Software Design Document For Digital Certificates Using Blockchain

Alley El-Dorry, Sherif Abd El Khalek, Mohamed Reda, Shehab El-Din Mohamed

March 19, 2020

1 Introduction

1.1 Purpose

This Software Design Document was created with the purpose of explaining the how each component interact with each other and also how our system's architecture design is done. The system's purpose is to prevent the forgery of the Egyptian academic certificates and making it easier for students to issue their certificates than the traditional way, this is done using the blockchain technology using Linux Foundation's Hyperledeger Fabric [1]. This document will define how each sub-component of the system interact with each other and also providing an overview of the system. Therefore, this Software Design Document is mainly for Stakeholders and Developers.

1.2 Scope

The Digital Certificates Using Blockchain system is designed to target students, universities, Ministry of Higher Education and companies. The main scope of this system is to take advantage of the main characteristics of blockchain which make it secure. Students will be able to request their certificates and receive it. Companies will be able to verify the certificates of their employees. Private universities will take approval of the Ministry of Education before issuing a certificate for the student. This process my be expanded in the future by adding more universities and companies to the system.

1.3 Overview

The proposed system aims at creating digital certificates for Egyptian higher education as a solution for the forgery problem. The proposed system enables the verifying of a student's academic certificates; hence, the organisations and institutions can verify the authenticity of the obtained certificates. The proposed

system is implemented using Linux Foundation's Hyperledger Fabric mainly because of its use of smart contract, modular architecture, and scalability.

1.4 Definitions and Acronyms

Term	Definition
CSN	Certificate Serial Number.
MIU	Misr International University.
MOHESR	The Ministry of Higher Education and Scientific Research
CA	Certificate Authority.
PKI	Public Key Infrastructure

2 System Overview

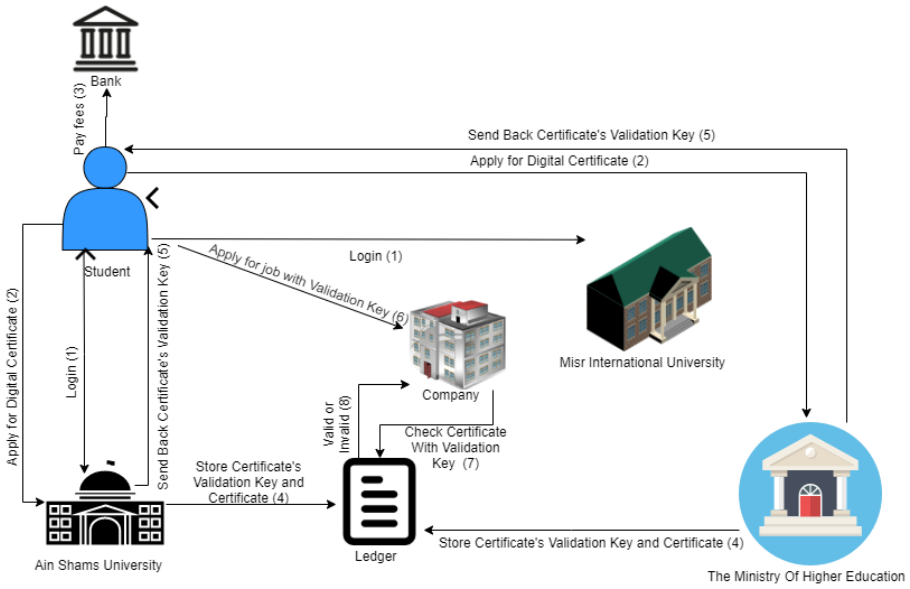

Figure 1: System Overview

As seen in Figure 1 the network consists of 4 types of physical organizations: public universities, private universities, companies and the Ministry of Higher Education and Scientific Research. There are also graduates who are considered as actors with a specific role within their university. Each participant or actor within the network has their specific permissions. Graduates can request their certificate from their university, companies can verify the validity of any certificate. Misr International University (MIU), which is a private university,

has the permission to register its graduates and also can query its graduates certificates but it can't issue certificates since it's a private university. However, Ain Shams University (ASU), which is a public university, has the permission to register its student and issue certificates to them and also can query its graduates certificates. Moreover, the Ministry Of Higher Education can query the whole ledger and can issue certificates to private universities' graduates. The companies will be considered as actors that have the permission of validating a certificate so they can access the validation portal and validate certificates. The transaction data will be the graduates certificates snapshot.
The process start with a student who logs in and requests a certificate and pays a certain fee to complete his request from his university , in case if it it is a public university, the university can directly issue the student's certificate to him, but if it is a private university it will need to request approval from the Ministry Of Higher Education to issue the certificate once it's approved the certificate is issued and once a certificate is issued it will store it and its validation key on the ledger, once the fresh graduate applies for a job in a company , the company will use his validation key to validate his certificate and check for its existence on the ledger.

