

POLITECNICO DI TORINO

Corso di Laurea Magistrale in Computer Engineering

Master Degree Thesis

Blockchain based storage of students career



POLITECNICO
DI TORINO

Supervisor

prof. Maurizio Morisio

Co-supervisor:

prof. Luca Ardito

Candidate

Bakhtiyor Yokubov

ID: 250755

December 2018

Acknowledgments

First of all I would like to thank my supervisor Prof. Maurizio Morisio. The special thanks to co-supervisor Prof. Luca Ardito for his help and advices. He guided me in the right direction whenever he thought I needed it.

I would like to thank rector Prof. Kongrad Sharipov, first pro-rector Prof. Jamshid Inoyatkhodjaev and pro-rector of academic affairs Prof. Igor Stievano at Turin polytechnic university in Tashkent.

Heartfelt thanks go to my parents, my wife and my friends for their support and understanding.

Abstract

Blockchain, the rapid developing technology behind Bitcoin, is increasingly becoming popular. Blockchain is a distributed ledger technology that distributes digital transactions peer-to-peer to a decentralized network of nodes that verify the transactions and keep a cryptographic secured copy of the entire history of transactions. The network automatically reaches consensus about the correct history of records, which makes the database transparent and immutable. This consensus role makes it possible to take away the third party in certain processes, such as the bank or the notary. Blockchain also enables digital payments and smart contracts. Smart contracts are digital contracts that can be executed automatically by the blockchain. This enables digital registration of for example identity, birth certificates and votes. But smart contracts have many more automation applications that can be coded in computer code, which has the potential of making many processes in both the public as the private sector more efficient and less costly. Governmental services are especially applicable for blockchain, as they could become more efficient and can even be made obsolete in some cases.

In this thesis, we review solutions using blockchain technology, identify key provenance features and implement a solution for student academic career platform on Ethereum blockchain. In this platform all the transactions represents the university exams passed by the students and their grades.

Transactions can only be created by authorized personnel. Students can read their career history, administrative staff and professors can record grades. So, nobody can be able to change grades after they are rated.

Keywords: Student academic career, grade, blockchain, ethereum, smart contract, ERC20, web3, truffle, metamask

Contents

1	Introduction	8
1.1	Problem Statement	8
1.2	Contribution	10
1.3	Technology used	10
1.3.1	Blockchain	10
1.3.2	Ethereum	16
1.3.3	Smart Contracts	20
1.3.4	MetaMask	21
1.3.5	Truffle	23
1.3.6	Node Packet Manager (NPM)	23
1.3.7	Decentralized Applications DApps	24
1.4	Thesis Structure	29
2	Related Works	31
2.1	MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain	32
2.2	Blockchain Governance	35
2.3	Blockchain for IoT Security and Privacy: The Case Study of a Smart Home	37

2.4	Towards Secure E-Voting Using Ethereum Blockchain	40
2.5	Blockchain Based Access Control	41
2.6	Academic Certificates System on the Blockchain	44
2.7	Student Diploma on Bitcoin Blockchain	46
2.8	Certification Diplomas on the Blockchain	47
2.9	Blockchain Based Professional Networking and Recruiting Platform	48
2.10	EduCTX: A Blockchain-Based Higher Education Credit Platform	50
3	Design and Implementation	57
3.1	Requirements	57
3.2	The Proposed Platform	57
3.2.1	Private Blockchain Setup	59
3.2.2	Install and configure Truffle	76
3.2.3	One-click Login with Blockchain	77
3.2.4	Student's registration	84
3.2.5	Professor's registration	85
3.2.6	Student's course completion	85
3.2.7	Implementation of Administrator Account	85
3.2.8	Implementation of Professor Account	88
3.2.9	Implementation of Student Account	89
3.2.10	Front-End implementation	90
4	Experimentation in TTPU	99

5 Conclusion	103
Bibliography	105

Chapter 1

Introduction

1.1 Problem Statement

Data management is an important aspect in the development and growth of every organization. Particularly, education institutions need a good data management system in order to function properly each student' academic record. A database management system (DBMS) is an aggregate of data, hardware, software, and users that help an enterprise manage its operational data. One of the primary functions of a DBMS is to provide efficient and reliable ways of data restitution to many users. DBMS as software structured to help in maintaining and utilizing large data collections. DBMS ensures a symmetric technique of creating, updating, retrieving data in a database. It enables end user and application programmers to share data and enable data to be shared among multiple application rather than propagated and accumulate in new files for every new application.

Traditional database system uses the client-server architecture for accessing information. In this centralized server, information can be modified by

an user. Designated authority takes all control of the database. Before accessing to the database, client's credentials should be authenticated. The responsibility for administration of the database is lies on the authority. If authority's security is jeopardized, the data can be altered, or even deleted. In a database system, data can be easily modified or deleted. In education institutions data record of grades should be unchangeable. Once a student is graded by teachers on a subject, grade should be saved on database system and no one are allowed to modify and delete.

Let's take as an example of academic career system in Turin polytechnic university in Tashkent. For many years Turin polytechnic university in Tashkent (TTPU) uses Moodle electronic educational platform. In this platform, from one side a teacher upload pdf or excel files with students grade and from the other side, students download those files in order to know about their grades on different subjects. There are several disadvantages using this platform:

- **Centralized entry point.** Moodle platform has centralized entry point for both teachers and students. Login and password needed to authenticate to Moodle educational platform. Nevertheless, we can consider that login and password authentication is unsafe for some reasons. Bad guys can steal login credentials.
- **Difficulty.** Moodle educational platform is a technological tool which could be difficult for some students.
- **Client-Server architecture.** Moodle database as traditional database system uses the client-server architecture for accessing information.

1.2 Contribution

We propose a Blockchain based storage of students career platform. This system is flexible, secure and resilient because of their storage capacity and resource sharing on a global scale. The platform transfers grading system from the analog and physical world into a globally efficient, more simplified, ever-present version based on the blockchain technology.

1.3 Technology used

This section aims to elaborate on the technologies used in developing the proposed project solution for students career.

1.3.1 Blockchain

Before describing the Blockchain we need to understand about a distributed ledger and how it is different from the concept of the database.

A distributed ledger is simplest form of the database which is held and upgraded by each participant (node) in a network. Every node has ledger record independently. Each node processes every transactions, with its own derivations and voting on those derivations. All nodes carry identical copy of the ledger[1].

The one of the main differences between a ledger and a usual database is that in ledger we only can add new transaction while in usual database we can do append, modification and deletion of record.

Blockchain is a shared database which consist of a ledger of transactions and all committed transactions are stored in a list of blocks in a secure and unaltered way. Figure 1.1 presents an example of a blockchain. Each block contains a cryptographic hash value of previous block, a time-stamp, and a

nonce - which is a random number for verifying the hash. This design ensures that any change on the block will immediately change the respective hash value. The initial block is called *genesis block* which has no parent block. For a block to be added on the chain, the network through a consensus mechanism has to agree on the validity of the transaction and the block. In every 10 minutes miners add new blocks to the blockchain in a liner, chronological order. When miners connects to the blockchain network, all the nodes will be downloaded automatically [2]. Data ownership is tracked by the ledgers of simple blockchains. Unlike a centralized database, everyone has a copy of the ledger and can verify each other's accounts. Each connected device with a copy of the ledger is called a node. Decentralization, persistency, anonymity and auditability are the main characteristics of the blockchain technology.[3]

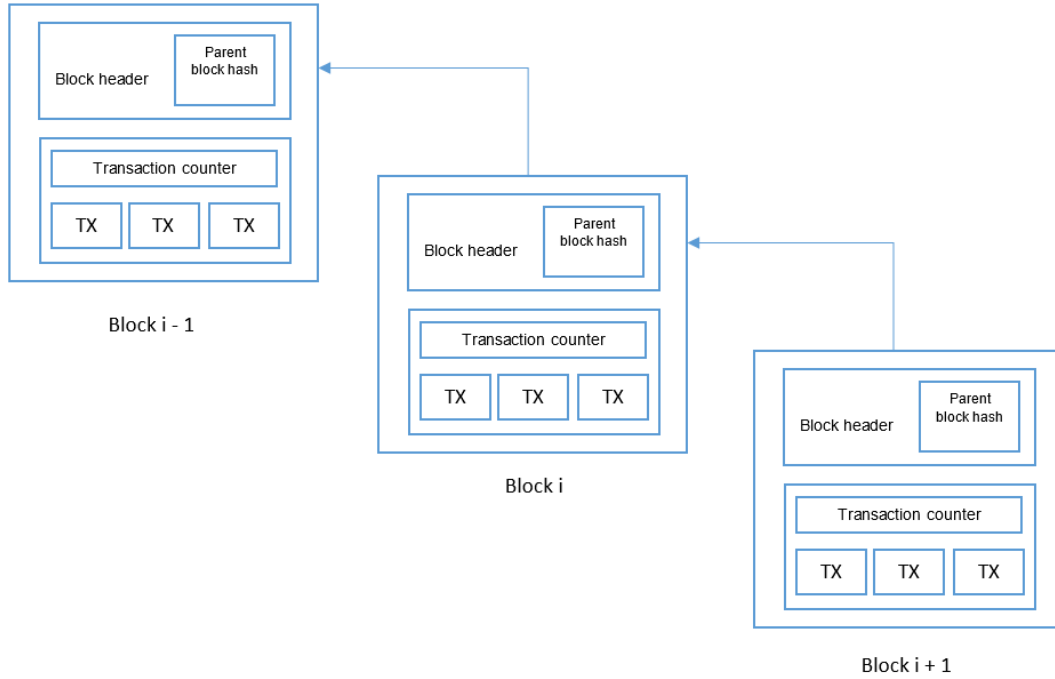


Figure 1.1: An example of Blockchain which consists of continuous sequence of blocks

Another definition could be, Blockchain is a timestamped, ordered and immutable list of all transactions.

There are four key distinguishing factors of the blockchain.

1. Through the use of smart contracts along with having no single point of failure, it will be possible to transfer things of value, such as, property, vehicles and so much more securely. Everything will be digitally secure by the power of cryptography on the blockchain.
2. It cuts out any middleman in the process. At the core of this technology is the peer to peer system which secured by the cryptography. Hence, they have the power to remove lawyers, accountants, stay agents and other professionals from the middle of the process. It will save us money.
3. It allows to deal transactions get completed far quicker. It will streamline time consuming tasks.
4. We can have far more increased capacity. Due to the peer to peer technology, can actually increase the capacity of the whole network.

Block Structure

Block is a container for data structure which has transactions included to the public ledger. The block structure has different fields, which are:

- **Block size:** a field that has the block size (usually 4 bytes)
- **Block Header:** a field that has the bellow header fields (usually 80 bytes)
- **Transaction Counter:** a field that maintains total number of transactions in the block

- **Transactions:** length of variable, has transactions recorded in the current block.

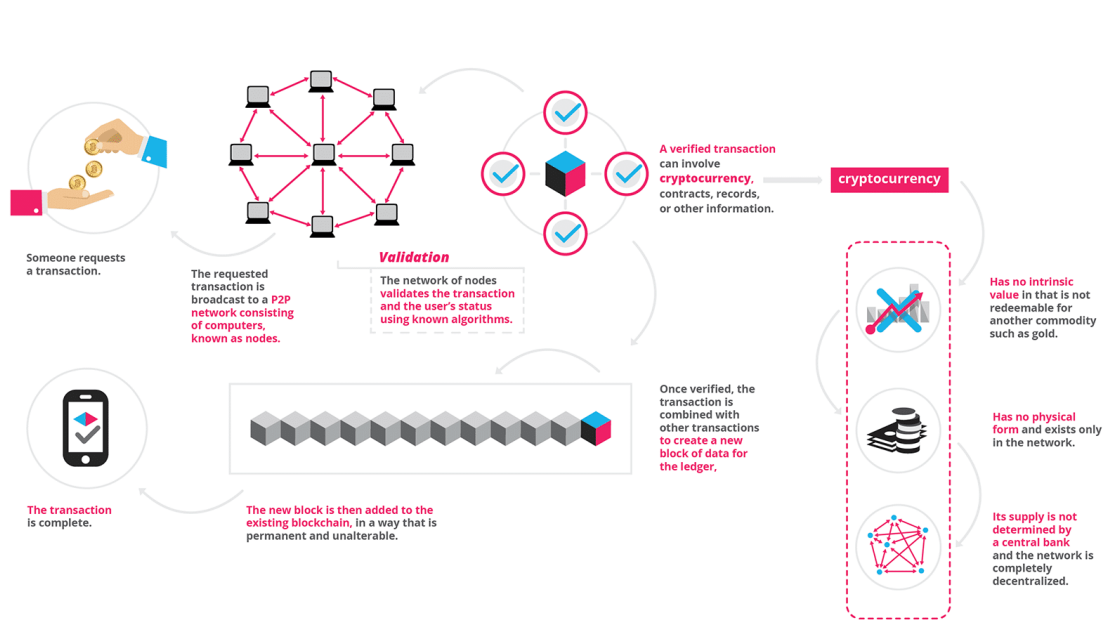


Figure 1.2: How a blockchain works?

The Genesis Block

As far as blocks in the Blockchain reference to the previous block, the blockchain has first block which is genesis block. It is assigned number 0 to this block. Because of no previous block exists for this block, it is the only block which doesn't reference to the previous one. It is possible two nodes to pair each other if and only if they are in the same network and have the same genesis block.

Mining

Mining is a resource intensive act in order to add a new block to the Blockchain. Each transactions which is inside blocks is confirmed using the

mining process. A block is created every 15 minutes and miners are prized with new coins.

There are three types of blockchain systems available: public blockchain, private blockchain and consortium blockchain[4]. In public blockchain everyone can see all records and could take part in the consensus process. In consortium blockchain, only a selected group nodes participate in the consensus process. In private blockchain only nodes in one organization can take part in the consensus process.

Nodes in Blockchain

Blockchain consists of network nodes. Nodes will automatically download copy of the blockchain upon joining to the blockchain network [5] (Figure 1.3).

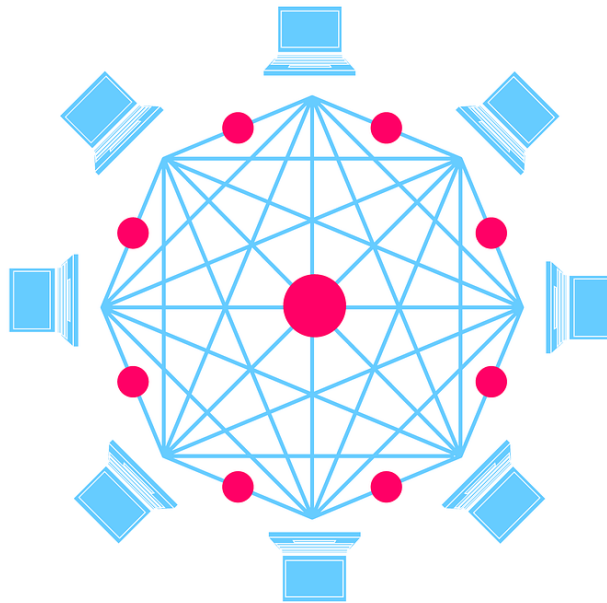


Figure 1.3

Joining all nodes will create powerful network different than usual one. In

this scenario any node can be an "administrator" of the blockchain and can join network by their wish. Any node has a chance to win Bitcoins.

"Bitcoin has the same character a fax machine had. A single fax machine is a doorstop. The world where everyone has a fax machine is an immensely valuable thing." Larry Summers [6]

Blockchain uses cryptography in order to allow users to trust each other, thereby central authority or intermediary is eliminated to perform peer to peer transactions over the Internet. It is possible to ensure that transactions are made by the legitimate owners. The blockchain ensures auditability and integrity of the whole network. Once block added to the ledger, it will not be possible to modify or delete.

Let us share a quote of bitcoin specialist which sums up the blockchain. He will compare blockchain as Google Docs. So, if we understand how the Google Docs works we will understand how blockchain works. So, here is the quote:

Blockchain as Google Docs "The traditional way of sharing documents with collaboration is to send a Microsoft Word document to another recipient, and ask them to make revisions to it. The problem with that scenario is that you need to wait until receiving a return copy before you can see or make other changes because you are locked out of editing it until the other person is done with it. That's how databases work today. Two owners can't be messing with the same record at once. That's how banks maintain money balances and transfers; they briefly lock access (or decrease the balance) while they make a transfer, then update the other side, then re-open access (or update again). With Google Docs (or Google Sheets), both parties have access to the same document at the same time, and the single version of that document is always visible to both of them. It is like a shared ledger,

but it is a shared document. The distributed part comes into play when sharing involves a number of people. Imagine the number of legal documents that should be used that way. Instead of passing them to each other, losing track of versions, and not being in sync with the other version, why can't all business documents become shared instead of transferred back and forth? So many types of legal contract would be ideal for that kind of work-flow. You don't need a Blockchain to share documents, but the shared documents analogy is a powerful one." William Mougayar[7]

1.3.2 Ethereum

Ethereum is an open generic blockchain platform developed by the Ethereum Foundation, a Swiss nonprofit organization. In Ethereum environment anybody can do transactions without any permission on both public and private network.

A transaction is a valid state change that is identified by a cryptographic hash value or a transaction ID (TxID). A valid state change represents a move from an original state to final state. Invalid state changes which are often may include debiting an account balance without its corresponding credit. Final state can include information such as account balances, data representing information to the physical world etc. Transactions are grouped into blocks which are then chained together using a cryptographic hash as means of reference [8].

Every transaction in Ethereum network is digitally signed by the sender. The signing is made possible by a mathematical function that takes in a hashed value of some data object and a private key. To verify a signature, a reverse function takes a public key and the signature, then it compares it against the hashed value of the data object.

Ethereum allows to run programs in its trusted environment. This contrasts with the Bitcoin blockchain, which only allows you to manage cryptocurrency. Ethereum has a virtual machine, called the Ethereum Virtual Machine (EVM) which allows to verify the code executed on the blockchain, providing guarantees it will be run the same way on everyone's machine. This code is contained in "smart contracts". Ethereum tracks account balances and maintains the state of the EVM on the blockchain. Smart contracts are processed to verify the integrity of the contracts and their outputs by all nodes [9].

Client	Language	Developers	Latest release
go-ethereum	Go	Ethereum Foundation	go-ethereum-v1.4.18
Parity	Rust	Ethcore	Parity-v1.4.0
cpp-ethereum	C++	Ethereum Foundation	cpp-ethereum-v1.3.0
pyethapp	Python	Ethereum Foundation	pyethapp-v1.5.0
ethereumjs-lib	Javascript	Ethereum Foundation	ethereumjs-lib-v3.0.0
Ethereum(J)	Java	<ether.camp>	ethereumJ-v1.3.1
ruby-ethereum	Ruby	Jan Xie	ruby-ethereum-v0.9.6
ethereumH	Haskell	BlockApps	no Homestead release yet

Figure 1.4: List of Ethereum clients

In order to follow all the requirement we can create Ethereum blockchain environment which is more useful for developing Decentralized Applications (DApp) rather than Bitcoin which is only intended to validate coinage. So, Ethereum blockchain made use cases which will help to create DApps.

Ethereum Clients

An Ethereum client is a node that implements the Ethereum Protocol which gives access to the blockchain and can do mining. There are different nodes available. First nodes were written in three languages. Go (Go Ethereum - Geth) is most popular node and we will use this node in our platform. Then there are implementation on C++ (Ethereum-cpp) and Python. Now there is also new node which calls Parity and it is written in Rust (Figure 1.4). And there are some other clients like MetaMask which is a plugin for MetaMask. MetaMask is not a real node, but, it is more like a bridge between browser and blockchain. In our project we will use MetaMask for more cases.

