The Superior College Lahore

Faculty of Computer Science & IT

Final Year Project

PROJECT REPORT

IDENTITY MANAGEMENT SYSTEM

Project ID: **FYP-BSCS-F18-012**

Project Team

Student Name	Student ID	Program	Contact Number	Email Address
Ali Hamza	**BCSM-F15-106**	BSCS	03224240761	alihamza2313@gmail.com
Hamza	**BCSM-F15-120**	BSCS	03454778071	hamzaarifigi7@gmail.com
Babar Sohail	**BCSM-F15-113**	BSCS	03415721181	babarsohail04@gmail.com

Mr. Muhammad Ahmed
Lecturer

Project Report

Identity Management System

Change Record

Author(s)	Version	Date	Notes	Supervisor's Signature
	1.0			

APPROVAL

PROJECT SUPERVISOR

Comments: _____

_____

Name:_____

Date:_____ Signature:_____

PROJECT MANAGER

Comments: _____

_____

Date:_____ Signature:_____

HEAD OF THE DEPARTMENT

Comments: _____

_____

Date:_____ Signature:_____

Dedication

We dedicate our final year project to our respectful teachers who supported our project idea, gave us proper guidance regarding our project and encouraged us in every step we took forward in completion of the project. Without their cooperation it would have not been possible to complete our graduation. We also dedicate Final Year Project to our friends who have supported us and taught us little things that helped us a lot towards our progress. We will always appreciate all of them and remember them in our good books.

Acknowledgements

I would like to express my special thanks of gratitude to our supervisor Mr. Muhammad Ahmed as well as head of FYP Mr. Dr.Shehryar Malik who gave us the golden opportunity to do this wonderful project 'Identity Management System which also helped us in doing a lot of Research and we came to know about so many new things we are really thankful to them.

Executive Summary

Our project is to build a tempered-proof Identity Management System (IMS) like NADRA on the stack of new emerging Block chain Technology which use cryptographic techniques and tools in order to provide Private and Permission system which delivers a high degree of confidentiality, provisioning ,resiliency (the capacity to recover quickly from difficulties), flexibility, and scalability (the capacity to be changed in size or scale).IMS used to issue unique identities to the citizens and their families so that they can be identified uniquely and authenticated in a large population and government can keep track of them with their family relations for the welfare of the country and for different analyzing purposes. We can restrict users from gaining unauthorized access to the government organization data by specifying their access control list in Block chain, so an unauthorized person could not get access illegally to government assets.

Table of Contents

Dedication iv
Acknowledgements v
Executive Summary vi
Table of Contents vii
List of Figures x
List of Tables xi
Chapter 1 1
Introduction 1
1.1. Background 2
1.2. Motivations and Challenges 2
1.3. Goals and Objectives 3
1.4. Literature Review/Existing Solutions 4
1.5. Gap Analysis 4
1.6. Proposed Solution 4
1.7. Project Plan 4
1.7.1. Work Breakdown Structure 5
1.7.2. Roles & Responsibility Matrix 6
1.7.3. Gantt Chart 7
1.8. Report Outline 8
Chapter 2 10
Software Requirement Specifications 10
2.1. Introduction 11
2.1.1. Purpose 11
2.1.2. Document Conventions 12
2.1.3. Intended Audience and Reading Suggestions 12
2.1.4. Product Scope 12
2.1.5. References 12
2.2. Overall Description 12
2.2.1. Product Perspective 12
2.2.2. Product Functions 13
2.2.3. User Classes and Characteristics 16
2.2.4. Operating Environment 16
2.2.5. Design and Implementation Constraints 17
2.2.6. User Documentation 17
2.2.7. Assumptions and Dependencies 17
2.3. External Interface Requirements 17
2.3.1. User Interfaces 17
Users will able to login, view their record, could see their task. 17
2.3.2. Hardware Interfaces 17
2.3.3. Software Interfaces 18
2.3.4. Communications Interfaces 18
2.4. System Features 18
2.4.1. System Feature 1 18
2.4.1.1 Description and Priority 18

2.4.1.2 Stimulus/Response Sequences 18
Functional Requirements 19
2.4.2. System Feature 2 19
2.4.2.1. Description and Priority 19
2.4.2.2. Stimulus/Response Sequences 19
2.4.2.3. Functional Requirements 20
Click "create Birth Certificate" Button: Create certificate 20
2.3.4 System Feature 4 20
2.3.4.1 Description and Priority 20
Stimulus/Response Sequences 20
2.3.5 System Feature 5 20
2.3.5.1 Description and Priority 20
Stimulus/Response Sequences 20
Functional Requirements 21
- Click " deploy chain-code" Button: deploy the chain-code 21
2.3.6 System Feature 6 21
2.3.6.1 Description and Priority 21
Stimulus/Response Sequences 21
Functional Requirements 21
2.5. Other Nonfunctional Requirements 22
2.5.1. Performance Requirements 22
2.5.2. Safety Requirements 22
2.5.3. Security Requirements 22
2.5.4. Software Quality Attributes 22
2.6. Other Requirements 22
Chapter 3 23
Use Case Analysis 23
3.1. Use Case Model 24
3.2. Fully Dressed Use Cases 29
Chapter 4 43
System Design 43
4.1. Architecture Diagram 44
4.2. Domain Model 45
4.3. Entity Relationship Diagram with data dictionary 46
4.4. Class Diagram 47
4.5. Sequence / Collaboration Diagram 48
Sequence Diagrams 48
4.6. Operation contracts 56
4.7. Activity Diagram 58
4.8. State Transition Diagram 59
4.9. Component Diagram 64
4.10. Deployment Diagram 65
4.11. Data Flow diagram 66
Chapter 5 68
Implementation 68
5.1. Important Flow Control/Pseudo codes 69

5.2. Components, Libraries, Web Services and stubs 69
5.3. Deployment Environment 69
5.4. Tools and Techniques 70
5.5. Best Practices / Coding Standards 70
5.6. Version Control 71
Chapter 6 72
Testing and Evaluation 72
6.1. Use Case Testing 73
6.2. Equivalence partitioning 81
6.3. Boundary value analysis 81
6.4. Data flow testing 82
6.5. Unit testing 83
6.6. Integration testing 83
6.7. Performance testing 84
6.8. Stress Testing 85
Chapter 7 87
Summary, Conclusion and Future Enhancements 87
7.1. Project Summary 88
7.2. Achievements and Improvements 88
7.3. Critical Review 88
7.4. Lessons Learnt 88
7.5. Future Enhancements/Recommendations 89
Reference and Bibliography 90
BS in Computer Science 92
Plagiarism Free Certificate 93

List of Figures

Figure 1-Ghant Chat 8
Figure 2-Use Case 24
Figure 3-Manage Asset 25
Figure 4-User Activities 25
Figure 5-Authentication 26
Figure 6-Manage Explorer 27
Figure 7-Blockchain 28
Figure 8-Architecture Diagram 44
Figure 9-Domain Model 45
Figure 10-ERD 46
Figure 11-Class Diagram 47
Figure 12-SD login 48
Figure 13-SD Admin Login 49
Figure 14-SD Role 50
Figure 15-SD Manage User 51
Figure 16-SD Manage Explorer 52
Figure 17-SD Manage Report 53
Figure 18-SD Manage BlockChain 54
Figure 19-SD Manage Identity Asset 55
Figure 20-SD Activity Diagram 58
Figure 21-STD STD View details 59
Figure 22-STD Auth 59
Figure 23-STD Update info 60
Figure 24-STD Manage Account 60
Figure 25-STD Manage report 61
Figure 26-STD Manage Explorer 61
Figure 27-STD Manage blockchain 62
Figure 28-STD Manage Admin 62
Figure 29-STD Manage Request 63
Figure 30-Component Diagram 64
Figure 31-Deployment Diagram 65
Figure 32-DFD 66
Figure 33-DFD level 1 67

List of Tables

Table 1-project plan 4
Table 2-Full Dress Use Case Authentication 29
Table 3-Full Dress Use Case Login 30
Table 4-Full Dress Use Case Logout 31
Table 5-Full Dress Use Case Manage Explorer 32
Table 6-Full Dress Use Case View Explorer 33
Table 7-Full Dress Use Case Manage Blockchain 34
Table 8-FDUC deploy chain code 35
Table 9-FDUC Update chain code 36
Table 10-FDUC Total chain code 37
Table 11-FDUC Approve transaction 38
Table 12-FDUC View Recent Transaction 39
Table 13-FDUC Request for New Certificate 40
Table 14-FDUC Request for View Certificate 41
Table 15-FDUC Request for Update Certificate 42
Table 16-Version Control 71
Table 17-Equivalence partitioning 81
Table 18-Boundary value analysis 82

Chapter 1

Introduction

1. Introduction

This will purpose a solution to organizations which stores the users profiles data in order to provide them the facility to issue them a unique identity which is required for legal purposes. System will provide mechanism to the organizations to store their users profiles data in an immutable digital ledger which cannot be tempered so the fear of using conventional databases of organization will be gone and identity theft will be impossible in a such a system.

1.1. Background

After the independence of Pakistan, Prime Minister Liaquat Ali Khan launched the Personal Identity System (PIS) program to register, manage and issue national identification cards to all the citizens of Pakistan and Muslim refugees settling in Pakistan.

After the 1971 war resulted in East-Pakistan gaining independence as Bangladesh, a new statistical database system was needed to ensure the safety of Pakistan's citizens as well as the national security of the country. In 1973, a new database system was codified under the Second Amendment, of the Constitution of Pakistan to perform and contain the statistical database of the citizens of Pakistan. Registration of Pakistan's citizens and statistic database in government's computer accounts was started in 1973, with the promulgation of the constitution of the country. This new program was visioned and started by then-Prime Minister Zulfikar Ali Bhutto.

In 1973, in a parliamentary session, Bhutto stated in parliament to the people of Pakistan, "due to the absence of full statistical database of the people of this country, this country is operating in utter darkness".The government started issuing the National Identity Card (NIC) numbers to its citizens and started to establish government databases of the people in the government computers.

1.2. Motivations and Challenges

Motivation for this project is that Blockchain is a technology have features like immutability and distributiveness .In other countries like Estonia has developed digital identity system based on

blockchain. In Pakistan, Blockchain technology scope is limited and needs serious market attention in this specific area. Existing system NADRA does not ensure the blockchain features in their system. So we are developing blockchain based identity management system which will overcome the issues we are facing in current systems. Society has looked for solutions to the problem of identifying individuals for hundreds of years, using available technologies to meet the need. As modern information technology offers the ability to collect and link vast amounts of data from multiple sources, to communicate data over great distance, to store and retrieve data from anywhere around the globe and by third parties collect your data form the network and use it for criminal's activities in simple word the Hacker asses your data.

1.3. Goals and Objectives

Main Goal to build a hacking prove identity and assess management system (IMS) using blockchain.

- **Satisfy the customer:**

Our main purpose is to satisfy the customers & fulfill their needs.

- **Web base system**

Our system is web based because the web is very commune used platform.

- **Decentralize data base**

The all data of the system is decentralized for backup of the data and increase the respond efficient of the system .

- **Asses control**

In this system we use the new blockchain technology the asses of system is very mush define and clear and any unauthorized person change the data the system generate the notification too all admin.

1.4. Literature Review/Existing Solutions

- People with the name NADRA are working on it, idea is bit similar, and they are working on web base identity management system.
- We are working on the new blockchain technology the all data of server is incepted and it is based on cryptography technique and asses of the system is very much and the unauthorized person can't asses the data of the customer and we will make the chain of the data. We take example of a tree which has parents and child's algorithm.
- We will make website in which we put the data identity no. the system will generate a report related to the this identity no. of a person

1.5. Gap Analysis

Currently targeting Lahore only at first, but we are planning to expand its operation in other major cities of Pakistan.

- **Planned Strategy:**

  Our main purpose is to achieve the identity secure for citizen.

1.6. Proposed Solution

Solution of the problems:

We will provide a solution for Expensiveness, Vulnerability, inefficiency, Identity Theft and a single Point of Failure by developing Identity Management System on stack of Blockchain Technology

1.7. Project Plan

	Description of Work	**Start and End Dates**
Phase One	Documentation	5 Months
Phase Two	Website	6 Months

Table 1-project plan

1.7.1. Work Breakdown Structure

National Identity System

1 Planning

1.1 Identify stakeholders

1.2 Identify Problem

2 Requirements Gathering and Analysis

2.1 Gather Requirements

2.1.1 Study Of Existing System

2.1.2 Observation

2.1.3 Brain Storming

2.2 Analyze Requirements

2.2.1 Identify Actors

2.2.2 Identify Use Cases

2.2.3 Identify Business Rules

2.2.4 Identify Business Requirements

2.2.5 Identify Functional Requirements

2.2.6 Identify Non-Functional Requirements

2.3 Document Requirements

2.3.1 SRS Document

2.4 get Approval

3 Design

3.1 Logical Design

3.1.1 Class Diagrams

3.1.2 UML diagrams

3.1.3 Flow Diagrams

3.2 Get Approval

4 Implementation

4.1 Installation

4.1.1 Setup Environment

4.1.2 Install Dependencies

4.2 Programming

4.2.1 Source Code

4.2.1.1 Prototyping

4.2.1.2 Construct Code

4.2.1.3 Unit Testing

4.2.1.4 Use Git

4.2.1.5 Code Integration

4.3 Operation

4.3.1 Operate System

4.3.2 Review

4.3.2.1 Review Performance

4.3.3 Get Approval

5 Testing

5.1 Unit Testing

5.2 Integration Testing

5.3 System Testing

6 Deployment

6.1 Deploy

6.2 User Documentation

1.7.2. Roles & Responsibility Matrix

- **Documentation :**

Ali Hamza
Hamza
Babar sohail

- **Website Front End Designing & Back end coding:**

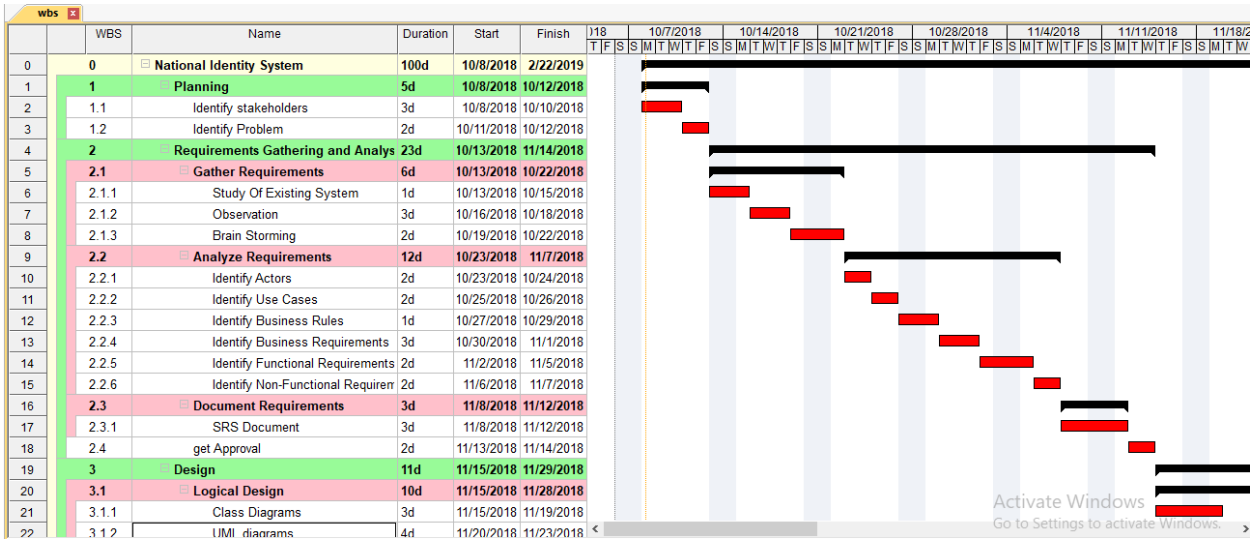
Ali Hamza
Hamza

Babar sohail

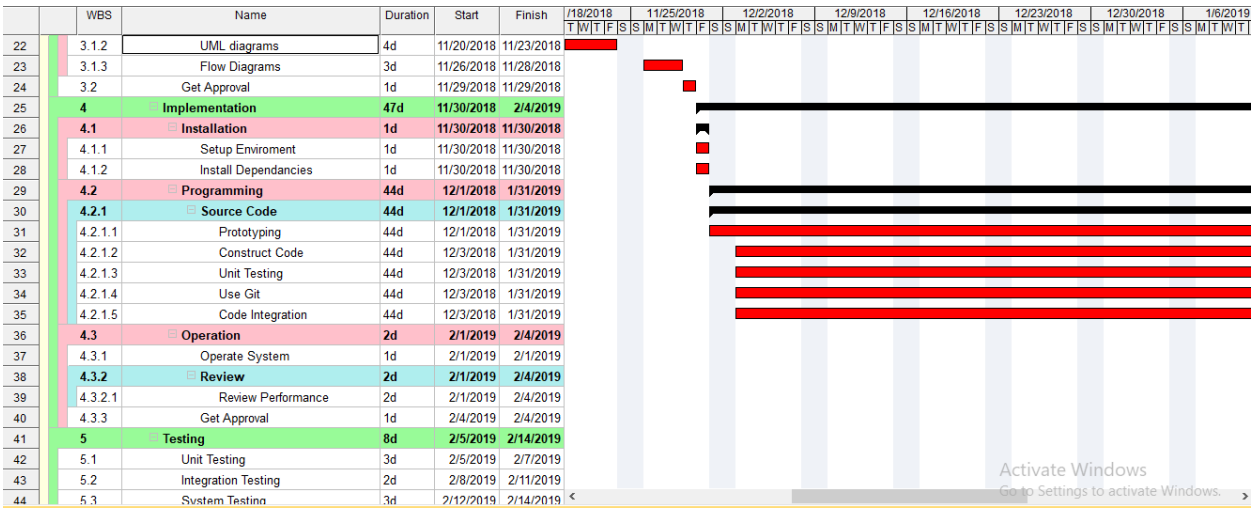
- **Web Database:**

Ali Hamza
Hamza
Babar Sohail

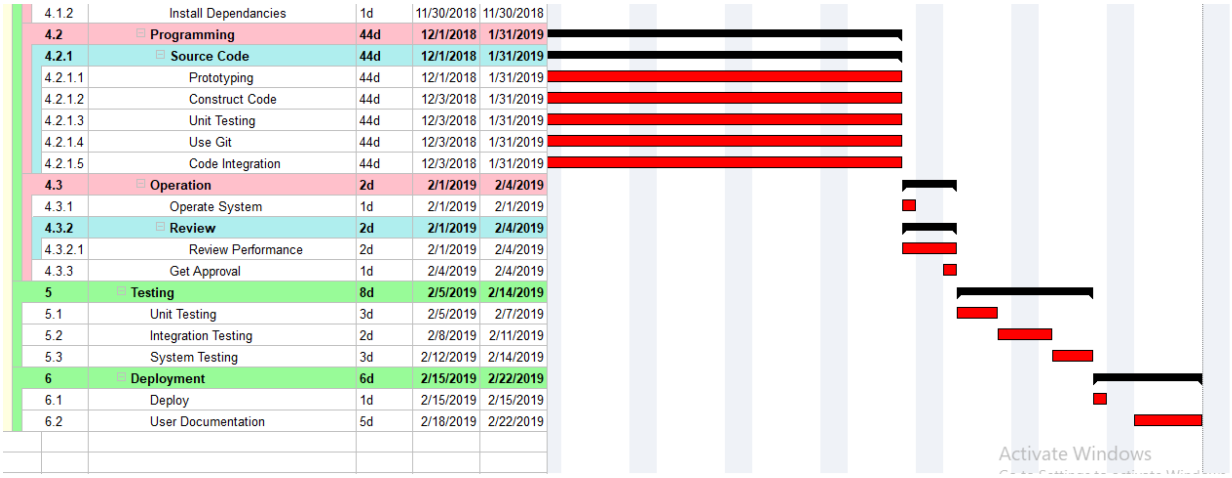
1.7.3. Gantt Chart



	WBS	Name	Duration	Start	Finish
0	0	National Identity System	100d	10/8/2018	2/22/2019
1	1	Planning	5d	10/8/2018	10/12/2018
2	1.1	Identify stakeholders	3d	10/8/2018	10/10/2018
3	1.2	Identify Problem	2d	10/11/2018	10/12/2018
4	2	Requirements Gathering and Analys	23d	10/13/2018	11/14/2018
5	2.1	Gather Requirements	6d	10/13/2018	10/22/2018
6	2.1.1	Study Of Existing System	1d	10/13/2018	10/15/2018
7	2.1.2	Observation	3d	10/16/2018	10/18/2018
8	2.1.3	Brain Storming	2d	10/19/2018	10/22/2018
9	2.2	Analyze Requirements	12d	10/23/2018	11/7/2018
10	2.2.1	Identify Actors	2d	10/23/2018	10/24/2018
11	2.2.2	Identify Use Cases	2d	10/25/2018	10/26/2018
12	2.2.3	Identify Business Rules	1d	10/27/2018	10/29/2018
13	2.2.4	Identify Business Requirements	3d	10/30/2018	11/1/2018
14	2.2.5	Identify Functional Requirements	2d	11/2/2018	11/5/2018
15	2.2.6	Identify Non-Functional Requirem	2d	11/6/2018	11/7/2018
16	2.3	Document Requirements	3d	11/8/2018	11/12/2018
17	2.3.1	SRS Document	3d	11/8/2018	11/12/2018
18	2.4	get Approval	2d	11/13/2018	11/14/2018
19	3	Design	11d	11/15/2018	11/29/2018
20	3.1	Logical Design	10d	11/15/2018	11/28/2018
21	3.1.1	Class Diagrams	3d	11/15/2018	11/19/2018
22	3.1.2	UML diagrams	4d	11/20/2018	11/23/2018



