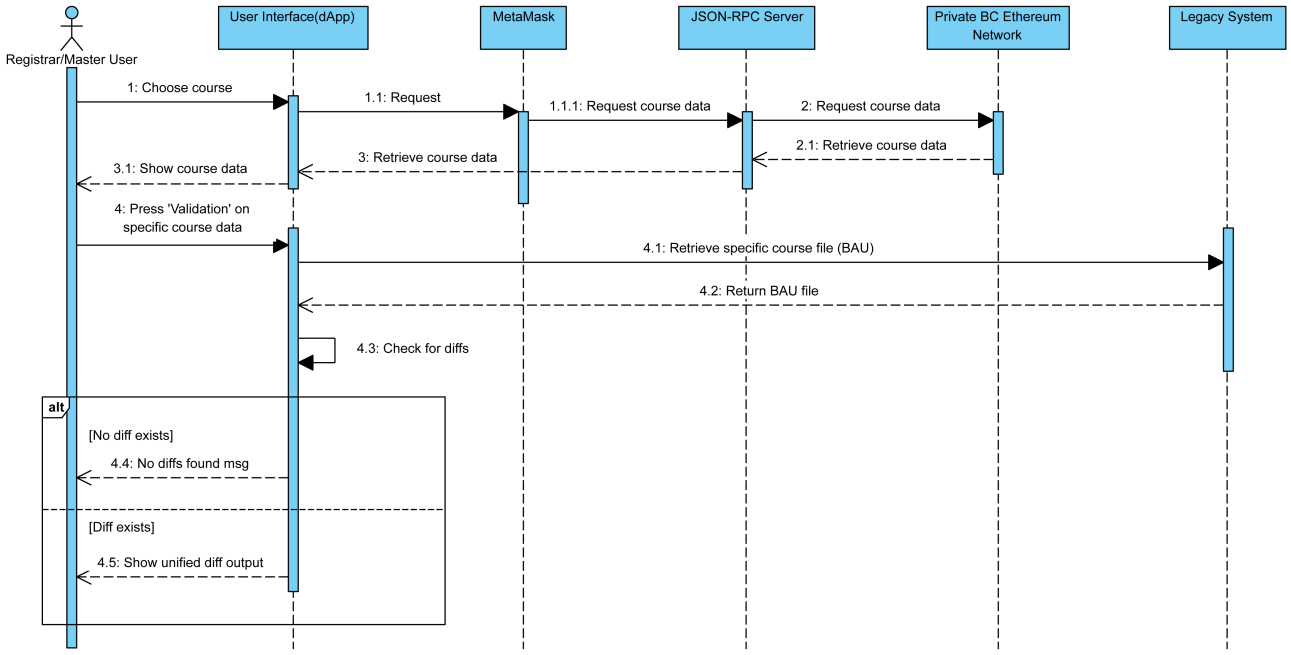


Figure 3.18. *UML Sequence Diagram: Show Course Information and Validate*

3.9.4 Show Courses LoFi Wireframe

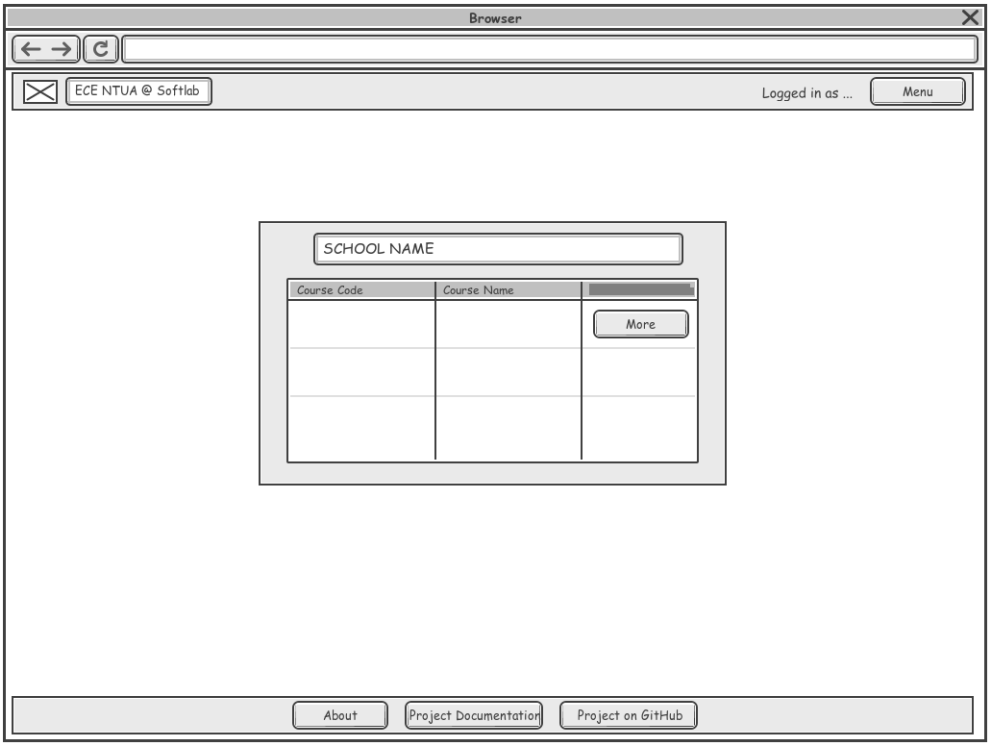


Figure 3.19. *Show Courses Wireframe*

3.9.5 Show Course Information LoFi Wireframe

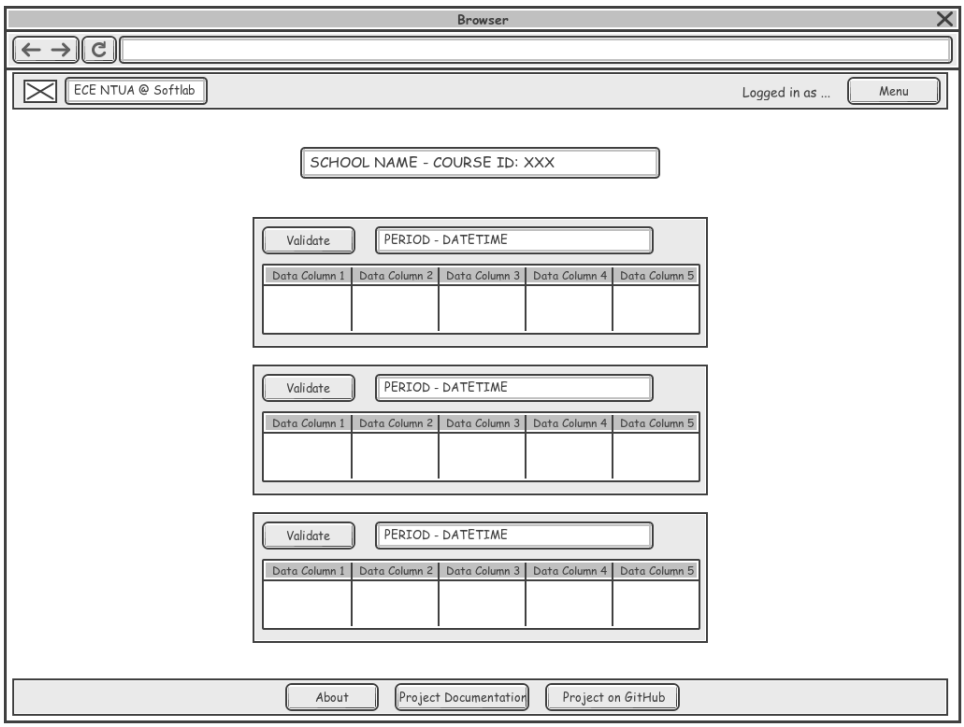


Figure 3.20. *Show Course Information Wireframe*

3.9.6 Show Diff Output LoFi Wireframe

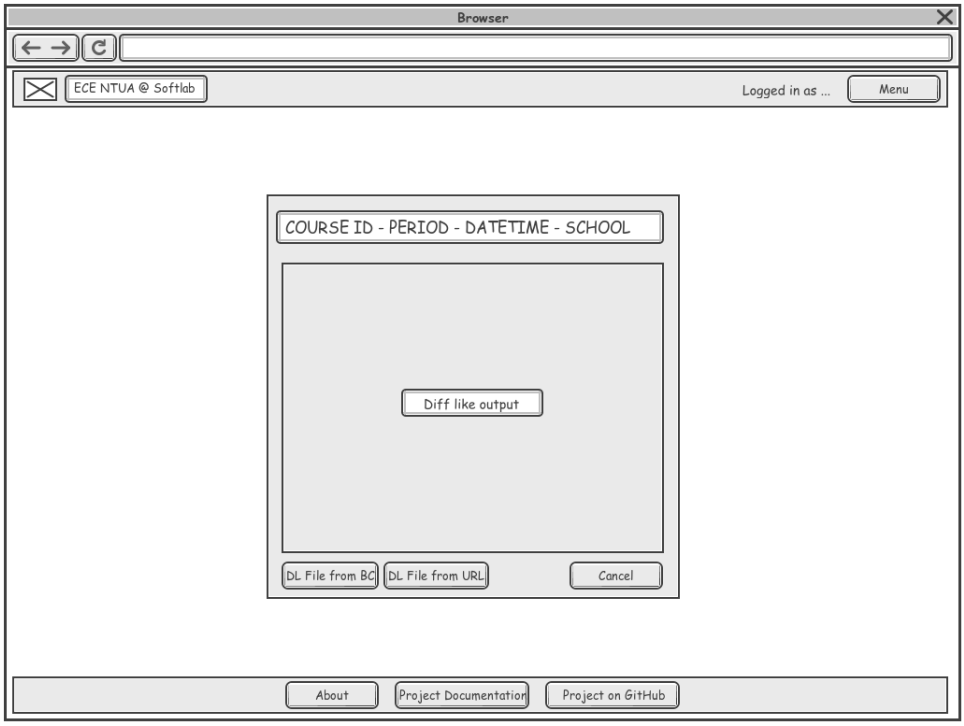


Figure 3.21. *Show Diff Output Wireframe*

3.10 Use Case 3: Vote for new Users

3.10.1 Description

All users that have access to use this dApp, are eligible to vote for or against any other user that tries to get access to use the dApp. They can only vote once for every ongoing vote(there can be many ongoing votes) and the applicant user only gains access in case of unanimity.

3.10.2 UML Activity Diagram

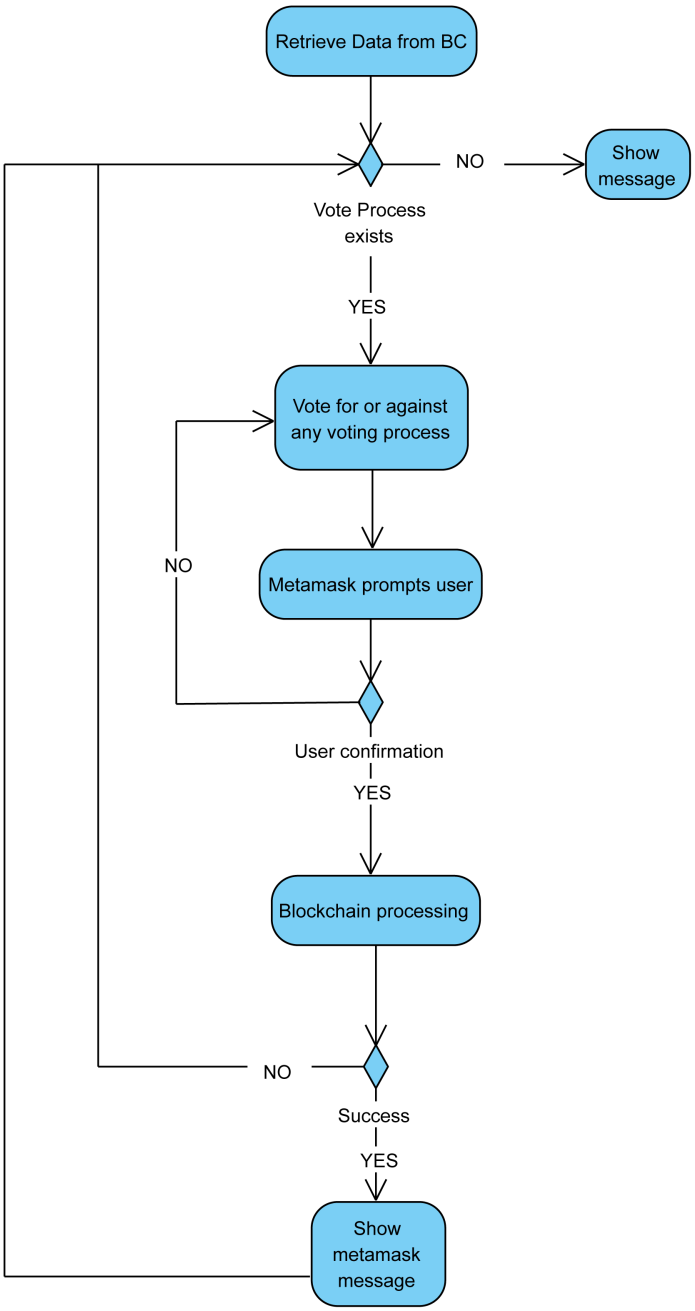


Figure 3.22. *UML Activity Diagram: Vote for new Users*

3.10.3 UML Sequence Diagram

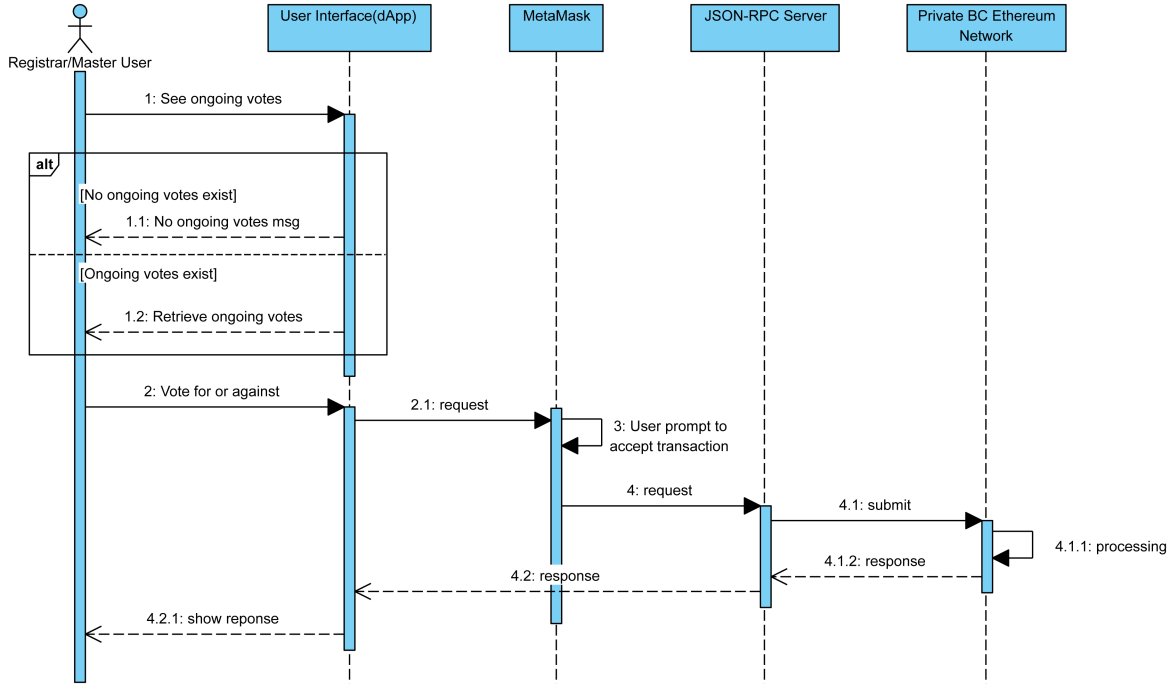


Figure 3.23. *UML Sequence Diagram: Vote for new Users*

3.10.4 Lo-Fi Wireframe

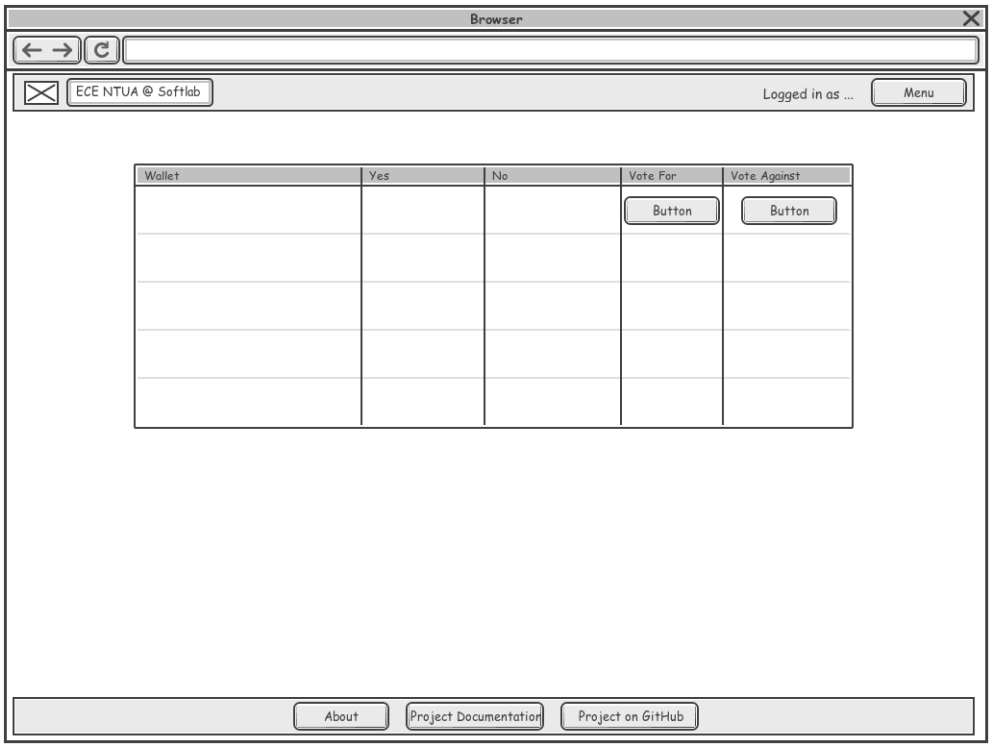


Figure 3.24. *Vote for or Against Wireframe*

3.11 Use Case 4: Start a new Vote

3.11.1 Description

Only master users can start a new vote for a new applicant. This means that the applicants do not directly apply, but a master user is the intermediate. To start a new vote, the master node must fill the form and press save. In case of any validation error, the user is prompted with the error and have to try again.

3.11.2 UML Activity Diagram

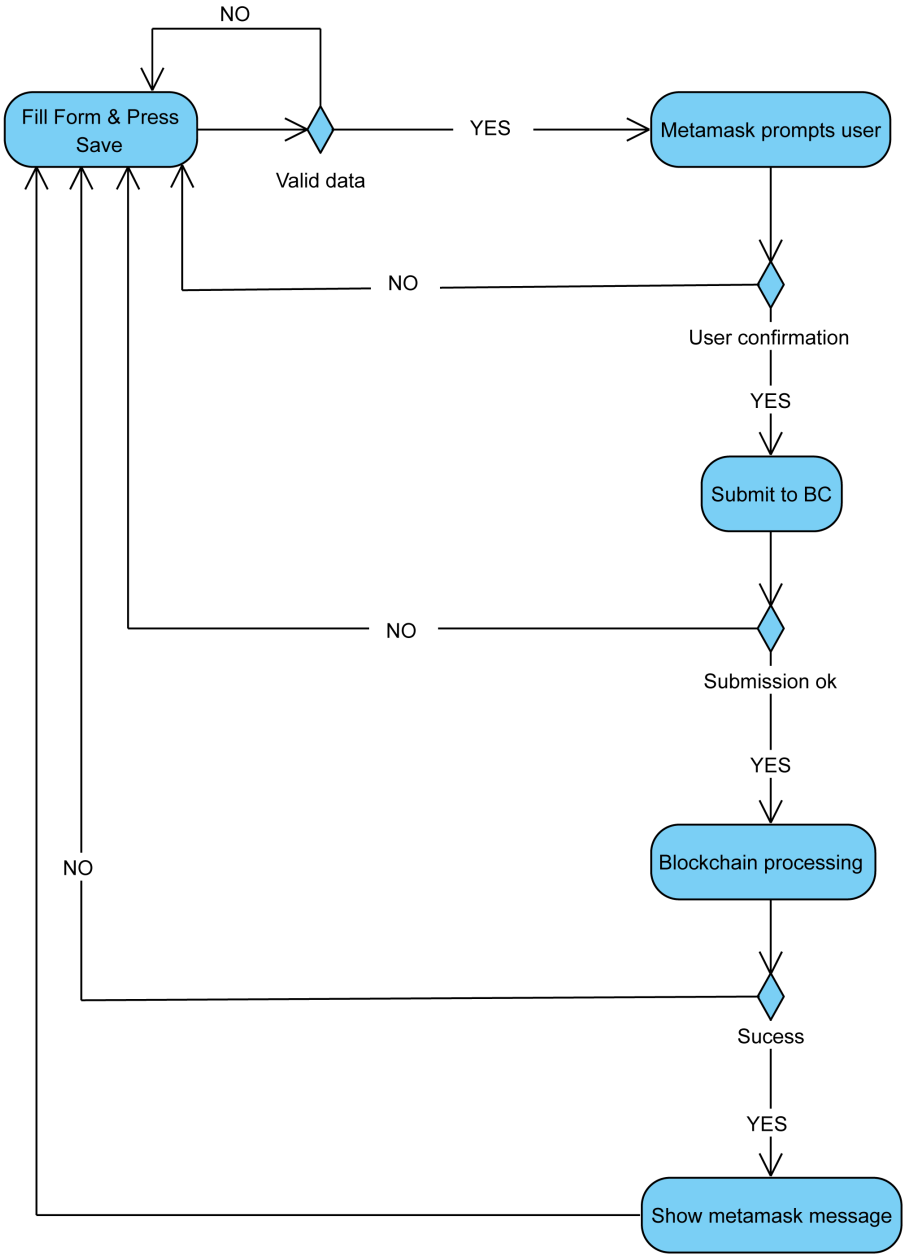


Figure 3.25. *UML Activity Diagram: Start a new Vote*

3.11.3 UML Sequence Diagram

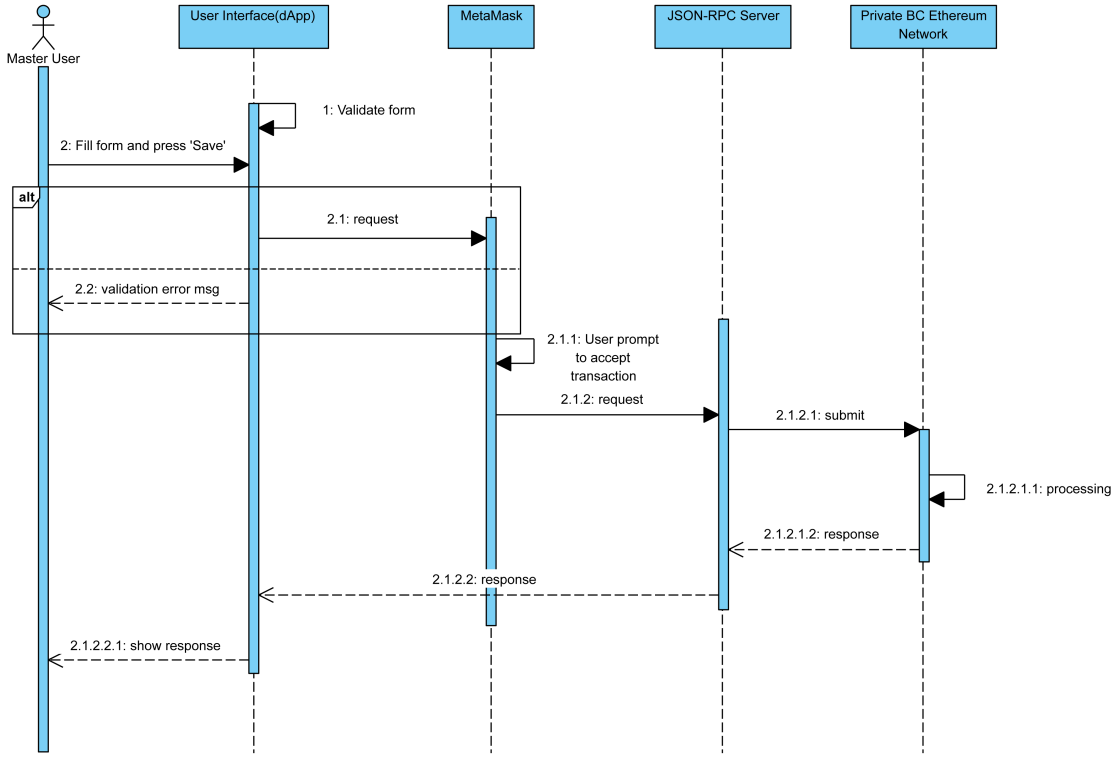


Figure 3.26. *UML Sequence Diagram: Start a new Vote*

3.11.4 Lo-Fi Wireframe

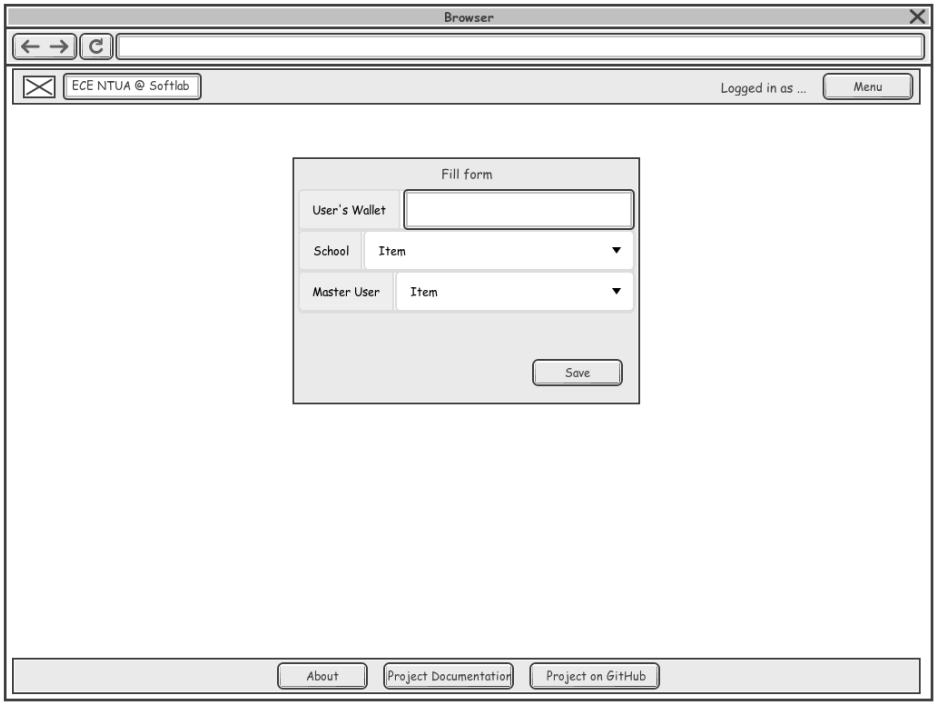


Figure 3.27. *Start Vote Wireframe*

3.12 Use Case 5: Show Users

3.12.1 Description

A master user is able to retrieve a list with all the participants that have access.

3.12.2 UML Activity Diagram

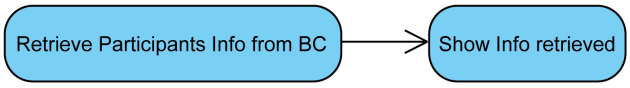


Figure 3.28. *UML Activity Diagram: Show Users*

3.12.3 UML Sequence Diagram

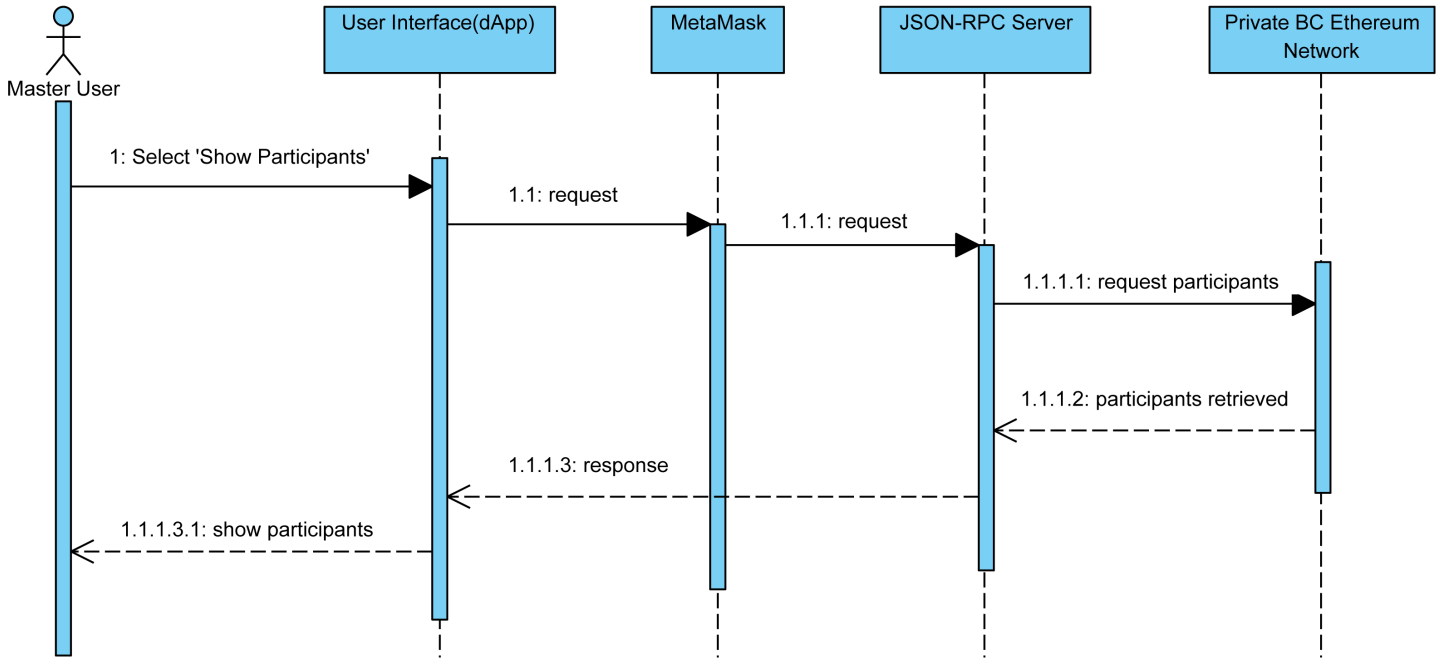


Figure 3.29. *UML Sequence Diagram: Show Users*

3.12.4 Lo-Fi Wireframe

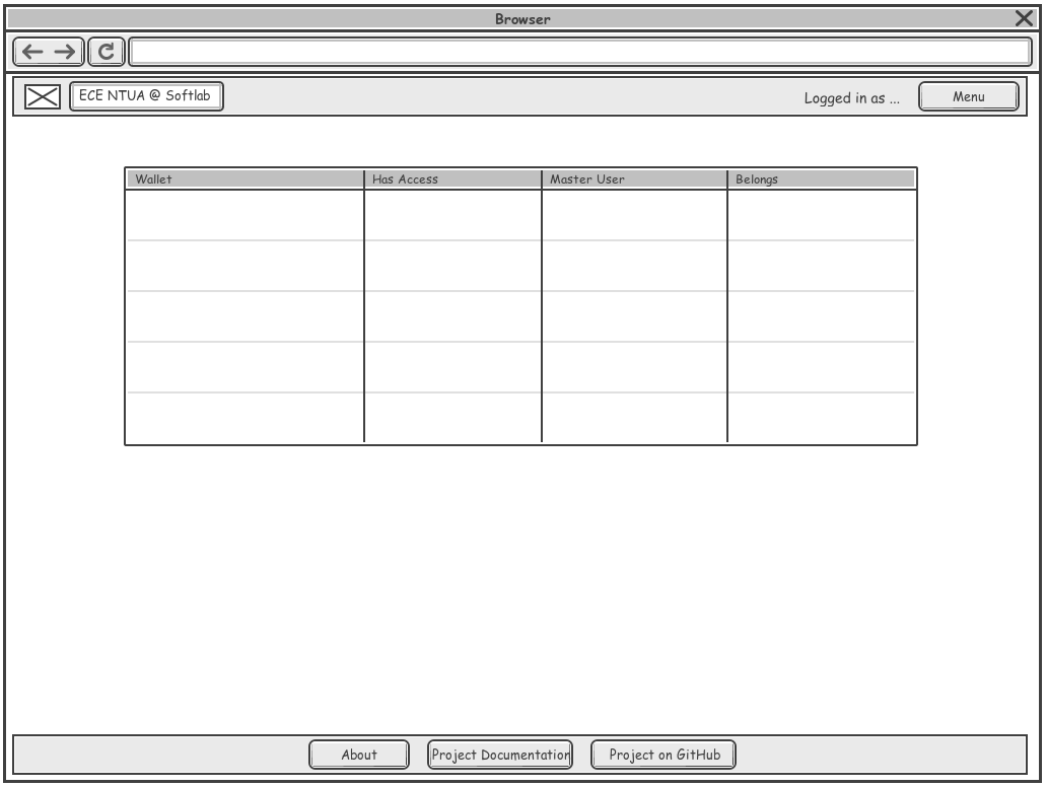


Figure 3.30. *Show Participants/Users Wireframe*

Chapter 4

Implementation: Difficulties encountered and different approaches

In this chapter, all the challenges met during the implementation of the dApp will be discussed without getting into the actual implementation that will be presented in Chapter 5 in detail.

4.1 Setting up the private Ethereum network

The set-up of the private Ethereum network was not so challenging as the documentation provided by Ethereum is detailed. The only issue that was faced in this step was the generation of the genesis block. The reason this was overwhelming was the parameterization of the genesis block. Many options had to be researched on what they offer and what limitations they have. An example is that when the private Ethereum network was set up, a specific parameter caused the Blockchain to add empty blocks every few seconds, which wasn't the expected outcome.

4.2 First Implementation: A Hello World dApp

The first implementation [21] to get hands-on experience in creating a decentralized application was a simple text field where a user could add a simple sentence (string) in the Blockchain. The difficulty of the first implementation was the lack of experience in developing such applications.

More specifically, because the Ethereum platform was chosen, the first task that had to be done was to learn Solidity, which is the language for writing smart contracts and eventually building a dapp, as smart contracts are the logic of such applications. After the first task was completed, the next step was to develop the front-end that enables users to interact with the smart contracts. To keep things simple, NodeJS, HTML and CSS were used to create the front-end. The struggle here was that the libraries that were needed for the front-end to invoke the smart contracts and eventually allow the user to interact with them were outdated and hard to find the right versions that everything was working properly.

When everything was functioning accordingly, this particular demo was enhanced

with a more complex smart contract that was able to keep each user their sentences and eventually retrieve them and present them.

Finally, because in the Grades System dApp different kinds of permissions were necessary, this demo was modified accordingly to gain an understanding of how this could be developed. There were created two different kinds of users in this demo, a simple user who could only retrieve what he entered as sentences and the master user who could fetch all data that were added.

4.2.1 UML Component Diagram

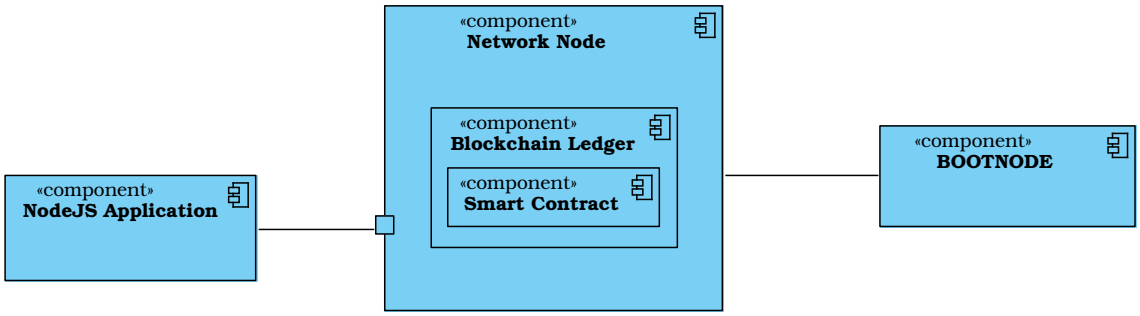


Figure 4.1. *Hello World dApp UML Component Diagram*

4.3 Second Implementation: Grades System dApp

When the necessary knowledge was acquired from the demo dApp, the development of the Grades System dApp [22] could start.

4.3.1 Technologies Used

The project was based on JavaScript and NodeJS was used as a runtime environment, due to familiarity from other projects. The main packages that used from npm were the following:

1. EJS, as the templating engine to generate HTML using javascript (user interface)
2. ExpressJS which is a minimal framework that provides a robust set of features to develop web and mobile applications
3. Web3 which enables the interaction with a (specified) blockchain network and its smart contracts
4. diff and diff2html to create a visualization changes between data stored in the blockchain and a text file

4.3.2 Issues

This implementation had the same issues as the first one because it still had to rely on the Web3 package which, according to the node package manager (npm) has several

security vulnerabilities. These vulnerabilities do not concern Blockchain but instead the client that supports the connection and the operations to that network.