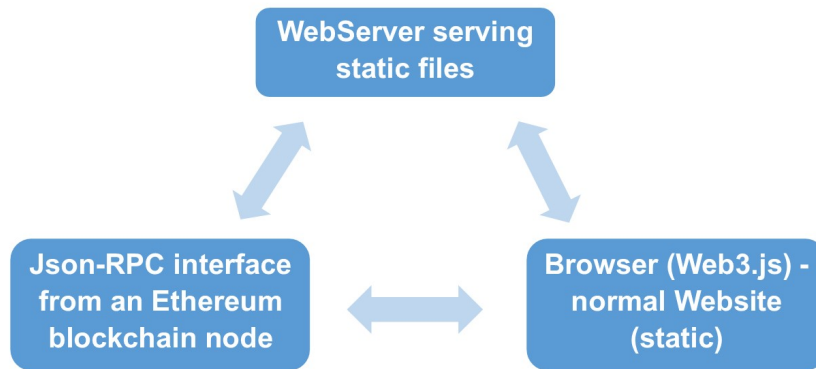


Figure 1.5

In order to browser speaks to the blockchain we need web3 which is collection of libraries which allow you to interact with a local or remote ethereum node, using a HTTP or IPC connection. Web3 is a javascript library it abstracts the Json-RPC from the blockchain node (in our case it is Geth).

Let's see how this visually works. WebServer is serving the static files. The browser is loading the static files from the WebServer, for example, also Web3.js file. This the normal website. Then if we want to interact with the blockchain we connect Json-RPC Interface from an Ethereum Blockchain Node (Figure 1.5).

Ethereum Transactions

Ethereum transaction is defined sending ether from one account to another one in signed data package, for invoking methods of a contract or to deploy a new one.

There could be different transaction types depending on the output such as message call transactions, where a message passed from one account to another one, or contract creation transactions that will create a new contract.

Transaction message contains the following information:

- Message recipient
- Sender signature
- The amount of ether for transmission
- Limit of the gas
- Price of the gas

Gas

For each transaction to be executed that takes place on the blockchain we need Gas in order to define computational cost.

In the real world example could be a car that travels from city A to city B. In order to do this, the owner needs the fuel (gas) to travel.

In a similar manner, on the blockchain, the node must have an Ether balance and pay the gas that is needed for execution the transaction. Gas is measured to be a unit for computational steps. The gas limit is the maximum amount of gas that is taken in order for transaction execution. For including transaction, miners collect the fee.

1.3.3 Smart Contracts

An attractive attribute of blockchain technology is smart contracts. A smart contract runs on the top of the blockchain. Smart contracts help us exchange money, property, shares, or anything of value in a transparent way. It avoids the middleman services [10].

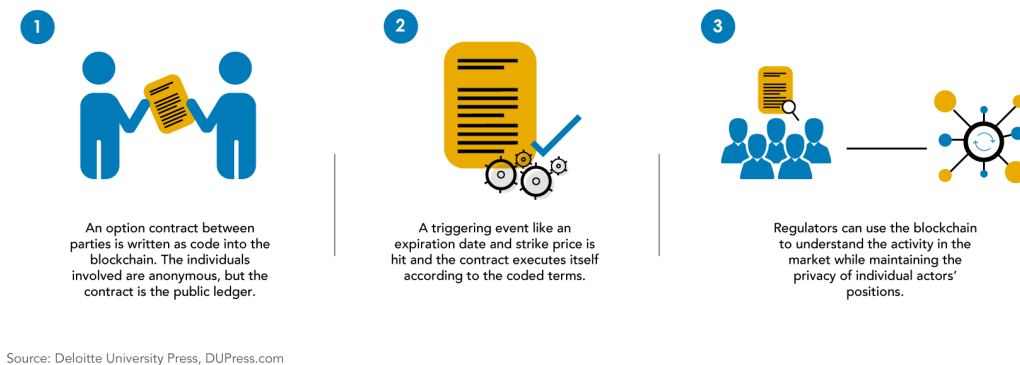


Figure 1.6: Smart Contract Execution

Smart Contracts which is responsible for transferring something of value,

such as, property in a secure way. So, Smart Contracts makes whole process super efficient eliminating expensive and time consuming middle party. There are seven reasons why smart contracts is more useful: autonomy, trust, backups, speed, saves money, accuracy. We can use smart contracts for all sort of situations that range from financial derivatives to insurance premiums, breach contracts, property law, credit enforcement, financial services, legal processes, crowd funding agreements and so on. [Figure 1.6](#)

Smart contracts use the concept of gas to control the consumption of resources. For a transaction to be created, currency in the form of ERC20 tokens must be used to purchase gas. If the gas runs out before the transaction reaches an ordinary stopping point, it is treated as an exception. At this point, the state is reverted as though the transaction had no effect, but the Ether used to purchase the gas is not refunded. When one contract sends a message to another, the sender can offer only a portion of its available gas to the recipient. If the recipient runs out of gas, control returns to the sender who can use its remaining gas to handle the exception and tidy up[\[11\]](#).

Smart Contracts are written in a Solidity programming language. Solidity syntax similar to JavaScript language which supports inheritance, libraries and complex user-defined types.

Solidity is a high-level language used for creation of smart contracts. The Solidity integration of C++, Python and JavaScript languages and it targets the Ethereum Virtual Machine (EVM)[\[12\]](#).

1.3.4 MetaMask

MetaMask is a plugin for different browsers which is a bridge that allows to visit to Ethereum wallet of each user. It allows to run Ethereum dApps right in the browser without running an Ethereum node[\[13\]](#). MetaMask will

help retrieve data from blockchain and letting users securely sign and manage transactions on blockchain. MetaMask supports different networks. We can connect to different networks or We can connect to localhost rpc node or We can connect to our custom RPC. MetaMask in the background connects to company called Infura. Infura offers scalable Blockchain infrastructure, Blockchain as a service. Currently, they are serving 500 Million requests per day. Let have a look how the transactions works visually on MetaMask. Figure 1.7

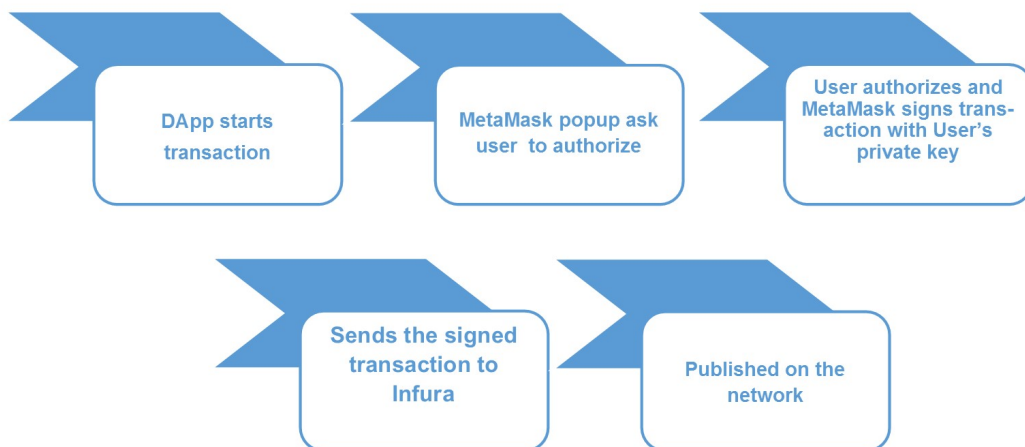


Figure 1.7

MetaMask carries data and transactions information of the Ethereum blockchain to a browser. Example could be shopping portal like Amazon where we do a purchase online and we would like to make a payment

with Ethereum. In this scenario, we can broadcast payment details on the Ethereum blockchain and from other side, Amazon will verify it and after that process our order. Here our PC and Amazon's work as a node on the Ethereum blockchain where they carry out different activities like broadcasting the transaction details.

First the DApp starts the transaction. Then MetaMask will have a popup the asks the user to authorize his transaction. After that the user clicks the authorize button and the MetaMask signs the transaction with the users private keys that is stored on the background in the key store. Then the transaction is sent to the Infure server as a sign transaction. After that it is published on the network.

1.3.5 Truffle

For development environment we used the Truffle. Truffle makes everything easy for developers. Truffle has built in smart contract compilation which does the linking, the deployment and the binary management for us. It has automated testing framework for rapid development. It has its own migrations framework for deployments and network management for deploying different networks (testnet, mainnet, private networks). Truffle comes with package management. For example, Ethereum Packet Manager (EthPM) or Node Packet Manager (NPM) and much more.

1.3.6 Node Packet Manager (NPM)

NPM opens all the doors for JavaScript programmers. It is considered the largest software registry. Each week there are about 3 billion downloads. There are 600,000 packages in the registry of npm. With the help of NPM packages many organizations and private developments can manage their

development.

There are three components of npm:

- the website
- the Command Line Interface (CLI)
- the registry

We can use website in order to reveal new packages, install accounts and manage other points of the npm.

We can interact with npm using CLI on a terminal.

The registry is a huge Javascript database and the information surrounding it.

We can use npm to following cases:

- adjustment of packages to our applications
- for downloading standalone tools that we can use
- using npx we can run packages without downloading them
- we can share our code with npm user as we want
- we can put limitation for concrete users
- applications will be update automatically...

1.3.7 Decentralized Applications DApps

Nowadays several Internet based applications is on centralized architecture. In order to use service of the server, the hosts (end user) access to the server which is owned by a company or person. So in this architecture,

developers creates centralized applications and users uses them. It is difficult to build certain types of applications due to some problem with this centralized architecture such as, transparency, single point of failure and so on. Taking into account these issues, it is more useful to use decentralized architecture for designing and implementing applications. The application which is designed using decentralized architecture called a DApp.

The back-end system of the DApp runs on a peer-to-peer network. Being that, nobody has full control over the DApp.

The DApps also adopt the features of the distributed network. In this case the developers need to solve the problem. System overuses in terms of altering the application data or sending and sharing improper information to others has to be treated with the for mentioned way [14].

The following standards should be considered for application to be considered as a DApp:

- The application need to be open-source.
- The application need to function autonomously.
- All the data and records of the application need to be stored in a public blockchain.
- The application need to use cryptographic token for access to the application.
- The tokens of the application need to be generated using cryptographic standard algorithm.

Classification of DApps

There are three type of classification of DApps available which is based on the fact whether the DApps have their own blockchain or they use the

blockchain of another DApp. These are followings:

- **TYPE I:** DApps have their own blockchain. Bitcoin and Litecoin are the most famous type I DApps.
- **TYPE II:** DApps use DApp blockchain type I. Type II: DApps are protocols and uses tokens which are necessary for their functionality. One of the example for type II is the Omni Protocol that was known as Mastercoin.
- **TYPE III:** DApps use DApp blockchain type II. At the same manner DApps of the type III are protocols and have uses tokens which are necessary for their functionality. Example for type III could be the SAFE Network that uses the Omni protocol that will be used to gain distributed file storage. [15]

Advantages and Disadvantages of DApps

Like any kind of applications, the DApps have some advantages and disadvantages that we should take into consideration. Here are some the advantages of DApps:

- **Fault tolerance:** There is no single point of failure, as far as they are distributed by default.
- **Prevent violation of net censorship:** As there is no central authority on which government can enforce to remove content, net censorship can be prevented. Since DApps are not accessed via a specific IP address or domain, governments cannot block the domain and IP address of DApps. Government can shutdown of the nodes, tracking the node IP address, but as far as number of nodes is huge and the nodes distributed between different countries, shutting down the whole application is impossible.

- **Trust:** As far as there is no control by a single authority, it is simple for users to trust the application.

Here are some disadvantages of DApps:

- **Fixing / Updating:** It is very difficult task to fix/update the node software for each peer.
- **Verification:** While developing applications that require user identity, it is difficult to issue the verification since there is no central authority.
- **Build difficulty:** Through the complexity of the protocols and achievement of the acceptable consensus, building is difficult.
- **Dependencies:** Applications independently get and store of third-part APIs. DApp can be dependent on other DApps, but shouldn't depend on centralized application APIs. [\[14\]](#)

User Identity

One of the main privilege of DApps is the it guarantees user anonymity. From other hand, it is difficult to verify user identity, as far as there is no central authority in a DApp. In the centralized architecture, human verify user identity by requesting the user to submit certain piece of information. This term is also know as KYC. While moving to the decentralized architecture, it is critical to address this issue without adding more complexity. All the participating sides will have to come to an convention on the accountability of the identification verification measure in a unreliable method. As far as majority of the directives on KYC are issued by governments, there is no way to accomplish that issue.

There are several types of digital identities. A digital certificate is the most popular and recommended form of digital identity. A digital certificate

is used to prove ownership of a public key. A digital certificate also called a public key certificate or identity certificate. Usually, a user has a public key, private key and digital certificate. The private key is a secret and user has to keep it secret, while user can share public key with anyone. The digital certificate holds the public key and information about owner.

The only possible way is to verify user identity manually by an authorized person of the company that provides the client.

KYC-Chain is a DApp that is used for identity management system. On this system, KYC-Chain employs Ethereum and works via the use of trusted gatekeepers, who can be individual or legal entity allowed by law to authenticate KYC documents.

The user's ID is checked and authenticated through the KYC chain platform by the trusted gatekeeper. The files will be saved on a distributed database where later be retrieved by the trusted gatekeeper or the use in order to demonstrate that his ID is original. [16]

User accounts in DApps

As far as majority applications need functionality of user accounts and data related to an account should be editable by the account owner only. The user have to sign with his/her/ private key, in order to make a change to the account's information. The most important thing in this approach is the safe storage of the users private keys.

A software named wallet is used in order to save users keys safely. The wallet can be downloaded and installed and user can create one or several accounts. There are several types of wallets available such as software wallets, online wallets, hardware wallets and mobile wallets.

1.4 Thesis Structure

The thesis is organized as follows:

- **Chapter 1** provides statement of the problem for traditional database system and used technologies to develop Blockchain based students career.
- **Chapter 2** analysis literature review of different projects using Blockchain system and makes comparison.
- **Chapter 3** introduces implementation of Blockchain based academic career system. It will focus for each details of the project
- **Chapter 4** covers real test case system which experimented in TTPU using real information of student grades.
- **Conclusion** summarizes the results, evaluates the benefits of the proposed solution and mentions further implementations of the project.

Chapter 2

Related Works

One of the main aims of Blockchain technology is creating decentralized environment where the transactions and data controlled without third party. Due to advantages in distributed data storage and the possibility of audit trails, it is made use of in different field.

In this section we will discuss several different examples of projects using the blockchain system, including education system.

Different approaches have been represented in the domain of electronic health records (EHR)[\[17\]](#)[\[18\]](#). EHR is a information model which integrates multiple healthcare providers. One of the approach where blockchain used in healthcare is "MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain" [\[19\]](#).

2.1 MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain

By authors point of view, the current methods for managing and saving medical records is insufficient. They implements more efficient method than previous methods with using blockchain system which provides data provenance auditing, and control for shared medical data in cloud from big data. Using this blockchain based method, it will be possible to share electronic medical records between untrusted group. In this design, with the help of smart contracts and an access control mechanisms it will be possible to effectively trace the behavior of the data, revoke access to violated rules and permissions on data. By realizing the given model, cloud service providers will be able to securely get data provenance and auditing among other cloud service providers.

In this method of sharing data between untrusted parties', the processing and consensus nodes are responsible for broadcasting blocks into the blockchain network after processing request. Requester identity is used to identify threads in the blockchain.

The framework safety is done by executing specific tasks using cryptographic keys. Using the blockchain based data sharing they achieved confidentiality of the transaction through insecure channels. The following keys they used in order to achieve security:

- Requester private key which is created by a requester and used to make signature for request.
- Requester public key is created by a requester and used to verify the requesters identity for data access. In order to encrypt a package that

authenticator sends to requester.

- Authenticator contract key pair is created by authenticator and affixed to smart contract in a package.

For a requester who would like to access to sets of records from a owner of data, the requester generates private and public key where it will store private key and shares the public key with owner of the data. The data owner encrypts the package with authenticator contract key and share the package with the requester. From other side, the requester decrypts the package and uses the information. The interfaces merged internally and externally by triggers.

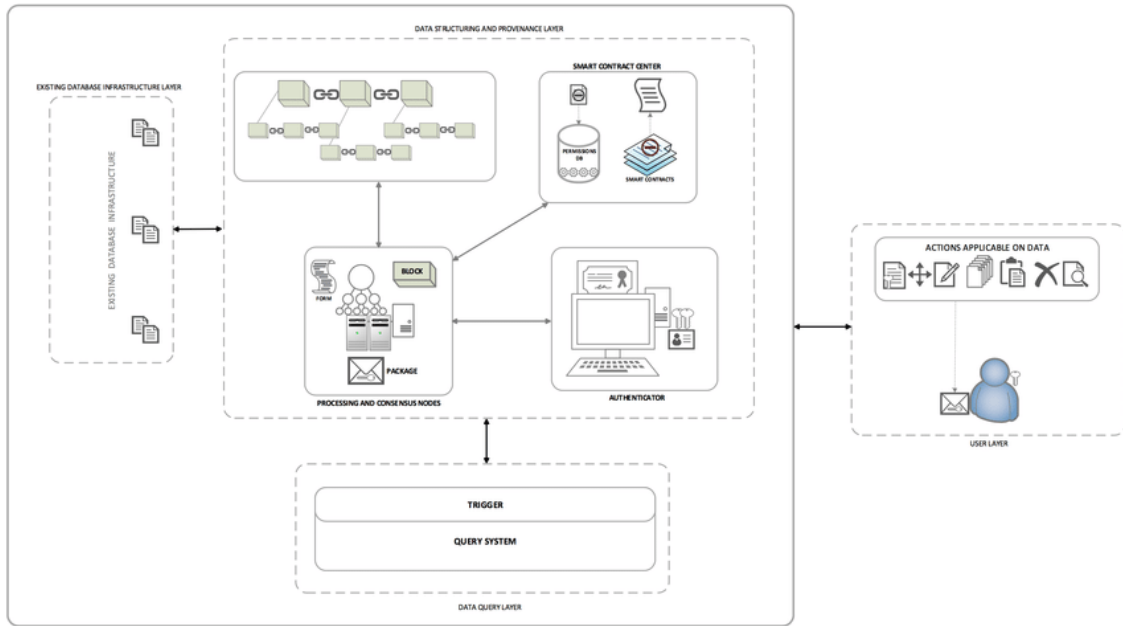


Figure 2.1

There are four main layers in order to categories of structures of the system Figure 2.1.

1. User layer: Users can access information from the system for research or other goals in user layer. One of the example of users can be healthcare

organizations, research institutions, universities and government.

2. Data Query layer: Using this layer it is possible to access, process, forward or reply to queries posed on the system. Users can request data directly with query layer. Querying system and trigger are components of the data query layer.
3. Data Structuring and Provenance layer: From this layer it can be possible to process request for access to data from present database infrastructure layer. Moreover, using this layer it is possible to perform computations on data. In order to guarantee trustless and fairness if auditing, every finished action results are broadcast into network. The layer also responsible to authenticate every request and data access of entire system. There are several entities for this layer:
 - Authenticator has duty to verify requests and encrypts data which is transmitted from requester to the data owner system.
 - Creating and broadcasting the requests is done by processing and consensus nodes.
 - The smart contract is used to store and action entity.
 - The main aim using the blockchain network is to maintain a chronologically distributed database of actions on the supply and request information from the system.
4. Present Database Infrastructure layer: It consists of already set database system.

In order to set up system a user sends a request for data access to the system. Then user signs the data using requester private key. In the first point, the query system gets request and forwards it data structuring and

provenance layer. As soon as authenticator gets the request, it does verification of signature using requesters public key. If the signature is true, the process is initiated, otherwise deleted as invalid request.

If request is true, the request is sent to the processing and consensus nodes by the authenticator where forms are generated. The generated form contains received requester time-stamp and requester ID in the form hash. Then the request is tagged to the form and sent to the database infrastructure where data is retrieved and sent to the structuring and provenance layer. At this point, the smart contract is created and tagged to the form.

The authors of this implementation believes that it is possible to to securely achieve data provenance and auditing sharing medical data between different cloud service providers by using this posed model.

Health-care is one of the possible blockchain application fields. Due to the clarity of the blockchain technology, the government and business try to apply the blockchain technology and collection of its benefits [20] - [23].

2.2 Blockchain Governance

Blockchain governance is the procuring of services in a more effective and decentralized manner, with eliminating state or government officialism, providing more distributed authority propagation. In this way representative agents can be substituted by smart contracts using the blockchain.

Using this technology solution, there is no need for lawyers to prepare length paper documents, instead they prepare self-executing legal documents that promote payments what specific situations occur. Moreover, blockchain can also be used to automatically verify birth certificates, land records, and other records. So, the blockchain technology provides secure record keeping.