	WBS	Name	Duration	Start	Finish
22	3.1.2	UML diagrams	4d	11/20/2018	11/23/2018
23	3.1.3	Flow Diagrams	3d	11/26/2018	11/28/2018
24	3.2	Get Approval	1d	11/29/2018	11/29/2018
25	4	Implementation	47d	11/30/2018	2/4/2019
26	4.1	Installation	1d	11/30/2018	11/30/2018
27	4.1.1	Setup Enviroment	1d	11/30/2018	11/30/2018
28	4.1.2	Install Dependancies	1d	11/30/2018	11/30/2018
29	4.2	Programming	44d	12/1/2018	1/31/2019
30	4.2.1	Source Code	44d	12/1/2018	1/31/2019
31	4.2.1.1	Prototyping	44d	12/1/2018	1/31/2019
32	4.2.1.2	Construct Code	44d	12/3/2018	1/31/2019
33	4.2.1.3	Unit Testing	44d	12/3/2018	1/31/2019
34	4.2.1.4	Use Git	44d	12/3/2018	1/31/2019
35	4.2.1.5	Code Integration	44d	12/3/2018	1/31/2019
36	4.3	Operation	2d	2/1/2019	2/4/2019
37	4.3.1	Operate System	1d	2/1/2019	2/1/2019
38	4.3.2	Review	2d	2/1/2019	2/4/2019
39	4.3.2.1	Review Performance	2d	2/1/2019	2/4/2019
40	4.3.3	Get Approval	1d	2/4/2019	2/4/2019
41	5	Testing	8d	2/5/2019	2/14/2019
42	5.1	Unit Testing	3d	2/5/2019	2/7/2019
43	5.2	Integration Testing	2d	2/8/2019	2/11/2019
44	5.3	System Testing	3d	2/12/2019	2/14/2019



4.1.2	Install Dependancies	1d	11/30/2018	11/30/2018
4.2	**Programming**	**44d**	**12/1/2018**	**1/31/2019**
4.2.1	**Source Code**	**44d**	**12/1/2018**	**1/31/2019**
4.2.1.1	Prototyping	44d	12/1/2018	1/31/2019
4.2.1.2	Construct Code	44d	12/3/2018	1/31/2019
4.2.1.3	Unit Testing	44d	12/3/2018	1/31/2019
4.2.1.4	Use Git	44d	12/3/2018	1/31/2019
4.2.1.5	Code Integration	44d	12/3/2018	1/31/2019
4.3	**Operation**	**2d**	**2/1/2019**	**2/4/2019**
4.3.1	Operate System	1d	2/1/2019	2/1/2019
4.3.2	**Review**	**2d**	**2/1/2019**	**2/4/2019**
4.3.2.1	Review Performance	2d	2/1/2019	2/4/2019
4.3.3	Get Approval	1d	2/4/2019	2/4/2019
5	**Testing**	**8d**	**2/5/2019**	**2/14/2019**
5.1	Unit Testing	3d	2/5/2019	2/7/2019
5.2	Integration Testing	2d	2/8/2019	2/11/2019
5.3	System Testing	3d	2/12/2019	2/14/2019
6	**Deployment**	**6d**	**2/15/2019**	**2/22/2019**
6.1	Deploy	1d	2/15/2019	2/15/2019
6.2	User Documentation	5d	2/18/2019	2/22/2019

Figure 1-Ghant Chat

1.8. Report Outline

- **Introduction**
    - What we are going to do?
- **Background**
    - What currently happening in the market
- **Motivation & Challenges**
    - Discussed in detail
- **Goals and Objectives**
    - Discussed how to achieve our goals.
- **Existing System**
    - Discussed what unique we are having, how could survive in a market
- **Gap Analysis**
    - Discussed Gap Analysis & Discussed what is our planned strategy?
- **Proposed Solution**
    - Discussed the solutions of the problems
- **Project Plan**
    - Made a project plan so we could complete our project in a time.
- **Work Breakdown Structure**

- Discussed in detail

- **Rules & Responsibility**
  - Divided task in team members
- **Gantt Chart**
  - Made a chart so could complete our project in a time.

Chapter 2

Software Requirement Specifications

2: Software Requirement Specifications

2.1. Introduction

Identity Management System (IMS) is a system like existing NADRAin which we will issue a unique identity to the citizens of Pakistan. This system is responsible for storing data of a human entity and their relationship with the other entities.

Blockchain is a distributed ledger, it can be defined as a ledger that is public. This ledger can be maintained by other nodes on the network. Blockchain is a secure system as compared to the existing system of NADRAas in blockchain one transaction is validated by all nodes in the system. Data in blockchain's ledger cannot be further modified by a node. To crack the entire network, one need to crack more than 50% of the entire network and also need to do the rework for it. So, it is highly difficult and almost impossible to crack a network. In case a node illegally changes data. In such case as data is verified by other nodes on the network, it will be pointed out and identified. So, blockchain system will not except the change. The entire block will be rejected. Each node has a copy of database. So, before updating data it is confirm and acknowledged by all other nodes on the network. It also consists of cryptography hash with makes it highly securable and helps in tracking the chains of block.

2.1.1. Purpose

The introduction of the Software Requirements Specification (SRS) provides an overview of the entire SRS with purpose, scope, references and overview of the SRS. The aim of this document is to defining the problem statement in detail.

We are making a website with a name IMS.pk. Our idea is basically to make a website that can secure the identity no of a person and all information against this identity no and control the asses of a system and the data of the server is totally encrypted by cryptographic technique so that any third-party or Hacker can't read the data.

2.1.2. Document Conventions

This document uses the following conventions.

DB: Hyperlager_fabrice

DB: Mongo DB

2.1.3. Intended Audience and Reading Suggestions

We are providing facility to our customers to identify the user of the system and check the Personal information of the user and verify the change in the system setting do by the right person or not. Hacking approve system

2.1.4. Product Scope

The scope of project is depending upon the security of the system. We are making the IMS(identity management system) in which we are using the blockchain technology in which unauthorized person try to break the system security the all information will be disable.

2.1.5. References

https://github.com/peacekeeper/blockchain-identity

2.2. Overall Description

2.2.1. Product Perspective

Our system will store the following information in our database:

Identity card detail:

Name, identity number, Gender, Father name, Address, issue Date.

Customer description/ home addresses:

It includes customer name, address and phone number. This information may be used for keeping the records of the customers for any emergency or for any other kind of information.

2.2.2. Product Functions

Functional Requirements:

Account Management

- Ability of system to create one administrator account with credentials on system creation.
- Ability of administrator to add new network users (Administrators, Auditors and Registrars) accounts.
- Ability of administrators to issue digital certificates to network users.
- Ability of administrators to revoke digital certificates of network users.
- System has the ability to issue new digital certificates on revocation.
- System has the ability to list all accounts with their personal details.

Update

- System has the ability to update network user's personal information.
- System has the ability to update digital certificates of network users on expiration

Delete

- System has ability to delete a network user.

Identity Management

- System has ability to create new birth certificate of a citizen.
- System has the ability to also save the applicant information corresponding to the new birth certificate.
- System can create new CNIIC.
- System will also save the applicant information corresponding to the new CNIC.
- System has the ability to create Marriage Registration Certificate.
- System will also save the applicant information corresponding to the new Marriage Certificate

Update

- Ability of system to update of birth certificate.
- Ability of system to update of CNIC.
- Ability of system to update of Marriage Certificate.
- .

Delete

- System has ability to delete birth certificates.
- System has ability to delete CNIC.
- System has ability to delete MRC (Marriage Registration Certificate).

Block chain Management

- Ability of system to deploy chain-code in the network.
- Ability of the system to show total chain-code deployed in the Blockchain network.
- Ability of system to determine transactions size.
- Ability of the system to list recent transactions.
- Ability of system to join peer of the organization.
- Ability of system Install the chain code.
- Ability of system to invoke the the chain code.

Update

- Ability to update chain-code from the network.

Manage Explorer

- The explorer uses by the admin for check the chain-code.
- Ability of administrator to all explorer block details.
- Ability of administrator to all explorer transaction details.
- Ability of administrator to all explorer channel details.
- Ability of system to view all network in the blockchain.

Manage user Activities

- The user will register and create it account.
- Ability of user send a request create new certificate.
- Ability of user sends a request view exiting certificate.
- Ability of user sends a request update information in existing certificate.

2.2.3. User Classes and Characteristics

The system will support three types of user privileges, Registrar/Employee, and Admin and citizen/users. registrar will have access to registrar functions, and the admin will have access of the system management and user only send the data for any purpose.

The user should be able to do the following functions:

- Create certificate
- Update certificate
- View certificate

The Registrar should be able to do the following functions:

- Add the record
- Select record to the report
- Edit record (in case of wrong record is selected or wrong information is added within specific time limit)

Admin management functions:

- Get all record of customers (view record)
- Manage those record (in case wrong information is provided or a wrong record is given, delete or verify them)
- Manage the report (view the report, edit, delete, ets)

2.2.4. Operating Environment

Operating environment for our system is as listed below.

- It can be open on window 10(optional), Linux
- The processor should be at least Pentium 3 or above.
- The processor speed should be greater than 400Mhz
- Operating system: Linux
- Ram should be or greater than 12 Gp.

2.2.5. Design and Implementation Constraints

- We should follow the IEEE standards.
- Default Language will be English.
- Project will follow all the copyright and cyber laws of PTA (Pakistan Telecommunication Authority).

2.2.6. User Documentation

Users of the website must know how to navigate in a website.

- We will organize help-line for customer support in the future.
- We will make a blog IMS.blogspot.com for online help for the customers.

2.2.7. Assumptions and Dependencies

Website IMS.com.pk is only accessible through the Internet it is assumed that the end user has a connection to the Internet. It is also assumed that the user has a web browser able to display the website. (I.E. Chrome, Microsoft Internet Explorer 4+ or compatible browser)

2.3. External Interface Requirements

2.3.1. User Interfaces

The system will provide the ability for users to access our website via the Internet.

Registrar

Users will able to login, view their record, could see their task.

User Interface:

- Front-end software: Angular website.
- Back-end software: hyperlagerfabric, Mongo DB.

2.3.2. Hardware Interfaces

- Processor: Pentium 4 or Higher.

- RAM: 12GB or Higher.

2.3.3. Software Interfaces

- Operating System: Linux, Windows
- Development tool: JavaScript, Angular
- Database: hyperlagerfabric& Mongo DB

2.3.4. Communications Interfaces

- This project supports most of the web browsers. E.g. Chrome, Internet Explorer.
- The protocol used shall be HTTP.
- The Port number used will be 80.

2.4. System Features

2.4.1. System Feature 1

The Authentication of system perform in this feature.

2.4.1.1 Description and Priority

Our system is used by the registrar, admin and user so if any registrar needs the change in user record argent-based than it sends the requested to admin perform the task if admin approve than registrar start it.

2.4.1.2 Stimulus/Response Sequences

Click "Register" Button: Account Registration

- only Admin will make an account of new registrar
- users will be able to create its account.
- The admin will enter the information of the new registrar like username, email, ets.
- The admin will send the username and password to new registrar.

- The admin store the information in the database.

Click "Login" Button: Account Login

- The system will allow a registered user/registrar to log-in to their account.
- The system will require a username and password from the user.
- The system will verify the username and password, and the user will be considered "logged-in".

Click "Logout" Button: for account Logout

- if user/registrar click the logout button than all session is created in account is discard
- The system is bank login page.

Functional Requirements

- Click "Register" Button for Account Registration
- Click "Login" Button: for Account Login
- Click "Logout" Button: for account Logout

2.4.2. System Feature 2

In this feature the admin/registrar manage the citizen assets.

2.4.2.1. Description and Priority

The admin will create the certificate and also manage the citizen assets for example create birth certificate of the person

2.4.2.2. Stimulus/Response Sequences

Click "create Birth Certificate" Button: Create certificate

- The Admin/registrar will create the certificate
- The user will enter the information of the like name, DOB, Gender, ets.
- The admin/registrar store the information in the database.

Click "create Marriage certificate" Button: Create Marriage certificate

- The Admin/registrar will create the certificate
- The user will enter the information of the like names, Date, ID Card No of both, ets.
- The admin/registrar store the information in the database.

2.4.2.3. Functional Requirements

Click "create Birth Certificate" Button: Create certificate

Click "create CNICCertificate" Button: Create Death certificate

Click "create Marriage certificate" Button: Create Marriage certificate

2.3.4 System Feature 4

In this feature the Admin will manage Reports

2.3.4.1 Description and Priority

The admin will create the report and also manage the report for example if expiry date of the ID card of existing users report

Stimulus/Response Sequences

Click "create task reports" Button: Create report

- only Admin will create the report.
- The admin store the information of report in the database.

2.3.5 System Feature 5

In this feature the Admin will manage blockchain

2.3.5.1 Description and Priority

The admin will check the blockchain chain-code on the existing record for example if the admin will check the whole chain that was available on blockchain

Stimulus/Response Sequences

Click " deploy chain-code" Button: deploy the chain-code

- only Admin will deploy
- The admin store the information in the database.

Click " update chain-code" Button: updating chain-code

- only Admin will create the update
- The admin will enter the information of the like date of update, reason of update,ets.

Click "view total chain-code" Button: view of chain-code

- only Admin will check this record
- The admin will send the request to check the total chain-code

Functional Requirements

- Click " deploy chain-code" Button: deploy the chain-code
- Click " update chain-code" Button: updating chain-code
- Click "view total chain-code" Button: view chain-code

2.3.6 System Feature 6

In this feature the Admin will manage the Explorer

2.3.6.1 Description and Priority

The admin will check the all Blockchain detail like how many node create ,block,

Transaction etc

Stimulus/Response Sequences

Click "users Activities" Button: check transaction

- only Admin will check these details
- The admin view the transaction detail .

Click "Nodes details" Button: check nodes details

- only Admin will check the nodes information
- The admin will check the information of the like storage, memory, location, ets.

Click "Record transaction details" Button: check Record

- only Admin will check the record of the citizens
- The admin will check the all record of the citizens like name, father name, ets.

Functional Requirements

- Click "users Activities" Button: check user’s activities
- Click "Nodes details" Button: check nodes details
- Click "Record transaction details" Button: check Record

2.5. Other Nonfunctional Requirements

2.5.1. Performance Requirements

For making a Website it is very necessary to improve speed, accuracy & performance of website.

2.5.2. Safety Requirements

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed-up log, up to the time of failure.

2.5.3. Security Requirements

Security systems need database storage just like many other applications. However, the special requirements of the security market we are using the blockchain technology to store the data in the database so the any one can attack on data the data store in block form if one block damage than all data will be locked only admin can open it

2.5.4. Software Quality Attributes

Availability: Website should be available on time so user would able to access.

Correctness: Correct information should display in the system

so user select correct option.

Maintainability: The administrators should be maintain website.

Usability: Should satisfy customers need.

2.6. Other Requirements

Other Requirement depend upon the system features.

Chapter 3

Use Case Analysis

3: System Analysis

Creating Use Cases to describing the whole scenario how it is going to work. Describing actors & their purposes. How users will login, create Certificate etc. And users can edit Certificate in case of wrong information is added.

