APPLYING FEDERATED LEARNING WITH SMART CONTRACTS IN HEALTHCARE

PROJECT SUPERVISOR

Dr. Farrukh Shahid

PROJECT CO-SUPERVISOR

Mr. Shahbaz Siddiqui

PROJECT TEAM

Sarmad Jamal	K191116
Khizer Jilani	K191057
Mansoor Butt	K191114

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering.

FAST SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

KARACHI CAMPUS

July 2023

Project Supervisor	Dr. Farrukh Shahid
Project Team	Sarmad Jamal K19-1116 Khizer Jilani K19-1057 Mansoor Butt K19-1114
Submission Date	May 2, 2023

Mr. Supervisor Name _____

Supervisor

Mr. Co-Supervisor Name _____

Co-Supervisor

Dr. Abdul Aziz _____

Head of Department

FAST SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

KARACHI CAMPUS

Acknowledgement

We would like to express our deep appreciation to our supervisor, Dr. Farrukh Shahid, whose invaluable support, constructive criticism, and unwavering encouragement were instrumental in the development and refinement of this project. His expert guidance and insights proved to be an invaluable asset, and we could not have completed this project without his guidance.

We also extend our sincere gratitude to our co Supervisor Mr. Shahbaz Siddiqui, particularly for their guidance and mentorship throughout the project. Their expertise and support helped us navigate various challenges and develop an optimized, efficient solution that met our project's objectives.

Furthermore, we would like to acknowledge the financial and emotional support of our parents. Their unwavering belief in our abilities and moral support helped us remain focused and motivated throughout the project's duration.

In conclusion, we express our heartfelt appreciation to all those who contributed to this project's success, and we hope that this work will prove to be a valuable contribution to the academic community.

Abstract

This FYP project aims to address the problem of the lack of medical data sharing in healthcare, which is a major obstacle in integrating artificial intelligence (AI) technology. The reason behind the data-sharing hurdle is that medical data is highly confidential, and hospitals are reluctant to share it due to privacy concerns. As a result, machine learning (ML) models cannot be trained, and progress in this field is stalled. To overcome this issue, this project proposes an alternative approach, which involves model sharing and federated learning. The proposed system consists of two entities, the Super user and hospitals. The Super user requests hospitals to participate in federated learning, and registered hospitals are granted access to global model files through IPFS hashes. These hashes are stored on the block chain via smart contracts. The technologies used in this project include Solidity for smart contract development, React.js for the frontend, and IPFS Piñata for hosting ML model files. This solution provides a viable and secure way to enable AI integration in healthcare while maintaining patient privacy.

Table of Contents

List of Figure

List of Tables

1 Introduction

Block chain is also used in the healthcare management system for effective maintenance of electronic health and medical records. The technology ensures security, privacy, and immutability. This work proposes a framework by integrating the block chain and Federated Deep Learning in order to provide a tailored recommendation system. Electronic medical records contain personal and confidential information that traditional storage methods must protect against cyberattacks and third-party authentication. To overcome this challenge, a method of distributed storage was proposed in this work. The focus of this work is also on making treatment recommendations to patients by comparing their medical records with historical data. This work is motivated by the limitations of existing work. The current study supports his EHR preservation, but the recommendation system is neither discussed nor incorporated, making it difficult to create a treatment recommendation system. Federated learning may therefore lead to more accurate treatment recommendations

1.1 Project Scope

Primarily, the scope of our final year project is limited. We will be limiting our scope to the medical sectors where a patient data will be recorded on our system and by means of recording, we will be using that data only to train our machine learning model. We will be using the concept of federated Learning Through Federated learning; multiple organizations or institutions work together to solve a machine-learning problem under the coordination of a central server or service provide. Thus, a deep-learning model is maintained and improved upon within a central server. The model is trained by distributing itself to hospitals which allows these sites to keep their data localized. Data from each collaborator is never exchanged or transferred during training. Instead of bringing the data to the central server, as in conventional deep learning, the central server maintains a global shared model, which is disseminated to all institutions. Each entity subsequently maintains a separate model based on its own patients' data. Thereafter, each center provides feedback to the server based on its individually trained model—either by its weight or the error gradient of the model. The central server aggregates the feedback from all participants, and based on predefined criteria, updates the global model. The predefined criteria allow the model to evaluate the quality of the feedback and therefore to only incorporate that which is value-adding. The feedback from centers with adverse or strange results can thus be ignored. This process forms one round of federated learning, and it is iterated until the global model is trained.

1.2 Not in Scope

We will be only targeting one/two disease for the recommendation and it will be running on local environment not globally deployed. It will be on private block chain and only people who are connected will have the feasibility to connect to it

1.3 Problem Statement

Recent advances in deep learning have shown many successful stories in smart healthcare applications with data-driven insight into improving clinical institutions' quality of care. Excellent deep learning models are heavily data-driven. The more data trained, the more robust and more generalizable the performance of the deep learning model. However, pooling the medical data into centralized storage to train a robust deep learning model faces privacy, ownership, and strict regulation challenges. **Federated learning** resolves the previous challenges with a shared global deep learning model using a central aggregator server. At the same time, patient data remain with the local party, maintaining data anonymity and security.

This method provides decentralized machine learning model training with-out transmitting medical data through a coordinated central aggregate server. Medical institutions, working as client nodes, train their deep learning models locally and then periodically forward them to the aggregate server. The central server coordinates and aggregates the local models from each node to create a global model, then distributes the global model to all the other nodes. It is worth noting that the training data are kept private to each node and never transmitted during the training process. Only the model's weight and parameters are transmitted, ensuring that medical data remain confidential. For these reasons, FL mitigates many security concerns because it retains sensitive and private data while enabling multiple medical institutions to work together. FL holds an excellent promise in healthcare applications to improve medical services for both institutions and patients—for instance, predict autism spectrum disorder, mortality and intensive care unit (ICU) stay-time prediction.

1.4 Project Objectives

The objective of this project is to develop a secure and efficient system for medical data sharing in healthcare using federated learning and model sharing techniques. The primary goal is to overcome the challenges associated with privacy concerns and facilitate the integration of artificial intelligence (AI) technology in the healthcare sector. By establishing a collaborative framework between the Super user and hospitals, the project aims to enable the training and sharing of machine learning (ML) models while ensuring the confidentiality of patient data. The project will leverage technologies such as Solidity, React.js, IPFS, and blockchain to create a robust and scalable solution that promotes AI integration while maintaining patient privacy

2 Requirement Analysis

2.1 Functional Requirements

2.1.1 Functional Hierarchy

The super Admin and providers have a specific identity number that identifies their account. The account is accessed via a special account address of web3 provider such as meta mask and for another way of authentication we will use firebase to authenticate the data

As data will be generated from multiple organizations, data must be accessible and consistent at all times. The integrity of the data must not be compromised.

Features

We have Implemented a user-friendly application user interface using React Js technology. It provides an interactive and responsive UI with adequate performance. The navigation from one page to another is explicitly shown.

