Helwan University

Faculty of Computer and Artificial Intelligence

Information systems department

CHAIN CARE

Graduation project presented by:

[Dina Hosny Mohammed -20180231]

[Esraa Mahmoud Abdalmonem -20180103]

[Hager Ahmed Ibraheim -20180663]

[Haidy Ashraf Thabet -20180671]

[Mahamad Tarek Zaki -20180507]

[Mazen Ehab Salah-Elden -20180451]

Submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in Computers & Artificial Intelligence, at the Information systems department, the Faculty of Computers & Artificial Intelligence, Helwan University

Supervised by [Prof. Laila Abd El-Hamed]

CHAIN CARE

Meet Our Team

Dina Hosny

Esraa Mahmoud

Hagar Ahmed

Haidy Ashraf

Mohamed Tarek

Mazen Ehab

Table of Content

Acknowledgment

First and foremost, we would express our deepest gratitude and appreciation to our Advisor **Dr. Laila Abd El-Hamed** for her support, outstanding guidance, and encouragement throughout our graduation project.

As well as **Engineer. Abdalla Sayed** for his continuous support and help as he was always there for us providing more than the needed time and effort from him since the very beginning of our project.

We would like to thank our families, especially our parents we hope they are proud of us on our last year of education, we hope that we will start giving back to the community very soon! Thank you for supporting us and providing us the needed time, effort, encouragement, patience and assistance over the years.

Finally, our faculty for providing us with the courses that guided us into the right direction of our lives and the help of all the professors that left a great impact at our lives, **Thank you**.

Abstract

Main idea:

This system provides a whole medical history of the patient since the subscribing in it, also a professional history for doctors and lab doctors.

The generated data are stored automatically in electronic records forms ledgers by blockchain platform which makes it secured and well managed.

-Users will be able to leverage their medical data to power a plethora of applications and services.

-ChainCare Whitepaper 2.1 will outline the vision of the web application and the current issues in healthcare, as well as give a brief summary of the blockchain technology used and how web application is utilizing it to address specific issues to make healthcare better for users.

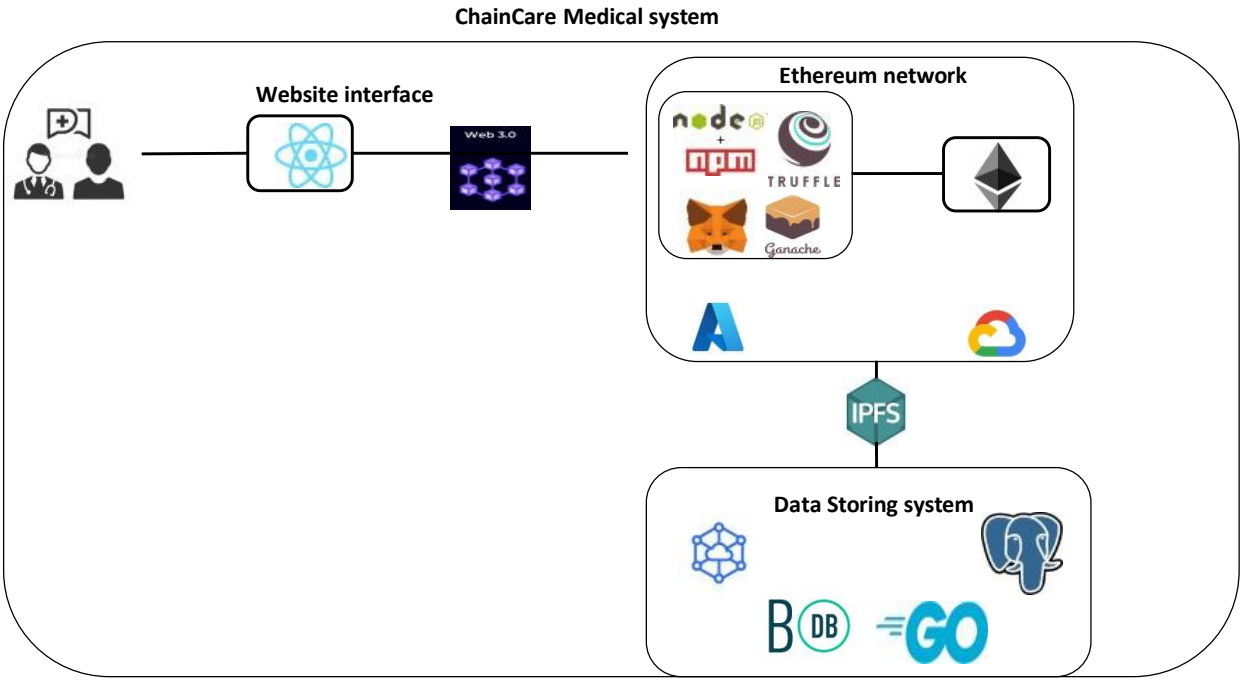
Structure:

The program is built by a connected various tools and suits and a supportive platform, to form a coherent network could maintain the implementation and architecture.

Technologies and process: -

Technologies & Tools: -

- illustrating the technologies connection



1) Node.js, npm

we use **node.js** framework to build our application network to control many connections, as it contains specific packages and libraries essential for the web application like **npm** library. They can be set up and update packages through cmd command.

Advantages:

- Used for setting up blockchain environment.
- Helps in building blockchain testing environment.
- Used in deploy and running Ethereum smart contracts.

2) Truffle

Essential tool for building dapp /smart contract. It is considered

A testing environment and an **Ethereum virtual machine (EVM).**

Advantages:

- Support direct compiling smart contacts.
- Link smart contracts together.
- Support deploying smart contracts.
- Testing smart contracts.
- Manage networks and packages.

3)MetaMask

We used it as a private **Ethereum** network. It is a browser plugin to connect Ethereum app with the browser which allows making smart contract transactions and deploy them. it provided electronic wallet with network IP and account to manage Ethereum smart contract.

Advantages:

- it is an open source with existing features.
- It provides built in coin purchasing.
- It provides a local key storage.

4)Ganache

We used it as a personal **Ethereum** blockchain enabled us to develop dapps it is used across all the entire application. It provided us with number of accounts allowed us to test and deploy the application and dividing them among application different end users. It has a gas price and gas limit to control the mining process.

Advantages:

- Allows employment and running smart contracts transactions.
- Enable quick and easy testing for **Ethereum** smart contracts.

5)Ethereum

In **ChainCare** application we used **Ethereum** to build its contact with the transactions to create a public decentralized blockchain dapp and create chains of structs includes blocks with addresses. It performs a network with the other tools after being connecting together, then being compiled and deployed.