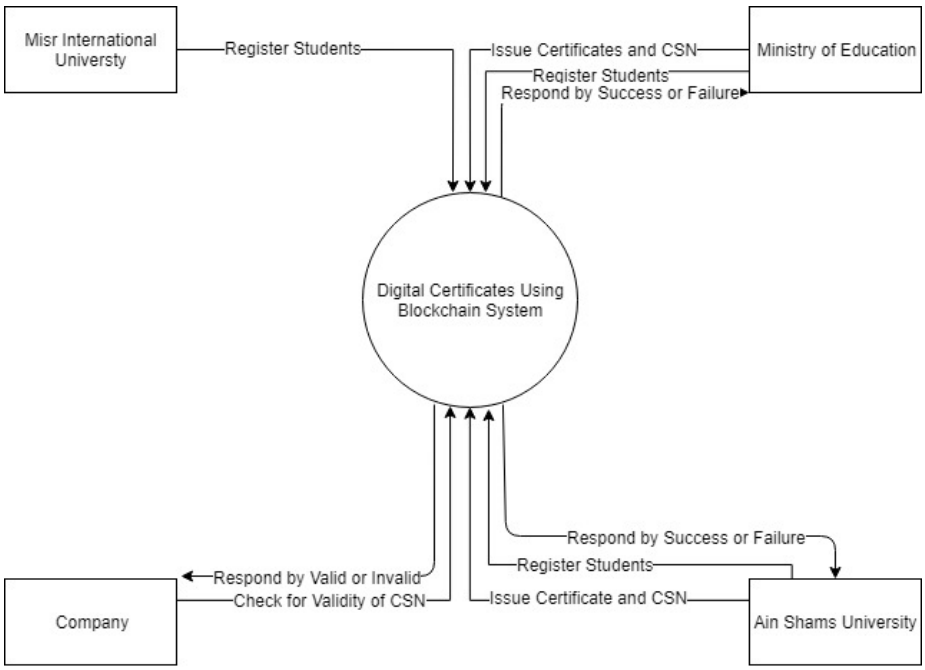

Figure 2: Context Diagram

3 System Architecture

3.1 Architectural Design

The architectural design of the system is represented with the fundamental components of the blockchain and hyperledger fabric technologies.

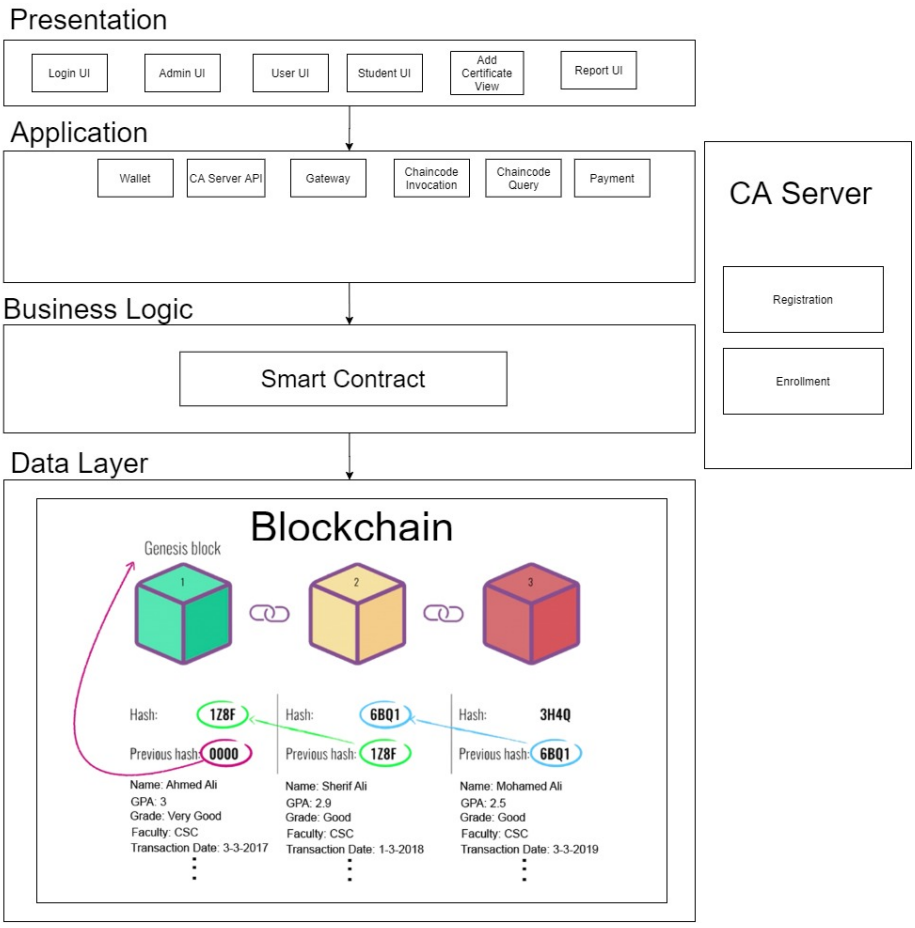

Figure 3: Architectural Design

3.1.1 Presentation

This part is responsible of displaying and representing of the User Interface of the whole system. it's divided to sub-component depending on the functionality, for each view on the system to make it easy to change it later on. The interface will be implemented using the Vue.js framework.