The second problem which led to the third implementation [23] was the fact that this particular project had to rely on environment variables from a .env file. Such variables are the RPC Server address, the public address of the wallet of the user who would interact with the blockchain through the dApp et cetera. It is important to note that this particular implementation invoked all smart contract functions from back-end , therefore MetaMask could not be supported without breaking changes. The reason is that MetaMask injects a global API into websites visited by its users at windowėthereum and this injected API only exists in the browser environment. Thus, this object could not be sent to the controllers of the back-end. Hence, for a more user-friendly and product-like implementation, NextJS was considered.

4.3.3 UML Component Diagram

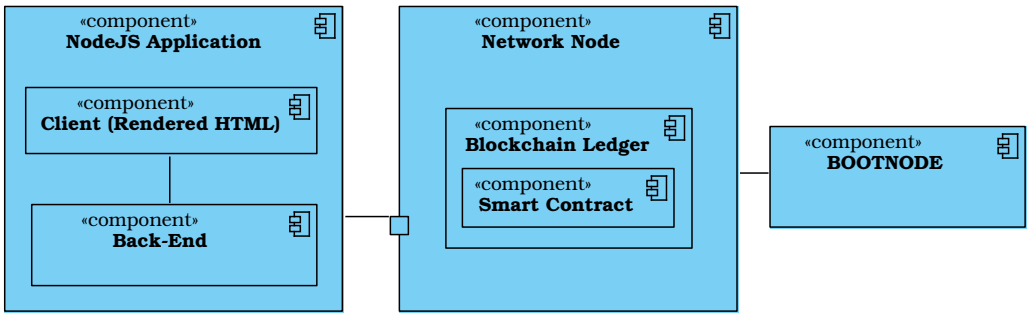


Figure 4.2. *Grades System dApp (NodeJS) UML Component Diagram*

4.4 Third Implementation: Grades System dApp - NextJS

As mentioned above, the primary reason for rewriting the complete project using NextJS was for a more product-like implementation, which the user would find more convenient. To achieve this and stop using environment variables MetaMask came into play.

The secondary reason which led to this decision was that NextJS and ReactJS (NextJS uses ReactJS), have a strong community on decentralized applications development. This results in packages that are constantly getting updates, fixes and of course much more material on the specified subject.

4.4.1 Issues

Through the implementation with the use of NextJS no major issues were faced. Every functionality that was developed in the original implementation was successfully written with NextJS. All packages that were needed during the development were up to date and had no issues. The only difficulty of NextJS was learning this React framework to be able to migrate from NodeJS.

4.4.2 UML Component Diagram

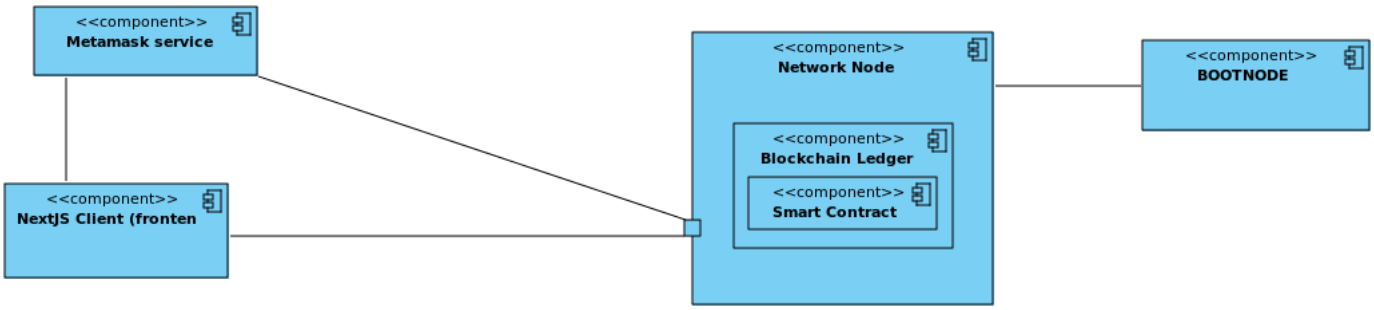


Figure 4.3. *UML Component Diagram*

4.4.3 UML Deployment Diagram

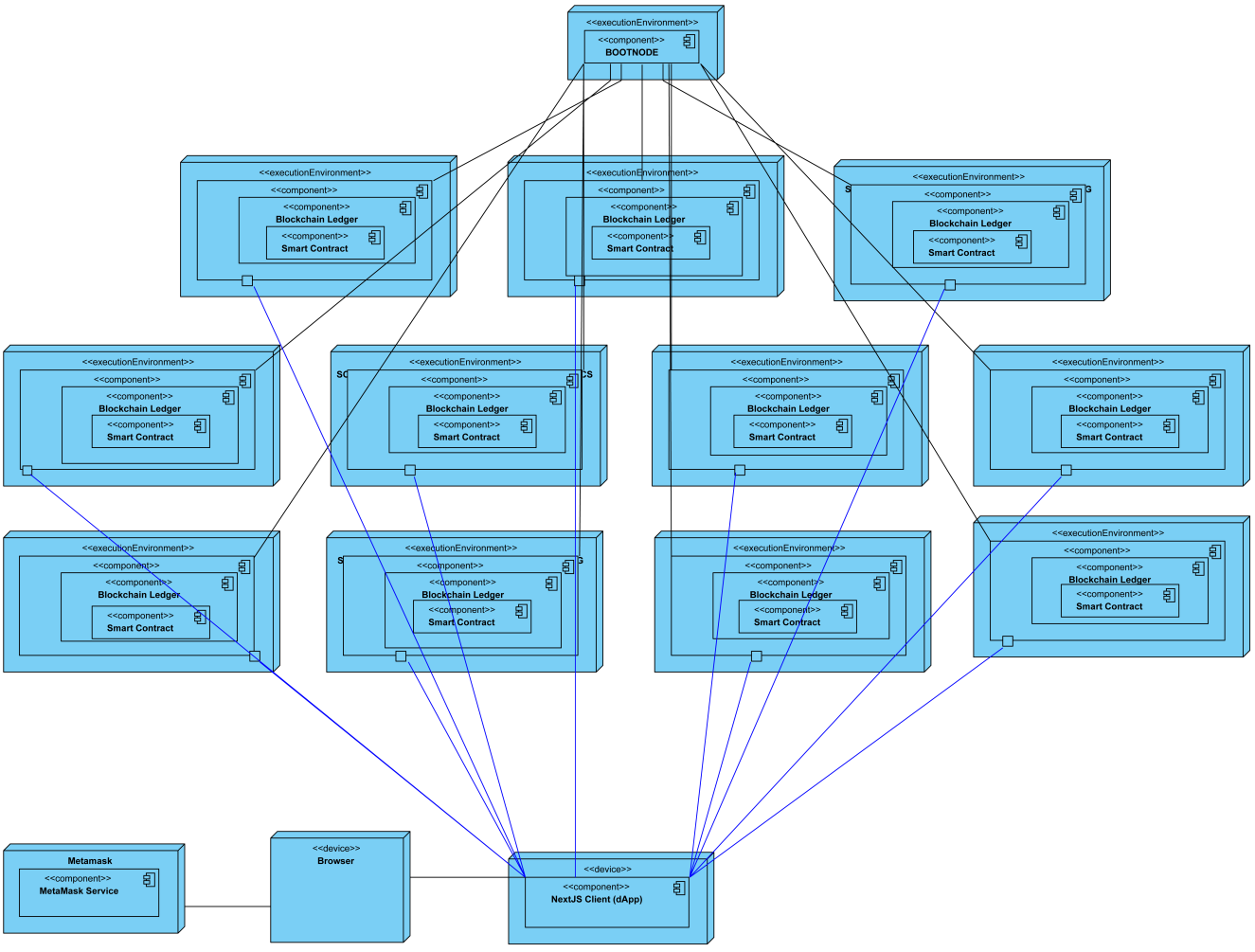


Figure 4.4. *UML Deployment Diagram*

Chapter 5

Implementation: Complete guide of the decentralized application

This chapter will focus on the implementation of the NextJS dApp as well as the creation of the private Ethereum Blockchain. It will be divided into the steps that were followed. The implementation is done in Linux operating system and the guide is strictly about it, however with the appropriate changes the development can be done in any OS. An activity UML diagram follows with the steps that were followed.

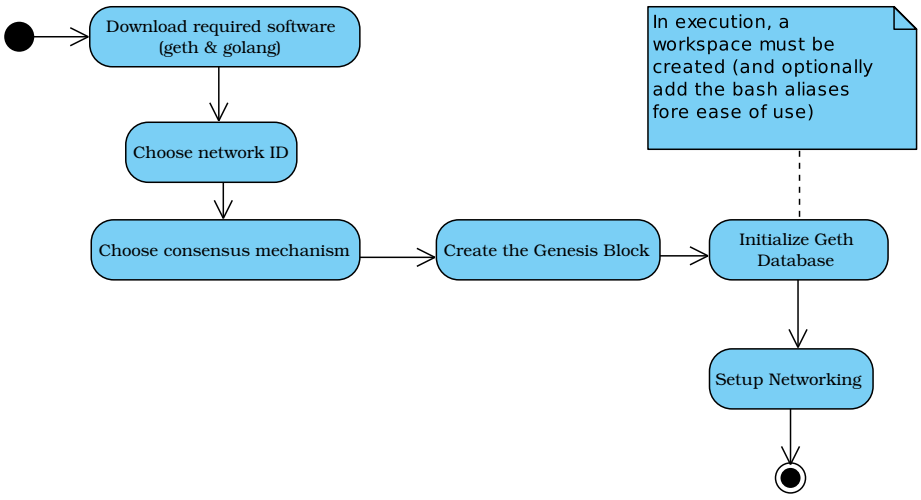


Figure 5.1. *Steps followed to develop a private Blockchain Network*

5.1 Create the Ethereum private Blockchain network

Geth will be used for the creation of the private network, which is a command-line interface (CLI) for running Ethereum nodes implemented with Go Language. Thus, geth and go language must be installed.

5.1.1 Install required software

1. Geth [24]
2. Go Language Download [25]

5.1.2 Steps - Explanation

An important note before describing the steps is that connections between blockchain nodes will occur only if both peers use the same genesis block and network ID. Both network ID and genesis block will be discussed below as cited in [26].

To create a private Ethereum BC network the procedure is fairly simple and it can be divided into the following tasks shown in the upcoming subsections.

Choose a network ID

Choose a network ID - which is an integer number which isolates Ethereum peer-to-peer networks. Several preserved ids are already used by other Ethereum blockchain networks and before choosing one a check must be done.

Choose a consensus algorithm

The consensus mechanism will be used in the blockchain. Ethereum blockchain networks can either use Proof of Work or Proof of Authority.

Creating the Genesis Block

Every blockchain starts with the genesis block. When Geth runs with the default settings for the first time, it commits the main net genesis to the database. For a private network, usually, a different genesis block is needed. The genesis block is configured using a JSON file. For the creation of the configuration of the genesis block decisions must be made on certain initial parameters for the blockchain network.

- ChainID which is explained above
- Initial gas block limit - this choice impacts how much EVM computation can happen in a single block. The block gas limit can be adjusted after launch using the following command-line flag

  ```
  --miner.gastarget
  ```

- Initial allocation of ether (alloc). This determines how much ether is available to the addresses that are listed in the genesis block
- Consensus engine to be used - Ethash (PoW) or Clique (PoA)
- Difficulty - the mining difficulty

Genesis block creation can be done with the help of puppeth [27] the Ethereum private network manager, by opening a terminal and executing puppeth command and following the prompts. When the procedure is completed the genesis JSON file will be created and additional changes can be done.

Initializing the Geth Database

The Ethereum node database must be initialized so that it uses the genesis block that was created for the private blockchain network. To initialize a node the following command must be executed

```
geth --datadir node/ init /path/to/genesis.json
```

Set up Networking

For the blockchain network nodes to be connected all the connections must be specified by either adding every node to every other as a peer or by adding a bootnode, which will be responsible to let every node know about the existence of every other node in the network. Bootnode was preferred for this implementation so that the peer-to-peer network can be set up. To create a bootnode the following commands must be executed

1. To create the enode

   ```
   bootnode -genkey boot.key
   ```

   The enode uniquely identifies the bootnode and it is stored in the boot.key file

2. To start the bootnode

   ```
   bootnode -nodekey boot.key -verbosity 9 -addr :30310
   ```

   where nodekey is given as an argument (the key generated in step 1), verbosity to provide as many details as possible and the address flag is where the bootnode is located

5.1.3 Steps - Execution

The following citations [28] [29] in combination with the Geth documentation led to the following actions on how to set up a private network.

Choose a network ID

The network ID for the private blockchain network that will be used is 980418, a random integer that is not reserved from any other public blockchain Ethereum network.

Choose a consensus algorithm

Proof of Authority was chosen as a consensus algorithm for this private blockchain network, as all the participant nodes are pre-authenticated and this mechanism provides high transaction and performance compared to Proof of Work, which is demanding high computing power, as discussed in Section 2.5.

Create Workspace

The workspace is where the required files and folders are located for the blockchain. Each node is meant to have the genesis.json file, a folder where all the data of the blockchain are saved (sync with all the other nodes/participants) and optionally some aliases for the commands so as not to specify every time all the flags and options.

The process that will be presented below is about one node of the blockchain network. To create other nodes, the same steps must be followed on another computer. Also, at this point, it is considered that all the necessary software is installed on the computer and that a bootnode exists, is created and running as described in Section 5.1.2 and specifically in subsection 'Set up Networking'.

1. Create a new folder for example named private-network
2. Change directory in the terminal to the private-network folder
3. Create a new folder for example named node
4. Execute the following command to create a new account

   ```
   geth --datadir node/ account new
   ```

5. Store both the Public Address of the account and its password to respective files inside the node folder

Bash aliases

The aliases that are mentioned below are an example of how they could be set to avoid these long commands. In addition, all of the aliases refer to the local network (localhost).

```
1 # Start bootnode
2 alias bootnode-start="bootnode -nodekey boot.key -verbosity 9 -addr :30310"
3 # start node
4 alias node-start="geth --datadir node/ --syncmode 'full' --port 30311 --allow-insecure
    -unlock --miner.gasprice 0 --http --http.vhosts '*' --http.corsdomain '*' --http.
    addr 'localhost' --http.port 8501 --http.api 'admin,personal,eth,net,web3,txpool,
    miner' --bootnodes [define enode as string] --networkid [networkID] -unlock [
    define account as string] --password node/password.txt --mine"
5 # attach to Node
6 alias node-attach="geth attach http://127.0.0.1:8501"
7 # init Node
8 alias node-init="geth --datadir node/ init /path/to/genesis.json"
```

Run Node and Attach

To run a node the command with the alias node-start mentioned above must be executed. This specific command has many options which are explained below from the help command of Geth [30]:

- datadir: Data directory for the databases and keystore (default: "/home/user/.ethereum")
- syncmode: Blockchain sync mode ("snap", "full" or "light") (default: snap)
- port: Network listening port (default: 30303)
- allow-insecure-unlock: Allow insecure account unlocking when account-related RPCs are exposed by http
- miner.gasprice: Minimum gas price for mining a transaction (default: 1000000000)
- http: Enable the HTTP-RPC server
- http.vhosts: Comma separated list of virtual hostnames from which to accept requests (server enforced). Accepts '*' wildcard. (default: "localhost")
- http.corsdomain: Comma separated list of domains from which to accept cross origin requests (browser enforced)
- http.addr: HTTP-RPC server listening interface (default: "localhost")
- http.port: HTTP-RPC server listening port (default: 8545)
- http.api: API's offered over the HTTP-RPC interface
- bootnodes: Comma separated enode URLs for P2P discovery bootstrap
- networkid: Explicitly set network id (integer)(For testnets: use –ropsten, –rinkeby, –goerli instead) (default: 1)
- unlock: Comma separated list of accounts to unlock
- password: Password file to use for non-interactive password input
- mine: Enable mining

To attach to a running node the command with the alias node-attach mentioned in the Bash aliases subsection needs to be executed. When attached to a node, several commands can be executed as the Geth JavaScript console exposes administrative APIs as well as the Web3 API. Such commands are eth.blockNumber which retrieves the latest block number of the blockchain, admin.addPeer to add a new peer (if a bootnode is not defined this is how the connection between participants can be achieved) et cetera [31] [32].

Creating the Genesis Block

As indicated previously, certain parameters must be defined for the genesis block of the private Ethereum blockchain. Below the JSON file is presented and every parameter used will be explained [33] [34]

All the accounts of each node can be prefunded with Ether to feed the users, as the cryptocurrency in this private blockchain does not have any actual value.

```
{
  "config": {
    "chainId": 980418,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000"
        ,
    "eip155Block": 0,
    "eip158Block": 0,
    "byzantiumBlock": 0,
    "clique": {
      "period": 0,
      "epoch": 30000
    }
  },
  "nonce": "0x0",
  "timestamp": "0x61aa0a45",
  "extraData": "0x0000000000000000000000000000000000000000000000000000000000000000C63
      Bc77e188C47922038b69a9d199d295Ed9209d266E28F24ceaF492Eab933d32D8929958E489BA3000
      0000000000000000000000000000000000000000000000000000000000000000000000000000000
      0000000000000000000000000000000000000000000000000000000000000000",
  "gasLimit": "0x47b760",
  "difficulty": "0x1",
  "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000000000000000000",
  "alloc": {
    "C63Bc77e188C47922038b69a9d199d295Ed9209d": {
      "balance": "0x2000000000000000000000000000000000000000000000000000000000000000"
    },
    "266E28F24ceaF492Eab933d32D8929958E489BA3": {
      "balance": "0x2000000000000000000000000000000000000000000000000000000000000000"
    }
  }
}
```

- chainId: The unique identifier of the blockchain network
- homesteadBlock: When set to 0 the network will be using the Homestead release of Ethereum. The mainnet (chainId = 1) has also this parameter set to 0
- eip150Block: EIP stands for Ethereum Improvement Proposal. As Ethereum is an open-source blockchain-based platform, developers can make proposals in the form of discussions and code. Some are accepted, others rejected. EIP150 is one such proposal that was accepted. This EIP took effect on block 2463000 and had mostly to do with increasing gas prices in response to denial-of-service concerns
- eip150Hash: The hash of the EIP150Block, which is needed for fast sync
- eip155Block: As mentioned above this is a proposal from developers which was accepted by the Ethereum platform. It aims the prevention of replay attacks
- eip158Block: Again an accepted proposal which changed how Ethereum clients deal with empty accounts. This new protocol began treating them as nonexistent, saving space on the blockchain
- byzantiumBlock: The Byzantium hard fork (it splits the blockchain in two, creating an old and a new version, where the new and old versions are incompatible, and all transactions are recorded on the new chain) was designed to make Ethereum BC lighter, faster and more secure [35]
- clique: The blockchain network will be using the PoA consensus mechanism
- nonce and mixHash: Used together to determine if the block was mined properly (PoW). These two values work together because, if an attacker forges blocks with a false nonce, it can and will be computationally costly for the other participants of the network to discover this falsy value. So mixHash is an intermediate calculation to find the nonce without so much computational power. Therefore, if other nodes on the network discover an errant mixHash when validating a block, they can discard the block without doing additional work to check the nonce
- timestamp: The output of the Unix time() function when the block was created
- extraData: An optional free, but max. 32-byte long space to conserve smart things for ethernity on the Blockchain
- gasLimit: The maximum number of computations any block on that chain can support
- difficulty: This value represents how hard it is to mine a block (PoW). Different blockchain networks use different mining algorithms
- coinbase: The 160-bit address to which all rewards (in Ether) collected from the successful mining of this block has been transferred. They are a sum of the mining

reward itself and the Contract transaction execution refunds. Often named "beneficiary" in the specifications, sometimes "etherbase" in the online documentation. This can be anything in the Genesis Block since the value is set by the setting of the Miner when a new Block is created

- alloc: This is the field which determines which accounts start with a certain balance of ether. In the Ethereum mainnet, this consisted of all the lucky ones that participated in the Ethereum presale

5.2 Determine dApp development technology

5.2.1 Back-End

In a dApp, smart contracts are used to store the business logic (program code) and the related state of your application. Smart contracts can be thought of as a replacement for the server-side (back-end) component of a normal application. One of the main differences between smart contracts and a back-end component is that any computation executed in a smart contract comes with a cost. It is therefore important to identify which aspects of the application need a trusted and decentralized execution platform. Ethereum smart contracts allow to build architectures in which smart contracts can call each other and pass data, read and write their own state variables and have no restrictions beyond the block gas limit. When a smart contract is deployed in the blockchain network, any developer could use it to build other dApps [36].

Smart contracts can not be changed, as they're a part of the blockchain ledger once they're deployed. However, they can be paused from the owner, which makes them unusable or even deleted if they are programmed with an accessible self-destruct opcode [37].

Finally smart contracts may cost a lot regarding to its architecture. A large monolithic smart contract may be expensive to deploy and use. Thus, some decentralized applications make some computations off-chain and have only a part of it on the blockchain. This means that every user in the dApp must trust these external resources for the off-chain computations [36].

For the Grades System dApp, the smart contracts, Solidity was the only option available, as the Blockchain chosen is Ethereum.

Front-End

To develop a front-end for a decentralized application, no new skills must be learned. Developers can use tools, frameworks and libraries they're familiar with. Every front-end dApp is usually linked with the Web3 library which offers the interaction with the blockchain and eventually the smart contracts.

At first, due to familiarity, NodeJS [38] was used for the creation of the decentralized application. As previously stated, this decision resulted in challenges that were difficult to overcome. More specifically, the core packages used by decentralized applications were

outdated and had severe vulnerabilities. Also, MetaMask [39] could not be integrated easily into this version of the dApp, as all the calls of the functions of the smart contracts were in the back-end. Because all the calls were in the back-end the global API that MetaMask injects could not be used, as it is defined only in a browser environment and cannot be sent to the back-end. Hence, crucial changes had to take place and NextJS was chosen for the implementation.

NextJS [40] offered essential features that would make development easier and the final result more product-like. First of all, the packages needed were up to date, bug-free and without any vulnerabilities. he integration with MetaMask was straightforward and the code could be broken down to components thanks to ReactJS [41]. Finally, the community and the resources for decentralized applications' development contributed in favour of NextJS.

5.3 Implement the smart contract

The smart contract was implemented with solidity compiler 0.8.12 version. The functionalities it offers are voting, different permissions on the users and storing. The smart contract was initially implemented with the help of Remix [42], an open-source Ethereum IDE you can use to write, compile and debug Solidity code.

Due to the fact that the whole logic would run on a private Ethereum Blockchain, the deployment and use costs of the smart contract did not affect the development. All the eleven nodes of the private blockchain are prefunded with a vast amount of ether, and are able to fund any new users that were accepted by all the other users to use the decentralized application. An activity diagram of the process followed is presented below.

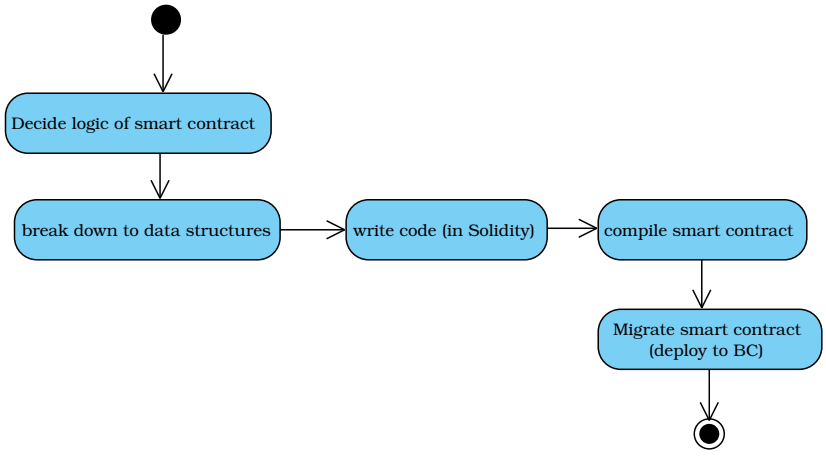


Figure 5.2. *Steps followed to develop the Smart Contract*

5.3.1 Smart Contract UML Class Diagram

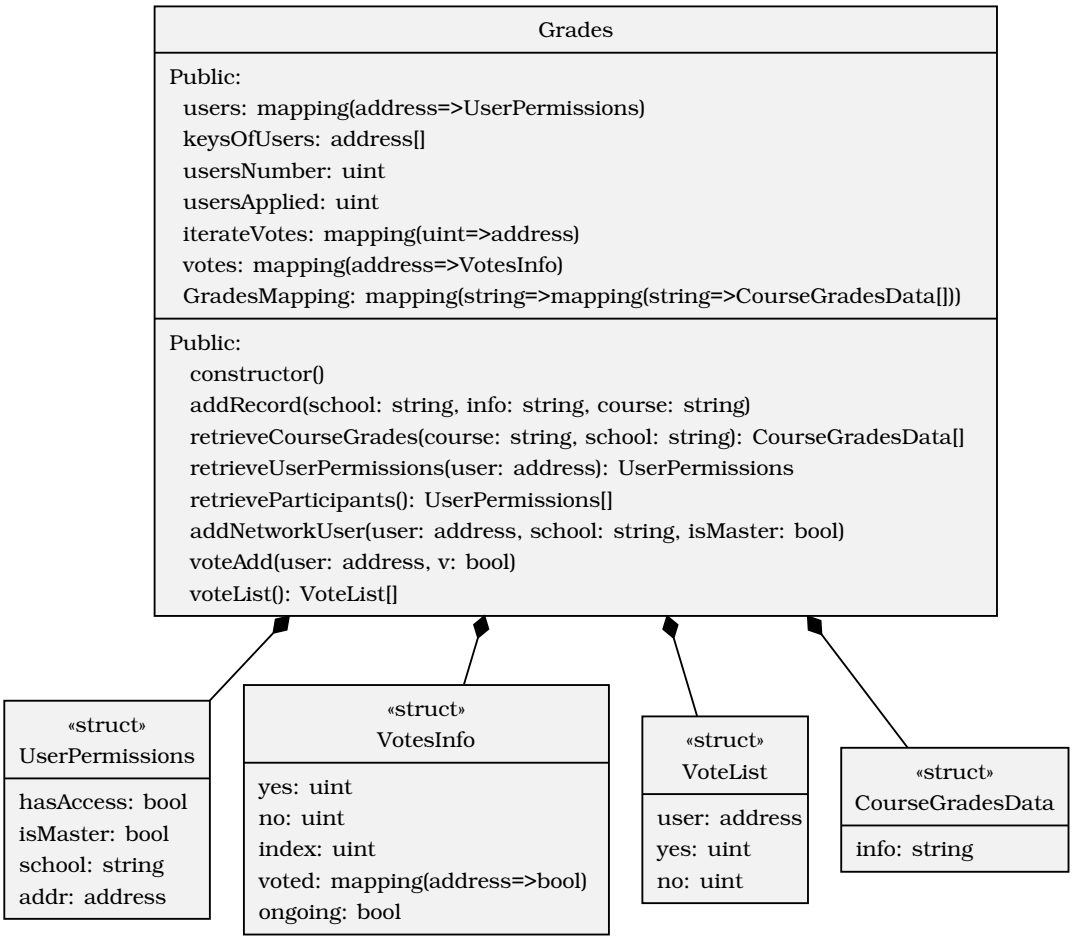


Figure 5.3. *Smart Contract UML Class Diagram*

5.3.2 Structures and Variables

Structures

Below all the structures that were used in the development of the smart contract are shown with comments.

```
// permissions of users
struct UserPermissions {
    bool hasAccess;     // if the user has access to the dApp
    bool isMaster;      // if the user is master
    string school;      // where the user belongs (in which school)
    address addr;       // public address of the user
}
```

```
// information for voting in order a new user to be added
struct VotesInfo {
    uint yes;                           // number of users said yes
    uint no;                            // number of users said no
    uint index;                         // used to bind votes mapping with iterateVotes (for iteration)
    mapping(address => bool) voted;     // which users (participants) voted for this specific user
    bool ongoing;                       // if there is an ongoing vote for this specific user
}
```

```
// used to construct the voting list that will be sent to a participant
// to see all users that applied to be able to use the dApp
// a participant user will have the opportunity to vote for every applying user
struct VoteList {
    address user;
    uint yes;
    uint no;
}
```

```
struct CourseGradesData {
    string info;
}
```

Variables

Below all the variables that were used in the development of the smart contract are shown with comments.

```
mapping(address => UserPermissions) users; // hold each user's permissions
address[] keysOfUsers; // holds the keys of the users
uint usersNumber; // holds the number of the users
// number of users that have applied (increases/decreases based on the status of all votes)
uint usersApplied;
// bind addresses with a uint (index of VotesInfo) so as to iterate the votes mapping
mapping(uint => address) iterateVotes;
// holds the votes information for every user that tries to be inserted to the network
mapping(address => VotesInfo) votes;
// mapping from school to mapping of string(course) to courseGradesData(JSON) information
mapping(string => mapping(string => CourseGradesData[])) GradesMapping;
```

5.3.3 Functions

Table 5.1. *Smart Contract Functions*

Function	Description
`addRecord(string memory school, string memory info, string memory course)`	Add new grade information
`retrieveCourseGrades(string memory course, string memory school) public view returns (CourseGradesData[] memory)`	Retrieve course information
`retrieveUserPermissions(address user) public view returns(UserPermissions memory)`	Return permissions of a user
`retrieveParticipants() public view returns( UserPermissions[] memory)`	Return all participants(users)
`startUserVote(address user, string memory school, bool isMaster)`	Start vote for a user
`voteAdd(address node, bool v) public`	Vote for or against
`voteList() public view returns(VoteList[] memory)`	Return the Vote list of a user

5.3.4 Compile and Migrate Smart Contracts

Truffle

Truffle is a product of the TruffleSuite [43], which is a world class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier. It offers several features such as [44]:

- Built-in smart contract compilation, linking, deployment and binary management.
- Automated contract testing for rapid development.
- Scriptable, extensible deployment migrations framework.
- Network management for deploying to any number of public private networks.
- Package management with EthPM NPM, using the ERC190 standard.
- Interactive console for direct contract communication.

- Configurable build pipeline with support for tight integration.
- External script runner that executes scripts within a Truffle environment.

Thus, truffle is used for the compilation and deployment of the Solidity smart contracts.

Installation

To install truffle a NPM must be installed and from the terminal the following command must be executed:

```
npm install truffle -g
```

Initialize Workspace

1. Create a new directory i.e named truffle and move inside
2. To create a bare Truffle project with no smart contracts included:

   ```
   truffle init
   ```

   When the initialization is completed four assets will appear

   - contracts/: Directory for Solidity contracts
   - migrations/: Directory for scriptable deployment files
   - test/: Directory for test files for testing your application and contracts
   - truffle-config.js: Truffle configuration file
3. Add smart contracts in the contracts directory

Set-up truffle-config

This particular file has several options such as the blockchain networks and which solidity compiler will be used.

```
module.exports = {
  rpc: {
    host: [IP],
    port: [NODE_PORT]
  },
  networks: {
    development: {
      host: [IP],
      port: [NODE_PORT],
      network_id: [NETWORK_ID],
      from: [NODE_ADDRESS],
      gas: [GAS_PRICE]
    },
  },
  compilers: {
    solc: {
      version: "0.8.12"
    }
  }
};
```

Networks truffle-config

Specifies which networks are available for deployment during migrations, as well as specific transaction parameters when interacting with each network (such as gas price, from address, etc.). When compiling and running migrations on a specific network, contract artifacts will be saved and recorded for later use. The networks object, is keyed by network names and each name contains a corresponding object that defines the parameters of the network [45]. When networks are defined, names can be provided as an option for certain commands:

```
truffle migrate --network live
```

Compile

To compile the smart contracts the following command must run inside the folder of the truffle project:

```
truffle compile
```

Migrate

To deploy the smart contracts the following command must run inside the folder of the truffle project:

```
truffle migrate
```

5.4 Implement the dApp Front-End

In the following subsections a basic overview of the NextJS implementation is presented. In addition, an activity UML diagram is presented with the steps that were followed during the implementation.

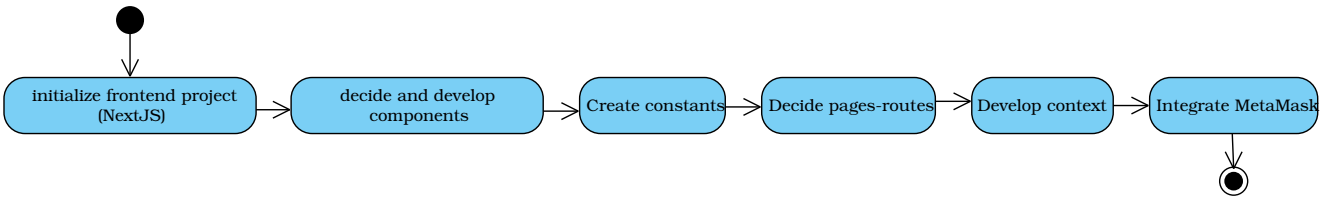


Figure 5.4. *Steps followed to develop the dApp*

5.4.1 Structure NextJS project

```
├──components
│   ├──Accordion
│   ├──Card
│   ├──DiffModal
│   ├──DownloadButton
│   ├──FileInputReader
│   ├──Footer
│   ├──Navbar
│   └──WithAuth
├──constants
├──pages
│   ├──404
│   ├──about
│   ├──api
│   │   ├──diff
│   │   └──form
│   │       ├──grades-info
│   │       └──user
│   ├──form
│   │   └──add
│   │       ├──grades-info
│   │       └──user
│   ├──login
│   ├──menu
│   └──show
│       ├──courses
│       ├──participants
│       ├──vote-list
│       └──[school]
│           └──course
│               └──[code]
├──public
│   └──images
├──store
└──styles
```

Figure 5.5. *NextJS Project Structure*

All folders except public and styles will be discussed in the following subsections.

Components

Components are independent and reusable bits of code and have the same purpose as a function. Instead returning a result they return HTML. There are two types of components, functional and class components. This implementation used functional components [46].

Several components were designed and created for this project. More specifically:

- Accordion component, which allows the user to show or hide sections of related content in a page. This component was used in the page where a specific course's information are shown grouped by exam period
- Card component, which contain content and actions about a single subject
- DiffModal component, a custom modal which appears only if a difference was found between the blockchain data and the data stored in the file. The modal that appears in case of a difference, is the visualization of a unified diff string [47]
- DownloadButton component, which was used inside the DiffModal. More specifically two DownloadButtons exist so as the user can download the content the file stored in the blockchain and the current content of the file
- FileInputReader component, which handles the file uploaded from the Add Grades page
- Footer component, which is the footer of the front-end
- Navbar component, which is the navigation bar of the front-end
- WithAuth component, a Higher Order Component (HOC) [48] which wraps all the pages that the user must be authenticated

Constants

Constants contained in the folder:

- The Contract Application Binary Interface (ABI), which is the standard way to interact with contracts in the Ethereum ecosystem, both from outside the blockchain and for contract-to-contract interaction [49]
- contract-addresses, which is an object containing key value pairs of all the addresses of the smart contracts with their name
- schools-info, which contains the information needed to render the courses of a specific school

Pages - Routes

In Next.js, a page is a React Component exported from a .js, .jsx, .ts, or .tsx file in the pages directory. Each page is associated with a route based on its file name [50].

In addition NextJS offers a solution to build API routes in the front-end. Any file inside the folder pages/api is mapped to /api/* and will be treated as an API endpoint instead of a page. They are server-side only bundles and won't increase the client-side bundle size [51].

All the pages (and routes) created, followed the design that was set from the wire-frames. More specifically:

- 404 page renders when the path the user tries to reach does not exist
- About page (/about) has some basic information about the project
- API is the special route that NextJS offers to build API routes which must return a response other than HTML. The API route of the project has two sub-routes, diff and form. Diff sub-route constructs the unified diff string if differences exist between the content of the file saved in the blockchain and the current data of the file. Form sub-route breaks down into two other sub-routes grades-info and user. Both routes under the /api/form exist for validation check of the forms
- Login page (/login) of the dApp (integrates MetaMask)
- Menu page (/menu), which loads if the authentication is successful. The user uses MetaMask to connect to the private Ethereum Blockchain network, but as soon as he is connected, the data of the user with that public address are retrieved from the blockchain network. If the user has access then Menu page is loaded with the corresponding options
- Show Courses page (/show/courses), which renders all the courses that the user has access to retrieve data
- Show Participants page (/show/participants), which renders all the users of the dApp
- Show VoteList page (/show/vote-list), which renders all the pending votes the user has
- Show specified Course page (/show/[school]/course/[code]) which renders all the information retrieved for a specific course. Every record in this page has a validation button. When this button is pressed, the diff check happens and in case of any differences the DiffModal loads

Context

In a typical React application, data is passed top-down via props, but this procedure can be cumbersome for props required by almost any component. This is where context comes in handy, as it provides a way to pass data through the component tree without having to pass props down manually at every level [52].

For this dApp two contexts were created:

- Contracts context, which holds all the contracts that are going to be used in many pages of the dApp. The reason this contract was created is that every time a function of the same smart contract would be invoked from a different component, the same action would be followed
- User context, which holds all the infromation of the user logged in

5.4.2 Integrate MetaMask

Many ways to use MetaMask in NextJS exist. In this project web3-react [53], which is one option to communicate with an Ethereum BC network and interact with the smart contracts. This library provides integration with a few digital wallets like MetaMask, WalletConnect and Coinbase-Wallet.

In this subsection a simple guide on how to integrate MetaMask in NextJS using Web3-React will be presented. In addition, dummy code will be shown on how to call a function of a smart contract which is deployed in a Blockchain network. Ethers.js library [54] will be used to retrieve the instance of the smart contract that is deployed. When the instance is created, function calls can be done and transactions will be sent on the Blockchain network.