Advantages:

- Natively supports smart contracts.
- It is. scalable, programmable, secure, distributed, and decentralized.
- Executes smart contracts and store data for third-party application.

6)Web3

We used **web3**, cause smart contract (**Ethereum**) can't deal with languages of structuring and presenting web pages like **HTML/CSS**. So, we used **web3** to connect smart contract with the web.

Advantages:

- **Web3** and **truffle** suit makes a framework serves dapps.
- **Web3** transfer instructions of transactions in the smart contracts to web structures, then returns the responses to the smart contract and so on.
- Provides ownership and control to end-users about their data through encryption it.
- Provides a trusted secured platform in which data is fully encryption.
- Anybody can create an address an interact in blockchain.

7) IPFs

we used **IPFs** to store and access files of the generated blocks.

Advantages:

- Enables us to exchange blocks between users.
- Creates a network of nodes hashed with public key.
- Can store small scripts or big databases.

8) GOLANG

We used **Golang** to build the API in which we store database.

Advantages:

- It is compiled directly without needing an interpreter, so it excites fast.
- It is scalable and considered as an open source.
- It has a cloud design.
- It has many reach libraries.

9) Big Chain

We used it as the way of storing and handling the enormous amounts of data, as dapps has a high growth rate of the generated data unlike the centralized systems.

Advantages:

- Creates a tamper-proof record of stored information.
- Data is non-volatile once it has been stored it cannot be deleted or modified.
- It provides peer-to-peer network with transparency where each node maintains a permanent copy of the ledger.
- Creates customized private or permissioned blockchain network with top-notch features like transparency, scalability and immutability.
- sets permissions at transaction level for a clear distribution of tasks and responsibility.

10) PostgreSQL

We used it in storing and scaling this complicated data workload.

Advantages:

- Support writing database functioning using various language.
- Support for a huge number of data types including.
- It helps in align customized database more closely with the way the development data is represented in the applications.
- provides powerful techniques for finding and operating on data in semi- and unstructured text.
- Robust authentication, access control, and privilege management systems suitable for organizations of any size.

11) Storj Crypto

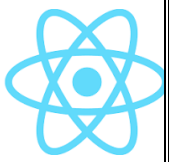
We used **Storj Crypto** for powering storing the decentralized files, as it enables to users to buy to other users for getting data.

Advantages:

- It allows any computer running its software to rent unused hard drive space to users looking to store files.

12) React JS

We used **React JS** for building our interactive website in its front end.



Advantages:

- It allows developers to utilize individual parts of their application on both client-side and the server-side.
- The React code is easier to maintain and is flexible due to its modular structure.
- It provides high performance in mind.
- Deploying React is fairly easy to accomplish if you have some basic knowledge of JavaScript.
- It saves time for developers as they don't have to write various codes for the same features.

13) Google Cloud Platform

We used **google cloud platform** for creating various virtual machines with (ubuntu or windows) operating systems for setting

up **Ethereum** environment in much less time and effort, to enable all team to work narrowly.

Advantages:

- It gives free trials for a period of time.
- Allows us to share the same pool of work and environment.

14) Microsoft Azure

We used it for creating a virtual machine to build the **Ethereum** environment.

Process: -

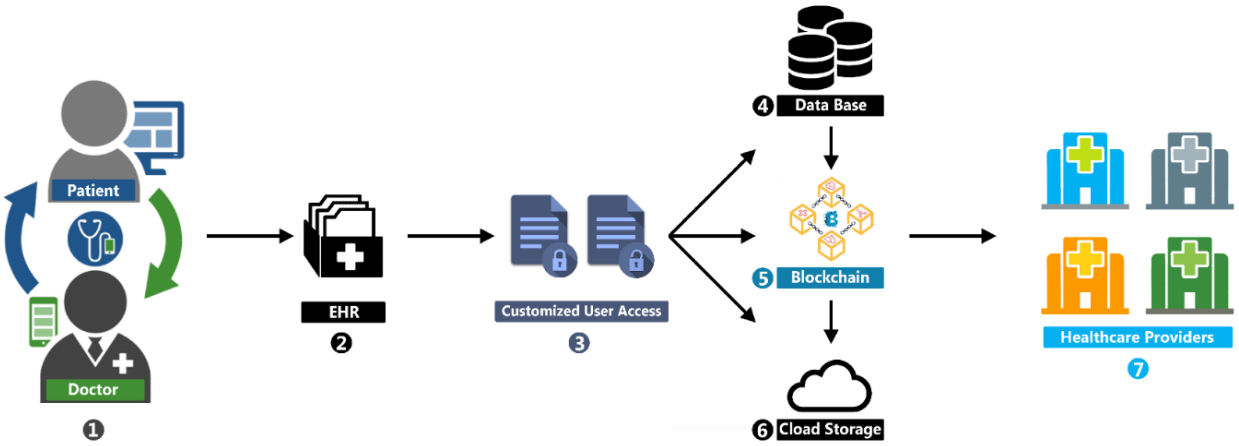
The patient registers in the system and enters his personal information using an (address, and private key).

Every patient will have his ledger which can be accessed through the specialist doctor during a specific session of time which will be saved then as a history record.

The specialist doctor and the lab doctors after submitting and get approved from the system Admin, each will have professional ledgers in which all the diagnostics and medical advices he gave is stored.

Admin is able to access doctors' professional histories. Admin could (Add, Remove, or Modify) doctors.

For illustration:


Patient
Doctor
1
EHR
2
Customized User Access
3
4 Data Base
5 Blockchain
6 Cload Storage
Healthcare Providers
7

Chapter 1: Introduction

In this chapter, we are going to discuss and go deeper in the overview of the project and know more about its scope and limitations and explain some terminologies we will find throughout the document.

Overview

ChainCare is a real-time official health record of a patient in a digital version that can be shared easily, securely and instantly among different facilities and the departments.

It includes every data required to get the details of the patient, including medical history, radiology images, diagnoses, treatment plans, allergies, laboratory results, etc.

It also helps to detect possible future genetic diseases and plays a significant role in healthcare because it delivers easy access to health records used in decision making related to the patient's care.

Objectives

- Storing all your health information, diseases you have had, medications, and medical tests.
- Detect the possible future genetic diseases.
- Improving the quality of care by reducing medical errors.
- gives easy access to patient's health records.
- Digitalize the paper-based methods in the hospitals.

Purpose

ChainCare aims to change the traditional health system with an e_health system and improving the quality of health service by delivering easy access to health records used in decision making related to the patient's care.
ChainCare Prepare accurate information about the health situation all over the country.