3.1.2 Application

This part includes the proper execution of the application, the wallet holds the identities of each actor, CA server API is the API used for utilizing the CA server, the payment is responsible for handling the transaction's fees and its confirmation, The Gateway component is responsible for handling the blockchain network, the chaincode invocation component is responsible for submitting a transaction by invoking the chaincode, the chaincode query is responsible for reading from the ledger. The chaincode functions will be implemented with Node.js.

3.1.3 Business Logic

The business logic of the system is represented with the smart contract (chaincode) which acts as the heart of the system, since the smart contract is the only component that can access the data layer and manipulate data in it allowing it to change the state of the ledger once a transaction is made.

3.1.4 CA Server

The Certificate Authority server responsible for the features of Enrollment and registration of identities to the system and certificate renewal and revocation.

3.1.5 Data Layer

This part is the root of the system which contains the ledger which stores the crucial data of the system that is stored in a sequence of interconnecting blocks where each block points to the hash of the previous block and each block contain data.

3.2 Decomposition Description

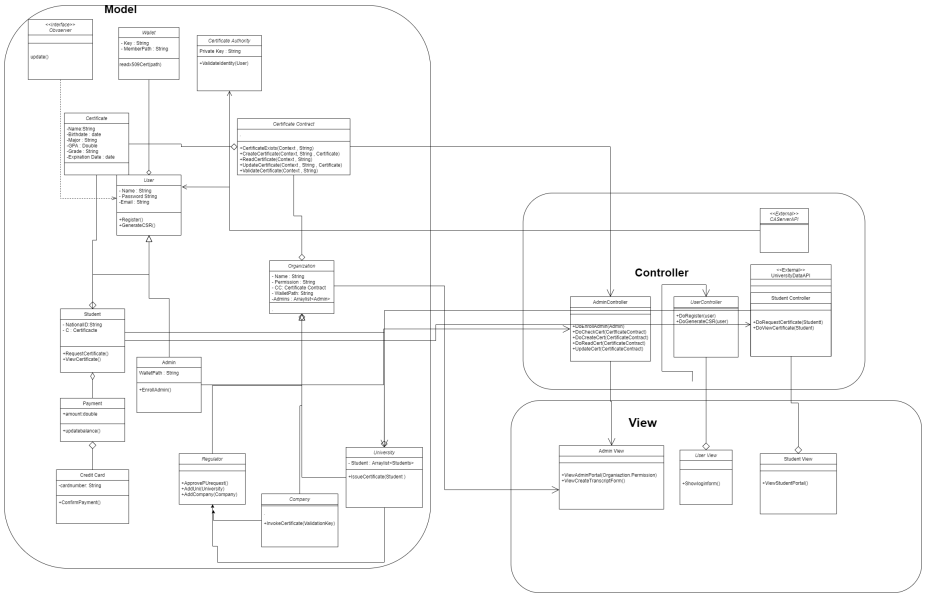

Figure 4: Class Diagram

3.3 Class Descriptions

Class Name	User
Super Classes	N/A
Sub Classes	Student , Admin .
Purpose	Encapsulate the basic attributes of all users in the system.
Collaboration	Aggregates Wallet , implements observer interface.
Functions	Register , GenerateCSR

Class Name	Student
Super Classes	User
Sub Classes	N/A .
Purpose	Presents the student.
Collaboration	Student aggregates Certificate and Payment , extends user class and associates class StudentController.
Functions	RequestCertificate , ViewCertificate.

Class Name	Admin
Super Classes	Student
Sub Classes	N/A .
Purpose	Represents the admin.
Collaboration	extends user class , aggregated Organization.
Functions	CertificateExists , CreateCertificate , ReadCertificate, Upate-Certificate, ValidateCertificate.

Class Name	Organization
Super Classes	N/A
Sub Classes	Regulator , Company , University .
Purpose	Encapsulates the basic information of the the organizations in the network.
Collaboration	Aggregates Admin and Certificate Contract.
Functions	N/A.

Class Name	Regulator
Super Classes	Organization
Sub Classes	N/A.
Purpose	Represents the regulator in the network.
Collaboration	Associates Company and University , extends University.
Functions	ApprovePURequet , AddUni , AddCompany

Class Name	Company
Super Classes	Organization
Sub Classes	N/A .
Purpose	Represents the company in the network.
Collaboration	extends Organization.
Functions	InvokeCertificate.