Add Web3 Provider (_app.js)

```
import { Web3ReactProvider } from "@web3-react/core";
import { Web3Provider } from "@ethersproject/providers";

const getLibrary = (provider) => {
    return new Web3Provider(provider);
};

function App({ Component, pageProps }) {
    return (
        <Web3ReactProvider getLibrary={getLibrary}>     // provides Web3 Context to the Application
            <Component {...pageProps} />
        </Web3ReactProvider>
  );
}

export default App;
```

Use Web3-React to Connect

```
import { useWeb3React } from "@web3-react/core";
import { InjectedConnector } from "@web3-react/injected-connector";
import { useState, useEffect } from "react";

// injects window.ethereum to the browser
export const injected = new InjectedConnector();

export default function Home() {
    // state variable to decide if user has MetaMask installed
    const [hasMetamask, setHasMetamask] = useState(false);

    // Check if user has MetaMask installed and set the state variable to true
    useEffect(() => {
        if (typeof window.ethereum !== "undefined") {
          setHasMetamask(true);
        }
    });

    // variables and functions provided by web3 library
    const {
        active,             // true if user is connected
        activate,           // function to connect through metamask
        chainId,            // the ID of the chain user is connected to
        account,            // user's public address
        library: provider,  // the web3 provider
    } = useWeb3React();

    // connection function (can be bound to a button)
    // when clicked user is prompted with the MetaMask extension window to connect
    async function connect() {
        if (typeof window.ethereum !== "undefined") {
          try {
            await activate(injected);
            setHasMetamask(true);
          } catch (e) { console.log(e); }
        }
    }

    return (
        <div> {
          hasMetamask
          ? (
            active
                ? ( "Connected! " )
                : ( <button onClick={() => connect()}>Connect</button> )
          )
          : ( "Please install metamask" )
        } </div>
    );
}
```

Interact with Smart Contract using EthersJS

To interact with a smart contract, an instance of it must be created. To create an instance with Ethers library the following information are required [55]:

1. Public Address of the smart contract
2. ABI of the smart contract
3. Signer: this represents an Ethereum account that has the ability to sign transactions

As soon as the instance of the smart contract is defined, function calls can be done.

```
import { ethers } from "ethers";
// import ABI from constants folder
// ABI is created when compiling the smart contract
import { abi } from "../constants/abi";

const getLibrary = (provider) => {
    return new Web3Provider(provider);
};

export default function FunctionCallExample() {

    async function exec() {
        // get signer from the provider (MetaMask)
        // the signer is the public address of the account that is authenticated
        const signer = provider.getSigner();
        // the public address of the smart contract when deployed
        const contractAddress = [CONTRACT_ADDRESS_AS_STRING];
        //
        const contract = new ethers.Contract(contractAddress, abi, signer);
        try {
            // as soon as the contract instance is created, function calls can be done
            // We suppose that this function stores the number 42 in the blockchain ledger
            await contract.store(42);
        } catch (error) { console.log(error); }
    }

}
```

Add the private network to MetaMask

To add a new network in MetaMask the following steps must be followed [56]:

1. Open the MetaMask extension
2. Click the button on the top right corner
3. Click Add Network
4. Fill the form
   (a) Network Name: It is an identification for the user, nobody else can see it
   (b) New RPC Url: The URL of the node/participant of the private network that has the RPC interface enabled. Remote procedural call (RPC) communication interface provides remote connection and communication services to RPC clients
   (c) Chain ID: The ID (integer) that was assigned to the private network
   (d) Currency Symbol: Set to ETH

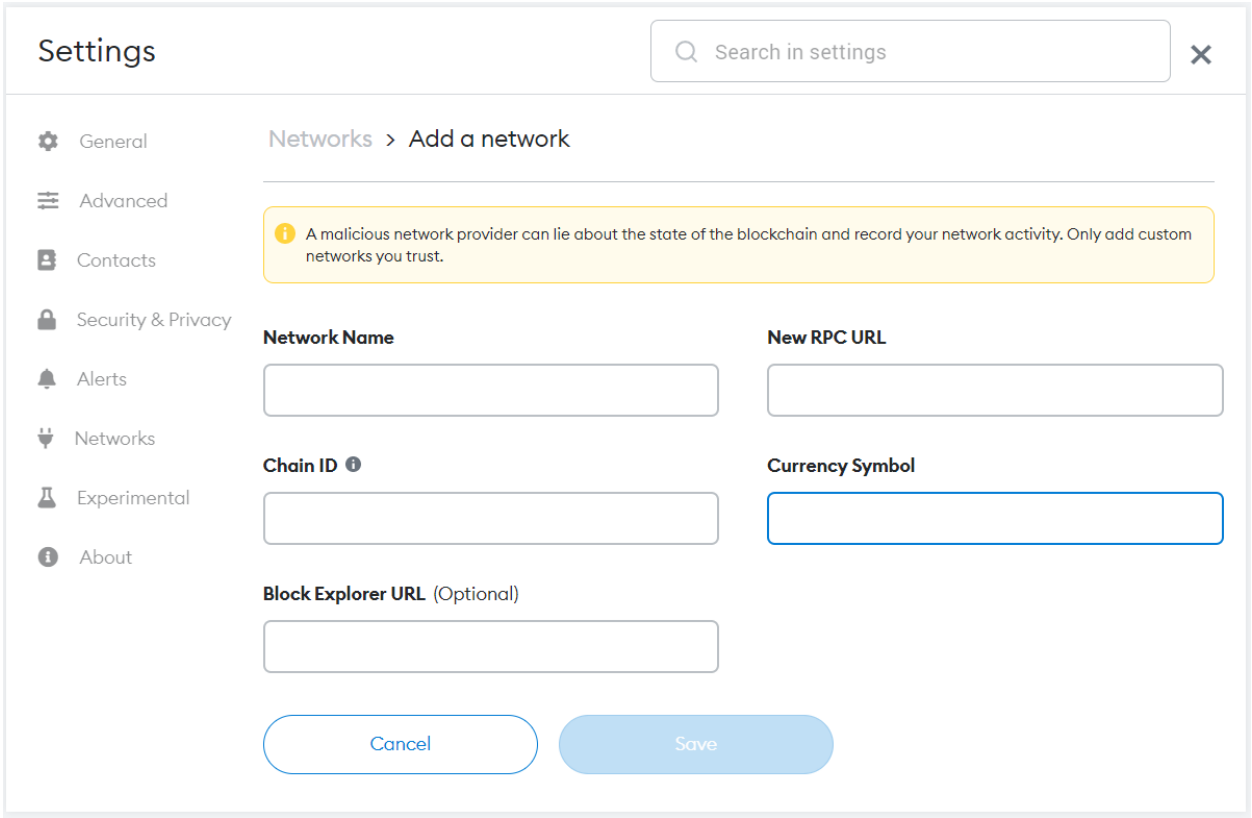


Figure 5.6. *MetaMask Add Network*

5.5 Dummy .bau files Generator

A specific files generator format had to be created. As mentioned above, the grades of a specific course are saved in a special file format which looks like CSV. To test the decentralized application and find any possible bugs, a script that creates files in this format was written. The script that is shown below uses Faker [57], an npm library which offers several functions that generate dummy data of all kinds.

```
let faker = require('faker');
var fs = require('fs');
DUMMY_DATA_NUMBER = 50;
DUMMY_FILES = 15;

for (let files = 0; files < DUMMY_FILES; files++) {
    let content = "U\n" + "{\n"
            + "#;School Code;" + faker.datatype.number({'min': 1,'max': 9}) + ";#\n"
            + "#;School Title;" + faker.lorem.word() + ";#\n"
            + "#;Exam Period Code Ticket;" + faker.datatype.number({'min': 1,'max': 9}) + ";#\n"
            + "#;Verbal Exam Period Ticket;" + faker.lorem.word() + ";#\n"
            + "#;Academic Year;" + faker.datatype.number({'min': 2000,'max': 2022}) + ";#\n"
            + "#;Semester Code X/0;" + faker.datatype.number({'min': 1,'max': 9}) + ";#\n"
            + "#;Semester Verbal X/0;" + faker.lorem.word() + ";#\n"
            + "#;Ticket Code;" + faker.datatype.number({'min': 1,'max': 9}) + ";#\n"
            + "#;Ticket Verbal;unified;#\n"
            + "#;Course Title;" + faker.lorem.word() + ";#\n"
            + "#;Course ID;"+ faker.datatype.number({'min': 1,'max': 9999}) + ";With grades;YES;#\n"
            + "#;Course's Semester;" + faker.datatype.number({'min': 1,'max': 9}) + ";#\n"
            + "#;Professors;" + faker.name.findName() + ";#\n"
            + "#;A/A;STUDENT.ID;LASTNAME;FIRSTNAME;FATHERNAME;STUDENT.SEMESTER;NN;GRADE;GRADE REVISED;
                FORA.ID;IMPROVEMENT;#\n";

    for (i = 0; i < DUMMY_DATA_NUMBER; i++) {
        [firstName, lastName] = faker.name.findName().split(" ");
        [fatherName, _] = faker.name.findName().split(" ");
        content += "#;" + (i+1) + ";"
                + faker.datatype.number({'min': 1000000,'max': 9999999}) + ";"
                + firstName + ";" + lastName + ";"
                + fatherName + ";"
                + faker.datatype.number({'min': 1,'max': 40}) + ";"
                + faker.datatype.number({'min': 1,'max': 40}) + ";"
                + faker.datatype.number({'min': 0,'max': 10}) + ";"
                + ";"
                + faker.datatype.number({'min': 1,'max': 10})
                + ";#\n"
    }
    content += "}";

    fs.writeFile('bauGenerator/dummy/dummy_bau_' + files + '.bau', content, (err) => {
        if (err) throw err;
        console.log('File is created successfully.');
    });
}
```

Chapter 6

Demonstration of the dApp

6.1 Login

6.1.1 Main Page

Main page of the dApp where user logs in through MetaMask. In case user does not have MetaMask installed as a browser extension, a paragraph will appear suggesting to the user to install the required extension.

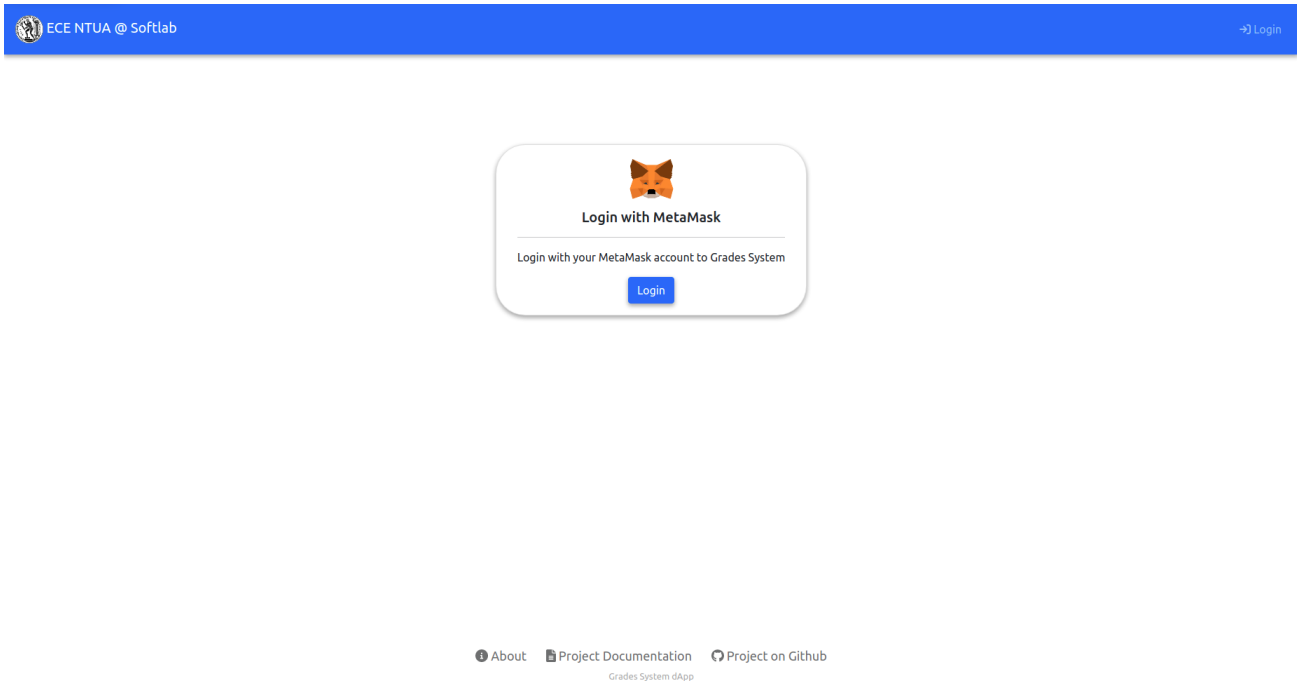


Figure 6.1. *Login Page*

6.1.2 Select Account

The user selects the account (public address) that wants to use to interact with the Blockchain.

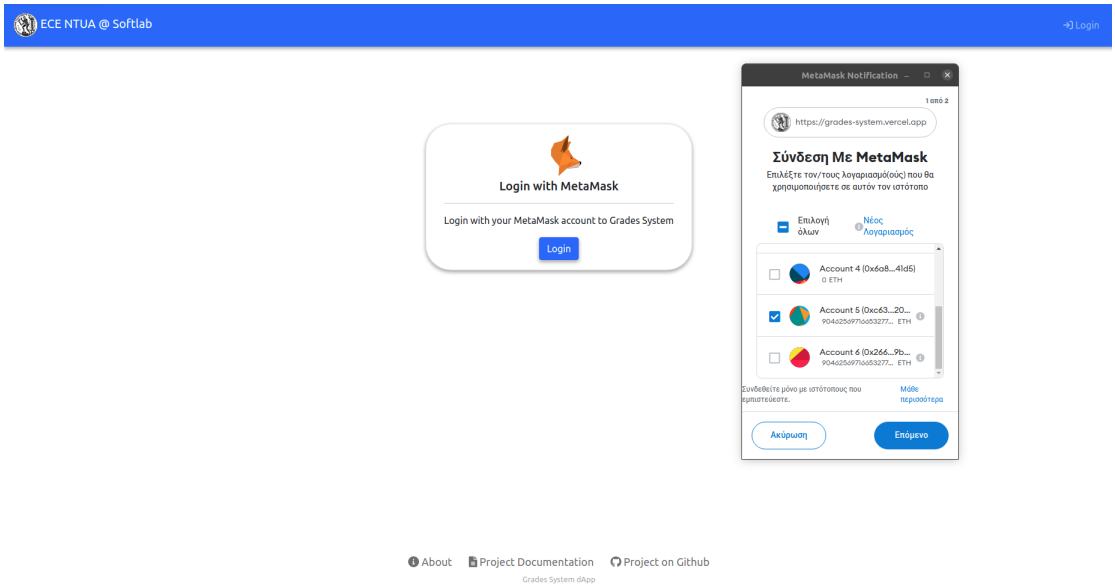


Figure 6.2. *Select account to connect with MetaMask*

6.1.3 Accept or Reject connection with MetaMask

The final step for the user to complete the login process. Users can either proceed to the connection or abort the login process.

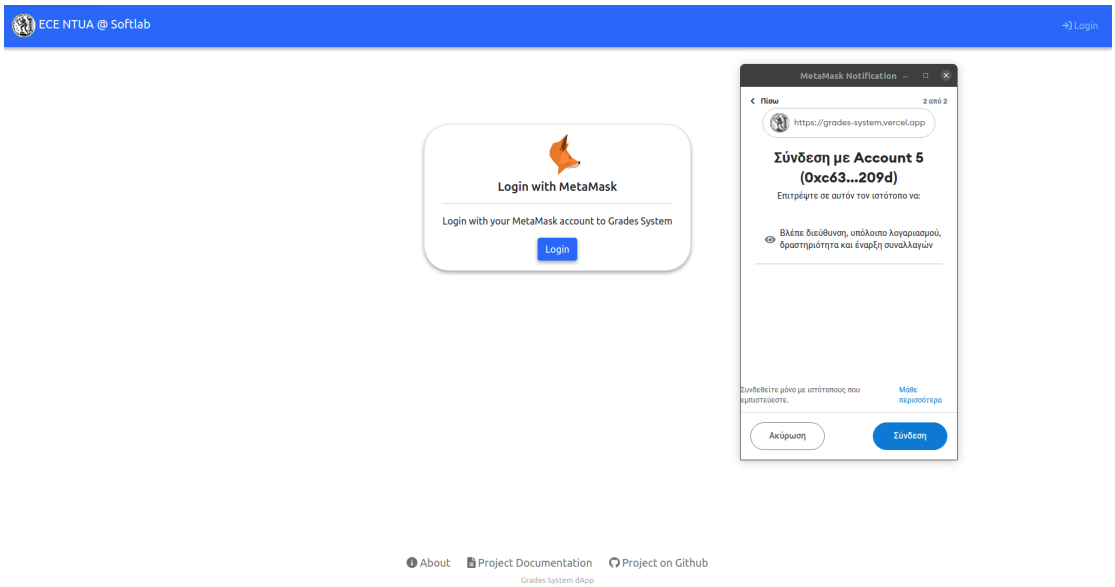


Figure 6.3. *Accept or reject connection with MetaMask*

6.2 Add Grades

6.2.1 Complete form

The form that users complete to add new grades information to the ledger of the blockchain. This new information can either be an initial or a corrective state.

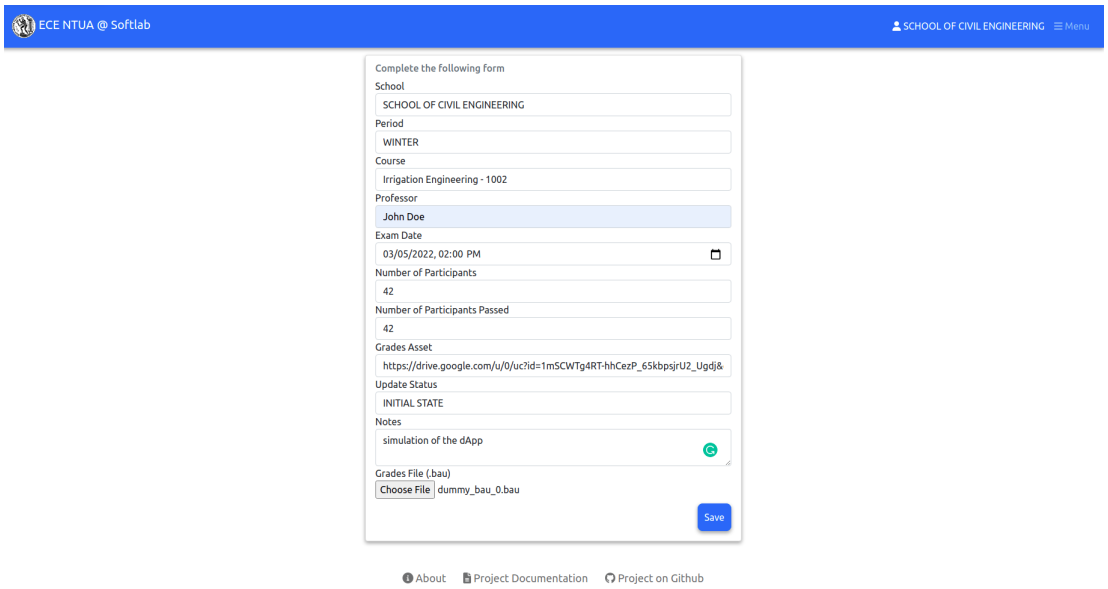


Figure 6.4. *Complete form*

6.2.2 MetaMask prompt

MetaMask prompts user to confirm or reject the transaction.

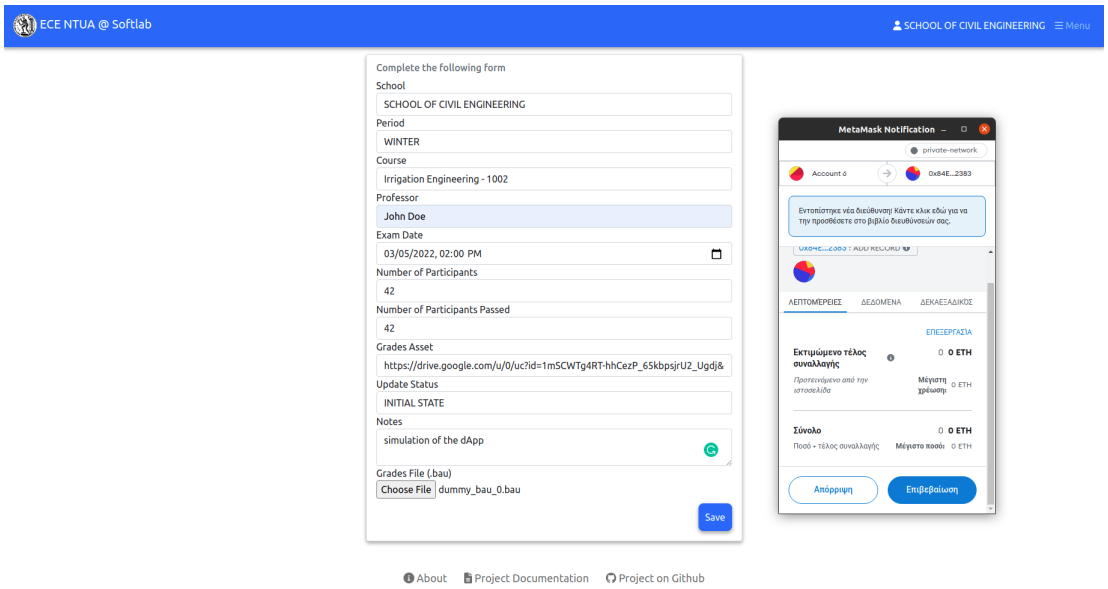


Figure 6.5. *MetaMask prompt*

6.2.3 Transaction confirmed

The above transaction that the user added some grade information is confirmed.

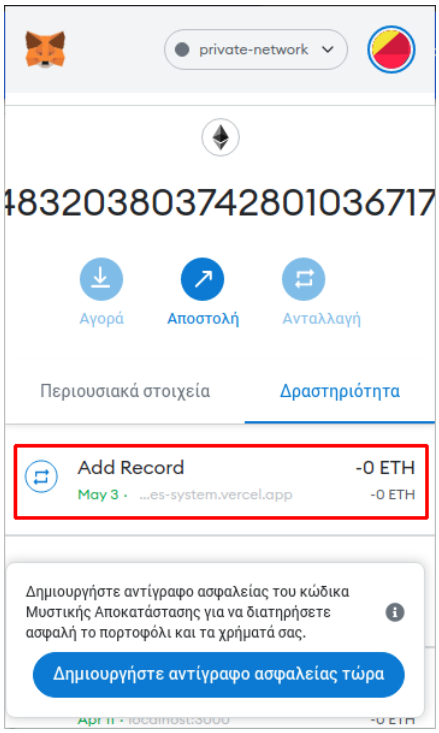


Figure 6.6. *Transaction confirmed*

6.3 Show Courses

6.3.1 Registrar User

The courses that this particular registrar user is authorized to retrieve. Since this registrar user is registered to the system as a registrar for the School of Civil Engineering, only the courses of this specific school are shown.

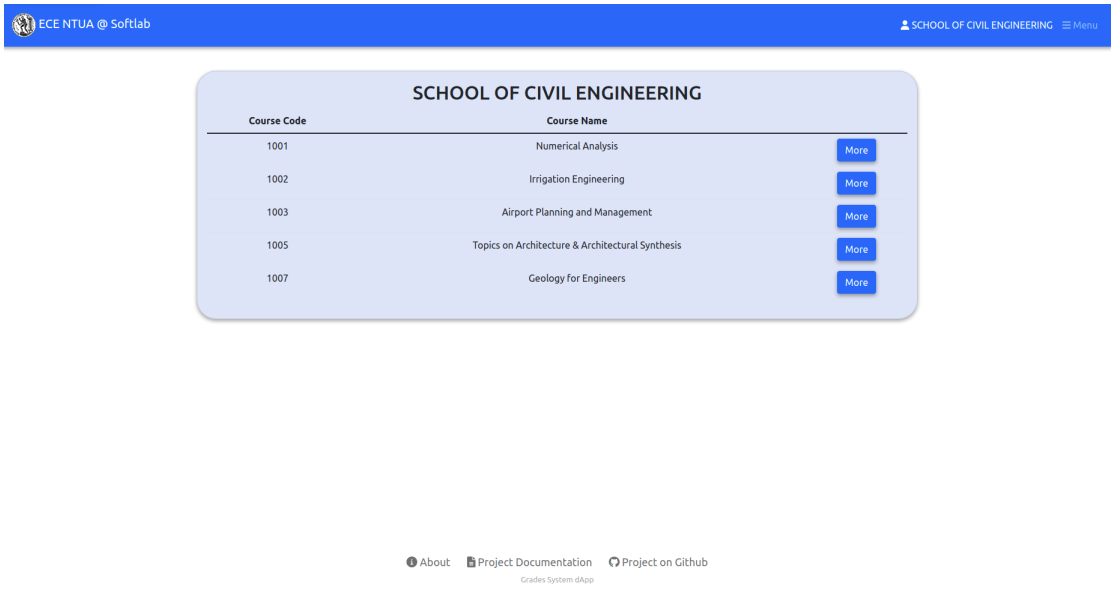


Figure 6.7. *Courses a registrar user can retrieve*

6.3.2 Master User

A master user can retrieve all courses information for every school, as shown below.

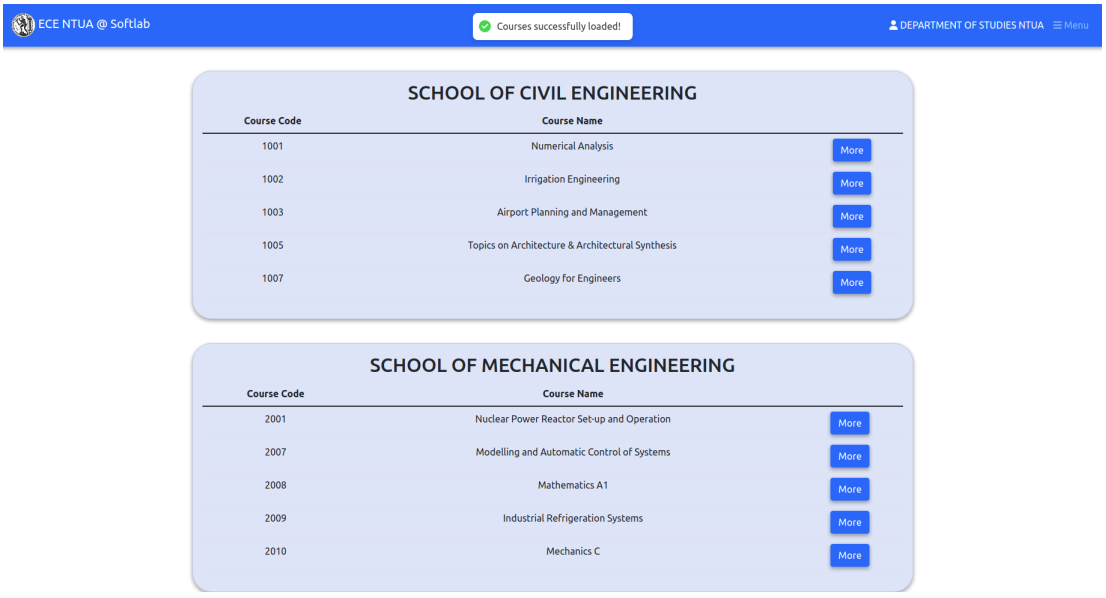


Figure 6.8. *Courses a master user can retrieve*

6.3.3 Retrieve information of specific course

Specific course information are shown when selecting a course. The information is retrieved from the blockchain and the data are grouped by the date and the exam period. In addition in each group many records can exist, since corrective state records can be inserted.

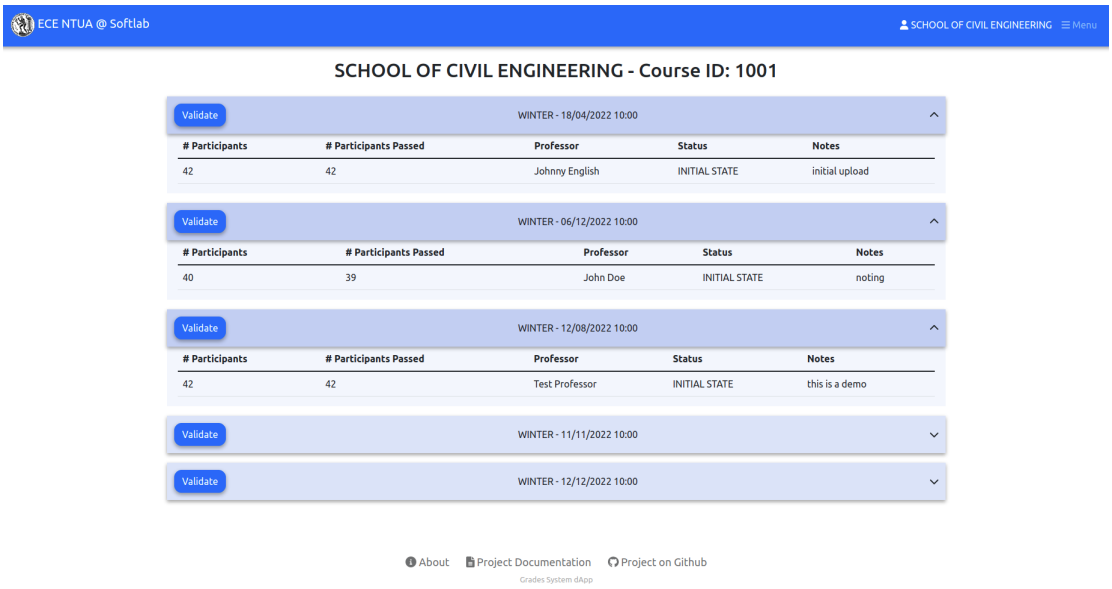


Figure 6.9. *Information retrieve for a specific course*

6.3.4 Validate information of a record

At any moment, users can perform validation of the latest record in each group of data. In case of any differences, a diff-like output is shown.

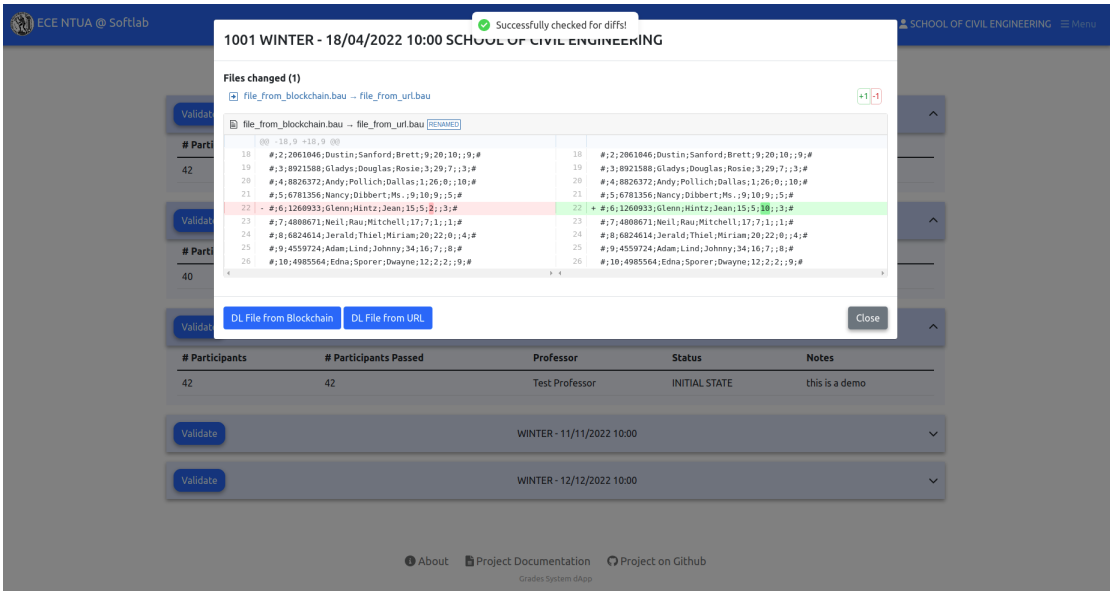


Figure 6.10. *Validate information - Diff found*

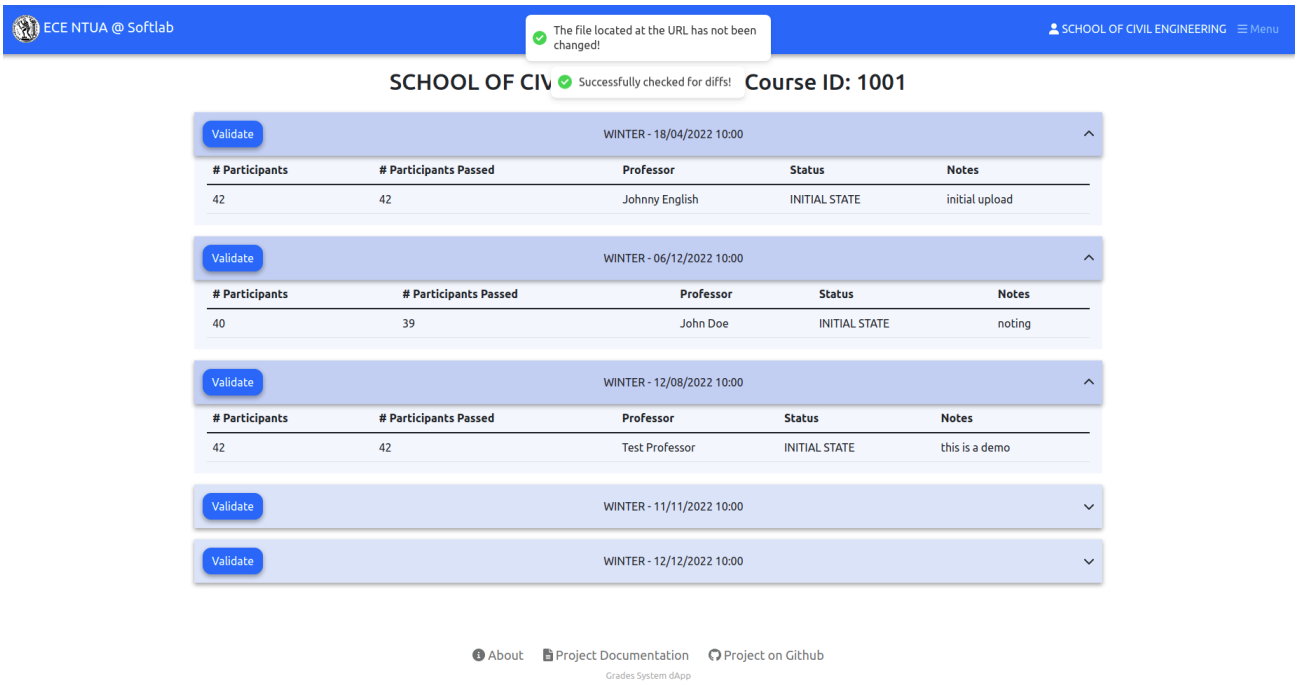


Figure 6.11. *Validate information - No Diff found*

6.4 Add a new user

6.4.1 Start a new vote - Complete form

A master user can start a new voting process for an applicant by completing the form.

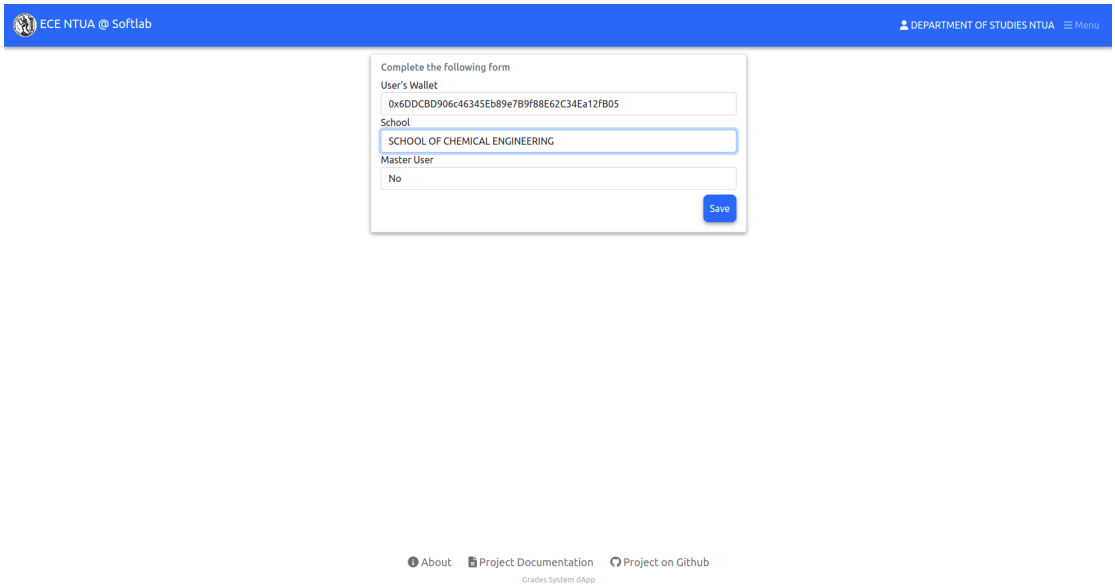


Figure 6.12. *Complete form*

6.4.2 MetaMask prompt

MetaMask prompts user to accept or reject the transaction.