Blockchain can allow easy and trusted value transfer between nodes as short-term and long term transfers. In addition, the effect of Blockchain governance on non-banking sectors are well presented.

Before the development of the Blockchain governance, all the task related for organizing business and state related activities is done by a centralized government authority. Since the Blockchain technology, the need for centralized authorities has been decreased. The peers (users) can manage their own business acting as middlemen using decentralized database. In addition, it is possible to use digital currencies as tax havens. When somebody are going to avoid paying taxes, they could set up digital account and transfer digital currencies among them. Furthermore, the governments that trying to intercept communication is prevented with creation severe obstacles with the help of blockchain technology strength on different aspects such as encryption of communication, the acceptance and using decentralized communication channels. Using the blockchain technology the data which is transfered among different endpoints can also be encrypted and stored on Blockchain in encrypted form using secret key, only known to the node participated in the communication process.

Blockchain governance allows a framework to establish digital assets and decentralized exchanges. By now, it was a mass raising money and set equity in organization without help of lawyer. These days, website can issue a cryptotoken to raise money to put up in ares of software development or reward new users by using the dynamics of Blockchain technology. The authors believes that the Blockchain based decentralized exchange will replace centralized physical exchange. The online world will be interacted implementing smart contracts and digital currencies. So, blockchain technology will be on every Internet browser and websites. Through Blockchain governance, it will be possible to match and carry out real time transactions on

Internet connected machines.

Banks also is interested implementing blockchain governance due to its less risks and expenses for potential to simplify the industry's multifaceted and wide-ranging payment and agreement networks. It could save banks billion of dollars eliminating mediators such as Advocates.

Nowadays, one of the most active banks implementing Blockchain governance is UBS where they investigate a different kind of examples of Blockchain technology such as settling trades and issuing bonds. In the same way, Deutsche Bank has also explore the business uses of Blockchain investing amount of resources. Possible uses could be technology in finance, security issuance, transfer, clearing and settlement. According to the Aite Group, it is invested 75 million US dollars in Blockchain technology in 2015.

In the near future, from authors standpoint Blockchain-based governance will be developed and implemented in each field of governance solutions. They think that blockchain is a radical technology innovation that propose a great amount of capacity and possibilities to organizations. From their opinion that the most successful firms will be those implement Blockchain technology faster than their rivals.

The blockchain based approaches can also be implemented on Internet of Things (IoT) security and privacy.

2.3 Blockchain for IoT Security and Privacy: The Case Study of a Smart Home

There is more attention for IoT security in academia and industry. Due to high energy consumption and processing overhead, there is no suitable existing security solutions for IoT. The authors of this work discussed different

kind of core components of smart home tier and various transactions and procedures. They also did security and privacy analysis. In this approach, all communication from external world to the home is handled by so called "miner" which is high resource device and it is always online. Each smart home is equipped with this device. For controlling and auditing communications the miner preserves a private and secure Blockchain. The authors of this works shows that Blockchain based smart home framework is secure with respect to security goals of confidentiality, integrity, and availability.

Internet of Things (IoT) is a device that prevents privacy-sensitive information from different cyber attacks. The devices that support IoT consumes low energy and lightweight. From authors point of view, current security methods is expensive for IoT in terms of energy usage and overhead process. In addition, majority security frameworks are centralized and not suited for IoT due to scale difficulty and single point of failure. The best solution for above mentioned security problems could be the Blockchain technology that sustain Bitcoin the first cryptocurrency system.

In this approach initial step is to demonstrate the process of adding devices and policy header to the local BC. In order to add a device to the smart home, the miner creates a genesis transaction with key sharing. The key which is only known the miner and the device is saved in the genesis transaction. The home owner creates its own policies using proposed policy structure in Figure 2.2 and puts this policy header to the first block. After that, the miner make use of policy header that is in the last block of the Blockchain. Each device have to do authentication to the storage using shared key in order to save data locally. The devices may require to save date on the cloud storage which is store transaction. For saving data the requester have to get starting point that has block number and a hash for authorless authentication aim. There are also access and monitor transactions which is generated by the home

owner or SPs in order to check requested information either in local storage or in cloud storage. If user has several home, he needs partition miners and storage for each homes. The shared overlay is used in order to lower the cost and managing overhead in the instance.

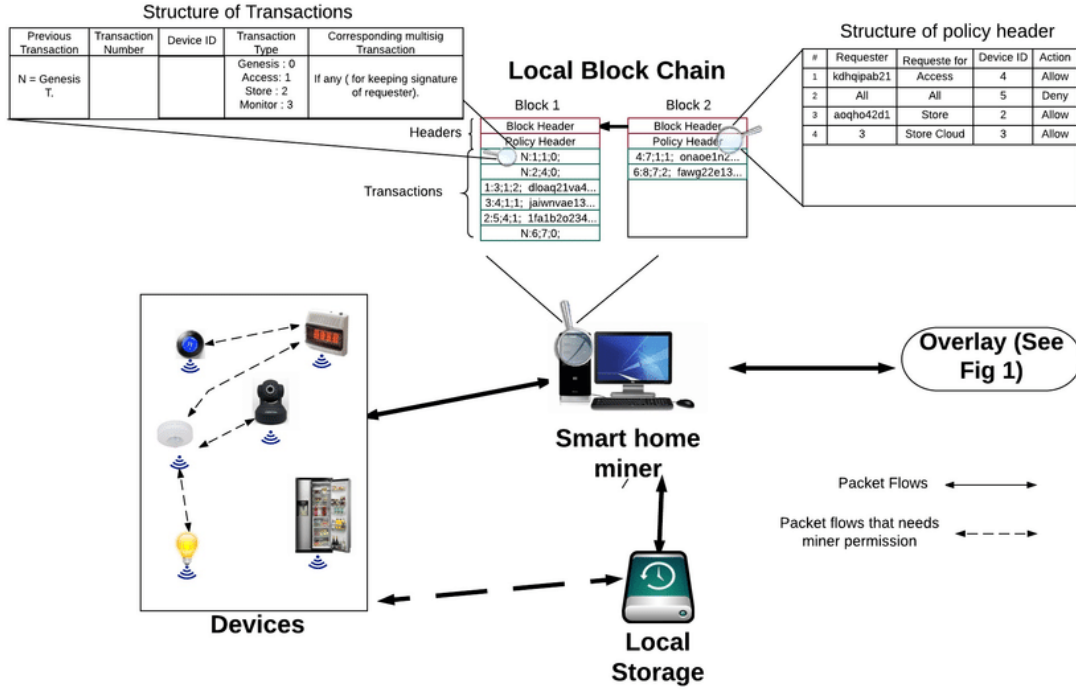


Figure 2.2

The smart devices can intercommunicate with each other or with devices from outside to the smart home. Such communication could be turn light off/on automatically when some one enters or leaves the home. In order to get user control over smart home transactions, it is used shared key that is allocated by the miner to smart home devices.

From point of view of authors, this work is first that aims to optimize Blockchain in the context of smart homes. They are planning to investigate the applications of their framework to other IoT domains. [26]

2.4 Towards Secure E-Voting Using Ethereum Blockchain

The main motivation of author in this projects is, using blockchain technology to provide secure and reliable voting environment. Because, every single administrative decision can be made by people and members who has a computer or a mobile. This will eventually lead humanity to the true direct democracy. Author think that using blockchain in e-voting is important as far as elections can be corrupted or manipulated.

The goal of this project is to create secure Ethereum blockchain based E-voting platform that will process online election of the university. This election could be for department chairs, university rector, or student councils. The main aim of this concept is to provide an election process that is easy for checking and keeping track and this process should be entirely online for everyone who wants to vote in elections of the university. Author's primary contribution is integration of Ethereum blockchain platform with online elections concept.

The authors point of view is Ethereum environment is preferred as the development platform and the blockchain network. The project was made on Ethereum blockchain which is useful for creating decentralized application on blockchain. In the Ethereum network we can use smart contracts rather than web servers. The Ethereum uses smart contract which is written in Solidity programming language (combination of C++ and JavaScript programming languages). In every 15 seconds the smart contracts executed by the peers and the smart contracts are considered if validated at least by 2 users.

There are some problems in order to have completely online elections. The voting platform should have clarity, authentication and provability. Only authenticated real people can vote only one time, only for himself and are not

allowed to change his/her vote after voting. So, the problem has solved using the Ethereum blockchain smart contracts. After smart contracts deployed they cannot be changed or deleted from the blockchain. The project was done on Rinkeby private network, because real Ethereum network cost real ethers. At the end of election, all ballot and vote records will be in the Ethereum blockchain. This project transfer e-voting to the blockchain platform.

Authors of this solution believes that using blockchain technology, smart contracts and Ethereum network in e-voting will solve all the security concerns, such as, privacy, integrity, verification and non-repudiation of votes, and transparency of counting.[\[24\]](#)

2.5 Blockchain Based Access Control

In computer security, Access Control systems are used to control the access data, services, computational systems, storage spaces, and so on. Control policies give rights to access resources.

The main aspect of authors propose in this work is to express right to access a resource and to transfer of such rights among users based on blockchain technology. Mainly, Blockchain "transactions" manages and stores rights to access a resource. The main advantages of the proposed approach are:

- The last right owner can give right to access a resource easily from a user to another user using a blockchain transaction.
- In the beginning, the resource owner define the right through a transaction, and all the other transactions are published on the blockchain. Any user can check who has the rights to perform a given action on a given resource.

Attribute-Based Access Control (ABAC) policies is the known way to express access control rights. From authors point of view eXtensible Access Control Markup Language (XACML) which is defined by the OASIS consortium is the known policy language allowing to express ABAC policies.

The actors of this reference scenario are the resource owner, P, and a number of subjects, S. The resource owner control the policy for each of its resources, R, and it creates, updates and revokes them. The resource owner P defines the access rights on the resource R and using a new transaction called Policy Creation Transaction (PCT), it is stored in the blockchain. From this work we can see that in the beginning, the resource owner decides to give access right.

From authors point of view, the problem of this approach is that the language XACML is a very verbose formalism and policies can be big. There will be problem with saving XACML policies directly on blockchain. The simple solution is to save only a link to the source policy. The authors approach in this situation is hybrid solution among saving the all policy in the blockchain or just a link to it. They decided to save policies in the blockchain but coded in a efficient format that favors compression and keep away from repeating of information. In this architecture of the framework we can see integration XACML reference architecture with blockchain technology in order to enforce blockchain based access control. The Policy Enforcement Point (PEP) and the Policy Administration Point (PAP) customized together in order to allow blockchain based access control policies.

Their implementation uses hybrid solution between storing the entire policy in the blockchain or just give a link to it. They decided to save policies directly in the blockchain and coded in a custom built effectual format (Figure 2.3).

Using Right Transfer Transaction (RTT) that is inside blockchain it is

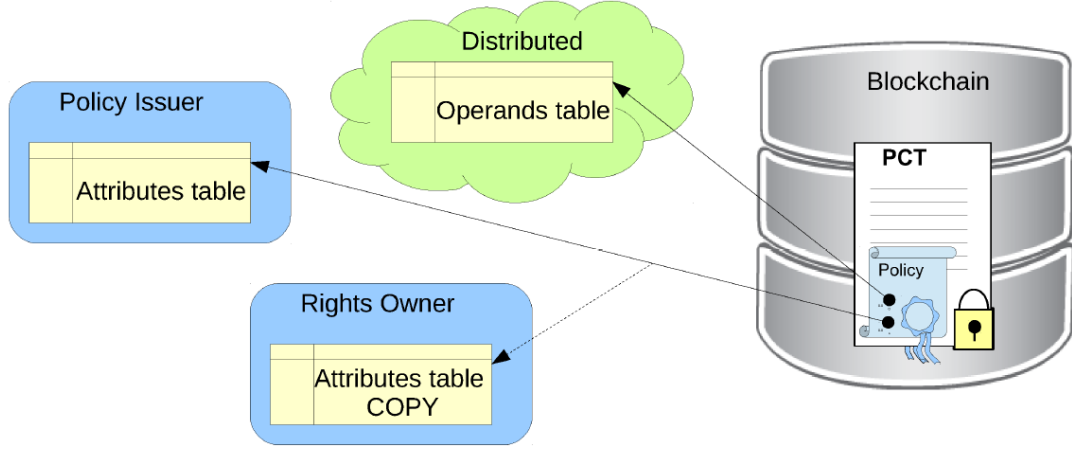


Figure 2.3

possible to send the right to access a resource from right holder to another one. Every RTT need have a link to the policy whose right are being changed. They care that only owner of rights can perform actions on the resource.

The architecture of the project is based on XACML reference architecture which has been joined with blockchain technology. Policy Enforcement Point (PEP) and the Policy Administration Point (PAP) has been customized for allowing the blockchain based access control policies. First of all the resource owner generates a new policy. After that, any number of policy upgrades and right transfers can be performed. At the end, the recourse owner can cancel the policy. In their approach each step is managed with a single Bitcoin transaction.

Authors plan for future is to embedded better access control system to their work in blockchain technology. [25]

Through studying and analyzing existing projects on Blockchain, we can assume that in many situations using Blockchain technology will give us

expected results, increasing efficiency, security, authentication and transparency.

The blockchain technology can be applied in higher education system as well. Higher education institutions have used the blockchain technology to implement on different solutions and approaches. Most of the implementations use the Bitcoin blockchain [27]. Media Lab Learning Initiative and Learning Machine at the Massachusetts Institute of Technology (MIT) made project that establishes an ecosystem for creating, sharing, and verifying blockchain-based educational certificates within the scope of the Digital Certificates Project. The project is based on the Bitcoin blockchain. This approach addressed the issue of digitizing academic certificates.

2.6 Academic Certificates System on the Blockchain

The main goal of the authors is to design a set of tools to issue, display and verify digital credentials using the Bitcoin blockchain and the open badges specification. They have made their project under the MIT open source license (Figure 2.4).

The general design of the certification architecture is simple. The owner of a certificate signs a digital certificate and saves its hash inside blockchain transaction. An output of a transaction is appointed to the recipient.

There are three repositories that make digital certificates architecture of the project:

- Cart-schema reports the data standard for digital certificates. On the blockchain it is placed a digital certificate (JSON file) with the fields needed for the cert-issuer code. They tried to retain the schema as close

to open badges specification as possible.

- Cert-issuer takes a JSON certificate, establish a hash of the certificate and issues a certificate.
- Cert-viewer is needed to display and verify digital certificates after issue. In addition, the viewer code allow users to request certificates and make a new Bitcoin identity.

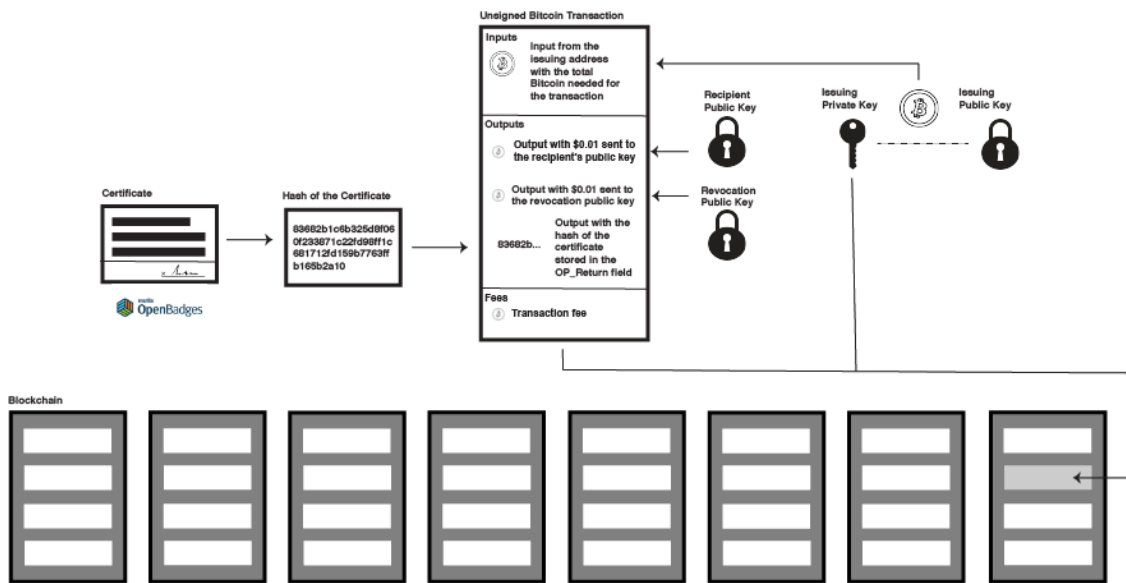


Figure 2.4: Overview of digital certification architecture

This system uses Bitcoin blockchain rather than Ethereum, because of testing and reliability of blockchain to date, interest of miners, and Bitcoin financial investment.

The system uses public/private key pairs to do authentication of a issuer and recipient. In order to hold and transact Bitcoin it is suggested to use wallets. The National University of La Plata (UNLP) and Argentinian College CESYT has developed a framework for a blockchain-based verification of academic achievement [28]. Both of the implementations use Bitcoin

blockchain technology and cryptography to issue diplomas for students.

2.7 Student Diploma on Bitcoin Blockchain

CESYT is an Argentinean College with 40 years of history teaching official careers, such as marketing, international trade, tourism and gastronomy. Moreover, they offer SAP certificates.

They implemented a solution for issuing diplomas to students using Bitcoin blockchain technology.

Apart from traditional diplomas, digital versions of diplomas were created and sent to each graduate, while keeping a copy of each of them. The Ministry of Education in Argentina says that official diplomas have to be signed by the College's Director and Academic Secretary using their own public and private keys and made backup of the seed on the software developed using blockchain. In this software it is used a cryptographic hash function with all diplomas and results were recorded in the Bitcoin blockchain after signing procedure by each authority. Furthermore, they have added a special place in their official website in order to check for diploma's authenticity.

In their opinion this solution is a great example of blockchain technology that brings transparency, authentication and counterfeit-proof certification.

In October 2015, the Holberton School of software engineering in San Francisco announced plans to share their academic certificates on blockchain from 2017. According to HireRight, 86 percent of employers surveyed said that screening had demonstrated a candidate who had lied on his or her resume. Holberton School knows the efforts that companies face, which is one of the main reasons it chose to store students' academic records in the blockchain. Employers no more need to verify candidates' educational credentials by calling universities or paying a third party organization to do the job. It

also saves school money for creating its own database records. So, students of the school are very happy about their academic qualifications will be secured in blockchain technology [29].

The University of Nicosia in Cyprus is also implementing blockchain technology as a way of recording students' achievements and it is proving popular according to Gerge Papageorgiou. They have encountered enthusiasm in specific uses by now and student are very happy to be able to check their digital certificate in blockchain [30].

2.8 Certification Diplomas on the Blockchain

The Leonardo da Vinci Engineering School (ESILV), a renowned academic institution in Paris, France are going to certify and issue diplomas using Bitcoin's blockchain on April 2016

ESILV has done progress in using Bitcoin and blockchain technology previously. In September 2015, they started new course on Bitcoin.

The ESILV is second after San Francisco-based Holberton School which delivers it academic certificates that is secured by the blockchain.

Each graduate will get a digital certificate number with a paper certificate including digital number for future employers to verify the certificate.

From authors point of view the blockchain is more efficient and secure. It is cheap and easy to use than usual verification methods. In blockchain system, a new certificate will cost a few cents and takes milliseconds to transact.

In 2017, it has been created open-source technology which proposes a global higher education credit platform "EduCTX: A Blockchain-Based Higher Education Credit Platform [32]."

2.9 Blockchain Based Professional Networking and Recruiting Platform

In this implementation they demonstrate a blockchain based system of verified education, skill and career information. The Echo system saves professional and related information on blockchain [33].

The information which provided by trusted information authorities such as career related information is saved in a hashed formation on public blockchains. In this system, student allowed to save grades, degrees and certificates in blockchain system securely. The potential employers and higher learning institutions can get student records in a trusted way. Using Echo academic and training institutions can handle degree and certificate checking. Academic institutions has power to give and get trusted academic records. For businesses Echo gives a trusted candidate information.