The project included the following features with respect to super admin and providers (Healthcare institutions).

Login: Login page for super Admin and provider

Signup: Signup page for provider

functionalities related to provider

View Form: Doctor can view patient available respective data

Fill form: Receptionist will submit the form of the patient

Request a particular doctor: Receptionist can request a particular doctor

Take part in training: provider will have a functionality to train its model

functionalities related to super Admin

Retrieve model: super Admin can retrieve hospital model and make it interact with the global model

Aggregation of model: super admin will aggregate the model based on the weights of the different models

2.2 Use Cases

2.2.1 Form Submission

[Use Case Diagram]

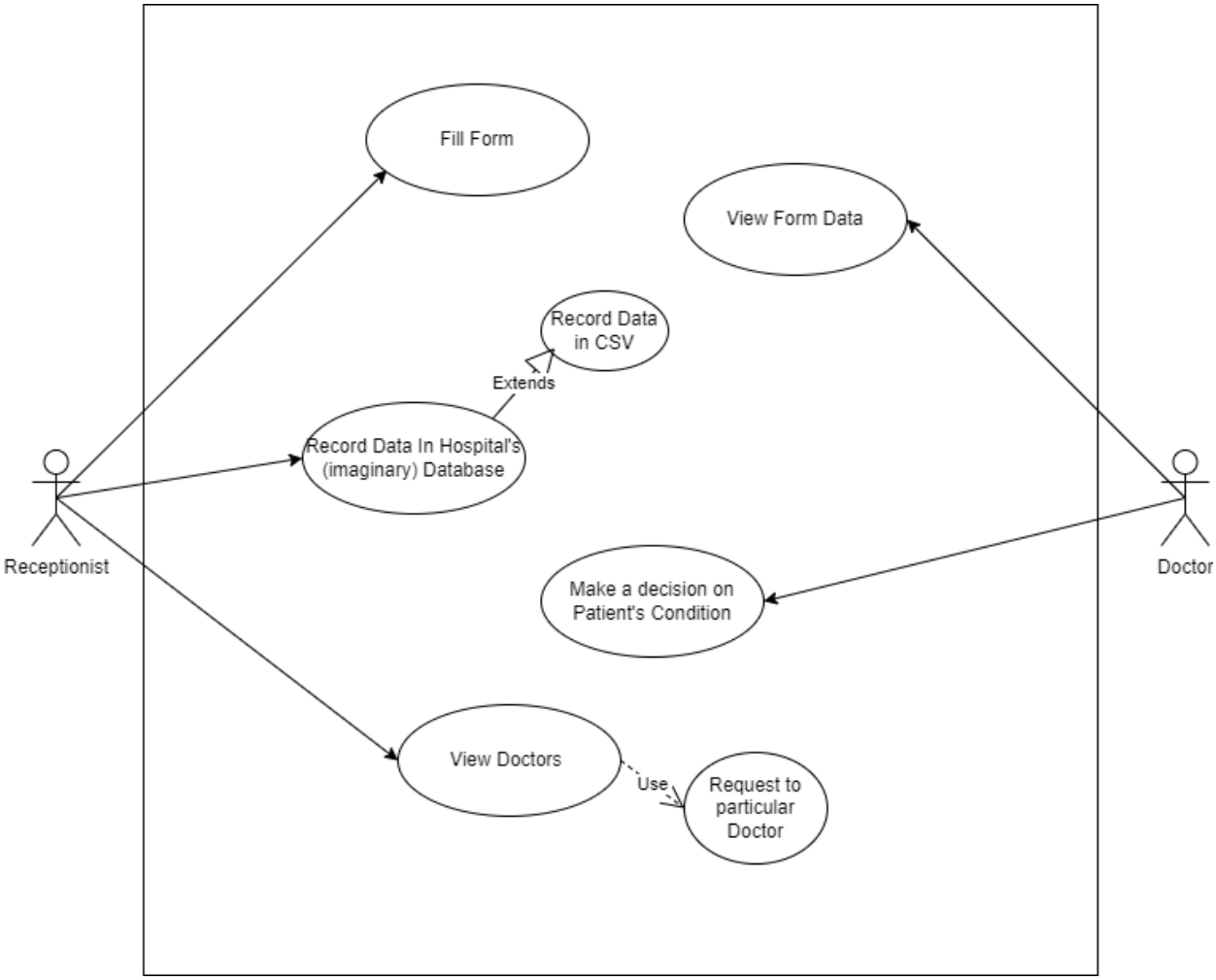


Figure 1

Form Submission		
Use case Id:	1	
Actors:	Receptionist, Doctor	
Feature:	Submission of form	
Pre-condition:	Receptionist must be logged in and have a list of doctors Doctor should be authenticated and have a verified account given by the provider	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Fill the form of the patient	Fill input fields provided on the user interface

		and submit it
2.	Doctor will make a decision on the given form data and patient condition	A form view option will be visible to the doctor and he can view the patient data on the form and will make a decision accordingly for the treatment
Alternate Scenarios:		
None		
Post Conditions		
Step#	**Description**	
1	A final decision for the patient treatment will be taken and if the patient agrees to take part in training aggregate weights will be updated on the global server	
Use Case Cross referenced	None	

Table 1

2.2.2 Hospital/Super user Use Case

Figure 2

[Use Case Description]

Model training	
Use case Id:	2
Actors:	Super User,Hospital
Feature:	Training of Model

Pre-condition:	SuperUser must be logged in and have access to global server model and have weights passed to it by the hospital	
Scenarios		
Step#	**Action**	**Software Reaction**
1.	Hospital will register itself on the portal	Hospital will get access to take part in training of model and interacting with our system
2.	Superuser can add or remove provider from the system	Hospitals with the penalty charges or the hospital which are sucessfully registered will act accordingly
Alternate Scenarios:		
None		
Post Conditions		
Step#	**Description**	
1	Trained model on the recent data will be passed to the hospital and all the nodes will be updated	
Use Case Cross referenced	1	

Table 2

2.3 Non-Functional Requirements

2.3.1 Performance Requirement

The System shall be based on web and has to be run from a web server. The system shall take initial load time depending on internet connection strength which also depends on the media from which the product is run. The performance shall depend upon hardware components of the client/customer

2.3.2 Safety Requirement

The software is completely environmentally friendly and does not cause any safety violations. The web page will have a flexible font that can be zoomed so as to not over constrain the eyes.

2.3.3 Security Requirement

Data Transfer

- The system shall use secure sockets in all transactions that include any confidential customer information.
- The system shall automatically log out all customers after a period of inactivity.
- The system shall confirm all transactions with the customer's web browser.
- The system shall not leave any cookies on the customer's computer containing the user's password.
- The system shall not leave any cookies on the customer's computer containing any of the user's confidential information.