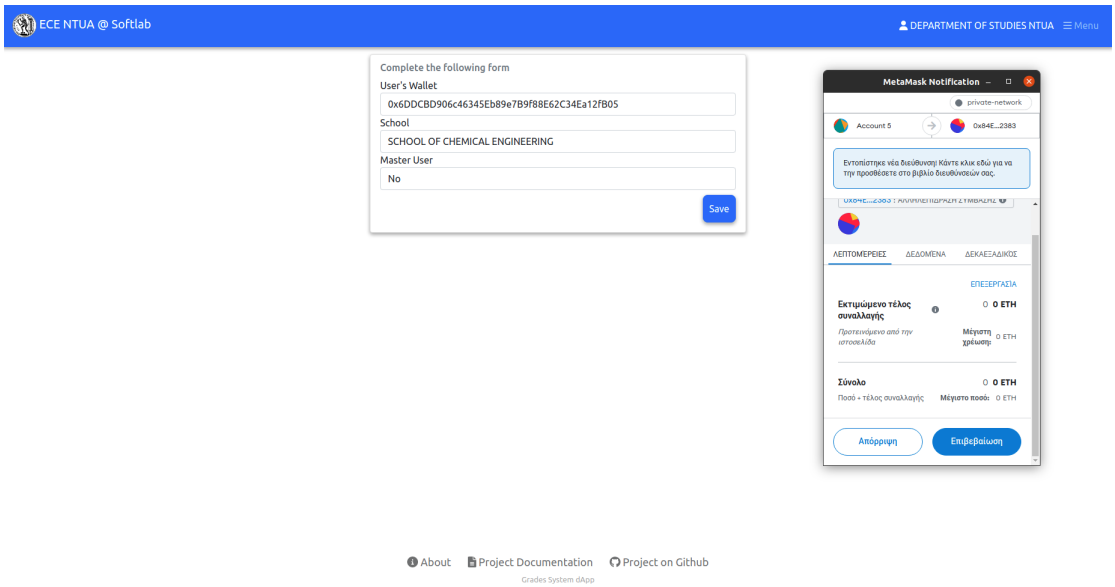


Figure 6.13. *MetaMask prompt*

6.4.3 Transaction confirmed

The above transaction that the user started a new voting process is confirmed.

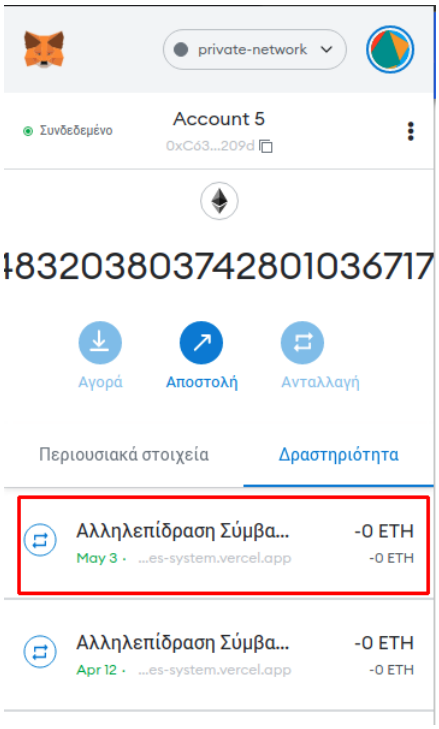


Figure 6.14. *Transaction confirmed*

6.5 Vote for or against

6.5.1 Retrieve pending votes

Every ongoing vote processes that a user has not yet voted are retrieved and shown. Then, user can either vote for or against an ongoing vote.

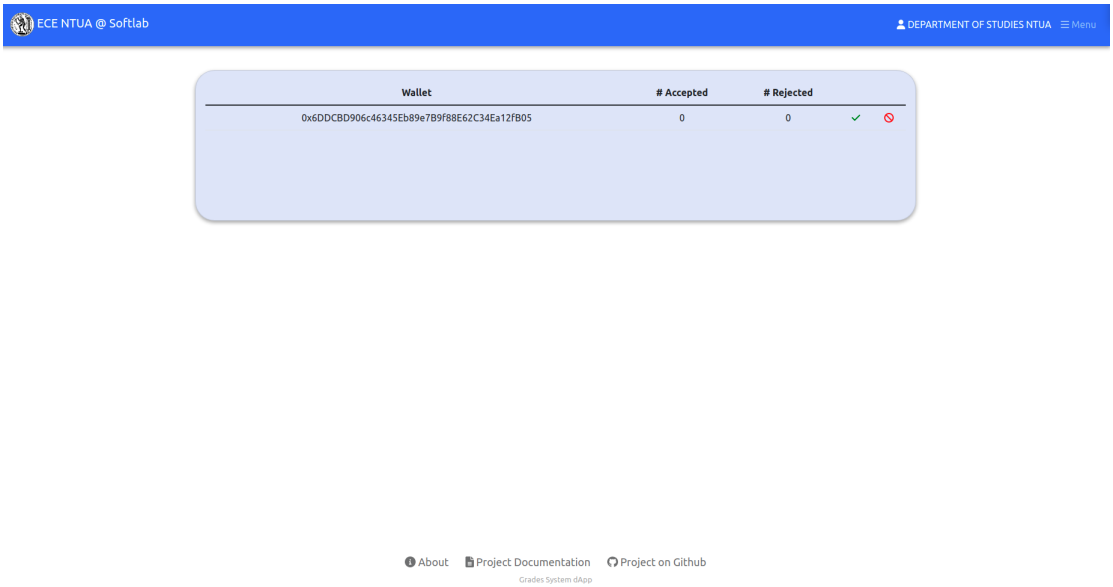


Figure 6.15. *Transaction confirmed*

6.5.2 MetaMask prompt

MetaMask prompts user to accept or reject the transaction.

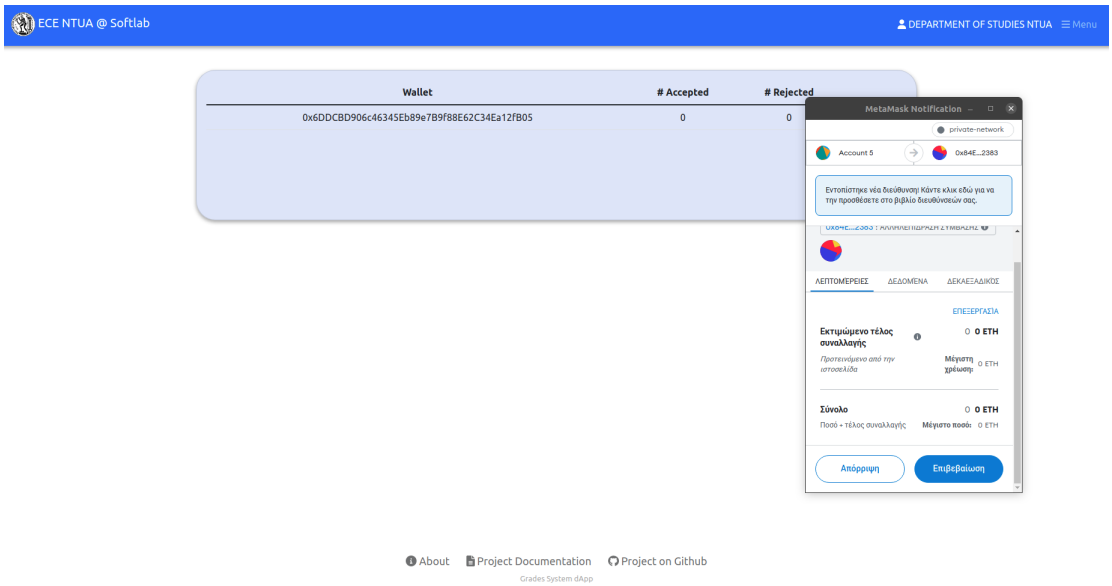


Figure 6.16. *Vote prompt MetaMask*

6.5.3 Transaction confirmed

The above transaction that the user voted in an ongoing vote is confirmed.

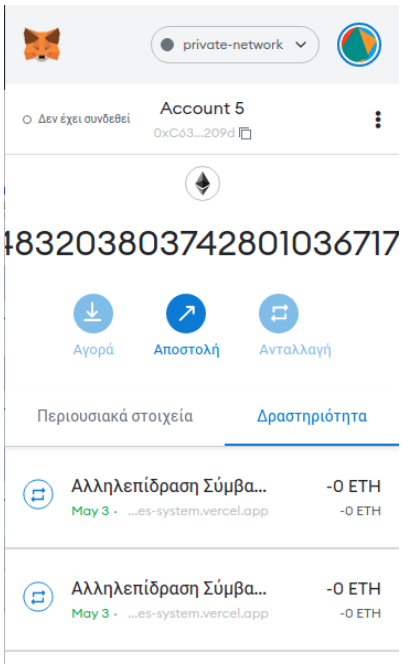


Figure 6.17. *Transaction confirmed*

6.6 Show Users

A Master user retrieves all users that have permission to interact with the decentralized application.

Figure 6.18. *Retrieve Users - Master User*

Chapter 7

Epilogue

In the present work, the creation of a system based on blockchain technology for the management of the grades of a university was examined. More specifically, it focused on creating a private network to which only certain university entities such as students, professors and registrars have access. Various features of the system were implemented, such as entering grades into the system, adding new members to the system, voting on whether or not a new member could use the system, and validating Blockchain data with an external data source to find any alternations of data. In our case, the external source is the file that the professor sent to the registrar and was eventually stored on a server. It is concluded that such a blockchain-based system can be implemented and offer all the major features of that technology to this problem. Integrity and security of the data that eventually end up in the legacy system can be achieved, as the Blockchain ledger can act as a validating mechanism between these two data sources.

7.1 Privacy

The privacy requirement is achieved as the network is set up accordingly. In addition a voting mechanism is implemented were all the participants are able to vote for or against a new user. This means that users must pass a voting process to have access to the decentralized application and the data that are stored in the Blockchain.

7.2 Integrity and Security

As mentioned above, integrity and security are offered by definition with Blockchain technology. Furthermore, a validating mechanism is implemented for the validation between the ledger of the Blockchain and the data that exists in the legacy system. This is a crucial feature as it can discover any inconsistencies of data.

7.3 Immediacy of procedures

With Blockchain technology this system can offer to every user immediacy of any procedure that in the legacy system requires time. For example a student can ask for a document (i.e all student's grades) but this request takes time as it involves a few steps.

More specifically, registrar has to see the request of the student, get the document ready, fetch the information from the legacy system and wait for the dean to sign. In contrary, when the ledger is immutable, this process can be avoided and the document can be derived instantly.

7.4 Further expansions

This section is focused on feature proposals that this system can be expanded with. Additionally, features that this thesis covers as proof of concept are discussed and explained on how they must be implemented in the real system.

1. One feature that is covered as proof of concept in this thesis is the validation between two sources. For demonstration purposes, this thesis validates the Blockchain ledger with the bau file that is stored in a server. In the production system, the distributed ledger should be compared with the data that exists in the legacy system's database. Thus, for this comparison, an API should exist to collect the necessary data for the comparison. Furthermore, it is important to note that the Blockchain system can be flexible; it is not necessary that it can only contain files in BAU form. Whatever the formation of the data exposed from the API which is fetched from the legacy system's database the dApp can support it.

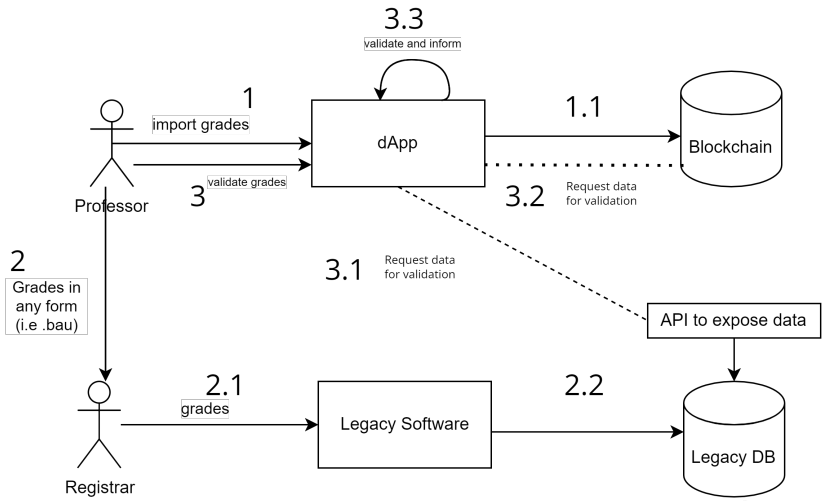


Figure 7.1. *Add grades and validate production workflow*

2. From the diagram above, it is clear that the suggested workflow is that the professors should add directly to the blockchain ledger the data instead of sending the file to the registrars. This could add further security as the file can not tamper with as it never leaves the source. Thus, the professors could add the data to the ledger, and send the file to the registrar which would use the specific software to register the students' grades to the legacy system. So, if the data ended up in the legacy system is tampered with, the changes can be tracked.

3. As the ledger holds students' grades, students could fetch all their information from the Blockchain without the need of requesting documents from the registrar, as the system can directly produce them and the certainty that the document is not fake. Also, students don't have to wait for the document to be created because there are no intermediaries which can slow down the process.

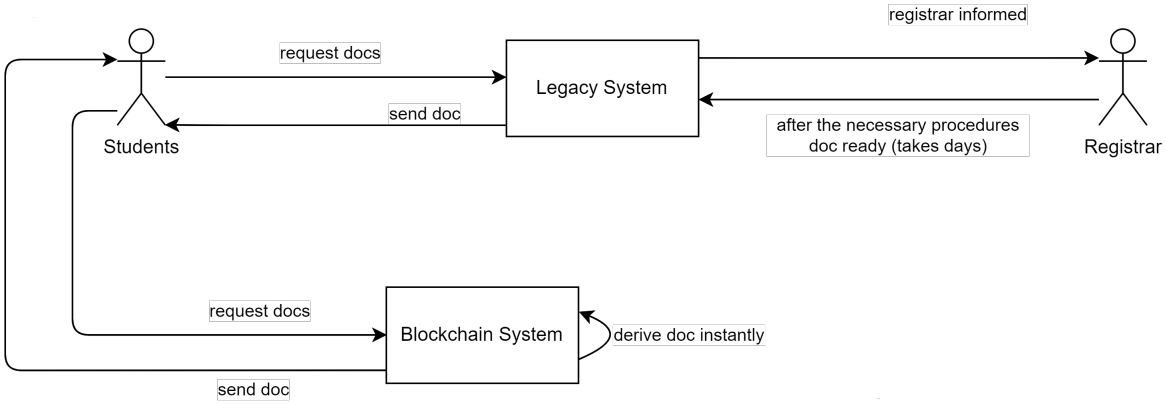


Figure 7.2. *Students requesting doc in Legacy System vs BC System*

4. Respectively, an API could be developed so that external entities such as companies can request data for specific students that they want to hire. This feature can ensure that companies fetch the correct data, with no alternations and without the need for the student to contact the school's registrar for the documents. Again, the process is similar to the above figure.

5. Similarly to the two previous points, students' process could be represented as a smart contract and when students complete all their courses an NFT [58] could be minted. This way, the system is kept decentralized without the need for a centralized system such as an API. In addition, different NFTs could be minted through the Blockchain system, such as an analytical document of all student's grades or an attendance certificate document.

6. This particular system stores complete grades files in a stringified form, which can be quite costly. Instead, these files could be stored in the InterPlanetary File System (IPFS) [59] and only the hash of the files could be stored in the Blockchain. Of course, a private IPFS network must be deployed to ensure the integrity of these files and deny access to users that are not supposed to retrieve such information. This part of the proposal requires a similar system to be studied that is capable of storing files and at the same time keeping them protected from third parties. Although in such private blockchain network costs do not matter, this could be a better and easier solution for scaling.

Bibliography

[1] *Blockchain representation figure*. https://www.paiementor.com/blockchain-explained-application-payments/.

[2] *Types of Blockchain: Public, Private, or Something in Between*. https://www.foley.com/en/insights/publications/2021/08/types-of-blockchain-public-private-between.

[3] *Consensus Mechanisms comparison table*. https://www.sebaversity.swiss/Lectures/what-is-mining/.

[4] Catherine Mulligan, Jennifer Zhu Scott, Sheila Warren και JP Rangaswami. *Blockchain Beyond the Hype*. The World Economic Forum, 2018.

[5] *Blockchain Technology Defined*. https://builtin.com/blockchain.

[6] *What is blockchain?* https://www.euromoney.com/learning/blockchain-explained/what-is-blockchain.

[7] *CONSENSUS MECHANISMS*. https://ethereum.org/en/developers/docs/consensus-mechanisms/.

[8] *Proof-of-Authority consensus*. https://apla.readthedocs.io/en/latest/concepts/consensus.html/.

[9] *What Is a Smart Contract and How Does it Work?* https://www.bitdegree.org/crypto/tutorials/what-is-a-smart-contract#what-is-a-smart-contract-what-yoursquore-going-to-find-in-this-guide.

[10] Vitalik Buterin. *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*. 2014.

[11] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008.

[12] *INTRO TO ETHEREUM*. https://ethereum.org/en/developers/docs/intro-to-ethereum/.

[13] *A quick introduction to Bitcoin*. https://www.bitcoin.com/get-started/a-quick-introduction-to-bitcoin/.

[14] Starantzis Dimitrios. *Software architectures for implementing decentralized autonomous organizations (DAO) with blockchain technologies*. Diploma Thesis, SoftLab, National Technical University of Athens, 2022.

[15] *R3 Corda*. https://www.r3.com/trust-technology/.

[16] *Quorum*. https://consensys.net/quorum/.

[17] *Ripple*. https://ripple.com/.

[18] *Hyperledger Foundation*. https://www.hyperledger.org/.

[19] *Hyperledger Fabric*. https://www.hyperledger.org/use/fabric.

[20] *MetaMask*. https://en.wikipedia.org/wiki/MetaMask#cite_note-CNET_2018-6.

[21] *Demo dApp*. https://github.com/ChristosHadjichristofi/HelloWorld-eth-private. Author: Christos Hadjichristofi.

[22] *Grades System dApp*. https://github.com/ChristosHadjichristofi/Grades-Blockchain-dApp. Author: Christos Hadjichristofi.

[23] *Grades System dApp NextJS*. https://github.com/ChristosHadjichristofi/Grades-Blockchain-dApp-NextJS. Author: Christos Hadjichristofi.

[24] *Go Ethereum (Geth)*. https://geth.ethereum.org/downloads/.

[25] *Go Language*. https://go.dev/dl/.

[26] *Private Networks*. https://geth.ethereum.org/docs/interface/private-network.

[27] *Puppeth - Ethereum private network manager (secondary repos)*. https://github.com/puppeth.

[28] *Setup your own private Proof-of-Authority Ethereum network with Geth*. https://hackernoon.com/setup-your-own-private-proof-of-authority-ethereum-network-with-geth-9a0a3750cda8.

[29] *How to Create a Private Ethereum Blockchain Network*. https://coinsbench.com/create-a-private-ethereum-blockchain-network-829be72658a5.

[30] *Geth Command-line Options*. https://geth.ethereum.org/docs/interface/command-line-options.

[31] *Geth JavaScript Console*. https://geth.ethereum.org/docs/interface/javascript-console.

[32] *Full list of geth terminal commands*. https://ethereum.stackexchange.com/questions/28703/full-list-of-geth-terminal-commands.

[33] *Explaining the Genesis Block in Ethereum*. https://arvanaghi.com/blog/explaining-the-genesis-block-in-ethereum/.

[34] *Explanation of genesis file*. https://medium.com/singapore-blockchain-dapps/explanation-of-genesis-file-583774a5f523.

[35] *Byzantium Fork*. https://www.investopedia.com/news/what-byzantium-hard-fork-ethereum/.

[36] *A Complete Guide to Building Ethereum dApps: Front-end and Back-end*. https://betterprogramming.pub/a-complete-guide-to-build-ethereum-dapps-front-end-and-back-end-6fa44b66554b.

[37] *Destroy Smart Contracts using selfdestruct*. https://ethereum-blockchain-developer.com/022-pausing-destroying-smart-contracts/04-destroy-smart-contracts/.

[38] *NodeJS*. https://nodejs.dev/.

[39] *MetaMask*. https://metamask.io/.

[40] *NextJS*. https://nextjs.org/.

[41] *ReactJS*. https://reactjs.org/.

[42] *Remix - Ethereum IDE*. https://remix.ethereum.org/.

[43] *TruffleSuite*. https://trufflesuite.com/.

[44] *Truffle Overview*. https://trufflesuite.com/docs/truffle/.

[45] *Truffle Config*. https://trufflesuite.com/docs/truffle/reference/configuration/.

[46] *React Components*. https://www.w3schools.com/react/react_components.asp.

[47] *Diff*. https://en.wikipedia.org/wiki/Diff.

[48] *Higher-Order Components*. https://reactjs.org/docs/higher-order-components.html.

[49] *Contract ABI Specification*. https://docs.soliditylang.org/en/v0.8.13/abi-spec.html.

[50] *NextJS Pages*. https://nextjs.org/docs/basic-features/pages.

[51] *NextJS API Routes*. https://nextjs.org/docs/api-routes/introduction.

[52] *ReactJS Context*. https://reactjs.org/docs/context.html.

[53] *Web3-React*. https://github.com/NoahZinsmeister/web3-react.

[54] *Ethers*. https://github.com/ethers-io/ethers.js/.

[55] *Ethers Contract*. https://docs.ethers.io/v5/api/contract/contract/.

[56] *Switching Networks on Metamask*. https://autofarm.gitbook.io/autofarm-network/how-tos/defi-beginners-guide/switching-networks-on-metamask.

[57] *Faker - NPM*. https://www.npmjs.com/package/faker.

[58] *What is an NFT*. https://www.forbes.com/advisor/investing/cryptocurrency/nft-non-fungible-token/,.

[59] *IPFS*. https://ipfs.io/.

List of Abbreviations

BC	Blockchain
DoS	Denial-of-Service
dApp	Decentralized Application
PoW	Proof of Work
PoS	Proof of Stake
PoA	Proof of Authority
CA	Certification Authority
DLT	Distributed Ledger Technology
LoFi	Low Fidelity
OS	Operating System
CLI	Command Line Interface
JSON	JavaScript Object Notation
CSV	Comma-Seperated Values
API	Application Programming Interface
NPM	Node Package Manager
HOC	Higher Order Component
ABI	Application Binary Interface
EVM	Ethereum Virtual Machine
NFT	Non-Fungible Token
RPC	Remote Procedural Call
UML	Unified Modeling Language
IPFS	InterPlanetary File System

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Πληροφορικής

Υλοποίηση εφαρμογής Blockchain για διαχείριση βαθμολογιών σε ΑΕΙ

Διπλωματικη Εργασια

του

ΧΑΤΖΗΧΡΙΣΤΟΦΗ ΧΡΙΣΤΟΣ

Επιβλέπων: Βασίλειος Βεσκούκης
Καθηγητής

Αθήνα, Ιούνιος 2022

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Πληροφορικής

Υλοποίηση εφαρμογής Blockchain για διαχείριση βαθμολογιών σε ΑΕΙ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΧΑΤΖΗΧΡΙΣΤΟΦΗ ΧΡΙΣΤΟΣ

Επιβλέπων: Βασίλειος Βεσκούκης
Καθηγητής

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 21η Ιουνίου.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Βασίλειος Βεσκούκης	Φωτάκης Δημήτριος	Άρης Παγουρτζής
Καθηγητής	Αναπληρωτής Καθηγητής	Καθηγητής

Αθήνα, Ιούνιος 2022

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Πληροφορικής

Copyright © – All rights reserved. Με την επιφύλαξη παντός δικαιώματος.
Χρίστος Χατζηχριστοφή, 2022.

Απαγορεύεται η αντιγραφή, αποθήκευση και διανομή της παρούσας εργασίας, εξ ολοκλήρου ή τμήματος αυτής, για εμπορικό σκοπό. Επιτρέπεται η ανατύπωση, αποθήκευση και διανομή για σκοπό μη κερδοσκοπικό, εκπαιδευτικής ή ερευνητικής φύσης, υπό την προϋπόθεση να αναφέρεται η πηγή προέλευσης και να διατηρείται το παρόν μήνυμα.

Το περιεχόμενο αυτής της εργασίας δεν απηχεί απαραίτητα τις απόψεις του Τμήματος, του Επιβλέποντα, ή της επιτροπής που την ενέκρινε.

(Υπογραφή)

.....
Χρίστος Χατζηχριστοφή

Ιούνιος

Περίληψη

Το Blockchain είναι ένα κατανεμημένο και αποκεντρωμένο αμετάβλητο δημόσιο μητρώο, στο οποίο έχουν πρόσβαση οι συμμετέχοντας του δικτύου. Λειτουργεί δηλαδή, ως μια κατανεμημένη βάση δεδομένων που μοιράζεται μεταξύ των κόμβων ενός δικτύου. Πιο συγκεκριμένα, αποθηκεύει πληροφορίες σε ψηφιακή μορφή, η οποία δεν μπορεί να επεξεργαστεί. Έτσι, εγγυάται την ακεραιότητα και την ασφάλεια οποιασδήποτε εγγραφής δεδομένων και ως εκ τούτου, είναι σε θέση να λειτουργεί χωρίς την ανάγκη ενός αξιόπιστου τρίτου. Είναι μια πολύ γνωστή τεχνολογία η οποία άνθισε τα τελευταία χρόνια, λόγω της ικανότητας της να διατηρεί τις συναλλαγές με ασφάλεια.

Η παρούσα διπλωματική εργασία στοχεύει την ανάπτυξη ενός ιδιωτικού δικτύου Blockchain, με τα αντίστοιχα έξυπνα συμβόλαια (smart contracts) και τη διεπαφή χρήστη για τη διαχείρηση βαθμών του Πανεπιστημίου. Το πρόβλημα που αντιμετωπίζει η διπλωματική θα αναλυθεί και θα συζητηθεί μια προτεινόμενη λύση. Επιπλεον, θα περιγραφούν βασικές έννοιες και αρχές που αφορούν το Blockchain και κατ'επέκταση τα έξυπνα συμβόλαια, τους αλγόριθμους συναίνεσης (consensus algorithms) αλλά και οι δυσκολίες που συναντήθηκαν κατά την υλοποίηση αυτού του συστήματος.

Τέλος, συνάγεται το συμπέρασμα ότι ένα σύστημα με τέτοιες ικανότητες μπορεί να αναπτυχθεί, να υπάρξει και να συνεργάζεται με όποιοδήποτε άλλο παλαιού τύπου σύστημα (legacy system), και να προσφέρει όλα τα προνόμια της τεχνολογίας Blockchain. Επιπλέον, αναφέρονται ιδέες για περαιτέρω επεκτάσεις του εξεταζόμενου συστήματος, μαζί με τα οφέλη που μπορούν να προσφέρουν.

Λέξεις Κλειδιά

Blockchain, Ethereum, Go Ethereum, Ιδιωτικό Δίκτυο Blockchain, Έξυπνα Συμβόλαια, Ledger, Chaincode, Αποκεντρωμένες Εφαρμογές, Ακεραιότητα, Ασφάλεια

Ευχαριστίες

Μετά την ολοκλήρωση της διπλωματικής μου εργασίας, ολοκληρώνεται το πενταετές ταξίδι μου στη Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου (ΕΜΠ). Αυτό δε θα ήταν δυνατό χωρίς την οικογένεια και τους φίλους μου, που με στήριξαν καθ'όλη τη διάρκεια των προπτυχιακών μου σπουδών.

Θα ήθελα να εκφράσω τις ευχαριστίες μου στον καθηγητή και επιβλέποντα μου κ. Βασίλη Βεσκούκη, για την ευκαιρία που μου δόθηκε να εργαστώ και να αποκτήσω πρακτική εμπειρία πάνω σε αυτό το θέμα, αλλά και για την πολύτιμη καθοδήγηση, τα σχόλια και την άψογη συνεργασία του κατά τη διάρκεια της εκπόνησης της διπλωματικής μου.

Επιπρόσθετα, θα ήθελα να ευχαριστήσω τη βοήθεια που παρείχε ο Υποψήφιος Διδάκτωρ και Ερευνητής Γιάννης Τζαννέτος, ο οποίος με βοήθησε καθ' όλη τη διάρκεια της εκπόνησης της διπλωματικής μου να κατανοήσω το θέμα σε βάθος.

Τέλος, θα ήθελα να εκφράσω τις ιδιαίτερες ευχαριστίες μου σε όλους τους ανθρώπους που συμμετείχαν σε αυτό το ταξίδι στο Εθνικό Μετσόβιο Πολυτεχνείο.

Αθήνα, Ιούνιος 2022

Χρίστος Χατζηχριστοφή

Περιεχόμενα

Περίληψη **1**

Ευχαριστίες **3**

1 Εισαγωγή **13**
1.1 Κίνητρο . . . 14
1.2 Δομή Πτυχιακής εργασίας . . . 15

2 Βασικές έννοιες Blockchain **17**
2.1 Τι είναι ένα Block . . . 17
2.2 Τι είναι ένας Κόμβος . . . 17
2.3 Τι είναι ένας Εξορύκτης (Miner) . . . 17
2.4 Τι είναι το Blockchain . . . 18
2.5 Τύποι Δικτύων Blockchain . . . 18
2.6 Αλγόριθμοι Συναίνεσης . . . 20
2.6.1 Proof of Work (PoW) . . . 20
2.6.2 Proof of Stake (PoS) . . . 20
2.6.3 Proof of Authority (PoA) . . . 20
2.6.4 Σύγκριση μεταξύ Συναινετικών Μηχανισμών . . . 22
2.7 Έξυπνα Συμβόλαια . . . 22
2.8 Πλεονεκτήματα και μειονεκτήματα της χρήσης της τεχνολογίας Blockchain . . . 23

3 Μελέτη περίπτωσης χρήσης **25**
3.1 Επεξήγηση του προβλήματος . . . 25
3.2 Μια προσέγγιση βασισμένη στο Blockchain . . . 27
3.2.1 Ιδιωτικότητα - Κλειστότητα . . . 28
3.2.2 Επικύρωση και Ακεραιότητα . . . 28
3.2.3 Διεπαφή χρήστη . . . 28
3.3 Μια προσέγγιση Blockchain με ιδιωτική άδεια, βασισμένη στο Ethereum . . . 30
3.4 Μοντέλο περίπτωσης χρήσης . . . 32
3.4.1 Γραμματείς . . . 32
3.4.2 Διαχειριστές . . . 32
3.5 UML Διάγραμμα δραστηριότητας ροής εργασίας υψηλού επιπέδου . . . 33
3.6 Wireflow . . . 34
3.7 Σύνδεση και διαφορετικοί χρήστες . . . 35
3.7.1 Περιγραφή . . . 35

3.7.2 Τι είναι το MetaMask . . . 35
3.7.3 Διάγραμμα δραστηριότητας UML . . . 35
3.7.4 Login - Metamask Lo-Fi Wireframe . . . 36
3.7.5 Menu - Διαχειριστή Lo-Fi Wireframe . . . 36
3.7.6 Menu - Χρήστη Lo-Fi Wireframe . . . 37
3.8 Περίπτωση Χρήσης 1: Προσθήκη Βαθμολογιών . . . 38
3.8.1 Περιγραφή . . . 38
3.8.2 Διάγραμμα δραστηριότητας UML - Ο καθηγητής δημιουργεί και διανέμει αρχείο . . . 38
3.8.3 Διάγραμμα δραστηριότητας UML - Προσθήκη Βαθμολογιών . . . 38
3.8.4 Διάγραμμα ακολουθίας UML . . . 39
3.8.5 Βοηθητικό Διάγραμμα ακολουθίας UML . . . 39
3.8.6 Lo-Fi Wireframe . . . 40
3.9 Περίπτωση Χρήσης 2: Εμφάνιση πληροφοριών μαθήματος και επικύρωση . . . 41
3.9.1 Περιγραφή . . . 41
3.9.2 Διάγραμμα δραστηριότητας UML . . . 41
3.9.3 Διάγραμμα ακολουθίας UML . . . 42
3.9.4 Εμφάνιση Μαθημάτων - LoFi Wireframe . . . 42
3.9.5 Εμφάνιση πληροφοριών μαθήματος - LoFi Wireframe . . . 43
3.9.6 Εμφάνιση διαφορών - LoFi Wireframe . . . 43
3.10 Περίπτωση Χρήσης 3: Ψηφίστε για νέους χρήστες . . . 44
3.10.1 Περιγραφή . . . 44
3.10.2 Διάγραμμα δραστηριότητας UML . . . 44
3.10.3 Διάγραμμα ακολουθίας UML . . . 45
3.10.4 Lo-Fi Wireframe . . . 45
3.11 Περίπτωση Χρήσης 4: Έναρξη μιας νέας διαδιασίας ψηφοφορίας . . . 46
3.11.1 Περιγραφή . . . 46
3.11.2 Διάγραμμα δραστηριότητας UML . . . 46
3.11.3 Διάγραμμα ακολουθίας UML . . . 47
3.11.4 Lo-Fi Wireframe . . . 47
3.12 Περίπτωση Χρήσης 5: Ανάκτηση χρηστών του συστήματος . . . 48
3.12.1 Περιγραφή . . . 48
3.12.2 Διάγραμμα δραστηριότητας UML . . . 48
3.12.3 Διάγραμμα ακολουθίας UML . . . 48
3.12.4 Lo-Fi Wireframe . . . 49

4 Επίλογος **51**
4.0.1 Ιδιωτικότητα - Κλειστότητα . . . 51
4.0.2 Ασφάλεια και Ακεραιότητα . . . 51
4.1 Αμεσότητα διαδικασιών . . . 52
4.2 Περαιτέρω επεκτάσεις . . . 52

Βιβλιογραφία **56**

Συντομογραφίες - Αρκτικόλεξα - Ακρωνύμια **57**

Απόδοση ξενόγλωσσων όρων **59**

Κατάλογος Σχημάτων

2.1 Αναπαράσταση ενός Blockchain [1] . . . 18
2.2 Τύποι δικτύων Blockchain [2] . . . 19

3.1 Διαδικασία που ακολουθεί το υπάρχων σύστημα . . . 26
3.2 Προτεινόμενη διαδικασία - Συνύπαρξη του υπάρχοντος συστήματος με το σύστημα Blockchain . . . 27
3.3 UML Activity Diagram - Διαδικασία επικύρωσης . . . 28
3.4 Αναπαράσταση κόμβων και χρηστών του συστήματος . . . 31
3.5 UML Use Case Διάγραμμα . . . 32
3.6 UML High-level work flow Activity Diagram . . . 33
3.7 Wireflow diagram . . . 34
3.8 Διάγραμμα δραστηριότητας UML: Διαδικασία login . . . 35
3.9 Login Wireframe . . . 36
3.10 Menu - Διαχειριστή Lo-Fi Wireframe . . . 36
3.11 Menu - Χρήστη Lo-Fi Wireframe . . . 37
3.12 Διάγραμμα δραστηριότητας UML - Ο καθηγητής δημιουργεί και διανέμει αρχείο 38
3.13 Διάγραμμα δραστηριότητας UML - Προσθήκη Βαθμολογιών . . . 38
3.14 Διάγραμμα ακολουθίας UML: Προσθήκη Βαθμολογιών . . . 39
3.15 Διάγραμμα ακολουθίας UML: MetaMask ρωτά τον χρήστη για αποδοχή/απόρριψη συναλλαγής . . . 39
3.16 Προσθήκη Βαθμολογιών Wireframe . . . 40
3.17 Διάγραμμα δραστηριότητας UML: Εμφάνιση πληροφοριών μαθήματος και επικύρωση . . . 41
3.18 Διάγραμμα ακολουθίας UML: Εμφάνιση πληροφοριών μαθήματος και επικύρωση . . . 42
3.19 Εμφάνιση Μαθημάτων - LoFi Wireframe . . . 42
3.20 Εμφάνιση πληροφοριών μαθήματος - LoFi Wireframe . . . 43
3.21 Εμφάνιση διαφορών - LoFi Wireframe . . . 43
3.22 Διάγραμμα δραστηριότητας UML: Ψήφος σε νέους χρήστες . . . 44
3.23 Διάγραμμα ακολουθίας UML: Ψηφός σε νέους χρήστες . . . 45
3.24 Ψηφίστε υπέρ ή κατά - Lo-Fi Wireframe . . . 45
3.25 Διάγραμμα δραστηριότητας UML: Έναρξη νέας ψηφοφορίας . . . 46
3.26 Διάγραμμα ακολουθίας UML: Έναρξη νέας ψηφοφορίας . . . 47
3.27 Έναρξη νέας ψηφοφορίας - Lo-Fi Wireframe . . . 47
3.28 Διάγραμμα δραστηριότητας UML: Ανάκτηση χρηστών του συστήματος . . . 48

3.29 Διάγραμμα ακολουθίας UML: Ανάκτηση χρηστών του συστήματος 48
3.30 Ανάκτηση χρηστών του συστήματος - Lo-Fi Wireframe 49

4.1 Διαδικασία προσθήκης βαθμολογιών και επικύρωσης 52
4.2 Φοιτητές που ζητούν έγγραφο απο το υπάρχων σύστημα vs το blockchain σύστημα 53

Κατάλογος Πινάκων

2.1 Σύγκριση μεταξύ Συναινετικών Μηχανισμών [3] 22
2.2 Πλεονεκτήματα και Μειονεκτήματα χρήσης της τεχνολογίας Blockchain . . . 23

Κεφάλαιο 1

Εισαγωγή

Τα τελευταία χρόνια το, Blockchain γίνεται όλο και πιο δημοφιλές και έχει κερδίσει μεγάλη αναγνωρισιμότητα. Το Blockchain έχει προταθεί ως λύση σε ένα τεράστιο αριθμό και ποικιλία προβλημάτων, όπως προβλήματα διαχείρισης της εφοδιαστικής αλυσίδας, κλοπή ταυτότητας, ψηφιακά πνευματικά δικαιώματα και πειρατεία, συγκέντρωση κεφαλαίων και ούτω καθεξής. Περιττό να πούμε ότι τα κρυπτονομίσματα έχουν συμβάλει σε τεράστιο βαθμό στην ανάπτυξη της τεχνολογίας Blockchain, είναι σαφές ωστόσο ότι το Blockchain δεν αφορά μόνο τα κρυπτονομίσματα. Αντίθετα, τα κρυπτονομίσματα βασίζονται σε μόνο ενός είδος Blockchains, απο πολλά που μπορούν να συνεισφέρουν σε εφαρμογές σε πολλούς τομείς. Ο λόγος που αυτή η τεχνολογία μπορεί να είναι τόσο γενική και να προσφέρει λύσεις σε διάφορα είδη προβλημάτων οφείλεται στην ακεραιότητα και την ασφάλεια που μπορεί να προσφέρει, η οποία γίνεται με την εξάλειψη του κυρίαρχου ¨έμπιστου τρίτου φορέα¨, στο οποίο οι περισσότερες τρέχουσες προσεγγίσεις για την ακεραιότητα και την ασφάλεια των δεδομένων βασίζονται. Καθώς οι ανθρώποι είδαν τις δυνατότητες αυτής της τεχνολογίας και το γεγονός ότι μπορεί να εφαρμοστεί σε τόσες πολλές περιπτώσεις χρήσης, ο ενθουσιασμός σχετικά με αυτήν συνέχισε να μεγαλώνει. Η δημοσιότητα που άρχισε να λαμβάνει η τεχνολογία αυτή, οδήγησε τους ανθρώπους να αρχίσουν να πειραματίζονται με λύσεις για τα προβλήματα που θέλουν να λύσουν με την χρήση του Blockchain χωρίς να λαμβάνουν υπόψη ότι μια καλύτερη και πιο βιώσιμη λύση θα μπορούσε να αναπτυχθεί χωρίς τη χρήση αυτής της συγκεκριμένης τεχνολογίας. Έτσι, κατέληξαν να σπαταλούν πόρους, να σπαταλούν μεγάλο μέρος χρημάτων και εν τέλει, να δημιουργείται ένας αρνητικός αντίκτυπος πάνω στην τεχνολογία αυτή.