Scope

The approximate work involved to finish the project is divided into these five phases:-

1. Planning:

- Collecting data about the project and the lack that made us in a need to the App.
- Making surveys.
- Determining the functional and non-functional requirements.
- Setting a Gantt chart for the project.
- Determining the resources of the team.

2. Designing:

Determining the diagrams to be carried out within the project:

- Activity Diagram
- Use-case Diagram
- ERD
- Sequence Diagram
- Class Diagram

3. Coding:

The main functions to be coded in this site are:

- **Login** as a patient/doctor/lab.
- **View** profile of patient/doctor.
- **Update** personal info of patient/doctor.
- **Revoke** the access request by patient.
- **Accept** the access request by patient.
- **Logout** as a patient/doctor.
- **Update** medical info of patient by doctor.
- **Add** medical info/tests and medications of patient by doctor.
- **Upload** results of medical tests by labs.

- **Check** (patient/doctor/lab) information by admin.
- **Disease prediction** Detect the possible diseases.
- **Reports generation**

4. Testing:

Functional Testing:

- Unit Testing.
- Regression Testing.
- Integration Testing.

Non-functional Testing:

- Performance Testing.
- Stress Testing.
- Security testing.

5. Documentation

At this stage we were trying to document everything our project passed by starting from validating the idea until it became a real touchable project, we organized our documentation into 8 main chapters as follow:

- **Introduction:**
  In this chapter, we gave a quick summary of the project highlights & limitations.
- **Planning & Analysis:**
  Here we talked about planning the project, this includes the feasibility study & the competitor analysis, what is missing in the available solution and what are the requirements we will fulfil in our solution.
- **Software Design:**
  At this chapter, we included all the diagrams that we worked on as a visualization of the functions and scenarios we have, this includes sequence diagram, activity diagram, class diagram and ERD.
- **Implementation:**

In this chapter, we will show you the architecture of the application and snapshots of the main functions we will also mention the technologies we used.

- **Testing:**
  In this chapter tested the application through different ways.
- **Result & Discussion:**
  In this chapter we discussed the expected results for the project and the actual results, what were the difficulties we faced, how we managed or tried to solve them and what our final thoughts on the project were.
- **Future Work:**
  Here we will discuss what is our future plans for Plantopia and what is the ideas we are planning to add to the application.
- **Conclusion:**
  This chapter summarizes our whole document.

General constraints

- Tasks division, which can be not fair enough.
- Indiscipline "Human factor" like being late in delivering tasks or attending meetings.
- Hesitation, especially when it is related to taking a serious step or learning a new technique.
- Time management.
- Learning new technologies may take much time.
- Underestimating the objectives that may not lead to realistic or achievable function.

Chapter 2: Project Planning and Analysis

In this chapter, we are going to discuss and go deeper in how we plan the project and show the steps and the instructions that we have followed to plan the application.

Project Planning

Feasibility Study

Being a web application, ChainCare will have an associated hosting cost. Since the system does not consist a lot of multimedia data transfer, bandwidth required for the operation of this application is low.
ChainCare will follow the freeware software standards.
No cost will be charged from the potential customers. Bug fixes and maintaining tasks will have an associated cost.
From these it is clear that the project ChainCare is financially feasible.
The web will be hosted in a paid web hosting space with a sufficient bandwidth.
ChainCare is a complete web-based application.
The main technologies and tools that are associated with system are

- HTML
- CSS
- Bootstrap
- React
- Java script
- Node JS
- Solidity
- GO

- Resource and Time Feasibility
  Resources that are required for ChainCare includes:
  - Programming device (Laptop)
  - Programming tools (freely available)
  - Programming individuals

- Social/Legal Feasibility

ChainCare uses freely available development tools and provide the system an open-source system.
Java script Software libraries that are used in this system are free open-source libraries.

- Schedule feasibility
  Of course, we divided the application development stages into small tasks with a specific working duration with specific deadline.

Estimated Cost

A cost estimate is approximation of the cost of a program, project or operation. The cost estimate is the product of the cost estimating process. Several studies estimate the cost of purchasing and installing an electronic health record range from $20,000 to $70,000 per provider.

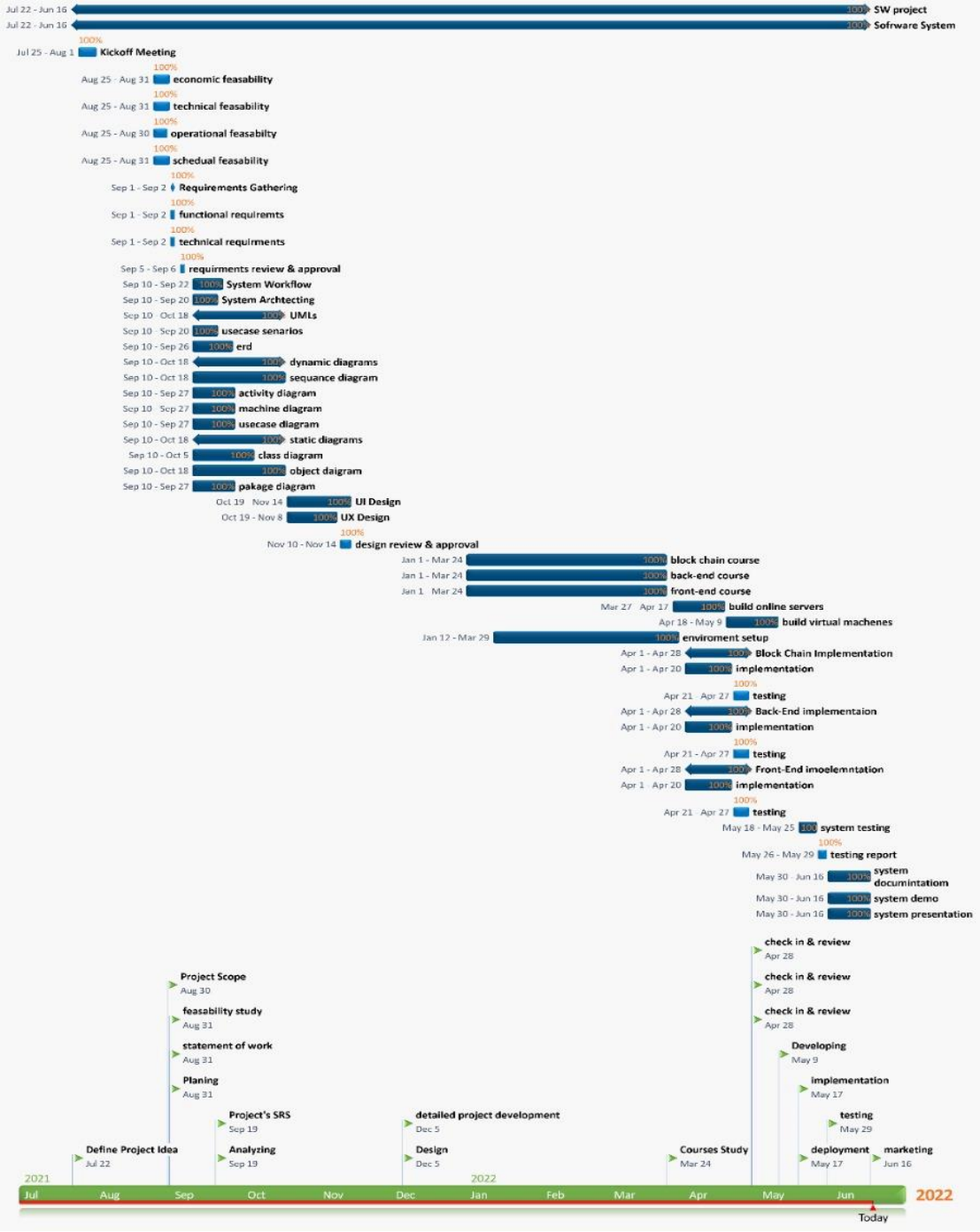
In our system, we have five components to estimate how much the system will cost:

- **Hardware:** Potential hardware costs may include clouds, desktop computers and tablets/laptops.
- **Software:** Potential software costs may include ChainCare web application, interface modules and upgrades to web application.
- **Implementation Assistance**: Potential implementation assistance costs include IT contractor, attorney, chart conversion, hardware/network installation and workflow redesign support.
- **Training:** We will need to train physicians, doctors and office staff to understand how to use the system and associated hardware.
- **Ongoing Network Fees and Maintenance:** Potential ongoing costs include hardware and software license maintenance agreements, ongoing staff education and IT support fees.

Gantt Chart

This chart describes the workflow of building the system during the whole period.

The workflows are moderated through agile methodology.



Analysis and limitation of existing system

Right now, most of the clinics in Egypt uses paper and pen to write the patient data and there is no system and if the system has been found, it is just local into the organization not all the health care facilities.

EHR technology is resolving some issues on one side but further create some issues on another side. It becomes easy to hack the data if the prior precautions have not been taken

Need for the new system

Information saves lives. This is especially true in the medical field.
Doctors and nurses use ChainCare to make data-driven decisions regarding various facets of patient care. For example, quick access to patient medical histories can bring previous treatments to light.
ChainCare are also invaluable to administrators, who can analyze statistics about different departments or procedures to better allocate the organization's resources.
Electronic health records contain highly crucial and critical patient-related medical data.
Blockchain can modernize the electronic health record exchanges by delivering a secure approach for medical data exchange of medical data in the field of healthcare productiveness, by safeguarding it through a dispersed peer-to-peer linkage.
Blockchain will handle the security, reliability, immutability and interoperability features.

Analysis of the new system

User Requirements

- The user uses the app with him at any place and any time so our app should be available in any place and any time.
- The user easy to see his medical history.
- The system secures all medical history and profiles for users to avoid data breaches.

System Requirements

- Browser: The application works on any browser ex." Chrome, Microsoft edge, Internet explorer"
- Internet connection: This app uses a cloud server to store data, so you need to be connected to the internet to fetch and view it.

Domain Requirements

- Multiple users must be able to use the application simultaneously without corrupting the cloud (whatever form it may be).
- A server must be set up to host the cloud, and the server must be accessible by all the systems running the inventory tracking software.
- The web application must verify all values before making the change in the cloud.
- The application must have update capabilities for future models.

Functional Requirements

- Patient registering in the system and own profile and ledger.
- Doctors (specialist/lab) can submit into the system.
- Specialist Doctor starts a diagnostic session to add new record in patient's ledger.
- Lab doctors start a session to add medical tests and analysis in a new record in patient's ledger.
- Admin add new doctor in the system.
- Doctor can access patient's medical history.
- Patients' records are added in sequential successive way in his ledger stringing his medical history.
- Admin can access doctors' professional history.
- Admin can modify and update doctors' profile information.
- System allows users to predict the disease according to the entered symptoms.
- System generates statistical reports about number of selected records of period of time.

Non-Functional Requirements

- **Data integrity and Availability:** Our application can be used in any time and it always "uptime" is the amount of time that it is operational and available for use. This is specified because some systems are designed with expected downtime for activities like database upgrades and backups. Integrity implies preserving the efficiency and consistency of the data. In ChainCare, it leads to the fact that data has not been damaged by unapproved use
- **Privacy and security:** security and privacy in the healthcare environment are intended to give patients the authority to manage their medical records through the provision of authorizations.

- **Confidentiality:** confidentiality is distinguished from privacy related to the dimension of reliable communication or contract between providers and the patients. Patient's records need to be held in confidence.
- **Access control:** Medical information should then be accessible only to authorized healthcare specialists and patients. Patients should obtain their data and provide control over who can access it.
- **Flexibility:** If the team intends to increase or extend the functionality of the software after it is deployed, that should be planned from the beginning; it influences choices made during the design, development, testing, and deployment of the system.
- **Data sharing:** exchange of medical records is an essential requirement with the patient's treatment being scattered over various health care providers; therefore, the data is shared with other medical institutions and government.

Advantage of the new system

- **Easy Access to Patient Data:** ChainCare means readily available patient data to the care providers. It is only a matter of few clicks and all the requisite information about a patient, from various departments in the hospital, can be available on the screen.
- **Cost Effective:** ChainCare cuts out on a lot of manual work that are essentially performed in hospitals, especially the ones where documentation and record keeping is required. It helps in cutting down manpower because a lot of work gets automated and does not require manual intervention to store or analyse the information. It also saves much on storage and the related costs.
- **Reduces Scope of Error:** Because processes on ChainCare are automated and a lot of tasks are assigned to the software to perform with utmost accuracy with minimum human intervention, the scope of error is reduced dramatically.
- **Increased Data Security & Retrieve-ability:** All the data is stored on the server or cloud, keeping it safe. Since ChainCare works

on logins, data security is becoming a non-issue offering data access based on the role of the person – Receptionist, doctor, nurse, radiologist etc.

- **Improved Patient Care:** Improved access to patient data and improved work efficiency means better and faster clinical decisions.
- **Blockchain technology** in healthcare is that it can include record interoperability, improved patient information access, and system monitoring spanning a device's entire life cycle in the blockchain substructure. Blockchain can be used along with other emerging technologies like the Internet of Things (IoT) and Cloud computing for better EHR solutions.