Data Storage

- The customer's web browser shall never display a customer's password. It shall always be echoed with special characters representing typed characters.
- The customer's web browser shall never display a customer's credit card number after retrieving from the database. It shall always be shown with just the last 4 digits of the credit card number.
- The system's back-end servers shall never display a customer's password. The customer's password may be reset but never shown.
- The system's back-end servers shall only be accessible to authenticated administrators.
- The system's back-end databases shall be encrypted.

2.3.4 User Documentation

The software is accompanied by the following materials for further help:

- Software Requirement Document
- Software Design Specification Document

3 Design Details

3.1 Design Extension, Interface & Data Management

3.1.1 Future System Extension or Enhancement

The system will be built using the latest edition of framework and incase a new update occurs in technology we must be ready. Stay updated by tracking orders with customized alerts and resolve issues proactively. Track real-time, Optimize routes and schedules and resource allocation in one centralized view.

3.1.2 User Interface Paradigm

The user interface is very reliable and understandable to the user, the interface designed in such a way that mostly any expertise of people can easily learn and understand that what happens if I click this option, interface is very effective and efficient to increase the usability of users.

3.1.3 Data Management

Main Data which are models will be stored on the IPFS and the relevant hashes of the model will be stored on smart contracts. Hospital will upload the model on web3 interface then store the model on the IPFS which will be use to apply federated leaning through aggregating of all these models. Form typically collects the following data:
Patient Name
Fever
Blood Pressure
Heart Rate
Platelets
Blood Group
Weight
Age
ETC

3.2 Design Strategy

3.3 Database Design

Not applicable (Database is being used to store data temporarily for authorization and registration purpose only, there are no other relations) 3.2, 3.3, 3.4 Not Applicable