Απο την άλλη, όταν η τεχνολογία Blockchain χρησιμοποιείται κάτω απο τις κατάλληλες συνθήκες και σε κατάλληλες περιπτώσεις χρήσης, μπορεί να επιφέρει θετικά αποτέλεσματα. Πιο συγκεκριμένα, όπως συμβαίνει με όλες τις τεχνολογίες, το Blockchain πρέπει να χρησιμοποιηθεί κατάλληλα, ώστε να αντιμετωπίζει το πρόβλημα στο οποίο προσπαθεί να εφαρμοστεί όσο το δυνατόν πιο αποτελεσματικά. Διαφορετικοί τύποι δικτύων Blockchain μπορούν να χρησιμοποιηθούν για την επίλυση διαφορετικών τύπων προβλημάτων. Αυτή η διπλωματική εργασία εστιάζει στον σχεδιασμό και την εφαρμογή ενός ιδιωτικού δικτύου με δικαιώματα σε ένα τομέα εφαρμογών που απαιτούνται υψηλά επίπεδα ιδιωτικότητας και ακεραιότητας δεδομένων, δηλαδή τη διαχείριση των βαθμολογιών των πανεπιστημιακών εξετάσεων. Ως εκ τούτου, το Blockchain είναι κατάλληλο για το πρόβλημα αυτό. Μια πιο λεπτομερής συζήτηση σχετικά με τους τύπους των δικτύων Blockchain είναι διαθέσιμη στην

ενότητα 2.3.

1.1 Κίνητρο

Η διαχείριση των βαθμών στα Πανεπιστήμια είναι ένα απο τα σηματνικότερα καθήκοντα που ανατίθενται στους Γραμματείς. Εκτός απο τις προφανείς λειτουργικές απαιτήσεις, όπως η διαχείριση εγγραφών σε μαθήματα, η αποθήκευση βαθμών σε μια βάση δεδομένων, η αναφορά κ.λπ., είναι οι μη λειτουργικές απαιτήσεις που συνήθως οδηγούν την αρχιτεκτονική και τον σχεδιασμό τέτοιων εφαρμογών. Μεταξύ αυτών, η ¨ασφάλεια¨ είναι η κυρίαρχη απαίτηση που μπορεί να αναλυθεί περαιτέρω ως εξής:

- ακεραιότητα: οι βαθμοί θα πρέπει να ενημερώνονται μόνο απο καθηγητές που τους παράγουν στο τέλος της παράδοσης των μαθημάτων. Οι εφαρμογές διαχείρισης βαθμών θα πρέπει να είναι δομημένες με τέτοιο τρόπο ώστε καμιά άλλη οντότητα, ανεξάρτητα απο τον ρόλο τους στη διαχείριση του πληροφοριακού συστήματος, να μην μπορεί να τροποποιήσει τα δεδομένα των βαθμολογιών.

- ασφάλεια: οι βαθμοί θα πρέπει να είναι ορατοί μόνο σε συγκεκριμένες οντότητες, δηλαδή στον μαθητή και το προσωπικό της γραμματείας αυστηρά για διαχειριστικούς σκοπούς.

- δυνατότητα ανίχνευσης: οποιαδήποτε έγκυρη τροποποίηση σε βαθμούς, όπως η διόρθωση σφαλμάτων, θα πρέπει να είναι ανιχνεύσιμη, πράγμα που σημαίνει ότι θα πρέπει να τηρείται αναλυτικά ένα ιστορικό των αλλαγών, με τις ίδιες απαιτήσεις ακεραιότητας και ασφάλειας όπως με την πρώτη ενημέρωση βαθμολογιών σε κάποιο μάθημα.

- πολλαπλά αντίγραφα κρίσιμων δεδομένων θα πρέπει να διατηρούνται και οι εξουσιοδοτημένες ενημερώσεις θα πρέπει να διαδίδονται σε πραγματικό χρόνο στους φορείς που κρατάνε τα αντίγραφα. Η πρόσβαση στο ψηφιακά στοιχεία που υλοποιεί αυτά τα αντίγραφα δε θα πρέπει να επιτρέπει την αποκρυπτογράφηση και την τροποποίηση αυτών των δεδομένων.

- θα πρέπει να είναι δυνατή η αυτόματη επαλήθευση οποιουδήποτε ψηφιακού στοιχείου που περιέχει τους αποκρυπτογραφημένους βαθμούς, έναντι του αξιόπιστου κατανεμημένου συστήματος διαχείρησης και αποθήκευσης βαθμολογιών. Η επαλήθευση αυτή θα πρέπει να είναι δυνατή μόνο απο εξουσιοδοτημένο προσωπικό.

Λαμβάνοντας υπόψη ότι τα προϋπάρχοντα συστήματα διαχείρισης βαθμών (legacy systems) είναι ήδη σε λειτουργία, οι πάραπανω λειτουργικές απαιτήσεις ασφαλείας θα πρέπει να μπορούν να ικανοποιούνται χωρίς να διακόπτεται η λειτουργία αυτών. Θα πρέπει να είναι δυνατή η διαφανής, παράλληλη λειτουργία μιας υπηρεσίας που εφαρμόζει το είδος των απαιτήσεων ασφαλείας που προαναφέρθηκαν. Ένα τέτοιο σύστημα θα παρείχε σημαντικές βελτιώσεις στο επίπεδο εμπιστοσύνης όσο αφορά τους βαθμούς. Θα ελαχιστοποιούσε επίσης τον φόρτο εργασίας που απαιτείται για την επαλήθευση των βαθμών με παραδοσιακές μεθόδους.

Υποστηρίζουμε ότι οι τεχνολογίες Blockchain μπορούν να χρησιμοποιηθούν για την υλοποίηση μιας υπηρεσίας που λειτουργεί παράλληλα με οποιοδήποτε προϋπάρχων σύστημα διαχείρησης βαθμολογιών και ενισχύει ταυτόχρονα την ασφάλεια του, εφαρμόζοντας τις απαιτήσεις που συζητήθηκαν πιο πάνω. Αυτό ηταν το κύριο κίνητρο πίσω απο την παρούσα διπλωματική εργασία.

1.2 Δομή Πτυχιακής εργασίας

Στο Κεφάλαιο 2 παρουσιάζονται οι βασικές αρχές ενός δικτύου Blockchain καθώς και μια εισαγωγή σε έννοιες όπως συναλλαγές, έξυπνα συμβόλαια, αλγόριθμοι συναίνεσης.

Στο Κεφάλαιο 3 θα εξαχθούν οι απαιτήσεις των χρηστών από μια εφαρμογή διαχείρισης πανεπιστημιακών βαθμών εξετάζοντας την τυπική ροή εργασιών της τρέχουσας διαδικασίας στο ΕΜΠ. Θα οριστούν περιπτώσεις χρήσης για την αντιμετώπιση αυτής της συγκεκριμένης περίπτωσης προσθέτοντας ένα Blockchain ¨στο πλάι¨ της τρέχουσας πρακτικής. Αυτό θα γίνει ακολουθώντας μια πειθαρχημένη μεθοδολογία μηχανικής λογισμικού και τεκμηρίωση, συμπεριλαμβανομένης της επίκλησης απαιτήσεων, αρχιτεκτονικής, σχεδίασης και υλοποίησης.

Στο Κεφάλαιο 4 θα συζητηθεί η αρχιτεκτονική και ο σχεδιασμός μιας κατανεμημένης εφαρμογής (dApp), μαζί με τις προκλήσεις που αντιμετωπίστηκαν κατά την υλοποίηση. Επίσης, θα περιγραφούν συγκρίσεις για διαφορετικές εναλλακτικές προσεγγίσεις υλοποίησης που έχουν εξεταστεί.

Στο Κεφάλαιο 5 θα περιγραφεί η υλοποίηση από την τεχνική πλευρά. Πώς έχει δημιουργηθεί αυτό το ιδιωτικό δίκτυο βλοσκσηαιν, ποια πλατφόρμα χρησιμοποιήθηκε και γιατί, πώς εφαρμόστηκαν αυτές οι απαιτήσεις και ποιες τεχνολογίες χρησιμοποιήθηκαν για την υλοποίηση του front-end.

Στο Κεφάλαιο 6 θα παρουσιαστεί μια προσομοίωση αυτής της αποκεντρωμένης εφαρμογής μέσω στιγμιότυπων οθόνης διαφορετικών χρηστών που συνδέονται στο dApp και αλληλεπιδρούν με το Blockchain είτε με ανάκτηση είτε με αποθήκευση πληροφοριών, ανάλογα με τις περιπτώσεις χρήσης του συστήματος.

Στον Επίλογο θα παρουσιαστούν προτάσεις για την ενίσχυση του συστήματος. Πώς μπορεί να επεκταθεί αυτό το σύστημα αλλά και άλλα προβλήματα που θα μπορούσε ενδεχομένως να λύσει αποτελεσματικά και κομψά. Τέλος, θα εξαχθούν συμπεράσματα της παρούσας διπλωματικής εργασίας.

Κεφάλαιο 2

Βασικές έννοιες Blockchain

Αυτό το κεφάλαιο περιγράφει μερικές πολύ βασικές αρχές του Blockchain τις οποίες ο αναγνώστης θα πρέπει να έχει υπόψη του. Ο σκοπός αυτού του κεφαλαίου δεν είναι να αποκτήσει εκτενή γνώση για το Blockchain και τις σχετικές έννοιες.

2.1 Τι είναι ένα Block

Ένα μπλοκ είναι το κύριο στοιχείο ενός Blockchain και κάθε μπλοκ μπορεί να αποσυντεθεί σε τρεις συνιστώσες [4]:

1. Τα δεδομένα που θα αποθηκευτούν
2. Το nonce, που είναι ένας αριθμός 32 bit. Δημιουργείται τυχαία όταν δημιουργείται ένα νέο μπλοκ, το οποίο στη συνέχεια δημιουργεί την κεφαλίδα κατακερματισμού του μπλοκ.
3. Το hash, ένας αριθμός 256-bit που συνδέεται με το nonce

2.2 Τι είναι ένας Κόμβος

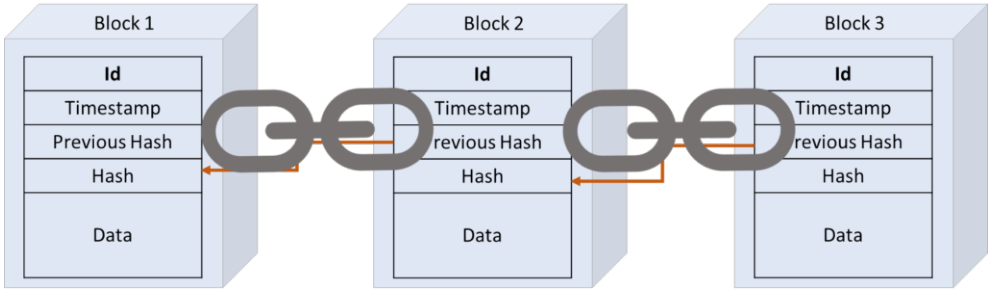
Ένας κόμβος είναι ένας συμμετέχων στο δίκτυο, το οποίο έχει το αντίγραφο του καθολικού. Κάθε ηλεκτρονική συσκευή που μπορεί να διατηρήσει ένα αντίγραφο του δημόσιου καθολικού και να διατηρήσει τη λειτουργία του δικτύου μπορεί να είναι ένας κόμβος ενός δικτύου Blockchain. Ο αριθμός των συμμετεχόντων (κόμβων) σε ένα δίκτυο Blockchain είναι πολύ σημαντικός. Τα δίκτυα Blockchain με πολλούς συμμετέχοντες (κόμβους) θεωρούνται συχνά πιο ασφαλή, καθώς η ασφάλεια του δικτύου προέρχεται από την αποκέντρωση. Έτσι, περισσότεροι κόμβοι προσφέρουν περισσότερη ακεραιότητα [4].

2.3 Τι είναι ένας Εξορύκτης (Miner)

Ένας κόμβος εξόρυξης είναι ένας συμμετέχων του δικτύου που θυσιάζει την υπολογιστική ισχύ για να λύσει το παζλ Proof-of-Work για να διατηρήσει την ακεραιότητα του δικτύου. Αυτοί οι κόμβοι βοηθούν στην παροχή ασφάλειας καθώς και στην απαιτούμενη αποκέντρωση του δικτύου.

2.4 Τι είναι το Blockchain

Το Blockchain είναι ένα κατανεμημένο σύστημα που μπορεί να αποθηκεύσει πληροφορίες με συγκεκριμένο τρόπο που καθιστά αδύνατη την αλλαγή ή την εξαπάτηση. Είναι ένα ψηφιακό καθολικό συναλλαγών, το οποίο αντιγράφεται και μοιράζεται σε όλο το δίκτυο. Κάθε φορά που πραγματοποιείται μια νέα συναλλαγή στο δίκτυο, μια νέα εγγραφή προστίθεται στο καθολικό κάθε συμμετέχοντα. Οι συναλλαγές καταγράφονται στο σύστημα με αμετάβλητη κρυπτογραφική υπογραφή (hash). Αυτό εγγυάται το αμετάβλητο του δημόσιου καθολικού σε όλο το δίκτυο, επειδή εάν ένα μπλοκ έχει αλλάξει, θα ήταν αμέσως εμφανές. Επομένως, εάν κάποιος θέλει να αλλάξει το καθολικό, κάθε μπλοκ της αλυσίδας σε κάθε συμμετέχοντα του δικτύου πρέπει να αλλάξει, κάτι που είναι αδύνατο, επομένως η ακεραιότητα των δεδομένων που μοιράζονται είναι εγγυημένη [5].

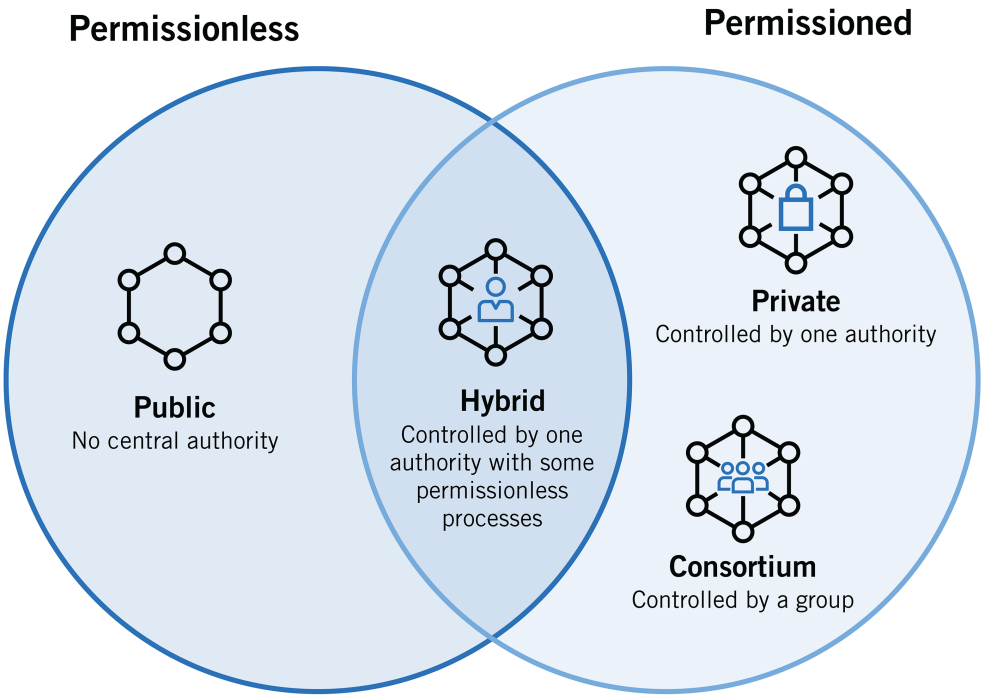


Σχήμα 2.1: *Αναπαράσταση ενός Blockchain [1]*

2.5 Τύποι Δικτύων Blockchain

Ο πρώτος τύπος δικτύου Blockchain είναι το δίκτυο χωρίς άδεια, το οποίο επιτρέπει την πρόσβαση σε οποιονδήποτε χρήστη να ενταχθεί ψευδο-ανώνυμα στο δίκτυο και να γίνει κόμβος, χωρίς κανέναν περιορισμό. Τα δίκτυα χωρίς άδεια ονομάζονται επίσης δημόσια δίκτυα και, όπως αναφέρθηκε παραπάνω, ο καθένας μπορεί να εγγραφεί. Αυτός ο συγκεκριμένος τύπος δικτύου είναι πλήρως αποκεντρωμένος λόγω του γεγονότος ότι ο καθένας μπορεί να γίνει κόμβος του δικτύου. Όλοι οι κόμβοι ενός δημόσιου δικτύου BC έχουν ίσα δικαιώματα πρόσβασης στο Blockchain, δημιουργία και επικύρωση νέων μπλοκ δεδομένων. Αυτό το συγκεκριμένο είδος δικτύου χρησιμοποιείται κυρίως για την ανταλλαγή και την εξόρυξη κρυπτονομισμάτων. Δημοφιλή δημόσια δίκτυα Blockchain είναι το Bitcoin και το Ethereum. Σε αυτά τα δίκτυα, οι κόμβοι εξορύσσουν κρυπτονομίσματα δημιουργώντας μπλοκ για τις συναλλαγές που ζητούνται στο δίκτυο λύνοντας κρυπτογραφικές εξισώσεις. Σε αντάλλαγμα για τους πόρους που προσφέρουν στο δίκτυο, οι κόμβοι εξόρυξης κερδίζουν ένα μικρό ποσό κρυπτονομισμάτων. Ο δεύτερος τύπος δικτύων, είναι τα δίκτυα με άδεια permissioned, τα οποία περιορίζουν την πρόσβαση στο δίκτυο σε ορισμένους κόμβους. Σε αυτόν τον τύπο δικτύου μπορεί να υπάρχουν περιορισμοί δικαιωμάτων. Αυτά τα ιδιωτικά δίκτυα Blockchain ελέγχονται από έναν μόνο οργανισμό που καθορίζει ποιος μπορεί να είναι ένας κόμβος. Τα

ιδιωτικά δίκτυα BC είναι εν μέρει αποκεντρωμένα επειδή η πρόσβαση του κοινού σε αυτά είναι περιορισμένη. Παραδείγματα ιδιωτικών δικτύων BC είναι το Ripple και το Hyperledger. Λόγω των μειονεκτημάτων των παραπάνω δικτύων, αναπτύχθηκαν κοινοπραξίες και υβριδικά Blockchains. Τα δίκτυα κοινοπραξίας είναι επιτρεπόμενα δίκτυα Blockchain που διέπονται από μια ομάδα οργανισμών. Αυτά τα δίκτυα είναι πιο αποκεντρωμένα από τα ιδιωτικά Blockchains που έχουν ως αποτέλεσμα υψηλότερα επίπεδα ασφάλειας. Αντίθετα, αυτοί οι τύποι δικτύων μπορεί να είναι απαιτητικός για τη δημιουργία τους, καθώς απαιτεί συνεργασία μεταξύ πολλών οργανισμών. Τα υβριδικά δίκτυα Blockchain ελέγχονται από έναν μόνο οργανισμό, αλλά με επιτήρηση του δημόσιου Blockchain που απαιτείται για την εκτέλεση ορισμένων επικυρώσεων συναλλαγών. Ένα παράδειγμα υβριδικών Blockchains είναι το IBM Food Trust το οποίο αναπτύχθηκε για να βελτιώσει την αποτελεσματικότητα σε ολόκληρη την αλυσίδα εφοδιασμού τροφίμων [2].



Σχήμα 2.2: *Τύποι δικτύων Blockchain [2]*

Συνοψίζοντας, τα δίκτυα Blockchain χωρίς άδεια μπορεί να είναι πιο ασφαλή από τα επιτρεπόμενα δίκτυα Blockchain, επειδή στα δίκτυα χωρίς άδεια ο αριθμός των κόμβων είναι πολύ μεγαλύτερος από ό,τι σε ένα Blockchain με άδεια, επομένως θα ήταν πιο δύσκολο για τους κακόβουλους χρήστες να δημιουργήσουν κάποιο πρόβλημα στο δίκτυο. Ωστόσο, σε Blockchain χωρίς άδεια, ο χρόνος διαδικασίας συναλλαγών τείνει να είναι μεγαλύτερος από ό,τι σε ένα δίκτυο Blockchain με άδεια, καθώς υπάρχουν περισσότεροι κόμβοι επικύρωσης. Και οι δύο τύποι δικτύων Blockchain έχουν τα πλεονεκτήματα και τα μειονεκτήματά τους, αλλά και οι δύο είναι εδώ για να προσφέρουν την ίδια ιδέα, η οποία είναι η ακεραιότητα του δημόσιου καθολικού που είναι κοινόχρηστο μεταξύ του δικτύου.

2.6 Αλγόριθμοι Συναίνεσης

Οι αλγόριθμοι συναίνεσης (γνωστοί και ως μηχανισμοί συναίνεσης ή πρωτόκολλα συναίνεσης) παρέχουν έναν ασφαλή τρόπο για τους υπολογιστές που αποτελούν μέρος ενός δικτύου (κατανεμημένα συστήματα) να συνεργάζονται. Αυτού του είδους οι μηχανισμοί έχουν χρησιμοποιηθεί εδώ και δεκαετίες για την επίτευξη συναίνεσης σε εταιρικές υποδομές, όπως οι κόμβοι βάσεων δεδομένων και οι διακομιστές. Τα τελευταία χρόνια, νέοι μηχανισμοί συναίνεσης έχουν δημιουργηθεί και εφαρμοστεί στα Blockchains για να εξυπηρετούν κρυπτοοικονομικά συστήματα και να συμφωνούν για την κατάσταση του δικτύου.

Θεωρητικά, ένας κακόβουλος χρήστης (εισβολέας) μπορεί να θέσει σε κίνδυνο τη συναίνεση ελέγχοντας το 51% του δικτύου, αλλά οι μηχανισμοί συναίνεσης έχουν σχεδιαστεί με τρόπο που αυτή η επίθεση είναι πρακτικά αδύνατη. Υπάρχουν διάφορα είδη μηχανισμών συναίνεσης που επινοούνται για χρήση σε κρυπτοοικονομικά συστήματα [6]. Μερικοί τυπικοί τέτοιοι μηχανισμοί συζητούνται στη συνέχεια.

2.6.1 Proof of Work (PoW)

Το Proof of Work γίνεται από τους miners, οι οποίοι ανταγωνίζονται για τη δημιουργία νέων μπλοκ γεμάτα με επεξεργασμένες συναλλαγές. Υπάρχει ένας αγώνας για το ποιος θα λύσει γρηγορότερα ένα μαθηματικό παζλ. Αυτό το παζλ είναι ο μηχανισμός συναίνεσης για την απόδειξη της εργασίας και παράγει την κρυπτογραφική σύνδεση μεταξύ του τρέχοντος μπλοκ και του προηγούμενου μπλοκ. Ο νικητής κερδίζει κάποιο κρυπτονόμισμα για την πολύτιμη υπολογιστική ισχύ που προσφέρθηκε στο δίκτυο και το μπλοκ που εξορύχθηκε πρόσφατα μοιράζεται σε όλους τους συμμετέχοντες. Ο μόνος τρόπος για να τεθεί σε κίνδυνο η ασφάλεια σε ένα δίκτυο που χρησιμοποιεί αυτόν τον αλγόριθμο συναίνεσης είναι κάποια οντότητα να κατέχει το 51% της υπολογιστικής ισχύος του δικτύου, κάτι που θα οδηγούσε σε τεράστιες επενδύσεις σε εξοπλισμό [6].

2.6.2 Proof of Stake (PoS)

Ο αλγόριθμος Proof of Stake γίνεται από επικυρωτές (validators) που έχουν ποντάρει κρυπτονομίσματα για να συμμετάσχουν στο σύστημα. Για να δημιουργηθεί ένα νέο μπλοκ, επιλέγεται ένας τυχαίος κόμβος επικύρωσης. Όταν δημιουργείται ένα νέο μπλοκ, μοιράζεται μεταξύ του δικτύου και ο επιλεγμένος κόμβος επικύρωσης κερδίζει ανταμοιβές. Αυτός ο αλγόριθμος συναίνεσης δεν χρειάζεται βαριά υπολογιστική εργασία όπως ο μηχανισμός Proof of Work. Το μόνο που χρειάζεται είναι να ποντάρονται τα ψηφιακά νόμισμα στο δίκτυο. Ο μόνος τρόπος για να τεθεί σε κίνδυνο η ασφάλεια σε ένα δίκτυο είναι κάποια οντότητα να κατέχει το 51% του συνολικού μεριδίου κρυπτονομίσματος, το οποίο θα ήταν ένα μεγάλο χρηματικό ποσό [6].

2.6.3 Proof of Authority (PoA)

Σε ένα επιτρεπόμενο δίκτυο Blockchain, όλοι οι κόμβοι που ανήκουν στο δίκτυο είναι εξουσιοδοτημένοι, γεγονός που επιτρέπει τη χρήση μηχανισμών συναίνεσης που παρέχουν υψηλό ποσοστό συναλλαγών και άλλα οφέλη. Το Proof of Authority είναι ένα από αυτούς

και είναι ένας αλγόριθμος που παρέχει υψηλή απόδοση και ανοχή σφαλμάτων μέσω ενός μηχανισμού συναίνεσης που βασίζεται στην ταυτότητα ως διακύβευμα. Έτσι, στο PoA μόνο οι κόμβοι που έχουν αποδείξει την ταυτότητά τους έχουν δικαιώματα για τη δημιουργία νέων μπλοκ και ο μόνος τρόπος για να αποκτηθεί αυτό το δικαίωμα είναι να περάσουν απο τον προκαταρκτικό έλεγχο ταυτότητας. Αυτός ο αλγόριθμος συναίνεσης καθιστά ακόμη πιο δύσκολη την επίθεση 51%, καθώς ο εισβολέας πρέπει να αποκτήσει έλεγχο στο 51% των κόμβων του δικτύου Blockchain. Επιπλέον, ο μηχανισμός PoA μπορεί να υπερασπιστεί με επιτυχία μια επίθεση DoS, επειδή όλοι οι συμμετέχοντες που περιλαμβάνονται στο δίκτυο είναι προεπιλεγμένοι και έτσι μπορεί να γίνει μια συγκεκριμένη επιλογή. Για παράδειγμα, μόνο κόμβοι που μπορούν να αντέξουν μια επίθεση DoS θα μπορούσαν να προστεθούν ως συμμετέχοντες στο δίκτυο. Επίσης, ακόμα κι αν ένας κόμβος δικτύου δεν είναι διαθέσιμος για μια συγκεκριμένη χρονική περίοδο, μπορεί να αφαιρεθεί προσωρινά από κόμβος επικύρωσης [7].

2.6.4 Σύγκριση μεταξύ Συναινετικών Μηχανισμών

Παρακάτω, οι τρεις μηχανισμοί συναίνεσης που συζητήθηκαν συγκρίνονται μεταξύ της κύριας ιδέας τους, της κατανάλωσης ενέργειας και του επιπέδου συγκέντρωσης.

Πίνακας 2.1: *Σύγκριση μεταξύ Συναινετικών Μηχανισμών [3]*

Αλγόριθμοι Συναίνεσης	**Κύρια Ιδέα**	**Κατανάλωση Ενέργειας**	**Επίπεδο Κεντροποίησης**
Proof-of-Work (PoW)	Η υπολογιστική ισχύς καθορίζει την πιθανότητα προσθήκης νέου μπλοκ	Υψηλή	Χαμηλό
Proof-Of-Stake (PoS)	Τα κρυπτονομίσματα που ποντάρονται καθορίζουν την πιθανότητα προσθήκης νέου μπλοκ	Χαμηλή	Μέτριο
Proof-Of-Authority (PoA)	Μόνο ορισμένοι εξουσιοδοτημένοι κόμβοι έχουν τη δυνατότητα να προσθέσουν ένα νέο μπλοκ	Χαμηλή	Υψηλό

2.7 Έξυπνα Συμβόλαια

Ένα έξυπνο συμβόλαιο είναι μια συμφωνία μεταξύ δύο μερών με τη μορφή κωδικού υπολογιστή. Τα έξυπνα συμβόλαια αποθηκεύονται στο Blockchain, επομένως αποτελούν μέρος του δημόσιου καθολικού και δεν μπορούν να αλλάξουν. Τα έξυπνα συμβόλαια τρέχουν στο Blockchain και οι συναλλαγές που συμβαίνουν υποβάλλονται σε επεξεργασία από αυτό, πράγμα που σημαίνει ότι μπορούν να αποσταλούν αυτόματα χωρίς κάποιο έμπιστο τρίτο φορέα. Οι συναλλαγές πραγματοποιούνται μόνο όταν πληρούνται συγκεκριμένες προϋποθέσεις [8].

Τα έξυπνα συμβόλαια εμφανίστηκαν για πρώτη φορά στο κύριο δίκτυο του Ethereum Blockchain. Πολλά άλλα δίκτυα Blockchain τα υιοθέτησαν και ανάλογα με την επιλογή δικτύου, τα έξυπνα συμβόλαια μπορούν να γραφτούν σε διαφορετικές γλώσσες. Για παράδειγμα, στα δίκτυα blockchain Ethereum Solidity, η γλώσσα προγραμματισμού Σολιδιτψ έχει αναπτυχθεί για τη σύνταξη έξυπνων συμβολαίων. Στο Hyperledger Javascript, Go, Java και Solidity.

Επιπλέον, τα έξυπνα συμβόλαια μπορούν να προσφέρουν ταχύτητα και αποτελεσματικότητα, καθώς όταν πληρούνται οι προϋποθέσεις, η σύμβαση θα εκτελεστεί άμεσα. Επιπλέον, προσφέρουν εμπιστοσύνη, διαφάνεια και ασφάλεια λόγω του ότι δεν εμπλέκεται τρίτος και η όλη διαδικασία λαμβάνει χώρα στο Blockchain.

2.8 Πλεονεκτήματα και μειονεκτήματα της χρήσης της τεχνολογίας Blockchain

Η τεχνολογία Blockchain έχει πάρα πολλές δυνατότητες χρήσης σε προβλήματα του πραγματικού κόσμου παρά την πολυπλοκότητα και τη δυσκολία κατανόησης της. Φυσικά, η πρόσβαση σε λεπτομέρειες σχετικά με την τεχνολογία Blockchain, καθώς και εξαντλητικές πληροφορίες σχετικά με τα πλεονεκτήματα και τα μειονεκτήματα αυτής της τεχνολογίας είναι εκτός πεδίου εφαρμογής στο πλαίσιο αυτής της διπλωματικής εργασίας. Περισσότερες πληροφορίες σχετικά με την τεχνολογία Blockchain μπορούν να βρεθούν απο [9] [10] [11] [12]. Ακολουθεί ένας πίνακας με ορισμένα πλεονεκτήματα και μειονεκτήματα αυτής της τεχνολογίας.

Πλεονεκτήματα	**Μειονεκτήματα**
Πλήρες διαφάνεια	Περιορισμοί αποθήκευσης δεδομένων
Πιο δύσκολο να παραβιαστούν τα δεδομένα λόγω της αποκεντροποιημένης φύσης και αμεταβλητότητας της τεχνολογίας	Περιορισμένος αριθμός συναλλαγών ανά δευτερόλεπτο
Βελτίωση ακρίβειας λόγω αφαίρεσης της ανθρώπινης συμμετοχής στην επαλήθευση	Κόστος τεχνολογίας
Αποτελεσματικές ασφαλείς και ιδιωτικές συναλλαγές	
Σταθερότητα	
Μείωση κόστους καταργώντας την επαλήθευση έμπιστου τρίτου	
Προσφέρει αυτοματοποίηση - Έξυπνα συμβόλαια	
Ευκολία ανιχνευσιμότητας αλλαγών	

Πίνακας 2.2: *Πλεονεκτήματα και Μειονεκτήματα χρήσης της τεχνολογίας Blockchain*

Κεφάλαιο 3

Μελέτη περίπτωσης χρήσης

3.1 Επεξήγηση του προβλήματος

Η παρούσα διαδικασία που διαχειρίζεται τους βαθμούς, την οποία ακολουθεί κάθε σχολή του πανεπιστημίου είναι αρκετά κοπιαστική και επιρρεπής σε λάθη. Πιο συγκεκριμένα, η διαδικασία που ακολουθεί ένας γραμματέας για να προσθέσει τελικά βαθμούς στο παρόν σύστημα παλαιού τύπου μπορεί να περιγραφεί με τα ακόλουθα βήματα:

Έναρξη εγγραφών ανάλογα με την ακαδημαϊκή περίοδο

Οι φοιτητές εγγράφονται σε περίοδο και μαθήματα

Μόλις κλείσει το σύστημα εγγραφής και κανένας σπουδαστής δεν παραλείψει να εγγραφεί, ένα άτομο από το Κέντρο Υπολογιστών επισκέπτεται τον γραμματέα για να ολοκληρώσει την εγγραφή στο σύστημα

- Σε περίπτωση τυχόν παραλείψεων κατά την εγγραφή αναφέρονται στη γενική συνέλευση του σχολείου
- Η γενική συνέλευση του σχολείου αποφασίζει για τις παραλείψεις
- Ο γραμματέας εισάγει χειροκίνητα εγγραφές που έχουν παραλειφθεί

Τα αρχεία BAU εκδίδονται για κάθε μάθημα. Τα αρχεία BAU είναι μια ειδική μορφή αρχείου που τελικά κρατά τους βαθμούς των φοιτητών όταν βαθμολογούνται από τον καθηγητή

Τα αρχεία BAU αποστέλλονται σε κάθε καθηγητή

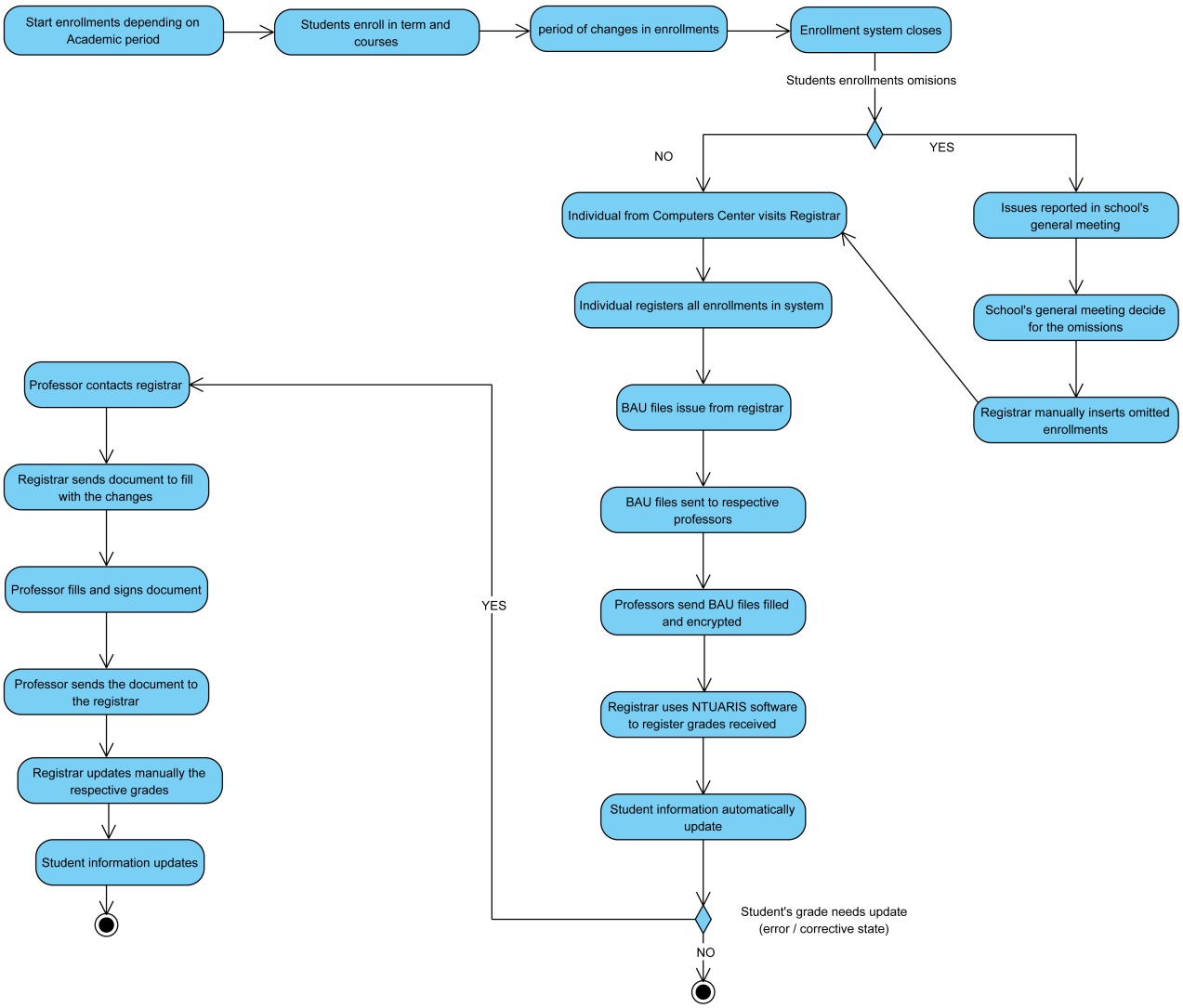
Οι καθηγητές συμπληρώνουν το αντίστοιχο αρχείο BAU με τους βαθμούς των μαθητών και στέλνουν το αρχείο πίσω κρυπτογραφημένο στον γραμματέα

Ο γραμματέας χρησιμοποιεί συγκεκριμένο λογισμικό για την εγγραφή βαθμών στο παρόν σύστημα

Οι φοιτητές μπορούν να δουν τον βαθμό από την αντίστοιχη διεπαφή που ανακτά τα δεδομένα τους

- Σε περίπτωση που ο βαθμός του φοιτητή χρειάζεται ενημέρωση λόγω οποιουδήποτε σφάλματος, ο καθηγητής πρέπει να επικοινωνήσει με τον γραμματέα
- Ο γραμματέας στέλνει το έγγραφο

- Ο καθηγητής το συμπληρώνει με αλλαγές, υπογράφει το έγγραφο και το στέλνει πίσω στον γραμματέα
- Ο γραμματέας ενημερώνει χειροκίνητα τους αντίστοιχους βαθμούς
- Ενημερώσεις πληροφοριών φοιτητή



Σχήμα 3.1: *Διαδικασία που ακολουθεί το υπάρχων σύστημα*

Το πρόβλημα που αντιμετωπίζει αυτή η διπλωματική εργασία είναι η ακεραιότητα και η ασφάλεια των βαθμών που τελικά αποθηκεύονται στο σύστημα. Οι βαθμοί αποστέλλονται από τον καθηγητή στον γραμματέα μέσα σε ένα αρχείο και αυτό το αρχείο μπορεί να παραποιηθεί, παρόλο που είναι κρυπτογραφημένο. Επιπλέον, όταν ο βαθμός ενός φοιτητή χρειάζεται να ενημερωθεί μετά την αποστολή των βαθμών από τον καθηγητή και την εισαγωγή στο σύστημα από τον γραμματέα, δεν αποστέλλεται νέο αρχείο. Αντίθετα, αποστέλλεται ένα απλό έγγραφο που αναφέρει ποιες ενημερώσεις πρέπει να γίνουν. Οι τρέχουσες εφαρμογές διαχείρισης βαθμού που δεν βασίζονται σε blockchain λαμβάνουν διάφορα μέτρα για να διασφαλίσουν τόσο την ακεραιότητα όσο και την εμπιστευτικότητα των δεδομένων. Αυτά τα

μέτρα βασίζονται συνήθως στην εμπιστοσύνη κάποιου τρίτου μέρους, όπως μια CA, οι διαχειριστές της βάσης δεδομένων, η κρυπτογραφία βάσης δεδομένων κ.λπ. Όπως αναφέρθηκε προηγουμένως, όλοι οι κόμβοι του δικτύου Blockchain έχουν ένα ακριβές αντίγραφο του καθολικού που διανέμεται στο δίκτυο. Έτσι, η τεχνολογία Blockchain προσθέτει ένα ακόμη επίπεδο ασφάλειας, αφού το καθολικό υπάρχει σε διαφορετικούς υπολογιστές/κόμβους που συμμετέχουν στο δίκτυο και είναι πρακτικά αδύνατο να παραβιαστεί ο καθένας από αυτούς. Τέλος, προτείνεται ένα σύστημα που λειτουργεί παράλληλα με το παλαιού τύπου σύστημα, ώστε να αποθηκεύει τα ίδια δεδομένα που αποθηκεύει το υπάρχων σύστημα, αλλά και να έχει τη δυνατότητα επικύρωσης εάν έχουν αλλάξει δεδομένα που τελικά κατέληξαν στη βάση δεδομένων του.