Class Name	University
Super Classes	Organization
Sub Classes	N/A .
Purpose	Represents the university in the network .
Collaboration	Extends Organization , aggregates Student
Functions	Issue Certificate.

3.3.1 Sequence Diagram

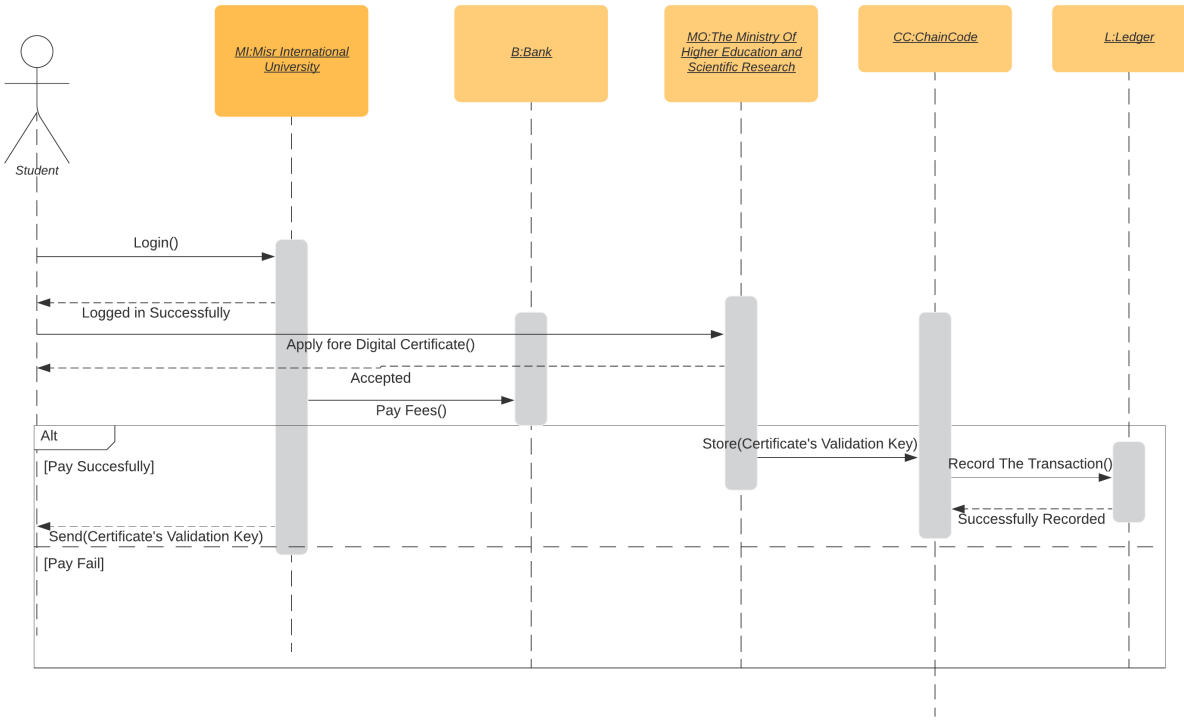

Figure 5: Private University Sequence Diagram

In this diagram, the Student can login the system, he can apply for a digital certificate , once his application is accepted the student pays via credit card then the university , since it is a private university it can't invoke the chaincode directly so it sends a request to the Ministry of higher education, once the request is accepted it proceeds to invoke the chaincode and creates a unique certificate validation key and also adds a new record of the transaction to the ledger , in case of failed payment the university will not proceed further.