The Echo system works on public blockchain which cheaper than a similar system in a traditional web environment. The cost of execution is 21,000 gas if it is used Ethereum smart contract. After finishing transaction and getting confirmation in about 60 seconds, the cost is about 0.003 dollar for each record creation. If there are about 10,000 student in the university, creating digital certificates cost about 30 dollar each year.

There are four main elements of the Echo system:

1. Digital Identity and Mapping System - do translation of digital identity into user known names and well known institutions
2. On-chain Credential Creation System - maintenance package creating and credentials storage
3. Off-chain Data I/O System - data read and write from web, database and feeds from outside

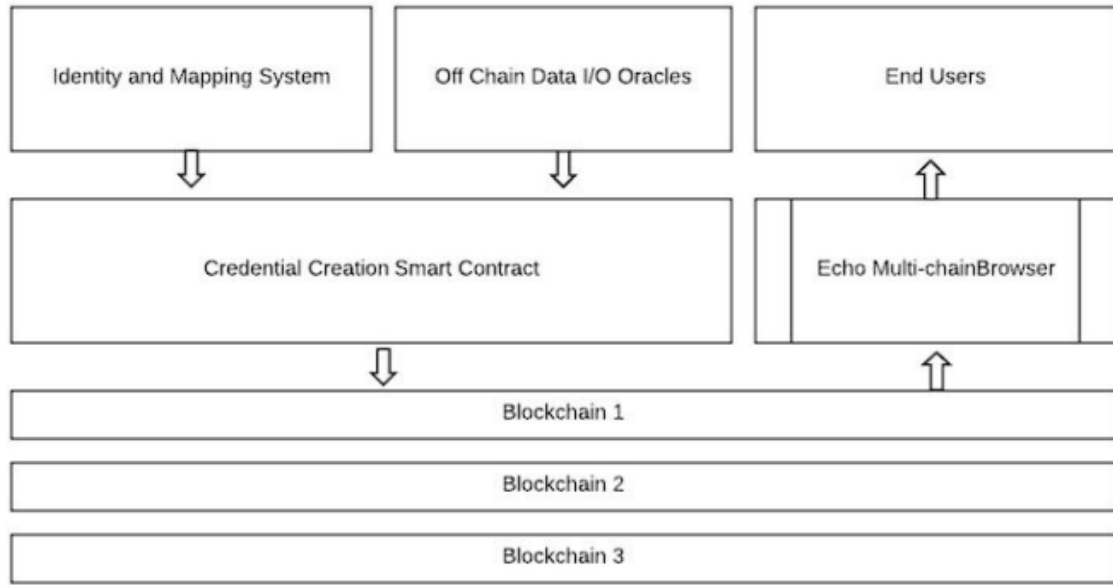


Figure 2.5: Echo System Overview

4. Multi-chain Data Browser for End User - interface for creating and saving records on blockchains.

All the information related student's certificate/degree are saved on blockchain in an encrypted form. Searching the Echo browser, users can look for candidates specific attributes in certificate/degree info. After querying the result is translated into human readable format. Echo browser is responsible to provide result to the user.

The participants' digital identities are saved in the blockchain in the form of public keys. Students and other part have a set of public keys and used in the generation of credentials in the Echo system. The mapping public keys to students, other part, and their accreditation is provided by the Echo system.

Smart contracts are executed in order to create credentials with state information and stored in the blockchain.

Another approach would be certification of diplomas on a bitcoin blockchain

by the Parisian Leonardo da Vinci Engineering School (ESILV) in 2016 [31]. They have partnered with the French Bitcoin startup Paymium for the new milestone.

2.10 EduCTX: A Blockchain-Based Higher Education Credit Platform

They propose a blockchain based higher education credit platform (EduCTX) build on the concept of the European Credit Transfer and Accumulation System (ECTS). This system proposes safety, anonymity, integrity, durability, transparency, and ecosystem simplification for creating a trusted higher education credit and grading using blockchain technology. They implemented a concept which is based on the Ark blockchain Platform.

Figure 2.6 shows depiction of the platform on a higher level. The platform aims for processing and managing ECTX tokens as academic credits, where students and organizations are higher education institutions (HEIs) and user of the platform as peers on the blockchain network.

For completion of the course students get appropriate ECTX token which represents their credit value on the course by his/her home HEI. The students will use EduCTX blockchain wallet in order to collect their tokens. Data about transfer is stored on the blockchain with following information:

1. sender identifier - official name of the sender
2. the receiver - student
3. token - course credit
4. course identification

In order to prevent student sending gained ECTX token to other addresses it will be assigned 2-2 multi signature address by home HEI. EduCTX blockchain API client will be responsible assigning students with ECTX tokens.

Any accepted HEI and their members can join network with set-upping a network node for maintaining safe network and global infrastructure. There is no need to mine transactions by the HEI and thus node as far as EduCTX blockchain is based on DPoS protocol. For this reason, no computing power is needed by nodes and random peers cannot join the network and create new ECTX tokens through mining.

There are four important steps for using EduCTX which are backed up by a Business Process Management diagrammatic representation:

- **Joining the HEI to the EduCTX network.** A new HEI tries to join the EduCTX blockchain network with its blockchain wallet and address containing public and private keys. Then HEI contacts existing HEIs or members of the EduCTX blockchain network.
- **Registration of a student.** The HEI produces a ID of a student and create a new blockchain address for that students with public and private key and 2-2 multisignature blockchain address with its public key and student's public key and these information is saved in the HEI's database. Then the HEI sends ECTX token to 2-2 multisignature address of the student including information about EduCTX blockchain wallet set up instructions, public and private keys of the student's blockchain address, the HEI's public key and script. From other hand, student receive these information and configures his/her

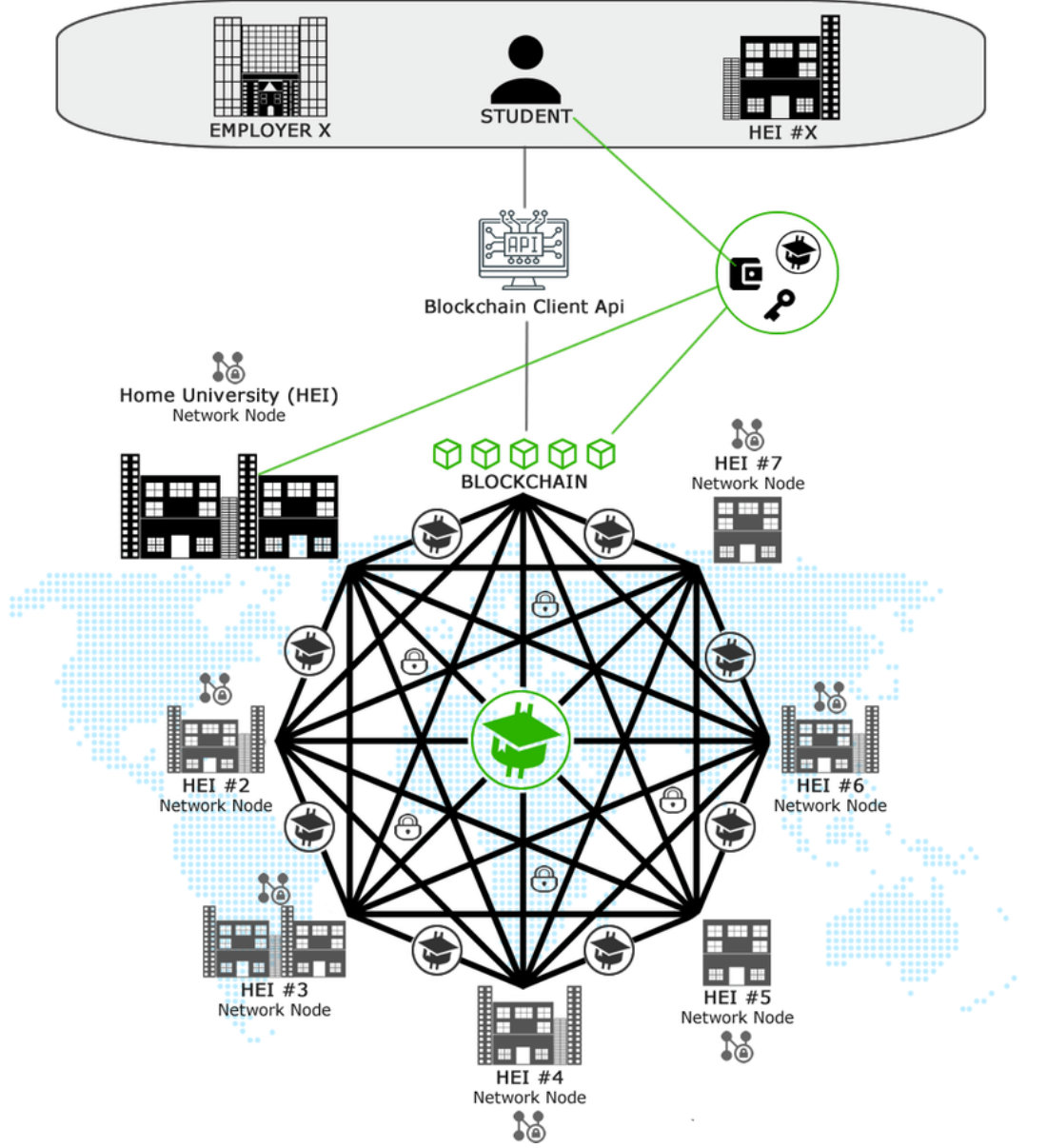


Figure 2.6: A high-level depiction of the EduCTX platform.

blockchain wallet and should sets up 2-2 multisignature blockchain address using his/her public key and HEI's public key. Then student creates and signs a transaction of ECTX token to the HEI's blockchain address. From other side the HEI make signing the transaction with its private key. After transaction successfully confirmed, the information is

saved in database and the HEI gives students' successful wallet creating confirmation.

- **Completion of students' course.** Professor need to verify and publish result after a student takes an exam. If the student passes the exam, professor register exam result and result will be saved in the centralized database. On the next step professor or administration office looks for student's blockchain address in the central database and finds ECTS for the course and send appropriate ECTX token to the 2-2 multisignature blockchain address of the student through blockchain network.
- **Verification of student's credit record.** In order to verify student's course commitment completion, the student need to transmit his/her blockchain address and 2-2 multisignature blockchain address and redeem script to the organization (e.g. university etc.). From other side the organization verifies redeem script, 2-2 multisignature address and checks ECTX token amount in the 2-2 multisignature address which is students credit. Then, student sign the message with his/her address and private key for verification of identity. Using this approach organization can trust a student and for his/her ECTS tokens value

They implemented prototype of EduCTX platform using ARK blockchain which is considered flexible, open source and there are client API implementations with more than 12 programming languages. ARK is being permissionless blockchain, anyone can join it.

They provide several rules in order to ensure safety and validity of the EduCTX records.

- Anonymity of the student
- Prevention student from sending tokens

- Identity prove

From authors point of view proposed EduCTX platform addresses a globally undefined viewpoint for students and organizations. They are planning to extend their future work based on smart contracts and different version of the blockchain technology.

The EduCTX platform which is supported by the Slovenian Research Agency based on ARK blockchain system. They used the blockchain system only to send ECTS credits to students address saving grades on centralized database. There is no use of Smart Contracts in their implementation. Moreover operation which is needed to prevent students sending tokens to other blockchain addresses a bit time consuming. The EduCTX platform has not been tested in real life yet.

In contrary, the platform which we propose uses Ethereum blockchain system which is used to developer most of Decentralized Applications. In our project we used for each subject a token (specified by symbol name) and if student passes an exam, quantity of tokens which is equal to student's grade will be transacted to the student address. In that case nobody are allowed to change the grade. We have used an ERC20 token Smart Contracts for each subject. There is a function `freezeAccount()` which will prevent students from sending any tokens to other blockchain addresses. Our implementation has been tested in Turin polytechnic university in Tashkent. The platform transfers grading system from the analog and physical world into a globally efficient, more simplified, ever-present version based on the blockchain technology.

The majority above-mentioned projects in the higher education area relies upon closed ideas and often do not consider detailed description or even keep on an idea level. Contrariwise, the implementation presented in this paper

relies on open-source technologies. The implemented platform is based on the Ethereum blockchain technology.

Chapter 3

Design and Implementation

3.1 Requirements

The requirement of the project is to create a blockchain in which all the transactions represents the university exams passed by the students and their grades. Transactions must only be create by authorized personnel. Students can read their career history, administrative staff and professors can record grades. So, nobody can be able to change grades after they are rated. We should provide web based application and a web service to support the functions described. A real case study is Turin polytechnic university in Tashkent (TTPU).

3.2 The Proposed Platform

This section outlines the proposed platform, blockchain based storage of students career. An abstract illustration of the platform on a higher level

is presented in Figure 3.1. The blockchain platform is implemented for processing, managing and controlling ERC20 tokens as grades for students on specific subjects and tested on distributed P2P network, where peers of the blockchain network are teachers and students.

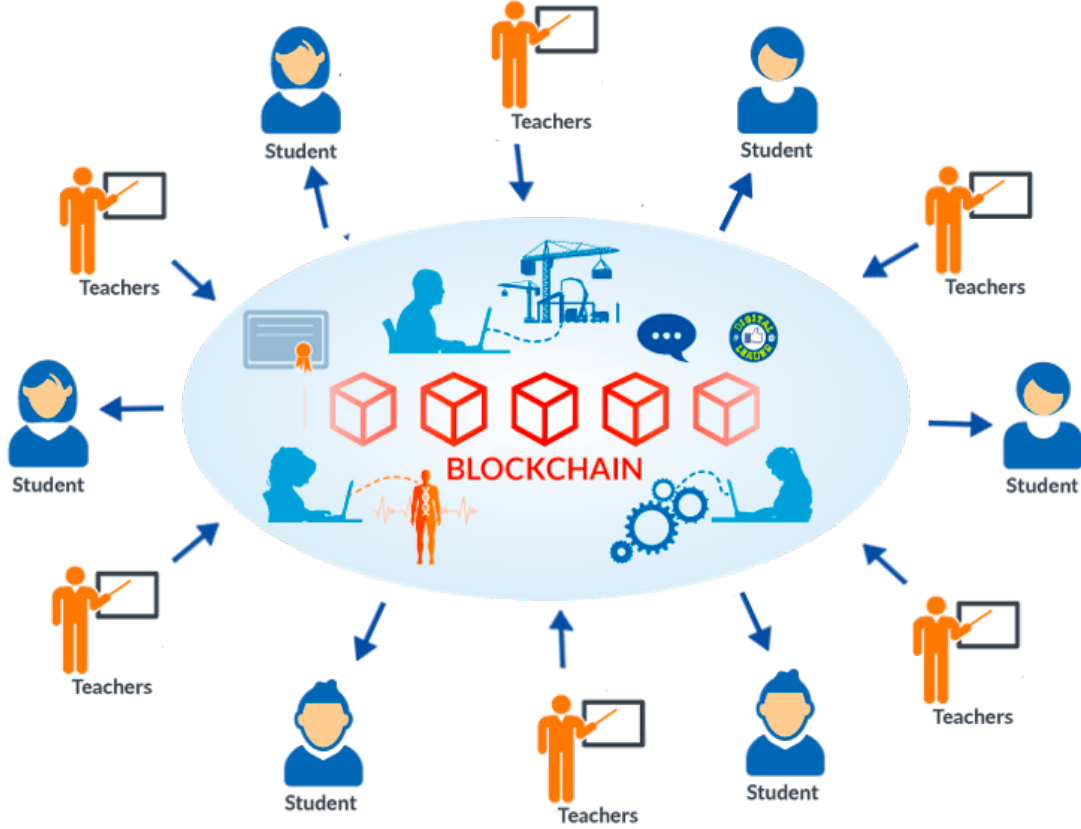


Figure 3.1: A high-level description of the proposed platform.

The ERC20 tokens represents an equivalent to student's grade value for completed courses. Each student will have a devoted blockchain wallet, where he/she will gain ERC20 tokens. Every time a student finishes a course, the teacher on that course will transfer the appropriate number of ERC20 tokens to his/her blockchain address. The transferred information is stored on the blockchain. Moreover, using his/her blockchain address, the student gets

ERC20 tokens that is used to prove his/her grade on specific course. Students are not able to transfer any of the received tokens to other blockchain addresses.

We selected the Ethereum Blockchain as the underlying technology of our platform. With Ethereum network we can use several use case and smart contracts which will be useful for our development. We can omit the server, using smart contracts as a web server. For these reasons, it is very difficult to change or harm the source code of the software.

In our development, we used ERC20 Token Smart contract for each subject of the university. We differentiate the subjects using the symbol name of the token. Tokens are marks for a student. Each smart contract has several functions for balance info, sending transactions (in our case tokens), allowing professors account for sending tokens and freeze students account from sending tokens to other accounts. When we deploy our contracts we pass arguments for them. Arguments are token name, symbol name and quantity of tokens. In the beginning all tokens will be in one account who will deploy contracts (administrator's account). Administrator will be responsible for tokens. He will give allowance for each professor's account and give them permission to spend tokens from his account. On the other hand, administrator will freeze students' accounts from sending token to other accounts. So, students can only receive tokens, but cannot send them.

3.2.1 Private Blockchain Setup

There are two kind of blockchain networks available to use. Public and private blockchain networks. In our implementation we have used private blockchain network. In order to setup environment required to build private ethereum blockchain we have installed go-ethereum (geth) node.

```
$ npm install geth
```

Before creating private network we need to create genesis.json file. Genesis.json file is configuration file for go-ethereum in order to create private network. It is the file that go-ethereum needs to create very first block and this is so called genesis block and for that reason the file is called genesis.json. So, it is a json file that is javascript object notation.

```
{
  "config": {
    "chainId": 15,
    "homesteadBlock": 0,
    "eip155Block": 0,
    "eip158Block": 0
  },
  "gasLimit": "0x8000000000000000",
  "difficulty": "0x400",
  "alloc": {}
}
```

Listing 3.1: genesis.json file

Let's explore the configuration file.

config: it is the main blockchain configuration and it has following configuration.

chainId: the chain identifier currently set 15, but we can set it to any number except 1,2 or 3. Because, 1, 2, or 3 reserved for: 1 - mainnet; 2 and

3 - testnets. chain identifier has to be integer number. We can also use this chain identifier to protect from reply attack.

there is the homestead block. Ethereum is separated in certain releases. The last one was the front year. We have been used homestead block beginning from 0 in our blockchain network.

difficulty: it is very important configuration if we want to mine ether or create contracts. The difficulty determines how the difficult it is to mine a block. In our case we used lower difficulty to mine blocks and to transfer transactions faster.

gasLimit: from the other hand is necessary for deploying contracts. Each contract written in solidity will be compiled to something like assembly ethereum virtual machine. And each of these instructions takes up some gas. The gas limit indicates maximum amount of gas that can be used in each block.

alloc(): preallocate ether to certain account.

Now we will create new directory where will store our new blocks in. And first, initialize this new directory with genesis.json file.

```
$ geth --datadir mychaindata init .\genesis.json
```

At this point, geth creates mychaindata directory inside my empty directory where we have our genesis.json file. At the beginning we have no etherbase set and no accounts as default. Now we can start our new private blockchain with this mychaindata data directory. On the same directory we start geth:

```
$ geth --datadir .\mychaindata\ --nodiscover
```

Here we will open new command line window and attach another geth instance to our running geth instance.

```
$ geth attach ipc:\\.\pipe\geth.ipc
```

Now we have attached to our running geth instance. Here we can use different kind of APIs. They are management APIs and standard ethereum management APIs which is the same for every single geth node around the world. At this moment we can create new accounts and we can start mining ether in order to interact with decentralized application.

In order to create accounts in private network we can create them one by one using API. In the beginning we do not have any accounts on our private network. We can check our account using

```
> eth.accounts
```

```
> personal.newAccount
```

This function will create new account in our private network. We will create as many accounts as we want. In our project our accounts are a administrator, professors and students. After creating accounts function will generate public and private keys for each account. So, everything in geth is file based database. The private keys will be stored on keystore folder. We can copy the private keys into another keystore folders. Each user will be responsible for their private keys. Private keys never should be loosed or deleted. Otherwise user cannot access to their accounts on blockchain.

Public key we will use as a user name and private key as a password for the accounts.