3.2 Μια προσέγγιση βασισμένη στο Blockchain

Είναι σημαντικό να σημειωθεί ότι αυτή η πρόταση δεν αλλάζει το σύστημα παλαιού τύπου, αλλά στην πραγματικότητα υπάρχει μαζί με αυτό.

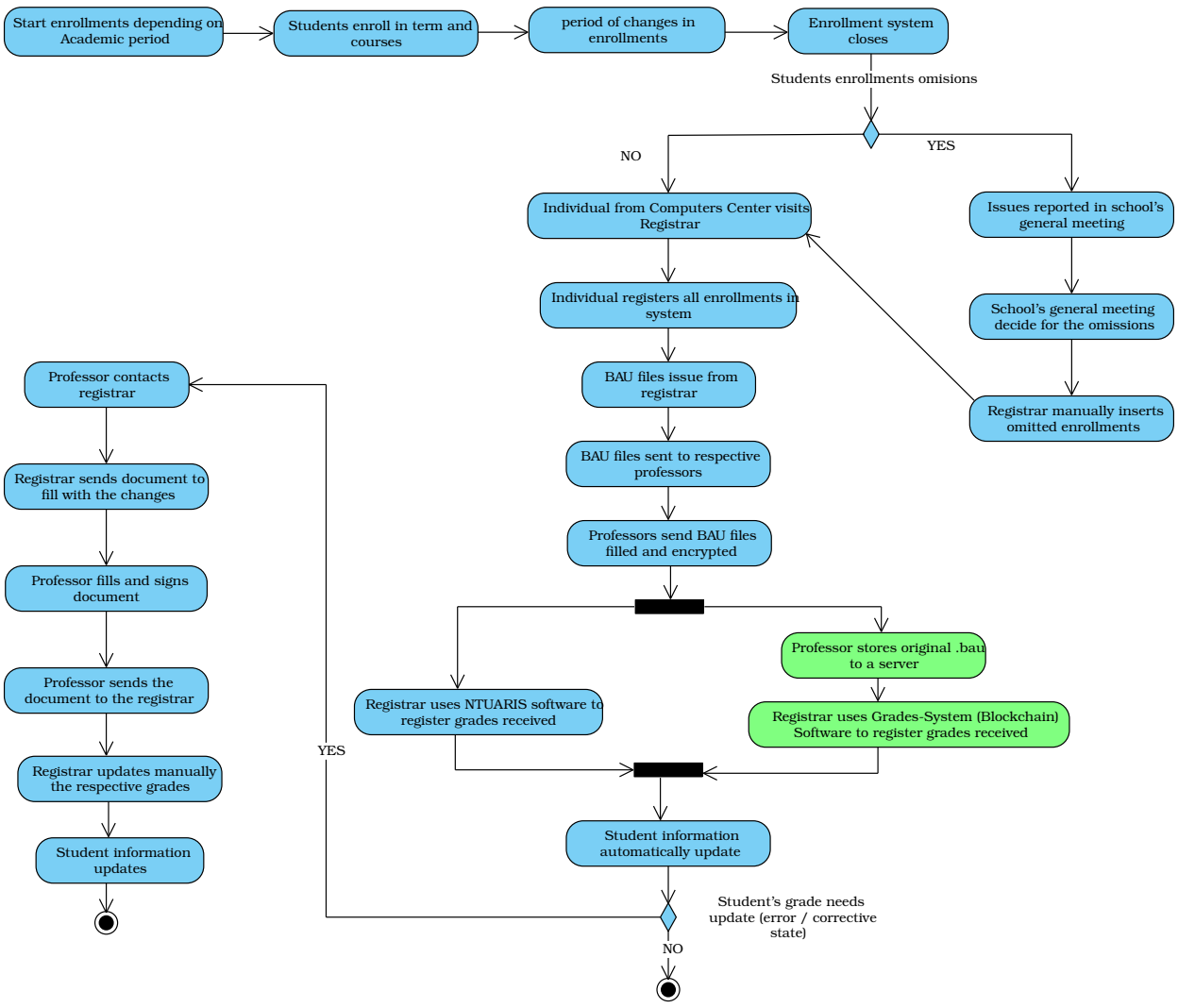


Figure 3.2: *Προτεινόμενη διαδικασία - Συνύπαρξη του υπάρχοντος συστήματος με το σύστημα Blockchain*

3.2.1 Ιδιωτικότητα - Κλειστότητα

Το απόρρητο είναι μια σημαντική έννοια σε αυτό το σύστημα, καθώς μόνο συγκεκριμένοι χρήστες πρέπει να μπορούν να αποθηκεύουν, να ανακτούν και να ολοκληρώνουν λειτουργίες με αυτά τα δεδομένα. Ο λόγος είναι ότι αυτά τα δεδομένα αποτελούν προσωπικά στοιχεία των φοιτητών και επομένως δεν πρέπει να εκτίθενται στο κοινό. Αντίθετα, πρέπει να εφαρμοστεί ένας συγκεκριμένος μηχανισμός για τη διατήρηση του απορρήτου των δεδομένων.

Έτσι, το απόρρητο του δικτύου μπορεί να επιτευχθεί με τη δημιουργία ενός ιδιωτικού δικτύου Blockchain καθώς και με τα αντίστοιχα έξυπνα συμβόλαια. Αφού τα έξυπνα συμβόλαια είναι η λογική της αποκεντρωμένης εφαρμογής, μπορεί να εφαρμοστεί ένα σύστημα αδειών, όπου οι υπάρχοντες χρήστες μπορούν να ψηφίσουν νέους χρήστες, αν μπορούν να χρησιμοποιήσουν το dApp ή όχι. Σε περίπτωση που ένας χρήστης δεν γίνει αποδεκτός από άλλους, δεν θα μπορεί να εκτελέσει καμία ενέργεια στο dApp και τελικά να αλληλεπιδράσει με τα έξυπνα συμβόλαια. Με την εφαρμογή ενός μηχανισμού ψηφοφορίας, μπορούν να αποκλειστούν κακόβουλοι χρήστες που προσπαθούν να αλληλεπιδράσουν με τα έξυπνα συμβόλαια.

3.2.2 Επικύρωση και Ακεραιότητα

Εφόσον η προτεινόμενη λύση πρέπει να αντιμετωπίσει το πρόβλημα της ακεραιότητας των βαθμών των φοιτητών, πρέπει να υπάρχει ένας αντίστοιχος μηχανισμός που να μπορεί να επικυρώσει εάν κάτι έχει αλλάξει στο αρχείο BAU που τελικά έχει καταλήξει στο σύστημα παλαιού τύπου.

Για το λόγο αυτό, αυτά τα αρχεία θα πρέπει να αποθηκεύονται στο blockchain και θα πρέπει να υπάρχει μια δυνατότητα επικύρωσης του αρχείου που αποστέλλεται από τον καθηγητή στον γραμματέα, ανά πάσα στιγμή. Όπως αναφέρθηκε παραπάνω, ο γραμματέας λαμβάνει και αποθηκεύει το αρχείο στο σύστημα παλαιού τύπου με τη βοήθεια συγκεκριμένου λογισμικού. Το περιεχόμενο του αρχείου υπάρχει σε μια βάση δεδομένων και το αρχείο αποθηκεύεται επίσης σε διακομιστή. Η επικύρωση μπορεί να γίνει συγκρίνοντας το αρχείο στον διακομιστή με τα αντίστοιχα δεδομένα στο blockchain και εάν εντοπιστούν διαφορές, μπορεί να δημιουργηθεί μια οπτικοποίηση αυτών.

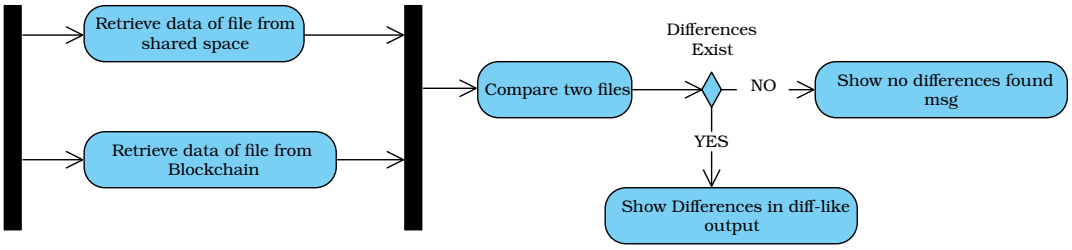


Figure 3.3: *UML Activity Diagram - Διαδικασία επικύρωσης*

3.2.3 Διεπαφή χρήστη

Πρέπει να αναπτυχθεί μια αποκεντρωμένη εφαρμογή (dApp) για τους χρήστες αυτού του συστήματος. Μέσω αυτής της αποκεντρωμένης δικτυακής διεπαφής, οι χρήστες μπορούν

να συνδεθούν με το πορτοφόλι τους και να εκτελέσουν όλες τις απαραίτητες ενέργειες στο σύστημα αλληλεπιδρώντας με τα έξυπνα συμβόλαια που έχουν αναπτυχθεί. Η διαδικασία για τους χρήστες θα πρέπει να είναι απλή χωρίς περίπλοκα βήματα και τα δεδομένα να παρουσιάζονται ανάλογα.

3.3 Μια προσέγγιση Blockchain με ιδιωτική άδεια, βασισμένη στο Ethereum

Η περιγραφόμενη λύση είναι ζωτικής σημασίας να εφαρμοστεί με ένα ιδιωτικό ή εξουσιοδοτημένο δίκτυο blockchain, καθώς οι μόνοι παράγοντες που πρέπει να αλληλεπιδρούν με το σύστημα είναι οι οντότητες ενός πανεπιστημίου που εμπλέκονται στη ροή εργασιών της διαχείρισης βαθμών, όπως οι γραμματείς, οι φοιτητές και οι καθηγητές.

Για την υλοποίηση ενός δικτύου με άδεια υπάρχουν διαφορετικές εναλλακτικές λύσεις, μερικές από τις οποίες συζητούνται εν συντομία στη συνέχεια [13].

1. Το R3 Corda [14] είναι μια καινοτόμος πλατφόρμα blockchain για επιχειρήσεις, η οποία στοχεύει στη μείωση του κόστους των επιχειρηματικών συναλλαγών και στην αύξηση της ταχύτητάς τους. Αυτό το έργο σχεδιάστηκε αρχικά για τον χρηματοοικονομικό τομέα, ωστόσο, μπορεί να εφαρμοστεί και σε άλλες χρήσεις, όπως η υγειονομική περίθαλψη, η αλυσίδα εφοδιασμού, οι κυβερνητικές και δημόσιες υπηρεσίες και η χρηματοδότηση του εμπορίου.

2. Η πλατφόρμα Quorum δημιουργήθηκε από την εταιρεία JP Morgan και είναι η εταιρική έκδοση του blockchain Ethereum [15]. Τροποποιώντας τον πυρήνα του Ethereum, η πλατφόρμα Quorum μπορεί να ενσωματώσει γρήγορα ενημερώσεις Ethereum. Είναι μια πλατφόρμα blockchain ανοιχτού κώδικα (open-source) που χρησιμοποιεί αλγόριθμους βασισμένους σε ψήφους για την εκτέλεση εκατοντάδων συναλλαγών ανά δευτερόλεπτο. Καθώς είναι μια ιδιωτική πλατφόρμα blockchain, επιτρέπει μόνο σε εξουσιοδοτημένους συμμετέχοντες να συμμετέχουν σε συναλλαγές.

3. Η πλατφόρμα Ripple στοχεύει στη σύνδεση χρηματοοικονομικών φορέων, εμπορικών εταιρειών, τραπεζών και παρόχων υπηρεσιών πληρωμών [16]. Το Ripple επιτρέπει διεθνείς πληρωμές μέσω ενός ψηφιακού στοιχείου που ονομάζεται Ripple ή XRP. Χρησιμοποιώντας μια πιθανολογική μέθοδο ψηφοφορίας, η πλατφόρμα Ripple επιτυγχάνει συναίνεση μεταξύ των κόμβων στο δίκτυο. Αρκετές μεγάλες εταιρείες όπως η American Express, η SBI Holdings και η Deloitte πειραματίζονται με τις δυνατότητες blockchain του Ripple για να μεταμορφώσουν τις διαδικασίες πληρωμής.

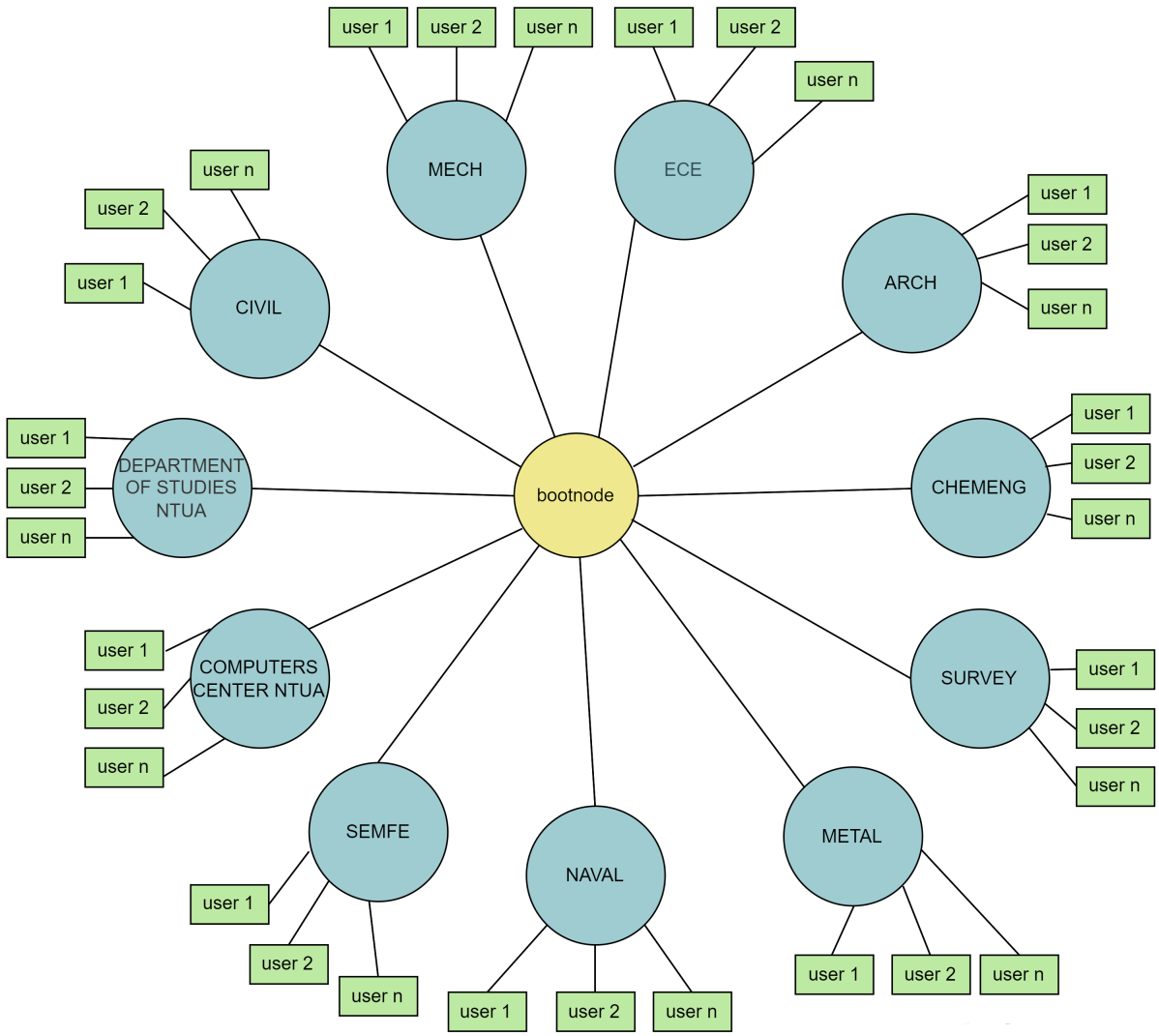
4. Το Hyperledger [17] Foundation προσφέρει λογισμικά και συστήματα blockchain ανοιχτού κώδικα που αποτελείται από τεχνολογίες κατανεμημένων καθολικών, βιβλιοθήκες και εργαλεία.

   Τα DLT που προσφέρει το Hyperledger Foundation είναι τα Fabric, Sawtooth, Iroha, Besu και Indy. Το Hyperledger Fabric έχει σχεδιαστεί για δίκτυα με άδεια και επιτρέπει μόνο σε οντότητες με γνωστές ταυτότητες να συμμετέχουν στο σύστημα. Μόνο εξουσιοδοτημένοι συμμετέχοντες μπορούν να συμμετέχουν στις συναλλαγές που πραγματοποιούνται στην πλατφόρμα Hyperledger Fabric. Έτσι, το Hyperledger Fabric [18] θεωρήθηκε ως επιλογή για την υλοποίηση του δικτύου Blockchain.

Παρά τις διαφορετικές επιλογές που υπάρχουν για την υλοποίηση του ιδιωτικού δικτύου blockchain, το Ethereum επιλέχθηκε για διάφορους λόγους. Πρώτα και κύρια είναι ότι οι

περισσότερες από τις αναφερόμενες επιλογές ήταν πολύ συγκεκριμένες για το συγκεκριμένο έργο και η μόνη εναλλακτική θα μπορούσε να είναι το Hyperledger Fabric. Ο δεύτερος λόγος είναι ότι το ιδιωτικό δίκτυο Ethereum ήταν αρκετά απλό στη ρύθμιση σε σύγκριση με το Hyperledger Fabric, το οποίο μπορεί να τεκμηριωθεί και από το Κεφάλαιο 7.1 του [13]. Τέλος, το Ethereum διαθέτει πλούσια τεκμηρίωση και μια ισχυρή κοινότητα που αποτελεί εξαιρετική πηγή για τυχόν προβλήματα που μπορεί να προκύψουν.

Παρακάτω, εμφανίζεται ένα απλό διάγραμμα με τους κόμβους του ιδιωτικού με άδεια δίκτυο blockchain και τον τρόπο με τον οποίο οι χρήστες μπορούν να συνδεθούν σε έναν κόμβο και να εκτελέσουν λειτουργίες στο blockchain μέσω των έξυπνων συμβολαίων.



Σχήμα 3.4: *Αναπαράσταση κόμβων και χρηστών του συστήματος*

3.4 Μοντέλο περίπτωσης χρήσης

Οι απαιτήσεις που εξηγήθηκαν προηγουμένως παρουσιάζονται παρακάτω σε ένα διάγραμμα περίπτωσης χρήσης UML. Οι φορείς που εμπλέκονται σε αυτές τις περιπτώσεις χρήσης είναι οι χρήστες διαχειριστές και οι χρήστες γραμματείς.

3.4.1 Γραμματείς

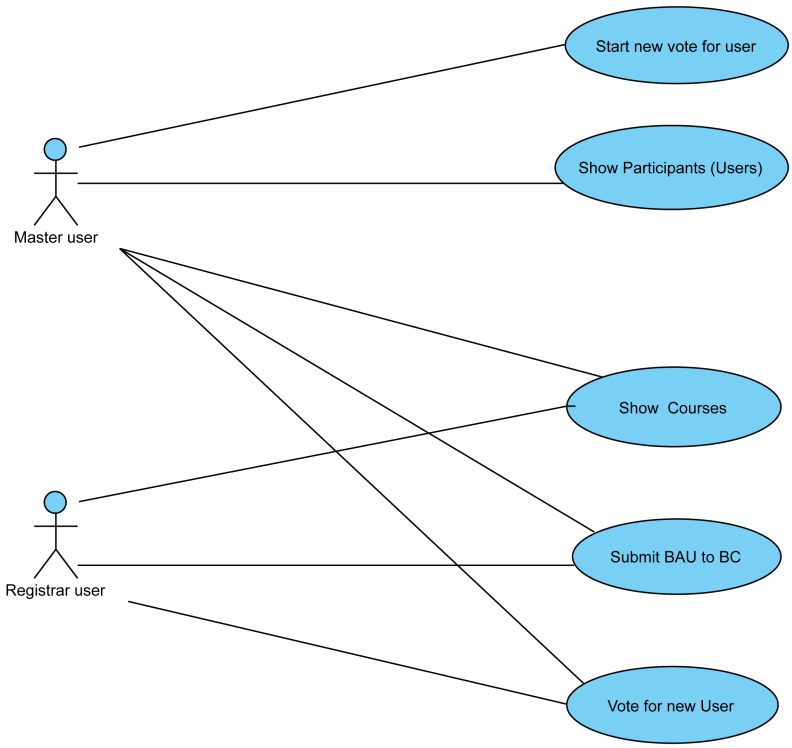
Οι γραμματείς μπορούν να εκτελέσουν τις ακόλουθες λειτουργίες στο σύστημα:

1. Να υποβάλουν ένα νέο αρχείο βαθμών στο blockchain
2. Να δούν τα αντίστοιχα μαθήματα της σχολής στην οποία υπάγονται
3. Να ψηφίσουν υπέρ ή κατά για ένα νέο χρήστη, ώστε να γίνει ή οχι δεκτός στο σύστημα

3.4.2 Διαχειριστές

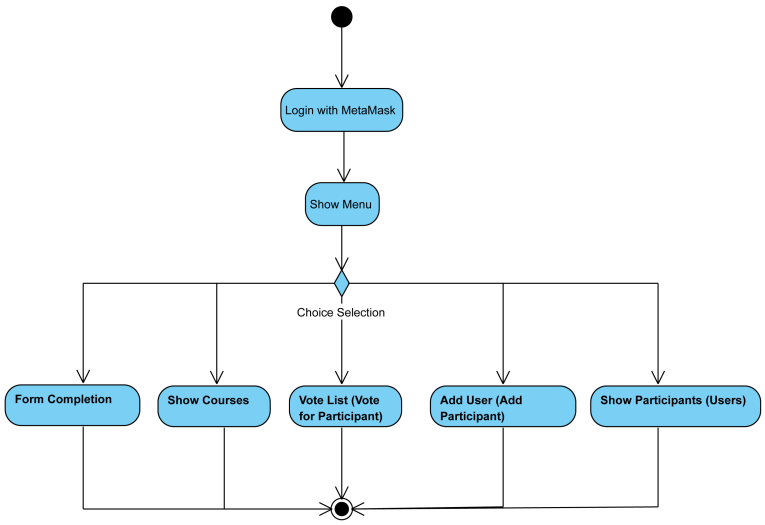
Οι διαχειριστές μπορούν να εκτελέσουν όλες τις λειτουργίες που μπορούν οι γραμματείς καθώς και:

- Να ξεκινήσουν μια νέα διαδικασία ψηφοφορίας για ένα νέο χρήστη
- Να ανακτήσουν όλους τους συμμετέχοντες του συστήματος



Σχήμα 3.5: *UML Use Case Διάγραμμα*

3.5 UML Διάγραμμα δραστηριότητας ροής εργασίας υψηλού επιπέδου



Σχήμα 3.6: *UML High-level work flow Activity Diagram*

Το παραπάνω διάγραμμα δραστηριότητας υπογραμμίζει τη ροή εργασίας υψηλού επιπέδου της αποκεντρωμένης εφαρμογής. Το πρώτο βήμα είναι ότι ο χρήστης πρέπει να συνδεθεί στο dApp χρησιμοποιώντας το MetaMask. Όταν ολοκληρωθεί η σύνδεση, οι πληροφορίες του χρήστη ανακτώνται από το Blockchain. Χρησιμοποιώντας αυτές τις πληροφορίες πραγματοποιείται έλεγχος αδειών. Σε περίπτωση που ο χρήστης έχει δικαιώματα, εμφανίζεται το κύριο μενού του dApp και μπορεί να εκτελέσει οποιαδήποτε από τις παρακάτω ενέργειες.

- Συμπλήρωση φόρμας: Τόσο οι διαχειριστές όσο και οι γραμματείς μπορούν να εκτελέσουν αυτήν την ενέργεια. Με τη συμπλήρωση αυτής της φόρμας και την προσθήκη όλων των απαραίτητων πληροφοριών, οι νέοι βαθμοί μαθημάτων καταχωρούνται στο Blockchain.

- Εμφάνιση μαθημάτων: Αυτή η ενέργεια μπορεί να εκτελεστεί και από τους δύο χρήστες, με μια σημαντική διαφορά. Οι διαχειριστές μπορούν να ανακτήσουν όλα τα μαθήματα και τελικά να δουν το καθολικό που είναι αποθηκευμένο στο Blockchain, ενώ οι γραμματείς μπορούν να ανακτήσουν μόνο πληροφορίες μαθημάτων σχετικά με τη σχολή στην οποία υπάγονται. Επιπλέον, μια άλλη λειτουργία αυτής της ενέργειας είναι ότι οι χρήστες μπορούν να επικυρώσουν τις ανακτημένες πληροφορίες μαθημάτων και σε περίπτωση οποιασδήποτε διαφοράς στα αποθηκευμένα δεδομένα και στα αρχικά δεδομένα, θα εμφανιστεί μια οπτική αναπαράστασή τους.

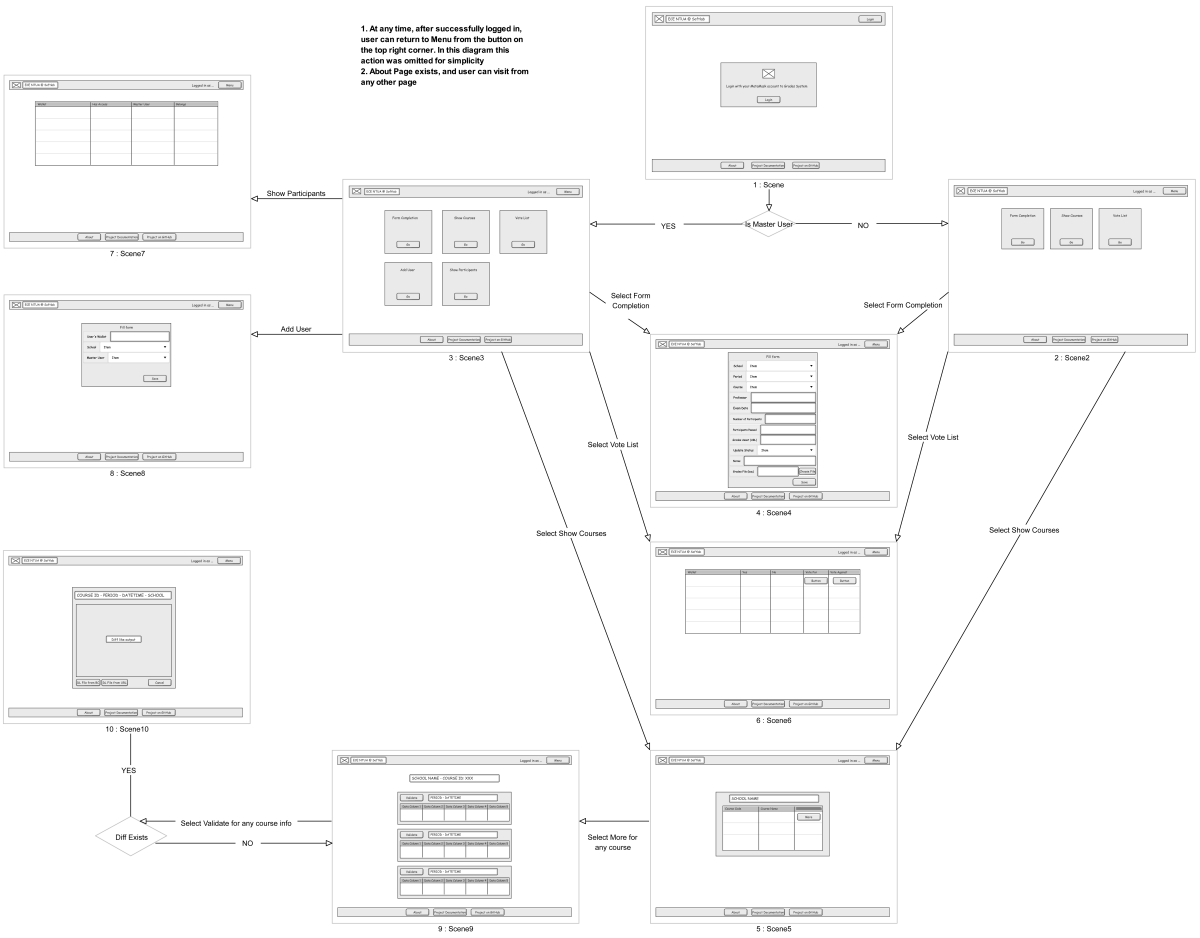
- Λίστα ψήφων (ψηφοφορία για συμμετέχοντα): Και οι δύο χρήστες μπορούν να χρησιμοποιήσουν αυτήν τη λειτουργία. Όταν ένας νέος χρήστης θέλει να έχει πρόσβαση στο κοινόχρηστο καθολικό, ξεκινά μια διαδικασία ψηφοφορίας από έναν κύριο χρήστη και

κάθε συμμετέχων με δικαιώματα στο σύστημα μπορεί να ψηφίσει υπέρ ή κατά αυτού του νέου χρήστη. Για να προστεθεί ο νέος χρήστης, πρέπει να υπάρχει ομοφωνία.

- Προσθήκη χρήστη: Αυτή η ενέργεια μπορεί να εκτελεστεί μόνο από κύριο χρήστη, όπου μια φόρμα συμπληρώνεται με τα απαραίτητα στοιχεία του αιτούντος.
- Εμφάνιση συμμετεχόντων: Κάθε χρήστης που έχει δικαιώματα στο σύστημα μπορεί να καταχωρηθεί εκτελώντας αυτήν την ενέργεια. Μόνο ένας διαχειριστής μπορεί να ανακτήσει όλους τους συμμετέχοντες.

Μια πιο λεπτομερής περιγραφή των παραπάνω μαζί με τα αντίστοιχα σχήματα είναι διαθέσιμη στη συνέχεια.

3.6 Wireflow



Σχήμα 3.7: *Wireflow diagram*

3.7 Σύνδεση και διαφορετικοί χρήστες

3.7.1 Περιγραφή

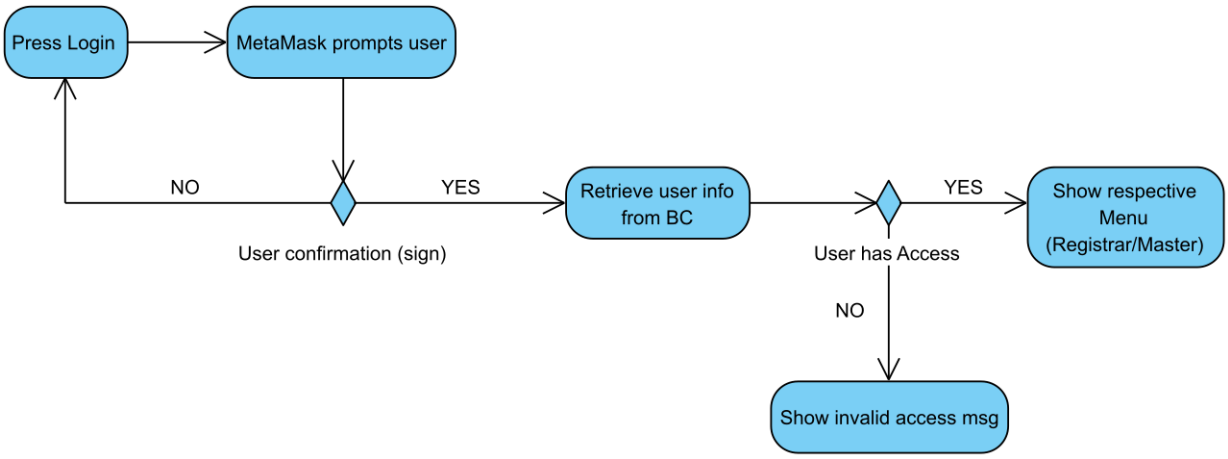
Όπως αναφέρθηκε παραπάνω υπάρχουν δύο διαφορετικά είδη χρηστών στο προτεινόμενο σύστημα με διαφορετικά δικαιώματα. Επομένως, η διεπαφή χρήστη τους θα πρέπει να διαφέρει. Κάτω εμφανίζεται το διάγραμμα δραστηριότητας της διαδικασίας σύνδεσης. Επιπλέον, η σελίδα σύνδεσης και η σελίδα μενού για τους αντίστοιχους χρήστες παρουσιάζονται σε Wireframes Lo-Fi.

