USMAN INSTITUTE OF TECHNOLOGY

Affiliated with NED University of Engineering & Technology, Karachi

Department of Computer Science

B.S. Computer Science / Software Engineering

FINAL YEAR PROJECT REPORT

Batch-2017

Blockchain Based E-Certification

By

Muhammad Rafay	17B-008-SE
Muhammad Aamir	17B-034-SE
Hassan Ahmed Siddiqui	17B-049-SE
Muhammad Sabih Mohsin	17B-058-SE
Muhammad Ismail	17B-062-SE

Supervised by

Dr. Muhammad Qasim Pasta

ST-13, Block 7, Gulshan-e-Iqbal, Abul Hasan Isphahani Road, Opposite Safari Park, P.O. Box 75300, Karachi, Pakistan. Phone: 34978274-5; 34994305; 34982476; http://www.uit.edu

Acknowledgments

There have been many individuals whom we are thankful for helping us to accomplish our final year project (FYP), and we would like to express our special thanks to all of them for their support and motivation.

First and foremost, during the project, we would like to thank the Almighty for his blessing. We would not be able to finish this mission without the grace of the Almighty.

Secondly, we would like to thank our director Dr. Zahir Ali Syed and our HoD Parkash Lohana for their efforts and support in overall curriculum.

We would now like to express our special thanks to our supervisor Dr. Muhammad Qasim Pasta who helped us and supported us in our work from the very first day, during this journey he has been an outstanding mentor and counselor because of his untiring efforts in giving directions that helped us to understand and tackle any kind of problem.

We would also like to thank our family who have been incredibly supportive during this project, in particular our parents for their non-stop prayers and true conviction that this could not have been achieved without their inspiration, and for supporting us financially as well as mentally.

We would also like to thank our university, all teachers, FYP team and friends who supported us somehow by offering vital knowledge and constructive criticism that helped us to properly finalize the project.

Abstract

The certification is one of the main sources of recognition of one's work but how do we maintain the authenticity of the certificates is the first thing that pops in mind. Is there any way we can verify the certificate do we always need to contact the organization to verify it what if something happened to the organizations or they decided to switch platforms then what? Our project aims to solve all these problems to maintain the authenticity of the certificate by using distributed architecture of blockchain technology so that no one will tamper the certificate your data is save with us. We provide hassle free verifications and recovery of certificates so even the organization gets shutdown or no matter where you are in world you can still verify and recover your certificates with ease. Some feature will be added in the future and these features include detail audit report, fully customizable certificate designer, etc. also currently we have services that are running on the single http server so we have decided to convert the whole architecture to micro service architecture to make the system more available to the user during the time of maintenance or any service failure.

Table of Contents

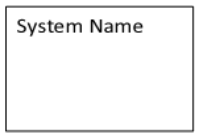
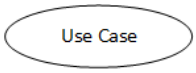
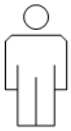
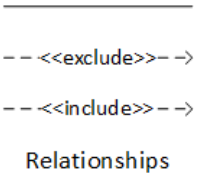
Contents

List of Tables

List of Figures

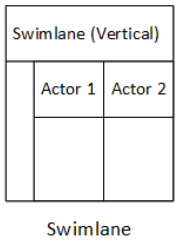
List of Symbols and Units

Use Case Diagram

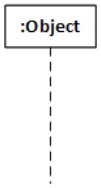
 System	Draw the boundaries of your system using a rectangle containing use cases. Place actors outside of the boundaries of the system.
 Use Case	Draw use cases using ovals. Mark the ovals with nouns and verbs which reflect the functions of the system.
 Actor	Actors are the users of a system.
 Relationships	Simple line represents relationships between an actor and a use case. For relationships among use cases, use arrows labeled either "include" or "exclude". A "include" relationship means that one use case is used by another in order to perform a task. In a particular use case, a "exclude" relationship implies additional choices.

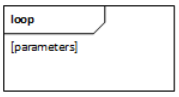
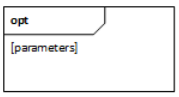
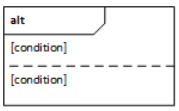
Activity Diagram

 Start Point	For every activity diagram, a small filled circle preceded by an arrow represents the start point. Make sure that the starting point is positioned in the top left corner of the first column for an activity diagram using swimlanes.
 Activity	An action state represents the objects' non-interruptible behavior.
Action Flow	The transformations from one action state to another are illustrated by action flows, often called edges and paths.
 Decision	A diamond reflects a choice of alternative routes. Attach a diamond between the two tasks where an action needs a decision before going on to the next activity. The outgoing alternates should be branded with the term of a state or guard. One of the routes can also be called "else."

 Swimlane	Swimlanes group related activities into one column.
 End Point	The end point depicts an arrow leading to a filled circle nested within another circle.

Sequence Diagram

 Class Role	The way an object behaves in context is defined by class role.
 Lifeline	The continuation of an object over time is indicated by lifelines, which are vertical dotted lines.
 Activation	The time it takes for an object to perform a task is expressed by activation boxes.
Synchronous Message	Before the connection can proceed, a synchronous message requires an response.
Asynchronous Message	For contact to proceed, asynchronous messages do not require a response.
 Self Message	A message received from an object to itself.
Return Message	A dotted line and an open arrowhead point back to the initial lifeline in a return letter.

 Loop	A rectangle represents a loop in a sequence diagram. Place the condition for leaving the loop in square brackets [ ] at the bottom left corner.
 Optional	Only if the supplied condition is correct does the fragment run. With just one trace, it's equivalent to an alt.
 Alternate	Only the condition that is valid will be executed.

1. Introduction

Certification has found its way into almost every industry for a reason: It helps advance the profession. Certification helps employers assess potential new hires, evaluate job performance, evaluate workers, pick contractors, market resources, and encourage employees to develop their skills and abilities. Certification provides recognition of skills, demonstrates devotion to the profession, and assists with career advancement. There has been an explosive growth in professional certification. Professional Certification helps one to: [1]

- Gain practical skills for the job, as it focuses on the knowledge and skills required to fulfill job responsibilities in the real world. [1]
- Realize the maximum benefit from a broad range of areas of expertise. [1]
- Learn all aspects of a particular type of career pursuit while providing these professionals with a standard of assured excellence for organizations. [1]
- Integrate these career pursuits effectively into one's specific work environment. [1]

1.1 Problems

The problem is that different certificates for events, courses, health, etc. are issued to the people from different organizations. But in case of any mishaps (natural disaster, misplace, etc.) person might lose his/her certificate, Sometime the organization shift the platform so it is very difficult process to get that certificate again due to lots of hurdles, verifying those certificates is really big issue, sometime if it is possible so it is very time consuming and costly process and sometime it is not possible due to lack of backup also preparing and distributing those certificates manually is so much time consuming.

1.2 Solution

We are providing a platform of blockchain based e-certification **CERTIFIS** which provides a way to verify the certificates not only by the certificate generating entity also by other users (either the part of any entity in the network or not) and in case any entity loss their data or change their platform then the certificates issued by them still be verifiable.

The Blockchain based e-certification project will meet the following objectives:

- Provide distributed platform for verifiability of certificate.
- No certificate or data loss due to any mishap.
- Tamper free securable environment.

1.2.1 Distributed Ledger

Distributed ledger technology, such as blockchain, are peer-to-peer networks that enable numerous users to manage a mutual ledger's own identical copy. DLTs allow their members to securely verify, execute, and record their own transactions without relying on a middleman, rather than having a central authority to update and communicate records to all participants. [2]

While there are a wide number of DLTs on the market, they all consist of the same building blocks: a public or private distributed ledger, a consensus algorithm (to ensure that all copies of the ledger are identical). [2]

1.2.2 Decentralization

As we know Hyperledger Fabric's design supports blockchain networks that are fully decentralized so we are providing a decentralized architecture with a consortium blockchain network that verifies such certificates by offering a means to validate certificates, not only by the certificate-generating organization, but by other organizations that are part of any entity in the network or not. In case of any server down or data loss due to any accident our data is still verifiable and secure due to decentralized architecture.

1.3 Blockchain and Smart Contract

Hyperledger Fabric smart contracts are written in chaincode and are invoked by an application external to the blockchain where the application wants to communicate with the ledger. In most instances, chaincode only communicates with the ledger's database part, the world state (for instance, querying it) and not the transaction log. [3]

1.3.1 Chaincode

Chaincode is software that identifies an asset or properties, and instructions for changing the asset(s) in the transaction. The chaincode enforces the rules for reading or changing key-value pairs or other information from the state database. Chaincode functions run against the existing state database of the ledger and are triggered by a proposal for a contract. The execution of Chaincode results in a series of key-value writes (write set) which can be sent to the network and added to all peers in the ledger. [4]

1.4 Background

There are approximately 10 companies that worked on digital verification of certificates with blockchain technology but no one is giving the solution that if they themselves dismissed than which entity will provide platform for verification of certificates. So, from here we enter the problem domain, we come with solution that there will be multiple organizations that will host the platform. Each organization will have the blockchain nodes and web servers for verification requests so that if any of the organization dismissed then other organizations hosting the network will provide the verifiability of certificates note that the user's verification request will be handle automatically from any of the available originations' servers.

1.5 System Diagram

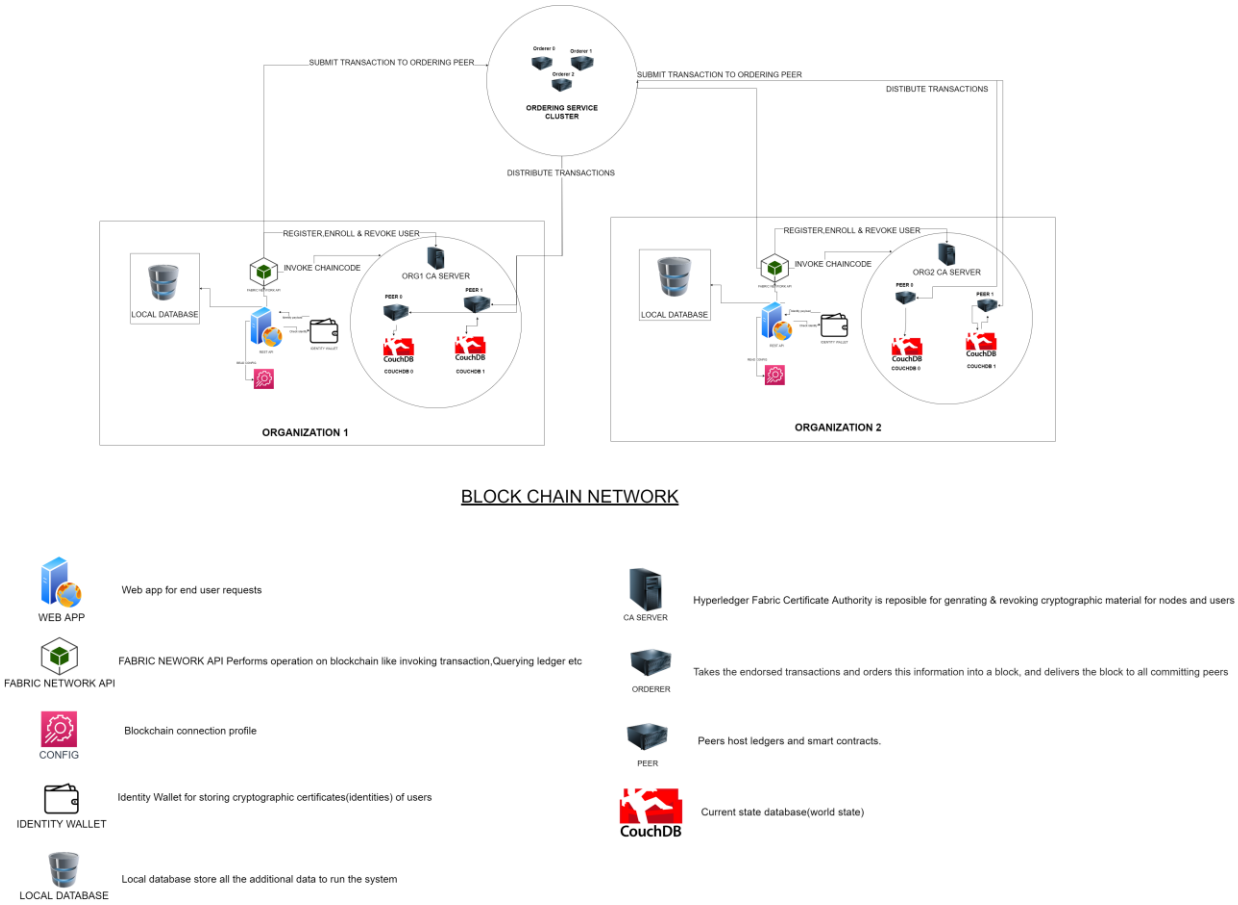


Figure 1.1 - System Diagram

The system includes the following:

- **Webserver**: Web app which handles the end user requests.
- **Local Database:** Store web generated data like user profiles, certificates data, configs for each tenant
- **Peers:** It hosts the ledger and hold the smart contracts.
- **Ordering Peers:** Take the endorsed transaction and orders this information into a block and deliver the block to all committing peers.
- **World State (CouchDB):** It stores current state of the ledger.
- **Fabric CA Server**: Fabric CA is responsible for generating and revoking cryptographic material for nodes and users.
- **Fabric SDK (Fabric Network API)**: A higher-level API for interacting with smart contracts and is the recommended API for client applications to interact with smart contracts deployed to a Hyperledger Fabric blockchain network.
- **Identity Wallet**: It stores the cryptographic identities of users.

1.5.1.1 System Workflow

1.5.1.1.1 Common Operations Flow

The flow is show in figure 1.1 describes when users send request to the webserver and the web server responds to the requests all the data of organization stored on the local DB may be single instance or clustered.

The common operation includes:

- Manage Organizations.
- Manage User.
- Manage Batches.
- Manage Publications.

1.5.1.1.2 User Creation Flow

The flow is show in figure 1.1 describes when users submit the registration request to the webserver now webserver store all the data in its local database and send the request to fabric CA server for cryptographic identity of the user so that the user perform operations on the blockchain.

1.5.1.1.3 Publishing to Blockchain Flow

The flow is show in figure 1.1 describes when the user send the certificate publishing request to the server, then for publishing the data on the blockchain web server check the identity wallet for the user's identity of blockchain to authenticate it and now using the fabric SDK it submits the request to the blockchain the fabric SDK read the connection profile of blockchain and run the discovery service to find the endorser after discovery it will send the endorsing proposal to the endorsing peer after successful response it submits that signed proposal to the ordering service cluster the ordering service orders the transaction and send to the peer organization where the transaction is validated and committed to the ledger.

1.6 Overview

In **chapter 1** we covered the brief introduction of our project. This chapter contains problem and how we are solving this problem. This chapter describe what the project is all about and why it was undertaken also we enlighten our project with some similar applications and discussed the system diagram of our project.

In **chapter 2** we covered background and literature review of our project. This chapter contains all the literature review that is covered for our project. In this chapter we briefly discussed the introduction of all the available blockchain technologies and in detail we discussed about the technology that we are using for our project also which development language and database platform is used, at last we discussed about all the similar software that are available till now.

In **chapter 3** we covered aim and statement of problem of our project. In this chapter we discussed about the strategies of how we are going to tackle the shortcomings of the similar software which are discussed in the previous chapter.

In **chapter 4** we covered software analysis and requirements of our project. This chapter contains the comparison of all the blockchain technologies, database platforms, web frameworks and the similar software also some of the important diagrams of our project and

the description of those diagrams, at last we discussed about the functional and non-functional requirements of our project.

In **chapter 5** we covered Software design and modeling of our project. This chapter contains the brief description of distributed architecture, its benefits and why we used it also this chapter contain all the interfaces of our web application.

In **chapter 6** we covered implementation of our project. This chapter contains all the implementation procedure require for our application along with how to implement it.

In **chapter 7** we covered testing of our project. In This chapter we have created blackbox and whitebox test case and performed testing against those test cases to insure proper working of our application.

2. Background and Literature Review

As we know that e-certificate are digitally verifiable and signed with the cryptographic signature of the issuing entity so the certificate legitimacy is trusted and anyone can verify who issued that certificate, so we are providing a blockchain based e-certification platform which is **CERTIFIS** and its aim is to provide a platform for organizations where users can generate certificates also make them able to verify those certificates. In case any organization removes its data or switches its platform, the certificates provided by them will still be verifiable due to decentralized architecture.

2.1 Blockchain

A special category of ledger is the blockchain. The way it manages information varies from a traditional database; blockchain stores data in blocks that are then chained together. When new data comes in a new block is joined. It is chained to the previous block until the block is loaded with data, which renders the details chained together in chronological order. It is possible to store various kinds of information on a blockchain, but the most common use to date has been as a transaction ledger. Decentralized blockchains are immutable, which means that they are irreversible in the knowledge reached. [5]

2.1.1 Different Type of Blockchains

Some of the different kind of available blockchain are mentioned below:

2.1.1.1 Ethereum

Ethereum is an open-source, decentralized blockchain with smart contract features. Ether is the platform's native crypto-currency. After Bitcoin, it is the second biggest crypto-currency by market capitalization. The most widely used blockchain is Ethereum. [6]

2.1.1.2 Hyperledger Fabric

Hyperledger is an open source blockchain and related tools umbrella project started by the Linux Foundation in December 2015 and has obtained contributions from IBM, Intel and SAP Ariba to promote the collaborative development of distributed blockchain-based ledgers. [7]

2.1.1.3 R3 Corda

Corda is a private blockchain approved network that allows corporations to use smart contracts to trade directly and in strict privacy with each other. [8]

The unique privacy paradigm of Corda Organization helps organizations to interact safely and seamlessly while reducing transaction and record-keeping costs and streamlining company processes in an environment of permission-less blockchain networks in which all data is exchanged with all stakeholders. [8]

2.1.1.4 Ripple

Based on a distributed open-source protocol, Ripple was launched in 2012 and supports tokens representing fiat money, cryptocurrencies, commodities, or other value units such as frequent flying miles or smartphone minutes. Ripple promises to make "secure, instantly and nearly free global financial transactions of any size with no chargebacks." The ledger uses the native cryptocurrency known as XRP. [9]

2.1.1.5 Quorum

Quorum is a blockchain network for companies. It is a fork of the 'geth' public ethereum client with many changes in protocol level to support business needs. The primary aim of the Quorum project is to build an ethereum client enterprise that empowers enterprises to accept blockchain technology and benefit from it. Since Quorum is an open-source initiative, the platform's code base is open to everyone to inspect, fostering trust in the platform. Open-sourcing further enhances acceptance and draws developers from numerous sectors to engage in this platform's growth. [10]

2.1.1.6 Hyperledger Sawtooth

Under the Hyperledger umbrella, Hyperledger Sawtooth is an open-source project that acts as an enterprise-level blockchain framework used for the development and management of distributed ledger applications and networks, primarily for enterprise use. [11]

2.1.1.7 EOS

EOSIO is the leading open-source blockchain platform that enables transparency in transactions at the speed and scale needed to solve real-world challenges. We believe transparent and decentralized systems will keep users and builders in control and help architect integrity into our world. [12]

2.1.1.8 Hyperledger Iroha

Hyperledger Iroha is planned to be quick and easy to integrate into initiatives involving distributed ledger technologies in infrastructure or IoT. A simple, scalable, domain-driven C++ design, emphasis on customer application development and a modern, crash fault tolerant consensus algorithm, called YAC, are features of Hyperledger Iroha. [13]

2.1.1.9 Stellar

Stellar is an open source, decentralized digital currency protocol that allows cross-border transactions between any pair of currencies for fiat money transfers. A 5013 nonprofit, the Stellar Development Foundation, is promoting the Stellar protocol. [14]

2.1.2 Hyperledger Fabric

In 2015, the Hyperledger project was created by the Linux Foundation to support cross-industry blockchain technology. Instead of declaring a common blockchain standard, it supports a collective approach through a group mechanism to the implementation of blockchain technology, including intellectual property protections that enable transparent development and the acceptance over time of key specifications. [3]

One of the blockchain ventures within Hyperledger is Hyperledger Fabric. It has a database, uses smart contracts, and is a mechanism in which users control their transactions, much as other blockchain technology. [3]

If Hyperledger Fabric splits from any other structures of the blockchain, it is private and approved. "The users of a Hyperledger Fabric network join through a trusted Membership Service Provider (MSP) instead of an open permission less scheme that requires anonymous identities to participate in the network (requiring protocols such as "proof of work" to verify transactions and protect the network). [3]

Several pluggable options are also offered by Hyperledger Fabric. In different formats, ledger data can be stored, compromise structures can be switched in and out and various MSPs are supported. [3]

The option to create channels is also provided by Hyperledger Fabric, enabling a group of members to create a different transaction ledger. For networks where some participants might be rivals and do not want any exchange they make, this is a particularly significant choice, a special price they give to some participants and not others, for example, available to every participant. If a channel is created by two participants, only those participants have copies of the ledger for that channel, and no others. [3]

We are using Hyperledger Fabric due to the below mentioned features:

- Permissioned membership.
- Performance, scalability, and levels of trust.
- Data on a need-to-know basis.
- Rich queries over an immutable distributed ledger.
- Modular architecture supporting plug-in components.
- Protection of digital keys and sensitive data.

2.1.2.1 Shared Ledger

Hyperledger Fabric has two elements of a ledger subsystem: the world state and the transaction log. Each member in each Hyperledger Fabric network to which they belong has a copy of the ledger. [3]

At a given point in time, the world state portion defines the status of the ledger. It's the Ledger Database. All transactions which have resulted in the current value of the world state are documented in the transaction log component; it is the world state update history. The ledger, then is a mixture of the record of the world condition and the history of the transaction log. [3]

The ledger has a replaceable data store for the world state. By default, this is a LevelDB key-value store database. The transaction log does not need to be pluggable. It simply records the before and after values of the ledger database being used by the blockchain network. [3]

2.1.2.2 Smart Contract

A smart contract is a computer program or a transaction protocol which is intended to automatically execute, control or document legally relevant events and actions according to the terms of a contract or an agreement. Hyperledger Fabric smart contracts are written in chaincode and are invoked by an application external to the blockchain. In most instances, chaincode only communicates with the ledger's database part, the world state (for instance, querying it and not the transaction log. [3]

In many programming languages, chaincode can be implemented. Go and Node are currently being supported. [3]

2.1.2.3 Privacy

Participants of a Business-to-Business (B2B) network can be highly sensitive about how much data they exchange, based on the needs of a network. Privacy is not going to be a top priority for most networks. [3]

Hyperledger Fabric embraces networks where both anonymity (using channels) and relatively transparent networks are a primary operating prerequisite. [3]

2.1.2.4 Consensus

Transactions must be written to the ledger in the order in which they occur, while within the network they may be between various sets of participants. In order for this to work, it is important to determine the order of transactions and to enforce a mechanism for refusing bad transactions that have been wrongly (or maliciously) entered into the ledger. [3]

This is a field of computer science that has been extensively studied, and there are several approaches to do it, each with multiple trade-offs. [3]

For e.g., a method for file replicas to interact with each other may be given by PBFT (Practical Byzantine Fault Tolerance) to keep each copy accurate, even in the event of corruption. Alternatively, ordering exists in Bitcoin via a mechanism called mining, where competing computers race to solve a cryptographic problem that determines the order that subsequently builds on all systems. [3]

Hyperledger Fabric has been developed to allow network starters to select a process of consensus that best reflects the interactions between participants. There is a variety of needs, as with anonymity, from networks that are heavily organized in their interactions to those that are more peer-to-peer. [3]

2.2 Development Language

- **Language:** Bash, js, html, css, yaml, json.
- **Runtime:** nodejs.
- **Back-End Frameworks:** Express Js.
- **Front-End-Framework:** vuejs.

2.3 Database Platform

We are using NoSQL (MongoDB) for storing the web generated data like user profiles, configs, tenant information etc.

The reason of choosing NoSQL (MongoDB) instead of RDBMS are mentioned below:

Certificate data models can be changed in future as different types of certificates have different requirements so by using MongoDB, we can easily create flexible database schema according to the certificate needs and the second reason to choose MongoDB is horizontal scaling as we are providing services to multiple organization so we can easily scale them according to the needs which means add more servers when load increases, in case of any mishap if server crashes, data loss due to attack or any other reason at any side then any of the replica's node will handle the requests so user will not face any downtimes due to any problem at particular side. Another reason for using NoSQL is performance. NoSQL provides faster read, write time as we are developing multi-tenant application so the performance really matters, also sharding and partitioning (separating very large databases into smaller, faster, more easily managed parts). To achieve this with SQL databases requires additional coding or more complex configuration while NoSQL databases include these features out-of-box. Both SQL and NoSQL offer high availability and replication, but SQL requires more complex configuration while many NoSQL databases automatically include these features.

2.4 Blockchain Advancement in Certification

2.4.1 KMI, OU - UK

2.4.1.1 Salient Features, Functionality

- Badges, certificates and web reputation in the blockchain.

2.4.1.2 Drawback

- Does not support employers as an entity.
- Data is stored on public blockchain.
- The certificate is vulnerable to manipulation.
- No clear method of authenticity of parties.

2.4.2 UNIC

2.4.2.1 Salient Features, Functionality

- Resolve fake certificates.
- Tools available for the authenticity of the certificate.
- Good in integrity, privacy and ownership.

2.4.2.2 Drawback

- Requirements for an employer to verify the certificate is inadequate.
- A student cannot authorize the prospective employer to verify the certificate.
- No clear method of authenticity of parties.

2.4.3 MIT Media Lab

2.4.3.1 Salient Features, Functionality

- Offers more control to students.
- Uses digital keys.

2.4.3.2 Drawback

- Level of trust is low.
- The certificate can be accessed by everyone.
- No clear method of authenticity of parties.

2.4.4 Blockcert

2.4.4.1 Salient Features, Functionality

- Open standard platform.

2.4.4.2 Drawback

- No separate verification services.
- Vulnerable to spoofing attacks.

2.4.5 Smartcert

2.4.5.1 Salient Features, Functionality

- Resolves the problem of fake certificates.
- Student shares the hash with the employer.

2.4.5.2 Drawback

- Vulnerable to attacks.
- Need for basic information security measures.
- No clear method of authenticity of parties.

2.4.6 RecordsKeeper

2.4.6.1 Salient Features, Functionality

- Proof of authenticity in the certificate.
- The entire verification process is based on ownership.

2.4.6.2 Drawback

- Certificate tampering vulnerability.
- Participants can verify after obtaining ownership.

2.4.7 Accredible

2.4.7.1 Salient Features, Functionality

- This platform lets you automatically create, manage, and distribute digital certificates, open badges and blockchain credentials for training, events, course completion, membership, etc.

2.4.7.2 Drawback

- Better filter options for the Credentials.
- Analytics could be better developed.

3. Aim and Statement of Problem

Since it helps advance the profession, certification has made its way into nearly every sector. Certificates are now relevant in today's world, but there are a lot of issues, on the other hand. This requires a solution like **CERTIFIS** to solve those problems. The goal of **CERTIFIS** is to address the challenges encountered by certificate issuing entities such as creation, validation and verification of certificates.

The concerns that **CERTIFIS** are trying to tackle are as follows:

3.1 Verification

Verification is often one of the key problems that occurs in certificate verification. Sometimes there is no proper channel and when there is a way to verify a certificate it is a very time consuming and tiring process and lots of time the verification of certificate is not trustworthy too but **CERTIFIS** solve all of these problems.

3.2 Tampering

As we know data tampering is the act of purposely altering data through unauthorized networks (destroying, controlling, or editing). There are two states where data exists: in transit or at rest. Data could be intercepted and exploited in both scenarios in case of certificate either it is in paper form or whether it is maintained on database the act of tampering can happen in both ways so **CERTIFIS** is going to be one of the platforms which provides tamper free environment.

3.3 Data loss due to Mismanagement of Document

Imagine that we lose any certificate or any valuable information connected to it if our record is not adequately handled or backed up by a certificate-generating body, then it places the protection of our organization at risk by accident if our records are not shielded from the wrong hands, then it breaks regulatory laws and could place us and our company in legal trouble. While weak record protection is an evergreen problem, technological advancements and new management techniques will help us prevent crucial failures and **CERTIFIS** is one of the best options for this task.

3.4 Cyber Attack

It is one of the risky stuffs that many certificate-generating companies are unable to manage and struggle in this area, since we realize that cyber-attacks can cause infrastructure loss and security secret breaches. Valuable, confidential details such as personal details, credential ID, code, applicant addresses, phone number, etc. can be compromised. Cyber-attacks interrupt computer networks or paralyze processes due to security vulnerabilities, rendering data inaccessibly, but **CERTIFIS** would be the first certificate agency to aim as best as possible to fix this problem.

3.5 Server down

As we know, several certificate producing organizations are running on centralized server in case of server down as they are highly reliant on network communication, system will fail if the nodes loss connectivity as there is only one central node there is no graceful deterioration of system abrupt failure no one can allow transaction, data backup is also one of the major

problems if the system abrupt failure occur you lose the data straight away but as we know **CERTIFIS** run on decentralized architecture so incase if any organization's server down or data lost due to any mishap so the verifiability of certificate is still possible due to distributed and decentralized architecture. So, in this matter **CERTIFIS** is still on top.

3.6 Security Loopholes

Security is one of the challenges in which many certificate-generating organizations still suffer because virtually every entity has security loopholes, but we know how critical security is, that is why **CERTIFIS** seek to have as much as possible a safe atmosphere due to decentralized architecture.

3.7 Data Redundancy

Data redundancy is one of the problems that arise on a regular basis when many organizations are unable to accommodate vast volumes of data, so they produce data redundancy due to their mismanagement and there is no check and balance and no proper way to plan and handle this whole method but **CERTIFIS** have upper hand their too because as defines earlier we are using NoSQL database so the main thing in NoSQL is that we de-normalize data to get faster read write times in simple words renormalization is increasing the data redundancy by storing related data in a single document. We have clear fully analyze the dataset and tried to denormalized only the data which is required frequently by the client and normalized the data into multiple collections which was not needed frequently by the client.