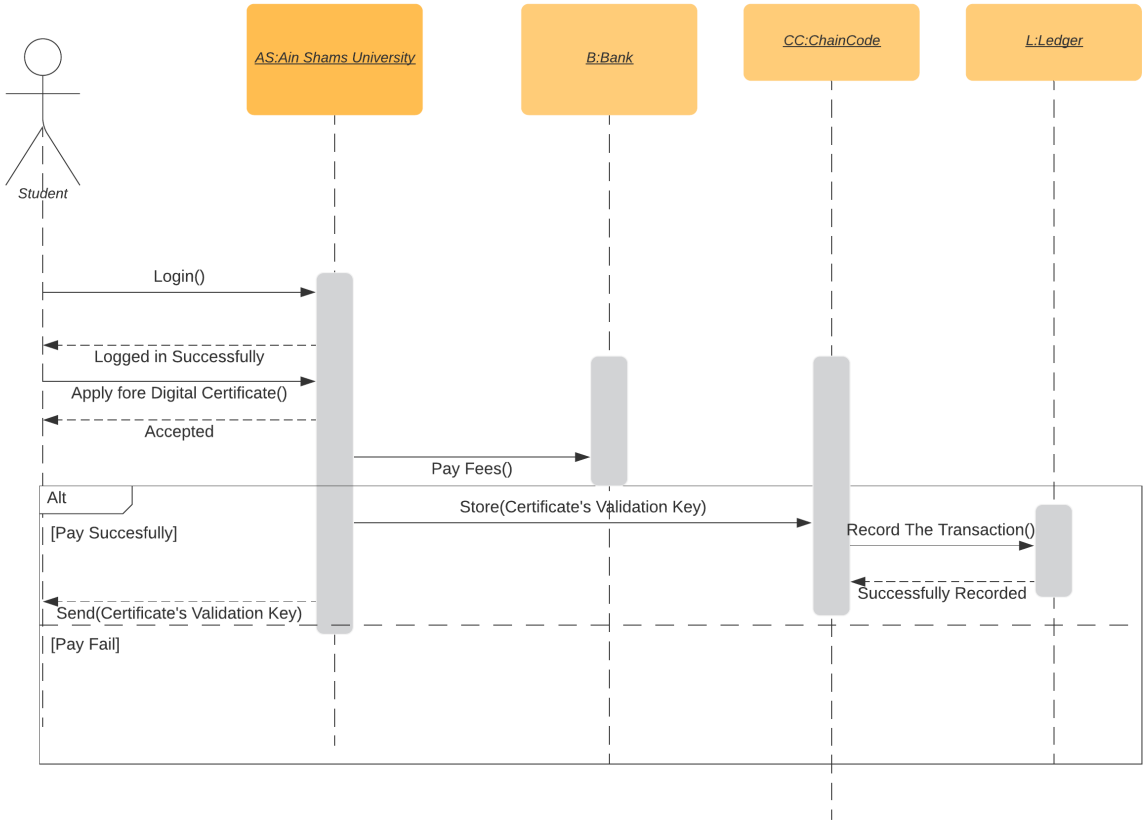

Figure 6: Public University Sequence Diagram

In this diagram, the Student can login the system, he can apply for a digital certificate , once his application is accepted the student pays via credit card then the university invokes the chaincode and creates a unique certificate validation key and also adds a new record of the transaction to the ledger , in case of failed payment the university will not proceed further.

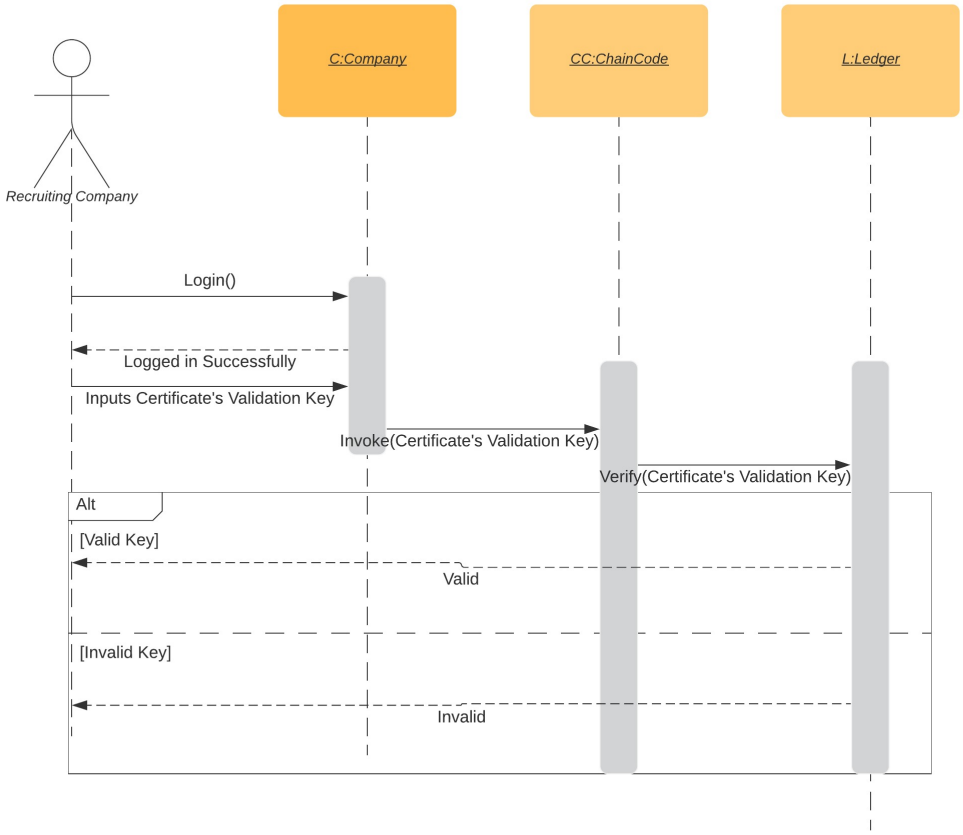

Figure 7: Recruiting Company Sequence Diagram

In this diagram, the Recruiter can login to the system , and he can input Certificate's Validation key of any student that applied for a job,then, the company will invoke the chaincode and query the certificate's validation key, if the

key is valid it will be verified successfully, if it's not will return feedback that is invalid.