3.5 Application Design

3.5.1 Class Diagram (Not Applicable)

3.5.2 Sequence Diagram

3.5.2.1<Sequence Diagram 1>

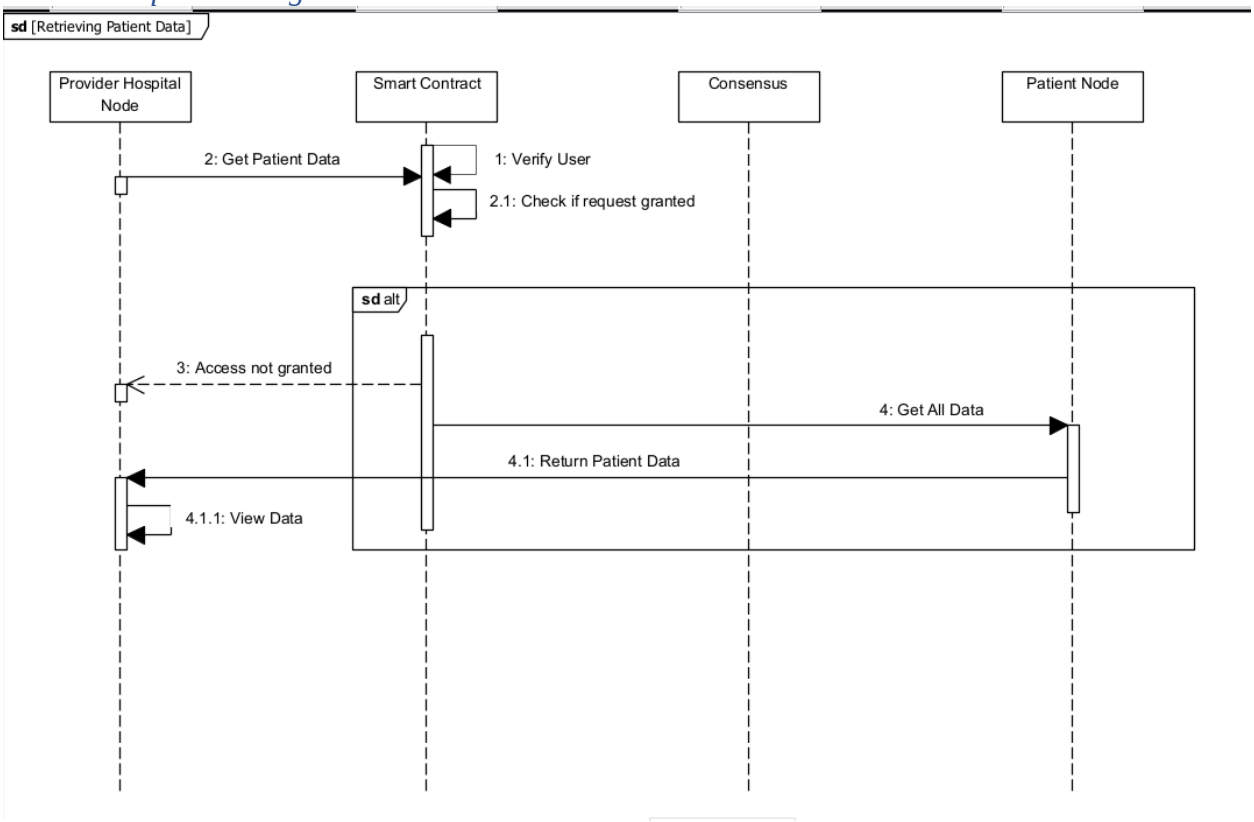


Figure 3

3.5.2.2<Sequence Diagram 2>

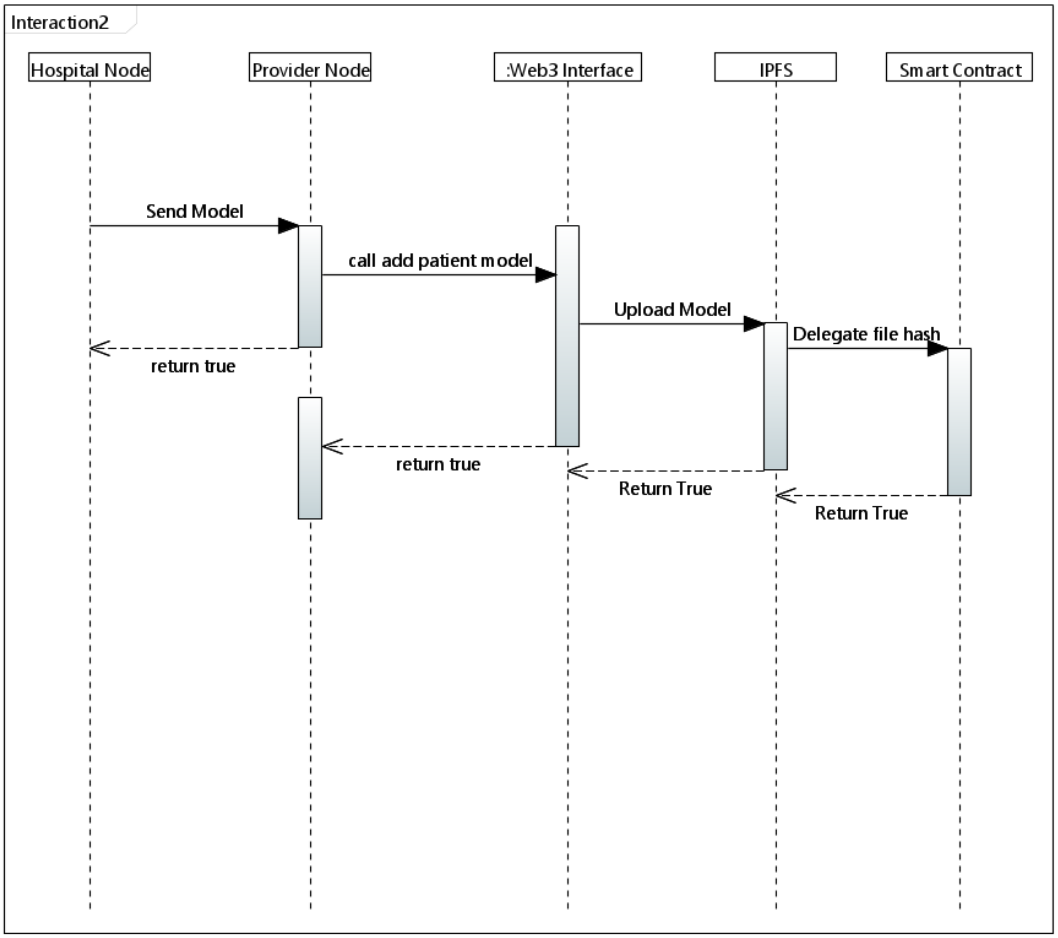


Figure 4

.