3.1. Use Case Model

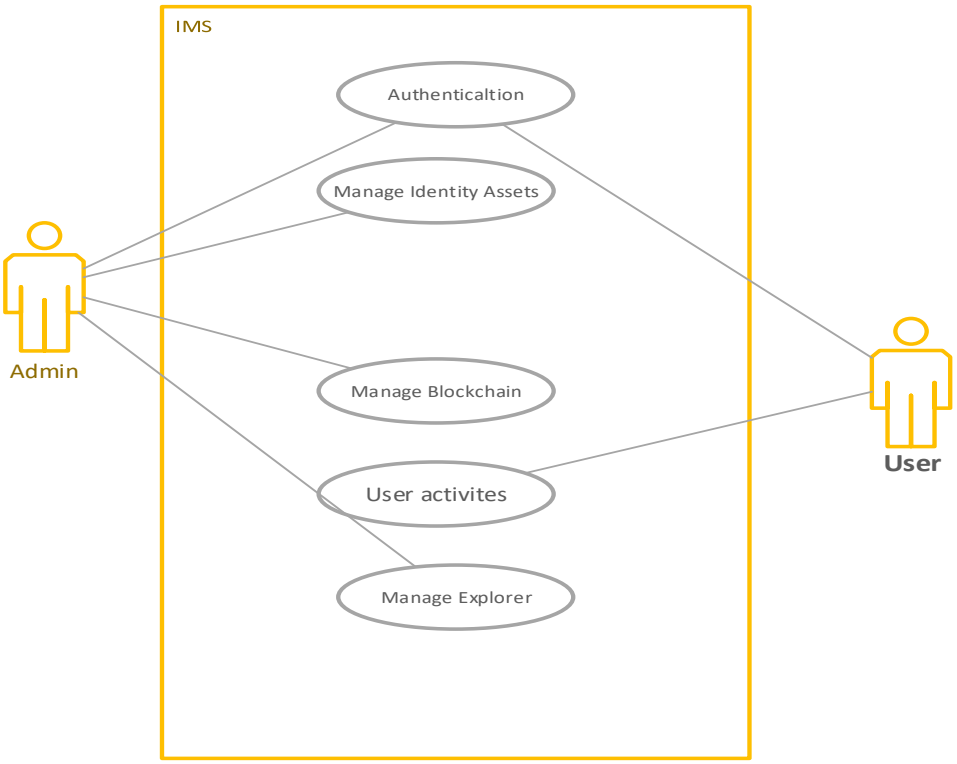


Figure 2-Use Case

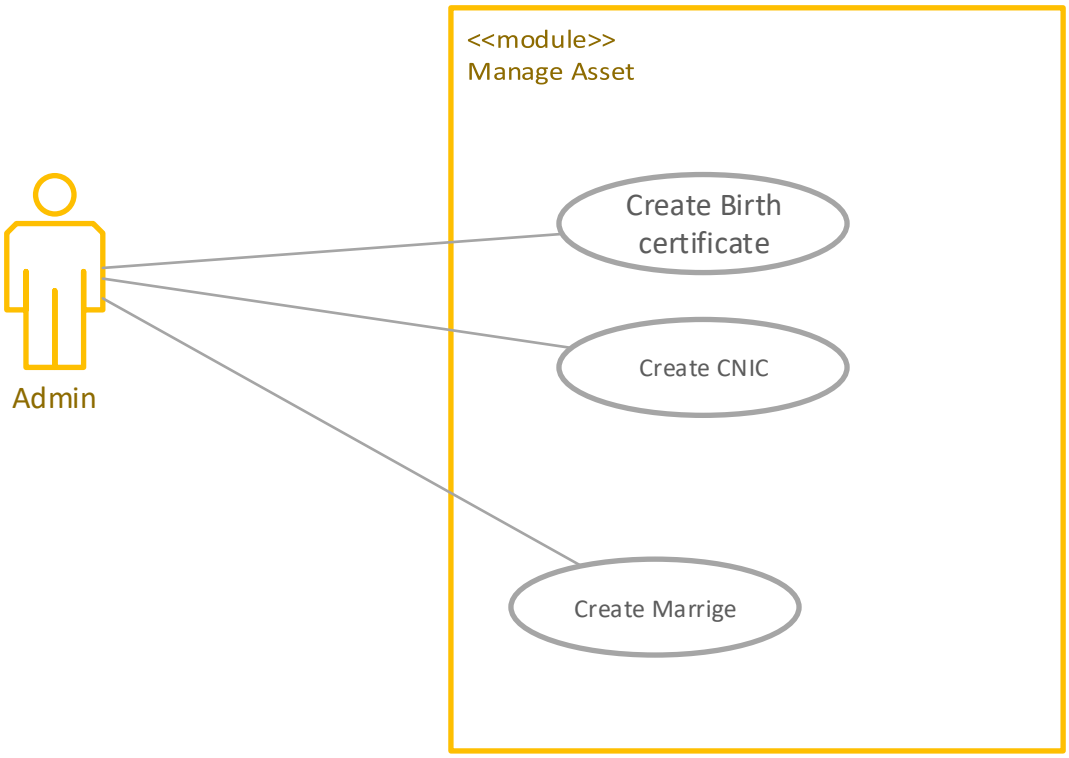


Figure 3-Manage Asset

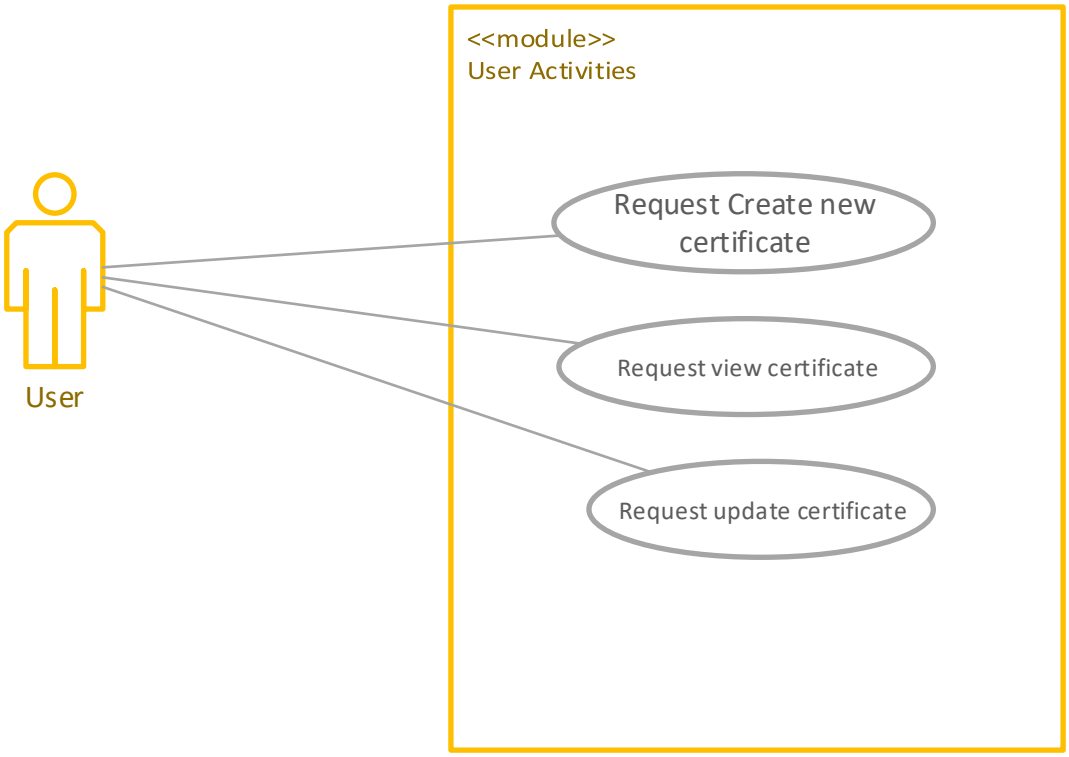


Figure 4-User Activities

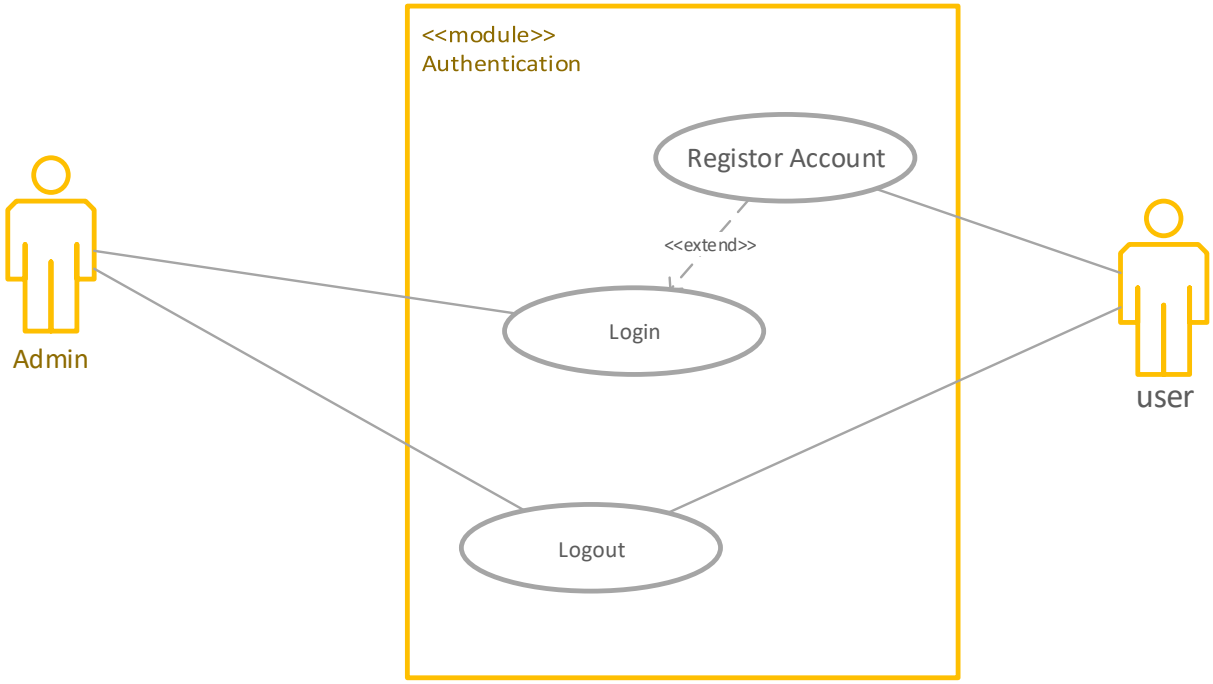


Figure 5-Authentication

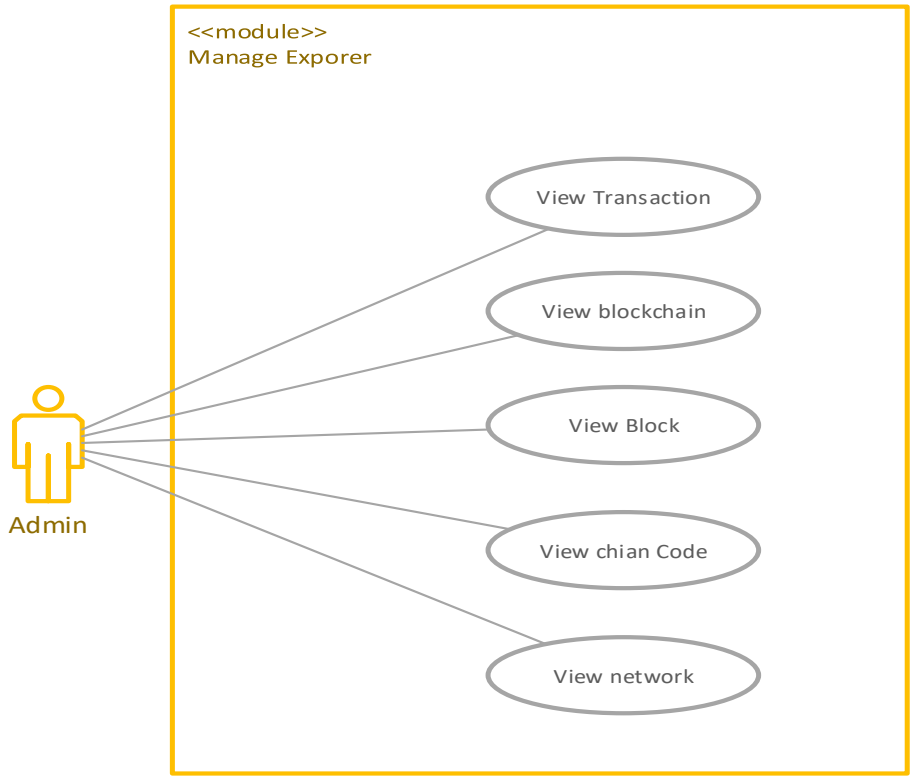


Figure 6-Manage Explorer

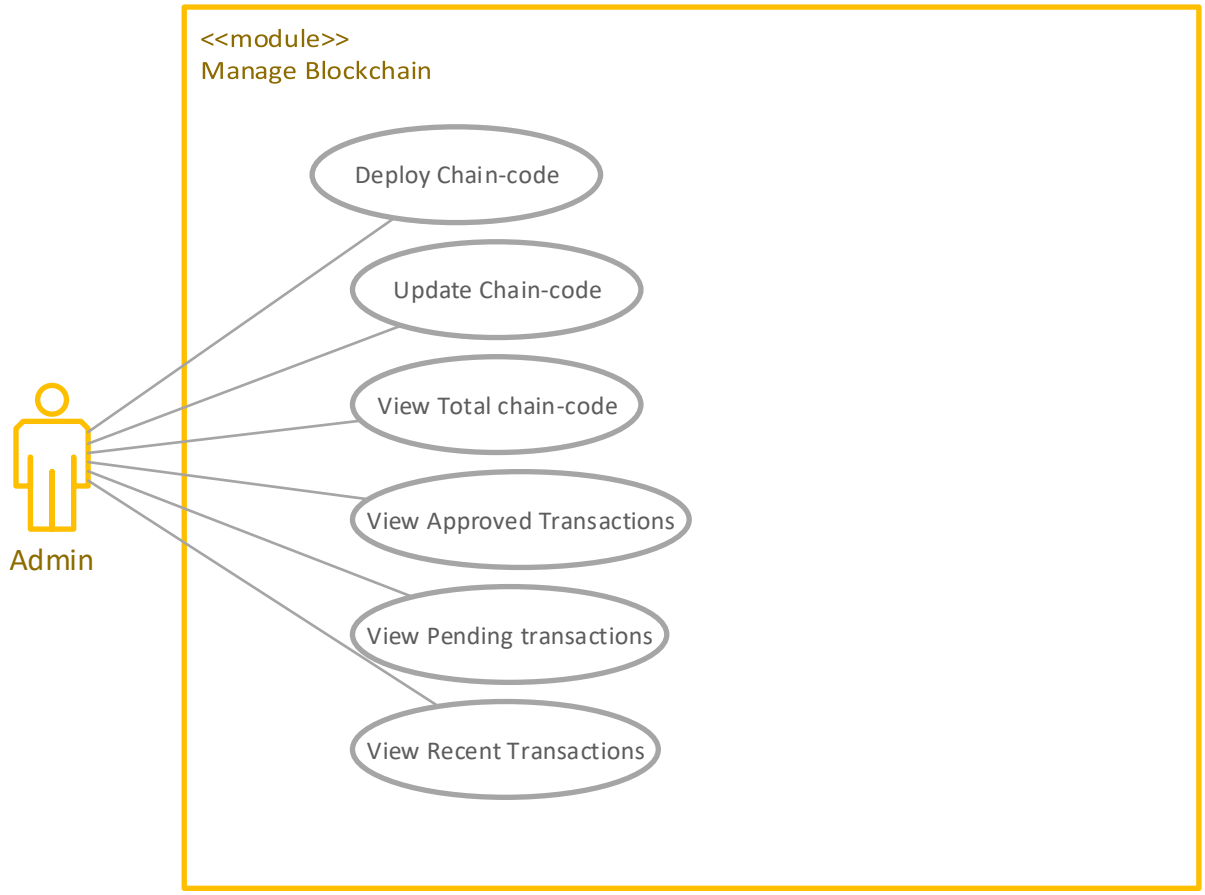


Figure 7-Blockchain

3.2. Fully Dressed Use Cases

Use Case ID:	UC-1.0.0		
Use Case Name:	Authentication		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Registrar, Admin, user		
Description:	In Authentication case user, registrar/admin will login in the system to get the access to the system.		
Trigger:	Admin/Registrar will enter username and password and will click the login button. As soon as user logged in, an admin/registrar dashboard will be displayed.		
Preconditions:	1. Admin/Registrar& user should be registered in system. 2. Admin/Registrar& user should have valid username and password.		
Post conditions:	1. Admin/Registrar login successfully.		
Normal Flow:	1. Admin/Registrar enters the username and password. 2. System validate the user credentials. 3. On success admin/Registrar dashboard will be displayed.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	1a. In step 1 of the normal flow, if user enter incorrect username or password. 1. System response, please enter correct user name and password and show input field again to user. 2. User enter the correct name and password. 3. Use case resume on step 3 of the normal flow.		
Exceptions:	1. Connection is not established with database. 2. User don't have an internet connection.		
Includes:	Login Logout		
Frequency of Use:	On demand		
Special Requirements:	User should have username and password.		
Assumptions:	The IMS user understands either English.		
Notes and Issues:			

Table 2-Full Dress Use Case Authentication

Use Case ID:	UC-1.1.0		
Use Case Name:	Login to the system		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Admin (primary) Registrar (Primary) User		
Description:	This use case help admin and registrar to log in to the system.		
Trigger:	User request to login.		
Preconditions:	1. The user should have an account in IMS. 2. The user will try to login with valid user name and password. 3. The user is not already logged in.		
Post conditions:	1. The user is logged in to the system. 2. The user has access to the function of the system.		
Normal Flow:	1. User access the URL. 2. The system prompts the user for their account credentials. 3. User enter user name and password. 4. Server validate user name and password from db(database) 5. The user gain access to the system functionality. 6. User perform activity. 7. User log out the system.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	1a. In step 1 of the normal flow, if user enter incorrect username. 1. User enter the incorrect URL. 2. System response incorrect page or page not found. 3. User enter the correct URL. 3a. In step 3 of the normal flow, if user enter incorrect username or password. 1. System response, please enter correct user name and password and show input field again to user. 2. User enter the correct name and password. 3. Use case resume on step 4 of the normal flow.		
Exceptions:	1. Connection is not established with database. 2. User don't have an internet connection.		
Includes:	None.		
Frequency of Use:	On demand.		
Special Requirements:	There is no special requirements associated with this use case.		
Assumptions:	1. The user want to use the system. 2. The IMS user understands either English or Urdu language.		
Notes and Issues:	This is include and precondition for all other use cases.		

Table 3-Full Dress Use Case Login

Use Case ID:	UC-1.3.0		
Use Case Name:	Logout		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Admin (primary) Registrar (Primary) User		
Description:	This user clicks on “logout” and their session is terminated.		
Trigger:	User wants to logout. User is done using the system.		
Preconditions:	1. The user has an account in IMS. 2. The user is login in system. 3. The user is done using the system.		
Post conditions:	The user is logged out.		
Normal Flow:	1. The user is done using the web application. 2. The user clicks on the “logout” button. 3. The system redirects to the default logout page.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	N/A		
Exceptions:			
Includes:	Login to the system.		
Frequency of Use:	Whenever the user wants to logout.		
Special Requirements:			
Assumptions:	No one can use the system after the user has successfully logout.		
Notes and Issues:			

Table 4-Full Dress Use Case Logout

Use Case ID:	UC-5.0.0		
Use Case Name:	Manage Explorer		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Admin (primary)		
Description:	This use case help admin to manage reports of system data e.g. no. of transaction vs size of transaction.		
Trigger:	Admin want to see summary of systems data.		
Preconditions:	Admin logged in. Admin want to perform certain operations with system data i.e. View reports, Generate reports, Delete reports, Export reports.		
Post conditions:	Admin manage reports successfully.		
Normal Flow:	1. Admin login the system. 2. System validate the admin id and password. 3. System show the dashboard page to the user. 4. Admin click on “Manage Report” button. 5. System show report page to admin in which admin perform certain operation i.e. Generate reports, View reports, Delete reports and Export reports. 6. Admin click on “Save” button. 7. Report save in database. 8. Admin do next that he wants.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	6a, In 6a of the normal flow, If admin does not want to save report. 1. Admin click on “Cancel” button. Use case resume on step 8 of the normal flow.		
Exceptions:	1. Reports are generated yet.		
Includes:	View reports Generate reports Delete reports Export reports		
Frequency of Use:	On demand		
Special Requirements:	Admin have rights to manage reports.		
Assumptions:	No one can manage monitoring other than admin.		
Notes and Issues:			

Table 5-Full Dress Use Case Manage Explorer

Use Case ID:	UC-5.2.0		
Use Case Name:	View Explorer		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Admin		
Description:	This use case help admin to view reports of system data.		
Trigger:	Admin want to see summary of system data e.g no. of transaction vs size of transaction.		
Preconditions:	Admin logged in. Admin want to view reports of system data.		
Post conditions:	Admin view reports.		
Normal Flow:	1. Admin login the system. 2. System validate the admin id and password. 3. System show the dashboard page to the user. 4. Admin click on “Manage Report” button. 5. Admin click on “view report” button. 6. System show all reports that were saved in database to admin. 7. Admin select report and click on “View” button. 8. Admin do next that he wants.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	6a, In 6a of the normal flow, If system generate report which is not useful for admin and admin does not want to save it. 1. Admin click on “cancel” button. Use case resume on step 11 of the normal flow.		
Exceptions:	1. Reports are not generated yet.		
Includes:	N/A		
Frequency of Use:	On demand		
Special Requirements:	Admin have rights to manage reports.		
Assumptions:	No one can manage monitoring other than admin.		
Notes and Issues:			

Table 6-Full Dress Use Case View Explorer

Use Case ID:	UC-6.0.0		
Use Case Name:	Manage Blockchain		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: Admin		
Description:	This use case modulehelps admin to manage blockchain. Admin can view approved pending and recent transactions. Admin can also update chain code.		
Trigger:	After successful login admin will click on “Manage Blockchain”. Admin will then choose an option either click on one of the functions among “Deploy Chain code”, “Update Chain-Code”, “View total Chain Code”, “View Approved Transactions”, “View Pending transactions”, “View Recent Transactions”.		
Preconditions:	1. User must be logged in. 2. User must have admin rights to access the Manage Blockchain section.		
Post conditions:	1. Admin successfully performs the required operation.		
Normal Flow:	1. Admin clicks the “Manage Blockchain” button. 2. Admin selects one of the options to perform an operation.		
Alternative Flows: [Alternative Flow 1 – Not in Network]			
Exceptions:			
Includes:			
Frequency of Use:	Admin may frequently use this module to monitor transactions.		
Special Requirements:	Application must be linked withBlockchain and must have a stable internet connection to generate the reports.		
Assumptions:			
Notes and Issues:			

Table 7-Full Dress Use Case Manage Blockchain

Use Case ID:	UC-6.1.0		
Use Case Name:	Deploy Chain-Code		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: Admin		
Description:	This use case modulehelps admin to deploy Chain code in the Blockchain.		
Trigger:	After login admin clicks the Deploy Chain Code button to deploy the chain code.		
Preconditions:	1. User must be logged in. 2. User must have admin rights to access the Manage Blockchain section.		
Post conditions:	1. Admin successfully performs the required operation.		
Normal Flow:	1. After successful login admin will click on “Manage Blockchain”. 2. Admin will then choose the option “Deploy Chain code”. 3. Admin will insert the chain-code. 4. Admin will click “Deploy” button.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	3a, In 3a of the normal flow, If user deploy chaincode which have syntax error then; 1. System show error for syntax error. 2. Admin will correct syntax error. 3. Admin will again deploy chaincode. Use case resume on step 4 of the normal flow.		
Exceptions:	1. Syntax error in deploying code.		
Includes:			
Frequency of Use:	Admin may not use this function frequently.		
Special Requirements:	Application must be linked with Blockchain. Chain-code will not deploy if application is not connected to Blockchain.		
Assumptions:			
Notes and Issues:			