3.3.2 Activity Diagram

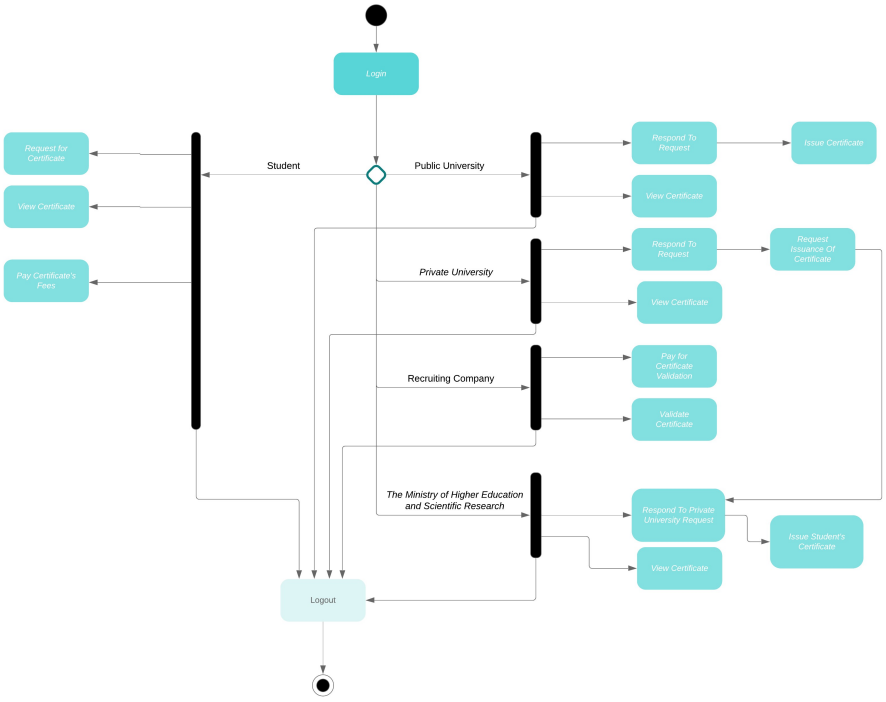

Figure 8: Activity Diagram

3.4 Design Rationale

This project deals with sensitive data of the certificates and user information so it was really important to make modifications to the system easily and efficiently. For our technology used in this project, we used the Linux Foundation's Hyperledger Fabric as it has the Smart Contracts (Chaincode) and because it's a private permissioned blockchain network. Their were other alternatives for the technology used such as:

Hyperledger Indy: [2]

It's a Linux Foundation's project. It's Public , Private or Permissioned network that doesn't feature Smart Contracts(Chaincode). It's mainly built for decentralized identity and doesn't need or contain any mining. We didn't choose this techonlogy because it didn't feature the Smart Contracts(Chaincode) as it was

an important asset of our development.

Ethereum Blockchain: [3]

It's an open source, public, blockchain-based distributed computing platform featuring smart contract functionality. It supports a modified version of Nakamoto consensus via transaction-based state transitions. We didn't choose this technology as it is a public network only and in our project we deal with sensitive information so we couldn't risk using a public network.

4 Data Design

4.1 Data Description

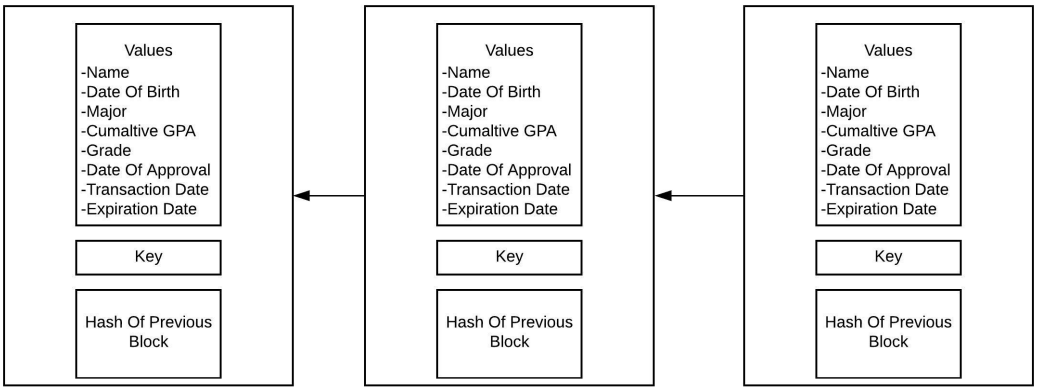

Figure 9: Blocks of data

4.2 Data Dictionary

4.2.1 Values

This part of the block contains the data of the academic certificate (Name, DOB, Major, GPA, etc..)