All the blocks on blockchain system will be stored on chaindata or lightchain-data folder inside the project folder. If we start mining the geth will create new blocks. Each of them has around 10 mbytes. Every single transaction on ethereum blockchain will be stored in the block and all the blocks are depending each other. Every node connected to this blockchain will have exactly same blocks on their nodes. So, this is the core point of the ethereum blockchain.

The administrator account will start mining ether in order to use them in blockchain system. The following function used in order to start mining ethers:

```
> personal.unlockAccount(eth.accounts[0]);
```

which will unlock the first account (account of the administrator). Then we will start mining with one thread:

```
> miner.start(1);
```

In our development of the project we have used Ethereum ERC20 Token Standards Interface for each subject of the course. Ethereum ERC20 Token standard interface consists of couple functions that makes easy to use in our implementation.

```
pragma solidity ^0.4.16;
```

```
contract owned {
    address public owner;

    function owned() public {
        owner = msg.sender;
    }

    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }

    function transferOwnership(address newOwner)
        onlyOwner public {
        owner = newOwner;
    }
}

interface tokenRecipient { function receiveApproval(
    address _from, uint256 _value, address _token,
    bytes _extraData) external; }

contract Token_01RKWLM is owned {
    // Public variables of the token
    string public name;
    string public symbol;
    uint8 public decimals = 18;
```

```
// 18 decimals is the strongly suggested default,
    avoid changing it
uint256 public totalSupply;

uint256 public sellPrice;
uint256 public buyPrice;

mapping (address => bool) public frozenAccount;

/* This generates a public event on the blockchain
    that will notify clients */
event FrozenFunds(address target, bool frozen);

// This creates an array with all balances
mapping (address => uint256) public balanceOf;
mapping (address => mapping (address => uint256))
    public allowance;

// This generates a public event on the blockchain
    that will notify clients
event Transfer(address indexed from, address indexed
    to, uint256 value);

// This generates a public event on the blockchain
    that will notify clients
event Approval(address indexed _owner, address
    indexed _spender, uint256 _value);
```

```
// This notifies clients about the amount burnt
event Burn(address indexed from, uint256 value);

/**
 * Constructor function
 *
 * Initializes contract with initial supply tokens to
 *   the creator of the contract
 */
function Token_01RKWLM(
uint256 initialSupply,
string tokenName,
string tokenSymbol
) public {
totalSupply = initialSupply * 10 ** uint256(decimals
); // Update total supply with the decimal amount
balanceOf[msg.sender] = totalSupply;
    // Give the creator all initial tokens
name = tokenName;
    // Set the name for display purposes
symbol = tokenSymbol;
    // Set the symbol for display purposes
}

/**
 * Transfer tokens
 */
```

```
* Send '_value' tokens to '_to' from your account
*
* @param _to The address of the recipient
* @param _value the amount to send
*/
function transfer(address _to, uint256 _value)
    public returns (bool success) {
    _transfer(msg.sender, _to, _value);
return true;
}

/**
* Transfer tokens from other address
*
* Send '_value' tokens to '_to' in behalf of '_from'
*
* @param _from The address of the sender
* @param _to The address of the recipient
* @param _value the amount to send
*/
function transferFrom(address _from, address _to,
    uint256 _value) public returns (bool success) {
require(_value <= allowance[_from][msg.sender]);
    // Check allowance
allowance[_from][msg.sender] -= _value;
    _transfer(_from, _to, _value);
return true;
}
```

```
/**
 * Set allowance for other address
 *
 * Allows '_spender' to spend no more than '_value'
 * tokens in your behalf
 *
 * @param _spender The address authorized to spend
 * @param _value the max amount they can spend
 */
function approve(address _spender, uint256 _value)
    public
returns (bool success) {
    allowance[msg.sender][_spender] = _value;
    emit Approval(msg.sender, _spender, _value);
    return true;
}

/**
 * Set allowance for other address and notify
 *
 * Allows '_spender' to spend no more than '_value'
 * tokens in your behalf, and then ping the contract
 * about it
 *
 * @param _spender The address authorized to spend
 * @param _value the max amount they can spend
 */
```

```
* @param _extraData some extra information to send
    to the approved contract
*/
function approveAndCall(address _spender, uint256
    _value, bytes _extraData)
public
returns (bool success) {
    tokenRecipient spender = tokenRecipient(_spender);
    if (approve(_spender, _value)) {
        spender.receiveApproval(msg.sender, _value, this,
            _extraData);
    }
    return true;
}

/**
 * Destroy tokens
 *
 * Remove ‘_value’ tokens from the system
    irreversibly
 *
 * @param _value the amount of money to burn
 */
function burn(uint256 _value) public returns (bool
    success) {
    require(balanceOf[msg.sender] >= _value);    // Check
        if the sender has enough
```

```
balanceOf[msg.sender] -= _value;           //
    Subtract from the sender
totalSupply -= _value;                     //
    Updates totalSupply
emit Burn(msg.sender, _value);
return true;
}

/**
 * Destroy tokens from other account
 *
 * Remove '_value' tokens from the system
 *   irreversibly on behalf of '_from'.
 *
 * @param _from the address of the sender
 * @param _value the amount of money to burn
 */
function burnFrom(address _from, uint256 _value)
    public returns (bool success) {
require(balanceOf[_from] >= _value);
    // Check if the targeted balance is enough
require(_value <= allowance[_from][msg.sender]);
    // Check allowance
balanceOf[_from] -= _value;
    // Subtract from the targeted balance
allowance[_from][msg.sender] -= _value;
    // Subtract from the sender's allowance
```

```
totalSupply -= _value;
    // Update totalSupply
emit Burn(_from, _value);
return true;
}

/* Internal transfer, only can be called by this
   contract */
function _transfer(address _from, address _to, uint
    _value) internal {
require (_to != 0x0);
    // Prevent transfer to 0x0 address. Use burn()
    instead
require (balanceOf[_from] >= _value);
    // Check if the sender has enough
require (balanceOf[_to] + _value >= balanceOf[_to]);
    // Check for overflows
require(!frozenAccount[_from]);
    // Check if sender is frozen
//require(!frozenAccount[_to]);
        // Check if recipient is
        frozen
balanceOf[_from] -= _value;
    // Subtract from the sender
balanceOf[_to] += _value;
    // Add the same to the recipient
emit Transfer(_from, _to, _value);
}
```

```
/// @notice Create 'mintedAmount' tokens and send it
    to 'target'
/// @param target Address to receive the tokens
/// @param mintedAmount the amount of tokens it will
    receive
function mintToken(address target, uint256
    mintedAmount) onlyOwner public {
balanceOf[target] += mintedAmount;
totalSupply += mintedAmount;
emit Transfer(0, this, mintedAmount);
emit Transfer(this, target, mintedAmount);
}

/// @notice 'freeze? Prevent | Allow' 'target' from
    sending & receiving tokens
/// @param target Address to be frozen
/// @param freeze either to freeze it or not
function freezeAccount(address target, bool freeze)
    onlyOwner public returns(bool success) {
frozenAccount[target] = freeze;
emit FrozenFunds(target, freeze);
return true;
}

/// @notice Allow users to buy tokens for '
    newBuyPrice' eth and sell tokens for 'newSellPrice
    ' eth
```

```
/// @param newSellPrice Price the users can sell to
    the contract
/// @param newBuyPrice Price users can buy from the
    contract
function setPrices(uint256 newSellPrice, uint256
    newBuyPrice) onlyOwner public {
sellPrice = newSellPrice;
buyPrice = newBuyPrice;
}

/// @notice Buy tokens from contract by sending
    ether
function buy() payable public {
uint amount = msg.value / buyPrice;           //
    calculates the amount
_transfer(this, msg.sender, amount);          //
    makes the transfers
}

/// @notice Sell 'amount' tokens to contract
/// @param amount amount of tokens to be sold
function sell(uint256 amount) public {
address myAddress = this;
require(myAddress.balance >= amount * sellPrice);
    // checks if the contract has enough ether to
    buy
_transfer(msg.sender, this, amount);          //
    makes the transfers
```

```
msg.sender.transfer(amount * sellPrice);           //
    sends ether to the seller. It's important to do
    this last to avoid recursion attacks
}
}
```

Our smart contracts need to be deployed before using the functions inside. Using `module.exports` function we will deploy each our contacts.

```
var Token_01TCAPT = artifacts.require("Token_01TCAPT");
var Token_030FZPT = artifacts.require("Token_030FZPT");
var Token_040GCPT = artifacts.require("Token_040GCPT");
var Token_11KXWPP = artifacts.require("Token_11KXWPP");
var Token_04LSEPT = artifacts.require("Token_04LSEPT");
var Token_05LSZPT = artifacts.require("Token_05LSZPT");
var Token_02LSIPT = artifacts.require("Token_02LSIPT");
var Token_01SXPPT = artifacts.require("Token_01SXPPT");
var Token_07KSIPT = artifacts.require("Token_07KSIPT");
```

```
var Token_06JEZPT = artifacts.require("Token_06JEZPT");
var Token_01SXOPT = artifacts.require("Token_01SXOPT");
var Token_040GHPT = artifacts.require("Token_040GHPT");
var Token_010GGPT = artifacts.require("Token_010GGPT");

module.exports = function(deployer){
  deployer.deploy(Token_01TCAPT, 30, "Algorithm and
    Programming 2", "01TCAPT");
  deployer.deploy(Token_030FZPT, 30, "Circuit Theory",
    "030FZPT");
  deployer.deploy(Token_040GCPT, 30, "Electronic
    Systems and Technologies", "040GCPT");
  deployer.deploy(Token_11KXWPP, 30, "Physics II", "11
    KXWPP");
  deployer.deploy(Token_04LSEPT, 30, "Computer
    Architecture", "04LSEPT");
  deployer.deploy(Token_05LSZPT, 30, "Databases", "05
    LSZPT");
  deployer.deploy(Token_02LSIPT, 30, "Mathematical
    Methods", "02LSIPT");
  deployer.deploy(Token_01SXPPT, 30, "Object oriented
    programming", "01SXPPT");
```

```
deployer.deploy(Token_07KSIPT, 30, "Computer  
networks", "07KSIPT");  
deployer.deploy(Token_06JEZPT, 30, "Operating  
systems", "06JEZPT");  
deployer.deploy(Token_01SXOPT, 30, "Automatic  
controls", "01SXOPT");  
deployer.deploy(Token_040GHPT, 30, "Applied  
electronics and measurements", "040GHPT");  
deployer.deploy(Token_010GGPT, 30, "Signal analysis  
and processing", "010GGPT");  
};
```

3.2.2 Install and configure Truffle

In this section we will discuss about how to install truffle and initialize new project. To install truffle we type:

```
> npm install -g truffle
```

In order to initialize our truffle project we use truffle-init-webpack javascript framework for creating our distributed application. For downloading the box installing the necessary dependencies.

```
> truffle unbox webpack
```

3.2.3 One-click Login with Blockchain

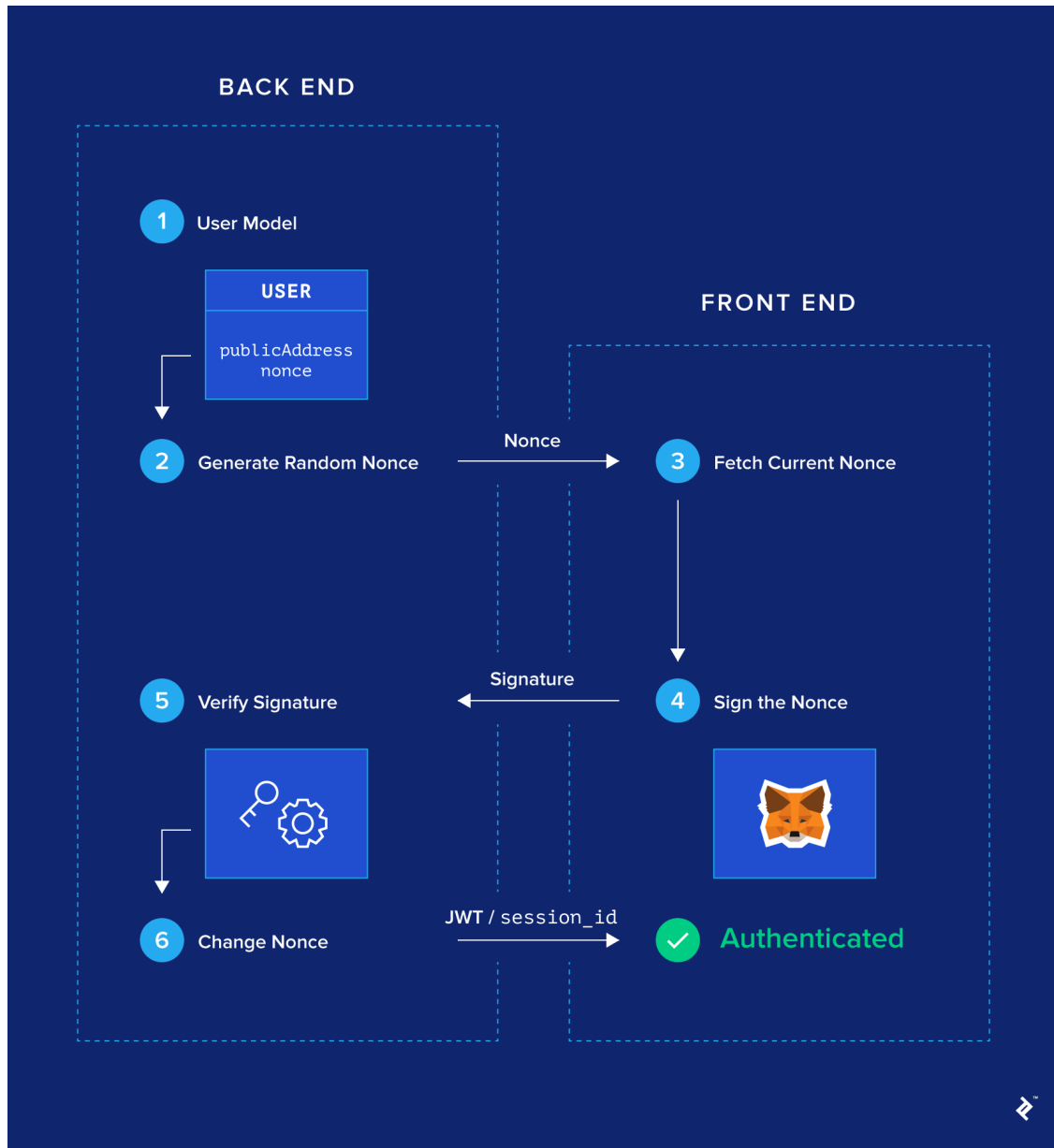


Figure 3.2

We implemented one click login flow which is cryptographically secure. This login flow uses MetaMask extension where all data will be stored on the back end.

Owner of an account will sign a piece of the data using his/her private key in order to prove securely his/her ownership on the account. The back end will generate a piece of data and if owner manages to sign then the back end will see you as the owner of that public address. Therefore, using user's public address as identifier we can do authentication of the user.

How the Login Flow Works. In our implementation all user visiting our web page should have MetaMask installed.

Step1: The User Model (Back-End). Firstly, We need to have two required fields in our User model: public address and nonce. In addition, publicAddress needs to be unique. We will use publicAddress as required field on signup. There is no need to type publicAddress by user. The publicAddress can be fetched using web3.eth.coinbase.

```
const User = sequelize.define('User', {
  nonce: {
    allowNull: false,
    type: Sequelize.INTEGER.UNSIGNED,
    // Initialize with a random nonce
    defaultValue: () => Math.floor(Math.random() *
      1000000) },
  publicAddress: {
    allowNull: false,
    type: Sequelize.STRING,
    unique: true,
    validate: { isLowercase: true }},
  username: {
```

```
type: Sequelize.STRING,  
unique: true }));
```

Step2: Nonce Generation (Back-End). For each user in the database there will be generated a random string in the nonce field. The nonce could be a any big random integer number. This is done in the `defaultValue()` function in the above model definition.

Step3: User Fetches Their Nonce (Front-End). We can access to `window.web3` using the JavaScript code in our front-end. After that we will call `web3.eth.coinbase` in order to get current public address of the MetaMask account. We fire an API call to response the nonce associated with their public address as soon as user clicks on the login button. If there is no any returned result, it means that the current public address hasn't signed up yet. In that case user should talk with administration of the system for sign up. On the other hand, if there is a result, then the nonce will be stored. Following code will run the user clicks on the login button:

```
class Login extends Component {  
  handleClick = () => {  
    // --snip--  
    const publicAddress = web3.eth.coinbase.toLowerCase()  
      ();  
    // Check if user with current publicAddress  
    is already present on back end  
    fetch(`${process.env.REACT_APP_BACKEND_URL}/  
users?publicAddress=${publicAddress}`)
```

```
.then(response => response.json())
// If yes, retrieve it. If no, create it.
.then(
users => (users.length ? users[0] :
this.handleSignup(publicAddress))
)};}
```

Step4: User Signs the Nonce (Front-End). When the front end gets nonce in the response of the previous API call, it runs the following code:

```
web3.personal.sign(nonce, web3.eth.coinbase,
    callback);
```

This will popup confirmation for signing the message. For preventing from malicious data the nonce will be displayed in this popup. The user will know that signing data is safe. When user accepts it, the callback function will be called with the signed message (called signature) as an argument. In that case, the front end makes another API call to POST `/api/authentication`, passing a body with both signature and `publicAddress`.

```
class Login extends Component {
  handleClick = () => {
    // --snip--
    fetch(`${process.env.REACT_APP_BACKEND_URL}/
users?publicAddress=${publicAddress}`)
```

```
.then(response => response.json())
// If yes, retrieve it. If no, create it.
.then(
  users => (users.length ? users[0] :
    this.handleSignup(publicAddress)))
// Popup MetaMask confirmation modal to sign
  message
.then(this.handleSignMessage)
// Send signature to back end on the /auth route
.then(this.handleAuthenticate)};
handleSignMessage = ({ publicAddress, nonce }) => {
  return new Promise((resolve, reject) =>
    web3.personal.sign(
      web3.fromUtf8('I am signing my one-time nonce: ${
        nonce}'),
      publicAddress,
      (err, signature) => {
        if (err) return reject(err);
        return resolve({ publicAddress, signature });});});
handleAuthenticate = ({ publicAddress, signature })
  =>
  fetch(`${process.env.REACT_APP_BACKEND_URL}/auth`,
    {
      body: JSON.stringify({ publicAddress, signature }),
      headers: {
        'Content-Type': 'application/json'},
      method: 'POST'
    }).then(response => response.json());}
```

Step5: Signature Verification (Back-End). The back end fetches the user in the database corresponding corresponding publicAddress as soon as it receives a POST /api/authentication request. In particular it fetches the associated nonce. Using the nonce, the public address, and the signature, the back end uses cryptography to verify that user has signed the nonce correctly. In that case, user considered as authenticated and JWT or session identifier can be returned to the front end.