Table 8-FDUC deploy chain code

Use Case ID:	UC-6.2.0		
Use Case Name:	Update Chain-Code		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: Admin		
Description:	This use case modulehelps admin to update Chain code in the Blockchain.		
Trigger:	After login admin clicks the Update Chain Code button to update the chain code.		
Preconditions:	1. User must be logged in. 2. User must have admin rights to access the Manage Blockchain section.		
Post conditions:	Admin successfully performs the required operation.		
Normal Flow:	1. After successful login admin will click on “Manage Blockchain”. 2. Admin will then choose the option “Update Chain code”. 3. Admin will insert the new chain-code to replace the old one. 4. Admin will click “Deploy” button. 5. Chain-code successfully updated and to the network.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	3a, In 3a of the normal flow, If user deploy chaincode which have syntax error then; 1. System show error for syntax error. 2. Admin will correct syntax error. 3. Admin will again deploy chaincode. Use case resume on step 4 of the normal flow.		
Exceptions:	1. Syntax error in deploying code. 2. User is not connected to blockchain network..		
Includes:			
Frequency of Use:	Admin may not use this function frequently.		
Special Requirements:	Application must be linked with Blockchain. Chain-code will not update if application is not connected to Blockchain.		
Assumptions:			
Notes and Issues:			

Table 9-FDUC Update chain code

Use Case ID:	UC-6.3.0		
Use Case Name:	View Total Chain-Code		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: Admin		
Description:	This use case modulehelps admin to view Chain code in the Blockchain.		
Trigger:	After login admin clicks the View Chain Code button to deploy the chain code.		
Preconditions:	1. User must be logged in. 2. User must have admin rights to access the Manage Blockchain section.		
Post conditions:	Admin successfully performs the required operation.		
Normal Flow:	1. After successful login admin will click on “Manage Blockchain”. 2. Admin will then choose the option “View Chain code”. 3. Admin get access to Total Chain-code.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	NILL		
Exceptions:			
Includes:			
Frequency of Use:	Admin may not use this function frequently.		
Special Requirements:	Application must be linked with Blockchain to view the updated information.		
Assumptions:			
Notes and Issues:			

Table 10-FDUC Total chain code

Use Case ID:	UC-6.4.0		
Use Case Name:	View Approved Transactions		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: Admin		
Description:	This use case modulehelps admin to view approved transactions in the Blockchain.		
Trigger:	After login admin clicks the View approved transactions button to View the transactions.		
Preconditions:	1. User must be logged in. 2. User must have admin rights to access the Manage Blockchain section.		
Post conditions:	Admin successfully get access to approved transactions.		
Normal Flow:	1. After successful login admin will click on “Manage Blockchain”. 2. Admin will then choose the option “View Approved Transactions”. 3. Admin get access to Approved transactions details.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	NILL		
Exceptions:			
Includes:			
Frequency of Use:	Admin may use this function frequently.		
Special Requirements:	Application must be linked with Blockchain to view the approved transaction information.		
Assumptions:			
Notes and Issues:			

Table 11-FDUC Approve transaction

Use Case ID:	UC-6.6.0		
Use Case Name:	View Recent Transactions		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: Admin		
Description:	This use case modulehelps admin to view recent transactions in the Blockchain.		
Trigger:	After login admin clicks the View recent transactions button to View the transactions.		
Preconditions:	1. User must be logged in. 2. User must have admin rights to access the Manage Blockchain section.		
Post conditions:	Admin successfully get access to recent transactions made.		
Normal Flow:	1. After successful login admin will click on “Manage Blockchain”. 2. Admin will then choose the option “View Recent Transactions”. 3. Admin get access to recent transactions details.		
Alternative Flows: [Alternative Flow 1 – Not in Network]	NILL		
Exceptions:			
Includes:			
Frequency of Use:	Admin may use this function frequently.		
Special Requirements:	Application must be linked with Blockchain to view the recent transaction information.		
Assumptions:			
Notes and Issues:			

Table 12-FDUC View Recent Transaction

Use Case ID:	UC-7.0		
Use Case Name:	Request for new certificate		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: User		
Description:	This use case will send the request for create a new certificate		
Trigger:	After login user clicks the button of create certificate		
Preconditions:	3. User must be logged in. 4. User must have valid user account and fill the form of certificate		
Post conditions:	User will check it's information that was enter in the certificate form.		
Normal Flow:	4. After successful login admin will click on "create certificate". 5. user will then choose the option "type of certificate".		
Alternative Flows: [Alternative Flow 1 – Not in Network]	NILL		
Exceptions:			
Includes:			
Frequency of Use:	User use this function frequently.		
Special Requirements:	Application must be linked with Blockchain to view the user information.		
Assumptions:			
Notes and Issues:			

Table 13-FDUC Request for New Certificate

Use Case ID:	UC-7.1		
Use Case Name:	Request for view certificate		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: User		
Description:	This use case will send the request for view certificate		
Trigger:	After login user clicks the button of view certificate		
Preconditions:	5. User must be logged in. 6. User must have valid user account and fill the form of certificate		
Post conditions:	User will check it's information of its certificate.		
Normal Flow:	6. After successful login admin will click on "view certificate". 7. user will then choose the option "type of certificate".		
Alternative Flows: [Alternative Flow 1 – Not in Network]	NILL		
Exceptions:			
Includes:			
Frequency of Use:	User use this function frequently.		
Special Requirements:	Application must be linked with Blockchain to view the user information.		
Assumptions:			
Notes and Issues:			

Table 14-FDUC Request for View Certificate

Use Case ID:	UC-7.2		
Use Case Name:	Request for update certificate		
Created By:		**Last Updated By:**	
Date Created:		**Last Revision Date:**	
Actors:	Primary Actor: User		
Description:	This use case will send the request for update certificate		
Trigger:	After login user clicks the button of update certificate		
Preconditions:	7. User must be logged in. 8. User must have valid user account and fill the form of certificate		
Post conditions:	User will check it's information that was enter in the certificate form.		
Normal Flow:	8. After successful login admin will click on "update certificate". 9. user will then choose the option "type of certificate".		
Alternative Flows: [Alternative Flow 1 – Not in Network]	NILL		
Exceptions:			
Includes:			
Frequency of Use:	User use this function frequently.		
Special Requirements:	Application must be linked with Blockchain to view the user information.		
Assumptions:			
Notes and Issues:			

Table 15-FDUC Request for Update Certificate

Chapter 4

System Design

Chapter 4: System Design

We are designing the whole system conceptually, just to clear out how our system will look like. We have made different diagrams, different designs, Model, ER- Diagram to see how our system will work& how we will manage it.