4.2.2 Key

Each block has a key value which is a unique value for identifying each transaction.

4.2.3 Hash Of Previous Block

Each block (Except the genesis block) points to the hash of the previous block to keep the chain of the blocks connected.

5 Component Design

The system consists of several interconnecting components(Hyperledger Fabric Network, Blockchain Platform, Node.js Server and Client Application). Each component contributes to form the whole system.

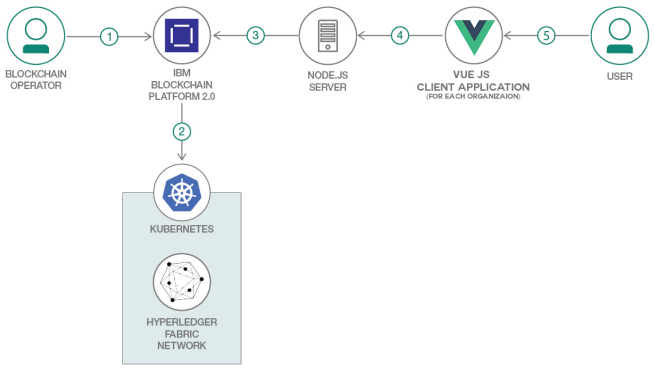

Figure 10: Application Architecture

5.1 Hyperledger Fabric Network

The system's core is the hyperledger fabric network. It provides the system infrastructure that represents the oragnizations participating and provides their users with the ledger service and Smart Contracts. The Kubernetes is used to orchestrate the network's docker containers, and this network is hosted on IBM Blockchain Platform.

5.1.1 Smart Contract (Chaincode)

The Smart Contract contains the implementation of the system's business logic. It represents the layer that can access the ledger for adding, validating, expiring and updating certificates. In case of updating a certificate a new transaction is made with the old transaction's key value. As in the following figure, the two marked keys in the blocks are the same value.

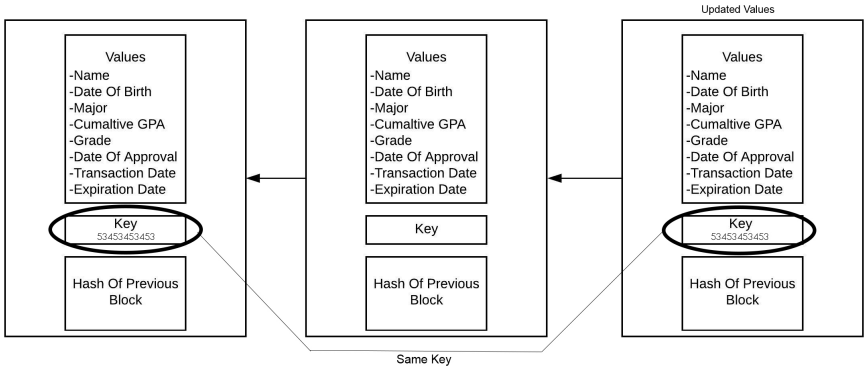

Figure 11: Application Architecture

5.2 Node.Js Server

This server interacts with the blockchain network, it uses the hyperledger fabric SDK that provides a powerful API to interact with the blockchain and enables the execution of the functions

- Creating Channels, Organizations.
- Invoking Chaincode.
- Querying Ledger using Chaincode.

Most importantly, this server exposes a number of endpoints in order for the client applications to be able to communicate with and to allow them to send requests that can be processed by the server.

5.2.1 Validation Key

The validation key is a unique number that is the key of any transaction on the ledger, through which the transaction is identified. It is generated in the process of issuance of any certificate. Its purpose is to validate a certificate, since it is considered a unique identifier for each certificate on the ledger. This validation key is generated by pairing the graduates national ID, which is unique across all graduates, and the timestamp in which the server invoked the chaincode's add certificate function. By pairing between both of those numbers a unique identifier across all certificates and all different versions of the same certificate is generated. The pairing process between those two number is made by using Cantor pairing function, which used to pair between any two natural number into a single natural number and from it the two natural numbers can be derived. Cantor pairing function is defined by:

$$z = \pi(x, y) = \frac{1}{2}(x + y)(x + y + 1) + y$$

Where x is the national ID and y is the timestamp. x and y can be calculated using z by doing this:

$$w = \left\lfloor \frac{\sqrt{8z+1}-1}{2} \right\rfloor$$

$$t = \frac{w^2 + w}{2}$$

$$y = z - t$$

$$x = w - y.$$

where $\lfloor\ \rfloor$ means floor function.