3.7.2 Τι είναι το MetaMask

Το MetaMask είναι μια υπηρεσία που προσφέρεται ως επέκταση προγράμματος περιήγησης και εφαρμογή για κινητά που δημιουργήθηκε από την ConsenSys και επιτρέπει στους χρήστες να αποθηκεύουν και να διαχειρίζονται κλειδιά λογαριασμού, να διεκπεραιώνουν συναλλαγές, να στέλνουν και να λαμβάνουν κρυπτονομίσματα και μάρκες που βασίζονται στο Ethereum και να συνδέονται με ασφάλεια σε αποκεντρωμένες εφαρμογές μέσω ενός συμβατού πρόγραμμα περιήγησης ιστού ή το ενσωματωμένο πρόγραμμα περιήγησης της εφαρμογής για κινητά. Οι προγραμματιστές επιτυγχάνουν μια σύνδεση μεταξύ του MetaMask και των αποκεντρωμένων εφαρμογών τους χρησιμοποιώντας ένα ΘααΣριπτ plugin όπως το Web3js ή το Ethers για να ορίσουν τις αλληλεπιδράσεις μεταξύ του MetaMask και των έξυπνων συμβολαίων [19].

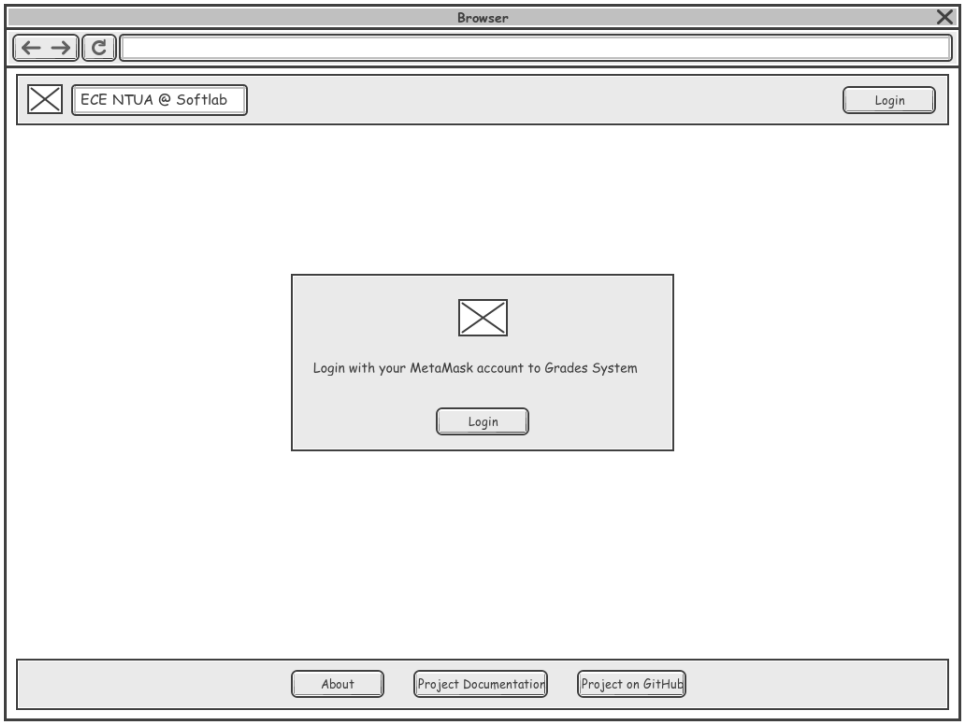
Οπότε, το MetaMask θα επιτρέψει στους χρήστες να αλληλεπιδρούν με το dApp με έναν εύκολο και διαφανή τρόπο.

3.7.3 Διάγραμμα δραστηριότητας UML



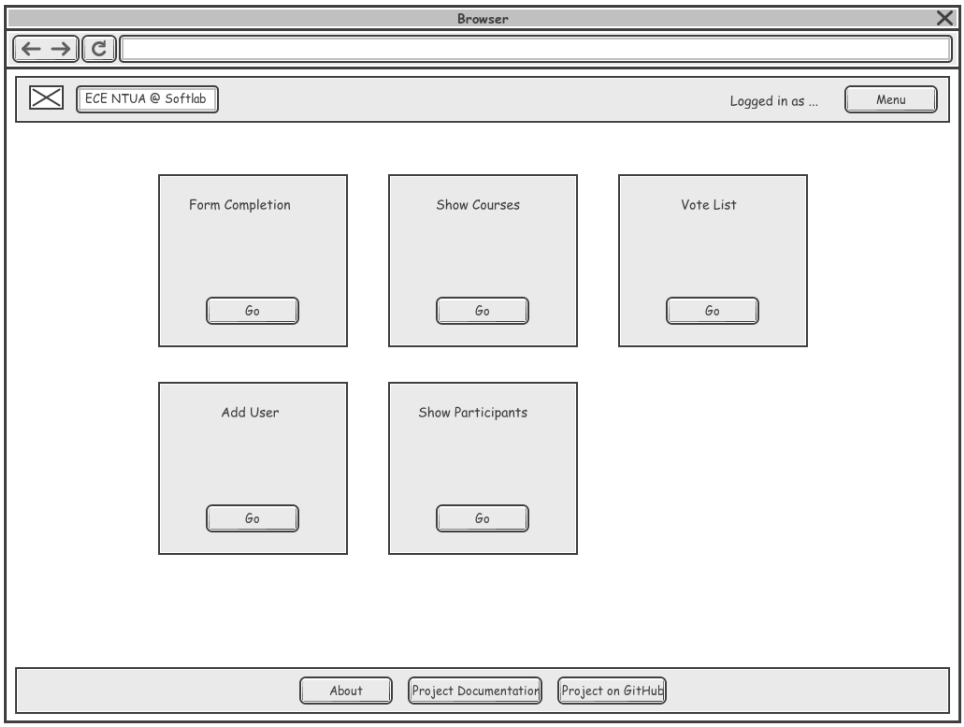
Σχήμα 3.8: *Διάγραμμα δραστηριότητας UML: Διαδικασία login*

3.7.4 Login - Metamask Lo-Fi Wireframe



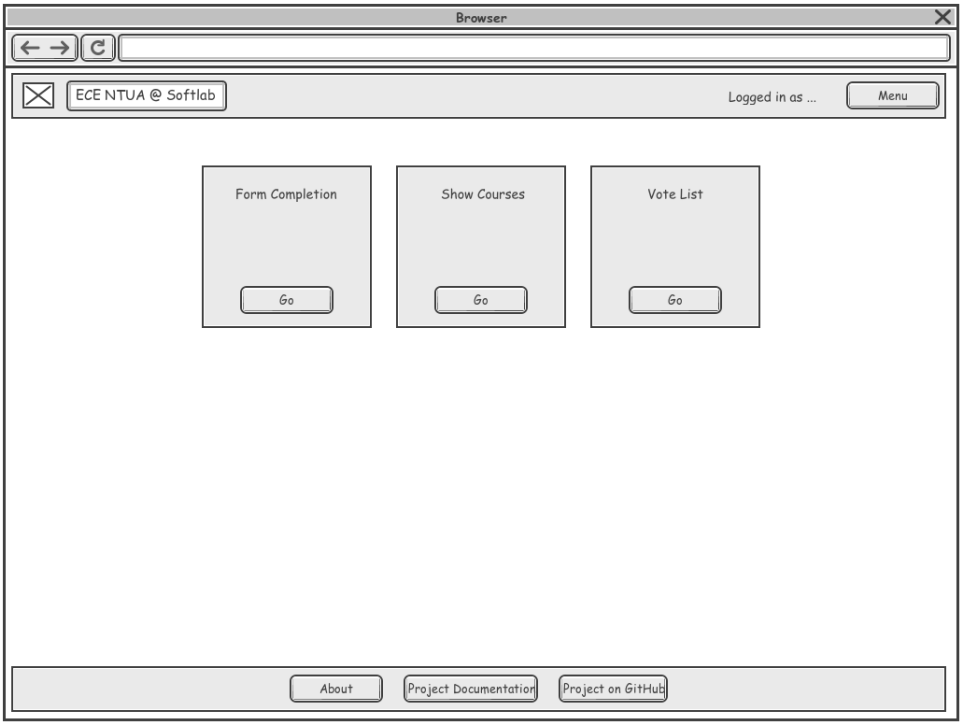
Σχήμα 3.9: *Login Wireframe*

3.7.5 Menu - Διαχειριστή Lo-Fi Wireframe



Σχήμα 3.10: *Menu - Διαχειριστή Lo-Fi Wireframe*

3.7.6 Menu - Χρήστη Lo-Fi Wireframe



Σχήμα 3.11: *Menu - Χρήστη Lo-Fi Wireframe*

3.8 Περίπτωση Χρήσης 1: Προσθήκη Βαθμολογιών

3.8.1 Περιγραφή

Αυτή η δυνατότητα μπορεί να χρησιμοποιηθεί τόσο από τους διαχειριστές όσο και από τους γραμματείς. Οι χρήστες πρέπει να συμπληρώσουν μια φόρμα με όλες τις πληροφορίες και όταν πατηθεί η αποθήκευση η φόρμα επικυρώνεται. Εάν εντοπιστούν σφάλματα επικύρωσης, η ενέργεια ακυρώνεται.

Οι διαχειριστές μπορούν να προσθέτουν βαθμούς για κάθε σχολή σε αντίθεση τον γραμματέα που μπορεί να προσθέσει βαθμούς μόνο για τη σχολή στην οποία υπάγεται.

3.8.2 Διάγραμμα δραστηριότητας UML - Ο καθηγητής δημιουργεί και διανέμει αρχείο

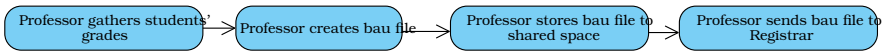
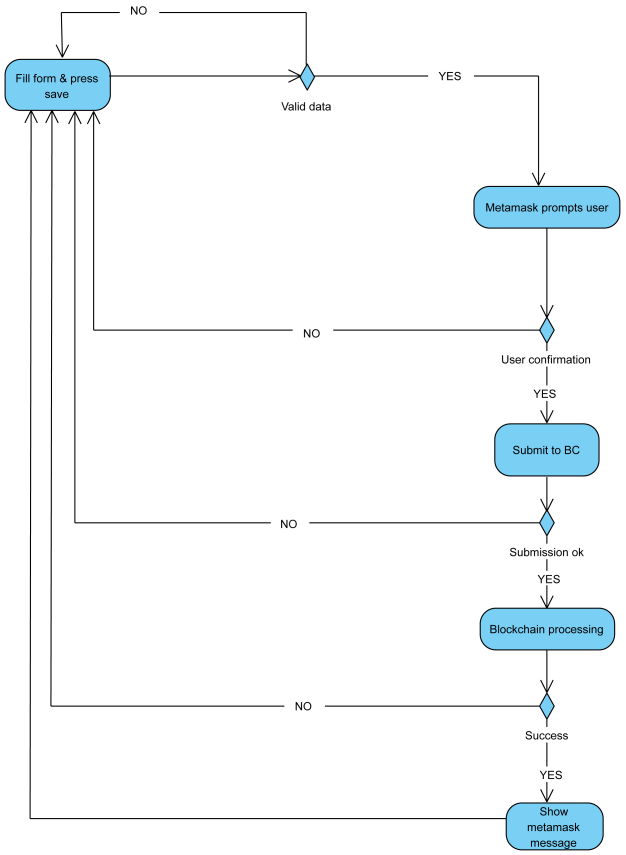


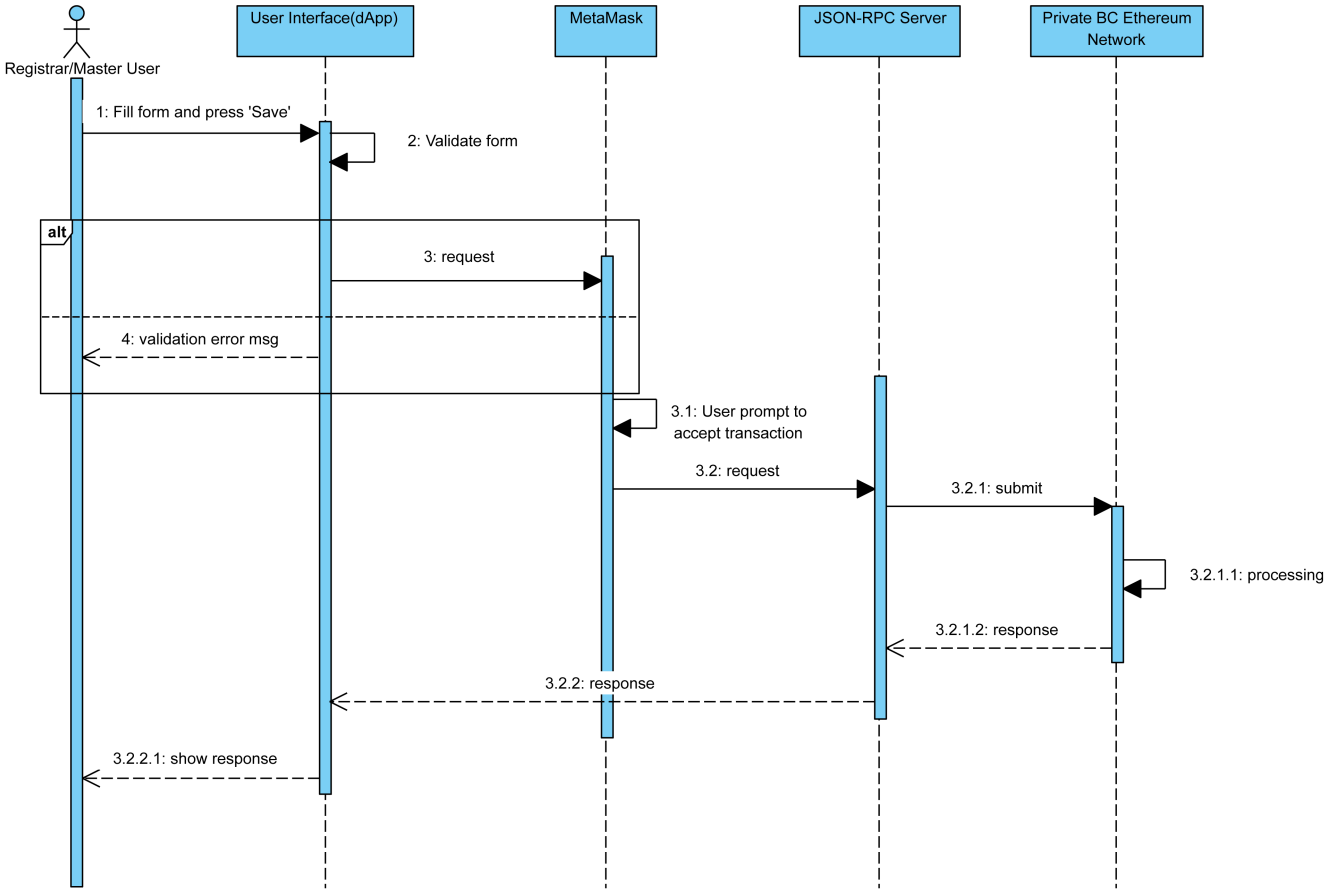
Figure 3.12: *Διάγραμμα δραστηριότητας UML - Ο καθηγητής δημιουργεί και διανέμει αρχείο*

3.8.3 Διάγραμμα δραστηριότητας UML - Προσθήκη Βαθμολογιών



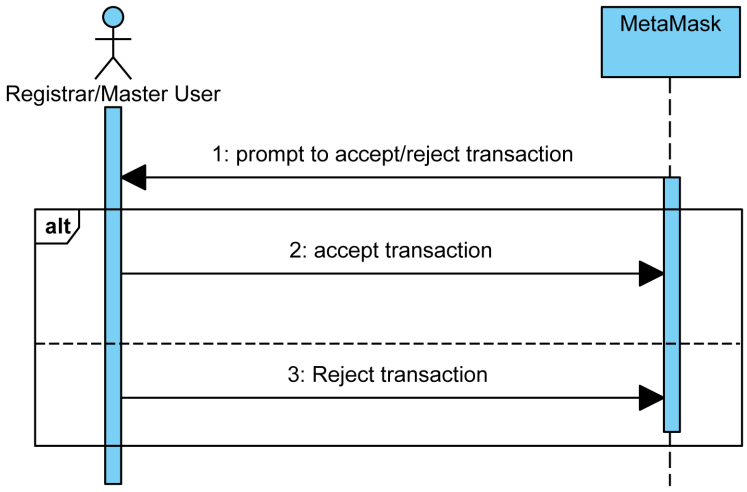
Σχήμα 3.13: *Διάγραμμα δραστηριότητας UML - Προσθήκη Βαθμολογιών*

3.8.4 Διάγραμμα ακολουθίας UML



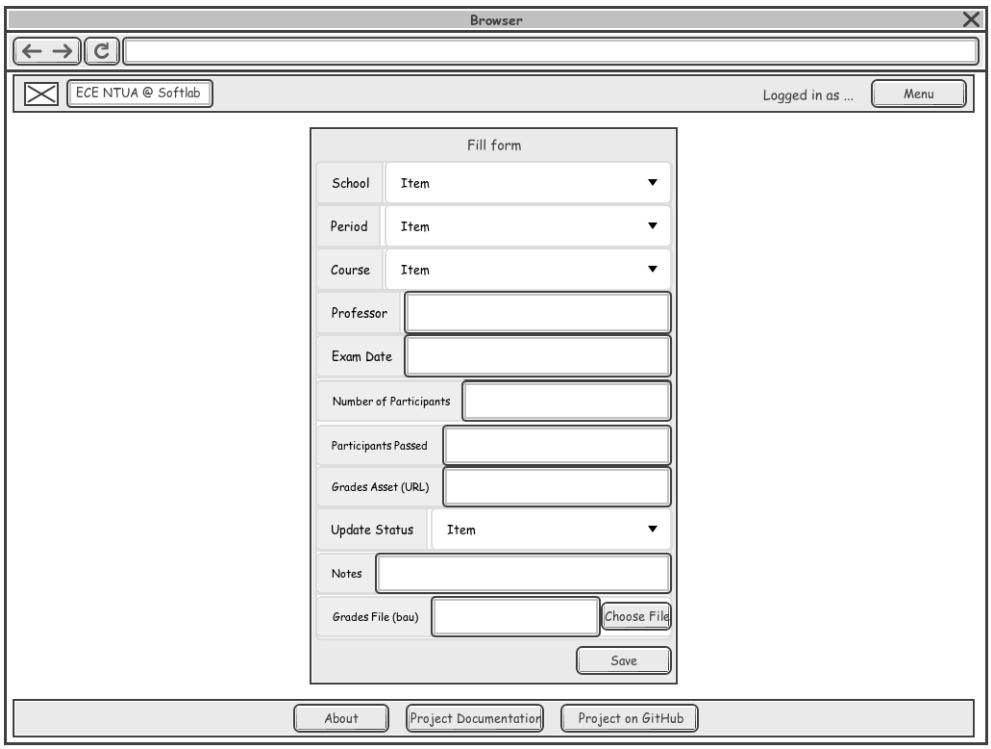
Σχήμα 3.14: *Διάγραμμα ακολουθίας UML: Προσθήκη Βαθμολογιών*

3.8.5 Βοηθητικό Διάγραμμα ακολουθίας UML



Σχήμα 3.15: *Διάγραμμα ακολουθίας UML: MetaMask ρωτά τον χρήστη για αποδοχή/απόρριψη συναλλαγής*

3.8.6 Lo-Fi Wireframe



Σχήμα 3.16: *Προσθήκη Βαθμολογιών Wireframe*

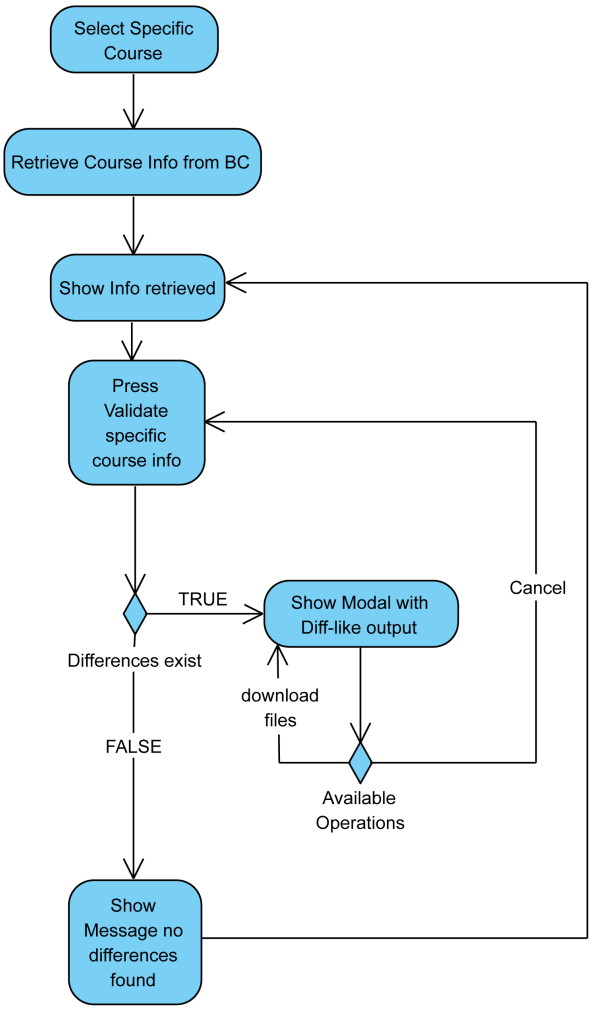
3.9 Περίπτωση Χρήσης 2: Εμφάνιση πληροφοριών μαθήματος και επικύρωση

3.9.1 Περιγραφή

Και οι δύο κατηγορίες χρηστών μπορούν να χρησιμοποιήσουν αυτήν τη λειτουργία, η οποία ανακτά πληροφορίες για ένα συγκεκριμένο μάθημα. Ένας διαχειριστής μπορεί να ανακτήσει τις πληροφορίες κάθε μαθήματος οποιασδήποτε σχολής, ενώ ένας γραμματέας μπορεί να ανακτήσει τις πληροφορίες του μαθήματος μόνο για την σχολή στην οποία ανήκει.

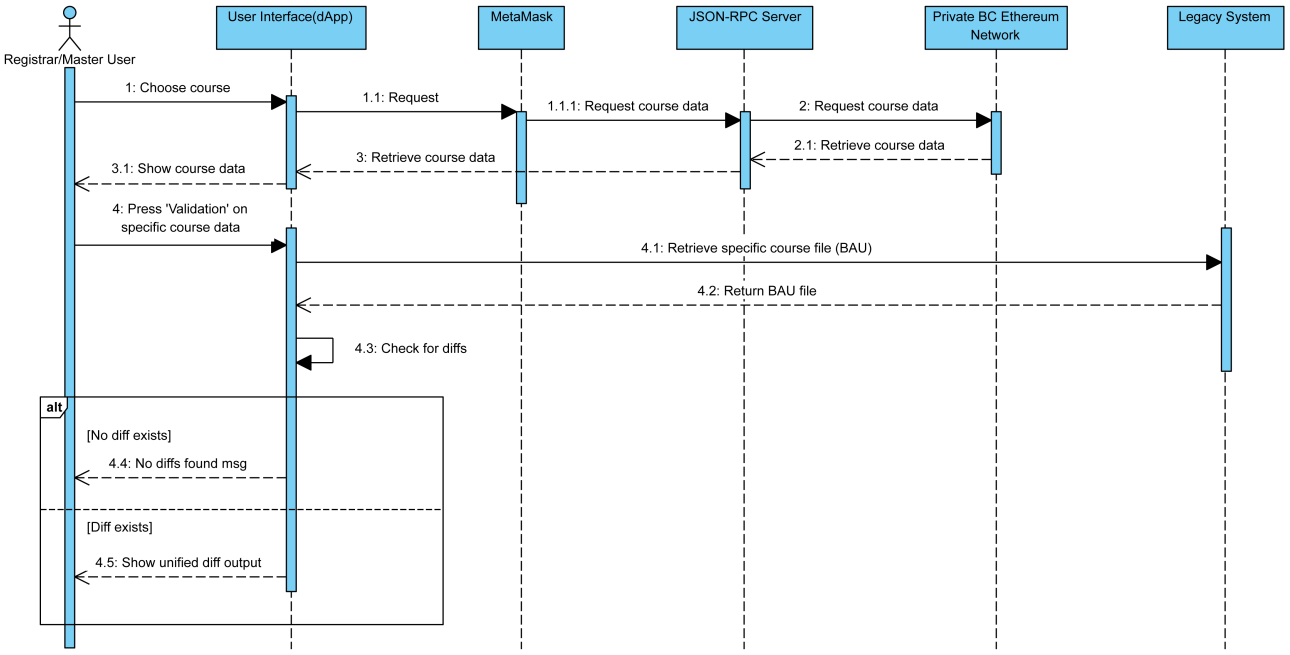
Οι χρήστες μπορούν επίσης να επικυρώσουν τις πληροφορίες του μαθήματος πατώντας το κουμπί Έπικύρωση'. Εάν εντοπιστούν διαφορές μεταξύ των δεδομένων που ανακτήθηκαν από το blockchain και των δεδομένων του αρχείου που μεταφορτώθηκε κατά την εισαγωγή, τότε φορτώνεται ένα παράθυρο με τις διαφορές και οι χρήστες μπορούν να πραγματοποιήσουν λήψη και των δύο αρχείων.

3.9.2 Διάγραμμα δραστηριότητας UML



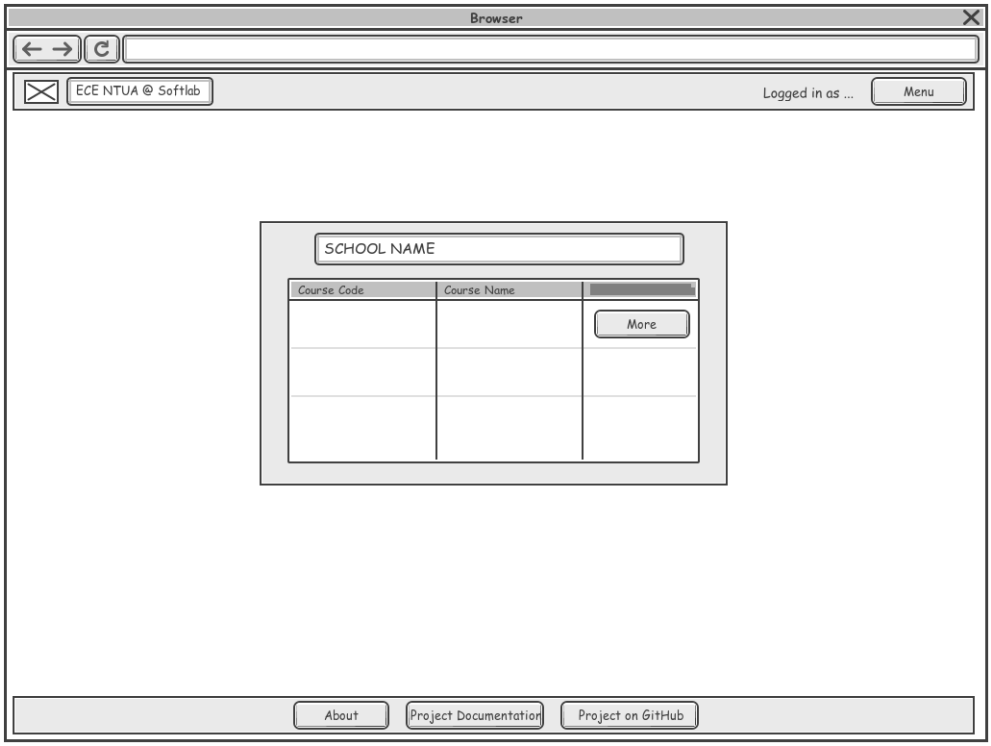
Σχήμα 3.17: *Διάγραμμα δραστηριότητας UML: Εμφάνιση πληροφοριών μαθήματος και επικύρωση*

3.9.3 Διάγραμμα ακολουθίας UML



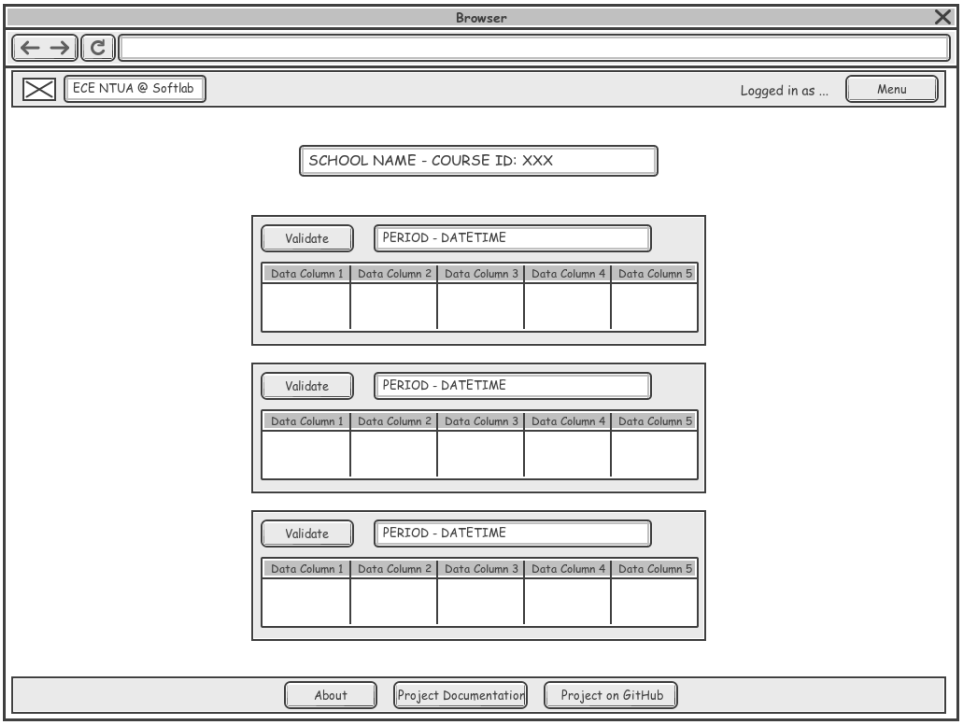
Σχήμα 3.18: *Διάγραμμα ακολουθίας UML: Εμφάνιση πληροφοριών μαθήματος και επικύρωση*

3.9.4 Εμφάνιση Μαθημάτων - LoFi Wireframe



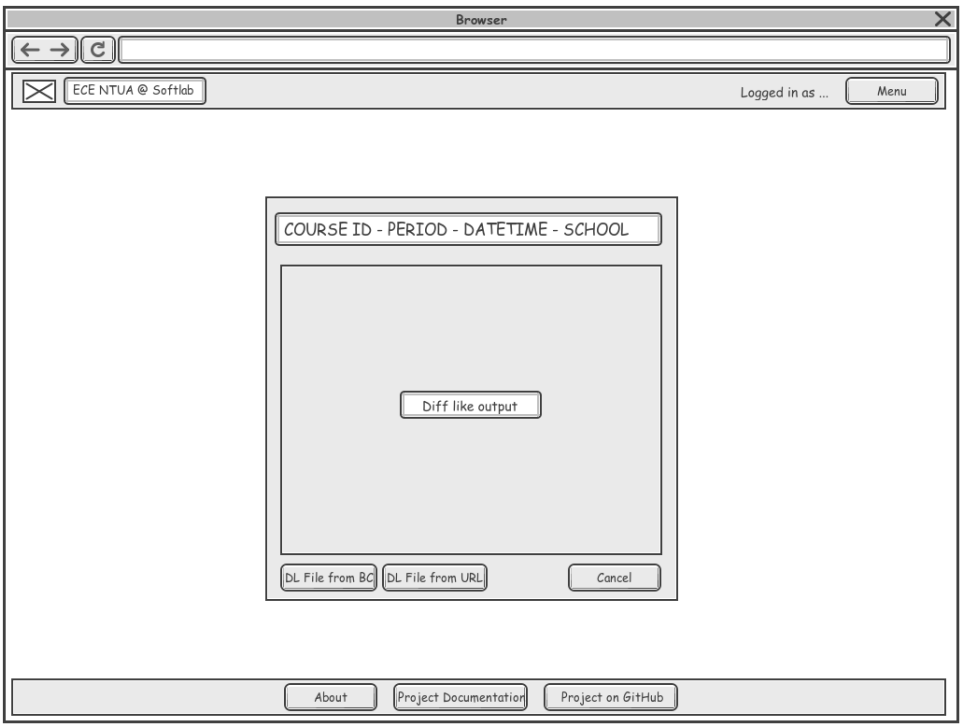
Σχήμα 3.19: *Εμφάνιση Μαθημάτων - LoFi Wireframe*

3.9.5 Εμφάνιση πληροφοριών μαθήματος - LoFi Wireframe



Σχήμα 3.20: *Εμφάνιση πληροφοριών μαθήματος - LoFi Wireframe*

3.9.6 Εμφάνιση διαφορών - LoFi Wireframe



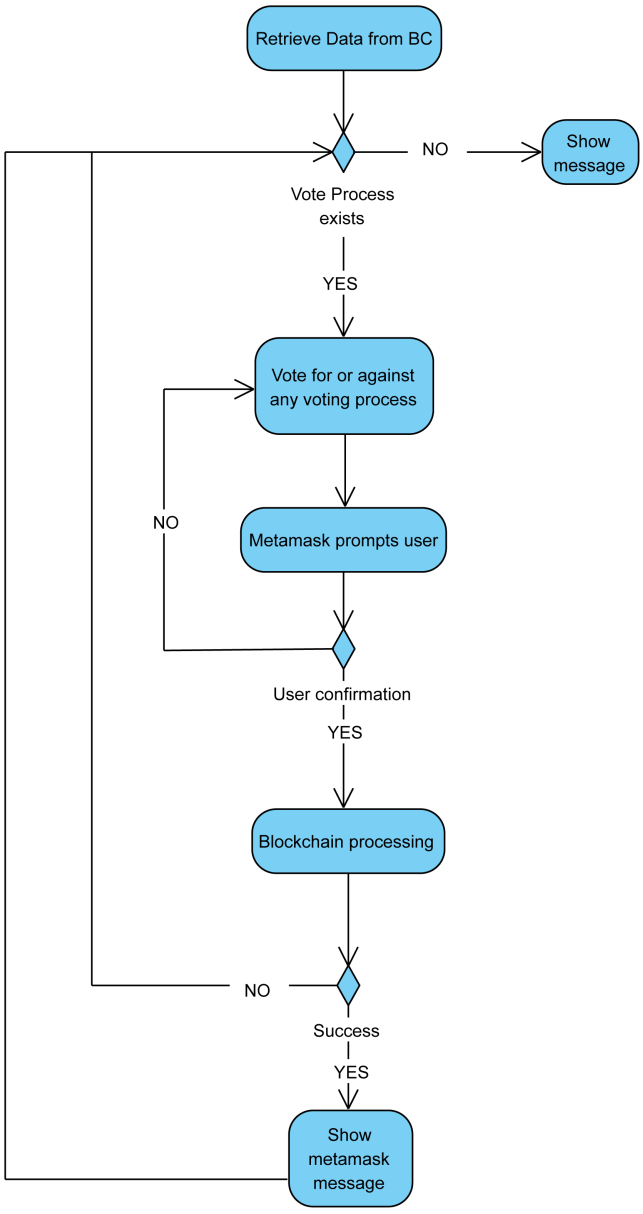
Σχήμα 3.21: *Εμφάνιση διαφορών - LoFi Wireframe*

3.10 Περίπτωση Χρήσης 3: Ψηφίστε για νέους χρήστες

3.10.1 Περιγραφή

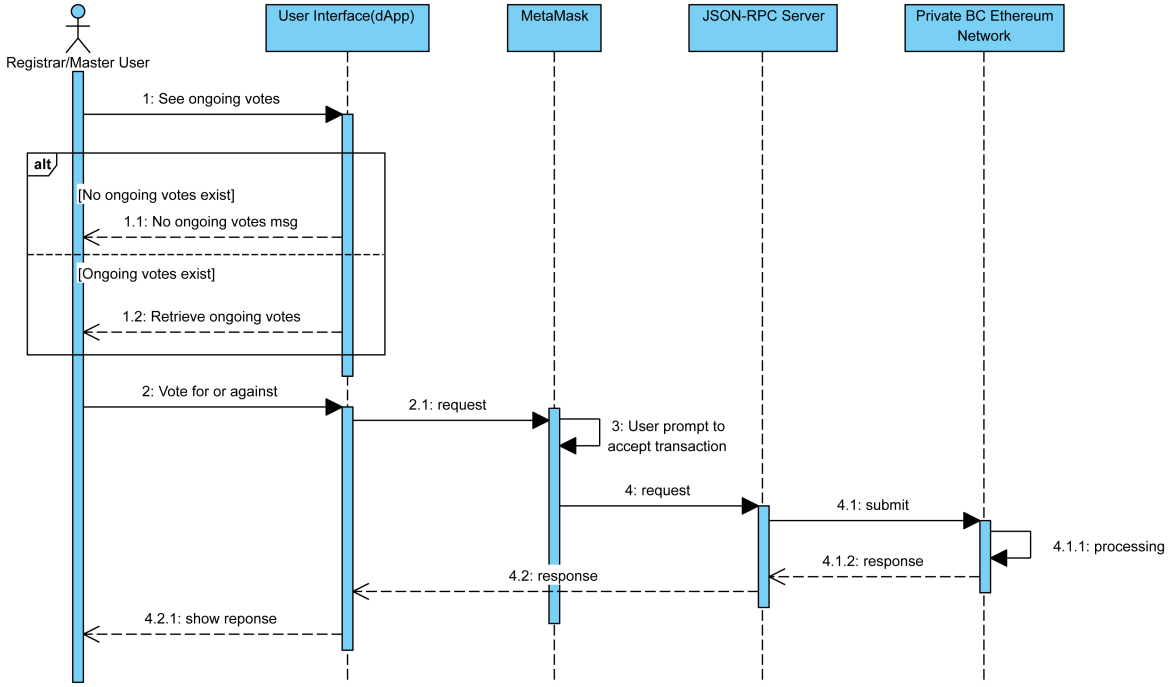
Όλοι οι χρήστες που έχουν πρόσβαση στη χρήση αυτού του dApp, έχουν δικαίωμα να ψηφίσουν υπέρ ή κατά οποιουδήποτε άλλου χρήστη που προσπαθεί να αποκτήσει πρόσβαση για χρήση του dApp. Μπορούν να ψηφίσουν μόνο μία φορά για κάθε ψηφοφορία που βρίσκεται σε εξέλιξη (μπορεί να υπάρχουν πολλές συνεχιζόμενες ψηφοφορίες) και ο αιτών χρήστης αποκτά πρόσβαση μόνο σε περίπτωση ομοφωνίας.