4.1. Architecture Diagram

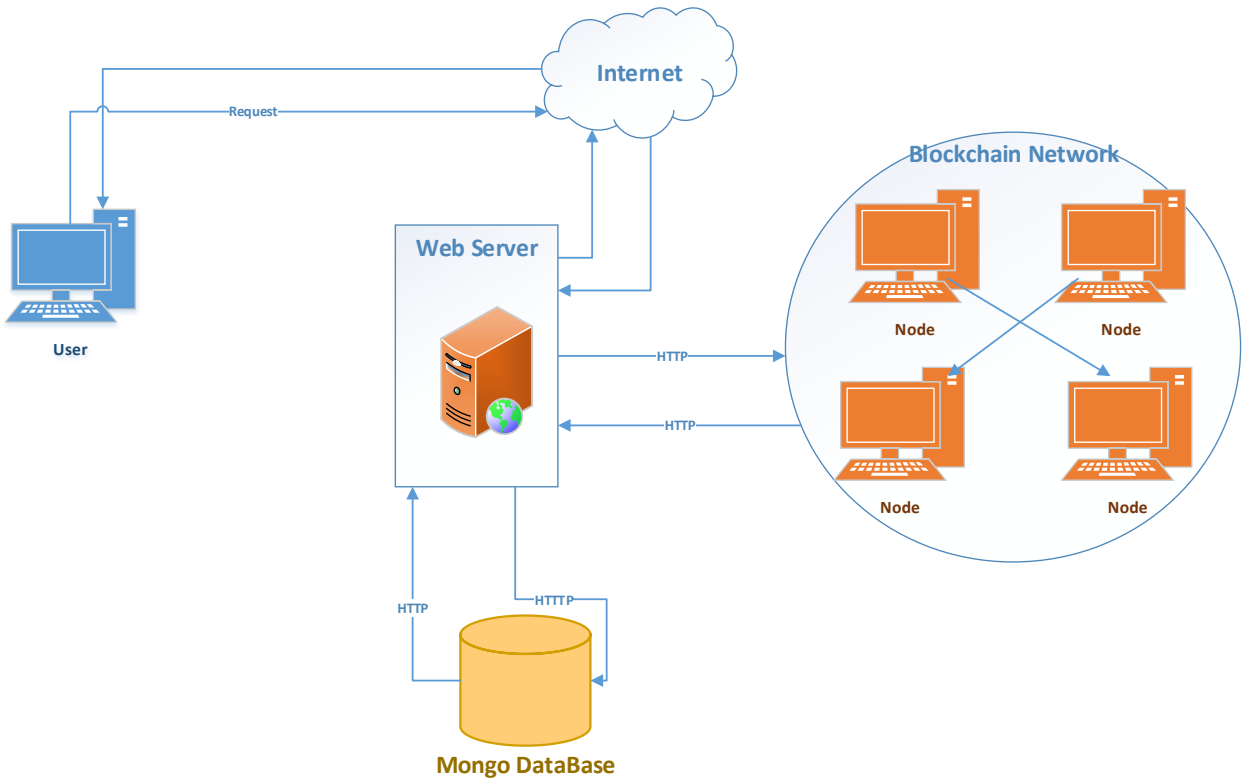


Figure 8-Architecture Diagram

4.2. Domain Model

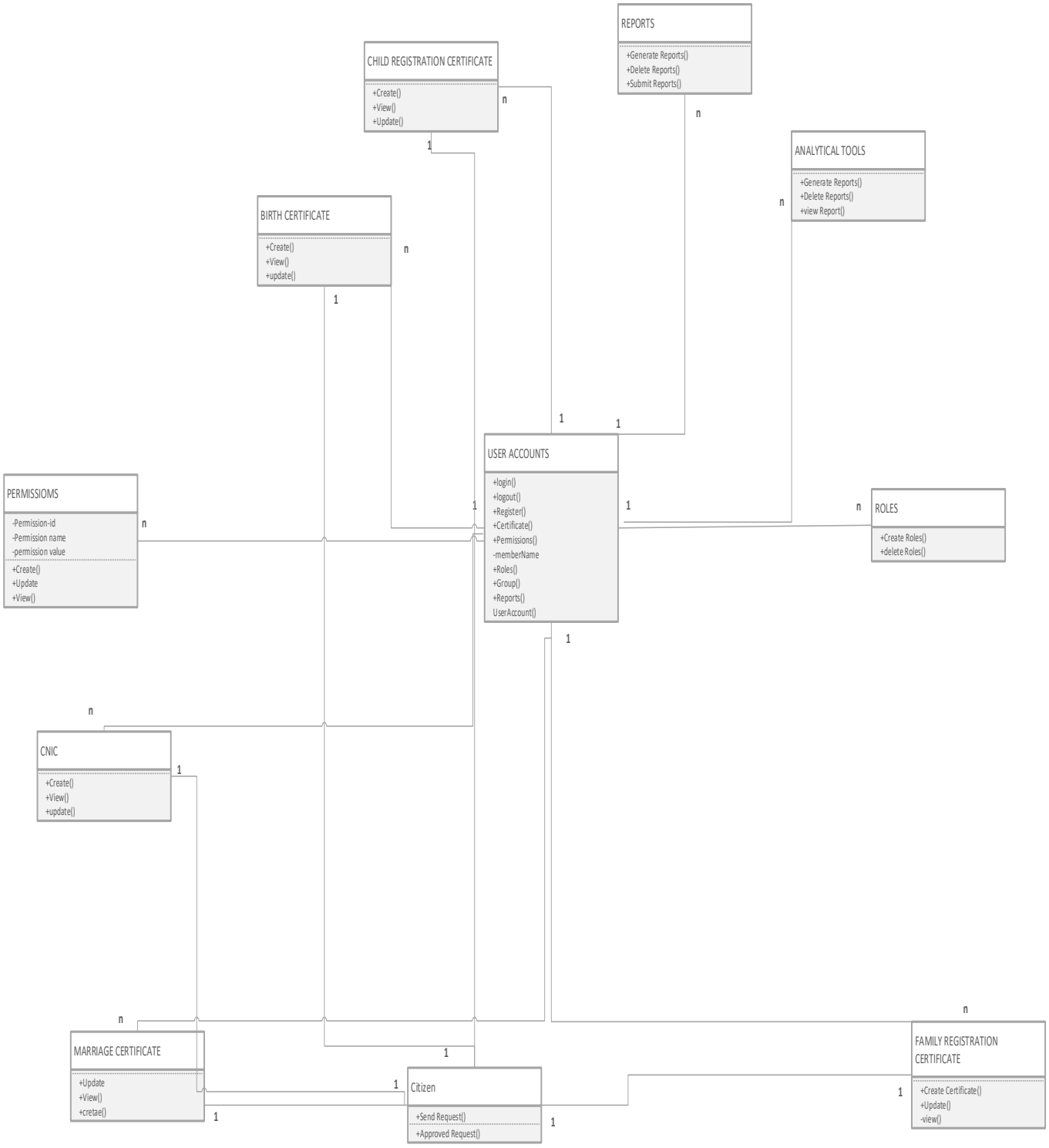


Figure 9-Domain Model

4.3. Entity Relationship Diagram with data dictionary

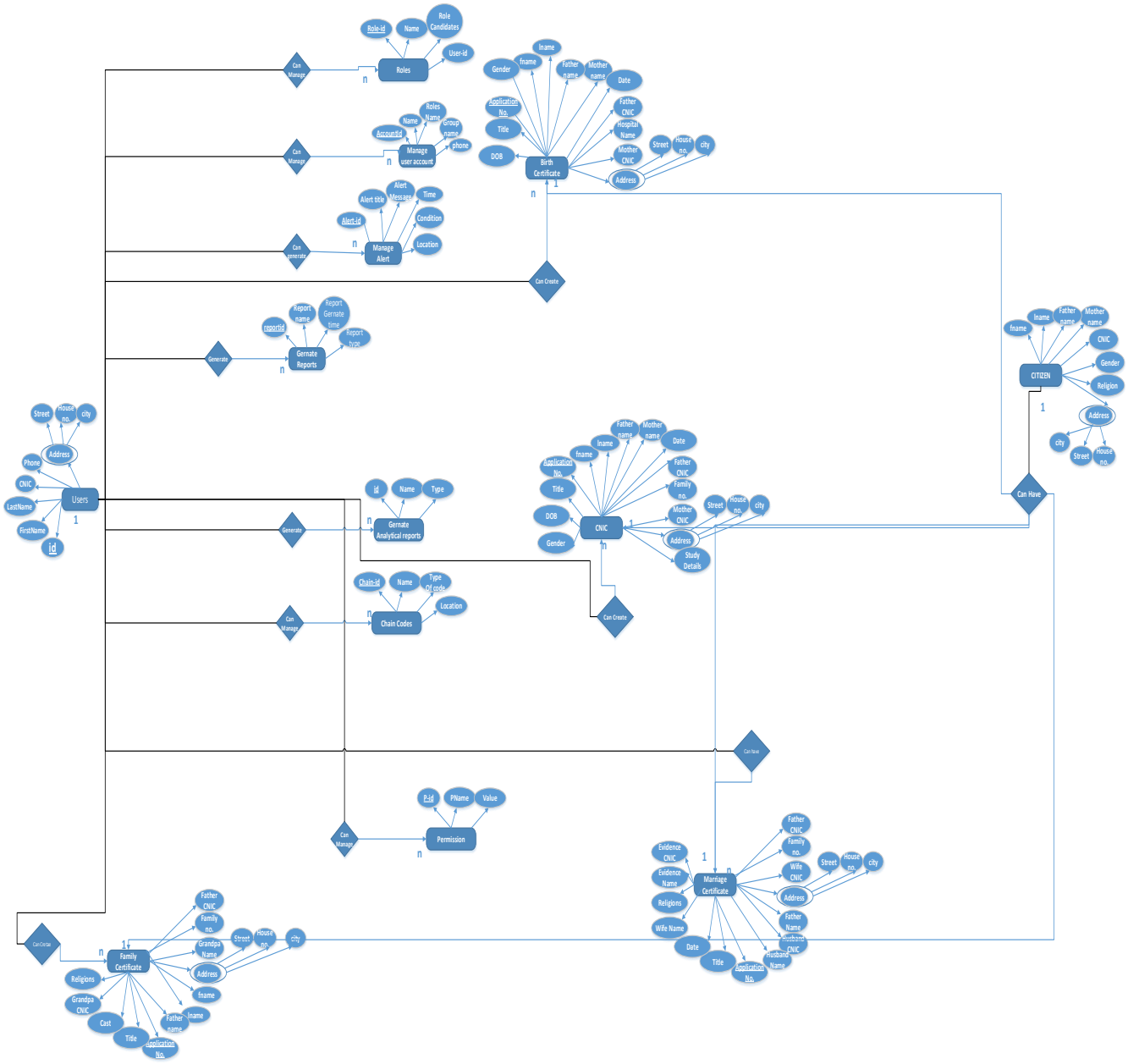


Figure 10-ERD

4.4. Class Diagram

Figure 11-Class Diagram

4.5. Sequence / Collaboration Diagram

Sequence Diagrams

<<Authentication>>
Login Sequence Diagram

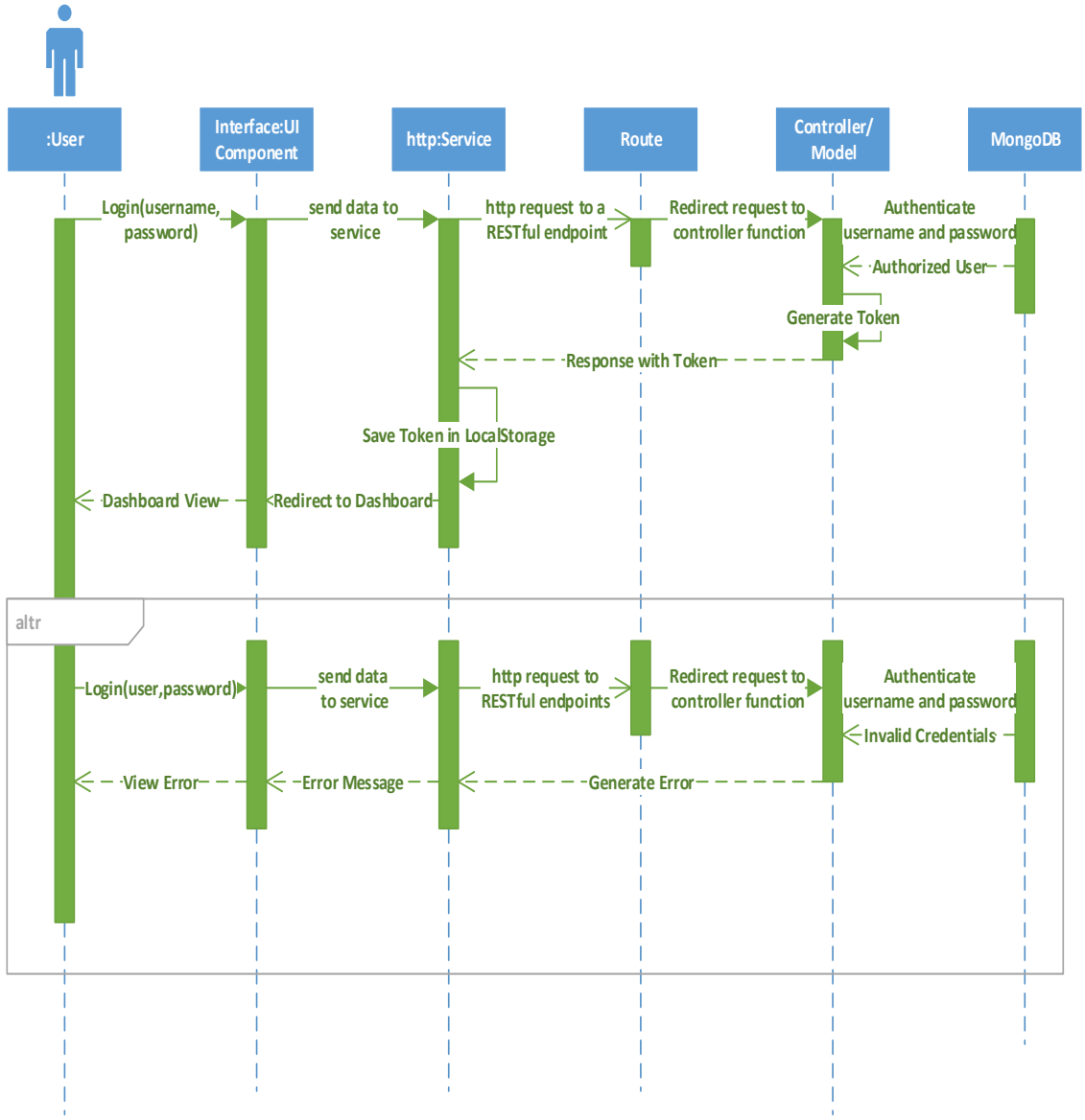


Figure 12-SD login

<<Authentication>>
Admin Login Sequence Diagram

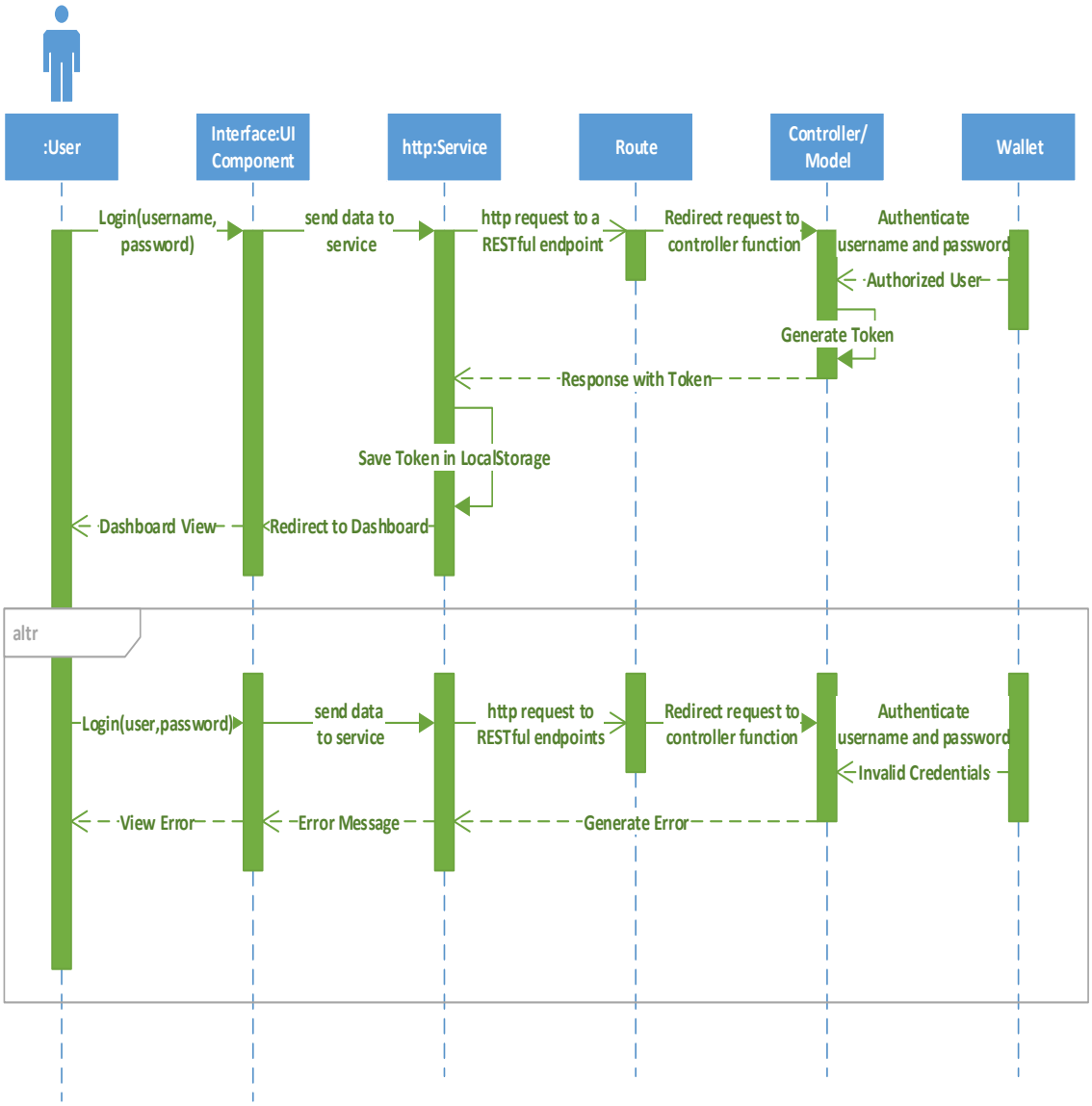


Figure 13-SD Admin Login

<<Manage Users Accounts >>
Manage Roles Sequence Diagram

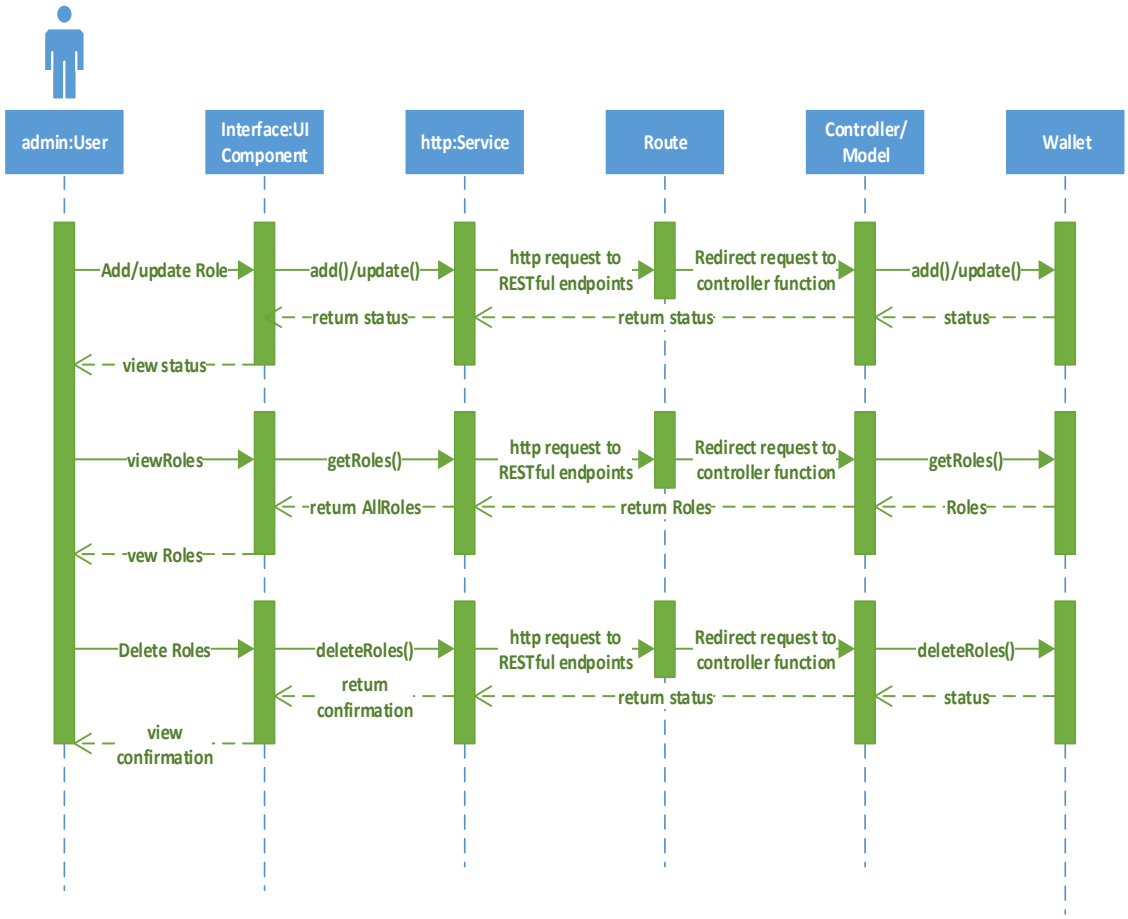


Figure 14-SD Role

<<Manage Users Accounts >>
Manage Users Sequence Diagram

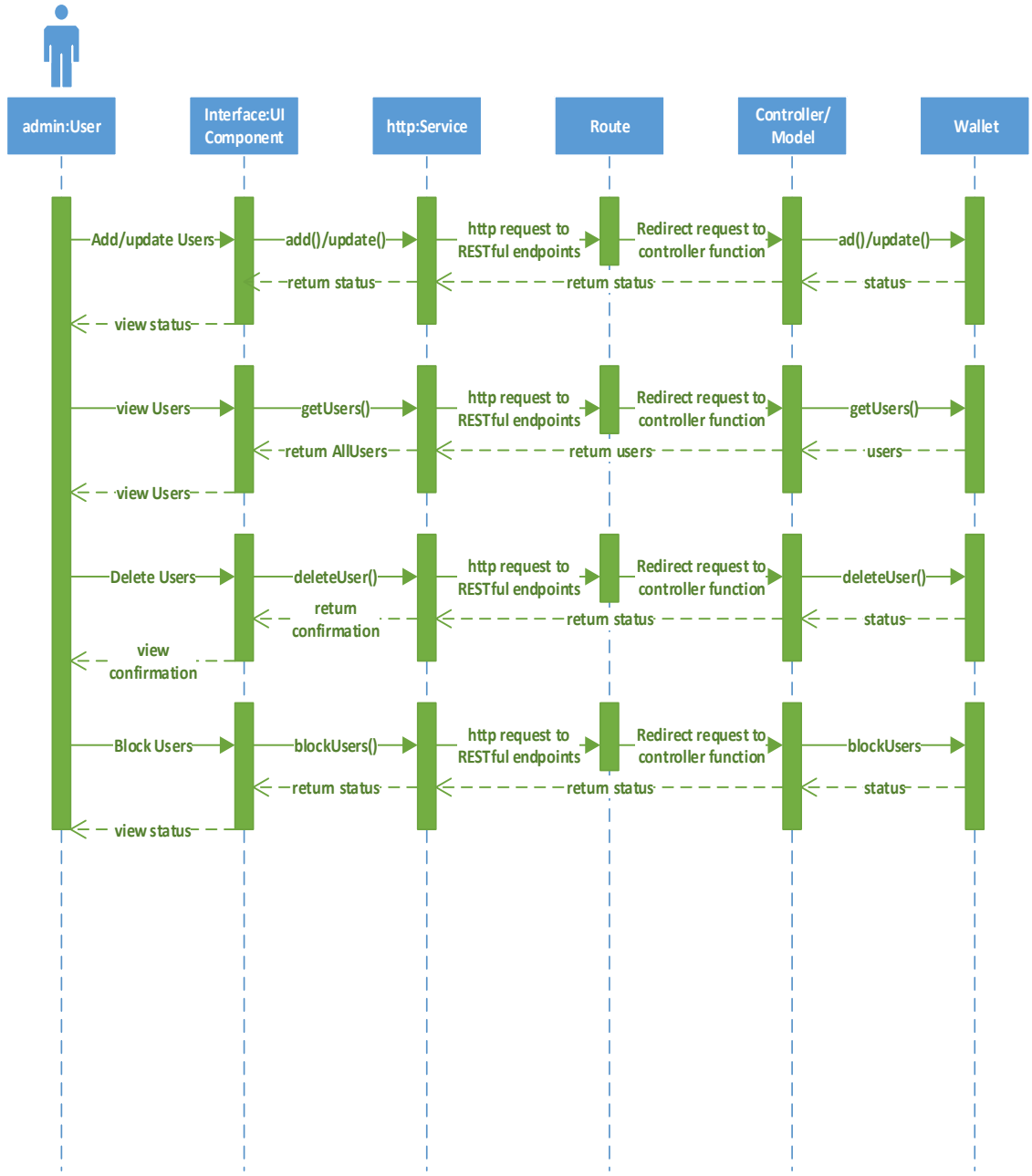


Figure 15-SD Manage User

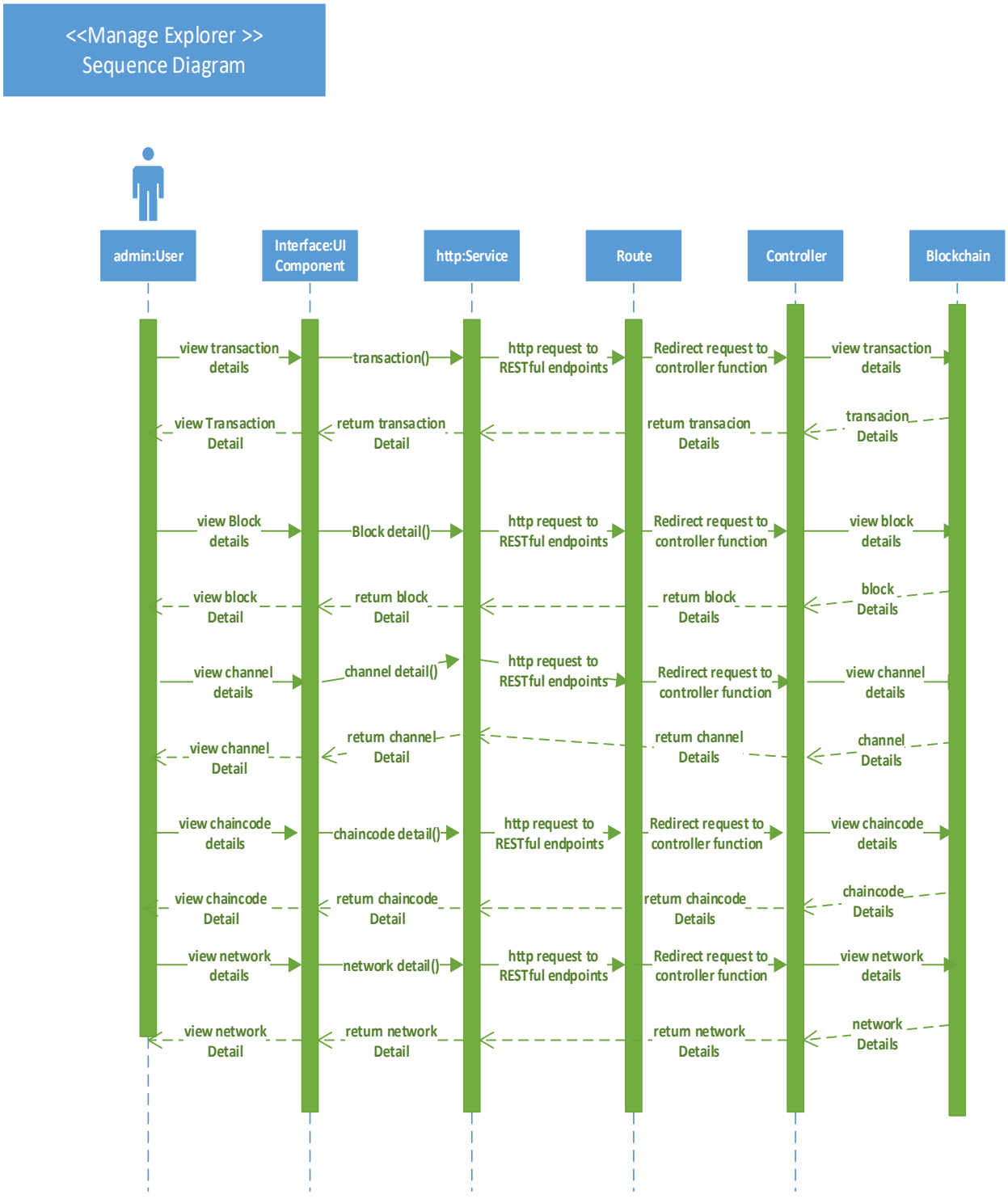


Figure 16-SD Manage Explorer

<<Manage Reports>>
Sequence Diagram

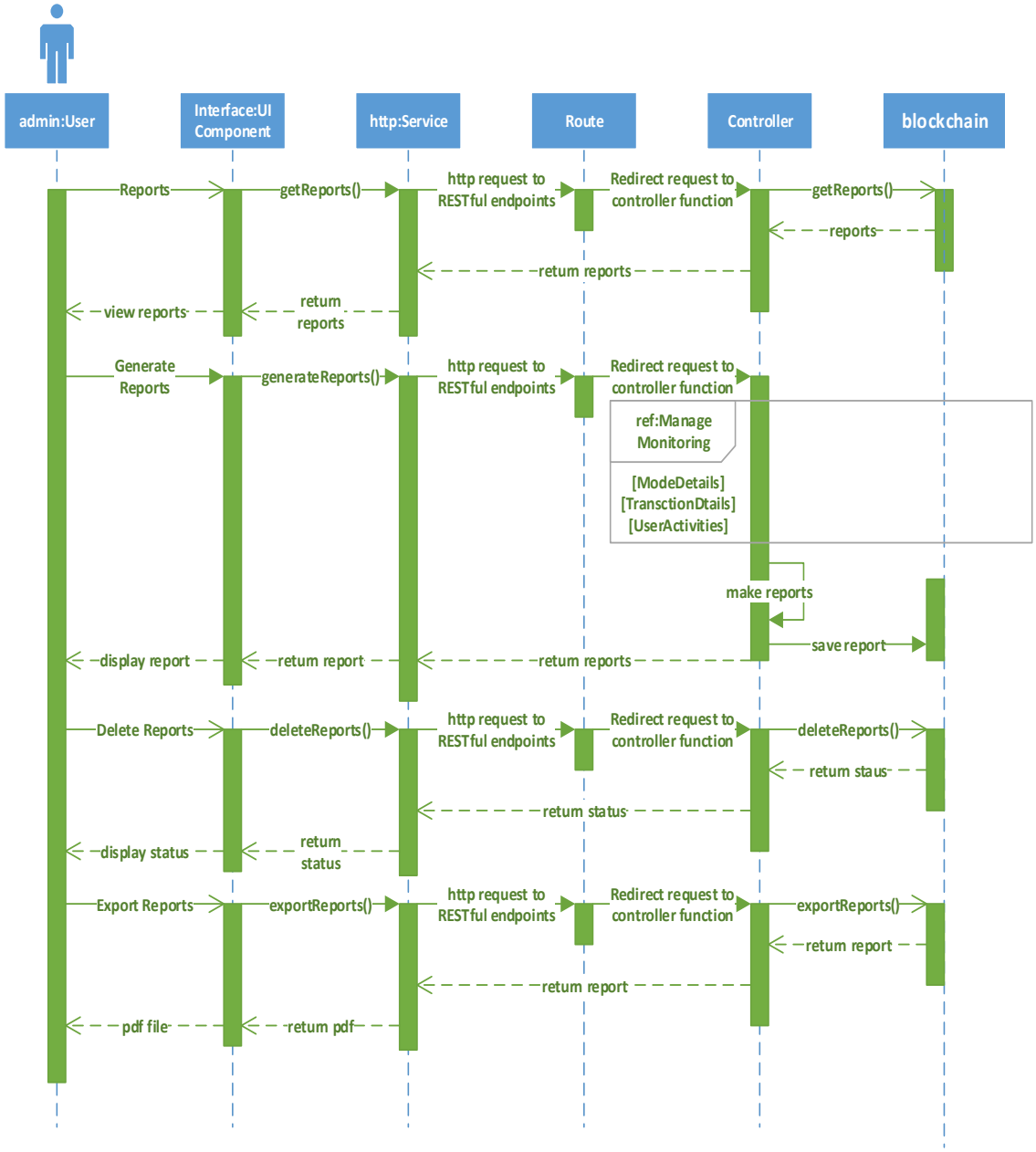


Figure 17-SD Manage Report

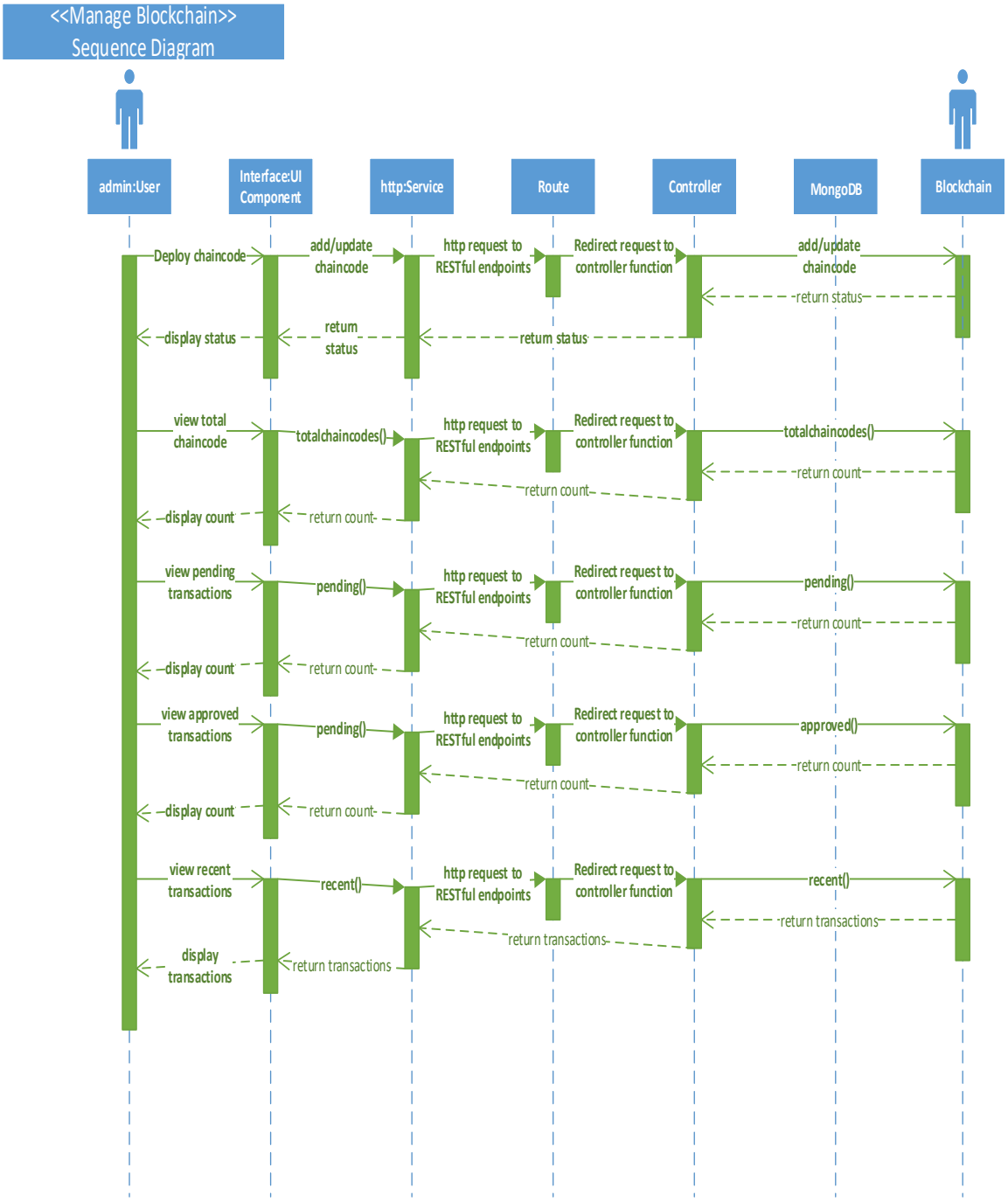


Figure 18-SD Manage BlockChain

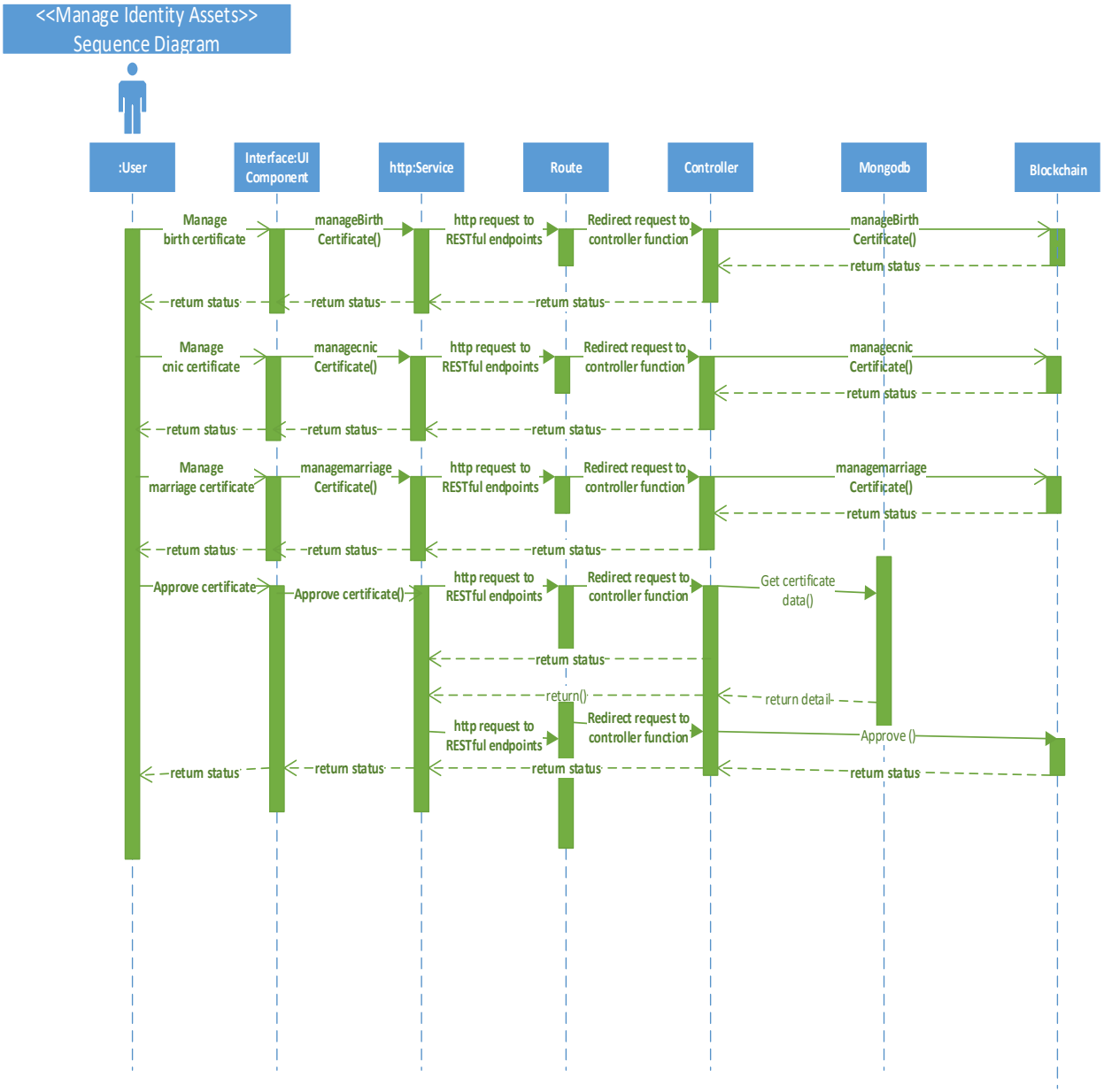


Figure 19-SD Manage Identity Asset

4.6. Operation contracts

4.6.1 Authentication

Operation#1: Login-request (username, password)
Cross Ref : view account
Pre-condition: valid login page
Post-condition:

- User-object
- userID, password

4.6.4 Manage Explorer