5.3 Client Application

The client application represents the presentation layer of our system through which the end user can interact with the system. it is implemented using Vue.js. it sends HTTP requests to the node.js server endpoints and receives responses and accordingly updates the user interface.

6 Human Interface Design

6.1 Screen Images

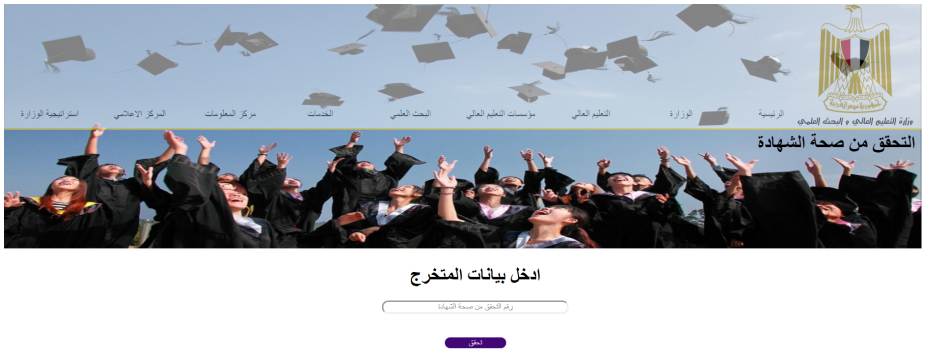

Figure 12: Certificate Verification Screen

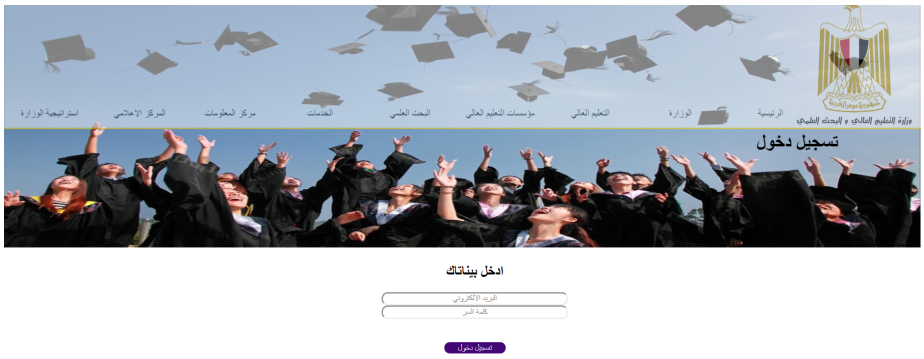

Figure 13: Login Screen

Figure 14: View Certificate Screen

Figure 15: Student Menu Screen

Figure 16: Request Certificate Screen

6.2 Screen Objects and Actions

6.2.1 Certificate Verification Screen

In this window, the company can write the job applicant's serial number to verify the certificate.

6.2.2 Login Screen

In this window, the user will write his credentials to login to the system.

6.2.3 Student Menu Screen

In this window contains the options the student has, either view his graduation certificate or request one.

6.2.4 View Certificate Screen

In this window, the student views his certificate.

6.2.5 Request Certificate Screen

In this window , the student can confirm his payment to request a graduation certificate.

7 Requirement Matrix

ID	Function	Status	Description
001	Certificate Exists	Completed	Checks the existence of the academic certificate.
002	Issue Certificate	Completed	Issues a certificate to a student.
003	Read Certificate	Completed	Views the data of a certificate from the world state.
004	Update Certificate	Completed	Updates the data of an existing certificate.
005	Delete Certificate	Completed	Deletes a certificate from the world state.
006	Enroll Admin	In Progress	Creates a new admin to the CA server.
007	Register User In CA	In Progress	Adds a new user to the CA server.
008	Validate Certificate	Completed	Checks the validity of the certificate (used by recruiting companies)
009	Request Certificate	In Progress	Requests issuance of a certificate.
010	Respond To Request	Completed	Respond to a request.
011	Generate QR Code	In Progress	Generates QR code.
012	Read QR Code	In Progress	Reads QR code.
014	Validate User	Completed	Uses CSR to validate the user.
015	Readx509Path	Completed	Use the member's path to read his x509 certificate.
016	Pay	In Progress	Confirms the payment to request a certificate.
017	Add Company	In Progress	Adds a company to the network.
018	Add University	In Progress	Adds a university to the network.
018	Invoke Chaincode	In Progress	Invokes chaincode functions.
019	Query Ledger	In Progress	Queries data from the ledger.
020	Create Wallet	In Progress	Creates a new file system based wallet for

Figure 17: Requirement Matrix

References

[1] Linux Foundation's Hyperledger Fabric (2020) https://www.hyperledger.org/about/charter

[2] Linux Foundation's Hyperledger Indy (2020) https://www.hyperledger.org/projects/hyperledger-indy

[3] Ethereum Blockchain (2020) https://ethereum.org/what-is-ethereum/