3.5.2.3 <Sequence Diagram 3>

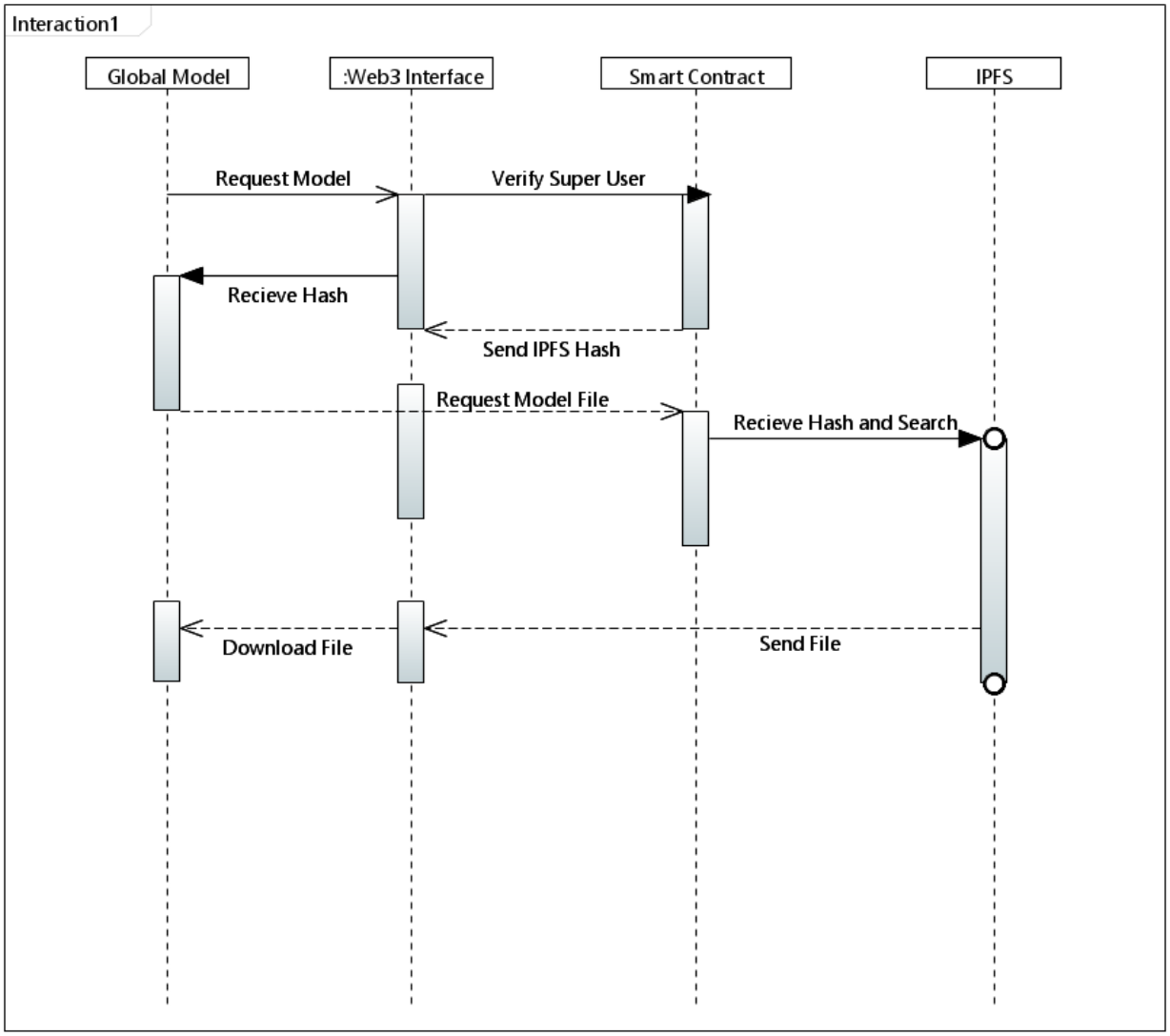


Figure 5

3.5.2.4<Sequence Diagram 4>

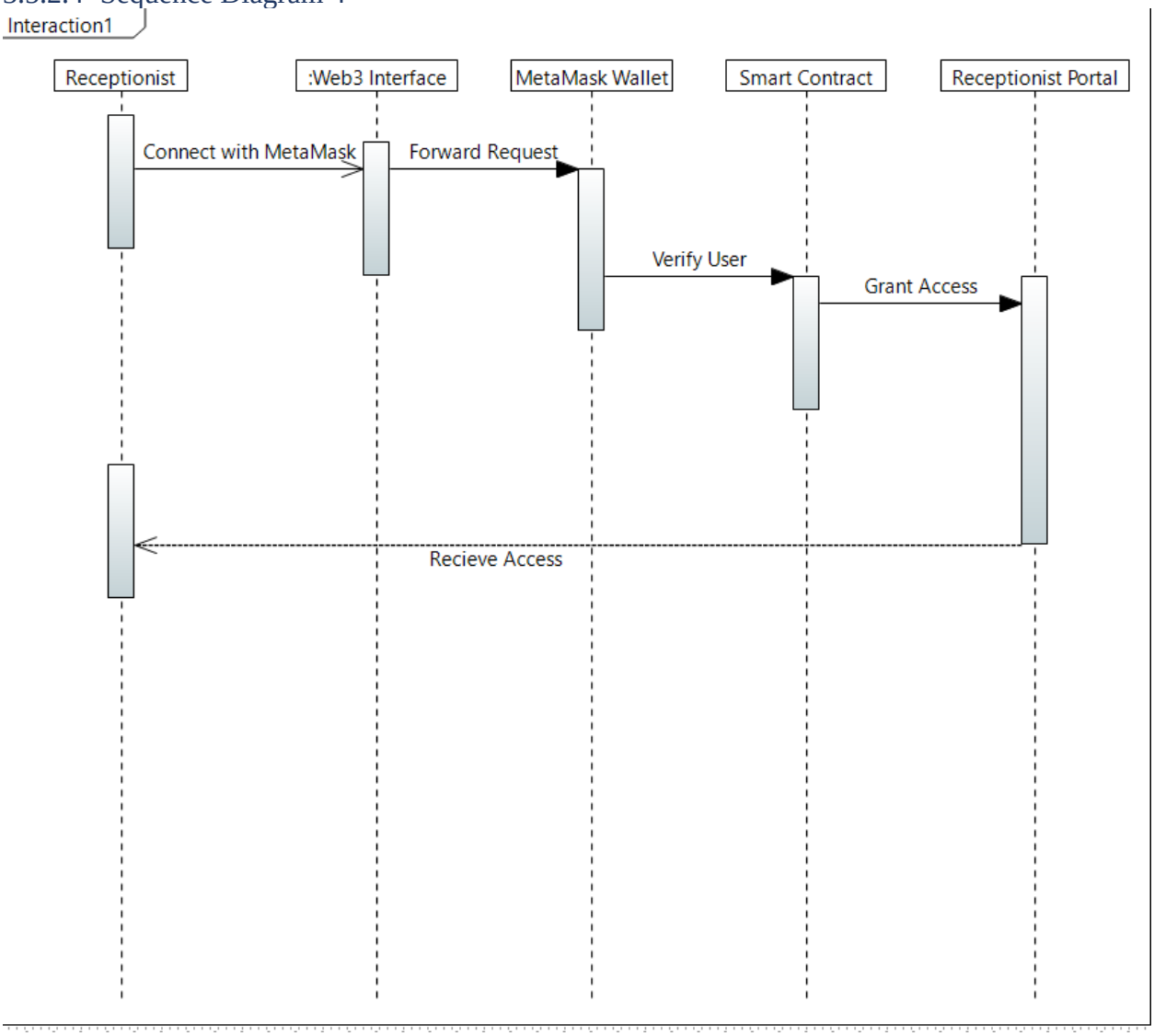
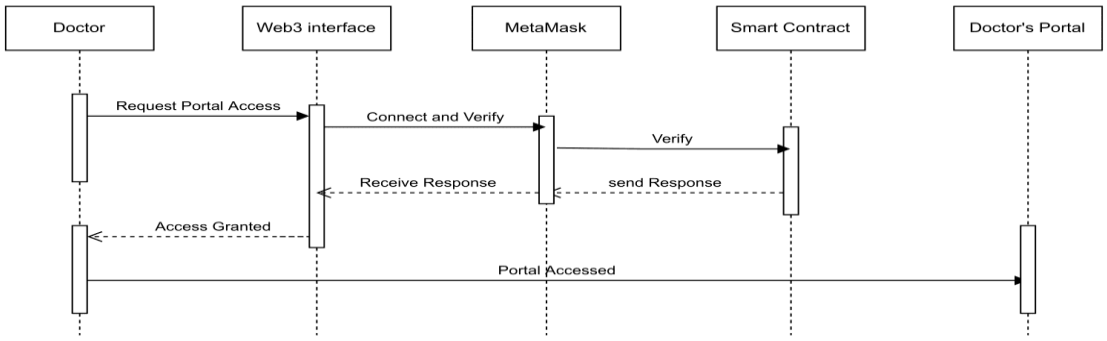