Risk Management

Schedule Risk

	Risk	Probability rate	Impact rate
Schedule Risk	Wrong time estimation.	Low likely	High
	Resources like team and their skills are not tracked properly.	Mid likely	High
	Failure to identify time required to develop functions.	Mid Likely	Mid
	Unexpected project growth.	Most likely	High

- **Can be avoided by:**
  - Following up the Gantt chart regularly and reduce the critical paths in theproject network.
  - Look after the major tasks -especially- that the other tasks depend on.
  - Monitor\review the agreed-on time plan.
  - Scheduling the risky tasks first.

- **Can be reduced (if happened) by:**
  - Updating the plan to adapt to any newly added requirements withoutaffecting the already existed plan.
  - Maximize the rest of the resources like team and tools.
  - Not wasting time because of the problem but move forward.

Budget Risk

	Risk	Probability rate	Impact rate
Budget Risk	Wrong budget estimation.	Most likely	Low
	Cost overruns	Low likely	Low
	Project growth	Low Likely	Mid

- **Can be avoided by:**
  - Good research.
  - Regularly checking the budget spent and needed.
  - Never skipping the budget through any newly added requirements.

- **Can be reduced (if happened) by:**
  - Sacrifice not-so-needed resources that consumes money.
  - Quick adaptation.
  - Replace cost-consuming parties with other available-resource-consumingparties.

Operational Risk

	Risk	Probability rate	Impact rate
Operational Risk	No resource planning.	Low likely	Mid
	No communication in the team.	Low likely	Mid
	No distribution for responsibilities.	Mid Likely	High
	Not enough training.	Mid likely	High

- **Can be avoided by:**
  - Monitoring and evaluations at regular intervals.
  - Cutting the project into very small tasks.
  - Full training before proceeding with the work.
  - Never skipping the budget through any newly added requirements.

- **Can be reduced (if happened) by:**
  - Adding pieces of training into the normal work.
  - Rotation through the training.
  - Changing the responsibilities to achieve the best-optimized output.
  - Learning and planning again.

Technical Risk

	Risk	Probability rate	Impact rate
Technical Risk	Changing requirements.	Most likely	Mid
	Flutter is in its initial stages.	Most Likely	High
	The database is not efficient.	Low likely	Mid
	Attacks or application failure	Mid likely	High

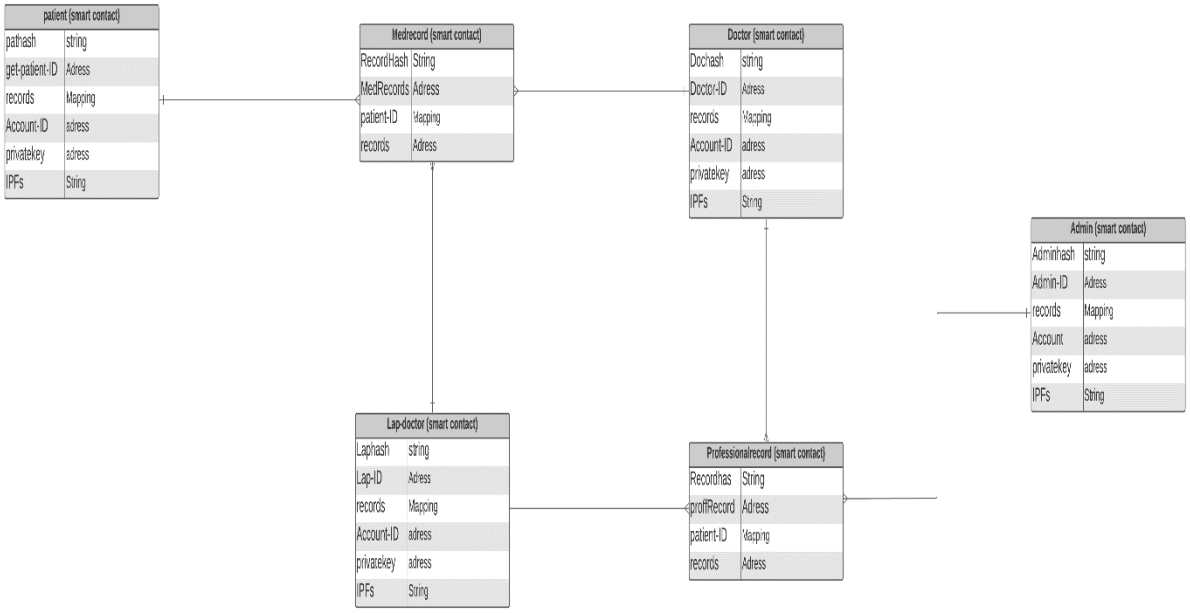
- **Can be avoided by:**
  - Understanding the platform\tools used.
  - Not rushing the analysis phase.
  - Specifying the desired outcome.
  - Detailed test cases and regulations.

- **Can be reduced (if happened) by:**
  - Analyzing the security threats.
  - Preparation for facing any attack.
  - Quick adaptation to any new requirements.
  - Evaluating \monitoring the Blockchain from time to time

Chapter 3: Software Design

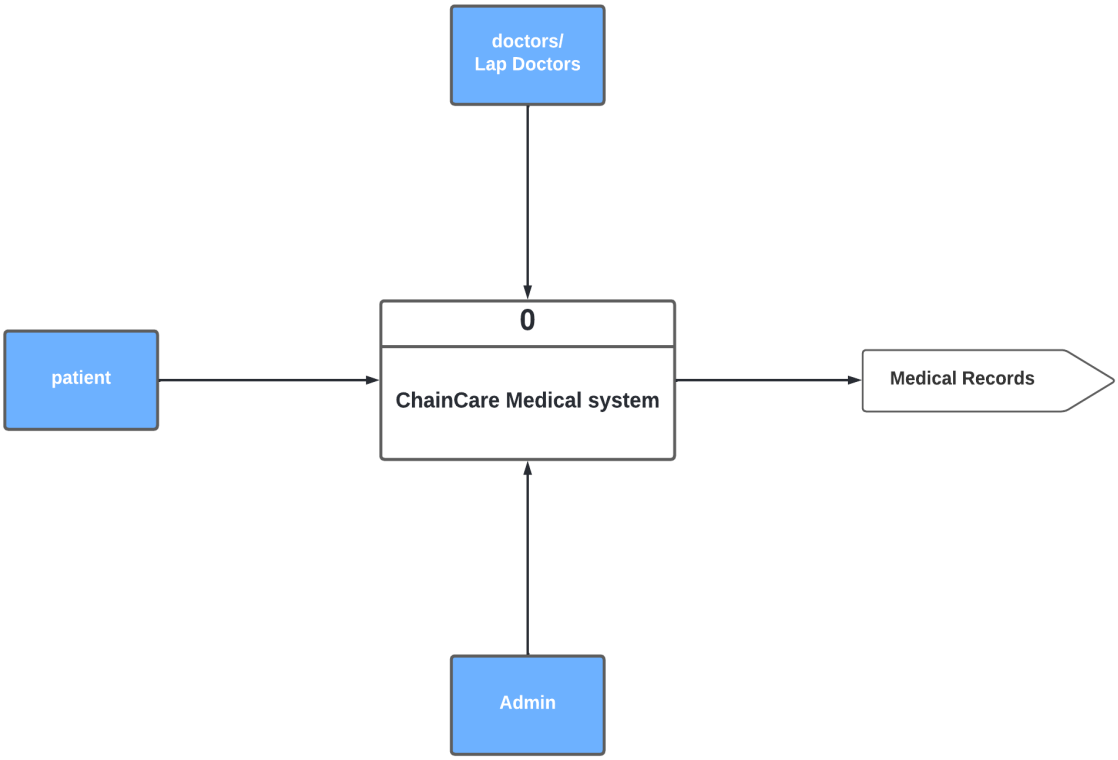
In this chapter, we are going to discuss and go deeper in **ChainCare** recommendation system design and present its diagrams and database.