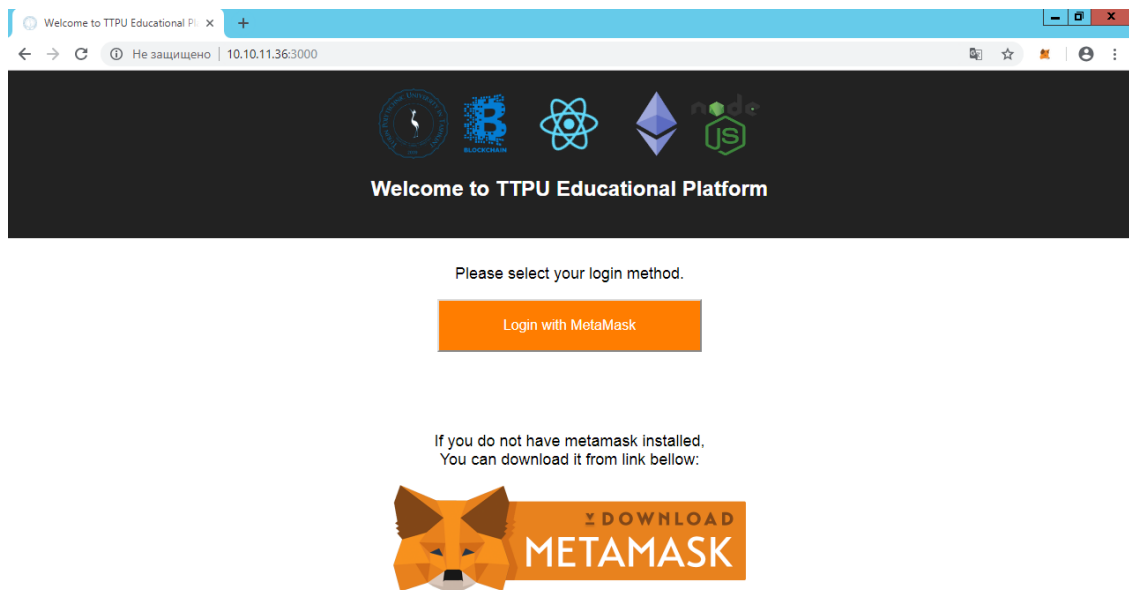


Figure 3.3

Step6: Nonce Change (Back-End). Every time when user wants to login again the nonce will be changed in the database.

```
// --snip--
.then(user => {
user.nonce = Math.floor(Math.random() * 1000000);
return user.save();
})
// --snip--
```

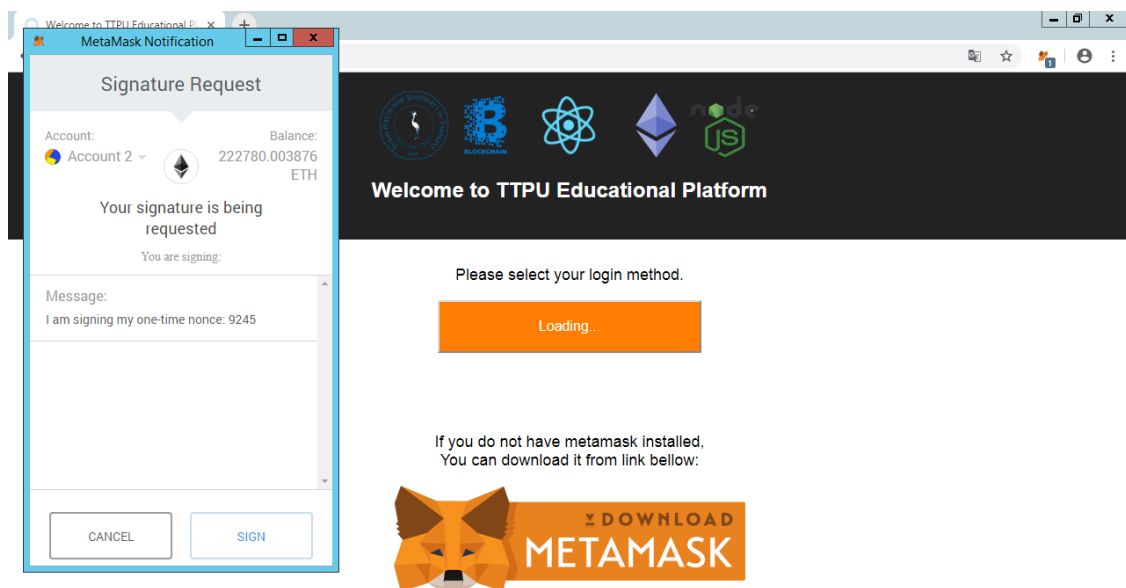


Figure 3.4

This is the way how we managed a nonce-signing passwordless login flow.

There several arguments why this login flow is better than login/password and social logins:

- **Security:** Login system with public-key encryption is more secure than email/password or by a third party. The MetaMask will store credentials on local computers which will decrease different attacks. Figure 3.3
- **Easy UX:** We do not need to write or remember login information (any passwords) in this one-click login flow. Figure 3.4
- **Privacy:** There is no need to use email and involve a third party.

3.2.4 Student's registration

When students enroll into the University, the administration of the university issues a student ID and generates a new blockchain address for the student with public and private keys and send these information to the student. Administration stores following information about student into its centralized database.

- Student ID;
- Username;
- Public key of the student's wallet;
- Enrolled course of the student;
- Subject information;

Student receives information from administration and setups his/her blockchain wallet and a single address with public and private key got from administration. The wallet data need be saved securely. Then administration finish student's wallet creation successfully. Figure 3.13 illustrates a process model for the above scenario using activity diagram.

3.2.5 Professor's registration

Professor registration is almost the same as student. There is a bit difference on information stored into the centralized database. Figure 3.14 illustrates a process model of the professor's scenario using the activity diagram.

3.2.6 Student's course completion

After a student takes an exam, the professor should verify the results. If the student passes the exam, professor sends corresponding ERC20 tokens to the students blockchain address. The transaction is processed through the blockchain network. When the transaction is confirmed, the information about the grade will be saved into central database. From other side student's can see their wallet on specific ERC20 token about their grades on subjects. Figure 3.15 illustrates a process model of the above scenario using the activity diagram.

3.2.7 Implementation of Administrator Account

In our implementation we created one administrator which is the creator of the private blockchain. He will deploy all smart contracts and own all the tokens in private blockchain. Administrator account consists of following pages:

- **User information.** There is information about an user (his public address name, surname, username). Figure 3.5
- **Allowance Token.** In the beginning, all tokens for each subjects will be in administrators accounts. Administrator will give permission for each professor accounts to use tokens from his (administrator) account

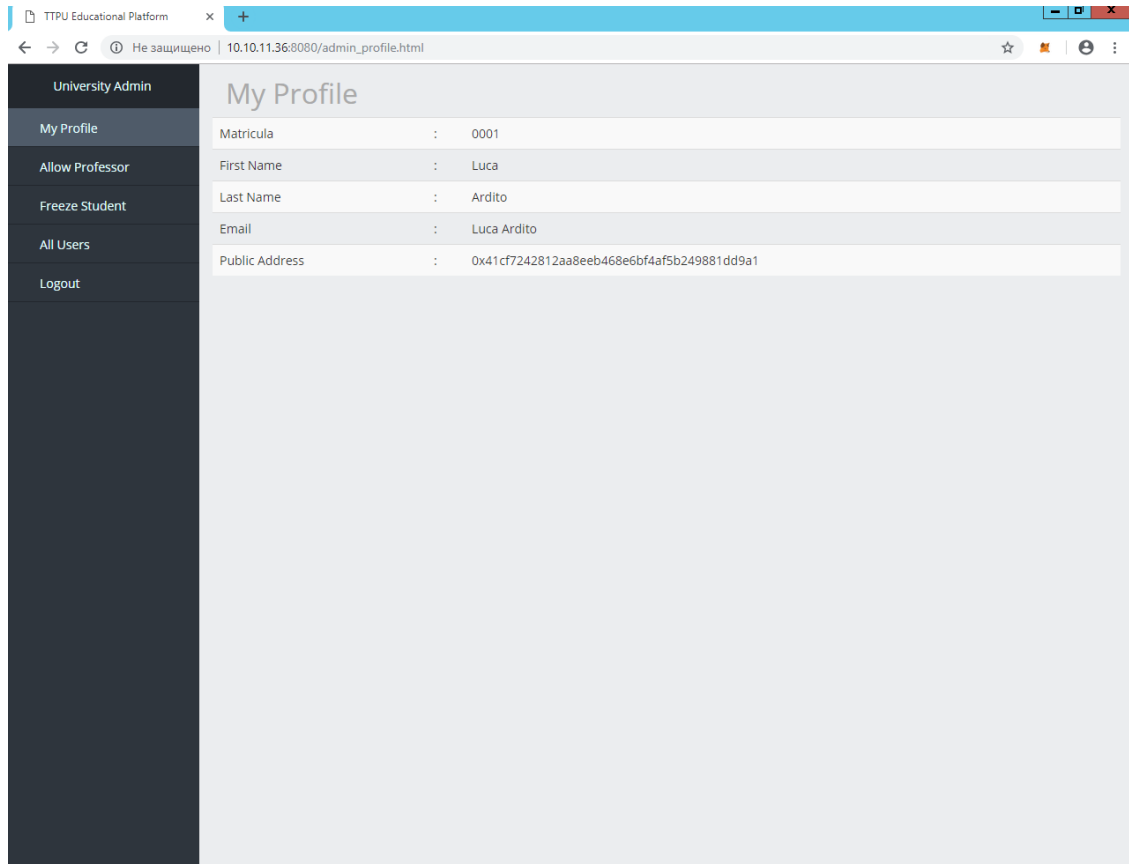


Figure 3.5

by subjects. In this page owner of administrator's account should enter public address of the professors account and amount of tokens that professors will be allowed to send. Then, he will press the submit button. This will pop-up confirmation window and he will give some gas (needed for transaction complete) and types submit button. If transaction is mined by minimum the peers it will be considered as confirmed and completed. Otherwise, the error message will be shown. [Figure 3.6](#)

- **Freeze Account.** In this page, administrator will freeze students account from sending tokens to other accounts. Frozen accounts only can receive tokens but not allowed send tokens to other accounts. This will

3.2 – The Proposed Platform

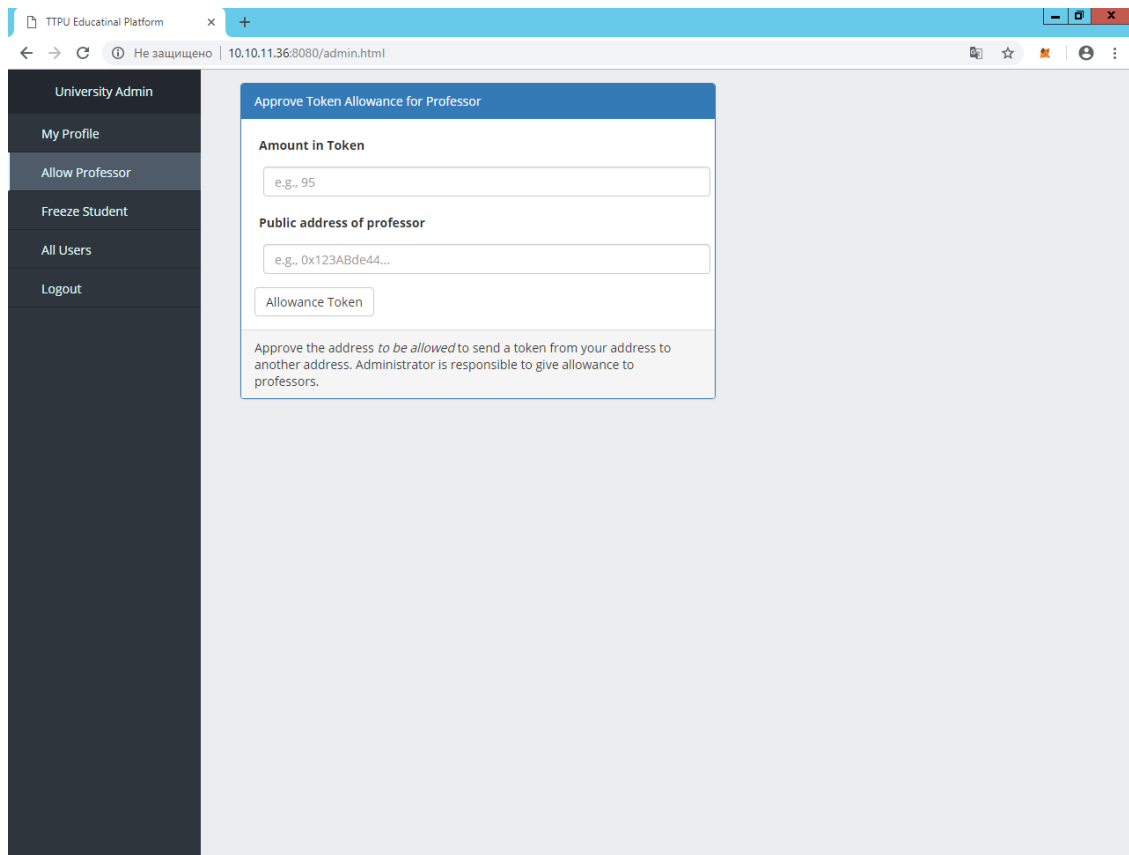


Figure 3.6

be done using public address of the student. Figure [3.7](#)

- **Users list.** Administrator will control all users using this page. He/she can see all the users (including professor and students) list on this page. And can add blockchain user information into centralized database. We will save blockchain users information in usual mysql database. Add users will pop-up modal with form. Form includes following inputs: matricula, public address, name, surname, username, position. Figure [3.8 - 3.9](#)

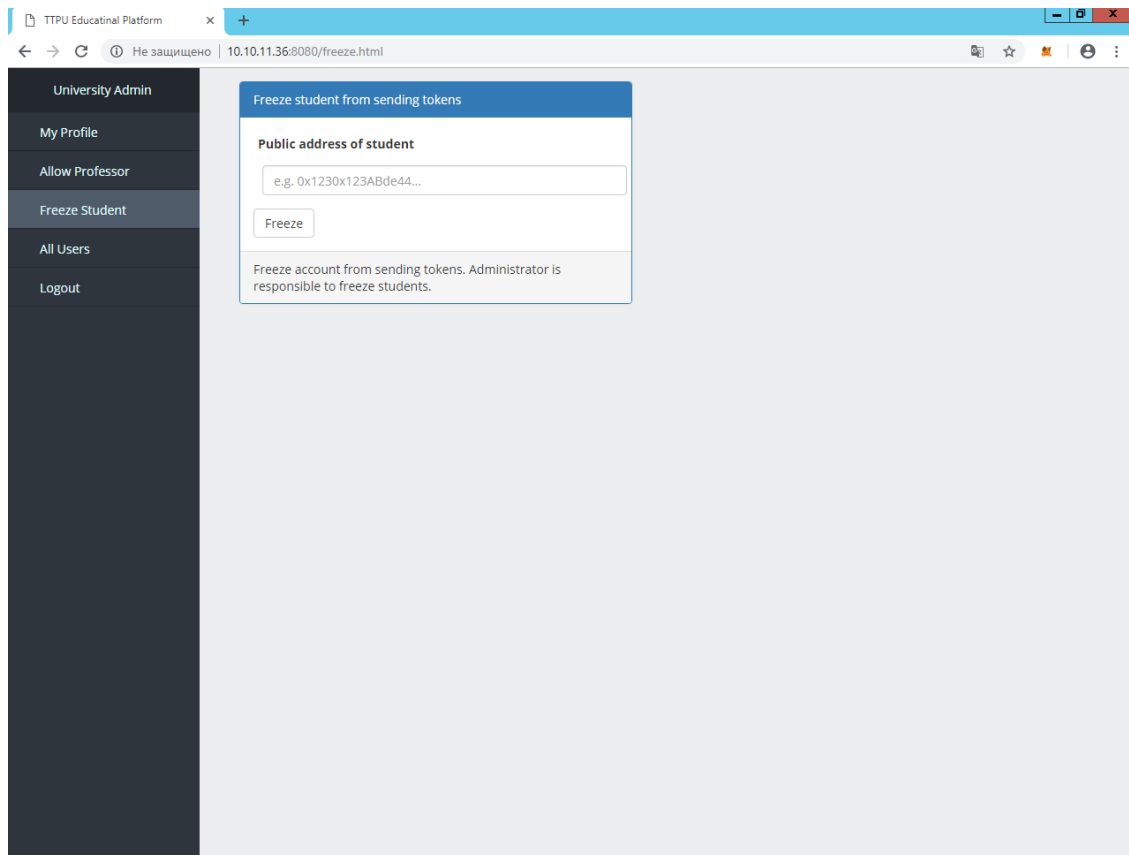


Figure 3.7

3.2.8 Implementation of Professor Account

We have implemented professor account page in order to allow professor to send tokens (marks) to students from their subjects. Professor account can access to the following pages:

- **User information.** There are information about user (his public address name, surname, username).
- **My courses** On this page, professor will get all the students from his/her subjects and mark each of them. Figure 3.10 - 3.11

User Matricula	Public Address	Name	Surname	Username	Role
0001	0x41cf7242812aa8eeb468e6bf4af5b249881dd9a1	Luca	Ardito	Luca Ardito	admin
1004	0xb6d1ffdc4360b606cb313353a2dbed5e84a625c4	Morisio	Maurizio	Morisio Maurizio	professor
1006	0x7ad54af43944189aeb2b025577e1507e6b22807a	Claudio	Sansoe'	Sansoe' Claudio	professor
1002	0x015ded8e51d4816ebe39d073a350a1181d1cb898	Igor	Stievano	Stievano Igor	professor
1003	0x4c7bf106982b535044178708e82fab20a7efba82	Giovanni	Ummarino	Ummarino Giovanni	professor
1001	0xd420e977b6263f38363e16e376901795cd98e41c	Giovanni	Squellero	Squellero Giovanni	professor
1011	0x103681affaa5ec3f896513d93609f6a2981aed1	Marchetto	Guido	Marchetto Guido	professor
1005	0x1d8fc5657a0b52e5895ffe5b97d94bf03cb15624	Roberto	Fontana	Fontana Roberto	professor
1007	0x1e2f29bdc533de50a0eee20b65f7e4baa31abc9	Giovanni	Risso Fulvio	Risso Fulvio Giovanni	professor
1008	0x2977dd44edafdb2943c620f72d5d9f3dba81713a	Diego	Regruto Tomalino	Regruto Tomalino Diego	professor
1009	0x40e8aa1beac5fafa79b4ccf9bf27bd363c3d6619	Francesco	Grigoretti	Grigoretti Francesco	professor
1010	0xa6ea34ade3d29bc6db84daf268ce280770bb9c1a	Sabrina	Ugazio	Ugazio Sabrina	professor
04001	0xd426c976378af5439058e3545bde46a019f30178	Javohir	Abduganiev	Javohir Abduganiev	student
04174	0x286d17b3075d4d76614c93be54d5e6e53b804e5e	Islombek	Durdiev	Islombek Durdiev	student
05001	0x4f4c1e8c94430dbeee67f6de407ec7fd61502c96	Shohjahon	Abdirayimov	Shohjahon Abdurayimov	student
05169	0xed687f491e6fd44c43c023032e8d75ec14b85de4	Farxodbek	Qahhorov	Farxodbek Qahhorov	student
06009	0x359fbc5f18fba0d0d8a5d5a8a1ca487074458d42	Shukurkhon	Akhrokhonov	Shukurkhon Akrokhonov	student
06133	0xadbf5d4ec08c600e6c6ae0efe71c5340b487a90e	Siraj	Madaminov	Siraj Madaminov	student
06167	0xdd51e73db1e362a52581474b20a6ca7973f39c1	Muattaroy	Bakhrudinova	Muattaroy Bakhrudinova	student
06186	0x9887cb6b343b118aef605daf153236563cb97c4c	Safarmurod	Yuldoshev	Safarmurod Yuldoshev	student
07020	0x17c17e9eaa7eaaa1e3c922064ade3e0e27ecc26	Islombek	Jlenbaev	Islambek Jlenbaev	student

Figure 3.8

3.2.9 Implementation of Student Account

Students can access to their accounts and see the tokens (marks) from their subjects. Student account has the following pages:

- **User information.** There are information about user (his public address name, surname, username).
- **My courses** On this page, student will have list of his/her subjects and corresponding tokens(marks). Figure 3.12