Figure 6

3.5.2.5 <Sequence Diagram 5>



3.5.3 State Diagram

3.5.3.1<State Diagram 1>

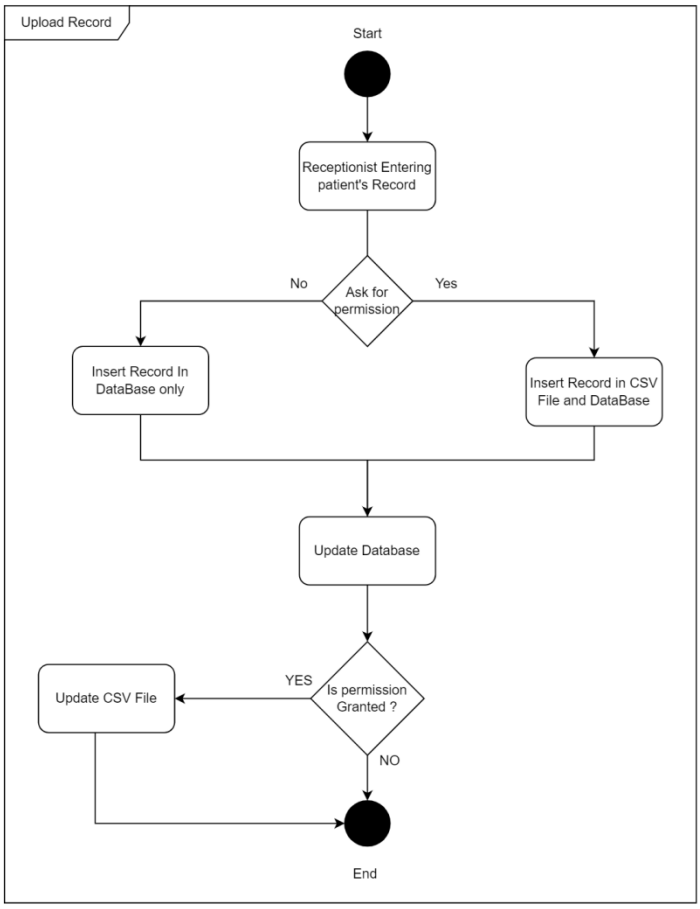
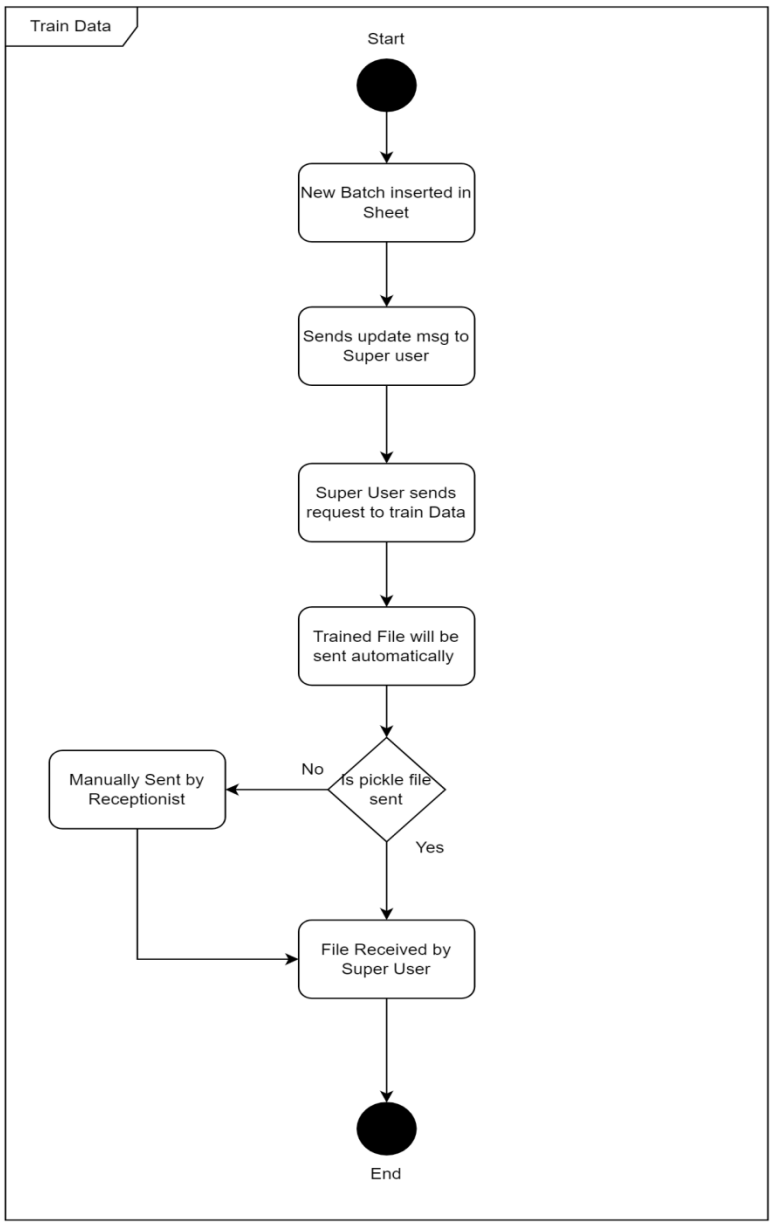