3.8 Data Backup

Data backup is one of the main issues that is faced by most of the organizations who are using central servers for storing their data. The organizations that are using central servers for storing the information of their certificate then in case any attacker attacks their server and they don't have proper backup of their data then they might lose their data also if the organizations have proper backup but the attacker also harm their backups so in this case organizations can lose their data completely. To resolve this issue **CERTIFIS** is using blockchain for storing the certificate data of organizations. By using blockchain **CERTIFIS** can solve backup problem because in case attacker attacks on any of the node on blockchain then data of the organization will be saved on many other nodes of the blockchain and this will make sure the data availability.

4. Hardware, Software Analysis and Requirements

There is no particular need of hardware, although there exist some similar software and the comparison of those are mentioned below:

4.1 Comparison of Different Blockchain Technologies

Table 4.1 – Blockchain Technologies' Comparison

	Ethereum	**Hyperledger Fabric**	**R3 Corda**	**Ripple**	**Quorum**	**Hyperledger Sawtooth**	**EOS**	**Hyperledger Iroha**	**Stellar**
Industry focus	Cross-Industry	Cross-Industry	Financial Services	Financial Services	Cross-Industry	Cross-Industry	Cross-Industry	Cross-Industry	Financial Services
Ledger Type	Permission less	Permissioned	Permissioned	Permissioned	Permissioned	Permissioned	Permissioned	Permissioned	Both Public & Private
Consensus Algorithm	Proof of Work	Pluggable Framework	Pluggable Framework	Probabilistic Voting	Majority Voting	Pluggable Framework	Delegated Proof-of-Stake	Chain-based Byzantine Fault Tolerant	Stellar Consensus Protocol
Smart Contract	Yes	Yes	Yes	No	No	Yes	Yes	Yes	Yes
Governance	Ethereum Developers	Linux Foundation	R3 Consortium	Ripple Labs	Ethereum Developers and JP Morgan Chase	Linux Foundation	EOSIO Core Arbitration Forum (ECAF)	Linux Foundation	Stellar Development Foundation
Cryptocurrency	Ether	No	No	Ripple (XRP)	Ether	-	-	-	-
Language for Smart Contract	Solidity	JavaScript, Java, Go	Kotlin, Java	C++	Solidity	Go, Rust, Python, JavaScript	C++	No Smart Contracts	JavaScript, Java, Go, Ruby, Python, C#

4.2 Comparison of Different NoSQL Databases

Table 4.2 – NoSQL Databases' Comparison

	MongoDB	**Cassandra**	**CouchDB**
Type	Document based	Flexible wide columns	Document
Query Language	JavaScript	CQL	Http API
Multi-Record Acid Transactions	Yes	No	No
Replication Modes	Master/ slave replication	Master/ slave replication	Multi master replication
Partitioning Methods	Sharding	Sharding	Sharding
Server-Side Scripts	JavaScript	No	View functions in JavaScript
Operation	On-premises or in the cloud	On-premises or in the cloud	On-premises or in the cloud
Map Reduce	Yes	Yes	Yes
Protocol	TCP/IP	Thrift	Http/Rest

4.3 Comparison of Different Web Frameworks

Table 4.3 - Web Frameworks' Comparison

	Express.js	**ASP.NET**	**Django**	**Laravel**
Category	Web application framework	Web application framework	Web application framework	Web application framework
Language	JS	C#	Python	PHP
Platform	Cross Platform	Only ASP.NET core is cross platform	Cross Platform	Cross Platform
Processing Model	Asynchronous, Single-thread non-blocking, I/O model	Synchronous, multi-threaded, blocking I/O	Synchronous, multi-threaded, blocking I/O	Synchronous, multi-threaded, blocking I/O
Leading Firms Using these Languages in their Projects	NASA, Walmart, eBay, Uber	AccuWeather, Dell, Stack Overflow	Instagram, Pinterest, Coursera	9GAG, BBC, Tour Radar
Templating Language	EJS, jade, Pug, etc.	Razor	Jinja	Twig, Smarty, Mustache, etc.
Package Manager	Npm	Nuget	PyPI	Packagist
Primarily Classified	Microframeworks (Backend)	Frameworks (Full Stack)	Frameworks (Full Stack)	Frameworks (Full Stack)

4.4 Comparison of Available Software in terms of Salient Features, Functionality and Shortcomings

Table 4.4 - Existing Solutions and their Shortcomings

Institution/ Solution	Salient Features, Functionality	Shortcomings in Feature/ Functionality
KMI, OU - UK	• Badges, certificates and web reputation in the block chain.	• Does not support employers as an entity. • Data is stored on public blockchain. • The certificate is vulnerable to manipulation. • No clear method of authenticity of parties.
UNIC	• Resolve fake certificates. • Tools available for the authenticity of the certificate. • Good in integrity, privacy and ownership.	• Requirements for an employer to verify the certificate is inadequate. • A student cannot authorize the prospective employer to verify the certificate. • No clear method of authenticity of parties.
MIT Media Lab	• Offers more control to students. • Uses digital keys.	• Level of trust is low. • The certificate can be accessed by everyone. • No clear method of authenticity of parties.
Blockcert	• Open standard platform.	• No separate verification services. • Vulnerable to spoofing attacks.
Smartcert	• Resolves the problem of fake certificates. • Student shares the hash with the employer.	• Vulnerable to attacks. • Need for basic information security measures. • No clear method of authenticity of parties.
RecordsKeeper	• Proof of authenticity in the certificate. • The entire verification process is based on ownership.	• Certificate tampering vulnerability. • Participants can verify after obtaining ownership.
Accredible	• This platform lets you automatically create, manage, and distribute digital certificates, open badges and blockchain credentials for training, events, course completion, membership, etc.	• Better filter options for the Credentials. • Analytics could be better developed.

4.5 Comparison of Available Software in terms of their Security Themes

Table 4.5 - Certification Solutions Maps with Requirements Theme

Institution/ Solution	Salient Features, Functionality	Essential Security Themes to Fulfill Certificate Verification in Blockchains				
		Authentication	*Authorization*	*Confidentiality*	*Ownership*	*Privacy*
KMI, OU - UK	• Badges, certificates and web reputation in the block chain.	No	No	No	No	Yes
UNIC	• Resolve fake certificates. • Tools available for the authenticity of the certificate. • Good in integrity, privacy and ownership.	No	No	No	Yes	Yes
MIT Media Lab	• Offers more control to students. • Uses digital keys.	No	Yes	No	Yes	Yes
Blockcert	• Open standard platform.	No	No	No	No	No
Smartcert	• Resolves the problem of fake certificates. • Student shares the hash with the employer.	No	Yes	No	Shared	No
RecordsKeeper	• Proof of authenticity in the certificate. • The entire verification process is based on ownership.	Yes	No	No	Shared	No
Accredible	• This platform lets you automatically create, manage, and distribute digital certificates, open badges and blockchain credentials for training, events, course completion, membership, etc.	Yes	Yes	Yes	Shared	No

4.5.1 Certifis

4.5.1.1 Salient Features, Functionality

The Blockchain based e-certification project will contain the following features:

- Creation of tamper free & verifiable certificates using blockchain technology.
- Any organization can join a blockchain network by deploying nodes.
- Organizations hosting the blockchain network can provide services to others.
- No certificate loss or any data loss due to any mishap.

4.6 Diagrams

4.6.1 System Diagram

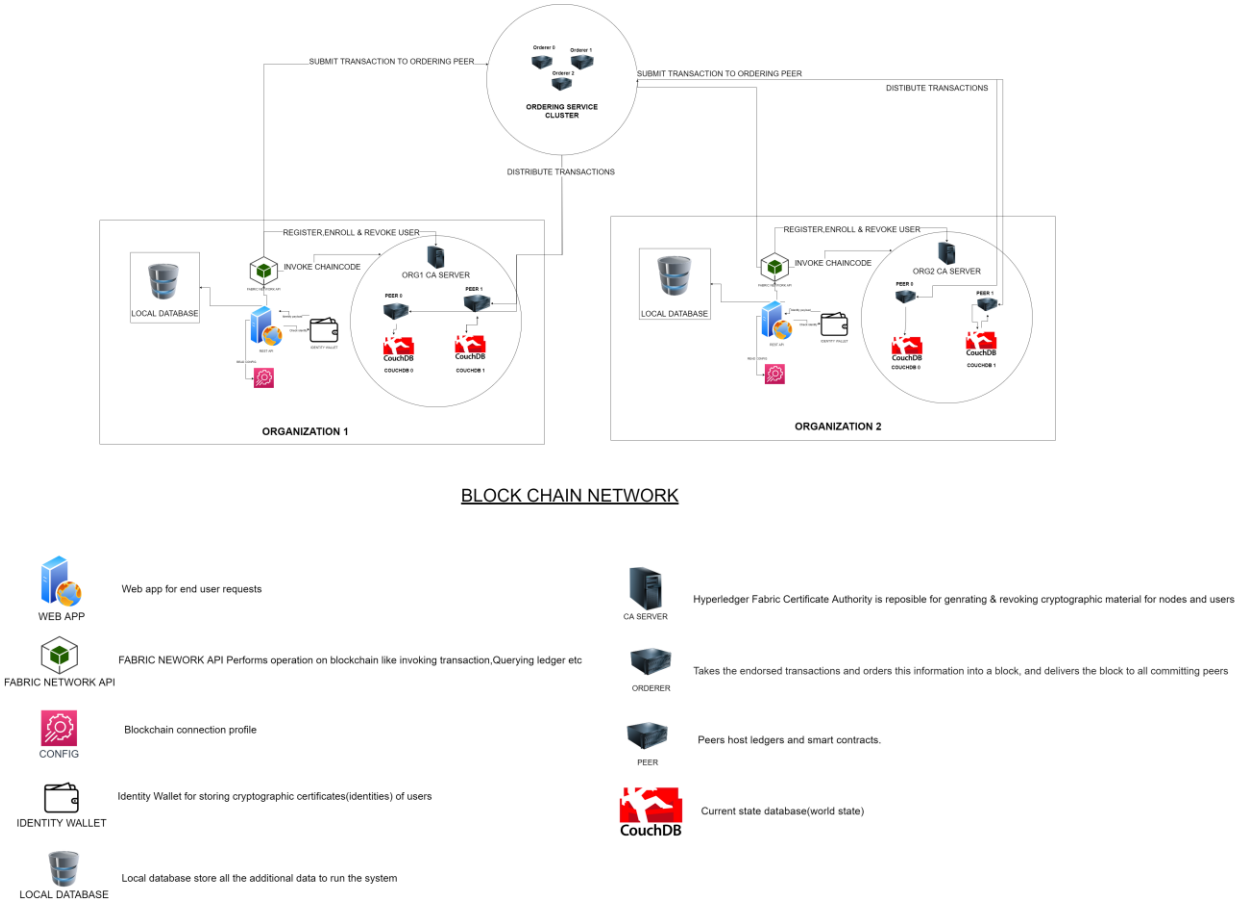


Figure 4.1 - System Diagram

The system includes the following:

- **Webserver**: Web app which handles the end user requests.
- **Local Database:** Store web generated data like user profiles, certificates data, configs for each tenant
- **Peers:** It hosts the ledger and hold the smart contracts.
- **Ordering Peers:** Take the endorsed transaction and orders this information into a block and deliver the block to all committing peers.
- **World State (CouchDB):** It stores current state of the ledger.
- **Fabric CA Server**: Fabric CA is responsible for generating and revoking cryptographic material for nodes and users.

- **Fabric SDK (Fabric Network API)**: A higher-level API for interacting with smart contracts and is the recommended API for client applications to interact with smart contracts deployed to a Hyperledger Fabric blockchain network.
- **Identity Wallet**: It stores the cryptographic identities of users.

4.6.1.1 System Workflow

4.6.1.1.1 Common Operations Flow

The flow is show in figure 4.1 describes when users send request to the webserver and the web server responds to the requests all the data of organization stored on the local DB may be single instance or clustered.

The common operation includes:

- Manage Organizations.
- Manage User.
- Manage Batches.
- Manage Publications.

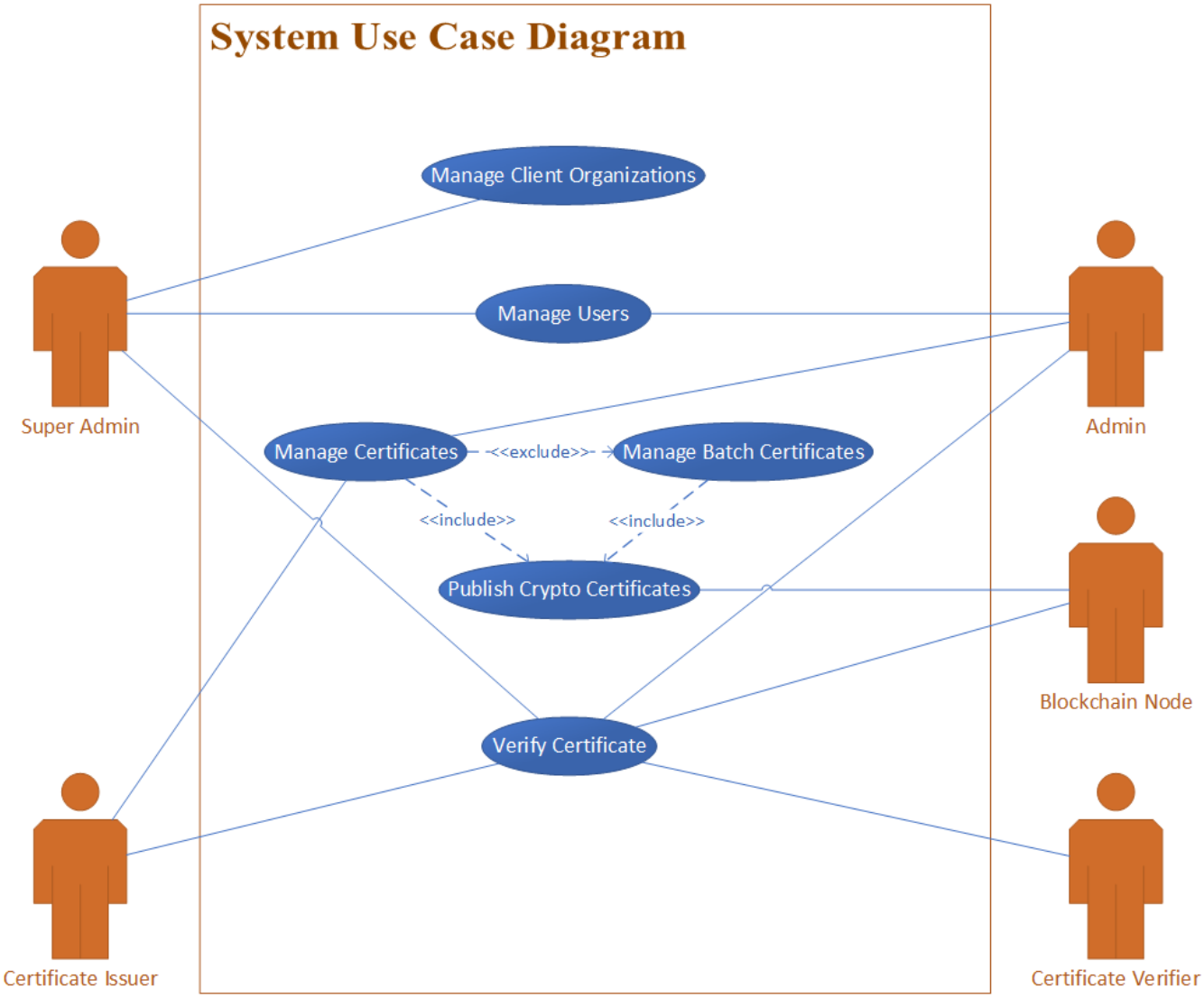
4.6.1.1.2 User Creation Flow

The flow is show in figure 4.1 describes when users submit the registration request to the webserver now webserver store all the data in its local database and send the request to fabric CA server for cryptographic identity of the user so that the user perform operations on the blockchain.

4.6.1.1.3 Publishing to Blockchain Flow

The flow is show in figure 4.1 describes when the user send the certificate publishing request to the server, then for publishing the data on the blockchain web server check the identity wallet for the user's identity of blockchain to authenticate it and now using the fabric SDK it submits the request to the blockchain the fabric SDK read the connection profile of blockchain and run the discovery service to find the endorser after discovery it will send the endorsing proposal to the endorsing peer after successful response it submits that signed proposal to the ordering service cluster the ordering service orders the transaction and send to the peer organization where the transaction is validated and committed to the ledger.

4.6.2 Actor Use Case Diagram



Use Case Details

Manage Client Organization: Register Organization, Enable/Disable Client Organization, Update User Limit, Update Organization's Certificate Limit, View Count History, View Single Certificate, View Batches, View Batch Certificates.
Manage User: Register Users, Enable/Disable Users, Reset Password.
Manage Certificates: Create Certificates, Edit Certificates, Delete Certificates, View Certificates, Publish Certificates.
Manage Batch Certificates: Create Batches, Edit Batches, Delete Batches, View Batches, Publish Batches.
Publish Crypto Certificates: Publish Batches, Publish Single Certificates.
Verify Certificates: Verify Certificates, Download Certificates, Recover Certificates.

Figure 4.2 - Actor Use Case Diagram

The figure 4.2 defines all the action that are performed by every actor. There are five actors:

- Super Admin.
- Admin.
- Certificate Issuer.
- Certificate Verifier.
- Blockchain Node.

4.6.2.1 Super Admin

The super admin will be able to perform the following task:

Add Organization, Enable/Disable Client Organization, Assign User Limit, Assign Organization's Certificate Limit, View Count History, View Certificate Balance, View Client Organization's Single Certificate, View Client Organization's Batches, View Client Organization's Batch Certificates, Register Users, Enable/Disable Users, Reset Password, Verify Certificates, Download Certificates, Recover Certificates.

4.6.2.2 Admin

The admin will be able to perform the following task:

Register Users, Enable/Disable Users, Reset Password, Create Certificates, Edit Certificates, Delete Certificates, View Certificate, Create Batches, Edit Batches, Delete Batches, View Batches, Publish Batches, Publish Single Certificates, Verify Certificates, Download Certificates, Recover Certificates.

4.6.2.3 Certificate Issuer

The certificate issuer will be able to perform the following task:

Create Certificates, Edit Certificates, Delete Certificates, View Certificate, Create Batches, Edit Batches, Delete Batches, View Batches, Publish Batches, Publish Single Certificates, Verify Certificates, Download Certificates, Recover Certificates.

4.6.2.4 Certificate Verifier

The certificate verifier will be able to perform the following task:

Verify Certificates, Download Certificates, Recover Certificates.

4.6.2.5 Blockchain Node

The blockchain node is the system actor and this will perform the following task:

Publish Batches, Publish Single Certificates, Verify Certificates, Download Certificates, Recover Certificates.

4.6.3 Activity Diagrams

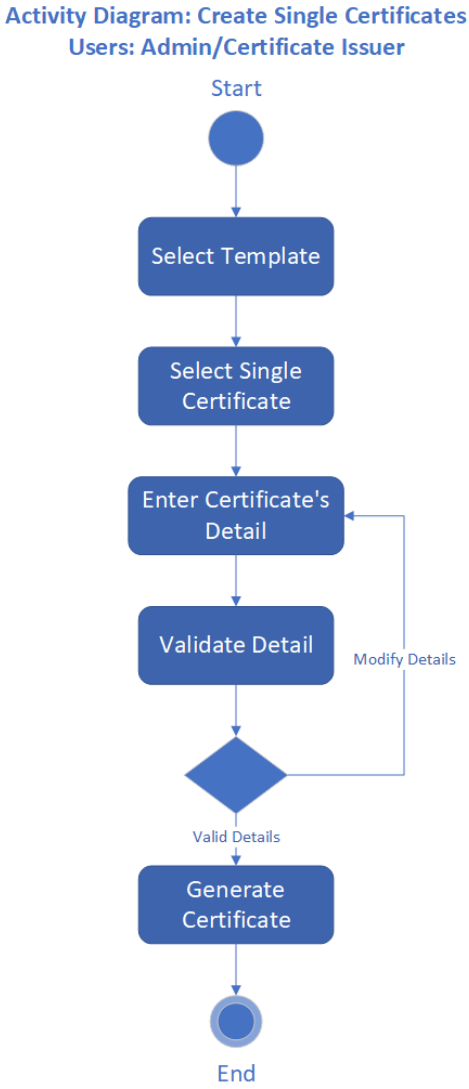


Figure 4.3 - Create Single Certificate Activity Diagram

If the user wants to create a single certificate this will provide him/her the feature of creating single certificates. By choosing a certificate template he/she will be able to create single certificates. The figure 4.3 shows the process of creating single certificates, first user need to go to the create option which is available on the navbar the system will render all the available certificate templates, user will select any of the template then he/she will be able to view or fill as per choice in case if user fills the selected template then system will ask user either they want to create single certificate or they want to creates batch for creating single certificate user needs to selects single certificate then he/she will be allowed to add all the required details of the certificate and when the user clicks on create button then system will validate the formats of all the inputs given by user if the format is correct then the system will generate the certificate else the system will prompt error for correcting the format of the inputted fields.

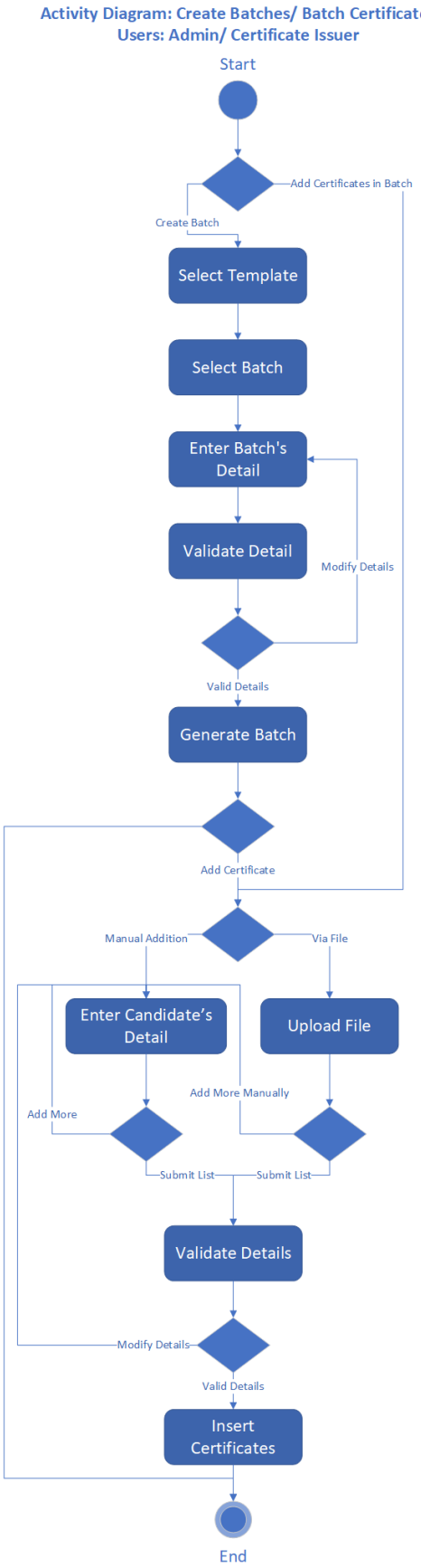


Figure 4.4 - Create Batches/ Batch Certificates Activity Diagram

If the user wants to create batches or wants to add certificates in the batches this will provide him/her the feature of creating batches and add certificates in the batches. In case user wants to create batch so by choosing a certificate template he/she will be able to create batches and if the user wants to add certificates in them then user can also add certificates in them. The figure 4.4 shows the process of creating batches and adding certificates in the batches, first user need to go to the create option which is available on the navbar the system will render all the available certificate templates, user will select any of the template then he/she will view or fill as per choice in case if user fill the selected template then system will ask user either they want to create single certificate or they want to creates batch for creating batch user needs to selects batch then he/she will be allowed to add all the required details of the batch and when the user clicks on create button then system will validate the formats of all the inputs given by user if the format is correct then the system will generate the batch else the system will prompt error for correcting the format of the inputted fields. Once the batch has been created the system will redirect user to add certificates page if the user wants to add certificates in the batch then user will be able to add certificates in it if user don't want to add certificates then this will end the process. If the user doesn't want to create batch and just want to add certificates in the batch this will provide him/her the feature of adding certificates. By choosing the batch he/she will be able to add certificates in that batch. The figure 4.4 also shows the process of adding certificates in the batch, first user need to go to the certificate option which is available on the navbar then system will display two options single certificates and batches for adding certificates in the batches user needs to select batches option so system will display all the unpublished batches to the user then user will select any of the batch and will go to batch detail and when the user clicks on add button then the system will render the page to add certificates, now user will be having multiple options of adding certificate one is manual addition and the second is add certificates by file if the user wants to add manually then he/she can add certificates manually and if user wants to add certificates using file then he/she can upload the excel file then the system will retrieve certificates data from the file also if the user wants to add more certificates after uploading the file then he/she can add them manually once the user clicks on submit button then the system will validate the detail of all the certificate if all certificates' detail is correct then the system will add those certificates to the batch if any of those certificates' detail is invalid so system will prompt an error to the user for correcting the detail.

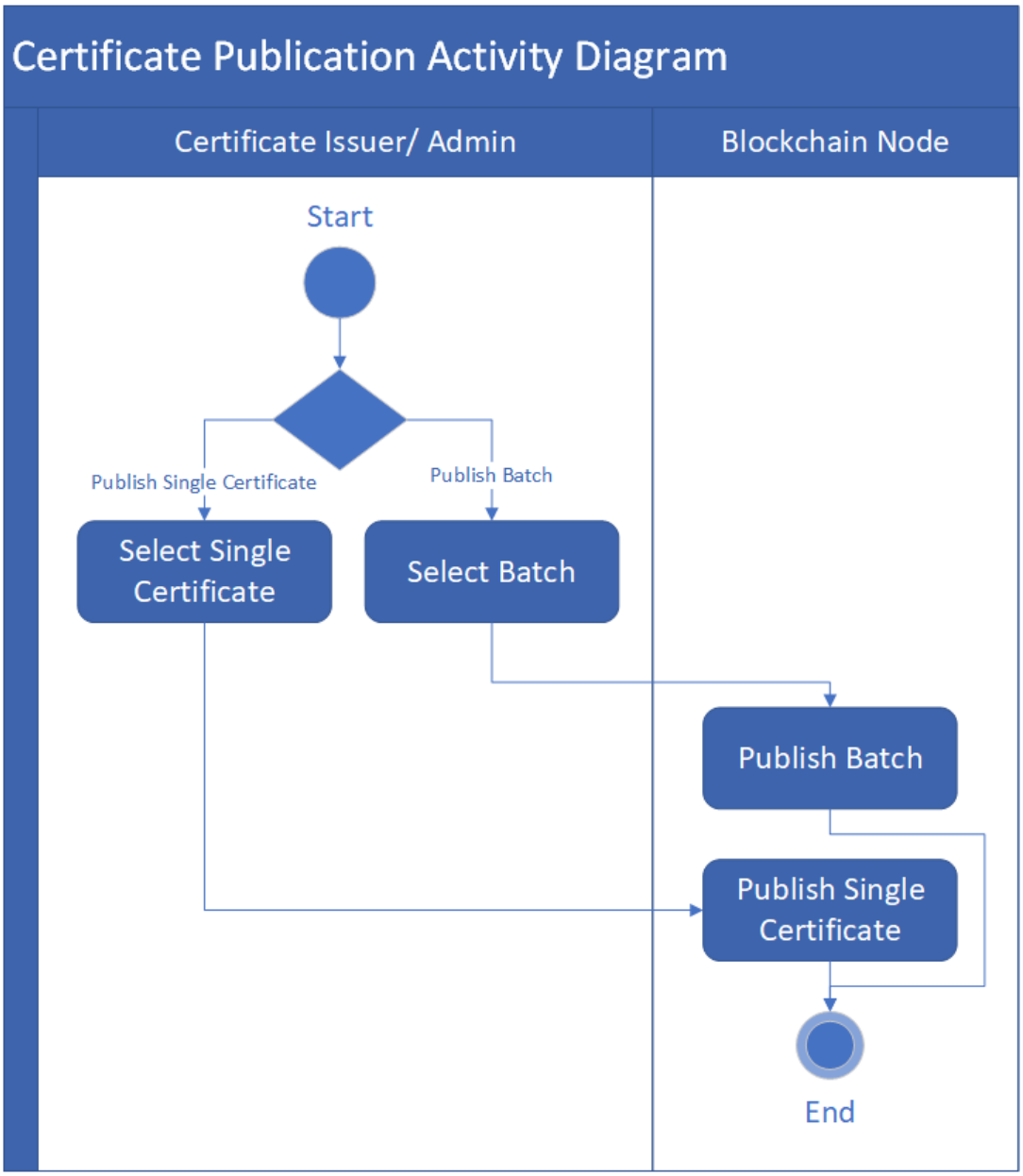


Figure 4.5 - Certificate Publication Activity Diagram

If the user wants to publish single certificates or batches to the blockchain the this will allow users to publish single certificates or batches to the blockchain after publishing, all the certificates will be verifiable. The figure 4.5 shows the process of publishing single certificates or batches to the blockchain, first user need to go to the certificate option which is available on the navbar then system will display two options single certificates and batches, incase if user selects single certificates option so system will display all the unpublished single certificates to the user then user will select any of the single certificate and click on publish button to publish that single certificate to the blockchain and if user selects batches option so system will display all the unpublished batches to the user then user will select any of the batch and click on publish button to publish that batch to the blockchain.