Operation#1: Nodes details (node-name, storage, memory, location)
Cross Ref: check the details of node
Pre-condition: valid Admin panel
Post-condition:system return the detail of node

Operation#2: Transaction details (record)
Cross Ref: check the details of given transaction
Pre-condition: valid Admin panel
Post-condition:system return the detail

Operation#3: user-Activities (username, login-time, logout-time)
Cross Ref: check the details of user
Pre-condition: valid Admin panel
Post-condition: system return the detail of given user

4.6.5 Manage Report

Operation#1: get-Report (Report-name, month, year)
Cross Ref: check the report
Pre-condition: valid Admin panel
Post-condition:system return the report

Operation#2: generate-Report (Att-name,)
Cross Ref: make the report
Pre-condition: valid Admin panel
Post-condition: system display the report

Operation#3: delete-Report (report-name, date)

Cross Ref: delete the report
Pre-condition: valid Admin panel
Post-condition: system send the massage the report is delete

4.6.7 Manage blockchain

Operation#1: deploy-chaincode(transaction,)
Cross Ref: apply the blockchain
Pre-condition: valid Admin panel
Post-condition: system send the massage record will change in chaincode

4.6.8 Manage Identity Assets

Operation#1: create-certificate (certificate-name,)
Cross Ref: create the certificate
Pre-condition: valid registrar/user panel
Post-condition: system will create the certificate