ERD diagram

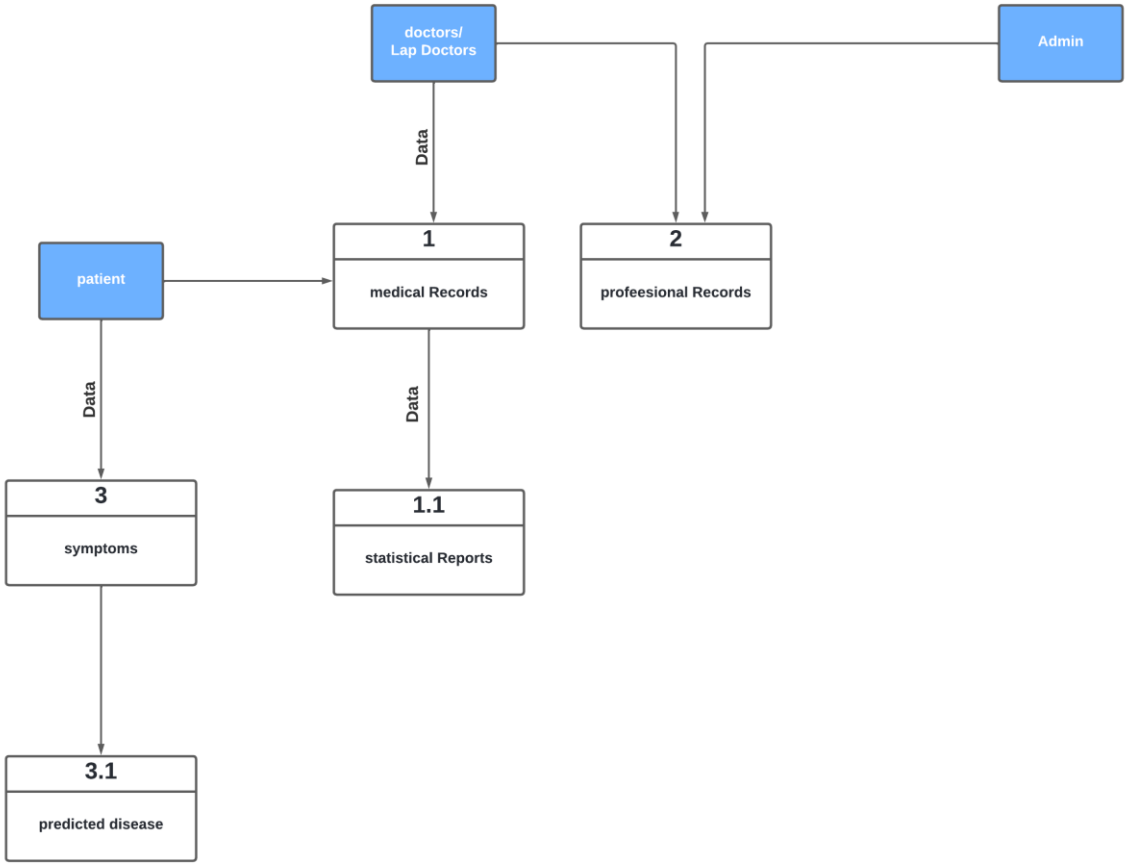

patient (smart contact)
pathash string
get-patient-ID Adress
records Mapping
Account-ID adress
privatekey adress
IPFs String
Medrecord (smart contact)
RecordHash String
MedRecords Adress
patient-ID Mapping
records Adress
Doctor (smart contact)
Dochash string
Doctor-ID Adress
records Mapping
Account-ID adress
privatekey adress
IPFs String
Admin (smart contact)
Adminhash string
Admin-ID Adress
records Mapping
Account adress
privatekey adress
IPFs String
Lap-doctor (smart contact)
Laphash string
Lap-ID Adress
records Mapping
Account-ID adress
privatekey adress
IPFs String
Professionalrecord (smart contact)
Recordhas String
proffRecord Adress
patient-ID Mapping
records Adress