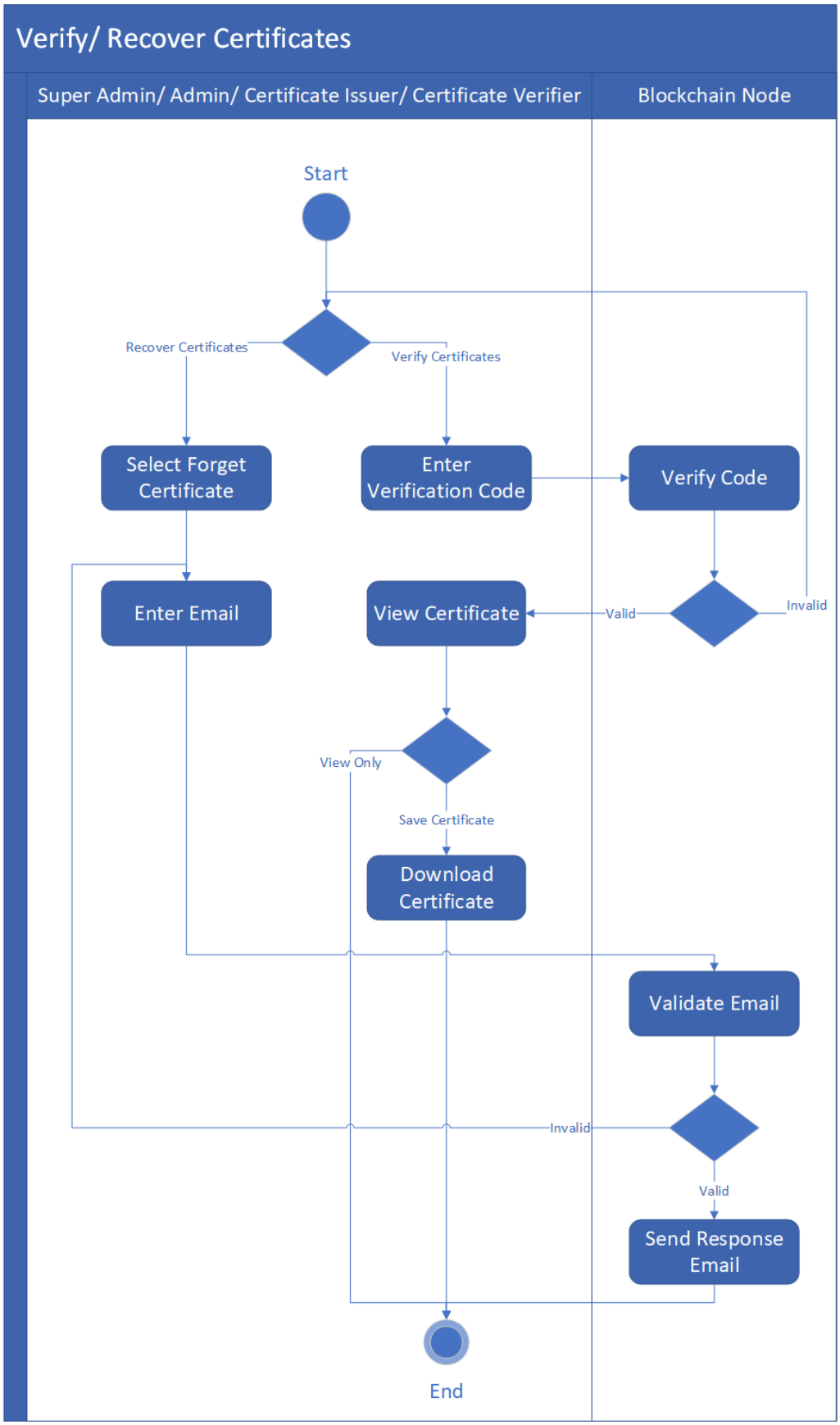


Figure 4.6 - Verify/ Recover Certificates Activity Diagram

If the user wants to verify certificates this will allow users to verify the certificates that have been published to blockchain. The figure 4.6 shows the process of verifying certificates either in batch or single certificates, first user needs to go to the verify option which is available on the navbar then system will render verification page if the user wants to verify the certificate he/she needs to enter the verification code if the code is valid then certificates will be displayed to the user and if user wants to save the certificates then the user can download that certificate and in case if the code is invalid then the system will prompt error of invalid code to the user also if user forgets the certificate code then user can recover those certificates by clicking on forget certificate which will render the forget certificate page to recover certificates now user needs to enter his/her email address, once user submit the request, the system will validate the email if the email is valid then all the certificates against that email will be sent to that email address.

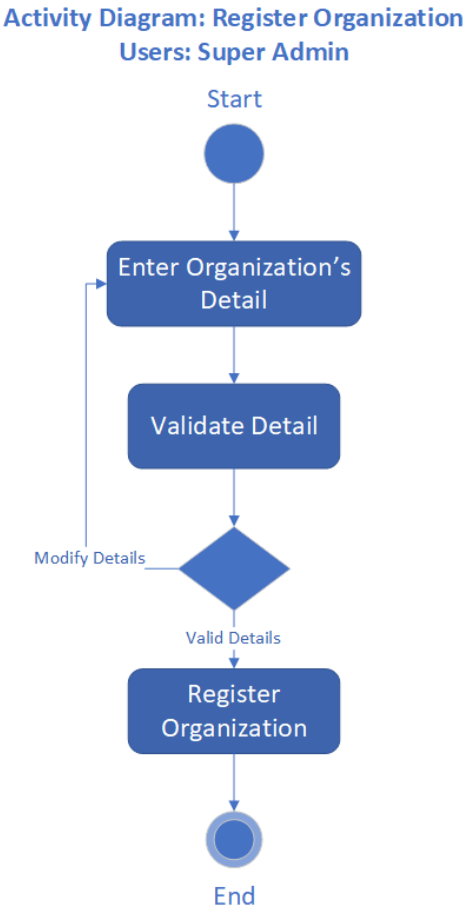


Figure 4.7 - Register Organization Activity Diagram

If the user wants to register new organization. So, figure 4.7 shows the process of registering organization, first user needs to go to the organization option which is available on the navbar then system will render organization page if the user wants to register organization user need to enter the detail of organization when the user submits the detail of the organization the system will validate the detail of the organization if the detail is valid then the system will register the organization if the detail is invalid then the system will prompt user to correct the detail.

4.7 Requirements

The main goal requirements of problem were given by our supervisor and some requirements related to user experience are self-defined also we collaboratively worked with our supervisor to mine many of the core and supportive requirements by prototyping, also held multiple meeting in which we brainstorm many scenarios and analyze the mockups to make the requirements more crystal clear.

4.7.1 Functional Requirements

- As a Super Admin, I want to create batch certificates so that I can issue the same certificates to as many candidates as required.
- As a Super Admin, I want to create single certificates so that I can issue certificates to single candidates.
- As a Super Admin, I want add user functionality so that I can add users in my organization and in client organizations.
- As a Super Admin, I want to enable/disable client organization so that I can disable any client organization due to any reason like policy violation, service unsubscription, etc.
- As a Super Admin, I want to enable/disable users so that I can disable any users of my organization and any users and admin of client Organizations due to any reason like policy violation, left organization, etc.
- As a Super Admin, I want to publish certificates so that I can publish single and batch certificates to the block chain.
- As a Super Admin, I want to register client organization so that our organization provides services to other organizations.
- As a Super Admin, I want certificate count assign feature so that I can set the certificate issuing limit of our client organization.
- As a Super Admin, I want forget password link generation feature so that I can generate the link by which users of my organization and users and admin of client organizations can reset their password.
- As a Super Admin, I want to verify certificates so that I can verify any certificates.
- As a Super Admin, I want forget certificates functionality so that I can recover all the certificates against a particular email.
- As a Super Admin, I want to view single certificates so that I can check all the single certificates issued by my organization
- As a Super Admin, I want to view batch certificates so that I can check all the batches created by my organization and certificates in those batches.
- As a Super Admin, I want to view client organizations' single certificates so that I can check all the single certificates issued by client organizations.
- As a Super Admin, I want to view client organizations' batch certificates so that I can check all the batches created by client organizations and certificates in those batches.
- As a Super Admin, I want to service utility report so that I can check usage of my organization as well as client organization.
- As an Admin, I want to create single certificates so that I can issue certificates to single candidates.
- As an Admin, I want to create batch certificates so that I can issue the same certificates to as many candidates as required.
- As an Admin, I want add user functionality so that I can add users in my organization.
- As an Admin, I want to enable/disable users so that I can disable any users of my organization due to any reason like policy violation, left organization, etc.

- As an Admin, I want to publish certificates so that I can publish single and batch certificates to the block chain.
- As an Admin, I want forget password link generation feature so that I can generate the link by which users of my organization can reset their password.
- As an Admin, I want to verify certificates so that I can verify any certificates.
- As an Admin, I want forget certificates feature so that I can recover all the certificates against a particular email.
- As an Admin, I want to view single certificates so that I can check all the single certificates issued by my organization.
- As an Admin, I want to view batch certificates so that I can check all the batches created by my organization and certificates in those batches.
- As an Admin, I want to service utility report so that I can check usage of my organization.
- As an Issuer, I want to create single certificates so that I can issue certificates to single candidates.
- As an Issuer, I want to create batch certificates so that I can issue the same certificates to as many candidates as required.
- As an Issuer, I want to publish certificates so that I can publish single and batch certificates to the block chain.
- As an Issuer, I want to verify certificates so that I can verify any certificates.
- As an Issuer, I want forget certificates feature so that I can recover all the certificates against a particular email.
- As an Issuer, I want to view single certificates so that I can check all the single certificates issued by organization.
- As an Issuer, I want to view batch certificates so that I can check all the batches created by organization and certificates in those batches.
- As a Verifier, I want to verify certificates so that I can verify any certificates.
- As a Verifier, I want forget certificates feature so that I can recover all the certificates against a particular email.
- As a Verifier, I want to download certificate so that I can use this certificate.

4.7.2 Non-Functional Requirement

4.7.2.1 Reliability

The **overall system** should handle any interruption due to internet failure or any other mishaps.

4.7.2.2 Authorization

The authorization should be imposed on **overall system** operations except the **verification process** which is open to the public.

4.7.2.3 File Integrity

All static content uploaded by user while **creating certificates and batches** should be stored in a manner so that when the user requests his file the system should return the correct files.

4.7.2.4 Audit Trail

Audit trail should be maintained for check and balance for **certificate creation, organization registration, certificate's count assign, publish single certificates and publish batches** so that we can track when and who perform these operations.

4.7.2.5 Continuity of Processing

The log should be maintained while **publishing single certificates and publishing batches** for making the system able to continue publishing where it stops in case of any interruption.

4.7.2.6 Service Level

The response time of **publishing single certificates and publishing batches** should be as minimum as possible and the process which takes long time should be pushed to the background processes and should not make user to wait for the response ones the process finished it should notify the user when he/she is available.

4.7.2.7 Access Control

The level of access should be clearly defined for **overall system**; the user can access only that resources for which they are eligible.

4.7.2.8 Methodology

The **overall system's** requirement, design, construction, documentation, testing, integration and maintenance should be according to the development, design, programming, test, installation and maintenance methodology.

4.7.2.9 Correctness

The features such as **create certificates, create batches, add batch certificates, register organization and register users'** requests should be validated for correctness before proceeding to the further steps.

4.7.2.10 Ease of Use

The features such as **create certificates, create batches, add batch certificates, publish single certificates and publish batches** should be simple and easy to use.

4.7.2.11 Maintainable

The **overall system** should be highly decoupled so that the maintenance will not cause the whole system to go down. The only component under maintenance could be unavailable instead of the whole system.

4.7.2.12 Portable

The **overall system** should not face compatibility issues when deployed to different environments; also, the system component should follow the standard protocol so it can easily integrate or communicate with third party components or services.

4.7.2.13 Coupling

The **verification services** should be highly decoupled so it can easily be integrated in third party applications.

4.7.2.14 Performance

The features such as **publish single certificates and publish batches** should be able to accept faster requests and provide faster responses and capable of providing faster services under heavy load.

4.7.2.15 Ease of Operation

The features such as **create certificates, create batches, add batch certificates, publish single certificates and publish batches** should be developed according to the HCI standards so different type of users can use it easily.

5. Software design and modeling

We are using three types of architecture in our application which includes service, decentralized and distributed architectures. Our backend is based on service-based architecture which means that it is highly decoupled from the front-end application. Our client application interacts with the backend through APIs for retrieving and sending data. The purpose of using services-based architecture is that we will not only be restricted with web applications. So that in future we will be able to extend our application to multiple platforms like iOS, Android, Desktop, etc. also if we want to sell our services to the third party then we will be able to sell those services. Now if we talk about complete system then the overall architecture of our system is distributed because the frontend, backend, database, message broker & blockchain (decentralized distributed system) are deployed on different server even the consumer and producer of message broker can run on different servers. This shows that our application is totally distributed. Now let's discuss some failure cases, in case our frontend application crashed and stopped working due to any reason then our backend will still remain unaffected and still able to serve the clients access the services from different channels, in case any of the backend service fails than other service will not get affected and will continue to run normally, in case of complete backend failure our front-end application still be able to communicate with user about the issues.

5.1 What is Distributed Architecture?

Components are presented on various platforms in a distributed architecture, and multiple components may collaborate over a communication network to accomplish a certain purpose or goal. [15]

- Information processing is spread through many different machines in this architecture, rather than being limited to a single processor. [15]
- The client-server architecture, which is the basis for multi-tier architectures, may be used to illustrate a distributed system; other solutions include broker architectures such as CORBA and Service-Oriented Architecture (SOA). [15]
- .NET, J2EE, CORBA, .NET Web services, AXIS Java Web services, and Globus Grid services are some of the application platforms that support distributed architectures. [15]

5.2 Benefits of Distributed Architecture

- **Resource sharing** − Hardware and software services are shared. [15]
- **Openness** − The ability to use hardware and applications from a variety of vendors. [15]
- **Concurrency** − To improve efficiency, use concurrent processing. [15]
- **Scalability** − By adding additional resources, we were able to increase throughput. [15]
- **Fault tolerance** − Since an error has arisen, the right to continue operating. [15]

5.3 Why are we Using Distributed Architecture?

The aim of using Distributed Architecture is to make our structure as self-contained as possible. Since we use a distributed architecture, our frontend, backend, and database and message broker all function separately. If our frontend application goes down for whatever reason, our backend will be unaffected; but, if one of the backend servers goes down, our frontend application will be unaffected due to other backend server(s).