4.7. Activity Diagram

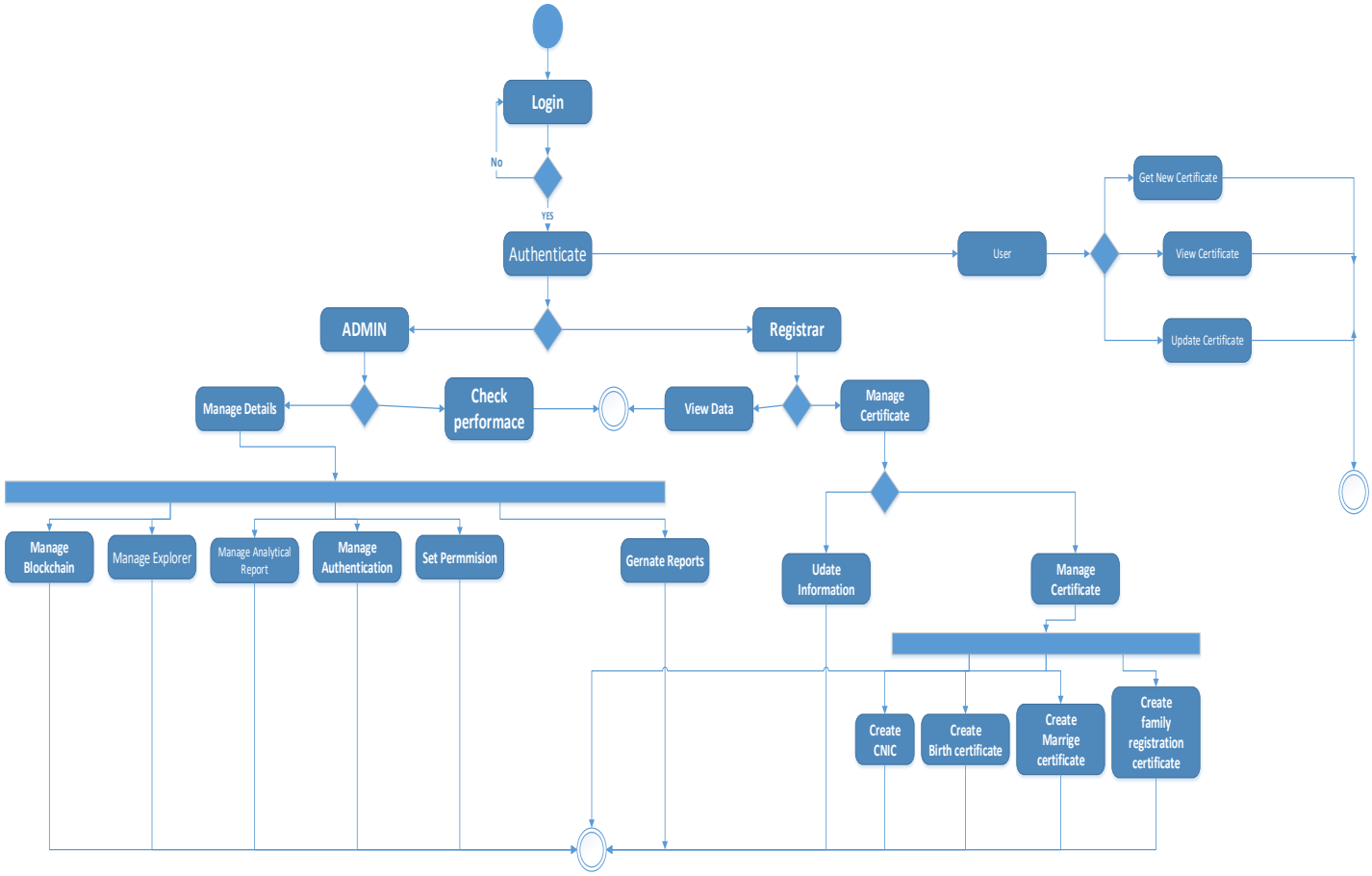


Figure 20-SD Activity Diagram

4.8. State Transition Diagram

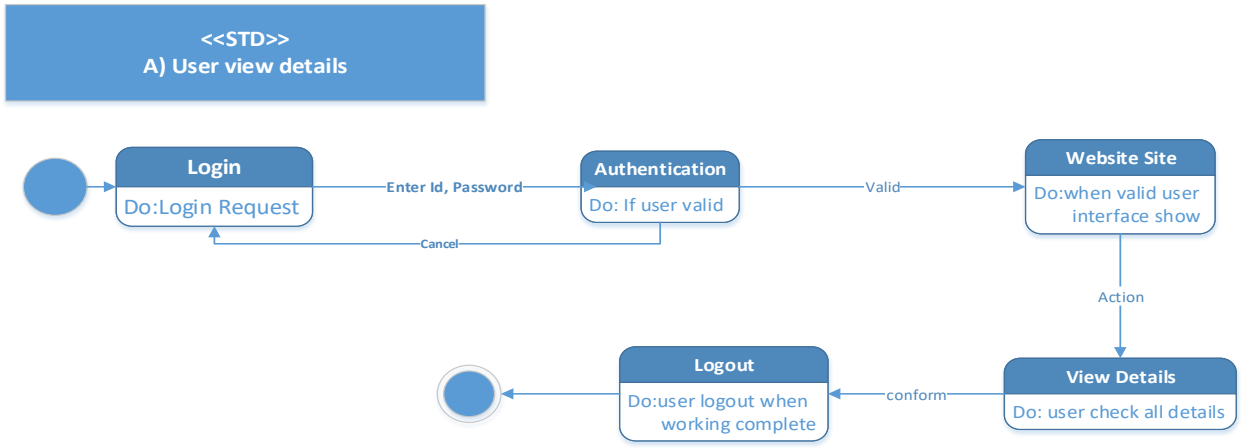


Figure 21-STD STD View details

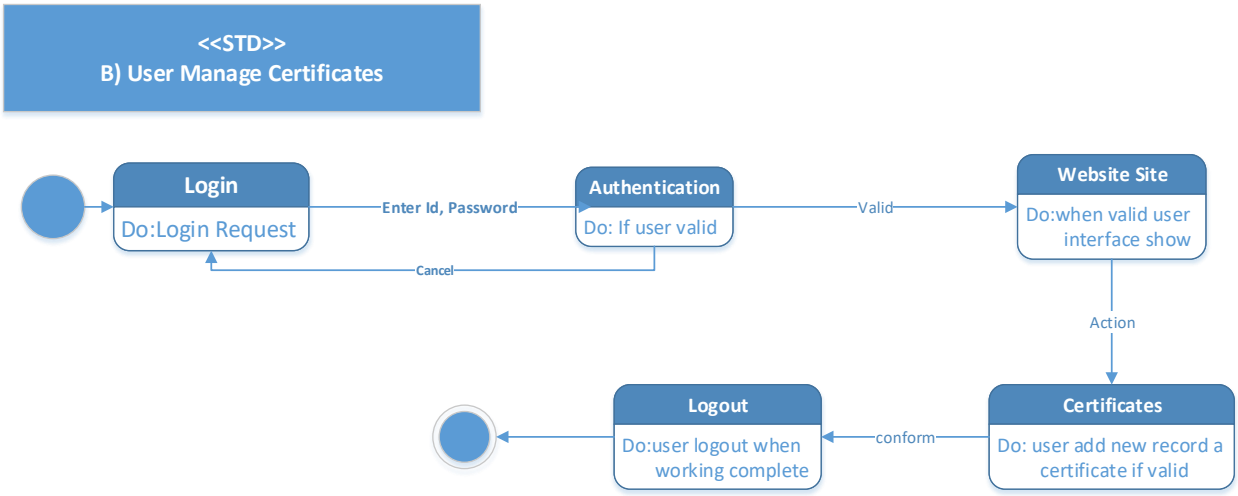


Figure 22-STD Auth

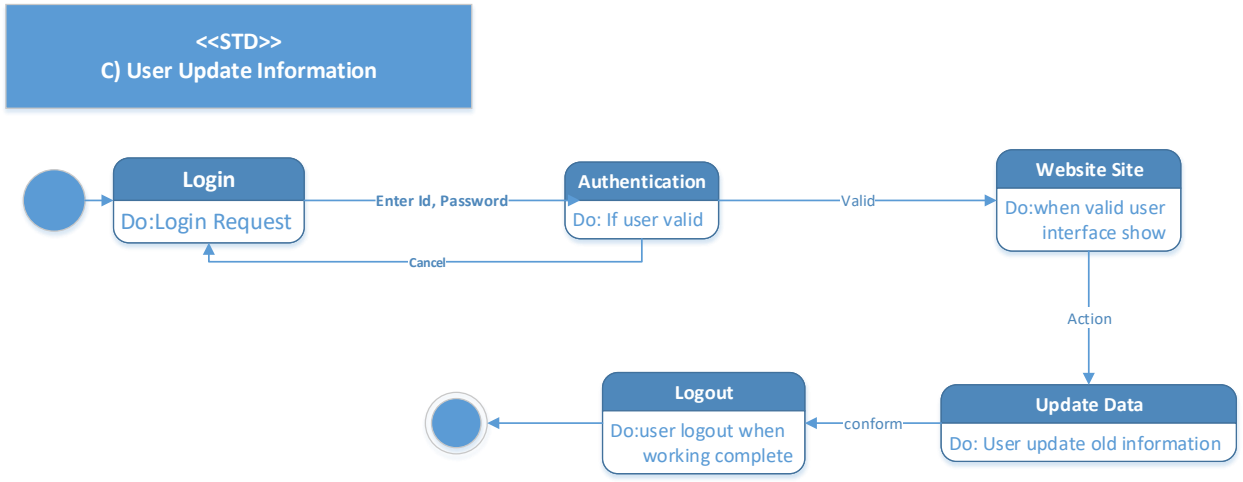


Figure 23-STD Update info

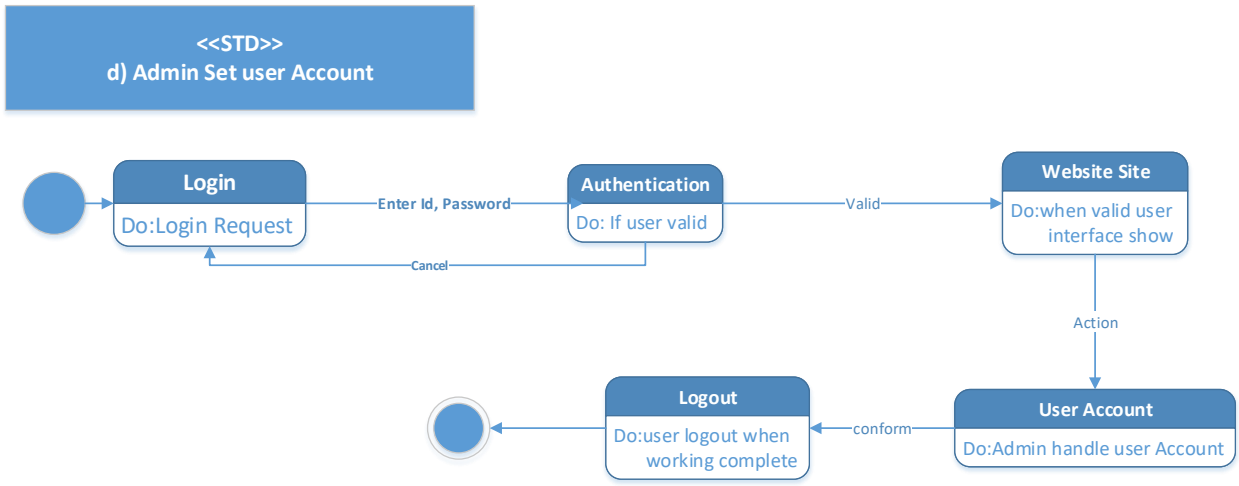


Figure 24-STD Manage Account

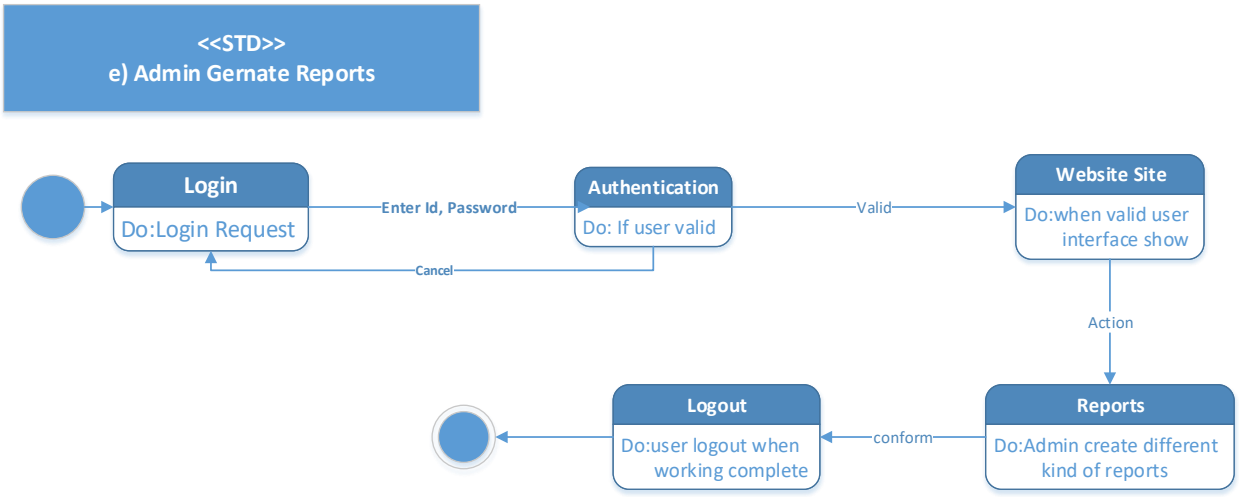


Figure 25-STD Manage report

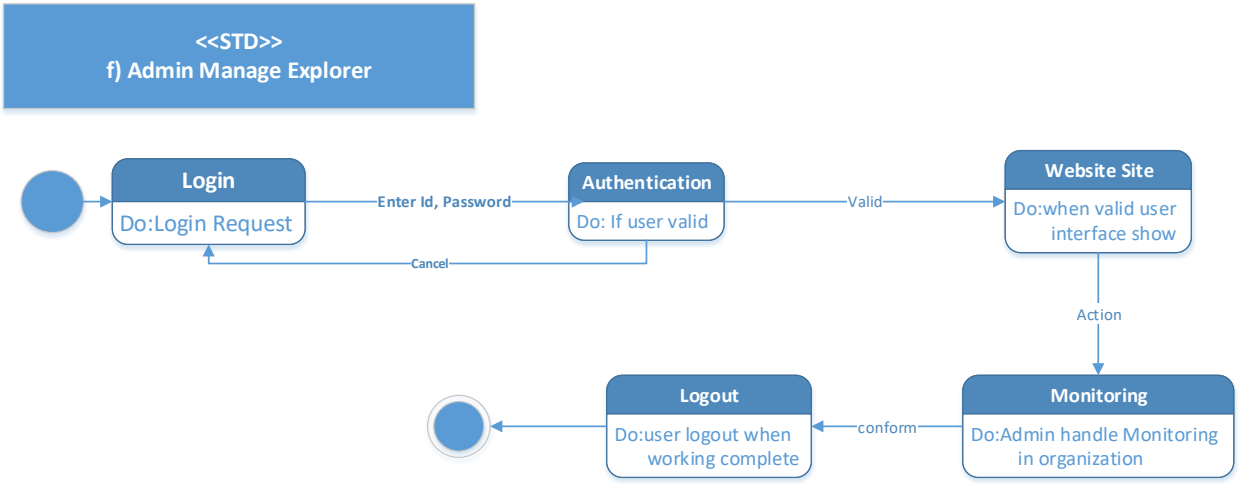


Figure 26-STD Manage Explorer

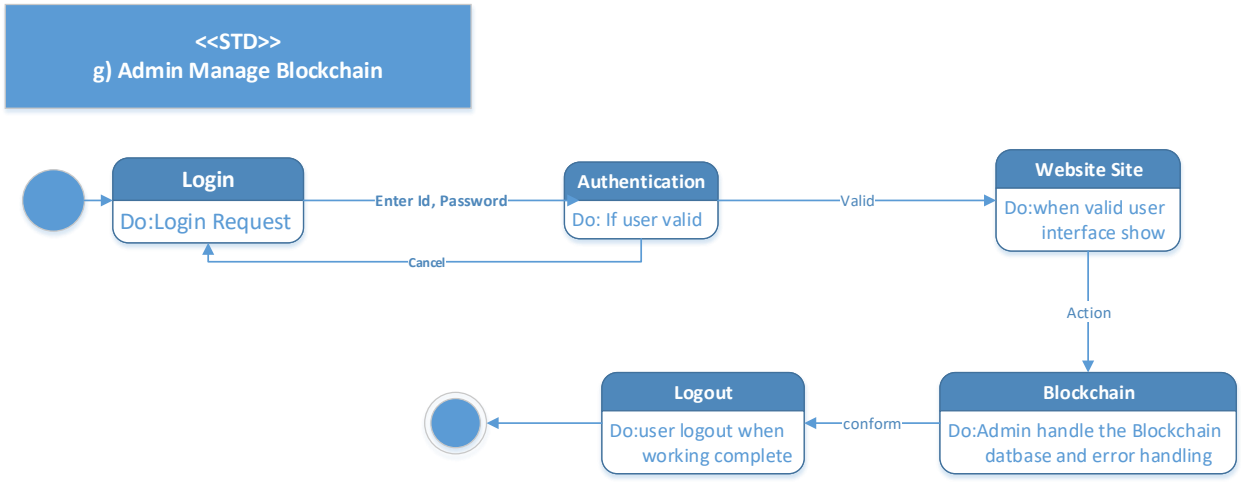


Figure 27-STD Manage blockchain

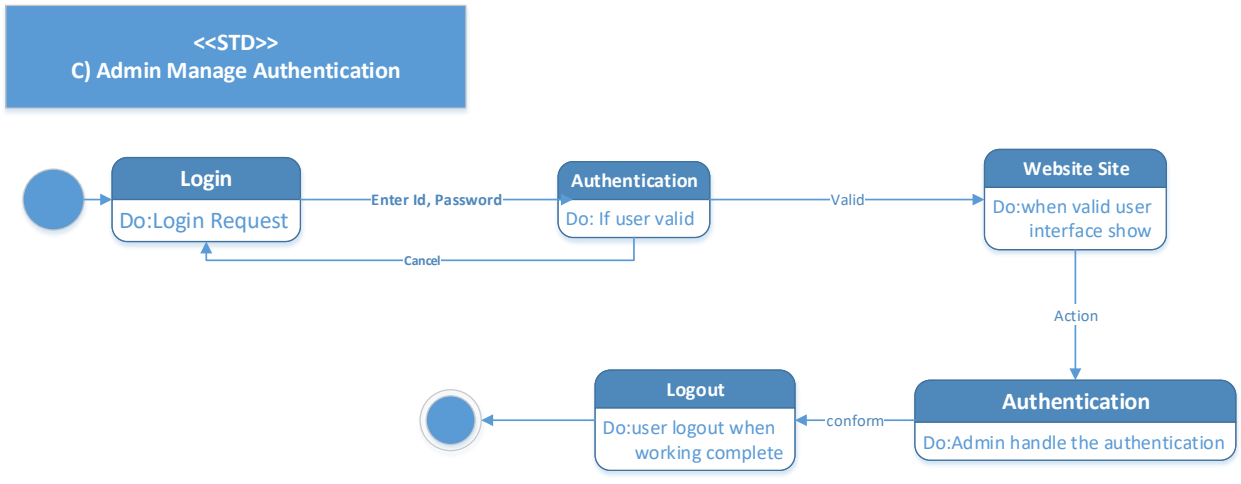


Figure 28-STD Manage Admin

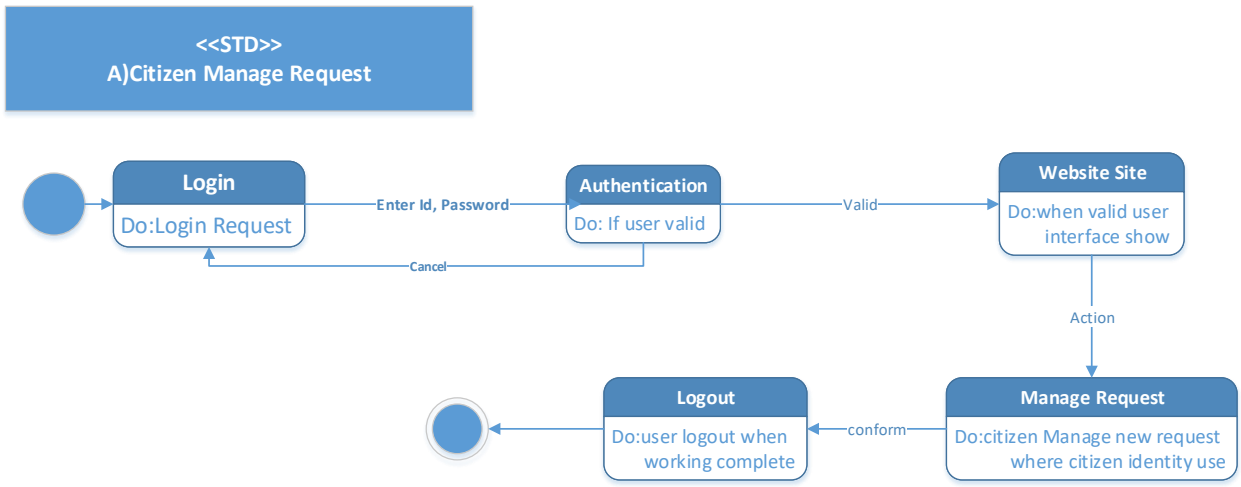


Figure 29-STD Manage Request

4.9. Component Diagram

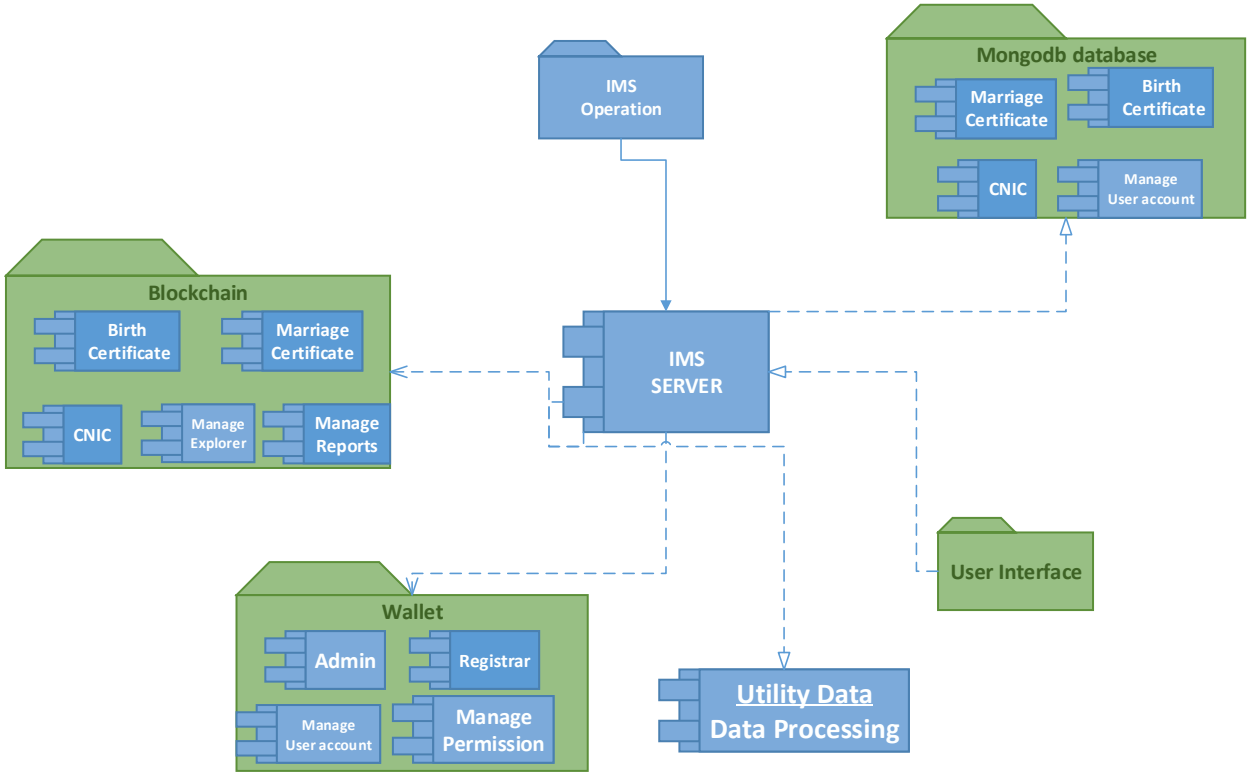


Figure 30-Component Diagram

4.10. Deployment Diagram

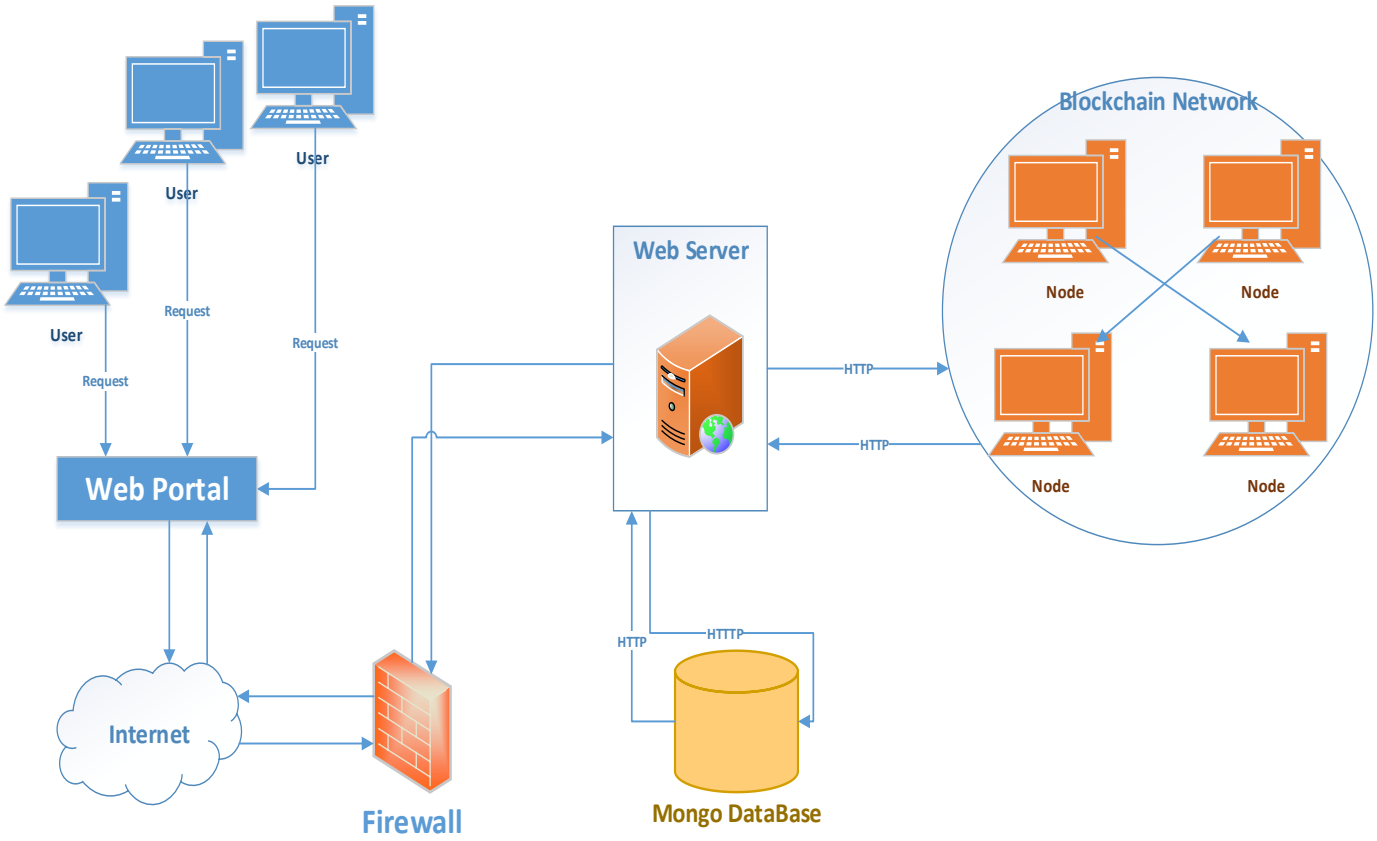


Figure 31-Deployment Diagram

4.11. Data Flow diagram

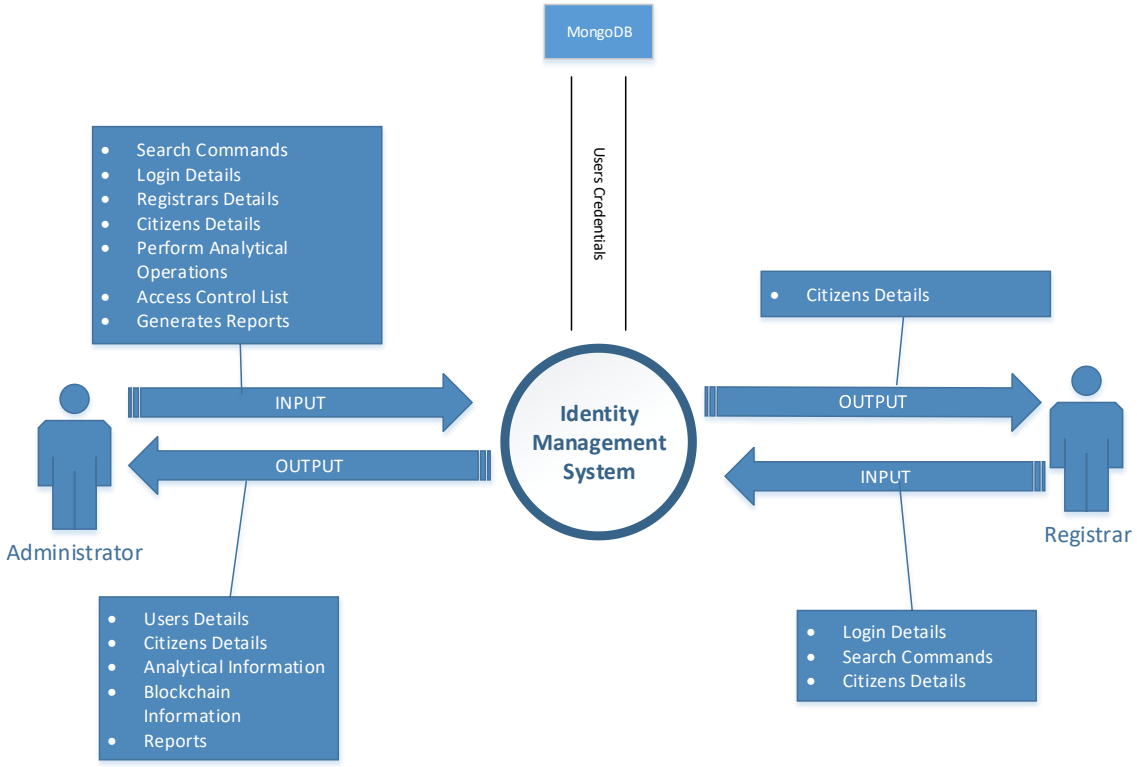


Figure 32-DFD

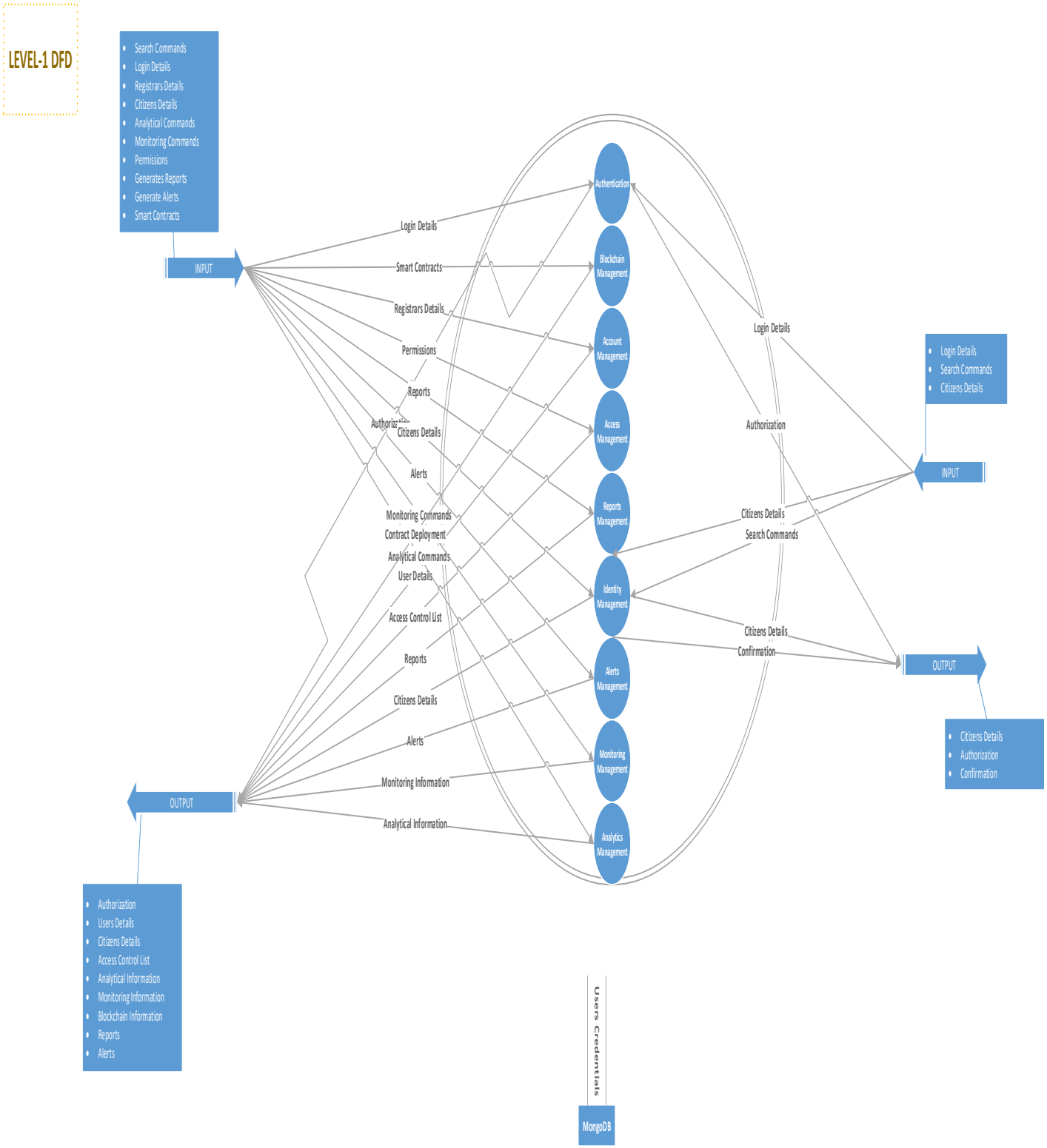


Figure 33-DFD level 1

Chapter 5

Implementation

Chapter 5: Implementation

After defining the whole System design, we are going to see how we will implement it, which types of tools & techniques that we are using to make it, which web servers are required for our website & which type of standards we should follow while coding.