Figure 8

3.5.3.2<State Diagram 2>



3.5.3.3<State Diagram 3>

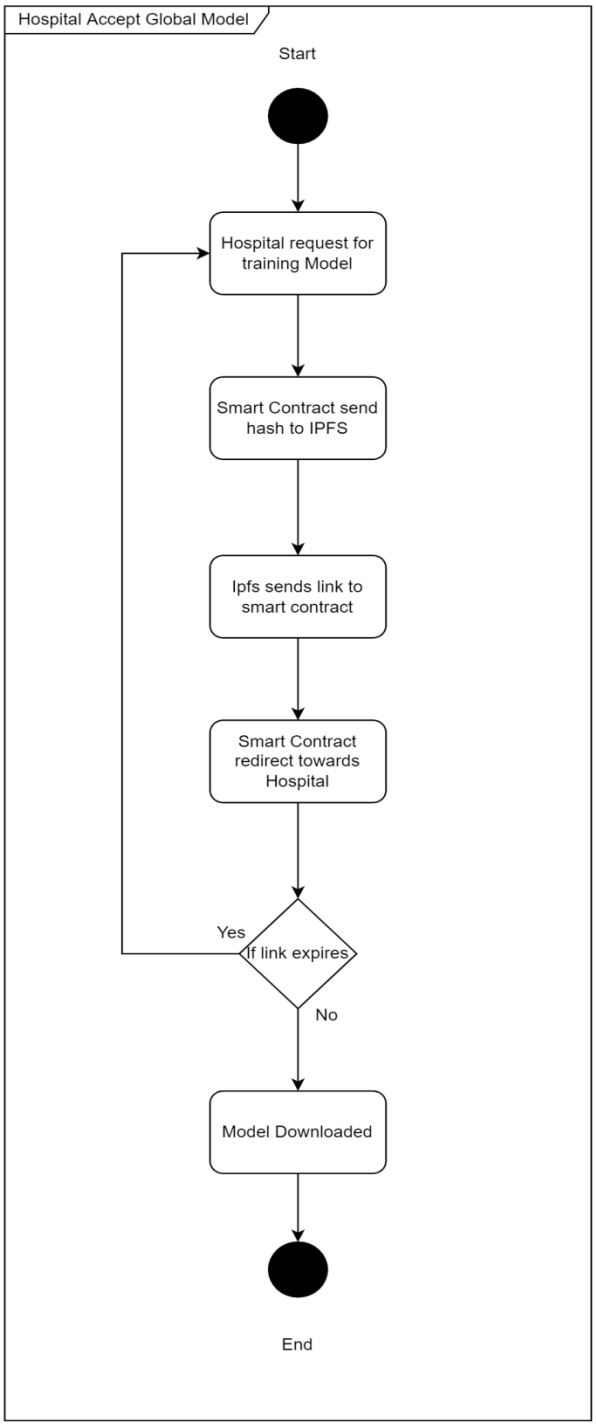


Figure 10

3.5.3.4<State Diagram 4>

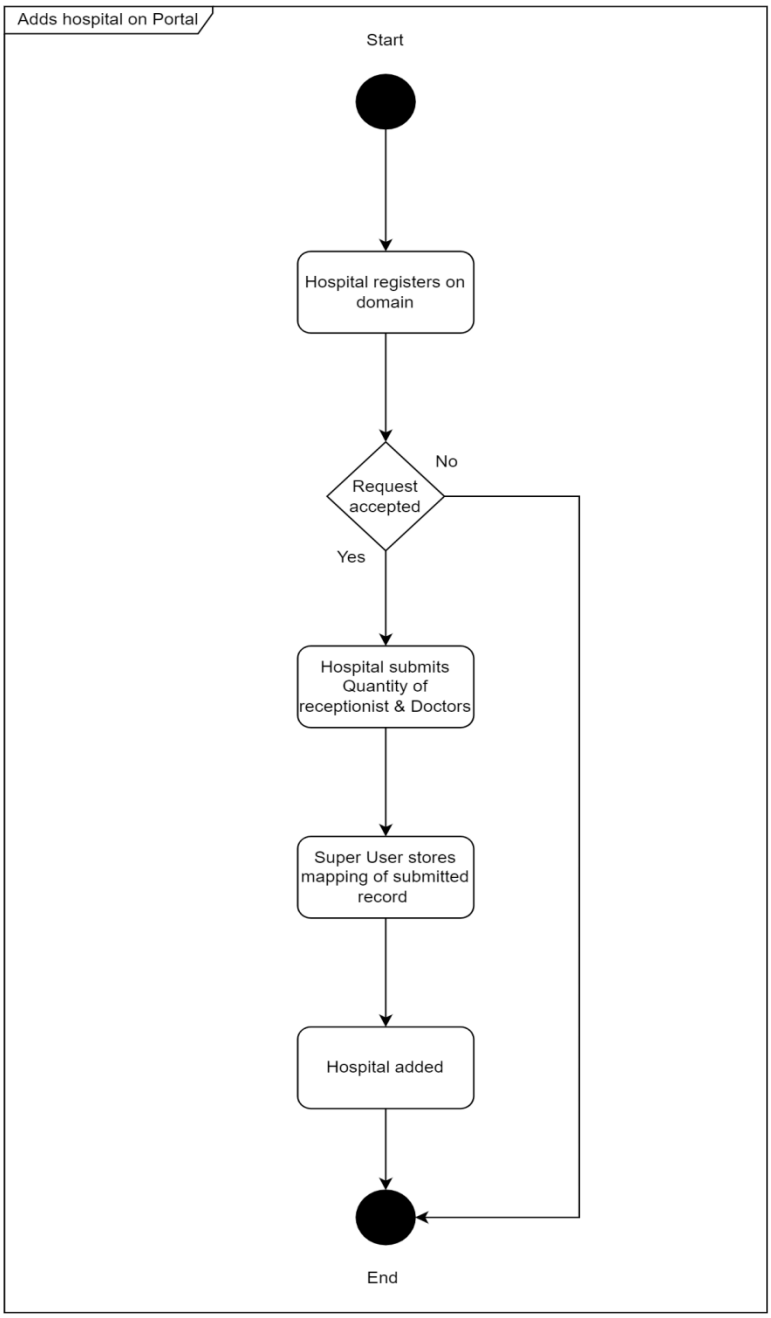


Figure 11

3.4 Data Flow Diagram (Not Applicable)

4 Implementation Details

4.1 System Architecture

Hospital can request the model from the Global server then download the model. After downloading the model, they will be training it on their own data. After the training is completed, the model will be sent back to the global server this process will be repeated by all the hospitals on the network who are taking part in this training after all the models are received by the global server it will then aggregate all these models through ensemble learning and a new model with combined intelligence of all the aggregated model will be formed This new model will be delegated back to all the hospital.

4.2 System Level Architecture

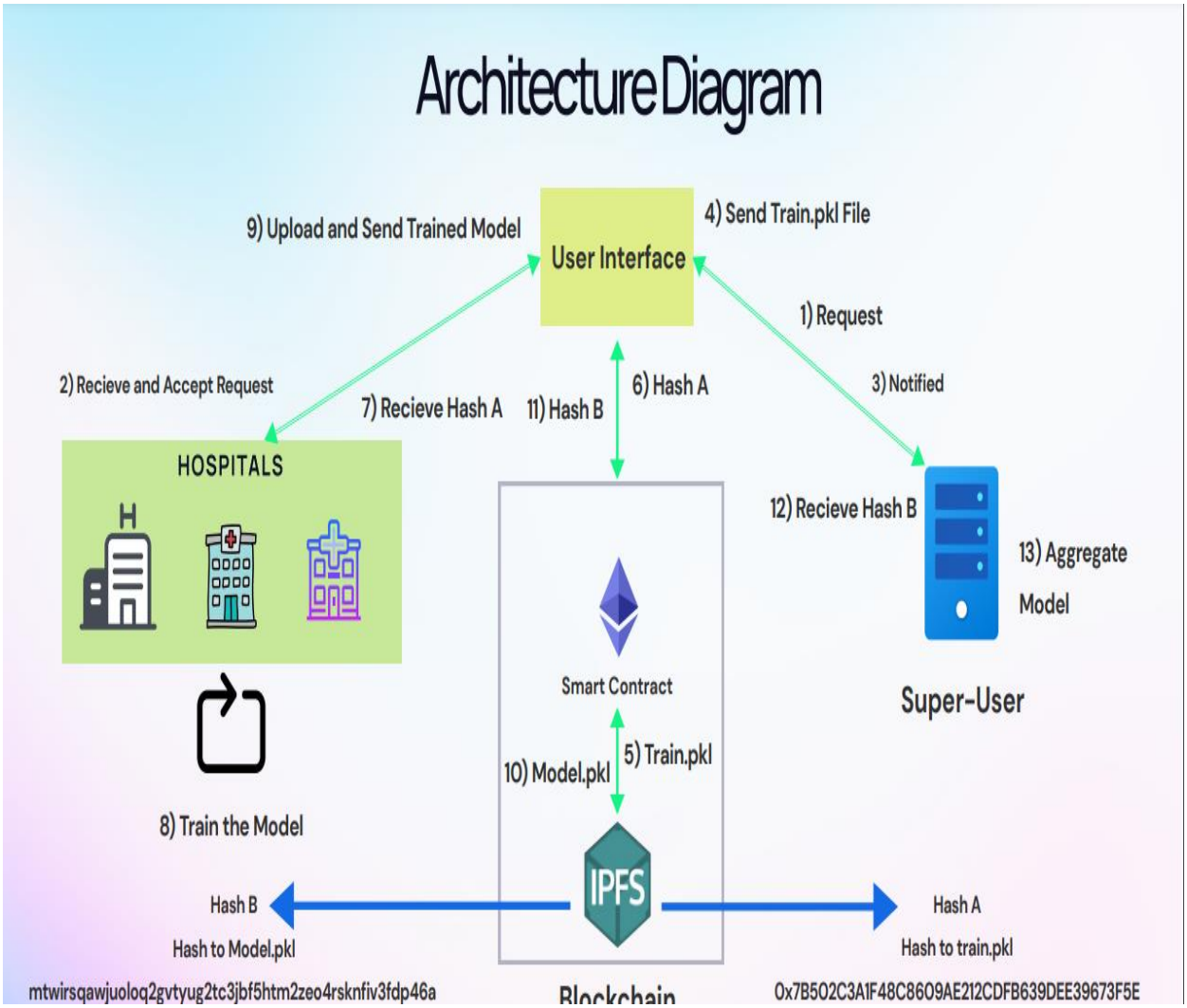
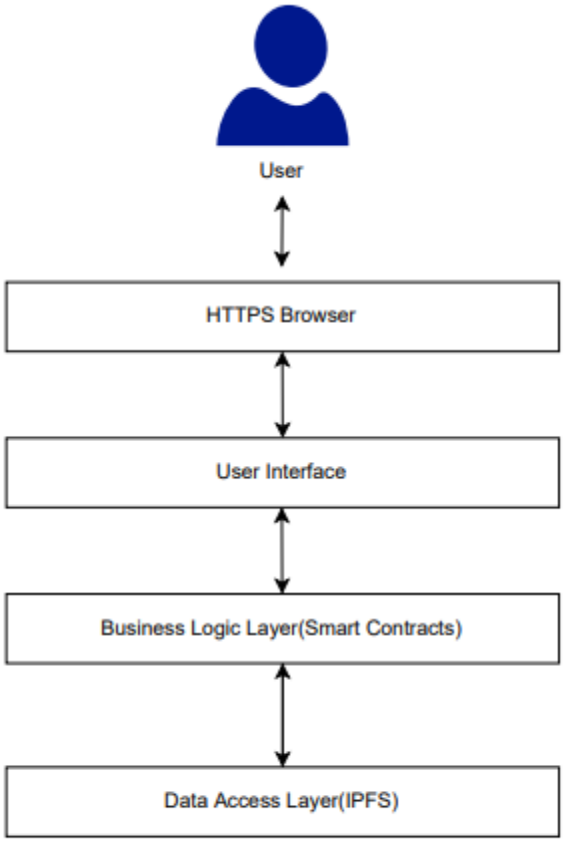