5.4 Web Interfaces

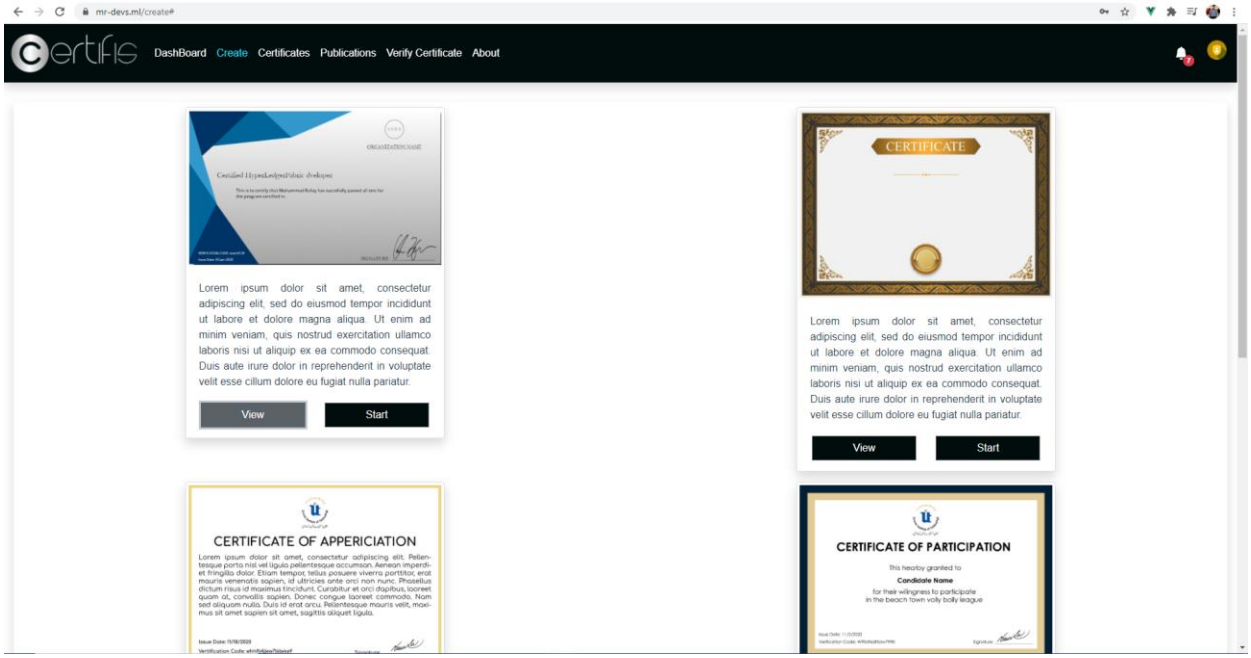


Figure 5.1 - Template Selection Web Interface

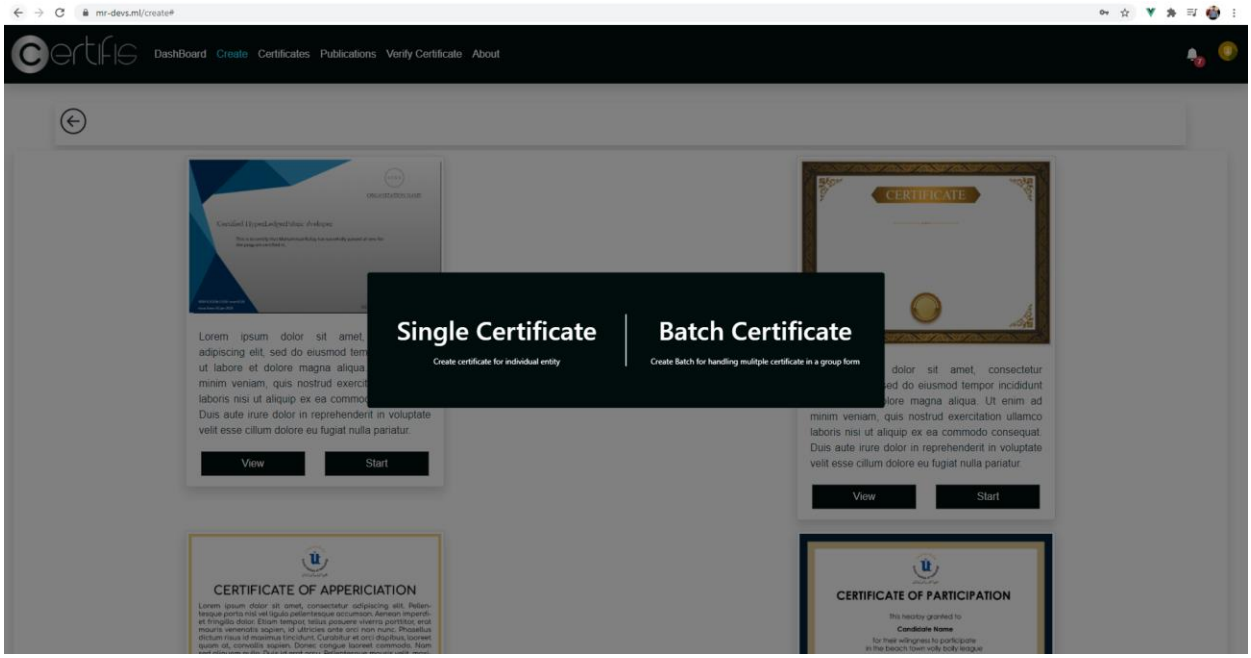


Figure 5.2 - Single or Batch Selection Web Interface

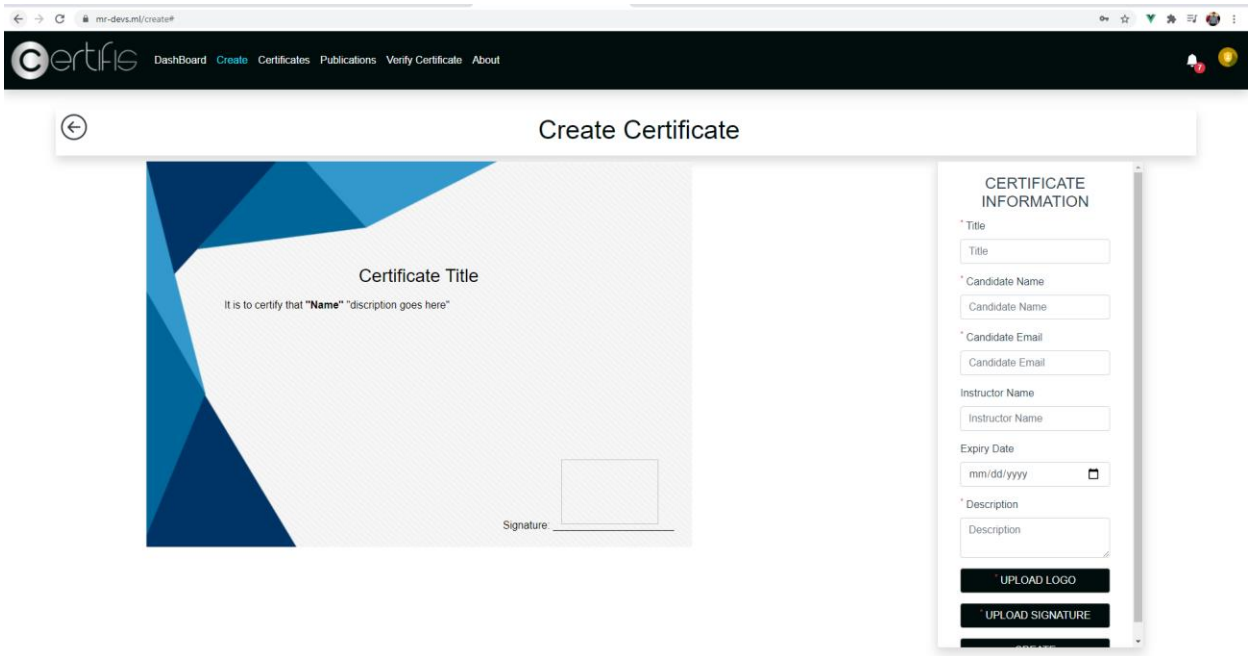


Figure 5.3 - Create Single Certificate Web Interface

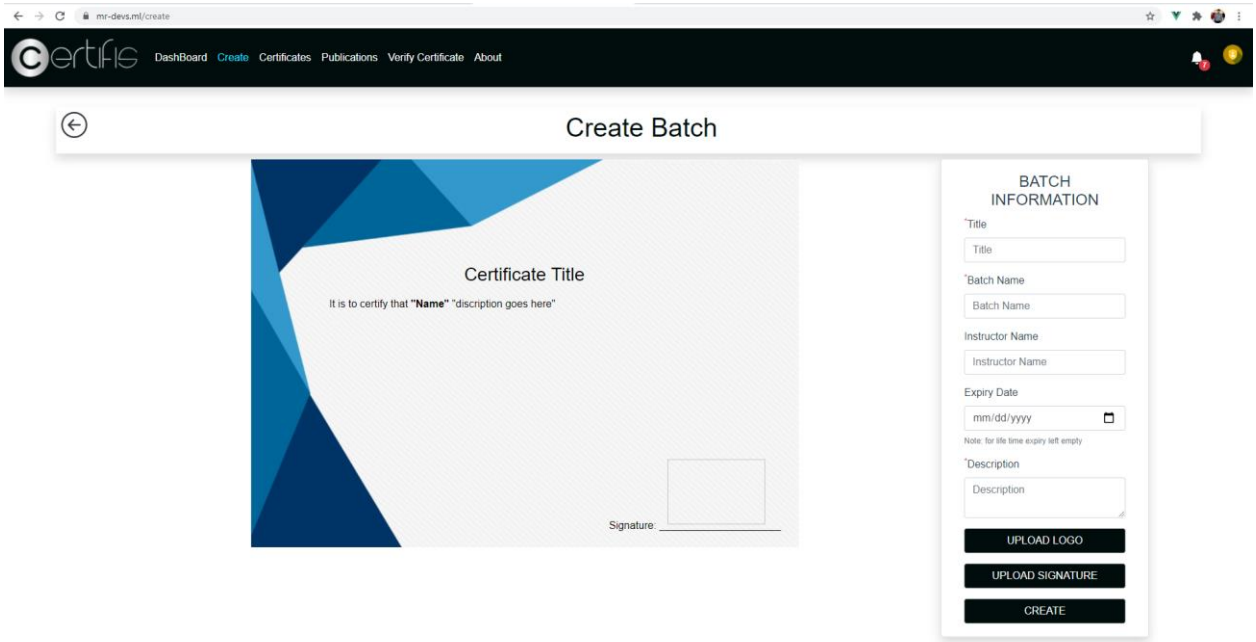


Figure 5.4 - Create Batch Web Interface

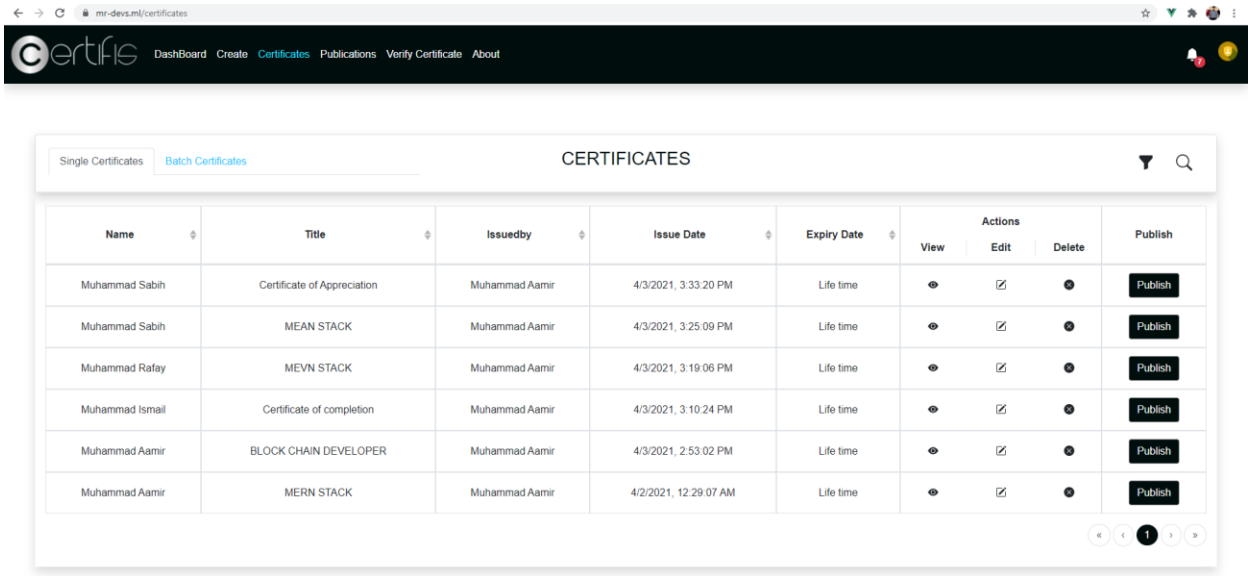


Figure 5.5 - Single Certificate List Web Interface

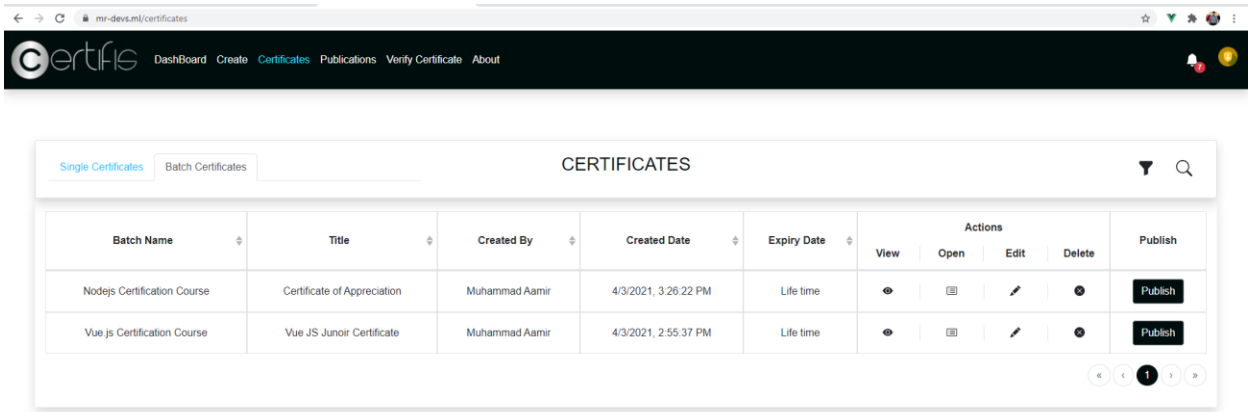


Figure 5.6 - Batch List Web Interface

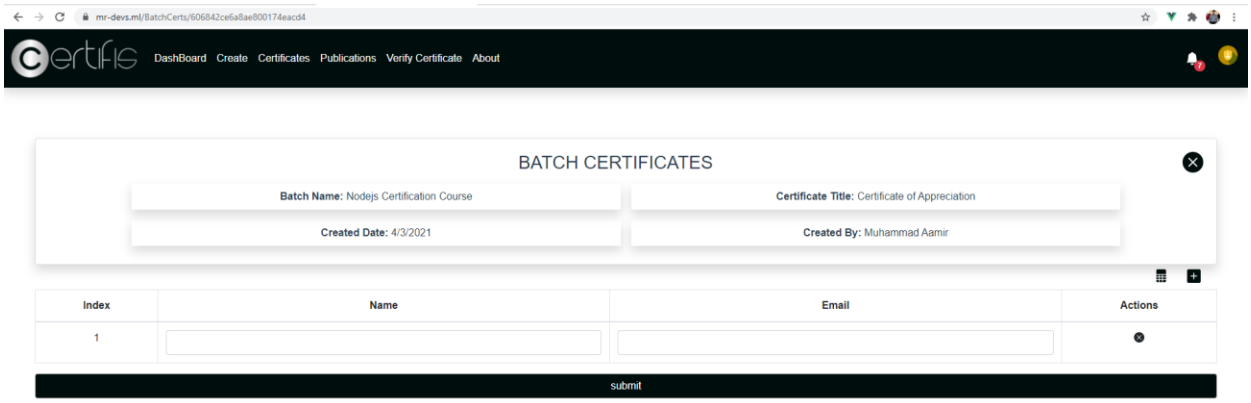


Figure 5.7 - Create Batch Certificates Web Interface

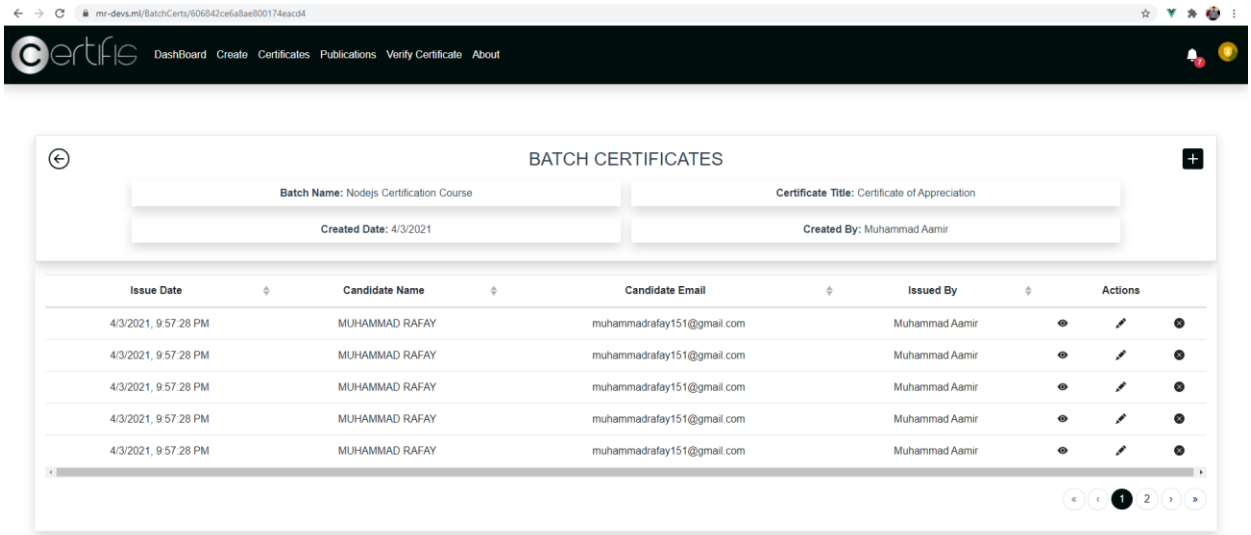


Figure 5.8 - Batch Certificates List Web Interface

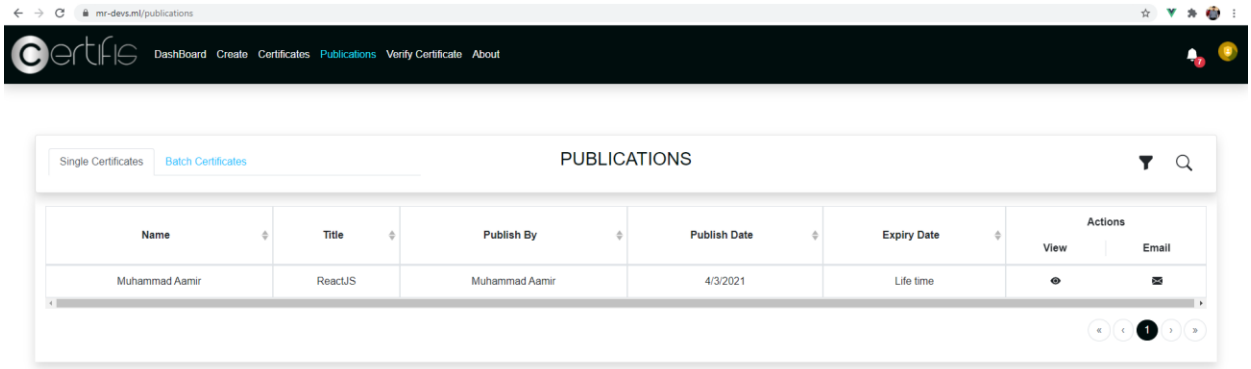


Figure 5.9 - Published Single Certificate Web Interface

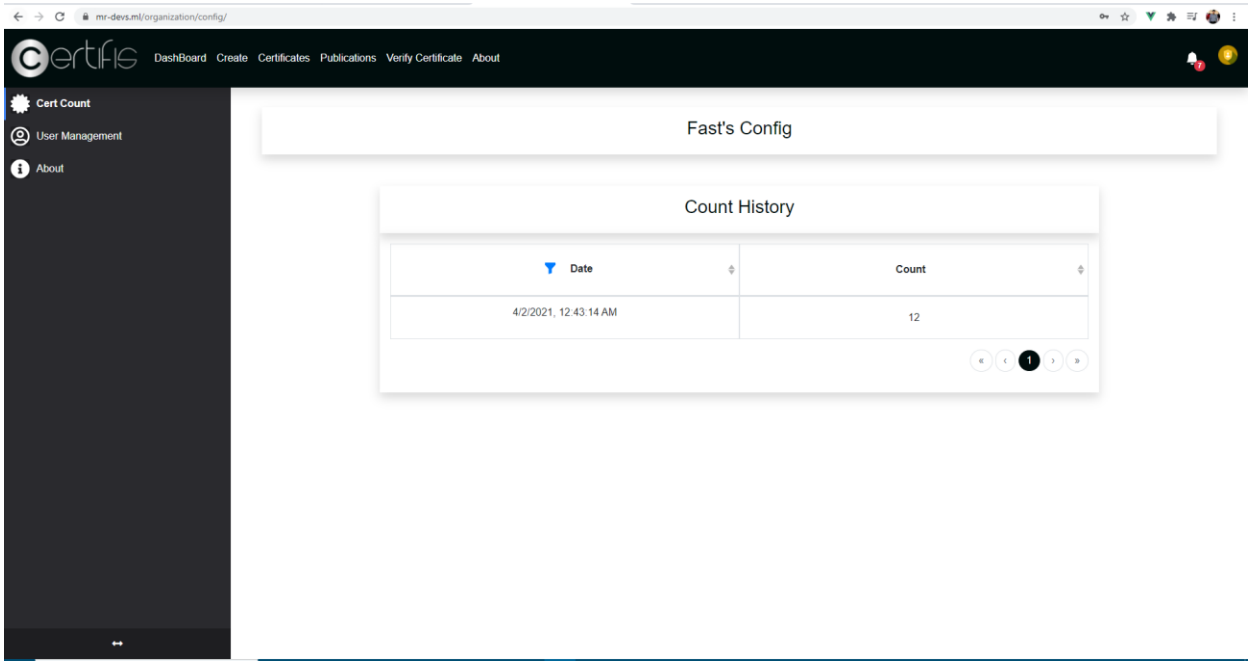


Figure 5.10 - Count History Web Interface

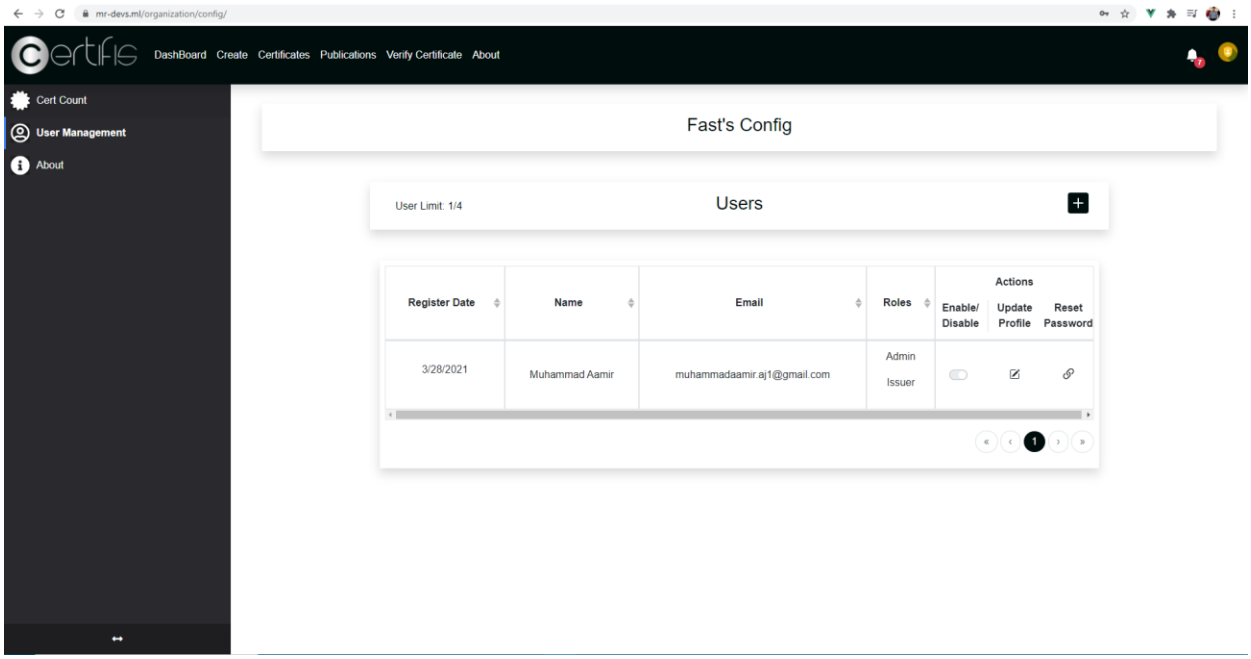


Figure 5.11 - User Management Web Interface

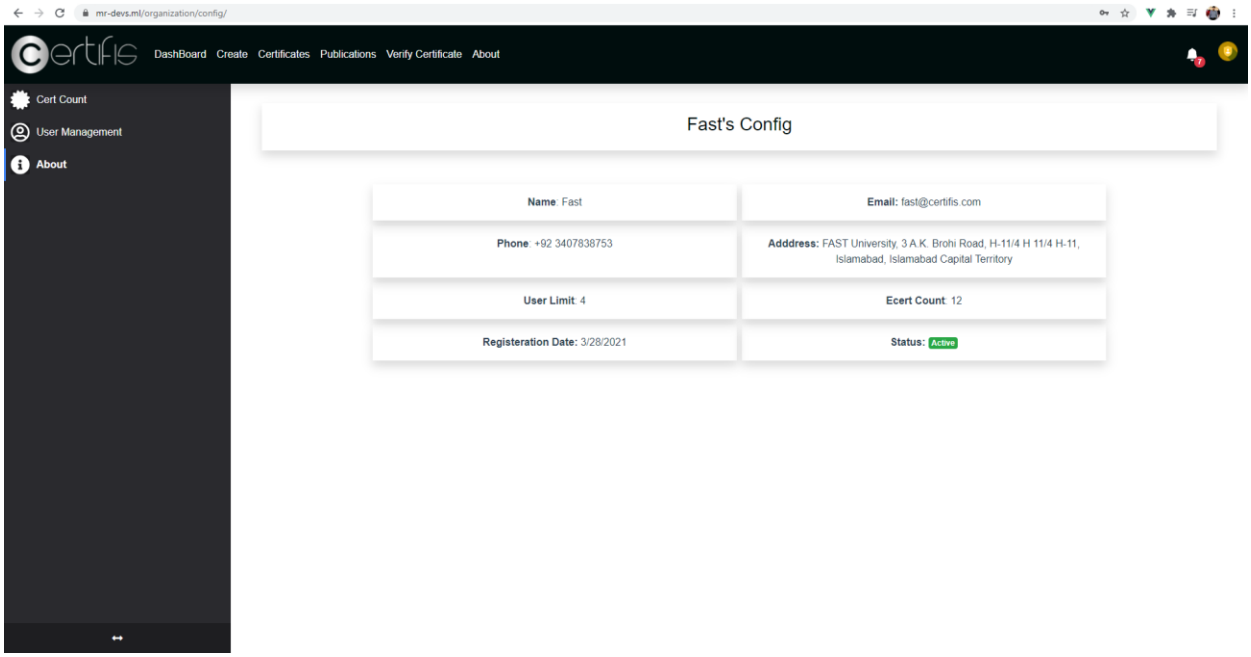


Figure 5.12 - About Web Interface

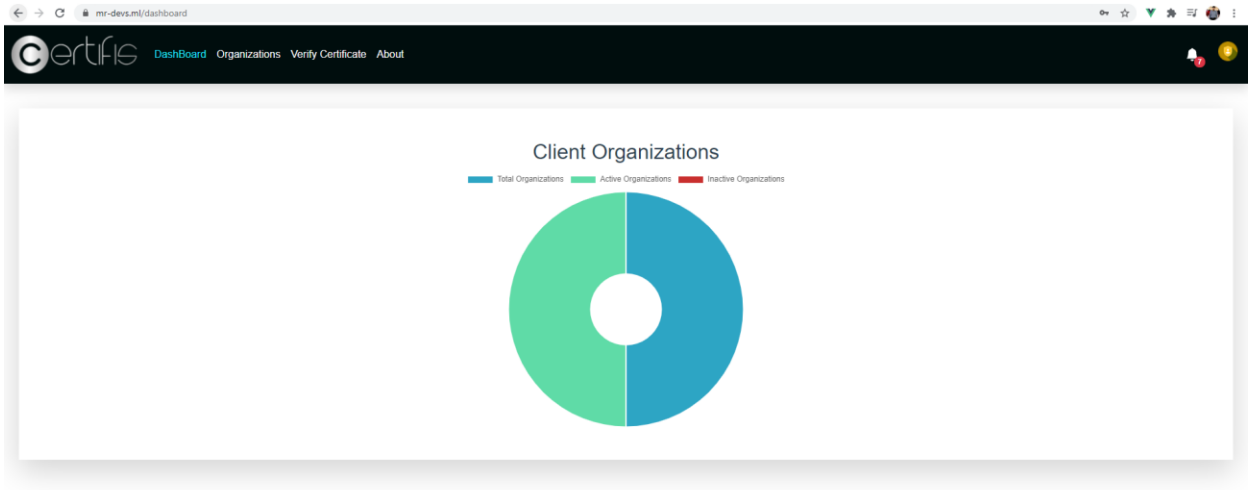


Figure 5.13 - Client Organization Graphical View Web Interface

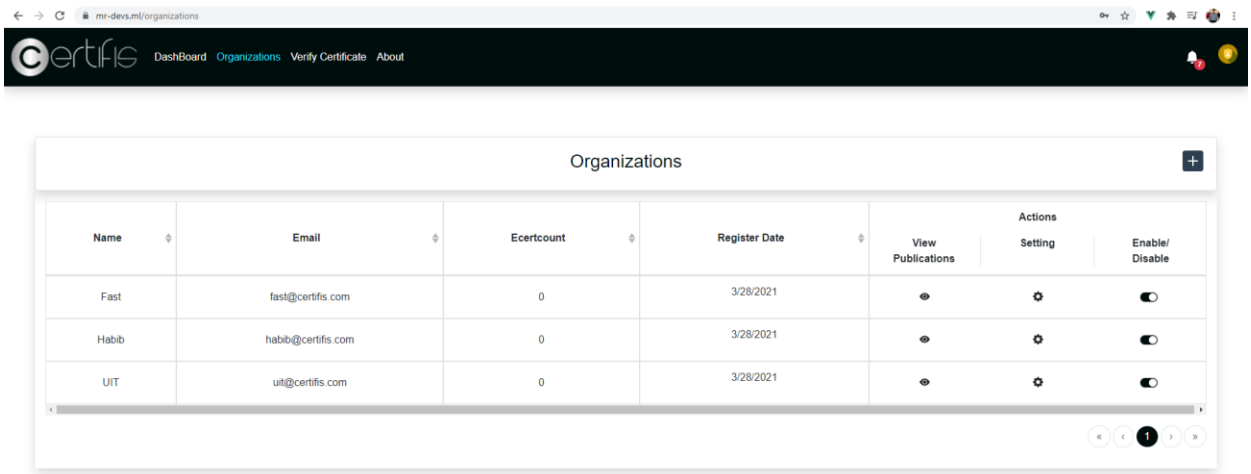


Figure 5.14 - Client Organization List Web Interface

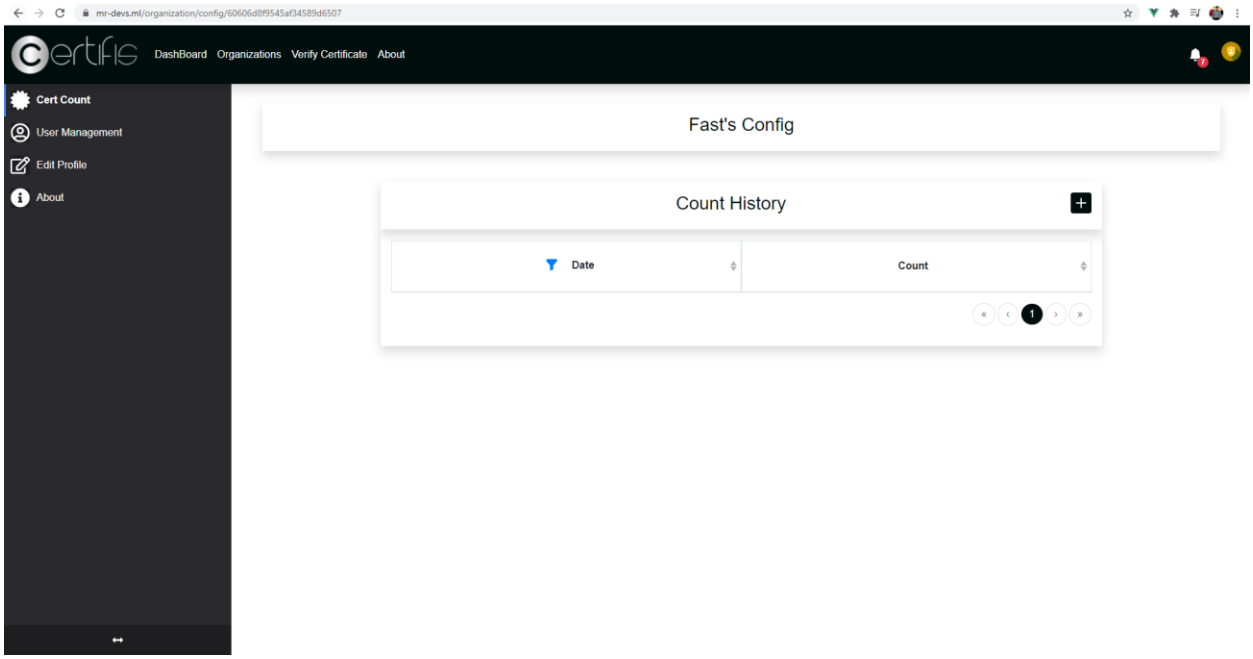


Figure 5.15 - Client Organization Count History Web Interface

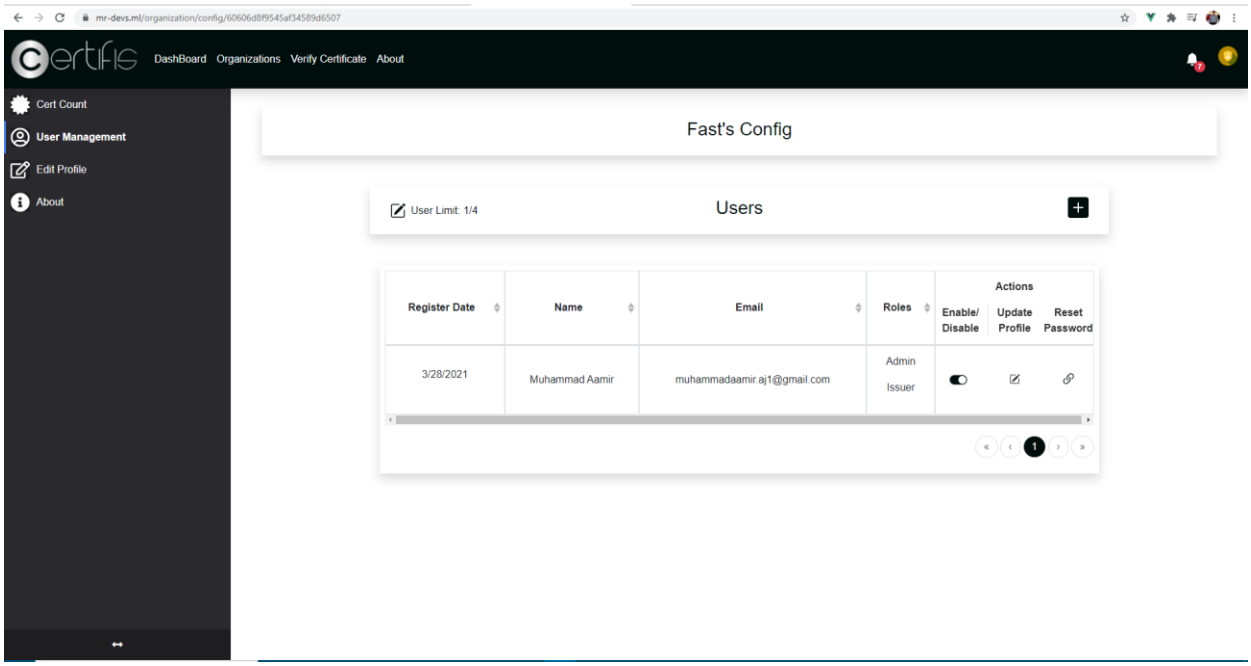


Figure 5.16 - Client Organization User Management Web Interface

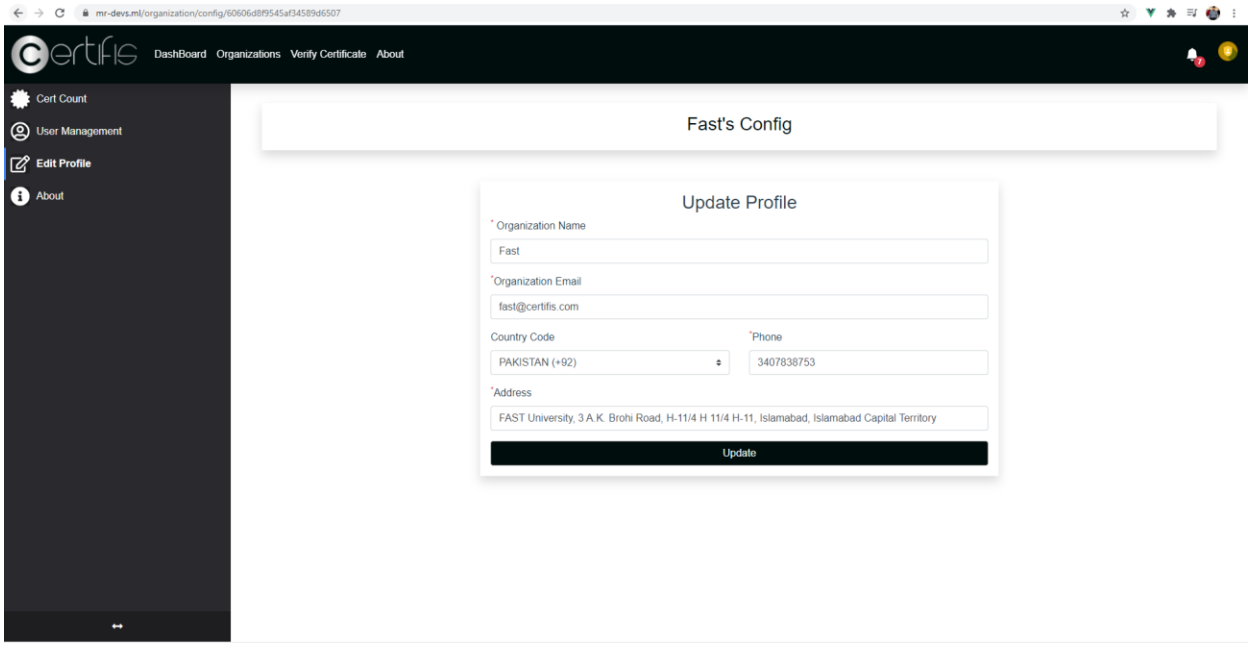


Figure 5.17 - Client Organization Update Profile Web Interface

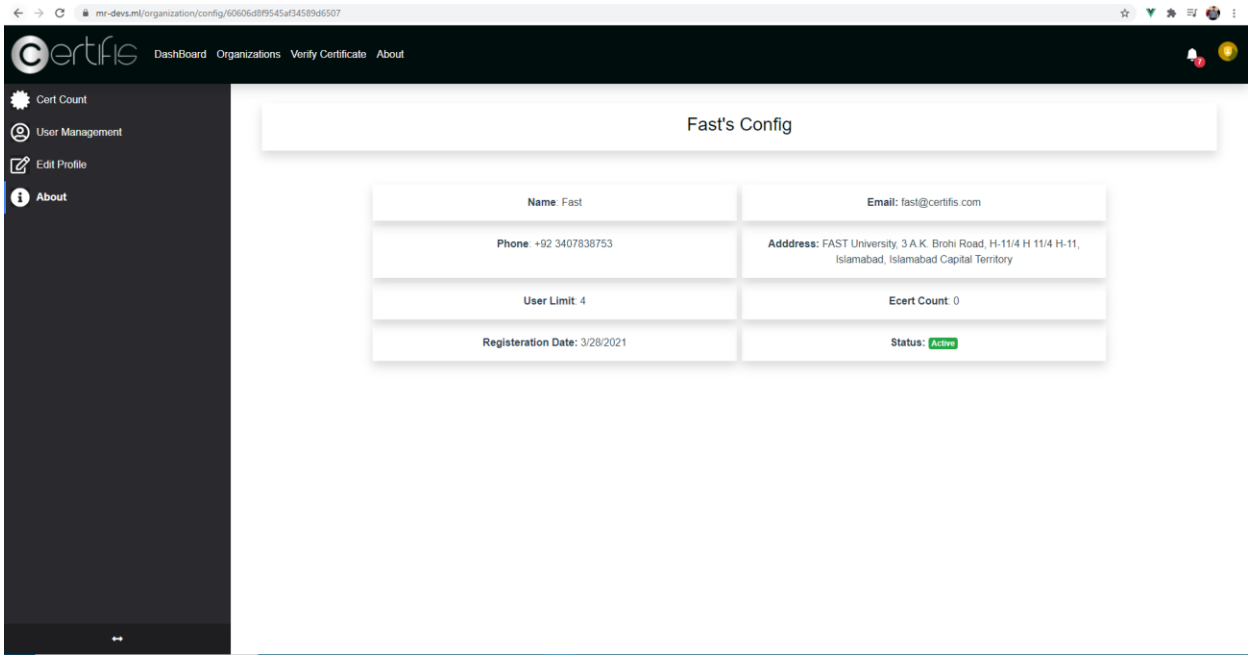


Figure 5.18 - Client Organization About Web Interface

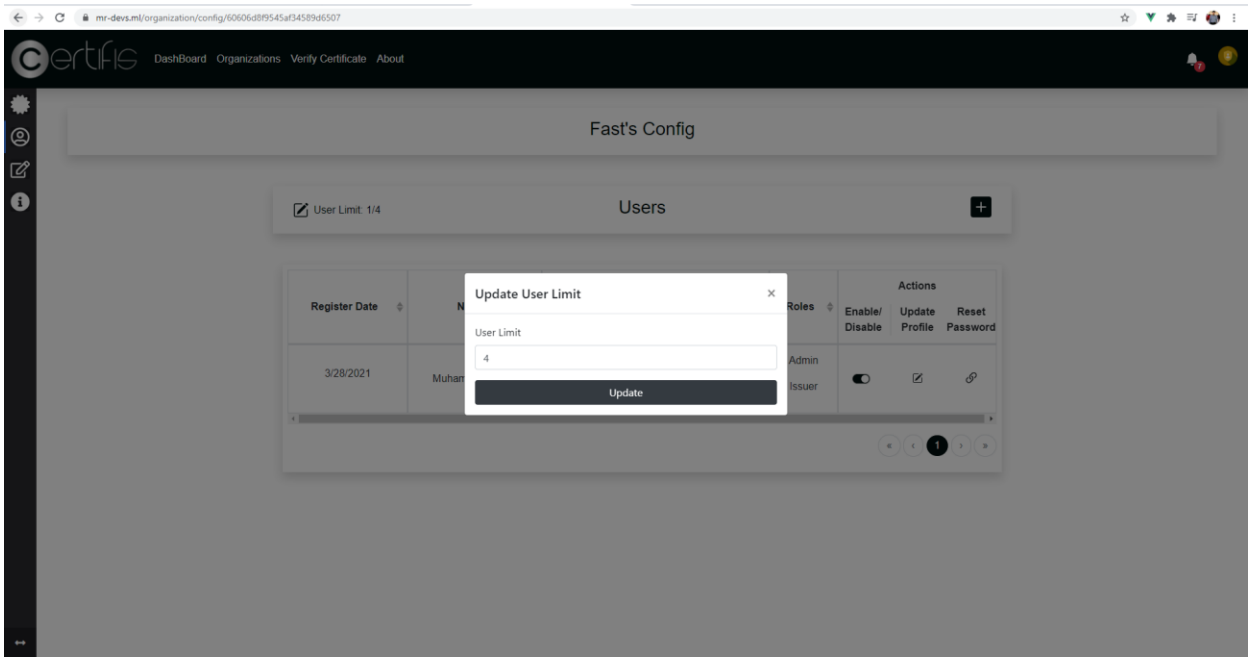


Figure 5.19 - Client Organization User Limit Web Interface

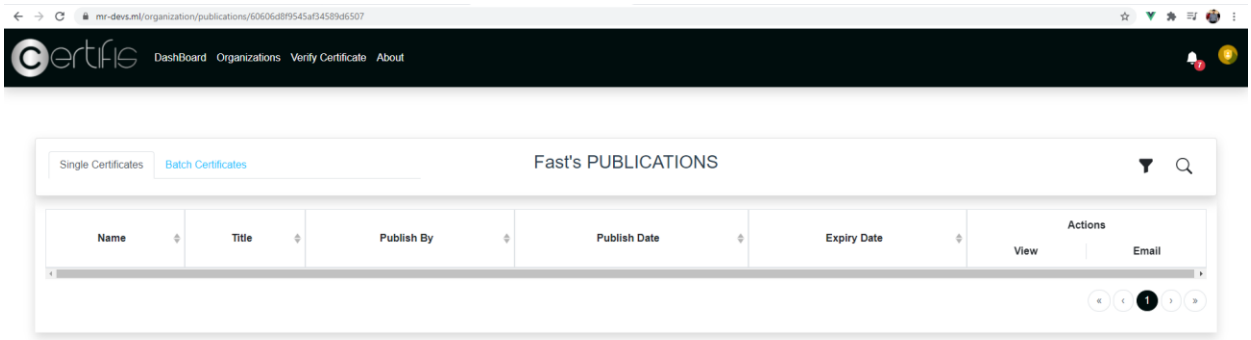


Figure 5.20 - Client Organization Published Single Certificate View Web Interface