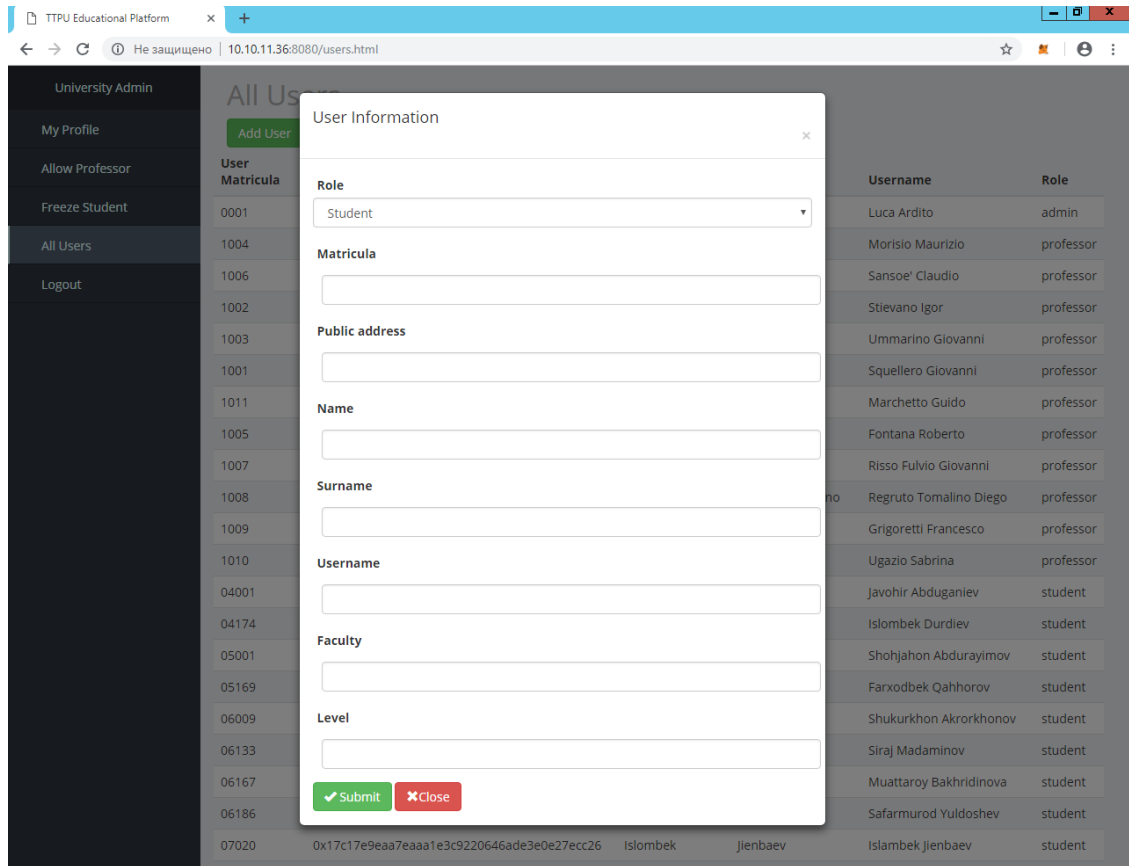


Figure 3.9

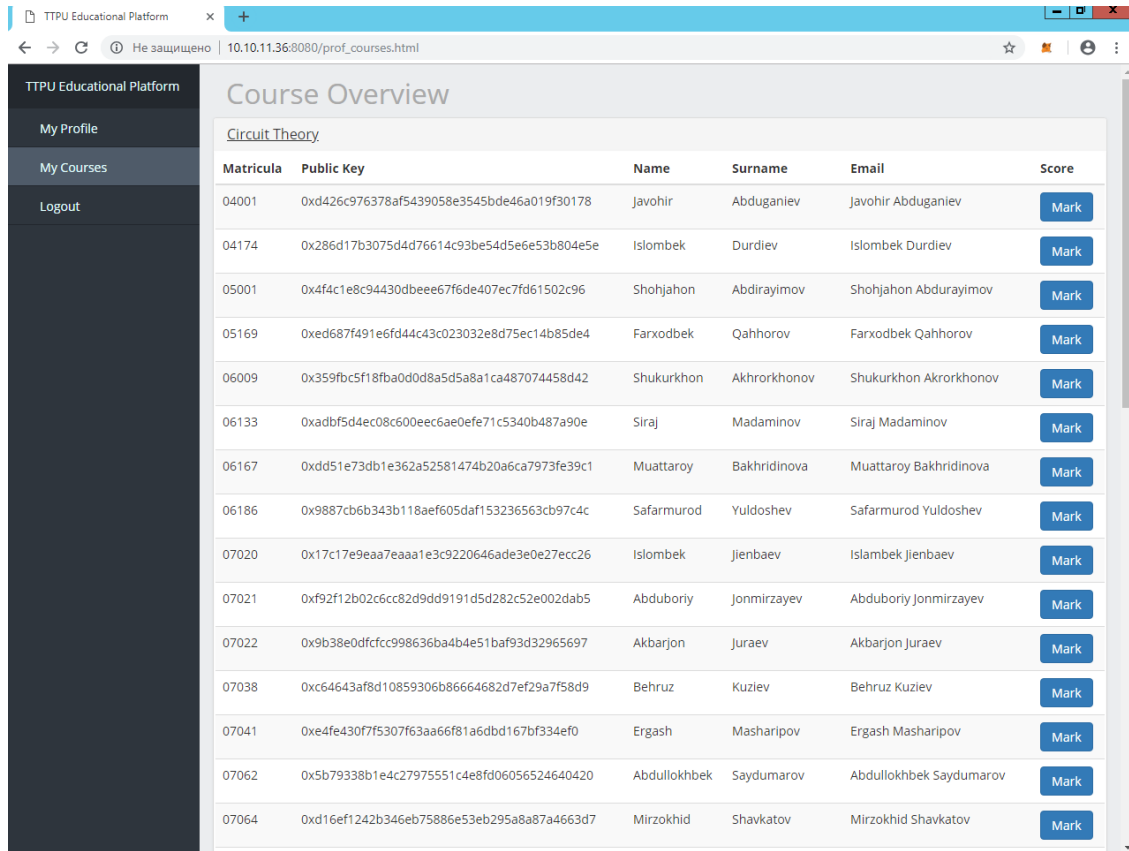
3.2.10 Front-End implementation

To implement our front-end we have used HTML5 and react.js languages. We have to install web3 in order to connect our browser and blockchain. Web3 offers us a great functionalities to interact with our contracts. In order to interact our front-end with smart contract we have implemented app.js javascript file. App.js file has different functions:

Freeze the student's address

In order to freeze student's address from sending tokens to other blockchain addresses we have used `addAddressToFreeze()` function.

3.2 – The Proposed Platform



Matricula	Public Key	Name	Surname	Email	Score
04001	0xd426c976378af5439058e3545bde46a019f30178	Javohir	Abduganiev	Javohir Abduganiev	Mark
04174	0x286d17b3075d4d76614c93be54d5e6e53b804e5e	Islombek	Durdiev	Islombek Durdiev	Mark
05001	0x4f4c1e8c94430dbeee67f6de407ec7fd61502c96	Shohjahon	Abdirayimov	Shohjahon Abdurayimov	Mark
05169	0xed687f491e6fd44c43c023032e8d75ec14b85de4	Farxodbek	Qahhorov	Farxodbek Qahhorov	Mark
06009	0x359fbc5f18fba0d0d8a5d5a8a1ca487074458d42	Shukurkhon	Akhrokhonov	Shukurkhon Akrokhonov	Mark
06133	0xadbf5d4ec08c600e6cae0efe71c5340b487a90e	Siraj	Madaminov	Siraj Madaminov	Mark
06167	0xdd51e73db1e362a52581474b20a6ca7973fe39c1	Muattaroy	Bakhrudinova	Muattaroy Bakhrudinova	Mark
06186	0x9887cb6b343b118aef605daf153236563cb97c4c	Safarmurod	Yuldoshev	Safarmurod Yuldoshev	Mark
07020	0x17c17e9eaa7eaaa1e3c9220646ade3e0e27ecc26	Islombek	Jienbaev	Islambek Jienbaev	Mark
07021	0xf92f12b02c6cc82d9dd9191d5d282c52e002dab5	Abduboriy	Jonmirzayev	Abduboriy Jonmirzayev	Mark
07022	0x9b38e0dfcc998636ba4b4e51baf93d32965697	Akbarjon	Juraev	Akbarjon Juraev	Mark
07038	0xc64643af8d10859306b86664682d7ef29a7f58d9	Behruz	Kuziev	Behruz Kuziev	Mark
07041	0xe4fe430f7f5307f63aa66f81a6dbd167bf334ef0	Ergash	Masharipov	Ergash Masharipov	Mark
07062	0x5b79338b1e4c27975551c4e8fd06056524640420	Abdullokhbek	Saydumarov	Abdullokhbek Saydumarov	Mark
07064	0xd16ef1242b346eb75886e53eb295a8a87a4663d7	Mirzokhid	Shavkatov	Mirzokhid Shavkatov	Mark

Figure 3.10

```

addAddressToFreeze: function(){
var addressToFreeze = document.getElementById("
    inputAddressToFreeze").value;
var tokenInstance;
for(var i=0; i<n; i++){
TokenContract[i].deployed().then(function(instance){
tokenInstance = instance;
return tokenInstance.freezeAccount(addressToFreeze ,
    true, {from: account});
}).then(function(){

```

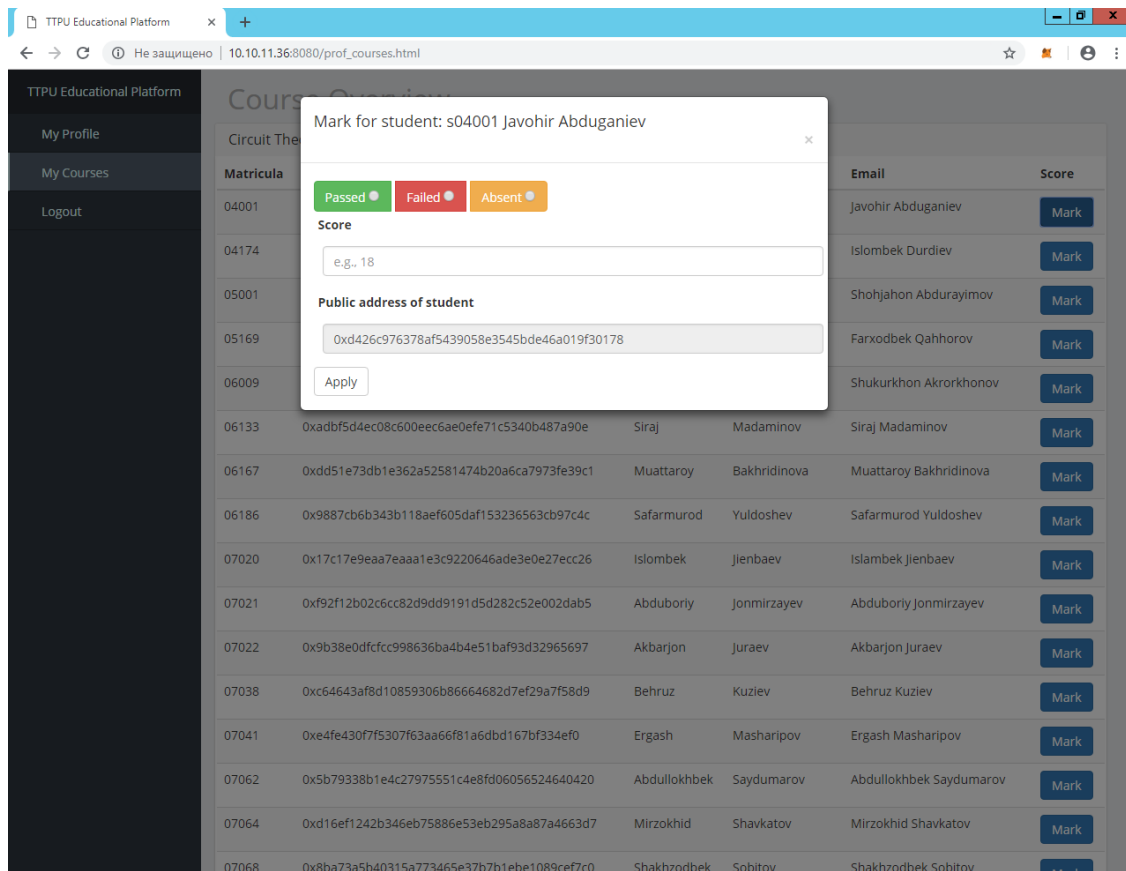
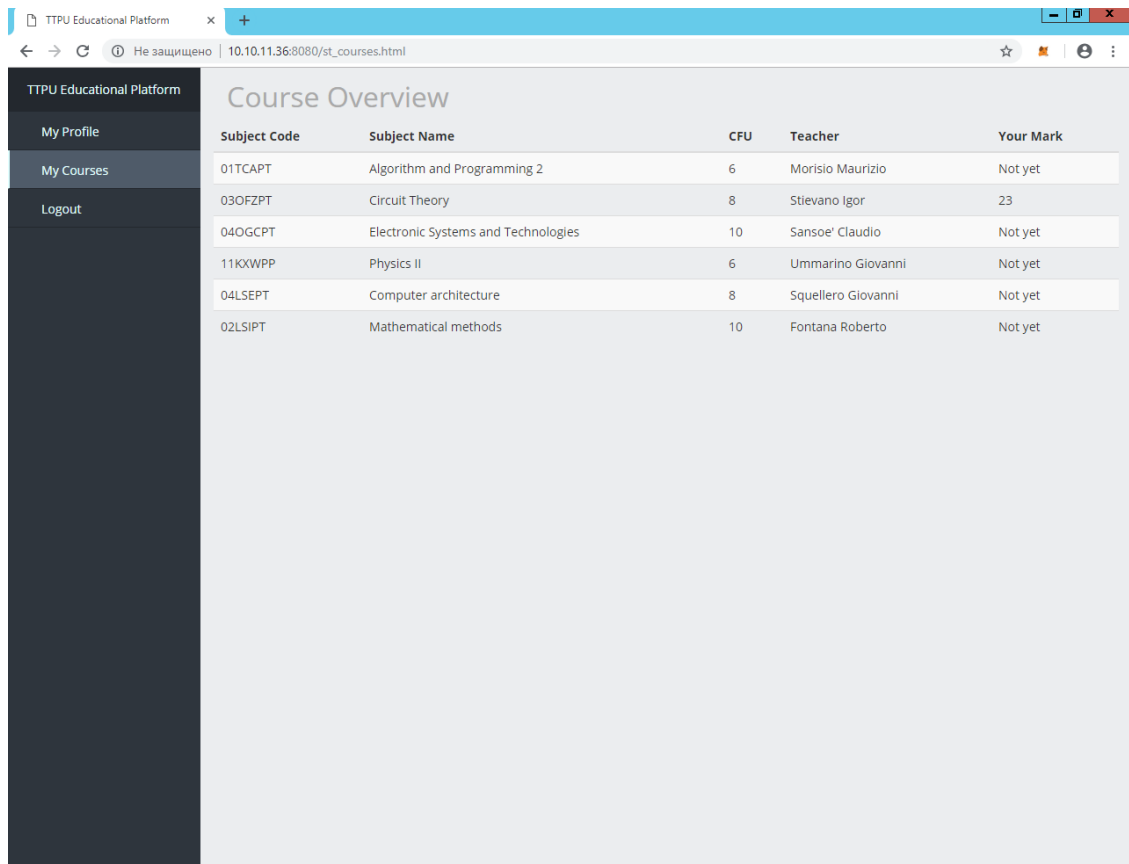


Figure 3.11

```
App.setStatus("Address Freezed");
}).catch(function(e){
document.write(e);
console.log(e);
App.setStatus("Error Freezing Address; see log.");
}); }
},
```

3.2 – The Proposed Platform



Subject Code	Subject Name	CFU	Teacher	Your Mark
01TCAPT	Algorithm and Programming 2	6	Morisio Maurizio	Not yet
03OFZPT	Circuit Theory	8	Stievano Igor	23
04OGCPT	Electronic Systems and Technologies	10	Sansoe' Claudio	Not yet
11KXWPP	Physics II	6	Ummarino Giovanni	Not yet
04LSEPT	Computer architecture	8	Squellero Giovanni	Not yet
02LSIPT	Mathematical methods	10	Fontana Roberto	Not yet

Figure 3.12

Allow the professor's account to send token

Administrator of the platform using `allowanceToken()` function will allow professor address to send token from his/her address to the student address.

```
allowanceToken: function(){  
var amount = parseInt(document.getElementById("  
    inputAmountAllowanceToken").value);  
var receiver = document.getElementById("  
    inputBeneficiaryAllowanceToken").value;
```

```
this.setStatus("Initiating transaction... (please  
    wait)");  
var tokenInstance;  
TokenContract.deployed().then(function(instance){  
    tokenInstance = instance;  
    return tokenInstance.approve(receiver, amount, {from  
        : account});  
}).then(function(){  
    App.setStatus("Transaction complete!");  
    App.updateTokenBalance();  
}).catch(function(e){  
    console.log(e);  
    App.setStatus("Error sending coin: see log.");  
}); } };
```

Sending tokens

Professor send tokens to the student as their grades. It is possible using the function `sendTokenFrom()` which sends tokens from an administrator's address.

```
sendTokenFrom: function(contractID){  
var amount =parseInt(document.getElementById("  
    inputAmountSendToken").value);  
var receiver = document.getElementById("  
    inputBeneficiarySendToken").value;
```

```
var owner = "0
    x41cf7242812aa8eeb468e6bf4af5b249881dd9a1";
var mark = document.getElementsByName("mark");

var tokenInstance;
return TokenContract[contractID].deployed().then(
    function(instance){
tokenInstance = instance;

for(var i = 0; i < mark.length; i++){
if(mark[i].checked){
if(mark[i].value == "passed"){
return tokenInstance.transferFrom(owner, receiver,
    amount, {from: account});
}else if(mark[i].value == "failed"){
amount = 1;
return tokenInstance.transferFrom(owner, receiver,
    amount, {from: account});
}else{
amount = 2;
return tokenInstance.transferFrom(owner, receiver,
    amount, {from: account});
}}}}).then(function(){
App.setStatus("Transaction complete");
App.updateTokenBalance();
}).catch(function(e){
console.log(e);
App.setStatus("Error sending coin; see log.");}); },
```

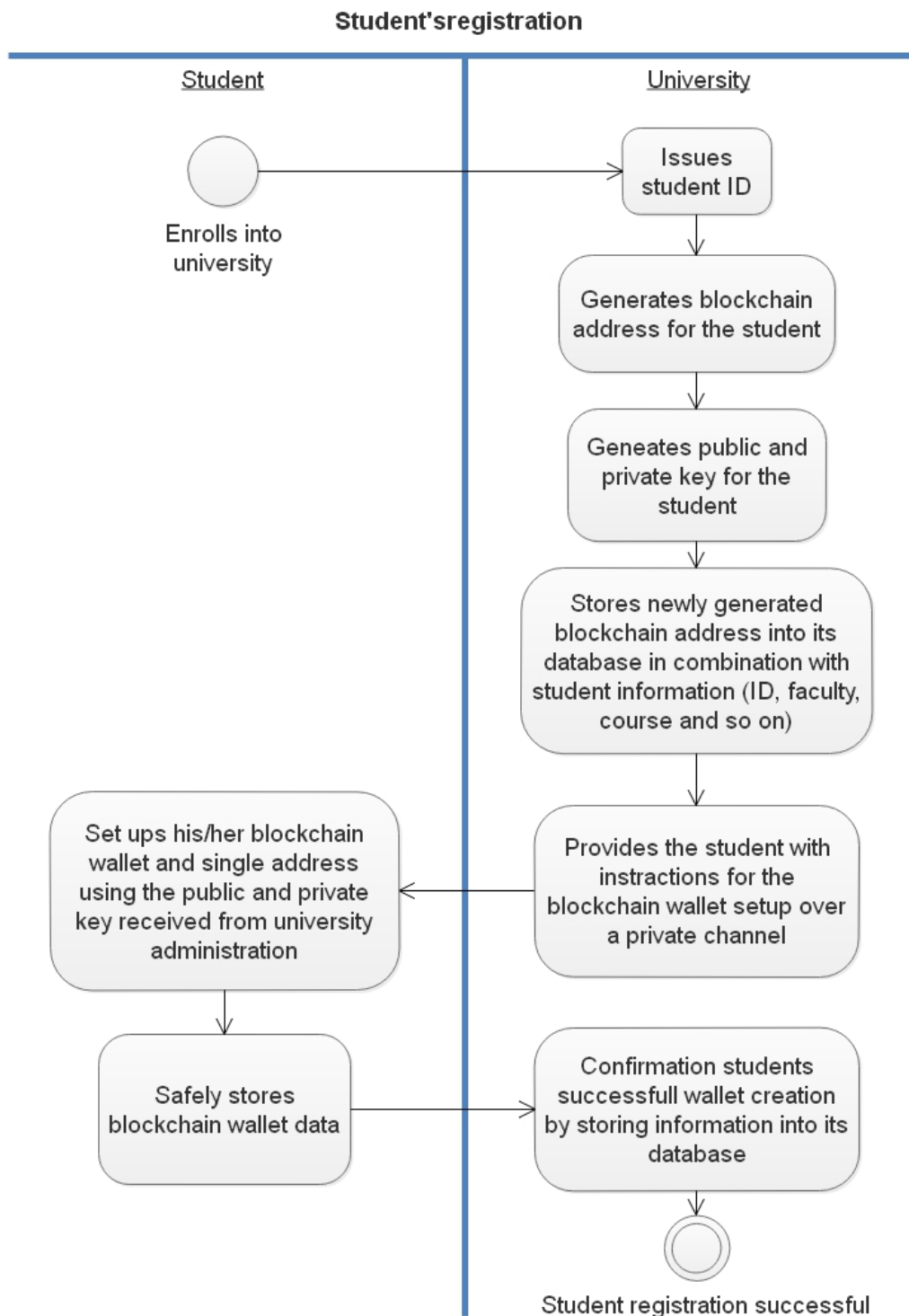


Figure 3.13: A process model of a student's registration into the blockchain network.