5.1. Important Flow Control/Pseudo codes

The diagram will show the all system flow for example the user (user ,registrar, admin) will used the system the registrar enter the new record than it have a right to perform the task. The admin also enter the record and the admin view the block chain network like how many block create , transaction done , channel , network , chain code. Ifthe client/user get new certificate the user fill request the admin to create new certificate. The user also view the certificate and update information for certificate. After the work done the user admin and registrar logout the website.

5.2. Components, Libraries, Web Services and stubs

Web Servers:

Ngnix

Components:

Docker Compose , Docker , Node.js (run-time environment) .

Libraries

Hyper ledger fabric node js SDK

5.3. Deployment Environment

In Deployment environment the MEAN STACK Application with private and permission blockchain network. Our website system will be hosted on cloud.

5.4. Tools and Techniques

- Angular (for building website)
- Hyperledgerfabric (for building blockchain network)
- HTML/CSS + Angular Material (for Front End Designing)
- NodeJS (Back End)
- Mongo DB
- Docker

5.5. Best Practices / Coding Standards

- Communicate with all project stakeholders
- Create a risk response team
- Use a detailed work definition document
- Create a detailed work plan
- Document everything
- Ask for feedback
- Communicate the impact of project add-ons
- Manage new agreements
- Hold a wrap-up meeting

5.6. Version Control

Task #	Deliverable	Activity #	Duration (# of Days)	Responsible Team Member(s) & Role(s)
1	Web Designing & Coding	1	4 Months	2- Front End Designing & coding, Hamza, Babar sohial
2	Blockchain network	2	4Month	1- Manage whole Blockchain Network, Ali hamza, Hamza, Babar
3	NodeJS (back End)/Mongo DB	3	4Month	Manage whole NodeJS back End, Ali hamza, Hamza, Babar

Table 16-Version Control

Chapter 6

Testing and Evaluation

Chapter 6: Testing and Evaluation

Software testing is very important because of the many reasons, Softwaretesting is required to point out the defects and errors that were made during the development phases. It's essential since it makes sure of the Customer's reliability and their satisfaction in the application. It is

very important to ensure the Quality of the product. Quality product delivered to the customers helps in gaining their confidence.Evaluation can help you identify areas for improvement and ultimately help you realize your goals more efficiently.

6.1. Use Case Testing

Use Case Testing is a functional black box testing technique that helps testers to identify test scenarios that exercise the whole system on each transaction basis from start to finish.

Use Case Test 1.0

Use Case Name	**Authentication(Register Account)**
Use Case Description	User registration for using the system
Actors	Admin, User
Pre-Condition	System must be connected to network
Post-Condition	User register with system after registration successfully to the system

Main Scenario	**Serial Number**	**Steps**
Actor System	**1**	User should be registering himself via username, Gmail, Phone, and Password.
	2	System Validate all the input fields.
	3	Registered with system
Extension	**2a**	If any input field was invalid then system generates error.

Use Case Test 1.1

Use Case Name	**Log in**
Use Case Description	A user login to access the system features.
Actors	Admin, User
Pre-Condition	System must be connected to network
Post-Condition	After login user can login to system dashboard.

Main Scenario	**Serial Number**	**Steps**
Actor System	**1**	Enter email and password
	2	System validate email and password
	3	Allow access to system dashboard
Extension	**2a**	Invalid password and username Then system should show error message

Use Case Test 1.3

Use Case Name	**Logout**
Use Case Description	A user logout to the system .
Actors	Admin, User
Pre-Condition	System must be connected to network
Post-Condition	After logout user will go on login page of the system .

Main Scenario	Serial Number	Steps
Actor System	**1**	Click the logout button
	2	The system will allow to go the login page
Extension	**2a**	If a network connection error then show the massage check the internet connection

Use Case Test 2.1

Use Case Name	User Activities(create certificate)
Use Case Description	The user will send request for create the new certificate
Actors	User
Pre-Condition	user must be login to system
Post-Condition	The user will successfully send request for create the new certificate

Main Scenario	Serial Number	Steps
Actor System	**1**	Full Name, Gender, Address, DOB, Phone, Email.
	2	System validate the input fields
	3	System register the user for any request
Extension	**2a**	Invalid input fields System show error message

Use Case Test 2.2

Use Case Name	**User Activities(view certificate)**
Use Case Description	The user will send request for view the exiting certificate
Actors	User
Pre-Condition	user must be login to system
Post-Condition	The user will successfully send request for view certificate

Main Scenario	**Serial Number**	**Steps**
Actor System	**1**	Actor send the request data of certificate view
	2	System validate the input fields
	3	System register the user for request
Extension	**2a**	Invalid input fields System show error message

Use Case Test 2.3

Use Case Name	**User Activities(update certificate)**
Use Case Description	The user will send request for update data in the exiting certificate
Actors	User
Pre-Condition	user must be login to system
Post-Condition	The user will successfully send request for update data in exiting certificate

Main Scenario	**Serial Number**	**Steps**

Actor System	**1**	Actor send the request data of certificate update
	2	System validate the input fields
	3	System register the user data for updating the certificate
Extension	**2a**	Invalid input fields System show error message

Use Case Test 3.1

Use Case Name	**Manage Identity Asset(create Birth certificate)**
Use Case Description	The admin will create certificate for create from user record
Actors	Admin
Pre-Condition	admin must be login to system
Post-Condition	The Admin will successfully create certificate from the exiting record

Main Scenario	**Serial Number**	**Steps**
Actor System	**1**	Actor send the message of certificate is create
	2	System validate the input fields
	3	System register the user data for creating the certificate
Extension	**2a**	Invalid input fields System show error message

Use Case Test 3.2

Use Case Name	**Manage Identity Asset(create CNIC certificate)**
Use Case Description	The admin will create certificate for create from user record
Actors	Admin
Pre-Condition	admin must be login to system
Post-Condition	The Admin will successfully create certificate from the exiting record

Main Scenario	**Serial Number**	**Steps**
Actor System	**1**	Actor send the message of certificate is create
	2	System validate the input fields
	3	System register the user data for creating the certificate
Extension	**2a**	Invalid input fields System show error message

Use Case Test 3.3

Use Case Name	**Manage Identity Asset(create marrige certificate)**
Use Case Description	The admin will create certificate for create from user record
Actors	Admin
Pre-Condition	admin must be login to system
Post-Condition	The Admin will successfully create certificate from the exiting record

Main Scenario	Serial Number	Steps
Actor System	**1**	Actor send the message of certificate is create
	2	System validate the input fields
	3	System register the user data for creating the certificate
Extension	**2a**	Invalid input fields System show error message

Use Case Test 4.0

Use Case Name	Manage Blockchain
Use Case Description	The admin will create chain-code for deploy the chain-code and approve the transition
Actors	Admin
Pre-Condition	admin must be login to system
Post-Condition	The Admin will successfully create block and peer node

Main Scenario	Serial Number	Steps
Actor System	**1**	Actor manage the blockchain
	2	System validate the input fields
	3	System register the user data in blockchain
Extension	**2a**	Invalid input fields System show error message

Use Case Test 5.0

Use Case Name	**Manage Report**
Use Case Description	The admin will create report and view report and delete report
Actors	Admin
Pre-Condition	admin must be login to system
Post-Condition	The Admin will successfully create report and view report

Main Scenario	**Serial Number**	**Steps**
Actor System	**1**	Actor manage the all report
	2	System validate the input fields
	3	System register the user data in report
Extension	**2a**	Invalid input fields System show error message

Use Case Test 5.0

Use Case Name	**Manage Explorer**
Use Case Description	The admin will check the node details and transitions details
Actors	Admin
Pre-Condition	admin must be login to system
Post-Condition	The Admin will check all operation of explorer

Main Scenario	**Serial Number**	**Steps**

Actor System	**1**	Actor manage the explorer
	2	System validate the input fields
	3	System register the user data in reports
Extension	**2a**	Invalid input fields System show error message

6.2. Equivalence partitioning

Equivalence partitioning is a Test Case Design Technique to divide the input data of software into different equivalence data classes. Test cases are designed for equivalence data class.

A use of this method reduces the time necessary for testing software using less and effective test cases, while validating Sign up form we've set values like enter the phone number:

- A text field permits only numeric characters (e.g. phone number)
- Length must be 12-14 characters long (e.g. phone number length)

In phone number length numbers from 12 to 14 are equivalent and are valid & numbers from 14 or above all are invalid. And numbers from 0-11 are invalid.

Test Scenario	Test Scenario Description	Expected Outcome
1	Enter 0 to 12 characters	System should not accept
2	Enter 12-14 characters	System should accept
3	Enter 14-20 characters	System should not accept

Table 17-Equivalence partitioning

6.3. Boundary value analysis

Boundary value analysis is a test case design technique to test boundary value between partitions (both valid boundary partition and invalid boundary partition).

A boundary value is an input or output value on the border of an equivalence partition, includes minimum and maximum values at inside and outside boundaries.While validating Sign up form we've set values:

- Email should only accept Gmail, Yahoo, and Hotmail.
- Password should be with a valid length of 10-20.

In test case 9, 10, 11 are the minimum values which is Boundary Values. And 19, 20, 21 are maximum values of Boundary Values.

Test Scenario Description	**Expected Outcome**
Boundary Value = 9	System should NOT accept
Boundary Value = 10	System should accept
Boundary Value = 11	System should accept
Boundary Value = 19	System should accept
Boundary Value = 20	System should accept
Boundary Value = 21	System should NOT accept

Table 18-Boundary value analysis

6.4. Data flow testing

Data flow testing is a family of test strategies based on selecting paths through the program's control flow in order to explore sequences of events related to the status of variables or data objects.

Data Flow testing is very important that can help us to pinpoint many issues:

- A variable that is declared but never used within the program.
- A variable that is used but never declared.
- A variable that is defined multiple times before it is used.

- Deallocating a variable before it is used.

So, while data flow testing some undeclared variables shouldn't be the part of our project. It is better to be removed.

6.5. Unit testing

The unit testing was done after the coding phase. The purpose of the unit testing was to locate errors in the current module, independent of the other modules. Some changes in the coding were done during the testing phase. Finally, all the modules were individually tested following bottom to top approach, starting with smallest and lowest modules and then testing one at a time.

Testing

- Checked all the validations on each field.
- Wrong inputs in the fields of the forms.
- Blockchain testing.
- Validation of HTML, CSS. Checked error e.g. syntax error.

Benefits of Unit Testing:

- Codes are more reusable. In order to make unit testing possible, codes need to be modular. This means that codes are easier to reuse.
- Unit testing increases confidence in changing/ maintaining code.

6.6. Integration testing

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing was to expose faults in the interaction between integrated units.Integration testing is an important part of the testing cycle as it makes it easier to find the defect when two or more modules are integrated.

Integration Test case:

Verifying the interface link between the login page and the home page i.e. when a user enters the credentials and logs it should be directed to the homepage.

6.7. Performance testing

Performance testing is the process of determining the speed or effectiveness of a computer, network, software program or device.

Software Performance testing is type of testing perform to determine the performance of system to major the measure, validate or verify quality attributes of the system like responsiveness, Speed, Scalability, Stability under variety of load conditions. It is very important to interact with the system.

- **Processor Usage -** amount of time processor spends executing non-idle threads.
- **Memory use -** amount of physical memory available to processes on a computer.
- **Disk time -** amount of time disk is busy executing a read or write request.
- **Bandwidth -** shows the bits per second used by a network interface.
- **Private bytes -** number of bytes a process has allocated that can't be shared amongst other processes. These are used to measure memory leaks and usage.
- **Committed memory -** amount of virtual memory used.
- **Memory pages/second -** number of pages written to or read from the disk in order to resolve hard page faults. Hard page faults are when code not from the current working set is called up from elsewhere and retrieved from a disk.
- **Page faults/second -** the overall rate in which fault pages are processed by the processor. This again occurs when a process requires code from outside its working set.
- **CPU interrupts per second -** is the avg. number of hardware interrupts a processor is receiving and processing each second.
- **Network bytes total per second -** rate which bytes are sent and received on the interface including framing characters.
- **Response time -** time from when a user enters a request until the first character of the response is received.
- **Throughput -** rate a computer or network receives requests per second.

- **Amount of connection pooling -** the number of user requests that are met by pooled connections. The more requests met by connections in the pool, the better the performance will be.
- **Maximum active sessions -** the maximum number of sessions that can be active at once.
- **Hit ratios -** This has to do with the number of SQL statements that are handled by cached data instead of expensive I/O operations. This is a good place to start for solving bottlenecking issues.
- **Hits per second -** the no. of hits on a web server during each second of a load test.
- **Rollback segment -** the amount of data that can rollback at any point in time.
- **Database locks -** locking of tables and databases needs to be monitored and carefully tuned.
- **Top waits -** are monitored to determine what wait times can be cut down when dealing with the how fast data is retrieved from memory
- **Thread counts -** An applications health can be measured by the no. of threads that are running and currently active.
- **Garbage collection -** It has to do with returning unused memory back to the system. Garbage collection needs to be monitored for efficiency.

Performance testing tool

Web Load: Web Load is a pioneer and leader in load testing, providing rich capabilities for managing large-scale performance tests in complex enterprise environments.

6.8. Stress Testing

Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions.

The goal of stress testing is to analyze the behavior of the system after failure. For stress testing to be successful, system should display appropriate error message while it is under extreme conditions.

Tool for Stress Testing

Stress Tester: **This tool provides extensive analysis of the web application performance, provides results in graphical format, and it is extremely easy to use. No high-level scripting is required and gives good return on investment.**

Chapter 7

Summary, Conclusion and Future Enhancements

Chapter 7: Summary, Conclusion & Future Enhancements

7.1. Project Summary

Our project is to build a tempered-proof Identity Management System (IMS) like NADRA on the stack of new emerging Blockchain Technology which use cryptographic techniques and tools in order to provide Private and Permissioned system which delivers a high degree of confidentiality, provisioning ,resiliency (the capacity to recover quickly from difficulties), flexibility, and scalability (the capacity to be changed in size or scale).IMS used to issue unique identities to the citizens and their families so that they can be identified uniquely and authenticated in a large population and government can keep track of them with their family relations for the welfare of the country and for different analyzing purposes. We can restrict users from gaining unauthorized access to the government organization data by specifying their access control list in Blockchain, so an unauthorized person could not get access illegally to government assets

7.2. Achievements and Improvements

- Learnt to work in a team.
- Learnt time management.
- Learnt how to survive in a competitive business environment.
- We will try to improve & take care of our project, we will try tomake it more secure, maintain its availability and it will be more responsive at a time.

7.3. Critical Review

We are working on the base of new technology that why no competitor is available in market. It's a big challenge for us introduce new technology to and convince to use this technology

7.4. Lessons Learnt

Learnt how to make a successful project with absolutely no chances of failure& learnt how to survive in a market.

7.5. Future Enhancements/Recommendations

- Expanding its operation in other major organizations of Pakistan it is our main goal.
- We will make android app where our customers will ableto see its identity card related data from our android app.

Reference and Bibliography

Reference and Bibliography

[1] 5 Dec- 2018, 12:33 PM
https://fabric-sdk-node.github.io/release-1.4/index.html\

[2] 10 January-2019, 2:33PM
https://hyperledger-fabric.readthedocs.io/en/release-1.4/

[3] 2 February - 2019, 2:56 PM
https://fabric-shim.github.io/release-1.4/index.html

[4] 23 November- 2018 3:10 PM
http://www.coindesk.com/state-of-blockchain-q1-2016/

[5] 19 October- 201810:00 AM
https://material.angular.io/

[6] 19 October- 20189:00 PM

https://angular.io/docs

[7] 23 September- 2018 11:23 PM
https://docs.mongodb.com/

[8] 19 October- 2018 9:00 PM

https://docs.docker.com/install/linux/docker-ce/ubuntu/

[9] 11 March- 2019 3:10 PM

https://devdocs.io/javascript/

[10] 19 January- 2019 6:50 PM

https://www.ibm.com/blockchain/platform\

[11] 24 September- 2018 12:00 PM

https://github.com/hyperledger/fabric-samples

IDENTITY MANAGEMENT SYSTEM

Final Year Project

Session 2015-2019

A project submitted in partial fulfillment of the degree of

BS in Computer Science

Department of Computer Science

Faculty of Computer Science & Information Technology

The Superior College , Lahore

Spring 2019

Type (Nature of project)	[✓] **D**evelopment [ ] **R**esearch [ ] **R**&**D**
Area of specialization	

Project Group Members

Sr.#	Reg. #	Student Name	Email ID	*Signature
(i)	**BCSM-F15-106**	**Ali Hamza**	alihamza2313@gmail.com	
(ii)	**BCSM-F15-120**	**Hamza**	hamzaarifigi7@gmail.com	
(iii)	**BCSM-F15-113**	**Babar Sohail**	babarsohail04@gmail.com	

*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

Plagiarism Free Certificate

This is to certify that, I **Ali Hamza** S/D of **AmanUllah** group leader of FYP under registration no **BCSM-F15-106** at Computer Science Department, The Superior College, Lahore. I declare that my FYP proposal is checked by my supervisor and the similarity index that is less than 20%, an acceptable limit by HEC. Report is attached herewith as Appendix D.

Date: 22-08-2019 Name of Group Leader: Ali Hamza Signature: _____

Name of Supervisor: Mr. Muhammad Ahmed
Designation: Lecturer

Co-Supervisor:
1 Designation: Associate Professor

Signature: _____

Signature: _____

HoD:

Signature: _____