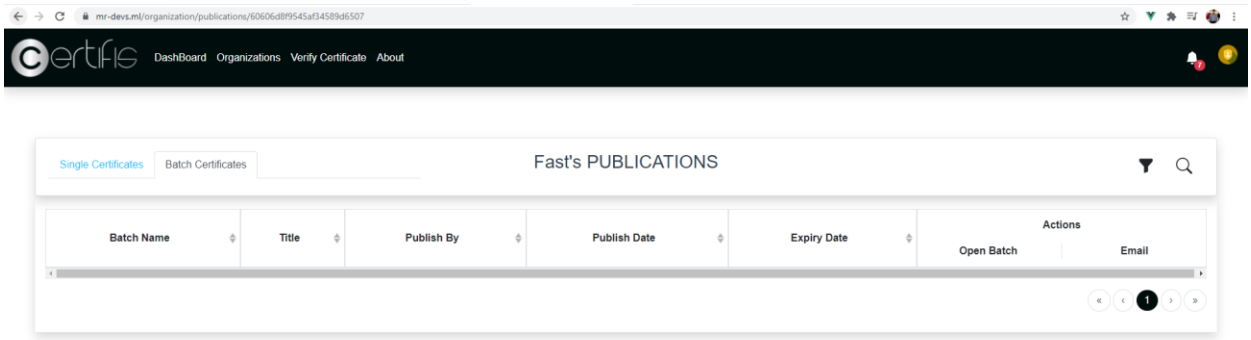


Figure 5.21 - Client Organization Published Batch View Web Interface

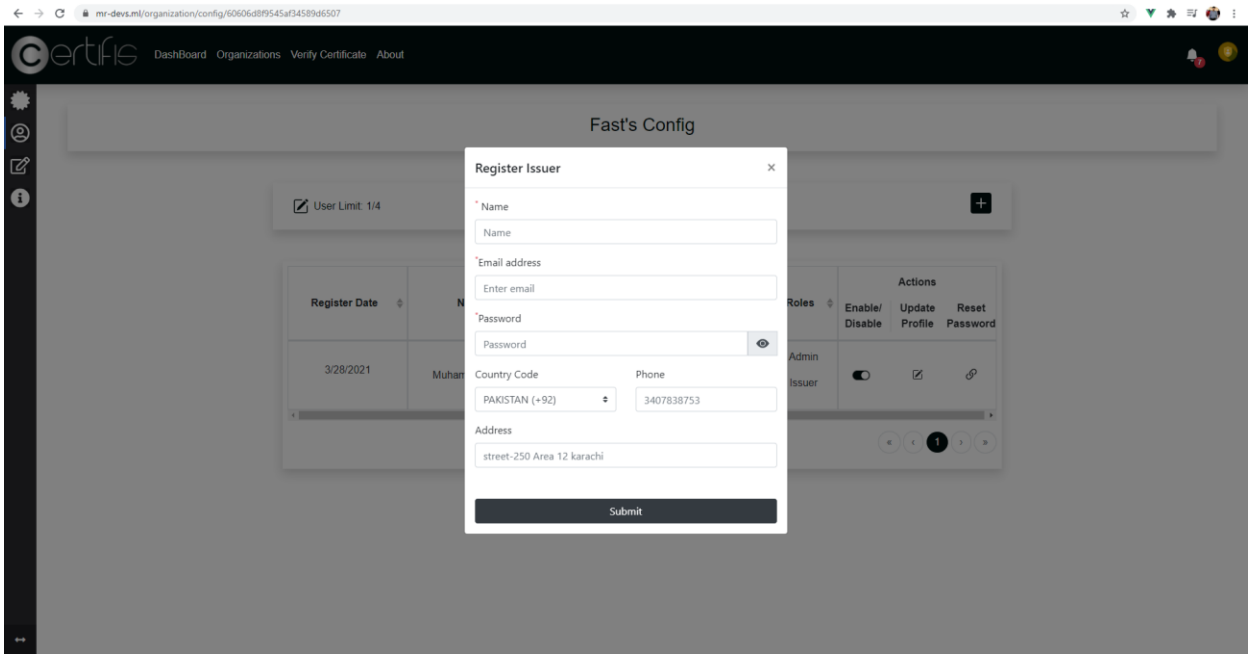


Figure 5.22 - User Registration Web Interface

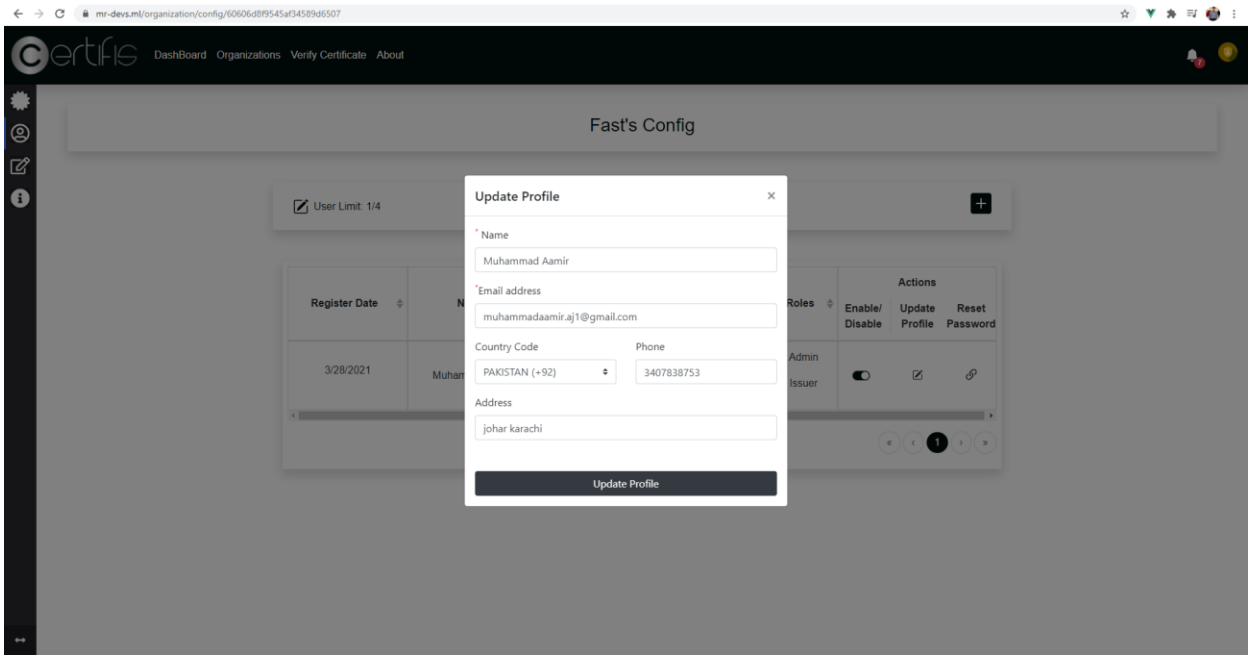


Figure 5.23 - Update User Profile Web Interface

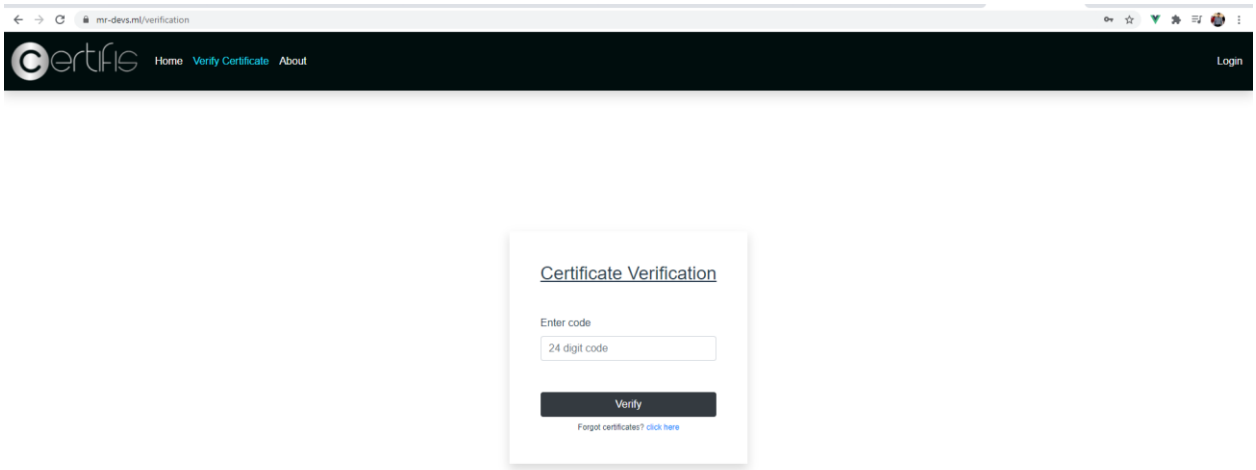


Figure 5.24 - Certificate Verification Web Interface

6. Implementation

This chapter contains code of most critical part of our project such as smart contract, publish and verify. This chapter also contain component, deployment and state transition diagram of our project.

6.1 Code

6.1.1 Smart Contract

```
'use strict';

const { Contract } = require('fabric-contract-api');

class ecert extends Contract {
   //use to initiate the ledger after installing chain code on peers
   async initLedger(ctx) {
      console.info('============= START : Initialize Ledger ===========');
      const Certificates = [
         {
            email: 'muhammadrafay151@gmail.com',
            Name: 'Muhammad Rafay',
            Title: 'Js-Certifications',
            description: 'xyz',
            Organizations: 'Uit',
         },
      ];
      for (let i = 0; i < Certificates.length; i++) {
         Certificates[i].docType = 'certificate';
         await ctx.stub.putState('ecert' + i, Buffer.from(JSON.stringify(Certificates[i])));
         console.info('Added <--> ', Certificates[i]);
      }
      console.info('============= END : Initialize Ledger ===========');
   }
   async QueryCertificate(ctx, CertificateNumber) {
      // retrieve the data from the ledger
      const CertificateAsBytes = await ctx.stub.getState(CertificateNumber);
      if (!CertificateAsBytes || CertificateAsBytes.length === 0) {
         // throw exception when certificate not found
         throw new Error(`${CertificateNumber} does not exist`);
      }
      // return data
      return CertificateAsBytes.toString();
   }
   async PublishCertificate(ctx, data) {
      console.info('============= START : Create Certificate ===========');
      //data parsing
      data = JSON.parse(data);
      let key = data._id;
      delete data._id;
      //insert the certificate data in the ledger
```

```
        await ctx.stub.putState(key, Buffer.from(JSON.stringify(data)));
        console.info('============= END : Certificate created ===========');
    }
    async PublishBatch(ctx, data) {
         //data parsing
        let batch = JSON.parse(data);
         //iteratively insert the batch certificate data in the ledger
        for (let x = 0; x < batch.length; x++) {
            let key = batch[x]._id;
            delete batch[x]._id;
            await ctx.stub.putState(key, Buffer.from(JSON.stringify(batch[x])));
        }
    }
}
module.exports = ecert;
```

6.1.1.1 Explanation

The smart contract contains the method which will be invoked in order to read and write data to the blockchain ledger.

6.1.2 Publish

```
const express = require('express');
const router = express.Router()
const Auth = require('../Auth/Auth')
const Roles = require('../js/Roles')
const batch_cert = require('../models/batch_certificates')
const batch = require('../models/batch')
const cert = require('../models/certificate');
const MsgBroker = require("../MessageBroker/publisher")
const CountHandler = require("../js/CountHandler");
const config = require("config")
const { StatusCodeException } = require('../Exception/StatusCodeException');
const NotificationHandler = require("../js/NotificationHandler");
const Constants = require("../Constants");
router.post("/single", Auth.authenticateToken, Auth.CheckAuthorization([Roles.Admin,
Roles.Issuer]), async (req, res) => {
    try {
        //reduce the count
        await CountHandler.ReduceCount(req.user.org_id, 1);

        //publishing info
        let publish = {
            status: true,
            publisher_name: req.user.name,
            publisher_email: req.user.email,
            publish_date: Date.now(),
            processing: false
        }
        //if blockchain enabled publish certificate to blockchain
        if (req.app.get("BlockChain_Enable")) {
```

```
        //load the certificate and mark it as processing
        let ct = await cert.findOneAndUpdate({ _id: req.body.id, 'issuedby.org_id': req.user.org_id, 'publish.status': false, 'publish.processing': false }, { $set: { 'publish.processing': true } }).lean()
        if (ct) {
          if (config.get("app.debugging") === true) {
            const io = req.app.get("socketio");
            ct.message = "send to message queue";
            io.to("debugging").emit("log", ct);
          }
          //send publish request to message broker
          await MsgBroker.send(true, { user: req.user, certid: req.body.id })

        }
        res.send("Processing started we will notify u soon")
      }
      else {
        // non blockchain instance
        try {
          //find certificate and chnge the status
          let crt = await cert.findOneAndUpdate({ _id: req.body.id, 'issuedby.org_id': req.user.org_id, 'publish.status': false }, { $set: { publish: publish } })
          if (crt) {
            //send notification
            let message = `${crt.title} certificate with id: ${crt._id} has been published`;
            await NotificationHandler.NewNotification(req.user, message, Constants.Private);
            res.status(200).send("Published successfully")
          } else {
            //certificate not exist rollback the count
            await CountHandler.IncreaseCount(req.user.org_id, 1);
            res.status(404).send("certificate not found")
          }
        } catch (err) {
          //error while updating certificate so rollback the count
          await CountHandler.IncreaseCount(req.user.org_id, 1);
          res.status(500).send(err)
        }
      }
    } catch (err) {
      //status code exception is used by logic handlers to throw error
      if (err instanceof StatusCodeException) {
        res.status(err.StatusCode).send(err.Message)
      } else {
        //unknown errors
        res.status(500).send(err)
      }
    }
})
```

```
router.post("/batch", Auth.authenticateToken, Auth.CheckAuthorization([Roles.Admin, Roles.Issuer]), async (req, res) => {
   try {
      //check if the batch exist against the provided id
      let bt = await batch.findOne({ _id: req.body.id, 'createdby.org_id': req.user.org_id, 'publish.status': false })
      if (bt) {
         let bcert = await batch_cert.find({ batch_id: req.body.id }).countDocuments()
         //batch should contain atleast more than 1 certificate
         if (bcert && bcert > 1) {
            await CountHandler.ReduceCount(req.user.org_id, bcert);

            //if blockchain enabled publish certificate to blockchain
            if (req.app.get("BlockChain_Enable")) {
               //load the certificate and mark it as processing
               let bt = await batch.findOneAndUpdate({ _id: req.body.id, 'createdby.org_id': req.user.org_id, 'publish.status': false, 'publish.processing': false }, { $set: { 'publish.processing': true } }).lean();
               if (bt) {
                  if (config.get("app.debugging") === true) {
                     const io = req.app.get("socketio");
                     bt.message = "send to message queue";
                     io.to("debugging").emit("log", bt);
                  }
                  //send publish request to message broker
                  await MsgBroker.send(false, { user: req.user, batchid: req.body.id })
                  res.send("Processing started we will notify u soon")
               }
               else
                  res.status(404).send()
            } else {
               //publishing info
               let publish = {
                  status: true,
                  publisher_name: req.user.name,
                  publisher_email: req.user.email,
                  publish_date: Date.now(),
                  processing: false
               }
               try {
                  //send notification
                  await batch.findOneAndUpdate({ _id: req.body.id, 'createdby.org_id': req.user.org_id, 'publish.status': false }, { $set: { publish: publish } })
                  let message = `${bt.batch_name} batch with id: ${bt._id} has been published`;
                  await NotificationHandler.NewNotification(req.user, message, Constants.Private);
                  res.send("Batch Published...")
               } catch (err) {
                  //error while updating certificate so rollback the count
```

```
                    await CountHandler.IncreaseCount(req.user.org_id, bcert);
                    res.status(500).send(err)
                }
            }
        } else {
            res.status(409).send("batch must have more than 1 certificate to publish")
        }
    } else {
        return res.status(404).send("batch not found")
    }

} catch (err) {
    //status code exception is used by logic handlers to throw error
    if (err instanceof StatusCodeException) {
        res.status(err.StatusCode).send(err.Message)
    } else {
        //unknown errors
        res.status(500).send(err)

    }
}
})

module.exports = router
```

6.1.2.1 Explanation

This API will be invoked in order to publish data to the blockchain, actually this API sends request to the message broker and then the available consumers process the request.

6.1.3 Verify

```
const express = require('express');
const router = express.Router()
const batch = require('../models/batch')
const batch_cert = require('../models/batch_certificates')
const cert = require('../models/certificate');
const mongoose = require("mongoose");
const { GetCertificate } = require("../BlockChain/query")
router.get('/:id', async (req, res) => {
    try {
        if (!mongoose.Types.ObjectId.isValid(req.params.id))
            return res.status(400).send("Invalid Code");
        //if blockchain enabled it will query from database
        if (req.app.get("BlockChain_Enable")) {
            //query the certificate from blockchain using id
            let result = await GetCertificate(req.params.id)
            if (!result)
                return res.status(404).send()
            result = JSON.parse(result)
            result._id = req.params.id
            result.blockchain = true
```

```
        }
        else {
            //for non blockchain instance of application
            //first find in single cert collection
            var result = await cert.findOne({ _id: req.params.id, 'publish.status': true }).lean()
            if (!result) {
                //if not found in single cert collection than find in batch cert collections
                var bcert = await batch_cert.findOne({ _id: req.params.id }, { updatedby: 0 })
                if (bcert) {
                    var b1 = await batch.findOne({ _id: bcert.batch_id, 'publish.status': true }, { updatedby: 0 }).lean()
                    if (b1) {
                        //mapping fields
                        b1.issue_date = bcert.issue_date
                        b1.batch_id = b1._id
                        b1._id = bcert._id
                        b1.name = bcert.name
                        b1.email = bcert.email
                        b1.blockchain = false
                    } else {
                        return res.status(404).send()
                    }
                }
                else {
                    return res.status(404).send()
                }
                result = b1
            }
        }
        //check the certificate validity
        if (result.expiry_date && new Date(result.expiry_date) - Date.now() < 0) {
            result.is_expired = true
            result.message = "The Certificate is verfied but the validity is expired"
        } else {
            result.is_expired = false
            result.message = "The Certificate is verfied"
        }
        res.json(result)
    }
    catch (err) {
        //unknown errors
        res.status(500).send()
    }
})
module.exports = router
```

6.1.3.1 Explanation

This API sends request directly to the blockchain to verify and retrieve the certificate.

6.2 Component Diagram

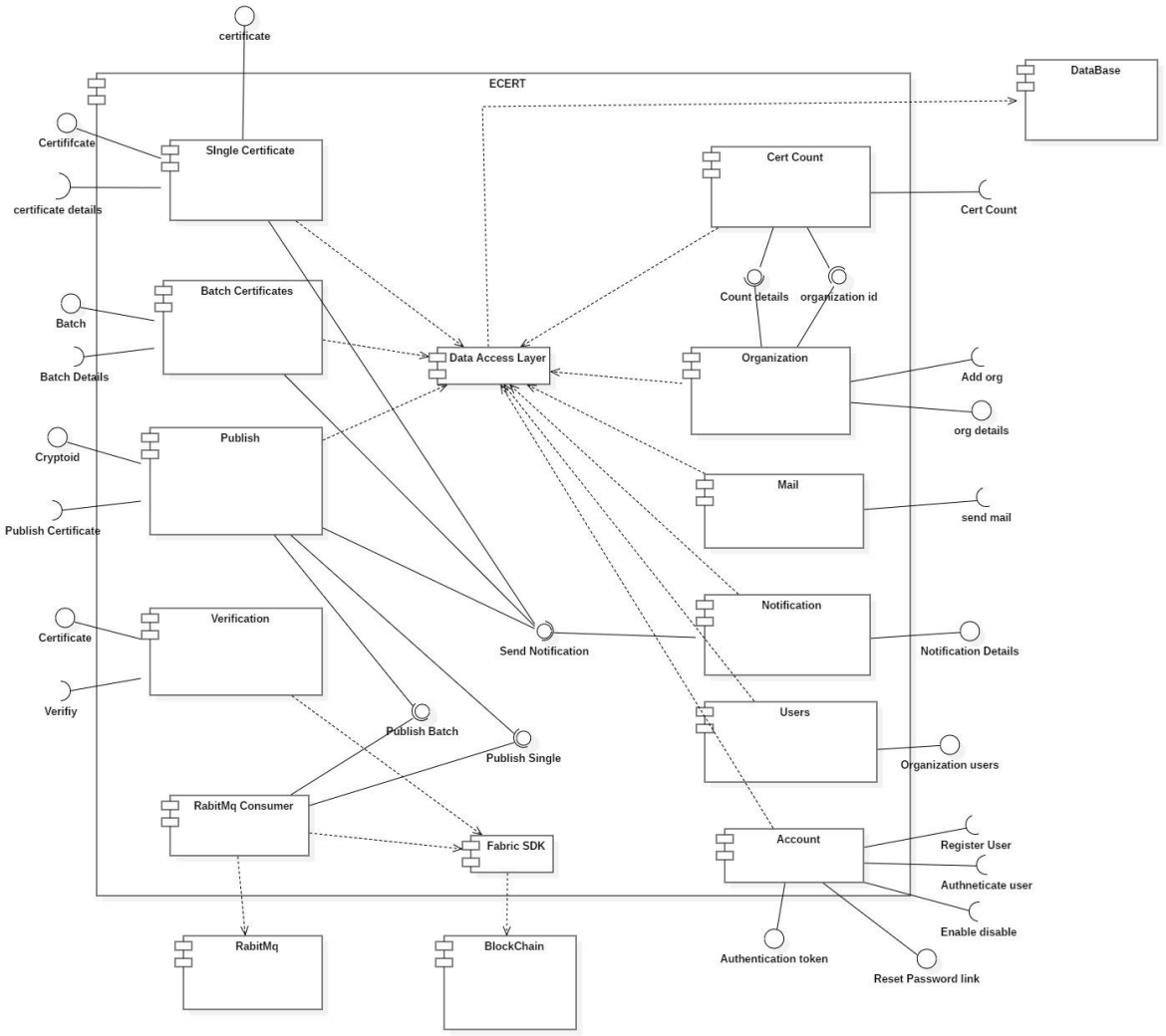


Figure 6.1 - Component Diagram

The figure 6.1 shows all the components of our project. This diagram shows how these components interact with each other.

6.3 Deployment Diagram

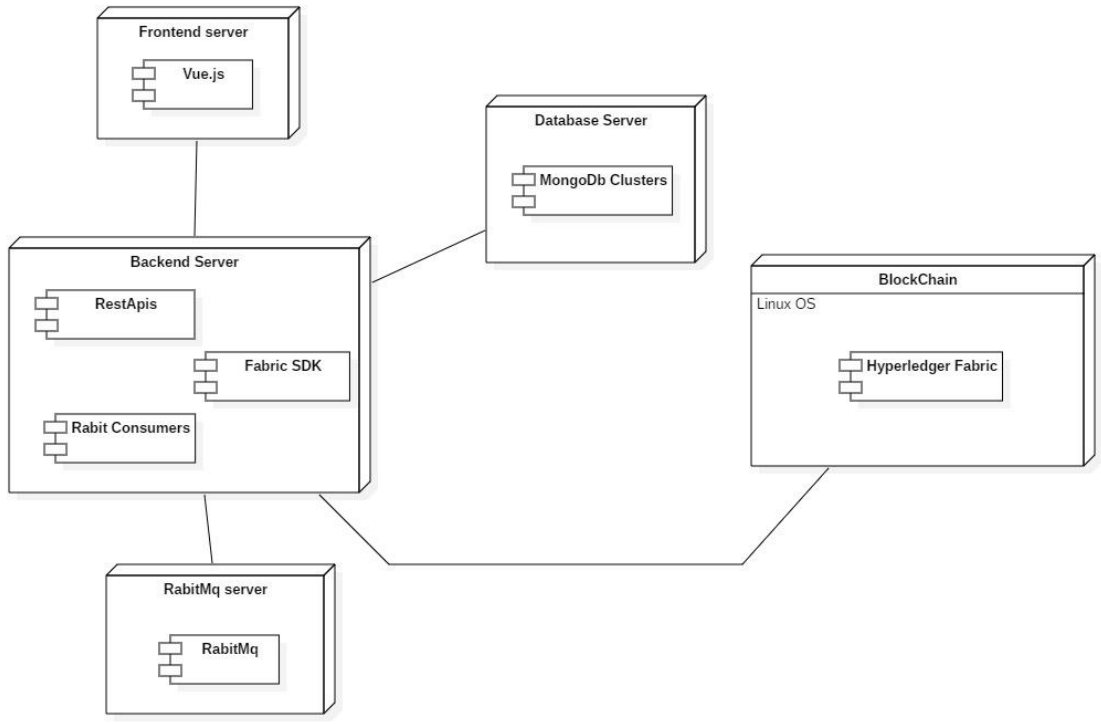


Figure 6.2 - Deployment Diagram

The figure 6.2 shows the process deployment of our project. This diagram shows how one can deploy our project what they need to install in which operating system, to run our project successfully.

6.4 State Transition Diagram

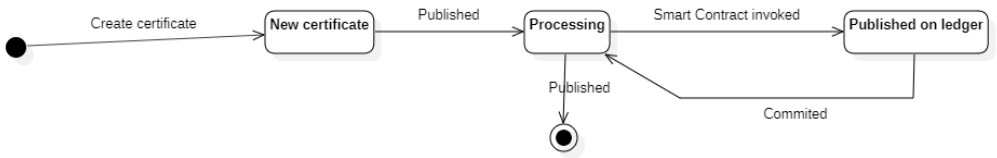


Figure 6.3 - Publish's State Transition Diagram

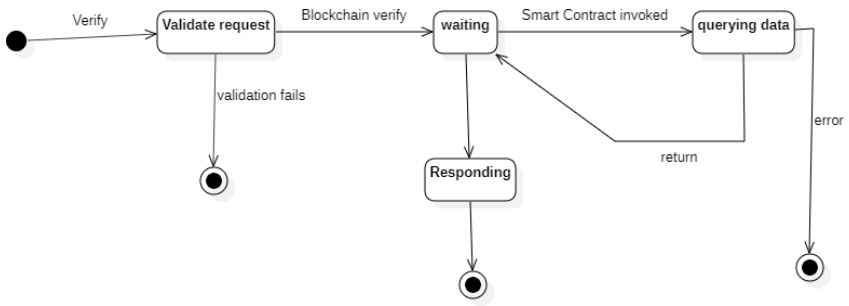


Figure 6.4 - Verification's State Transition Diagram

7. Testing

In this chapter we have cover black box and white box testing of our most logical components. Below we have attached all test cases with their results. The black box testing was carried out manually and for white box testing we used mocha and chaijs package for JavaScript where Mocha is a JavaScript test framework running on Node. Mocha allows asynchronous testing, test coverage reports, and use of any assertion library. White chai is a BDD / TDD assertion library for NodeJS and the browser that can be delightfully paired with any JavaScript testing framework.

7.1 White Box Testing

The testing plan for white box testing is given below:

First, we identified the most logical code from our project then from that we selected the most critical code and performed white box testing. The code we selected is publish single certificates, publish batches and verify certificates. Then we drew cyclometric complexity and selected the appropriate tool to perform testing.