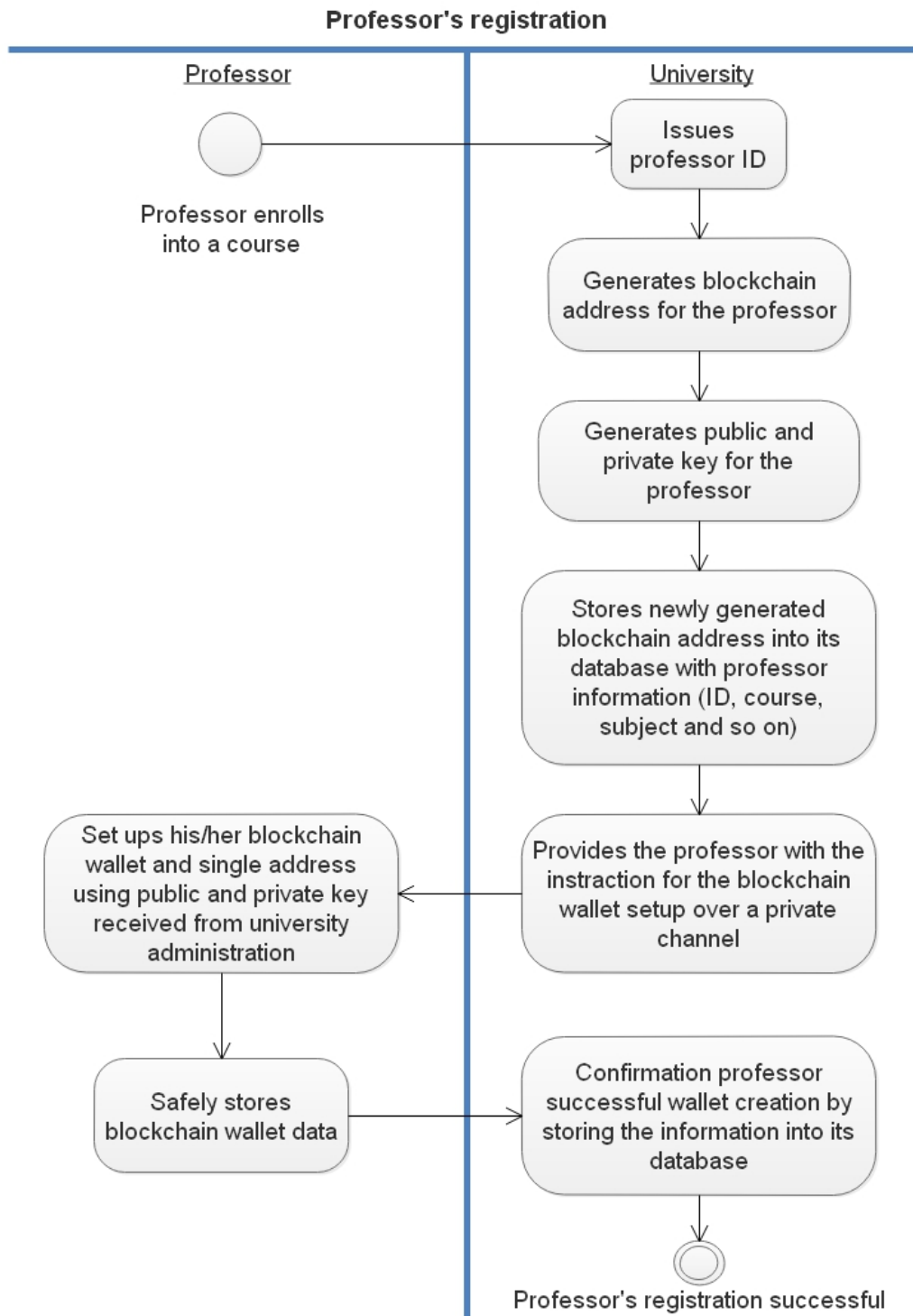


Figure 3.14: A process model of a professor's registration into the blockchain network.

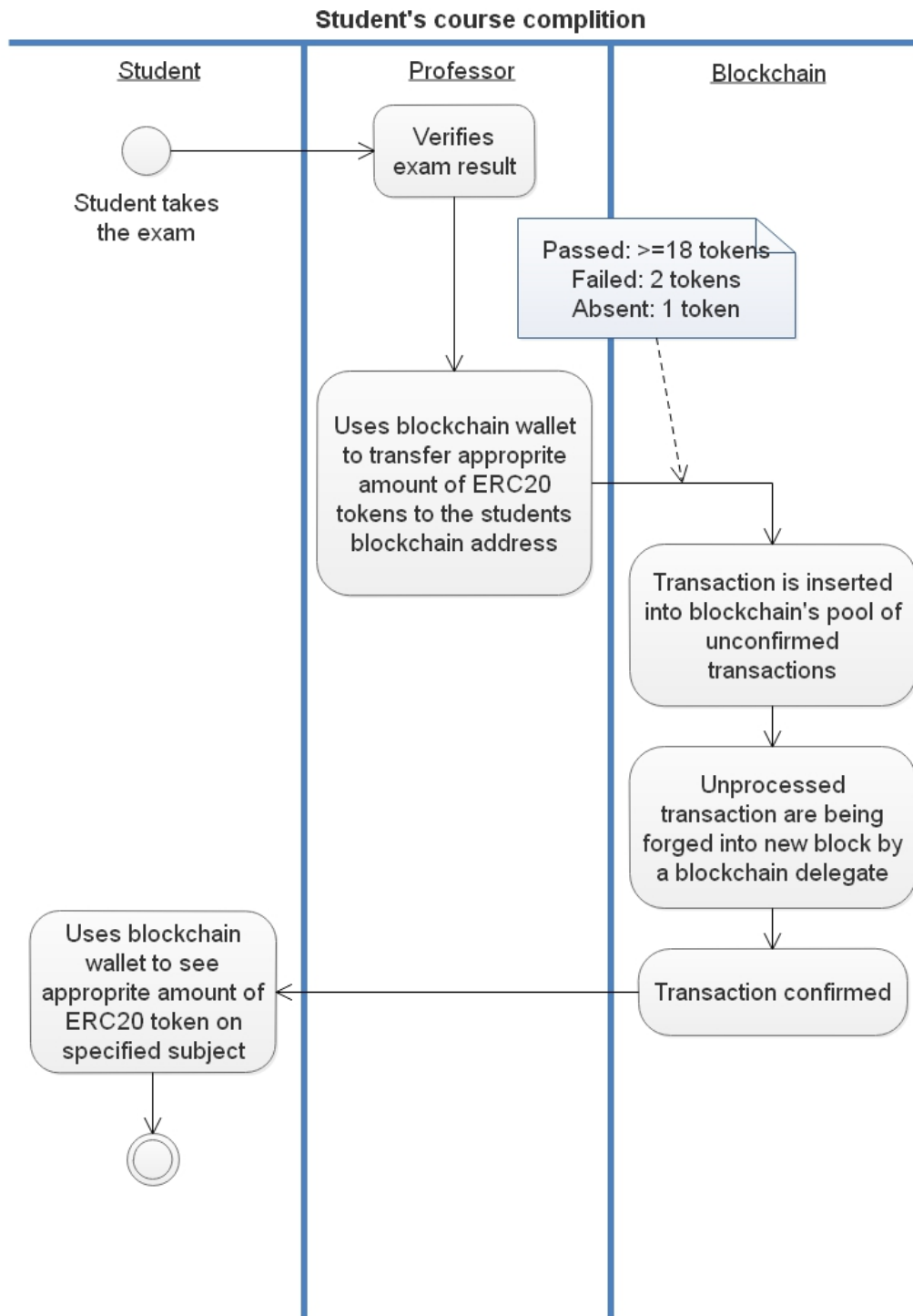


Figure 3.15: A process model of a course grading system with blockchain network.

Chapter 4

Experimentation in TTPU

Our project has been implemented in Turin polytechnic university in Tashkent (TTPU) as a real test case. We have created private blockchain in our local server. All users can access to the blockchain using their public key and private key. In our private blockchain we have deployed token ERC20 smart contract for each subject with their symbol name as subject ID. After deployment all tokens will be on administrators account. Then we have created accounts for each professors and for each students. Now all the accounts are in the blockchain. They all have public key and private keys. In our project we have integrated mysql database to our project. Administrator will create each user information in mysql database table. We need these information in order differentiate between professors and students. Moreover, inside mysql database we have saved information about subjects for professors and students. The public addresses of the user has been saved as well (for one click login system).

We have done experimentation in TTPU with following information:

- **Academic year 2017/2018.** As far as we do not have any result for current academic year 2018/2019. We have chosen previous academic

year for our implementation.

- **2nd and 3rd year IT faculty.** For our experimentation, we have gotten information about previous year 2nd and 3rd year students and marks on corresponding subjects.

there has been involved 30 students from second level and 30 students from third level and We created overall 60 nodes (wallets) for students. Overall we made a token for each thirteen subjects. We have created ERC20 token for following subjects:

	Subject Name	Symbol name	Level
1.	Algorithm and Programming 2	01TCAPT	2
2.	Circuit Theory	03OFZPT	2
3.	Electronic Systems and Technologies	04OGCPT	2
4.	Physics II	11KXWPP	2
5.	Computer Architecture	04LSEPT	2
6.	Database	05LSZPT	2
7.	Mathematical Methods	02LSIPT	2
8.	Object oriented programming	01SXPPT	3
9.	Computer networks	07KSIPT	3
10.	Operating systems	06JEZPT	3
11.	Automatic controls	01SXOPT	3
12.	Applied electronics and measurements	04OGHPT	3
13.	Signal analysis and processing	01OGGPT	3

For 2 level (30) IT students there are 7 subjects: $30 \times 7 = 210$ transactions in order to transact grades for students. For 3 level (30) IT students there are

6 subjects: $30 \times 6 = 180$ transactions. After sending the tokens (marks) to the students. All the transaction information will be stored in the blockchain system. It will not be possible to change or delete them.

We spent about a minute to create a user in the private blockchain and a minute to store user information into centralized database. So, in order to create 60 students in our private blockchain and store information related to students we needed two hours of time. A professor accesses to their wallet and send tokens (marks) to students address. For each transaction to be completed, a professor needs maximum 20 seconds. So, overall a professor can spend their 10 minutes for transact grades for 30 students.

Using our implementation, all information about user have been saved effectively and securely. From the one side, professor can do transaction (send tokens) to student accounts for assigning score for his/her subject. From the other side, student can see their balance (tokens) on different subject which are in his/her academic year.

Chapter 5

Conclusion

By building this proposed token ERC20 smart contract of ours, we have succeeded in moving academic careerer platform to the blockchain platform and we addressed some of the fundamental issues that current database system has, by using power of Ethereum network and the blockchain structure. As a result of our project, the concept of blockchain and the security methodology which it uses, namely tokens, has become adoptable to grades. This achievement may even pave the way for other blockchain applications that have impact on every aspect of human life. At this point, Ethereum and the smart contracts, which made one of the most revolutionary breakthroughs since the blockchain itself, helped to overturn the limited perception of blockchain as a cryptocurrency (coin), and turned it into a broader solution-base for many Internet-related issues of the modern world, and may enable the global use of blockchain.

Grading students is still a controversial topic within academic circle. Despite the existence of several very good examples, most of which are still in use; many more attempts were either failed to provide the security and privacy features of a traditional grading system or have serious usability and

scalability issues. On the contrary, blockchain-based grading system solutions, including the one we have implemented using the token ERC20 smart contracts and the Ethereum network, address (or may address with relevant modifications) almost all of the security concerns, like privacy of users, integrity, verification, authentication and transparency of grades.

In the future, we plan to test our prototype with other universities connecting them in one private blockchain. In this scenario all control will be on government who will be responsible of all tokens. Moreover, we would like to implement solution for verification of the ECTS credit system.

Bibliography

- [1] Nolan Bauerle "*What is a Distributed Ledger?*" 2017 [Online] Available: <https://www.coindesk.com/information/what-is-a-distributed-ledger/>

- [2] Melanie Swan "*Blockchain: Blueprint for a new economy.*" Book Preface X Available: <http://shop.oreilly.com/product/0636920037040.do>

- [3] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, Huaimin Wang "*An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends.*" 2017 IEEE 6th International Congress on Big Data. Available: <https://ieeexplore.ieee.org/document/8029379>

- [4] V. Buterin "*On public and private blockchains.*" 2015 [Online] Available: <http://blog.ethereum.org/2015/08/07/on-public-and-private-blockchain>

- [5] "*What is Blockchain Technology? A Step-by-Step Guide For Beginners.*" [Online] Available: <https://blockgeeks.com/guides/what-is-blockchain-technology/>

- [6] Larry Summers *"Blockchain Can Succeed Without Bitcoin."* 2016 [Online] Available: <https://www.coindesk.com/larry-summers-us-treasury-blockchain-can-succeed-without-bitcoin/>
- [7] William Mougayer *"Explaining the Blockchain via a Google Docs Analogy."* 2016 [Online] Available: <http://startupmanagement.org/2016/09/06/explaining-the-blockchain-via-a-google-docs-analogy/>
- [8] Gavin Wood *"Ethereum: A secure decentralised generalised transaction ledger. EIP-150 Revision."* Available: <https://gavwood.com/paper.pdf>
- [9] Dejan Vujicic, Shaoan Xie, Randjic Sinisa *"Blockchain technology, bitcoin, and Ethereum: A brief overview."* March 2018 Conference Paper ResearchGate. Available: <https://ieeexplore.ieee.org/document/8345547/>
- [10] *"A Beginner's Guide to Smart Contracts."* [Online] Available: <https://blockgeeks.com/guides/smart-contracts/>
- [11] K. Delmolino, M. Arnett, A. Kosba, A. Miller and E. Shi *"Step by Step Towards Creating a Safe Smart Contract: Lessons and Insights from a Cryptocurrency Lab."* Available: https://www.researchgate.net/publication/307507489_Step_by_Step_Towards_Creating_a_Safe_Smart_Contract_Lessons_and_Insights_from_a_Cryptocurrency_Lab
- [12] *"Solidity."* [Online] Available: <https://solidity.readthedocs.io/>

[en/v0.4.25/index.html](#)

- [13] "MetaMask." [Online] Available: <https://metamask.io/>

- [14] Narayan Prusty "*Building Blockchain Projects*" April 2017 Book Available: <https://www.oreilly.com/library/view/building-blockchain-projects/9781787122147/>

- [15] D.Johnston, S.Yilmaz, J.Kandah, N.Bentenitis, F.Hashemi, R.Gross, Sh.Wilkinson and S.Mason "*The General Theory of Decentralized Applications, DApps*" Book 2015 Available: <https://github.com/DavidJohnstonCEO/DecentralizedApplications>

- [16] Daniel Palmer "*7 Cool Decentralized Apps Being Built on Ethereum*" Online Available: <https://www.coindesk.com/7-cool-decentralized-apps-built-ethereum/>

- [17] Alex Roehrs, Cristiano Andrea da Costa, Rodrigo da Rosa Righi "*OmnipHR: A distributed architecture model to integrate personal health records.*" Journal of Biomedical Informatics 71 (2017) 70-81 Available: https://www.researchgate.net/publication/317112353_OmniPHR_A_Distributed_Architecture_Model_to_Integrate_Personal_Health_Records

- [18] Dr. Onkar Kemkar, Priti Kalode "*Formulation of Distributed Electronic Patient Record (DEPR) System Using Openemr Concept*" International Journal of Engineering Innovation and Research Volume 4, Issue 1, ISSN: 2277 - 5668 Available:

https://ijeir.org/administrator/components/com_jresearch/files/publications/IJEIR-1328_final.pdf

- [19] Qi Xia, Emmanuel Sifah, Kwame Omono Asamoah, Jianbin Gao, Xiaojiang Du, Mohsen Guizani *"MeDShare: Trust-Less Medical Data Sharing Among Cloud Service Providers via Blockchain"* IEEE July 24, 2017. Available: <https://ieeexplore.ieee.org/document/7990130/>

- [20] H. Hou *"The application of blockchain technology in E-government in China"* Proc. 26th Int. Conf. Comput. Commun. Netw. (ICCCN), Jul./Aug. 2017, pp. 1-4. Available: <https://ieeexplore.ieee.org/document/8038519/>

- [21] S. Olnes and A. Jansen *"Blockchain Technology as s Support Infrastructure in e-Government."* Springer, Sep. 2017, pp. 215-227. Available: https://www.researchgate.net/publication/318879959_Blockchain_Technology_as_s_Support_Infrastructure_in_e-Government

- [22] V. Morabito *"Blockchain Governance"* Springer, 2017, pp. 41-59. Available: https://www.researchgate.net/publication/313021355_Blockchain_Governance

- [23] R. Qi, C. Feng, Z. Liu and N. Mrad *"Blockchain-Powered Internet of Things, E-Governance and E-Democracy."* Springer, 2017, pp. 509-520. Available: https://www.researchgate.net/publication/317226195_Blockchain-Powered_Internet_of_Things_E-Governance_and_E-Democracy

- [24] Ali Kaan Koc, Umut Can Cabuk, Emre Yavuz, and Gokhan Dalkilic. *"Towards Secure E-Voting Using Ethereum Blockchain."* IEEE 2017. Available: <https://ieeexplore.ieee.org/document/8355340>

- [25] Damiano Di Francesco Maesa, Paolo Mori, Laura Ricci. *"Blockchain Based Access Control."* IEEE. Available: https://www.iit.cnr.it/sites/default/files/main_21.pdf

- [26] Ali Dorri, Salil Kanhere, Raja Jurdak, Praveen Gauravaram. *"Blockchain for IoT Security and Privacy: The Case Study of a Smart Home."* ResearchGate 2017. Available: <https://ieeexplore.ieee.org/document/7917634>

- [27] Media Lab Learning Initiative *"Digital Certificates Project."* [Online] Available: <http://certificates.media.mit.edu/>

- [28] Franco Amati *"First official career diplomas on Bitcoin's blockchain"* [Online] 2015 Available: <https://blog.signatura.co/first-official-career-diplomas-on-bitcoin-s-blockchain-69311acb544d>

- [29] Rebecca Campbell *"Holberton School Begins Tracking Student Academic Credentials on the Bitcoin Blockchain"* [Online] 2016 Available: <https://bitcoinmagazine.com/articles/>

- [30] Luke Graham *"Schools are using bitcoin technology to track students"* [Online] 2016 Available: <https://www.cnn.com/2016/05/09/schools-are-recording-students-results-on-the-blockchain>.

[html](#)

- [31] Bitcoin education "*Parisian Engineering School Will Certify Diplomas on the Blockchain*" 2016 Available: <https://www.ccn.com/parisian-engineering-school-will-certify-diplomas-blockchain/>

- [32] M. Turkanovic, M. Holbl, K. Kotic, M. Hericko, and A. Kamisalic "*EduCTX: A Blockchain-Based Higher Education Credit Platform*" IEEE February 28, 2018. Available: <https://ieeexplore.ieee.org/document/8247166>

- [33] Steve Chan "*Blockchain Based Professional Networking and Recruiting Platform*" Available: <https://icos.icobox.io/uploads/whitepaper/2017/11/59f9568215146.pdf>