Figure 12

High Level System Architecture

4.3 Software Architecture



Layered Software Architecture
Figure 13

4.4 Development Tools Used

The development tools and software technologies used are as follows:

- React JS
- Node JS
- Express JS
- Git version controlling
- PyCharm
- Visual Studio Code
- Postman
- Thunder Client
- Solidity
- Remix
- EtherJS
- Web3
- Hardhat
- MetaMask
- Ganache

5 Testing and Evaluation

The purpose of this document is to provide a comprehensive outline of the validation and testing procedures necessary for ensuring the effectiveness and proper functionality of our project. The primary objective of this endeavor is to ascertain whether the project satisfies its requirements. In order to achieve this goal, three distinct types of testing willbe conducted: unit testing, performance testing, and system testing. Through these rigorous testing procedures, the entire system will be thoroughly scrutinized to guaranteeits efficient operation and robust performance. It is hoped that the results of these tests will provide valuable insights that will enable us to improve and optimize the project, thus enhancing its overall efficacy and value.

5.1 Environmental Needs

Following are the hardware assumptions for using the system

- **I3** core processor or above
- **4 GB** RAM or above
- **256 GB** SSD space
- Keyboard and Mouse
- Internet Browser (any)
- English language supported only

5.2 Validation Testing

5.2.1 Testing Approach(s)

Unit Testing: The main objective is to test individual components of a system aretested in isolation to ensure that they are working properly. This helps to identifyand fix any defects or bugs in the system's components before they are integrated.

Integration Testing: Testing all integrated modules to verify that the combined functionality after integration is in a workable state. The objective of this testing is to ensure that all the integrated modules are working properly and are producing the desired output.

System Testing: The entire system is tested as per the requirements. Overallrequirement specifications and all the combined parts of a system are tested.

Usability Testing: The objective of this testing is to ensure that the system is easyto use, user-friendly, and meets the user's expectations. It involves testing the system's navigation, interface, functionality, and user experience to identify any usability issues and to provide recommendations for improvement.

Performance Testing: The objective of this testing is to ensure the system worksunder heavy traffic that includes large size video length processing.

5.2.2 Test Pass/Fail Criteria

If the application failed to generate results, provide errors or crashes then it is a failure else it isPass.

5.2.3 Test Cases

Test Case: 01				
Description: Test Connectivity with Meta Mask				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	User visits on the landing	Landing page will be displayed	Page shown	Pass
2	Connect with meta mask	Login Button will be visible to IPFS	Login button will be displayed	Pass

Table 3 Test Case 1

Test Case: 02				
Description: Test login functionality				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	User route to the login page	Display the page	Page displayed	Pass
2	User enters their credentials	emptyfields filled	Verify emptyfields	Pass
3	User clicks on login button	Validate user login credentials	Validated credentials	Pass
4	User is redirected to respective page	Display the respective page	Displaying therespective user page	Pass

Table 4 Test case 2

Test Case: 03				
Description: Test Hospital registration functionality by admin				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	Admin route to the register userpage	Display the page	Page displayed	Pass

2	Admin enters the hospital credentials	Check for emptyfields	Verify emptyfields	Pass
3	Admin clicks on register button	Check credentials and register user in the database	Validated credentials and a new entry is added in the database	Pass

Table 5 Test case 3

Test Case: 04				
Description: Test Upload Model				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	User route to the upload Model page	Display the page	Page displayed	Pass
2	User enters the Model name & JSON and pickle files	Check for emptyfields	Verify emptyfields	Pass
3	User clicks on submit button	Upload Model Files on IPFS & hash in the smart contract	Upload Model Files on IPFS & hash in the smart contract	Pass

Table 6 Test Case 4

Test Case: 05				
Description: Test update Model				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	User route to the update Model page	Display the page	Page displayed	Pass
2	User enters the Model name & JSON and pickle files	Check for empty fields	Verify empty fields	Pass
3	User clicks on update button	Update Model Files on IPFS & hash in the smart	Update Model Files on IPFS &	Pass

		contract	hash in the smart contract	
4	User selected model will be uploaded	New Version of Model Uploaded	New Version of Model Uploaded	Pass

Table 7 Test Case 5

Test Case: 06				
Description: Hospital receiving doc token after participation				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	Hospital takes part in the training	Display the update model page	Page displayed	Pass
2	Hospital uploads the local model	Display successfully uploaded	Successful Bar Displayed	Pass
3	System checks if the model is being uploaded before the deadline	True statement will return	True Statement will return	Pass
4	Doc token is transferred to the hospital wallet	Tokens transferred in MetaMask wallet	Tokens received in MetaMask Wallet	Pass
5	Hospital can see doc token earned on the rating page	Doc token earned	Doc token will be added in doc wallet	Pass
6	Hospital can also see when is next training due	Display the rating page where doc token earned & next training date can be viewed	Doc Token earned & next training date will be shown	Pass

Table 8 Test Case 6

Test Case: 07				
Description: Hospital receiving penalty due to violation of deadline				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	Hospital takes part in the training	Display the update model page	Page displayed	Pass
2	Hospital uploads the local model	Display successfully uploaded	Successful Bar Displayed	Pass
3	System checks if the model is being uploaded after the deadline	D		Pass
4	Penalty Count is incremented of hospital	Penalty Count Increment will be displayed	Penalty Count will be added	Pass
5	Hospital can see Penalty Count & warning	Warning displayed	Warning displayed	Pass

Table 9 Test Case 7

Test Case: 08				
Description: Test Hospital registration functionality				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	Hospital route to the register Hospital page	Display the page	Page displayed	Pass
2	Hospital enters their own information	emptyfields will be filled	Verify fields	Pass
3	Hospital clicks on register button	Hospital Information will be sent to superuser via email	Validated credentials and a new entry is added in the database	Pass
4	Display page thank you for registration	Display the messages	Messages are displayed	Pass

Table 10 Test Case 8

Test Case: 09				
Description: Test Hospital fails to upload model				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	Hospital Upload model	Model is Uploaded	Model is uploaded	Pass
2	System checks if the model accuracy is below the benchmark set by superuser	File upload failed to IPFS	Upload will be failed	Pass
3	Uploaded Files rejected by system	Display rejection message	Rejection is displayed	Pass

Table 11 Test Case 9

Test Case: 10				
Description: Test Duplicate Content Rejection				
No.	**Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
1	If existing model is already present on the IPFS network new same upload will be rejected	Rejection will be displayed	Rejection will be displayed	Pass

Table 12 Test Case 10