7.1.1 Publish Single Certificate

7.1.1.1 Code

```
1//Entry router.post("/single", Auth.authenticateToken,
Auth.CheckAuthorization([Roles.Admin, Roles.Issuer]), async (req, res) =>
{
2    try {
3        let ct = await cert.findOneAndUpdate({ _id: req.body.id,
'issuedby.org_id': req.user.org_id, 'publish.status': false, 'publish.processing':
false }, { $set: { 'publish.processing': true } }).lean()

4        if (ct)
        {
5            if (config.get("app.debugging") === true)
            {

6                const io = req.app.get("socketio");
7                ct.message = "send to message queue";
8                io.to("debugging").emit("log", ct);
9//endif              }
10            await MsgBroker.send(true, { user: req.user, certid: req.body.id })

11//endif        }
12        res.send("Processing started we will notify u soon")

    }
    catch (err)
    {
13        console.log(err)
14       res.status(500).send()
15//endtrycatch}
16//exit})
```

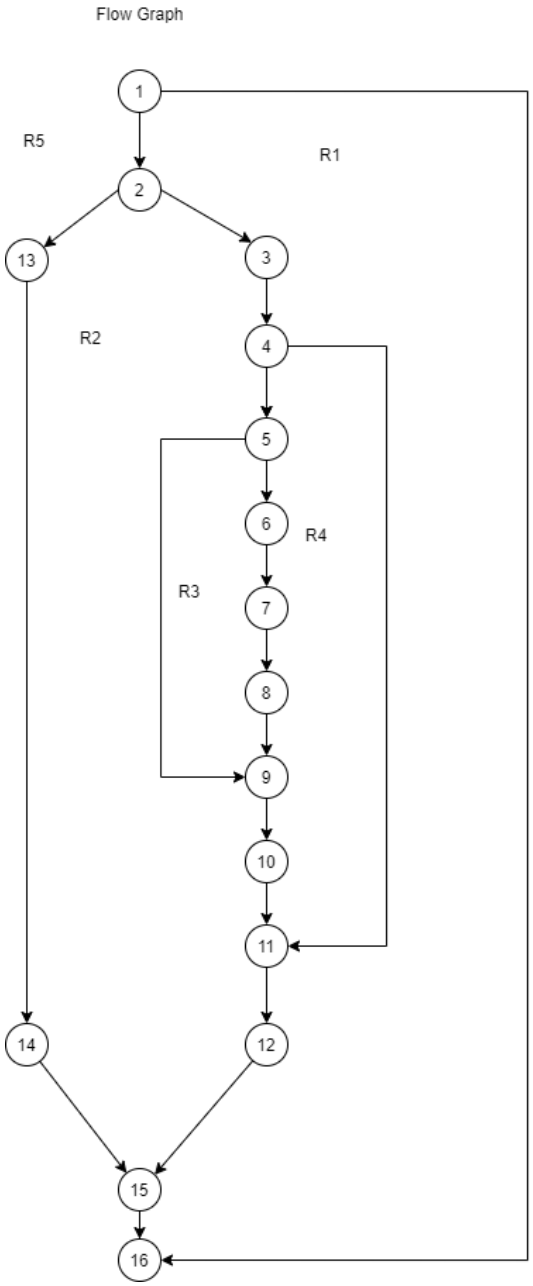


Figure 7.1 - Publish Single Certificate Flow Graph

7.1.1.2 Cyclometric Complexity

V(G)=Edges – Node + 2 = 18-15+2=5

V(G)=Predicate node + 1 = 4+1=5

V(G)=Open Region + Close Region = 5

7.1.1.3 Path

1. 1-16
2. 1-2-13-14-15-16
3. 1-2-3-4-11-12-15-16
4. 1-2-3-4-5-9-10-11-12-15-16
5. 1-2-3-4-5-6-7-8-9-10-11-12-15-16

7.1.1.4 Test Cases

Table 7.1 - Publish Single Certificate Test Cases

Test Case ID	**Test Scenario**	**Test Data**	**Expected Result**	**Actual Result**	**Valid/ Invalid**
Path-01.	Restrict accesses of user who are not an admin nor an issuer	{“id”:” 60927dd270c7f13be8c7e39f”}	Return message “Unauthorized”.	Message is returned “Unauthorized”.	Valid
Path-02.	Check If any unexpected error happens	null	Log the error and return status code 500.	Status of 500 is returned and Log error obtained.	Valid
Path-03.	Ff certificate is already under publishing process	{“id”:” 60927dd270c7f13be8c7e39f”}	Return message "Processing already started we will notify you soon".	Message Returned "Processing already started we will notify you soon".	Valid
Path-04.	Check if debugging mode is disable	{“id”:” 60927dd270c7f13be8c7e39f”}	Don’t Generate debugging logs under debugging section.	Debugging mode is Disabled.	Valid
Path-05.	Check if proper valid single certificate is provided with debugging logs enabled	{“id”:” 60927dd270c7f13be8c7e39f”}	Return message "Processing started we will notify you soon" and generate debugging logs.	Message Returned "Processing started we will notify you soon" and logs are generated.	Valid

7.1.2 Publish Batch Certificate

7.1.2.1 Code

```
1//Entry router.post("/batch", Auth.authenticateToken,
Auth.CheckAuthorization([Roles.Admin, Roles.Issuer]), async (req, res) =>
{
2    try {
3        let bt = await batch.findOneAndUpdate({ _id: req.body.id,
'createdby.org_id': req.user.org_id, 'publish.status': false,
'publish.processing': false }, { $set: { 'publish.processing': true } }).lean();

4        if (bt)
         {
5            var bcert = await batch_cert.find({ batch_id: req.body.id
}).countDocuments()

6            if (bcert && bcert > 1)
            {

7                if (config.get("app.debugging") === true)
                {
8                    const io = req.app.get("socketio");
9                    bt.message = "send to message queue";
10                   io.to("debugging").emit("log", bt);
11//endif         }
12               await MsgBroker.send(false, { user: req.user, batchid: req.body.id
})
13               res.send("Processing started we will notify u soon")
            }
            else
            {
14               res.status(409).send("batch must have more than 1 certificate to
publish")
15//endif    }
        }

        else
        {
16          return res.status(404).send("batch not found")
17//endif }

    }
    catch (err)
    {
18     res.status(500).send(err)
19//endtrycatch}
20//exit}
```

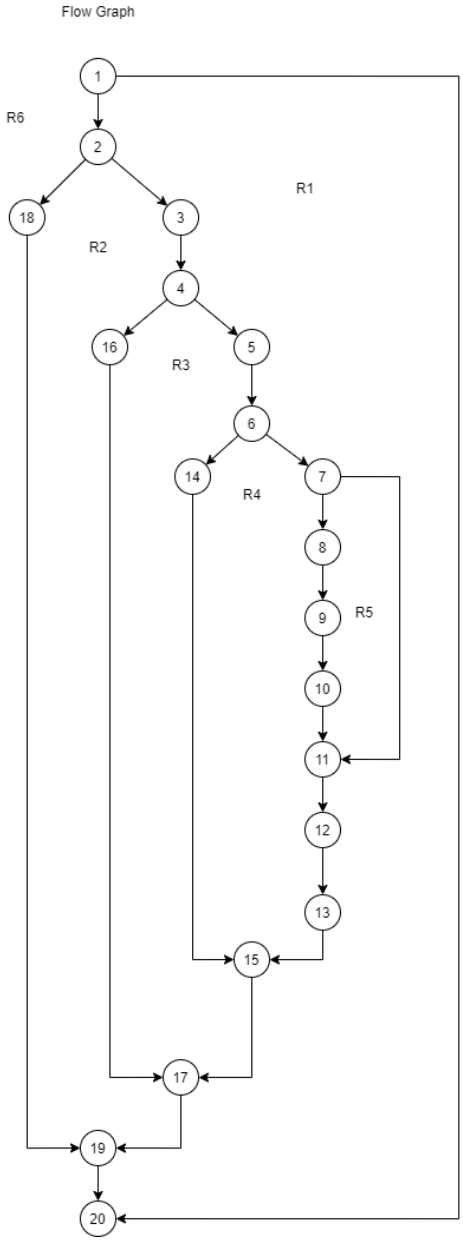


Figure 7.2 - Publish Batch Flow Graph

7.1.2.2 Cyclometric Complexity

V(G)=Edges – Node + 2 = 24-20+2=6

V(G)=Predicate node + 1 = 5+1=6

V(G)=Open Region + Close Region = 6

7.1.2.3 Paths

1. 1-20
2. 1-2-18-19-20
3. 1-2-3-4-16-17-19-20
4. 1-2-3-4-5-6-14-15-17-19-20
5. 1-2-3-4-5-6-7-11-12-13-15-17-19-20
6. 1-2-3-4-5-6-7-8-9-10-11-12-13-15-17-19-20

7.1.2.4 Test Cases

Table 7.2 - Publish Batch Test Cases

Test Case ID	Test Scenario	Test Data	Expected Result	Actual Result	Valid/ Invalid
Path-01.	Restrict accesses of user who are not an admin nor an issuer	{“id”:” 6092e2e4939a4600176f77fa”}	Return error message “Unauthorized”.	Message is returned “Unauthorized”.	Valid
Path-02.	Check If any unexpected error happens	null	Log the error and return status code 500.	Status of 500 is returned and Log error obtained.	Valid
Path-03.	if certificate is already under publishing process	{“id”:” 6092e2e4939a4600176f77fa”}	Return message "Processing already started we will notify you soon".	Message Returned "Processing already started we will notify you soon"	Valid
Path-04.	Check if a batch is sent for publishing with less than 2 certificates in it.	{“id”:” 6092e2e4939a4600176f77fa”}	Return status code “409” with error message "batch must have more than 1 certificate to publish".	Batch must have more than 1 Certificate and status of 409 returned.	Valid
Path-05.	Check if debugging mode is disable	{“id”:” 6092e2e4939a4600176f77fa”}	Don’t Generate debugging logs under debugging section.	Debugging mode is Disabled.	Valid
Path-06.	Check if proper valid batch certificate is provided with debugging logs enabled	{“id”:” 6092e2e4939a4600176f77fa”}	Return message "Processing started we will notify you soon" and generate debugging logs	Message Returned "Processing started we will notify you soon" and logs are generated.	Valid

7.1.3 Verify Certificate

7.1.3.1 Code

```
1//Entry router.get('/:id', async (req, res) =>
 {
2    try {
3        if (!mongoose.Types.ObjectId.isValid(req.params.id))
         {
4            return res.status(400).send("Invalid Code");
5//endif }

6        var result = await GetCertificate(req.params.id)
7        result = JSON.parse(result)
8        result.blockchain = true

9        if (result.expiry_date && result.expiry_date - Date.now() < 0)
        {
10            result.is_expired = true
11            result.message = "The Certificate is verfied but the validity is
expired"

        }
        else
        {
12            result.is_expired = false
13            result.message = "The Certificate is verfied"
14//endif}

15        res.json(result)
    }
    catch (err)
    {
16        console.log(err)
17        res.status(500).send()
18//endtrycatch    }
19//exit})
```

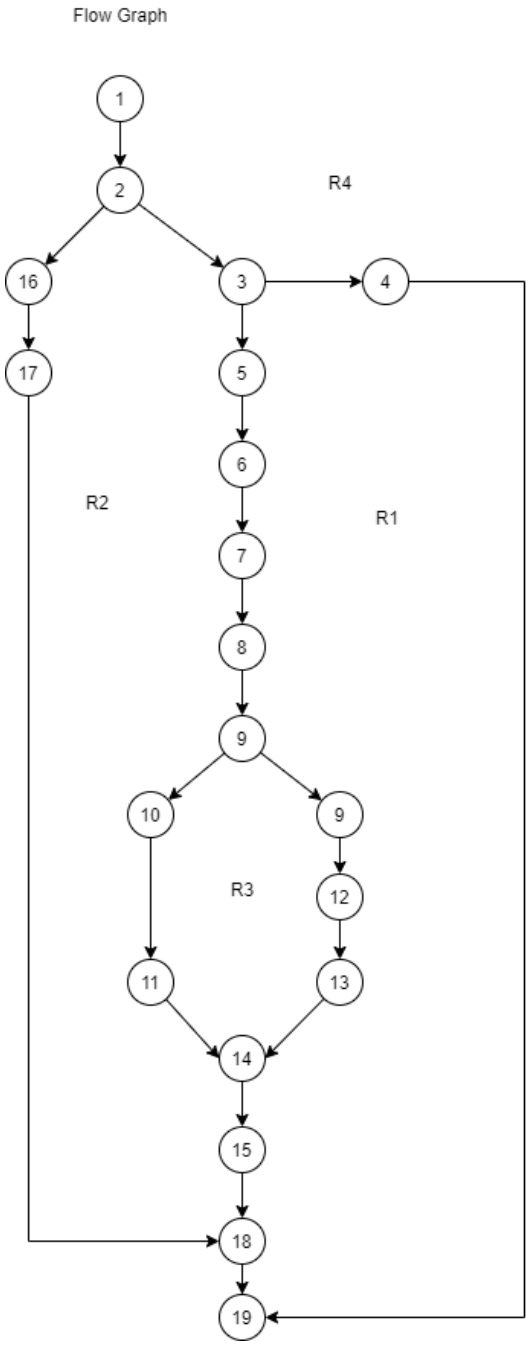


Figure 7.3 - Verify Certificate Flow Graph

7.1.3.2 Cyclometric Complexity

V(G)=Edges – Node + 2 = 21-19+2=4

V(G)=Predicate node + 1 = 3+1=4

V(G)=Open Region + Close Region = 4

7.1.3.3 Path

1. 1-2-16-17-18-19
2. 1-2-3-4-19
3. 1-2-3-5-6-8-9-10-11-14-15-18-19
4. 1-2-3-5-6-8-9-12-13-14-15-18-19

7.1.3.4 Test Cases

Table 7.3 - Verify Certificate Test Cases

Test Case ID	**Test Scenario**	**Test Data**	**Expected Result**	**Actual Result**	**Valid/ Invalid**
Path-01.	Check If any unexpected error happens	{“code”: “606cac98ec653783a1e62” }	Log the error and return status code 500	Status of 500 is returned and Log error obtained.	Valid
Path-02.	Check If Invalid verification code is provided	{“code”: “606cac98eb783a1e62”}	Return status code 400 with error message “invalid code”	The code is invalid and status 400 is obtained.	Valid
Path-03.	Check if expired certificate code is provided	{“code”: “606ca3fa11c1641728c127ec”}	Return message "The Certificate is verified but the validity is expired"	The certificate is expired and verified.	Valid
Path-04.	Check if a valid certificate code is provided	{“code”: “606cac98eb64c653783a1e62”}	Return message "The Certificate is verified"	The Certificate is verified.	Valid

7.2 Black Box Testing

Black box testing is a form of software testing in which the functionality of a software is tested without regard for its implementation or source code. Black box testing is solely focused on the software's features and requirements. In black box testing, we just rely on two simple operations: a system's input and output. For Black box testing, we used the interfaces of Publish Single Certificate, Publish Batches and Verify Certificate.

7.2.1 Publish Single Certificate Interface

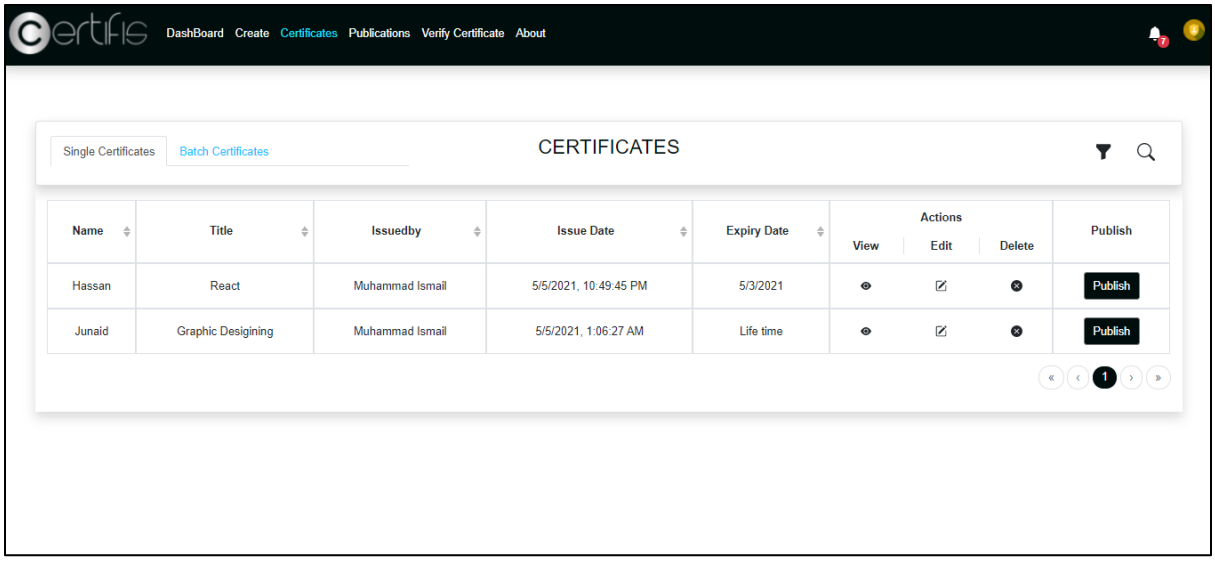


Figure 7.4 - Publish Single Certificate Interface

7.2.1.1 Functionality Testing

Table 7.4 – Publish Single Certificate Functionality Testing

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Valid/ Invalid
TF01.	Does it publish certificate to the blockchain?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Select certificate 4. Click Publish.	Should publish certificates from local database to the blockchain.	Publish certificates to the blockchain.	Valid
TF02.	Does it prompt error while publishing to the blockchain when blockchain is not connected?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Select certificate 4. Click Publish.	Should prompt error.	Prompting error.	Valid
TF03.	Does it decrease certificate count balance after successfully publishing to the blockchain?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Select certificate 4. Click Publish.	Should decrease certificate count balance.	Decreasing certificate count balance.	Valid
TF04.	Does it prompt error while publishing to the blockchain when there is insufficient certificate count balance?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Select certificate 4. Click Publish.	Should not publish certificates to the blockchain.	Not publishing to the blockchain.	Valid

7.2.1.2 Validation Testing

TV01. Button (Publish Single Certificate)

Guidelines

TV0101. Does the Cursor change?
Yes.

TV0102. Does it prompt confirmation before publishing to the blockchain?
Yes.

TV0103. Do the font size similar to the other attributes in the interface?
Yes.

TV0104. Do the font color similar to the other attributes in the interface?
Yes.

TV0105. Does the spelling correct?
Yes.

TV0106. Do the button size similar to the other button?
Yes.

TV0107. Does the expected result is displayed on click?
Yes.

7.2.2 Publish Batch Interface

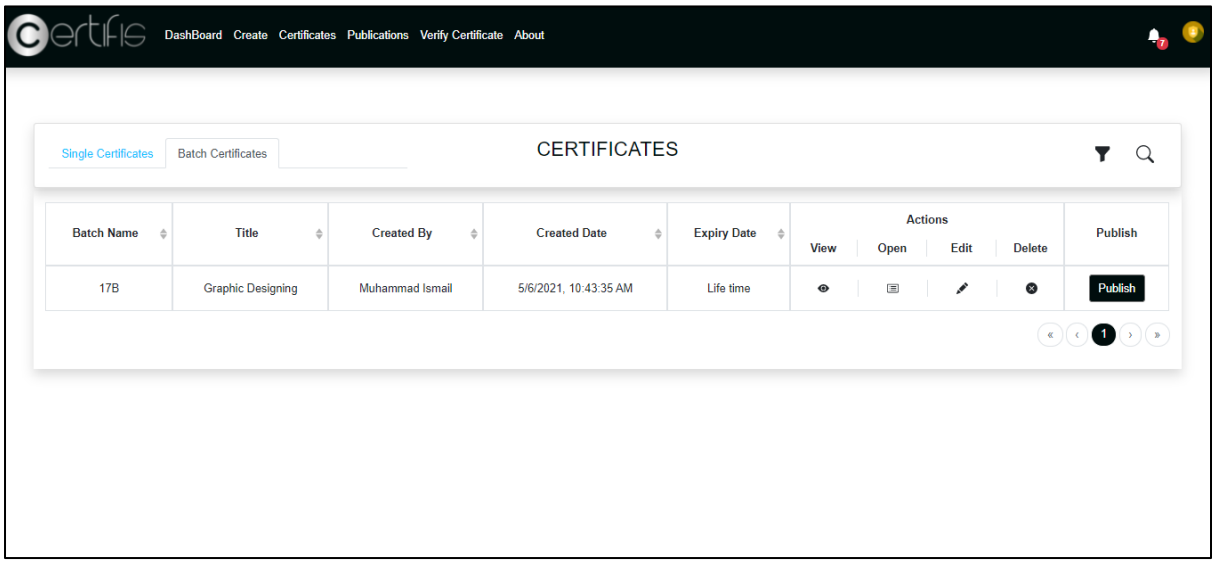


Figure 7.5 - Publish Batch Interface

7.2.2.1 Functionality Testing

Table 7.5 - Publish Batch Functionality Testing

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Valid/ Invalid
TF01.	Does it publish batch to the blockchain?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Click on Batch Certificate. 4. Select Batch. 5. Click Publish.	Should publish Batch from local database to the blockchain.	Publish certificates to the blockchain.	Valid
TF02.	Does it prompt error while publishing to the blockchain when blockchain is not connected?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Click on Batch Certificate. 4. Select Batch. 5. Click Publish.	Should prompt error.	Prompt's error.	Valid
TF03.	Does it decrease certificate count balance after successfully publishing to the blockchain?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Click on Batch Certificate. 4. Select Batch. 5. Click Publish.	Should decrease certificate count balance.	Decreases certificate count balance.	Valid
TF04.	Does it prompt error while publishing to the blockchain when there is insufficient certificate count balance?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Click on Batch Certificate.	Should not publish certificates to the blockchain.	Not publishing to the blockchain.	Valid

		4. Select Batch. 5. Click Publish.			
TF05.	Does it prompt error while publishing to the blockchain when there are less than 2 certificates in the batch?	1. Go to site: https://mr-devs.ml/ 2. Click on Certificate. 3. Click on Batch Certificate. 4. Select Batch. 5. Click Publish.	Should not allow to publish batch.	Not allowing to publish the batch.	Valid

7.2.2.2 Validation Testing

TV01. Button (Publish Single Certificate)

Guidelines

TV0101. Does the Cursor change?
Yes.

TV0102. Does it prompt confirmation before publishing to the blockchain?
Yes.

TV0103. Do the font size similar to the other attributes in the interface?
Yes.

TV0104. Do the font color similar to the other attributes in the interface?
Yes.

TV0105. Does the spelling correct?
Yes.

TV0106. Do the button size similar to the other button?
Yes.

TV0107. Does the expected result is displayed on click?
Yes.

7.2.3 Verify Certificate Interface

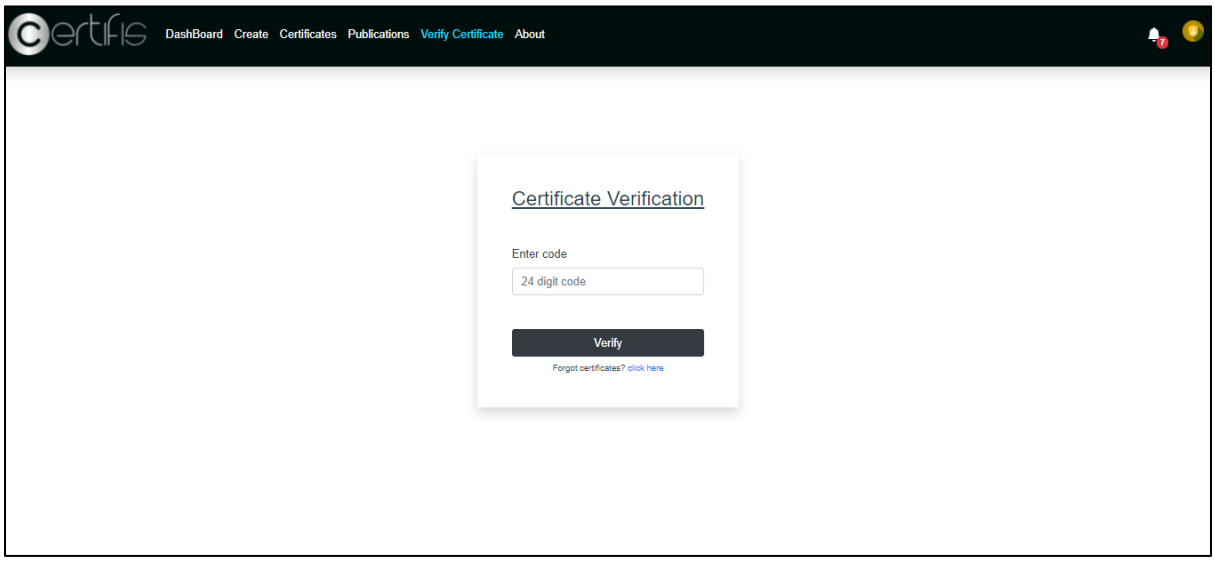


Figure 7.6 - Verify Certificate Interface

7.2.3.1 Functionality Testing

Table 7.6 - Verify Certificate Functionality Testing

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Valid/ Invalid
TF01.	Does it verify the certificate on the valid certificate ID?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5c	Certificate should be displayed.	Displays the certificate.	Valid
TF02.	Does it verify the certificate on the invalid certificate ID?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5s	Should prompt Invalid code error.	Prompts an invalid code error.	Valid
TF03.	Does it verify the expired certificate along with expiry status?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606c9f8511c1641728c127c9	Certificate should be displayed along with expiry status.	Displays the certificate along with expiry status.	Valid

7.2.4 Validation Testing

TV01. Textbox (Code)

Table 7.7 - Verify Certificate Textbox Validation Testing

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Valid/ Invalid
TV0101.	Do the textbox editable?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5c	Textbox should be editable.	Textbox is editable.	Valid
TV0102.	Does the textbox allow alphanumeric characters?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5c	Data should be acceptable.	Data is acceptable.	Valid
TV0103.	Does the textbox allow uppercase?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606CAB67EB64C653783A1E5C	Data should be acceptable.	Data is acceptable.	Valid

TV0104.	Does the textbox allow lowercase?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5c	Data should be acceptable.	Data is acceptable.	Valid
TV0105.	Does the textbox allow uppercase and lowercase both?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606CAB67EB64C653783a1e5c	Data should be acceptable.	Data is acceptable.	Valid
TV0106.	Does the textbox have maximum character limit?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5ct	Size limit should not be more than 24 characters.	Data is not acceptable.	Valid
TV0107.	Does the textbox have minimum character limit?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5	Size limit should not be less than 24 characters.	Data is not acceptable.	Valid

TV0108.	Does the textbox allow entering “Spaces” in the prefix and suffix?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606CAB67EB64C653783a1e5c	Data should not be acceptable.	Textbox removes space automatically.	Valid
TV0109.	Does the textbox allow entering “Spaces” in between the verification code?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606CAB67EB64C653783a1e5c	Data should not be acceptable.	Data is not acceptable.	Valid
TV0110.	Does the textbox accept only spaces as input?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code:	Data should not be acceptable.	Data is not acceptable.	Valid
TV0111.	Does the textbox allow any special characters?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606CAB67EB64C653783a1e(c	Data should not be acceptable.	Prompts Invalid Code error.	Valid

TV0112.	Does the textbox allow any special characters at the start of the verification code?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: _06CAB67EB64C653783a1e5c	Data should not be acceptable.	Prompts Invalid Code error.	Valid
TV0113.	Does the textbox allow any special characters at the end of the verification code?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606CAB67EB64C653783a1e5_	Data should not be acceptable.	Prompts Invalid Code error.	Valid
TV0114.	Does the textbox allow the user to copy verification code?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Select the text. 5. Press Ctrl+C	Code: 606CAB67EB64C653783a1e5c	Should allow coping data.	Allows coping data.	Valid

Guidelines

TV0115.	Do the textbox be selected from the Keyboard "tab " Button? Yes.
TV0116.	Do the cursor blink when clicking in the textbox? Yes.
TV0117.	Do the value visible when typed to the textbox? Yes.
TV0118.	Does the textbox’s placeholder visible or not? Yes.
TV0119.	Does the textbox is null by default? Yes.
TV0120.	Does the height and alignment of text boxes are the same throughout the site? Yes.

TV02. Button (Verify)

Table 7.8 - Verify Certificate Button Validation Testing

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Valid/ Invalid
TV0201.	Does it verify the certificate on the valid certificate ID?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5c	Certificate should be displayed.	Displays the certificate.	Valid
TV0202.	Does it verify the certificate on the invalid certificate ID?	1. Go to site: https://mr-devs.ml/ 2. Click on Verify Certificate. 3. Enter 24-digit Code. 4. Click Verify.	Code: 606cab67eb64c653783a1e5s	Should prompt Invalid code error.	Prompts an invalid code error.	Valid

Guidelines

TV0203.	Does the Cursor change? Yes.
TV0204.	Do the font size similar to the other attributes in the interface? Yes.
TV0205.	Do the font color similar to the other attributes in the interface? Yes.
TV0206.	Does the spelling correct? Yes.
TV0207.	Do the button size similar to the other button? Yes.

8. Conclusion

The goal of the project "Blockchain Based E-Certification" is to address the problems that "Certificate Issuing Organizations" are facing. The project primarily addresses the issue of certificate verification, allowing anybody to quickly verify certificates at any time.

To protect and maintain those certificates, our team determined that it was necessary to transfer the useful data into electronic data, which, of course, will require a high level of security. As a result, we created a system based on "blockchain" that allows all Certificate Issuing Organizations to be transformed into blockchain technology.

We chose blockchain technology because of its distributed and decentralized nature. When we say "distributed," we mean that each certificate will be held on numerous nodes throughout the world, which is a significantly superior form of data security. We believe that this method is secure since tampering is practically impossible.

Our technology not only ensures the security of certificates, but also allows anybody to verify certificates at any moment. We are able to do so by using Hyperledger Fabric. Hyperledger Fabric is a platform that enables the creation of Smart Contracts.

9. Future Enhancement

There can be many Future enhancement in our project there are lot of possibilities that we can add in our final year project. Starting from the user interface first of all we will focus on the designer of the certificate we will make it completely dynamic so that user can create certificates according to his/ her choice via drag and drop. Also, we will enhance our UI/ UX for better user experience.

In our application there is no feature of user's profile management so in future we will add the feature of user's profile management so that users can manage their profile. Also, we will add FAQs and Guidelines so the user can understand the system easily.

Also, we will provide developer API portal and enhance system architecture like distributed RabitMQ Consumers, Distributed Socket Servers, Dedicated Email Servers and HLF Automator for production ready HFL network.

In future we will add the feature of payment gateway and reports in our application.

10. Achievements

We have applied in Ignite funding but we have not received any funding and due to Covid’19 we have not applied in any other competitions.

11. Appendices

11.1 Annex A: Class Diagram

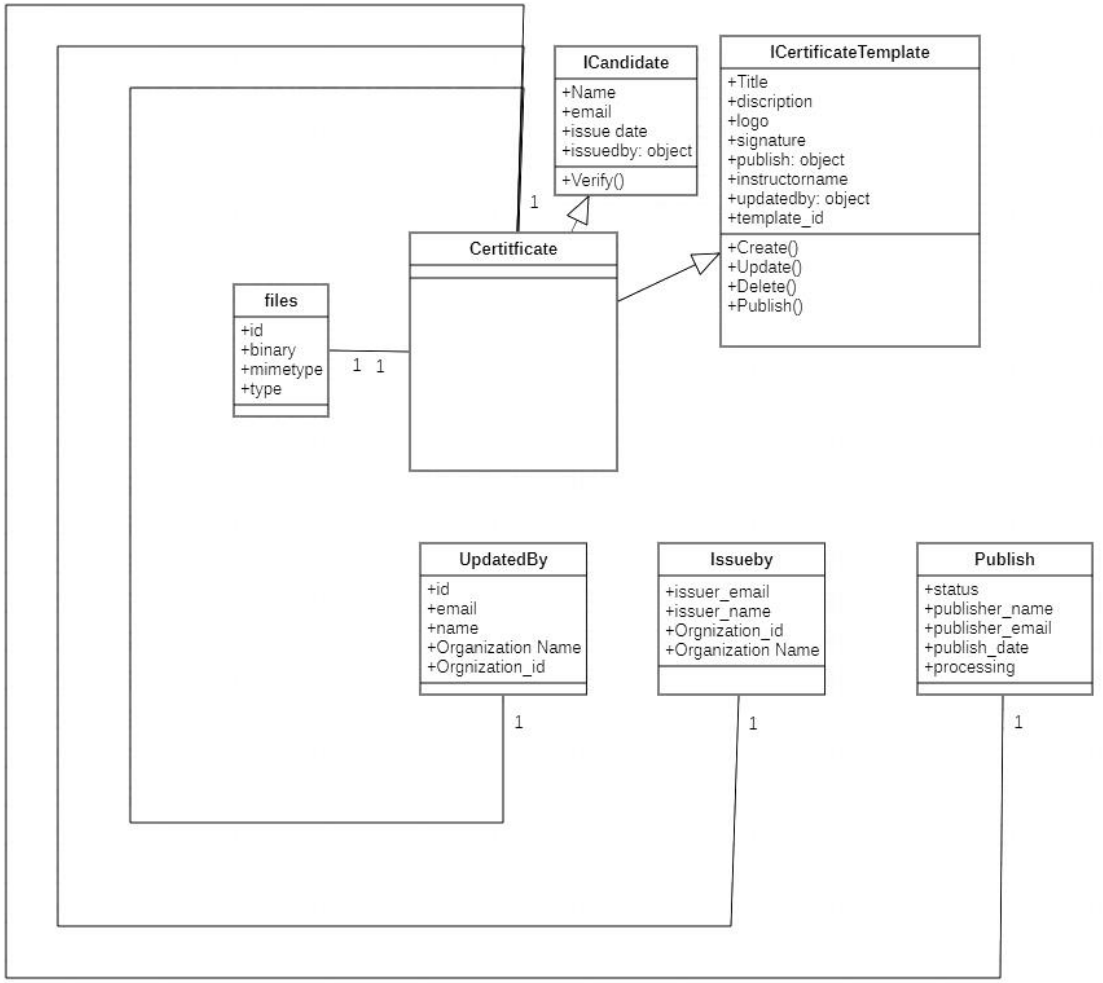


Figure 11.1 - Manage Certificate Class Diagram

The figure 11.1 shows the process of managing certificates also it shows the relationship of certificate class with all other classes.