3.10.2 Διάγραμμα δραστηριότητας UML



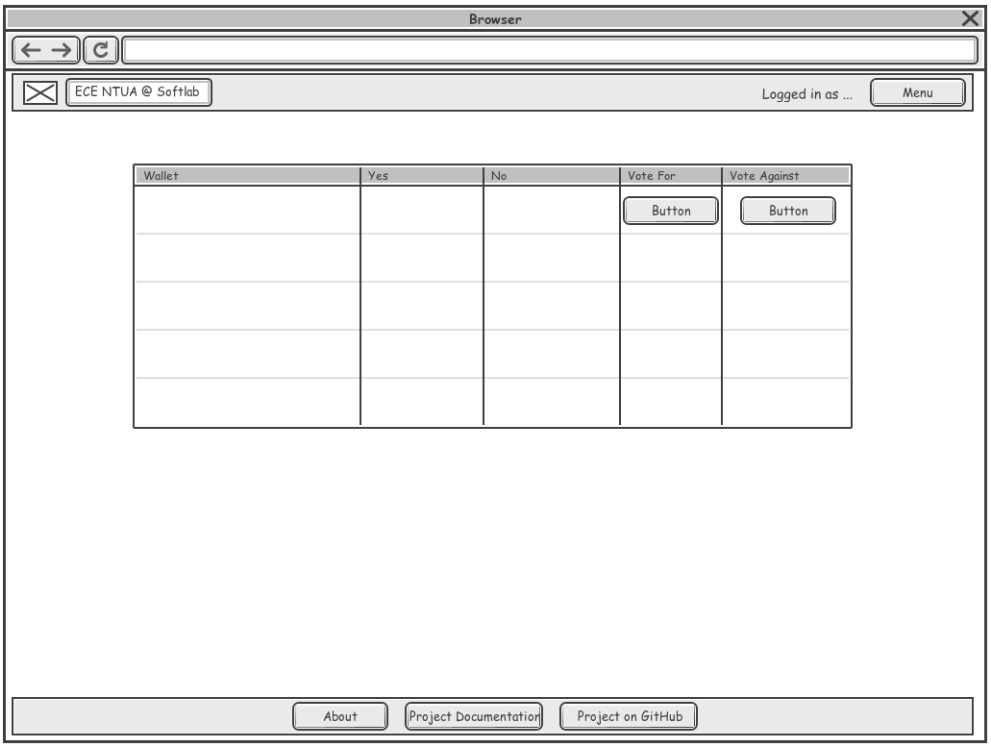
Σχήμα 3.22: *Διάγραμμα δραστηριότητας UML: Ψήφος σε νέους χρήστες*

3.10.3 Διάγραμμα ακολουθίας UML



Σχήμα 3.23: *Διάγραμμα ακολουθίας UML: Ψηφός σε νέους χρήστες*

3.10.4 Lo-Fi Wireframe



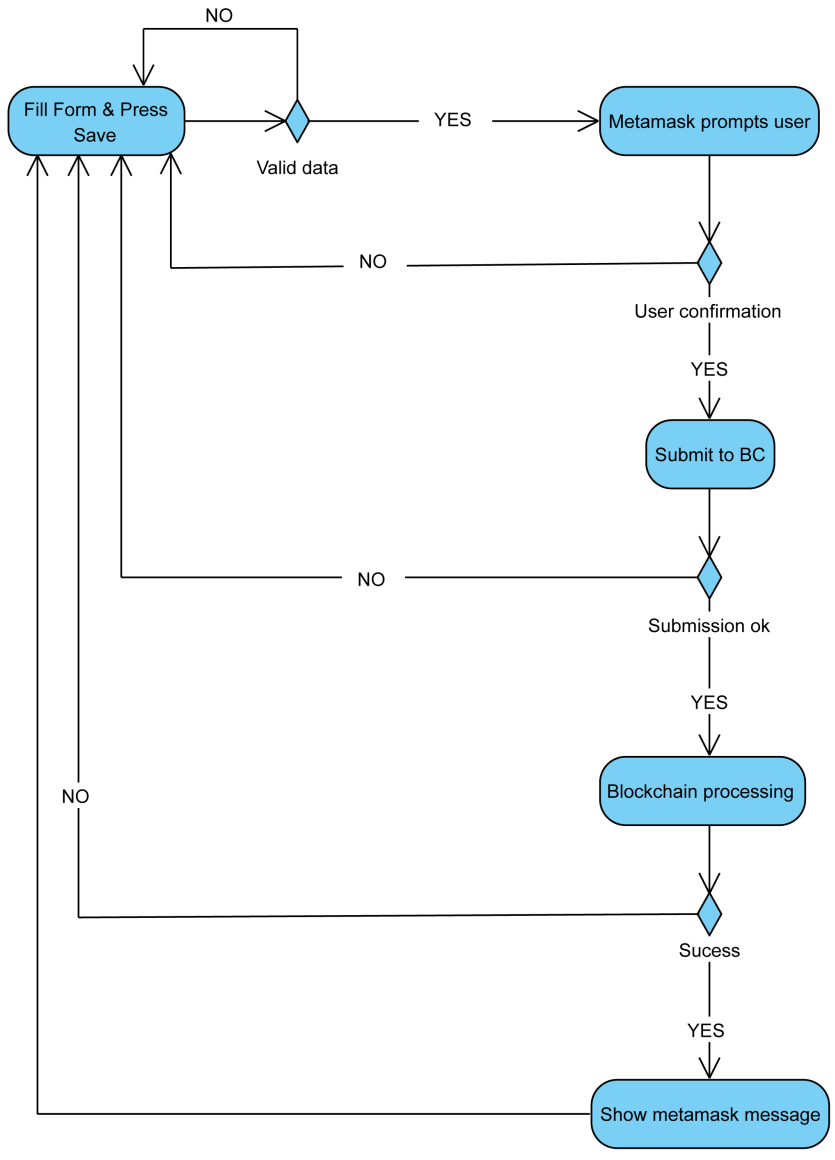
Σχήμα 3.24: *Ψηφίστε υπέρ ή κατά - Lo-Fi Wireframe*

3.11 Περίπτωση Χρήσης 4: Έναρξη μιας νέας διαδιασίας ψηφοφορίας

3.11.1 Περιγραφή

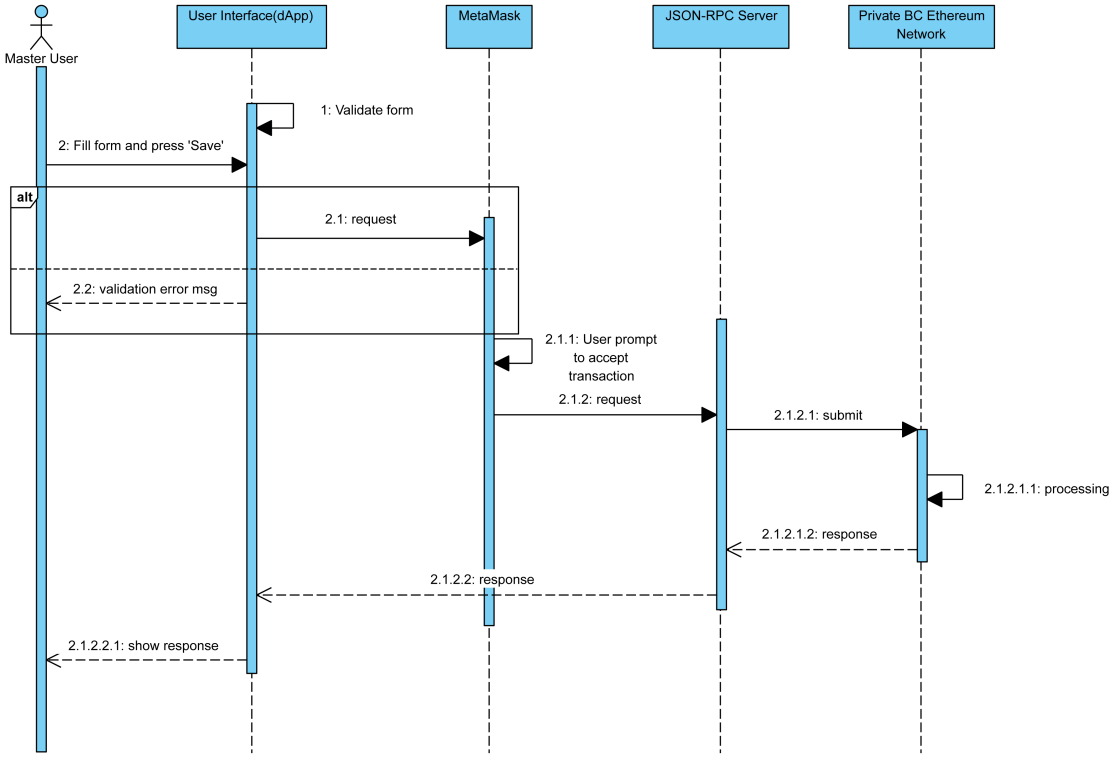
Μόνο οι διαχειριστες μπορούν να ξεκινήσουν μια νέα ψηφοφορία για έναν νέο υποψήφιο. Αυτό σημαίνει ότι οι αιτούντες δεν υποβάλλουν απευθείας αίτηση, αλλά ο διαχειριστής είναι ο ενδιάμεσος. Για να ξεκινήσει μια νέα ψηφοφορία, ο διαχειριστής πρέπει να συμπληρώσει τη φόρμα και να πατήσει αποθήκευση. Σε περίπτωση οποιουδήποτε σφάλματος επικύρωσης, ζητείται από τον χρήστη το σφάλμα και πρέπει να προσπαθήσει ξανά.

3.11.2 Διάγραμμα δραστηριότητας UML



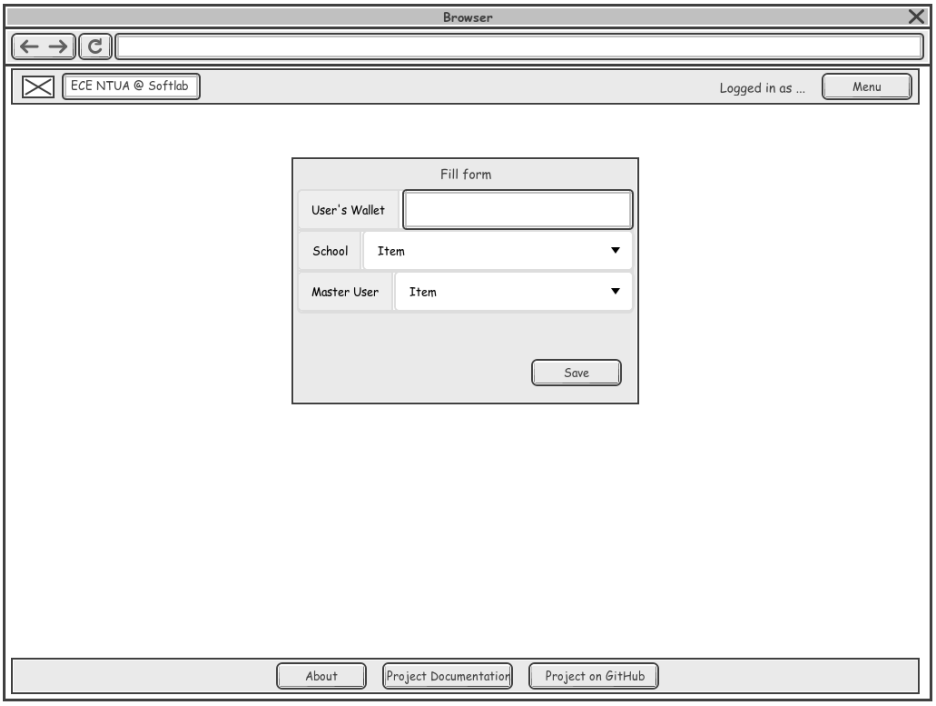
Σχήμα 3.25: *Διάγραμμα δραστηριότητας UML: Έναρξη νέας ψηφοφορίας*

3.11.3 Διάγραμμα ακολουθίας UML



Σχήμα 3.26: *Διάγραμμα ακολουθίας UML: Έναρξη νέας ψηφοφορίας*

3.11.4 Lo-Fi Wireframe



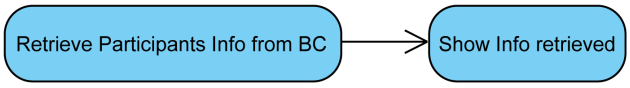
Σχήμα 3.27: *Έναρξη νέας ψηφοφορίας - Lo-Fi Wireframe*

3.12 Περίπτωση Χρήσης 5: Ανάκτηση χρηστών του συστήματος

3.12.1 Περιγραφή

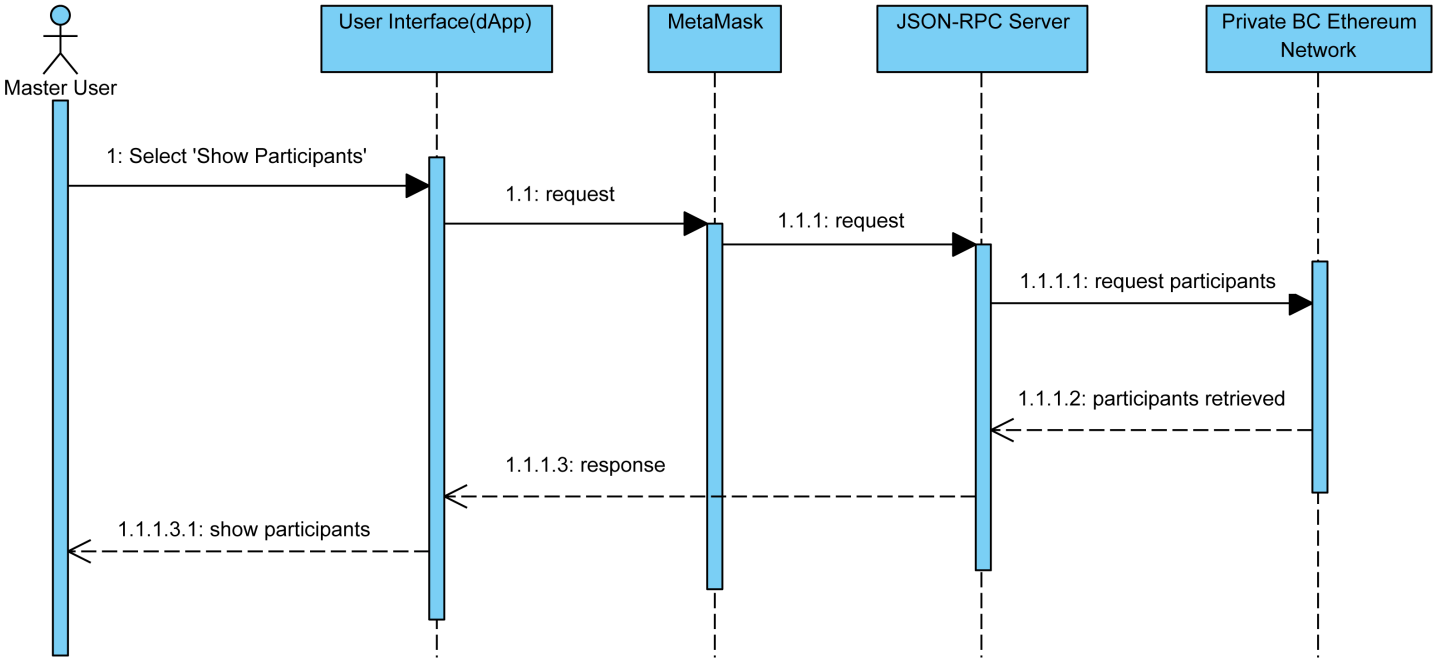
Ένας διαχειριστής μπορεί να ανακτήσει μια λίστα με όλους τους συμμετέχοντες που έχουν πρόσβαση.

3.12.2 Διάγραμμα δραστηριότητας UML



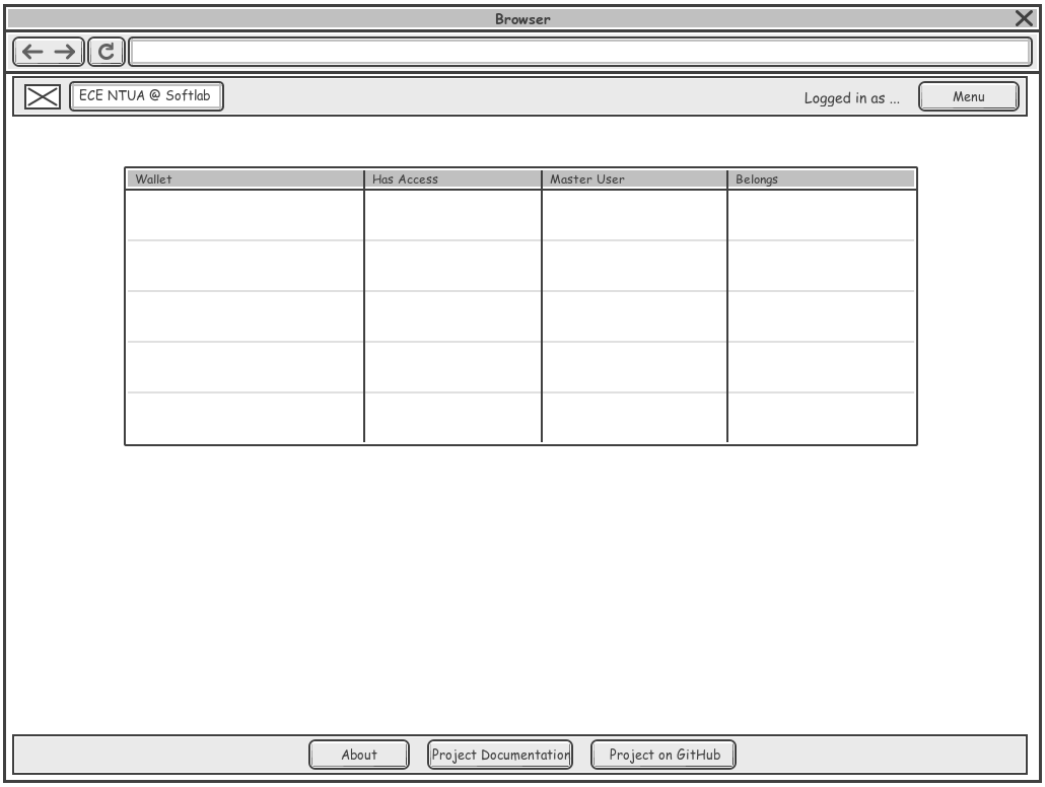
Σχήμα 3.28: *Διάγραμμα δραστηριότητας UML: Ανάκτηση χρηστών του συστήματος*

3.12.3 Διάγραμμα ακολουθίας UML



Σχήμα 3.29: *Διάγραμμα ακολουθίας UML: Ανάκτηση χρηστών του συστήματος*

3.12.4 Lo-Fi Wireframe



Σχήμα 3.30: *Ανάκτηση χρηστών του συστήματος - Lo-Fi Wireframe*

Κεφάλαιο 4

Επίλογος

Στην παρούσα εργασία εξετάστηκε η δημιουργία ενός συστήματος βασισμένου στην τεχνολογία blockchain για τη διαχείριση των βαθμών ενός πανεπιστημίου. Πιο συγκεκριμένα, εστίασε στη δημιουργία ενός ιδιωτικού δικτύου στο οποίο έχουν πρόσβαση μόνο ορισμένες πανεπιστημιακές οντότητες όπως φοιτητές, καθηγητές και γραμματείς. Εφαρμόστηκαν διάφορα χαρακτηριστικά του συστήματος, όπως η εισαγωγή βαθμών στο σύστημα, η προσθήκη νέων μελών στο σύστημα, η ψηφοφορία για το εάν ένα νέο μέλος θα μπορούσε ή όχι να χρησιμοποιήσει το σύστημα και η επικύρωση δεδομένων Blockchain με μια εξωτερική πηγή δεδομένων για την εύρεση τυχόν εναλλαγών δεδομένα. Στην περίπτωσή μας, η εξωτερική πηγή είναι το αρχείο που έστειλε ο καθηγητής στον καταχωρητή και τελικά αποθηκεύτηκε σε διακομιστή. Συνάγεται το συμπέρασμα ότι ένα τέτοιο σύστημα που βασίζεται σε blockchain μπορεί να εφαρμοστεί και να προσφέρει όλα τα κύρια χαρακτηριστικά αυτής της τεχνολογίας σε αυτό το πρόβλημα. Η ακεραιότητα και η ασφάλεια των δεδομένων που τελικά καταλήγουν στο σύστημα παλαιού τύπου μπορεί να επιτευχθεί, καθώς το καθολικό Blockchain μπορεί να λειτουργήσει ως μηχανισμός επικύρωσης μεταξύ αυτών των δύο πηγών δεδομένων.

4.0.1 Ιδιωτικότητα - Κλειστότητα

Η απαίτηση απορρήτου επιτυγχάνεται καθώς το δίκτυο ρυθμίζεται ανάλογα. Επιπλέον, εφαρμόζεται ένας μηχανισμός ψηφοφορίας όπου όλοι οι συμμετέχοντες μπορούν να ψηφίσουν υπέρ ή κατά ενός νέου χρήστη. Αυτό σημαίνει ότι οι χρήστες πρέπει να περάσουν μια διαδικασία ψηφοφορίας για να έχουν πρόσβαση στην αποκεντρωμένη εφαρμογή και στα δεδομένα που είναι αποθηκευμένα στο Blockchain.

4.0.2 Ασφάλεια και Ακεραιότητα

Όπως αναφέρθηκε παραπάνω, η ακεραιότητα και η ασφάλεια προσφέρονται εξ ορισμού με την τεχνολογία Blockchain. Επιπλέον, εφαρμόζεται ένας μηχανισμός επικύρωσης για την επικύρωση μεταξύ του καθολικού του Blockchain και των δεδομένων που υπάρχουν στο σύστημα παλαιού τύπου. Αυτό είναι ένα κρίσιμο χαρακτηριστικό, καθώς μπορεί να ανακαλύψει τυχόν ασυνέπειες των δεδομένων.

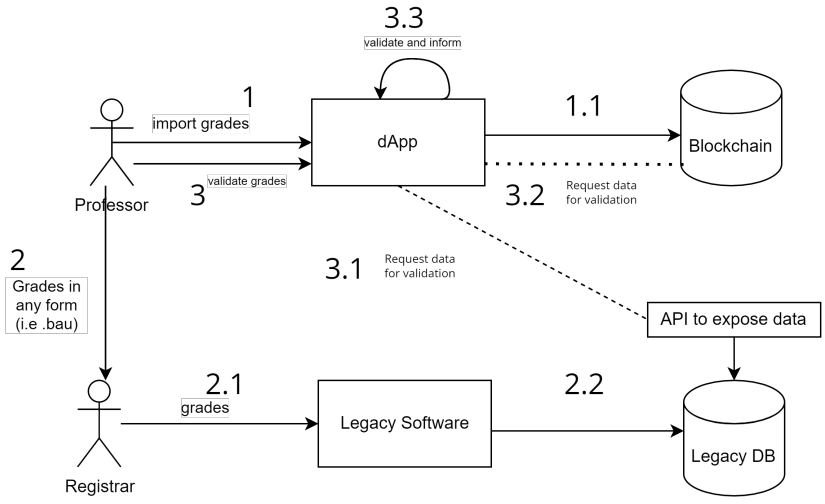
4.1 Αμεσότητα διαδικασιών

Με την τεχνολογία Blockchain αυτό το σύστημα μπορεί να προσφέρει σε κάθε χρήστη αμεσότητα οποιασδήποτε διαδικασίας στο σύστημα παλαιού τύπου απαιτεί χρόνο. Για παράδειγμα, ένας φοιτητής μπορεί να ζητήσει ένα έγγραφο (δηλαδή όλους τους βαθμούς του φοιτητή), αλλά αυτό το αίτημα απαιτεί χρόνο καθώς περιλαμβάνει μερικά βήματα. Πιο συγκεκριμένα, ο γραμματέας πρέπει να δει το αίτημα του φοιτητή, να ετοιμάσει το έγγραφο, να πάρει τις πληροφορίες από το σύστημα παλαιού τύπου και να περιμένει να υπογράψει ο κοσμήτορας. Αντίθετα, όταν το καθολικό είναι αμετάβλητο, αυτή η διαδικασία μπορεί να αποφευχθεί και το έγγραφο μπορεί να εξαχθεί αμέσως.

4.2 Περαιτέρω επεκτάσεις

Αυτή η ενότητα εστιάζει σε προτάσεις χαρακτηριστικών με τις οποίες μπορεί να επεκταθεί αυτό το σύστημα. Επιπλέον, τα χαρακτηριστικά που καλύπτει αυτή η διπλωματική εργασία ως proof-of-concept συζητούνται και εξηγούνται για το πώς πρέπει να εφαρμοστούν σε ένα πραγματικό σύστημα.

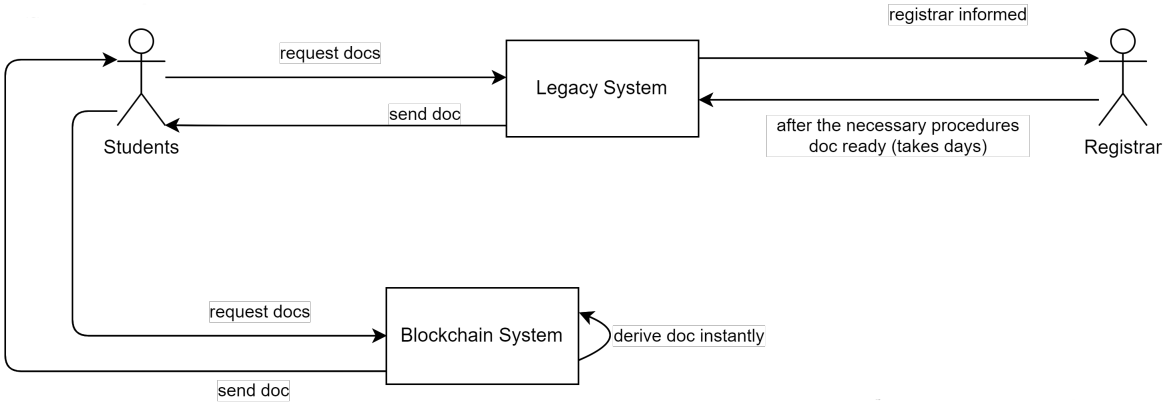
1. Ένα χαρακτηριστικό που καλύπτεται ως proof-of-concept σε αυτή τη διπλωματική είναι η επικύρωση μεταξύ δύο πηγών. Για λόγους επίδειξης, αυτή η διπλωματική επικυρώνει το καθολικό Blockchain με το αρχείο bau που είναι αποθηκευμένο σε έναν διακομιστή. Στο σύστημα παραγωγής, το κατανεμημένο καθολικό θα πρέπει να συγκρίνεται με τα δεδομένα που υπάρχουν στη βάση δεδομένων του παλαιού τύπου συστήματος. Επομένως, για αυτήν τη σύγκριση, θα πρέπει να υπάρχει ένα API για τη συλλογή των απαραίτητων δεδομένων για τη σύγκριση. Επιπλέον, είναι σημαντικό να σημειωθεί ότι το σύστημα Blockchain είναι ευέλικτο. Δεν είναι απαραίτητο να μπορεί να περιέχει αρχεία μόνο σε μορφή BAU. Όποιος κι αν είναι ο σχηματισμός των δεδομένων που εκτίθενται από το API που λαμβάνονται από τη βάση δεδομένων του παλαιού συστήματος, το dApp μπορεί να το υποστηρίξει.



Σχήμα 4.1: *Διαδικασία προσθήκης βαθμολογιών και επικύρωσης*

2. Από το παραπάνω διάγραμμα, είναι σαφές ότι η προτεινόμενη ροή εργασίας είναι ότι οι καθηγητές θα πρέπει να προσθέτουν απευθείας στο καθολικό του blockchain τα δεδομένα αντί να στέλνουν το αρχείο στους γραμματείς. Αυτό θα μπορούσε να προσθέσει περαιτέρω ασφάλεια, καθώς το αρχείο δεν μπορεί να παραβιαστεί καθώς δεν φεύγει ποτέ από την πηγή. Έτσι, οι καθηγητές θα μπορούσαν να προσθέσουν τα δεδομένα και να στείλουν το αρχείο στον γραμματέα ο οποίος θα χρησιμοποιούσε το συγκεκριμένο λογισμικό για να καταχωρήσει τους βαθμούς των μαθητών στο σύστημα παλαιού τύπου. Έτσι, εάν τα δεδομένα που κατέληξαν στο σύστημα παλαιού τύπου παραβιαστούν, οι αλλαγές μπορούν να ανιχνευτούν.

3. Εφόσον το καθολικό κρατά τους βαθμούς των φοιτητών, οι φοιτητές θα μπορούσαν να ανακτήσουν όλες τις πληροφορίες τους από το Blockchain χωρίς να χρειάζεται να ζητήσουν έγγραφα από τον γραμματέα, καθώς το σύστημα μπορεί να τα παράγει απευθείας και τη βεβαιότητα ότι το έγγραφο δεν είναι πλαστό. Επίσης, οι φοιτητές δεν χρειάζεται να περιμένουν τη δημιουργία του εγγράφου επειδή δεν υπάρχουν μεσάζοντες που μπορούν να επιβραδύνουν τη διαδικασία.



Σχήμα 4.2: *Φοιτητές που ζητούν έγγραφο απο το υπάρχων σύστημα vs το blockchain σύστημα*

4. Αντίστοιχα, θα μπορούσε να αναπτυχθεί ένα API ώστε εξωτερικές οντότητες όπως εταιρείες να μπορούν να ζητούν δεδομένα για συγκεκριμένους φοιτητές που θέλουν να προσλάβουν. Αυτή η δυνατότητα μπορεί να διασφαλίσει ότι οι εταιρείες λαμβάνουν τα σωστά δεδομένα, χωρίς εναλλαγές και χωρίς να χρειάζεται ο φοιτητής να επικοινωνήσει με τον γραμματέα της σχολής του για τα έγγραφα. Και πάλι, η διαδικασία είναι παρόμοια με την παραπάνω εικόνα.

5. Ομοίως με τα δύο προηγούμενα σημεία, η διαδικασία των μαθητών θα μπορούσε να αναπαρασταθεί ως ένα έξυπνο συμβόλαιο και όταν οι μαθητές ολοκληρώσουν όλα τα μαθήματά τους θα μπορούσε να δημιουργηθεί ένα NFT [20]. Με αυτόν τον τρόπο, το σύστημα διατηρείται αποκεντρωμένο χωρίς την ανάγκη για ένα κεντρικό σύστημα όπως ένα API. Επιπλέον, θα μπορούσαν να δημιουργούνται διαφορετικά NFT μέσω του συστήματος Blockchain, όπως ένα αναλυτικό έγγραφο όλων των βαθμών των μαθητών ή ένα έγγραφο πιστοποιητικού φοίτησης.

6. Αυτό το συγκεκριμένο σύστημα αποθηκεύει πλήρη αρχεία βαθμών σε μορφή string, η οποία μπορεί να είναι αρκετά κοστοβόρα. Αντίθετα, αυτά τα αρχεία θα μπορούσαν να αποθηκευτούν στο InterPlanetary File System (IPFS) [21] και μόνο ο κατακερματισμός των περιεχομένων των αρχείων θα μπορούσε να αποθηκευτεί στο Blockchain. Φυσικά, πρέπει να αναπτυχθεί ένα ιδιωτικό δίκτυο IPFS για να διασφαλιστεί η ακεραιότητα αυτών των αρχείων και να απαγορεύεται η πρόσβαση σε χρήστες που δεν υποτίθεται ότι θα ανακτήσουν τέτοιες πληροφορίες. Αυτό το κομμάτι της προτεινόμενης λύσης, απαιτεί να μελετηθεί ένα παρόμοιο σύστημα που να μπορεί να αποθηκεύει αρχεία και ταυτόχρονα να τα προστατεύει από τρίτους. Αν και σε ένα τέτοιο ιδιωτικό δίκτυο blockchain το κόστος δεν έχει σημασία, αυτό θα μπορούσε να είναι μια καλύτερη και ευκολότερη λύση για την κλιμάκωση.

Βιβλιογραφία

[1] *Blockchain representation figure*. https://www.paiementor.com/blockchain-explained-application-payments/.

[2] *Types of Blockchain: Public, Private, or Something in Between*. https://www.foley.com/en/insights/publications/2021/08/types-of-blockchain-public-private-between.

[3] *Consensus Mechanisms comparison table*. https://www.sebaversity.swiss/Lectures/what-is-mining/.

[4] *Blockchain Technology Defined*. https://builtin.com/blockchain.

[5] *What is blockchain?* https://www.euromoney.com/learning/blockchain-explained/what-is-blockchain.

[6] *CONSENSUS MECHANISMS*. https://ethereum.org/en/developers/docs/consensus-mechanisms/.

[7] *Proof-of-Authority consensus*. https://apla.readthedocs.io/en/latest/concepts/consensus.html/.

[8] *What Is a Smart Contract and How Does it Work?* https://www.bitdegree.org/crypto/tutorials/what-is-a-smart-contract#what-is-a-smart-contract-what-yoursquore-going-to-find-in-this-guide.

[9] Vitalik Buterin. *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*. 2014.

[10] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008.

[11] *INTRO TO ETHEREUM*. https://ethereum.org/en/developers/docs/intro-to-ethereum/.

[12] *A quick introduction to Bitcoin*. https://www.bitcoin.com/get-started/a-quick-introduction-to-bitcoin/.

[13] Starantzis Dimitrios. *Software architectures for implementing decentralized autonomous organizations (DAO) with blockchain technologies*. Διπλωματική εργασία, SoftLab, National Technical University of Athens, 2022.

[14] *R3 Corda*. https://www.r3.com/trust-technology/.

[15] *Quorum*. https://consensys.net/quorum/.

[16] *Ripple*. https://ripple.com/.

[17] *Hyperledger Foundation*. https://www.hyperledger.org/.

[18] *Hyperledger Fabric*. https://www.hyperledger.org/use/fabric.

[19] *MetaMask*. https://en.wikipedia.org/wiki/MetaMask#cite_note-CNET_2018-6.

[20] *What is an NFT*. https://www.forbes.com/advisor/investing/cryptocurrency/nft-non-fungible-token/,.

[21] *IPFS*. https://ipfs.io/.

Συντομογραφίες - Αρκτικόλεξα - Ακρωνύμια

βλπ	βλέπε
κ.λπ.	και λοιπά
κ.ο.κ	και ούτω καθεξής
ΤΕΙ	Τεχνολογικό Εκπαιδευτικό Ίδρυμα
BC	Blockchain
DoS	Denial-of-Service
dApp	Decentralized Application
PoW	Proof of Work
PoS	Proof of Stake
PoA	Proof of Authority
CA	Certification Authority
DLT	Distributed Ledger Technology
LoFi	Low Fidelity
OS	Operating System
CLI	Command Line Interface
JSON	JavaScript Object Notation
CSV	Comma-Seperated Values
API	Application Programming Interface
NPM	Node Package Manager
HOC	Higher Order Component
ABI	Application Binary Interface
EVM	Ethereum Virtual Machine
NFT	Non-Fungible Token
RPC	Remote Procedural Call
UML	Unified Modeling Language
IPFS	InterPlanetary File System

Απόδοση ξενόγλωσσων όρων

Απόδοση	Ξενόγλωσσος όρος
Έξυπνο συμβόλαιο	Smart contract
Κόμβοι	Nodes
Ιδιωτικό με άδεια δίκτυο	Private permissioned network
Έμπιστος τρίτος φορέας	Trusted third party
Παλαιού τύπου σύστημα	Legacy system
Αλγόριθμος συναίνεσης	Consensus algorithm
Μπλοκ	Block
Κεφαλίδα κατακερματισμού	Header hash
Κατακερματισμός	Hash
Αποκέντρωση	Decentralization
Κεντροποίηση	Centralization
Ιδιωτικότητα	Privacy
Επικύρωση	Validation
Ακεραιότητα	Integrity
Διακομιστής	Server
Ακεραιότητα	Integrity
Ροή εργασίας υψηλού επιπέδου	High level work flow
Κρυπτονομίσματα	Cryptocurrencies
Κοινοπραξία	Consortium
Ιδιωτικά δίκτυα	Private networks
Υβριδικά δίκτυα	Hybrid networks
Δημόσια δίκτυα	Public networks
Κακόβουλος χρήστης	Malicious user
Παζλ	Puzzle
Κρυπτογραφική σύνδεση	Cryptographic link
Υπολογιστική Ισχύς	Computational power
Επικυρωτής	Validator
Καθολικό	Ledger
Αλυσίδα	Chain
Εξορύκτης	Miner
Εξόρυξη	Mining
Διαφανής	Transparent
Χωρίς άδεια	Permissionless
Με άδεια	Permissioned

Χωρίς άδεια	Permissionless
Κλιμακωσιμότητα	Scalability
Αποκεντρωμένες Εφαρμογές	Decentralized Applications
Διεπαφή χρήστη	User interface
Εργαλεία	Tools