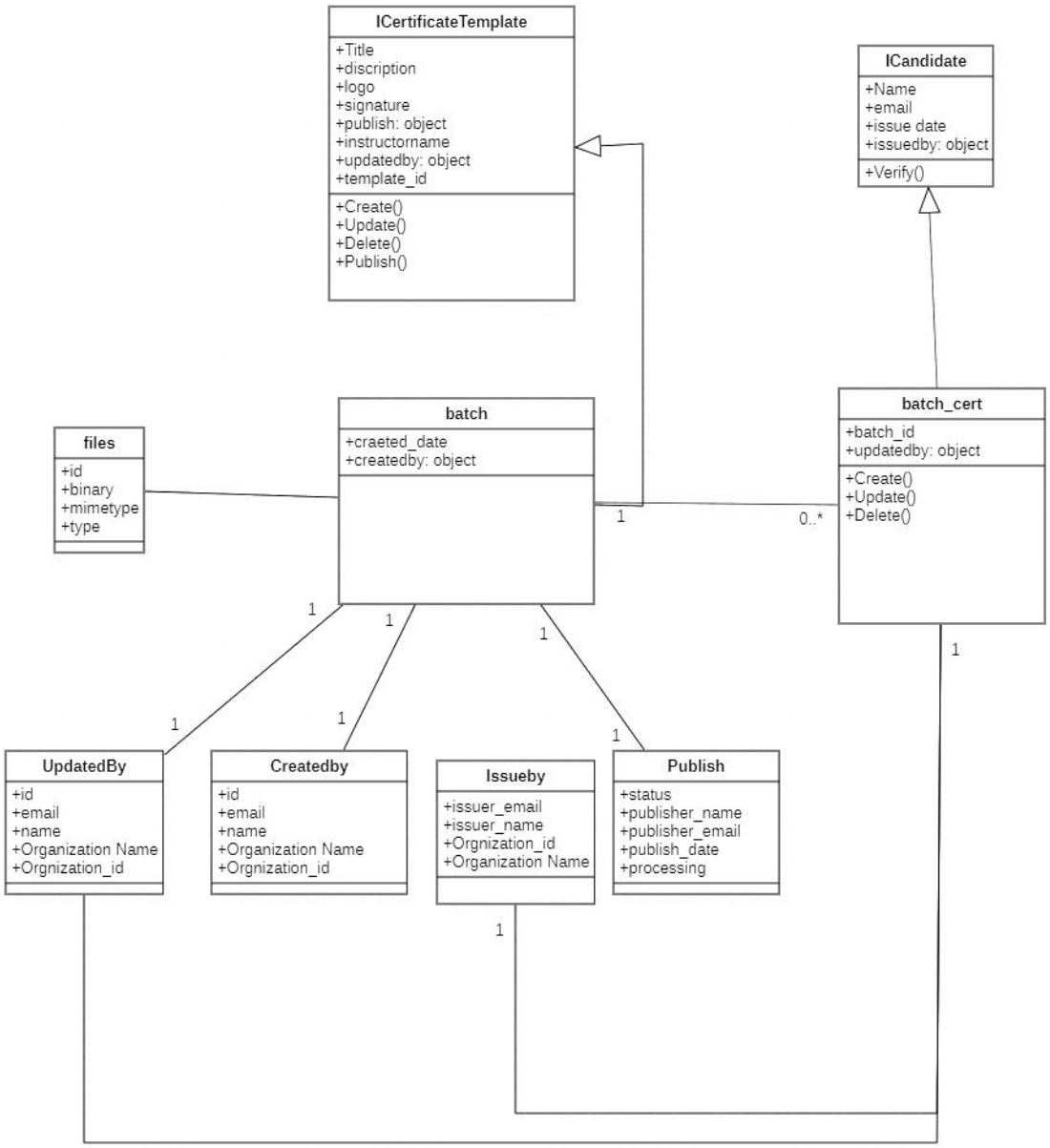


Figure 11.2 - Manage Batch and Batch Certificate Class Diagram

The figure 11.2 shows the process of managing batch and batch certificates also it shows the relationship of batch and batch certificates class with all other classes.

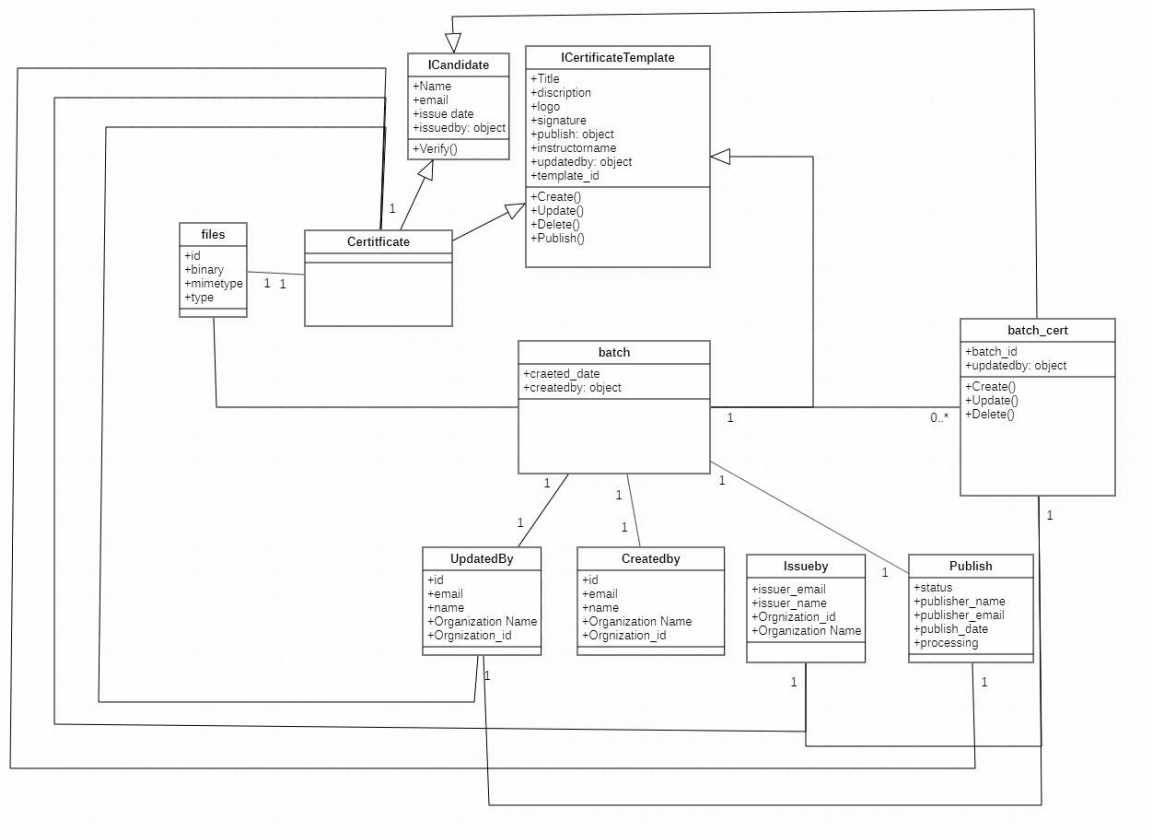


Figure 11.3 - Verify Certificate Class Diagram

The figure 11.3 shows the process of verifying certificates also it shows the relationship of all class with each other.

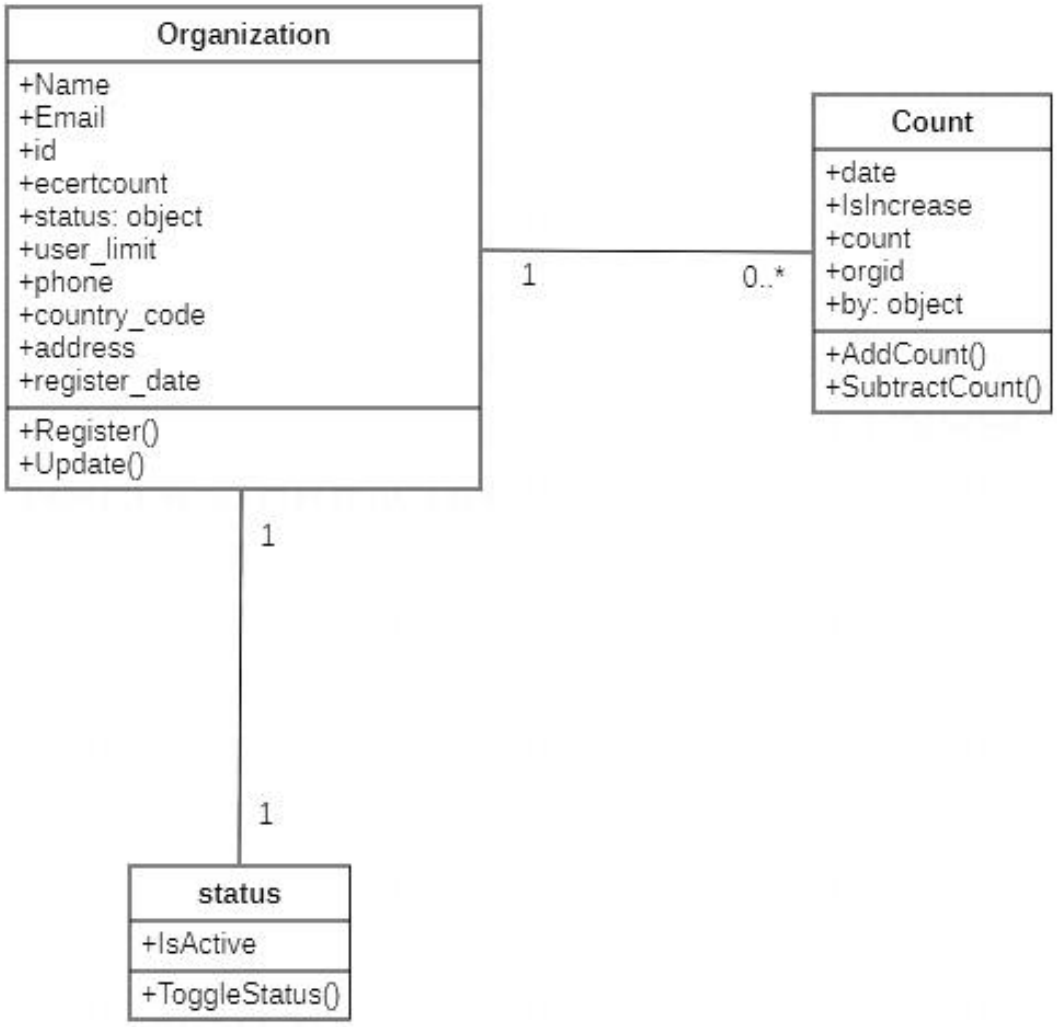


Figure 11.4 - Manage Organization Class Diagram

The figure 11.4 shows the process of managing organization also it shows the relationship of organization class with all other classes.

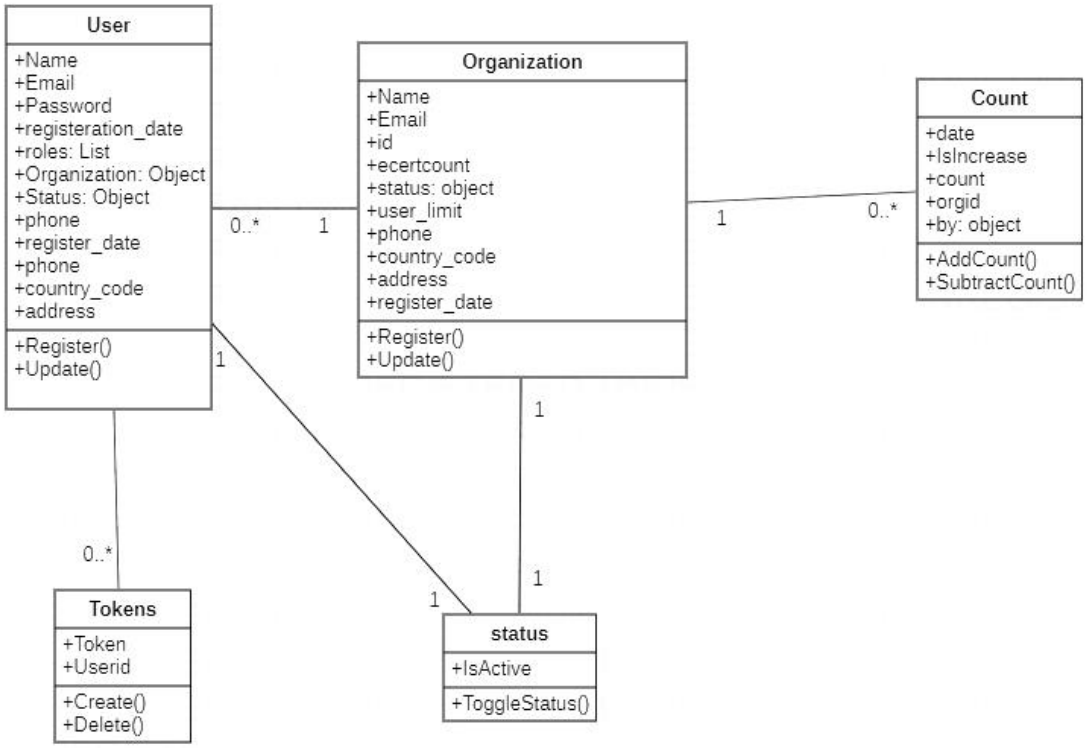


Figure 11.5 - Manage User Class Diagram

The figure 11.5 shows the process of managing user also it shows the relationship of user class with all other classes.

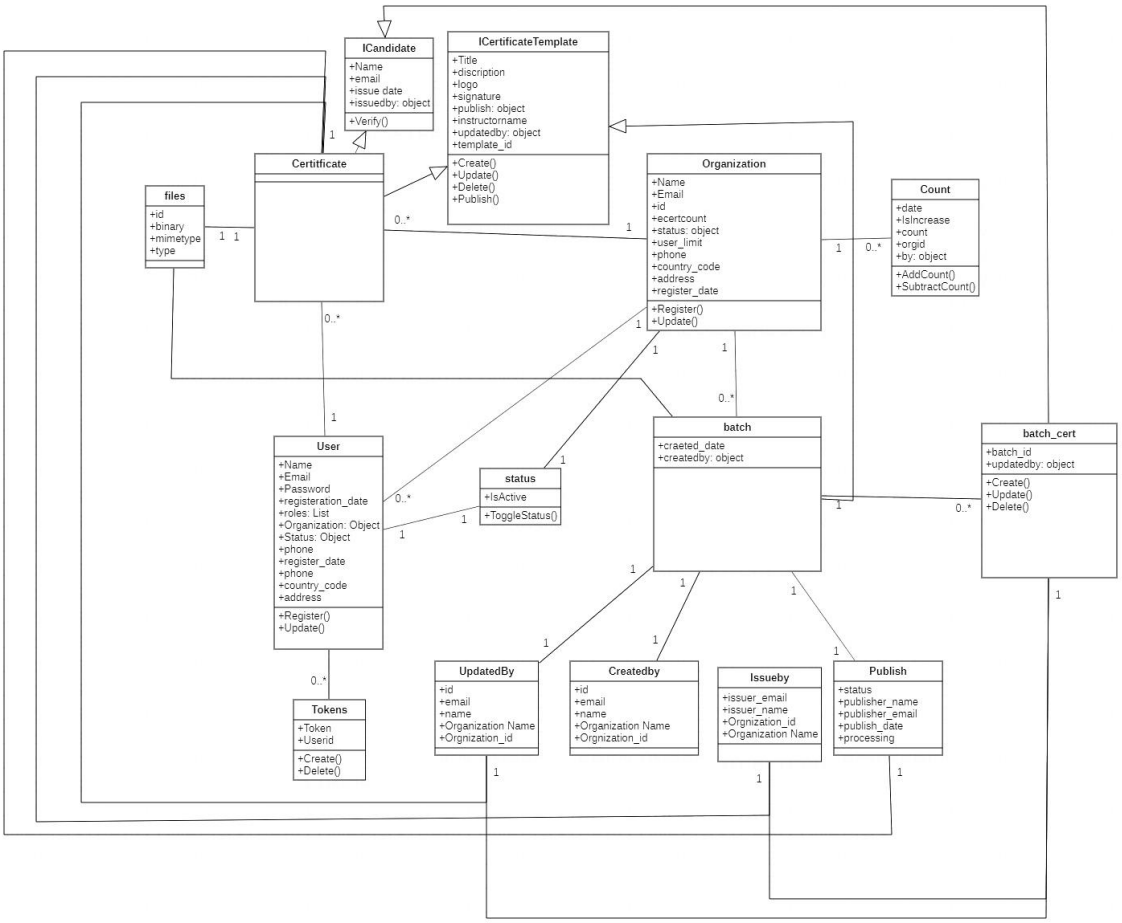


Figure 11.6 - Complete Class Diagram

The figure 11.6 shows the interaction of all classes with each other.

11.2 Annex B: Sequence Diagram

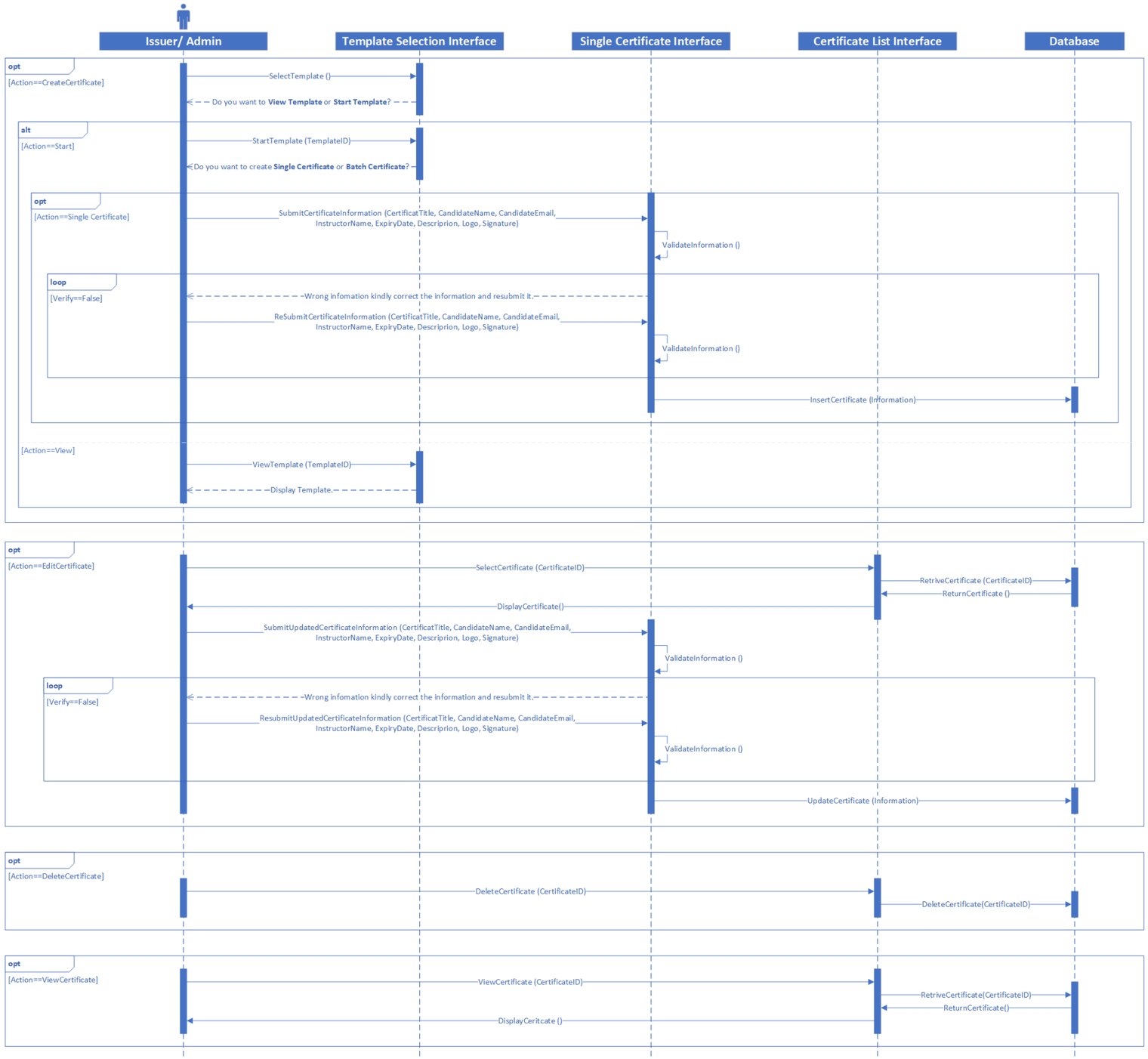


Figure 11.7 - Manage Certificate Sequence Diagram

If the user wants to create a single certificate this will provide him/her the feature of creating single certificates. By choosing a certificate template he/she will be able to create single certificates. The figure 11.7 shows the process of creating single certificates. First user need to go to the create option which is available on the navbar the system will render all the available certificate templates, user will select any of the template then he/she will be able to view or fill as per choice in case if user fills the selected template then system will ask user either they want to create single certificate or they want to creates batch for creating single certificate user needs to selects single certificate then he/she will be allowed to add all the required details of the certificate and when the user clicks on create button then system will validate the formats of all the inputs given by user if the format is correct then the system will generate the certificate else the system will prompt error for correcting the format of the inputted fields. The figure 11.7 also shows the process of edit, delete and view. Now if user want to edit, delete and view generated certificate so he/ she need to select the certificate of his/ her own choice to perform any of the mentioned action. For edit user need to select edit option to update certificate, for delete user need to press the delete button to delete the selected certificate and for view user need to press the view button for viewing the certificate.

Figure 11.8 – Manage Batches/ Batch Certificates Sequence Diagram

If the user wants to create batches or wants to add certificates in the batches this will provide him/her the feature of creating batches and add certificates in the batches. In case user wants to create batch so by choosing a certificate template he/she will be able to create batches and if the user wants to add certificates in them then user can also add certificates in them. The figure 11.8 shows the process of creating batches and their certificates, first user need to go to the create option which is available on the navbar the system will render all the available certificate templates, user will select any of the template then he/she will view or fill as per choice in case if user fill the selected template then system will ask user either they want to create single certificate or they want to creates batch for creating batch user needs to selects batch then he/she will be allowed to add all the required details of the batch and when the user clicks on create button then system will validate the formats of all the inputs given by user if the format is correct then the system will generate the batch else the system will prompt error for correcting the format of the inputted fields. Once the batch has been created the system will redirect user to add certificates page if the user wants to add certificates in the batch then user will be able to add certificates in it if user don't want to add certificates then this will end the process. If the user doesn't want to create batch and just want to add certificates in the batch this will provide him/her the feature of adding certificates. By choosing the batch he/she will be able to add certificates in that batch. The figure 11.8 also shows the process of adding certificates in the batch, first user need to go to the certificate option which is available on the navbar then system will display two options single certificates and batches for adding certificates in the batches user needs to select batches option so system will display all the unpublished batches to the user then user will select any of the batch and will go to batch detail and when the user clicks on add button then the system will render the page to add certificates, now user will be having multiple options of adding certificate one is manual addition and the second is add certificates by file if the user wants to add manually then he/she can add certificates manually and if user wants to add certificates using file then he/she can upload the excel file then the system will retrieve certificates data from the file also if the user wants to add more certificates after uploading the file then he/she can add them manually once the user clicks on submit button then the system will validate the detail of all the certificate if all certificates' detail is correct then the system will add those certificates to the batch if any of those certificates' detail is invalid so system will prompt an error to the user for correcting the detail. The figure 11.8 also shows the process of edit, delete and view of batches and their certificates. For edit user need to select edit option either for batch or for their certificate, for delete user need to select delete option either for batch or for their certificate and for view user need to select view option either for batch or for their certificate.

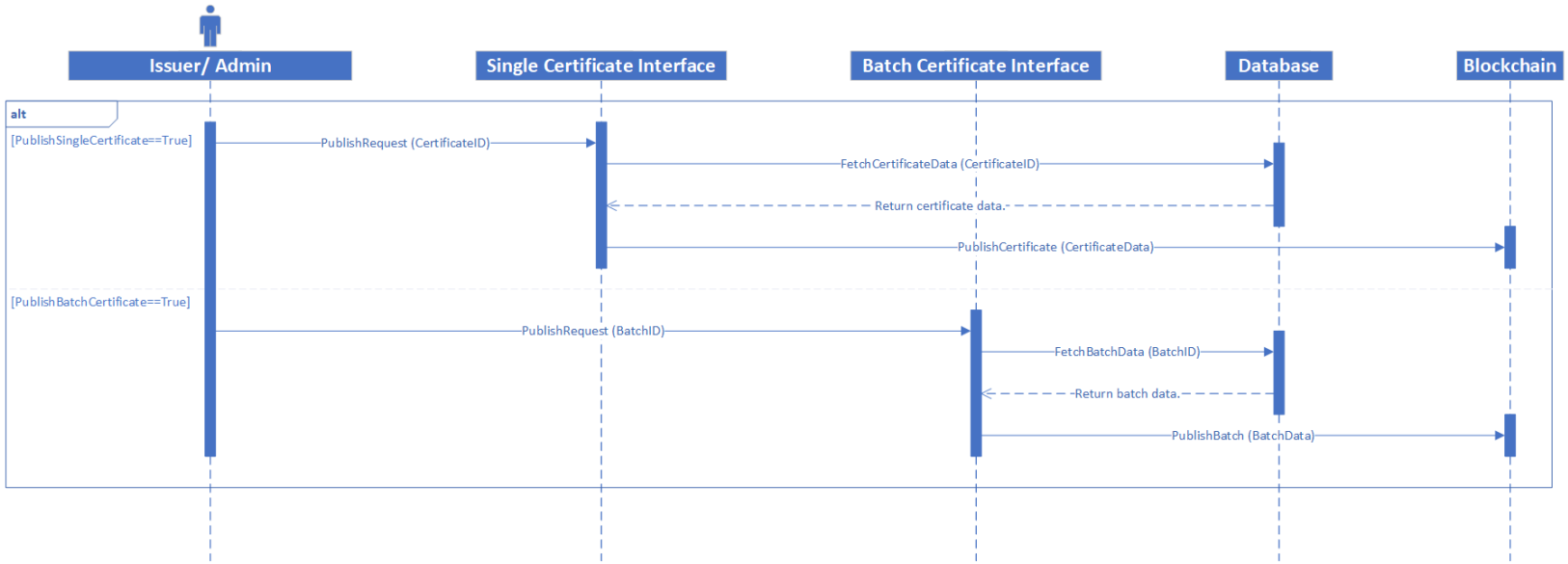


Figure 11.9 - Certificate Publication Sequence Diagram

If the user wants to publish single certificates or batches to the blockchain the this will allow users to publish single certificates or batches to the blockchain after publishing, all the certificates will be verifiable. The figure 11.9 shows the process of publishing single certificates or batches to the blockchain, first user need to go to the certificate option which is available on the navbar then system will display two options single certificates and batches, incase if user selects single certificates option so system will display all the unpublished single certificates to the user then user will select any of the single certificate and click on publish button to publish that single certificate to the blockchain and if user selects batches option so system will display all the unpublished batches to the user then user will select any of the batch and click on publish button to publish that batch to the blockchain.

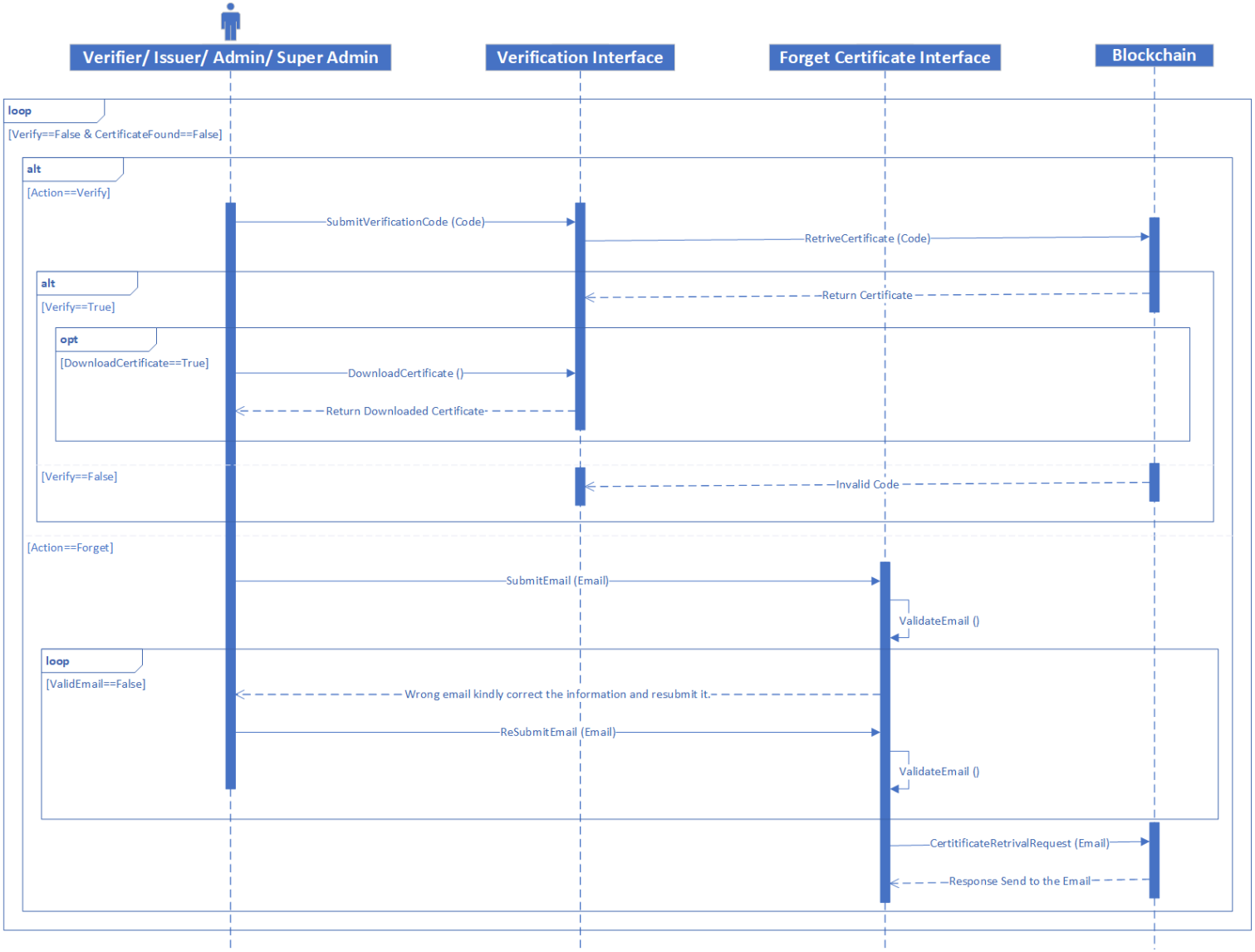


Figure 11.10 - Verify/ Recover Certificates Sequence Diagram

If the user wants to verify certificates this will allow users to verify the certificates that have been published to blockchain. The figure 11.10 shows the process of verifying certificates either in batch or single certificates, first user needs to go to the verify option which is available on the navbar then system will render verification page if the user wants to verify the certificate he/she needs to enter the verification code if the code is valid then certificates will be displayed to the user and if user wants to save the certificates then the user can download that certificate and in case if the code is invalid then the system will prompt error of invalid code to the user also if user forgets the certificate code then user can recover those certificates by clicking on forget certificate which will render the forget certificate page to recover certificates now user needs to enter his/her email address, once user submit the request, the system will validate the email if the email is valid then all the certificates against that email will be sent to that email address.

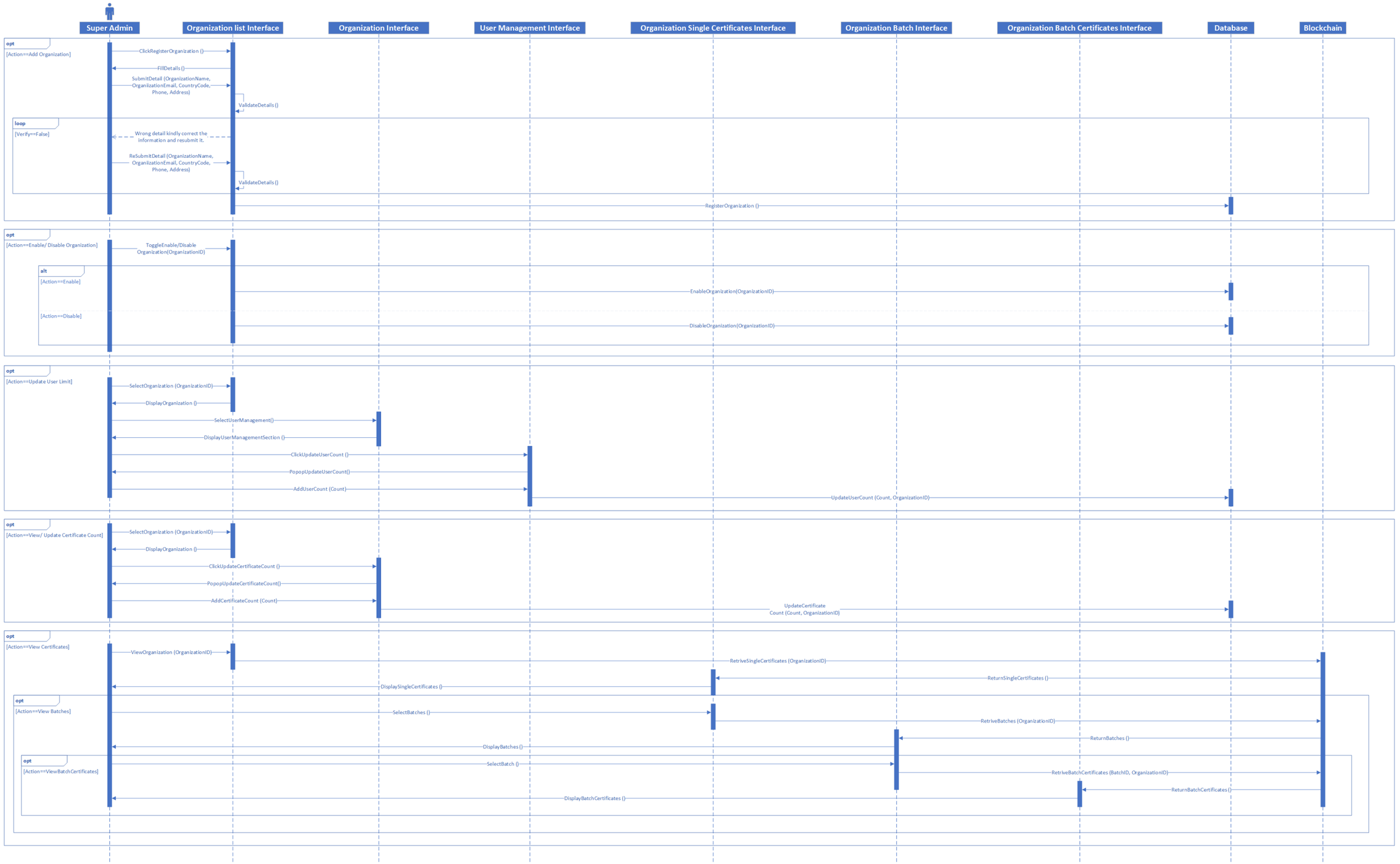


Figure 11.11 - Manage Organization Sequence Diagram

If the user wants to register new organization. So, figure 11.11 shows the process of register organization, first user needs to go to the organization option which is available on the navbar then system will render organization page if the user wants to register organization user need to enter the detail of organization when the user submits the detail of the organization the system will validate the detail of the organization if the detail is valid then the system will register the organization if the detail is invalid then the system will prompt user to correct the detail. The figure 11.11 also shows the process of enable/ disable organization, update user limit, view/ update certificate count, view certificate and view batch. For enable/ disable organization user need to go to the organization list interface and toggle enable/ disable organization switch, for update user limit user need to select organization from the organization list and go to the user management interface then update user limit, for view/ update certificate count user need to select organization from the organization list this will show the old count history and by clicking update count user can update the certificate count, for view certificates and batches user need to view the organization from the organization list this will show all the published certificates of the client organization.

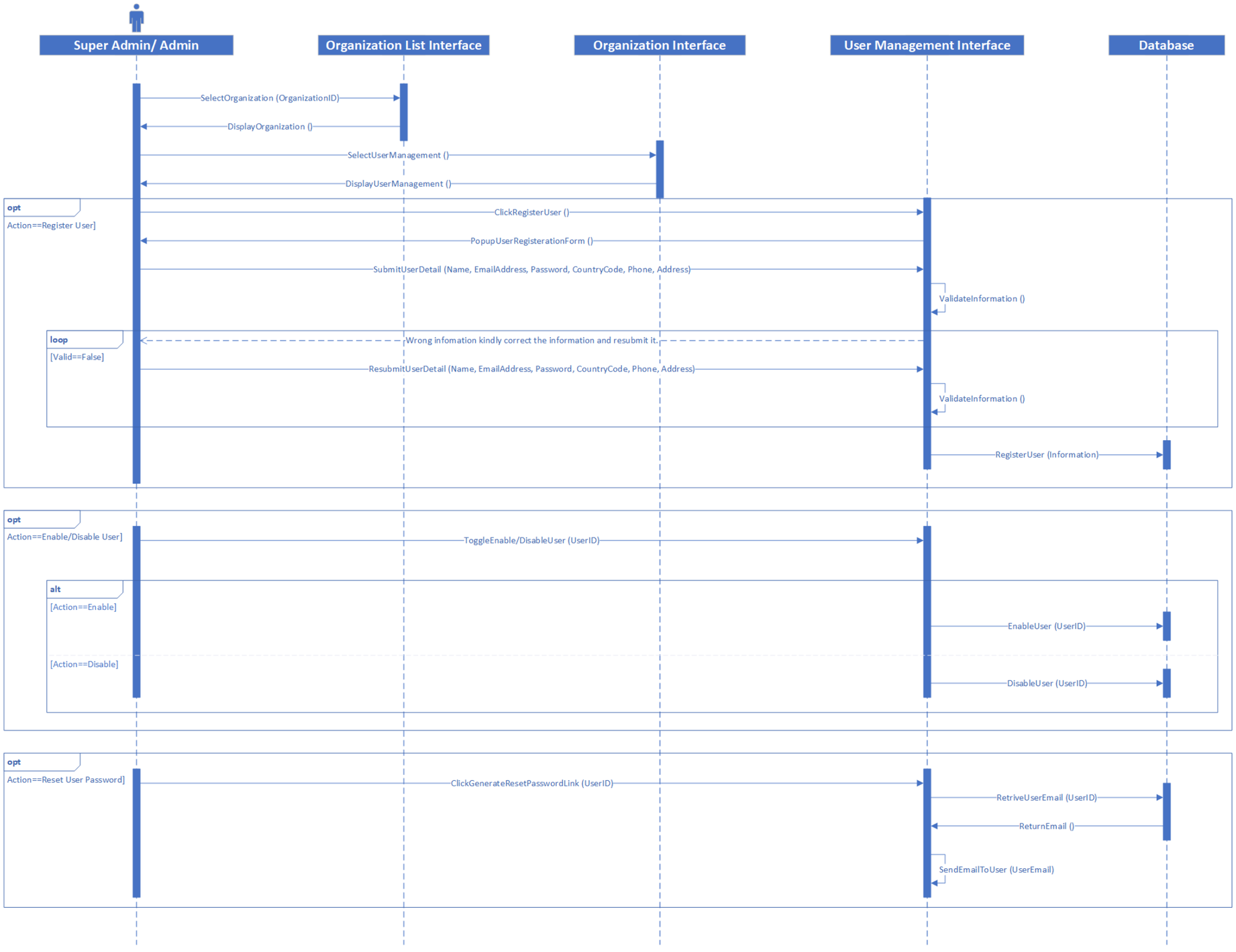


Figure 11.12 - Manage User Sequence Diagram

The Figure 11.12 shows the process of user management that contain the process of registering user, enable/ disable user and reset user password. For registering user, user need to go to the organization in the user management section register user allow to create user and by providing detail user can register user. For enable/ disable user, user need to toggle enable/ disable user switch in the same interface to enable/ disable user and for reset password link generation user need to just click on the reset password button in the same interface.

11.3 Annex C: Object Diagram

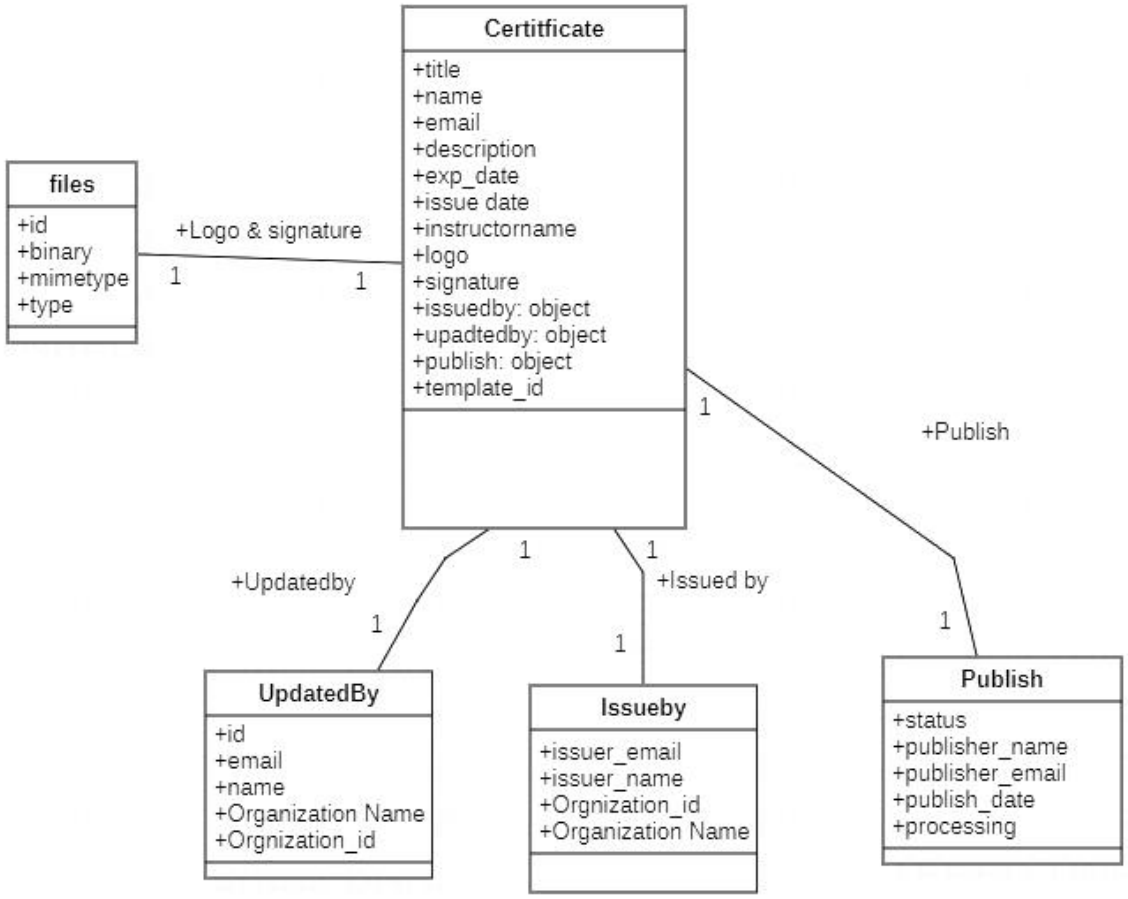


Figure 11.13 - Manage Certificate Object Diagram

The figure 11.13 shows the process of managing certificate also it shows the relationship of certificate object with all other objects.

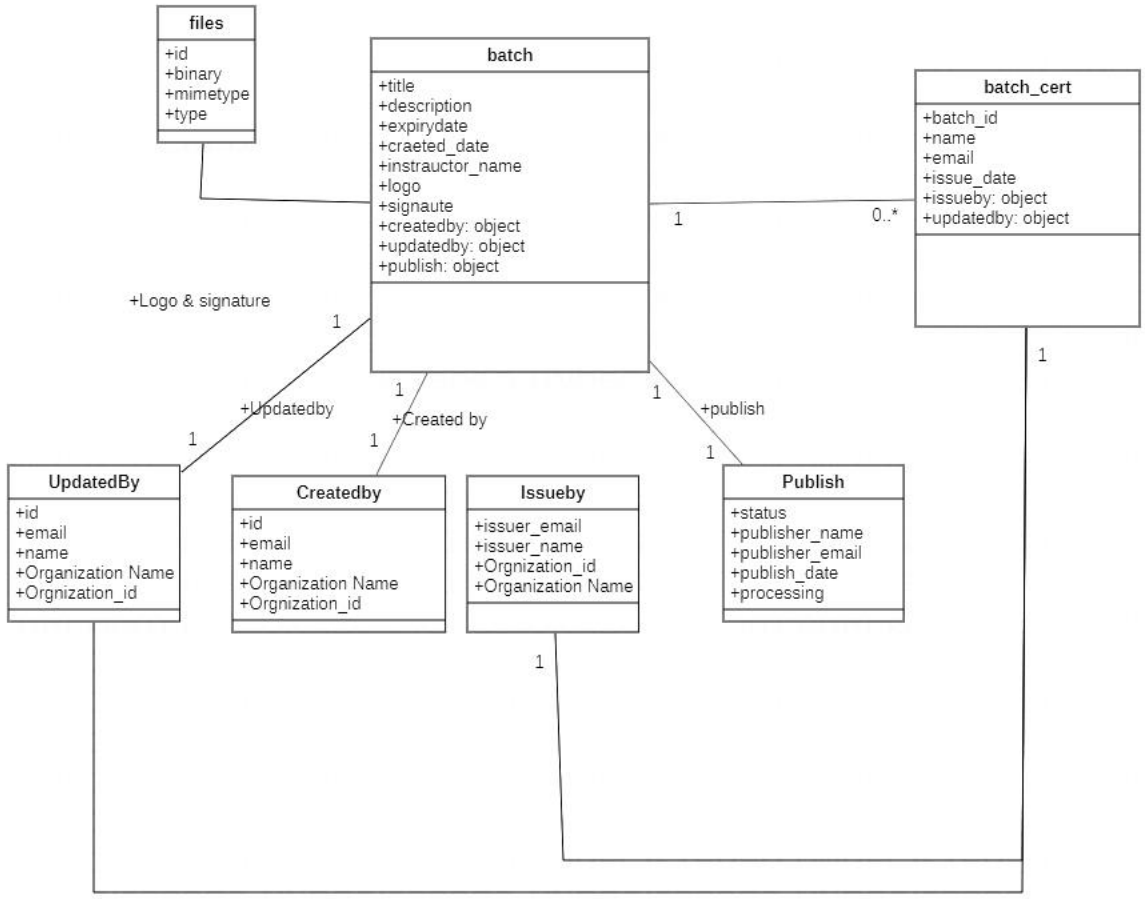


Figure 11.14 - Manage Batch and Batch Certificates Object Diagram

The figure 11.14 shows the process of managing batch and batch certificate also it shows the relationship of batch and batch certificate object with all other objects.

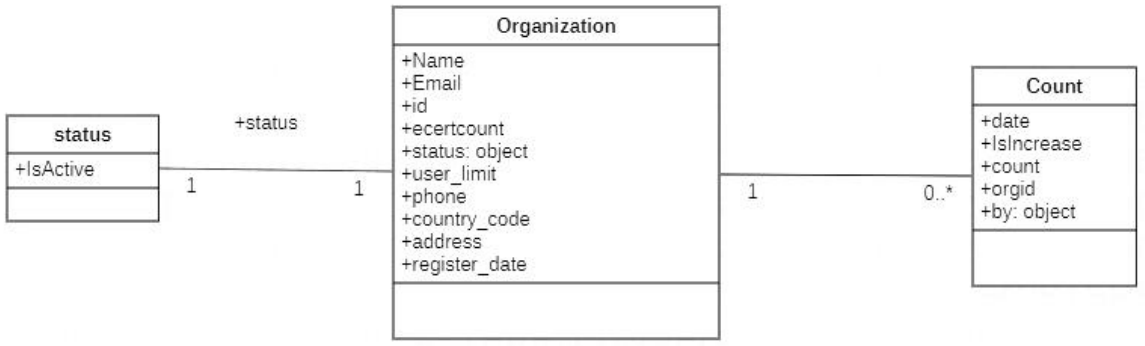


Figure 11.15 - Manage Organization Object Diagram

The figure 11.15 shows the process of managing organization also it shows the relationship of organization object with all other objects.

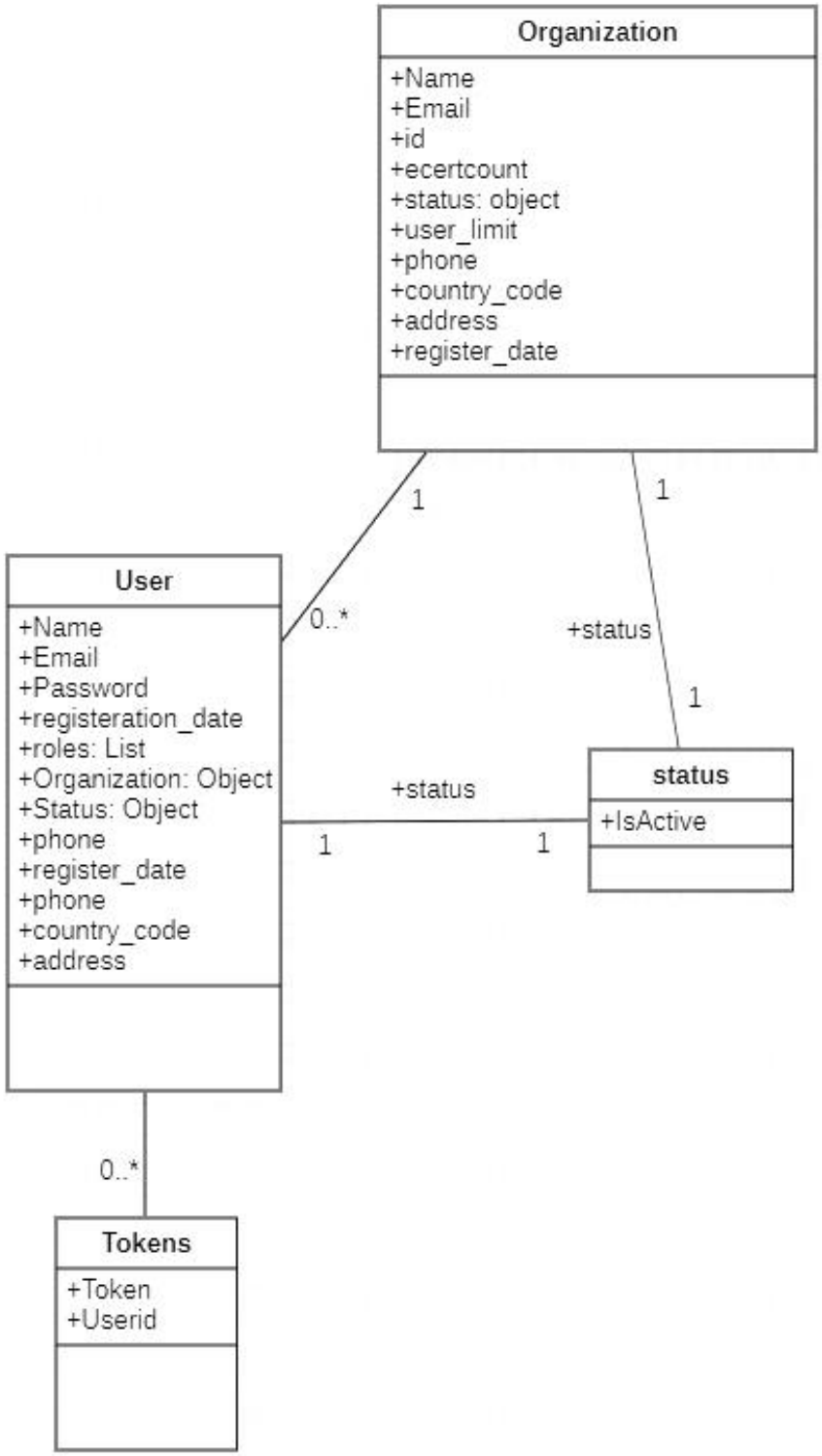


Figure 11.16 - Manage User Object Diagram

The figure 11.16 shows the process of managing user also it shows the relationship of user object with all other objects.

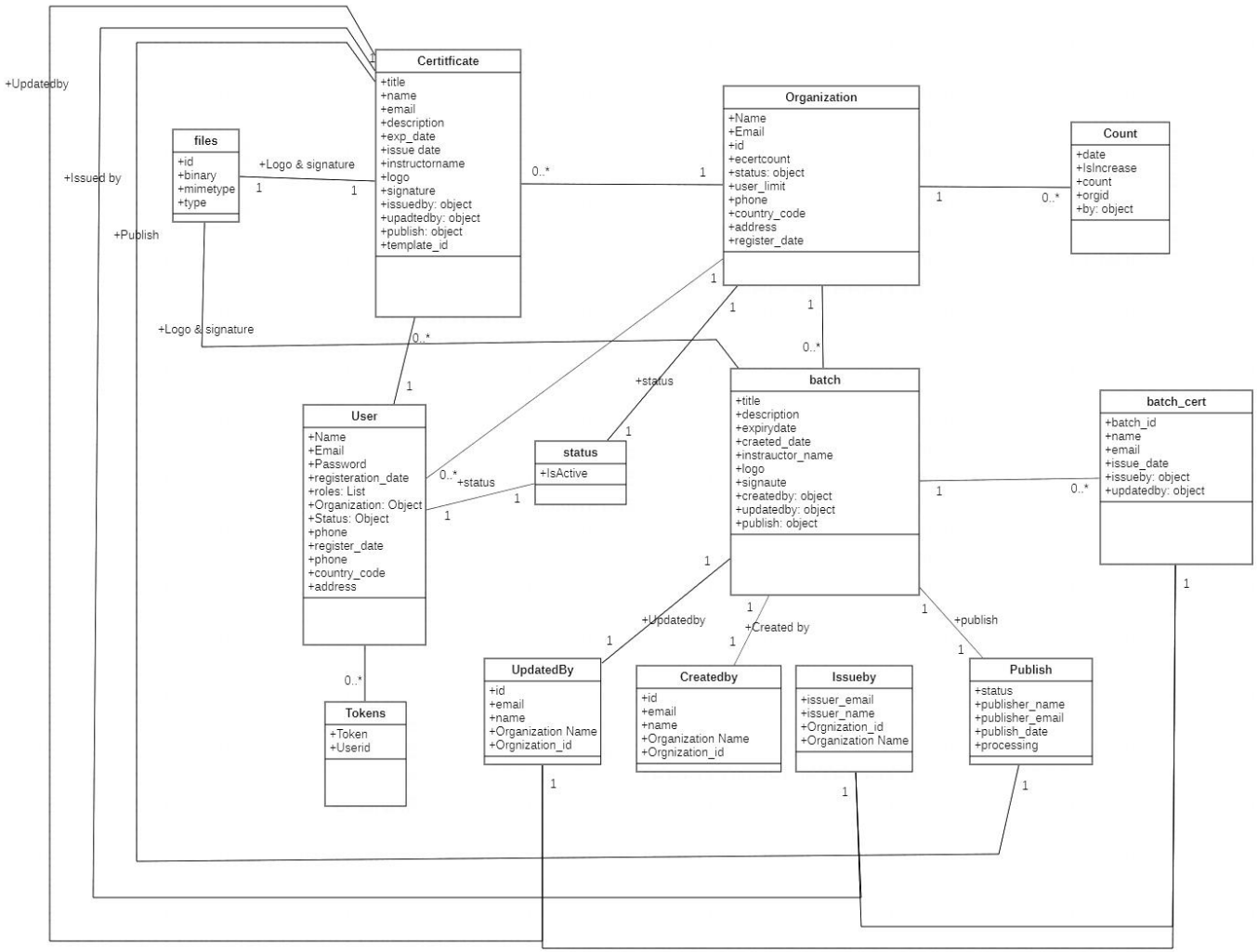


Figure 11.17 - Complete Object Diagram

The figure 11.17 shows the interaction of all objects with each other.

12. References

[1] Certification Importance, www.aarm.org/certification_importance.html#:~:text=Certification%20has%20found%20its%20way,enhance%20their%20skills%20and%20knowledge.

[2] "What Are Distributed Ledger Technologies?" Hedera Hashgraph, hedera.com/learning/what-are-distributed-ledger-technologies-dlts?gclid=CjwKCAiAnvj9BRA4EiwAuUMDf1GdDYHz0AhHsnzidNKPWu_unJcQyKhr5vaUHNdlLwFHQDkMVdf4QBoCel0QAvD_BwE.

[3] "Introduction." Hyperledger, hyperledger-fabric.readthedocs.io/en/release-2.2/blockchain.html.

[4] "Hyperledger Fabric Model." Hyperledger, hyperledger-fabric.readthedocs.io/en/release-2.2/fabric_model.html.

[5] Conway, Luke. "Blockchain Explained." Investopedia, Investopedia, 18 Nov. 2020, www.investopedia.com/terms/b/blockchain.asp.

[6] "Ethereum." Wikipedia, Wikimedia Foundation, 6 Jan. 2021, en.wikipedia.org/wiki/Ethereum.

[7] "Hyperledger." Wikipedia, Wikimedia Foundation, 8 Dec. 2020, en.wikipedia.org/wiki/Hyperledger.

[8] "Enterprise Blockchain Platform: Corda Enterprise by R3." R3, www.r3.com/corda-enterprise/.

[9] "Ripple (Payment Protocol)." Wikipedia, Wikimedia Foundation, 9 Jan. 2021, en.wikipedia.org/wiki/Ripple_(payment_protocol).

[10] Author , and Blockgeeks. "What Is Quorum Blockchain? A Platform for The Enterprise." Blockgeeks, 14 May 2020, blockgeeks.com/guides/quorum-a-blockchain-platform-for-the-enterprise/.

[11] Frankenfield, Jake. "Hyperledger Sawtooth Definition." Investopedia, Investopedia, 16 Sept. 2020, www.investopedia.com/terms/h/hyperledger-sawtooth.asp.

[12] "Blockchain Software Architecture." EOSIO, 15 Dec. 2020, eos.io/.

[13] "Hyperledger Iroha." Hyperledger, 27 Apr. 2020, www.hyperledger.org/use/iroha.

[14] "Stellar (Payment Network)." Wikipedia, Wikimedia Foundation, 7 Jan. 2021, en.wikipedia.org/wiki/Stellar_(payment_network).

[15] "Distributed Architecture." *Tutorialspoint*, www.tutorialspoint.com/software_architecture_design/distributed_architecture.htm.