DFDs Diagrams

DFD Diagram Level 0

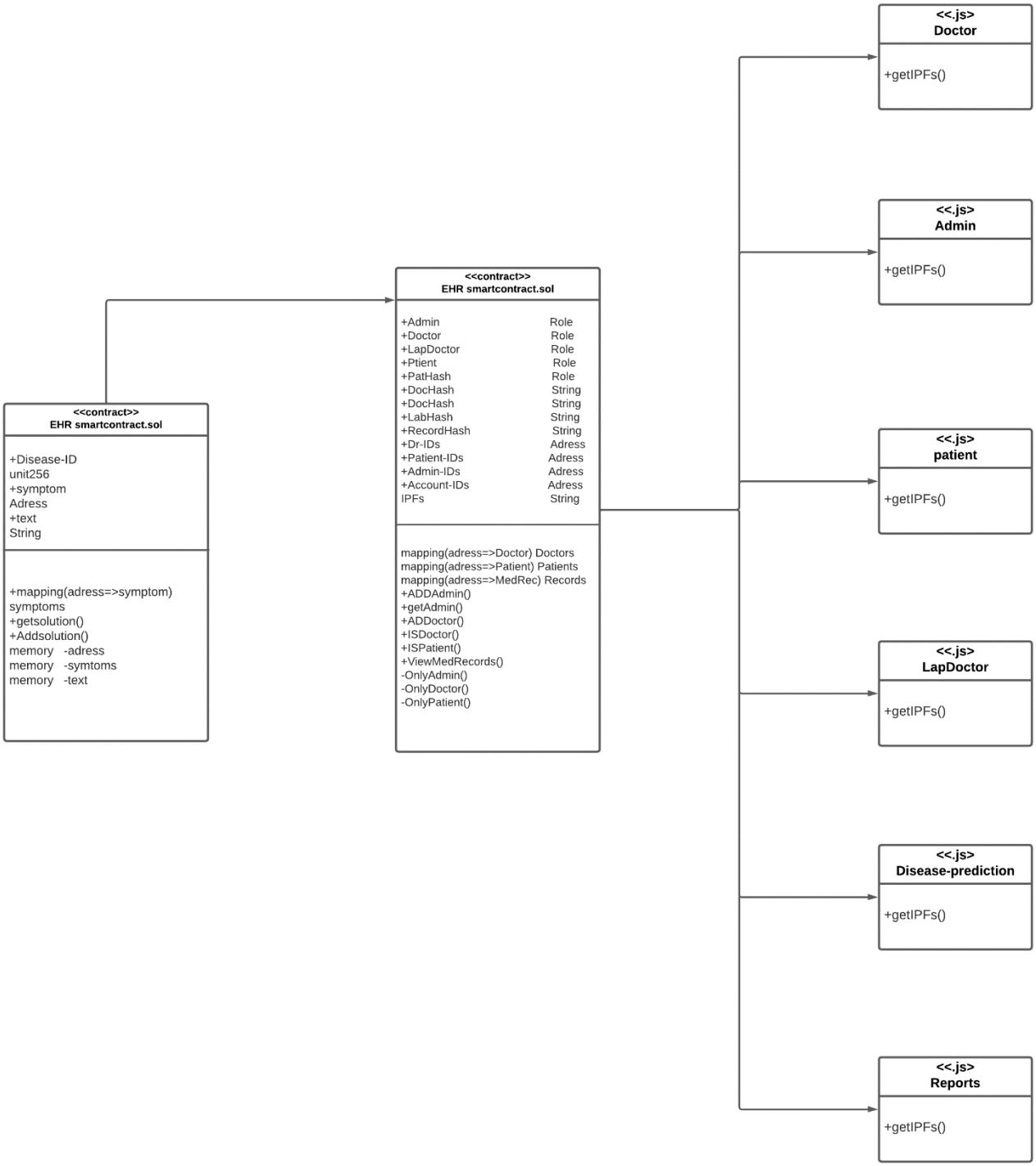


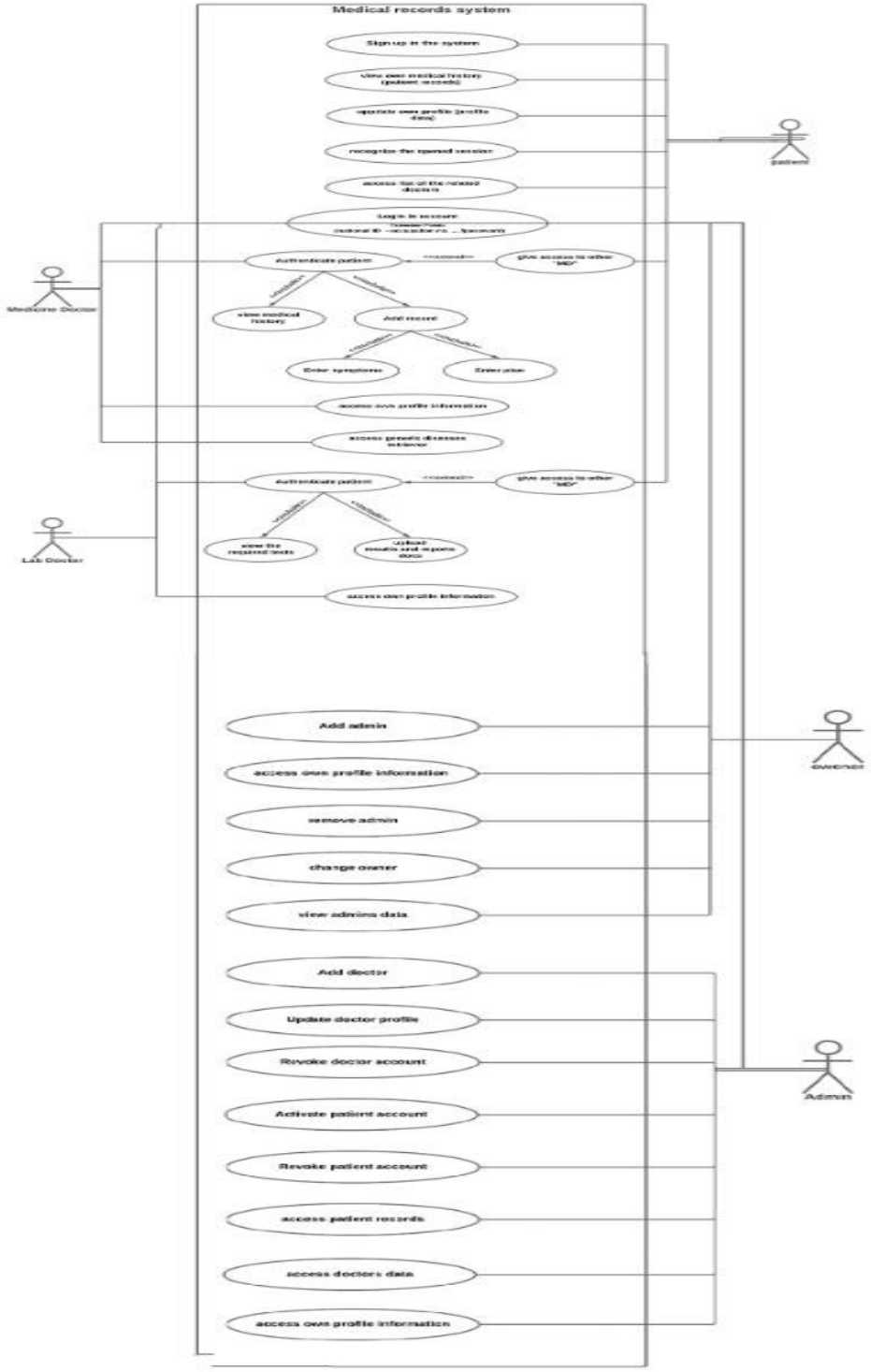
DFD Diagram Level 1



Class Diagram

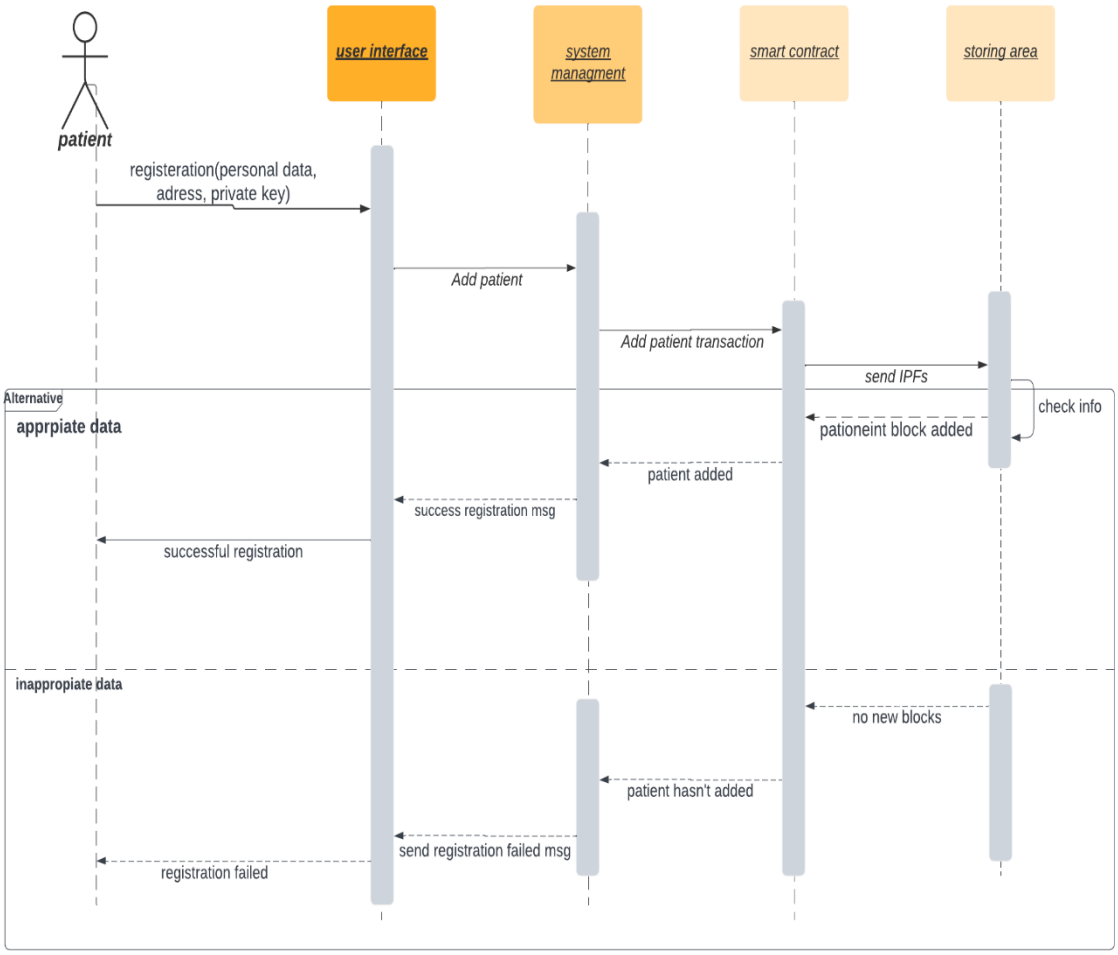
USE-CASE Diagram



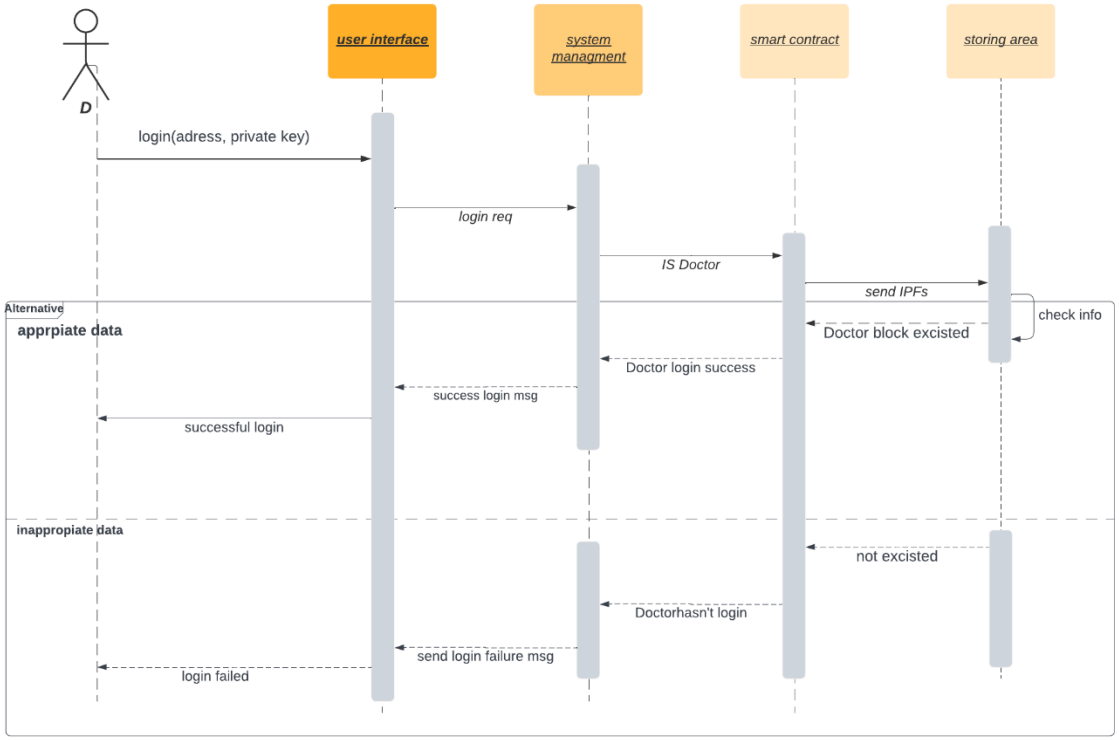

Medical records system
Sign up in the system
patient
Medicare Doctor
Lab Doctor
Add admin
access own profile information
remove admin
change owner
view admins data
Add doctor
Update doctor profile
Revoke doctor account
Activate patient account
Revoke patient account
access patient records
access doctors data
access own profile information
Admin

USECASES sequence Diagrams

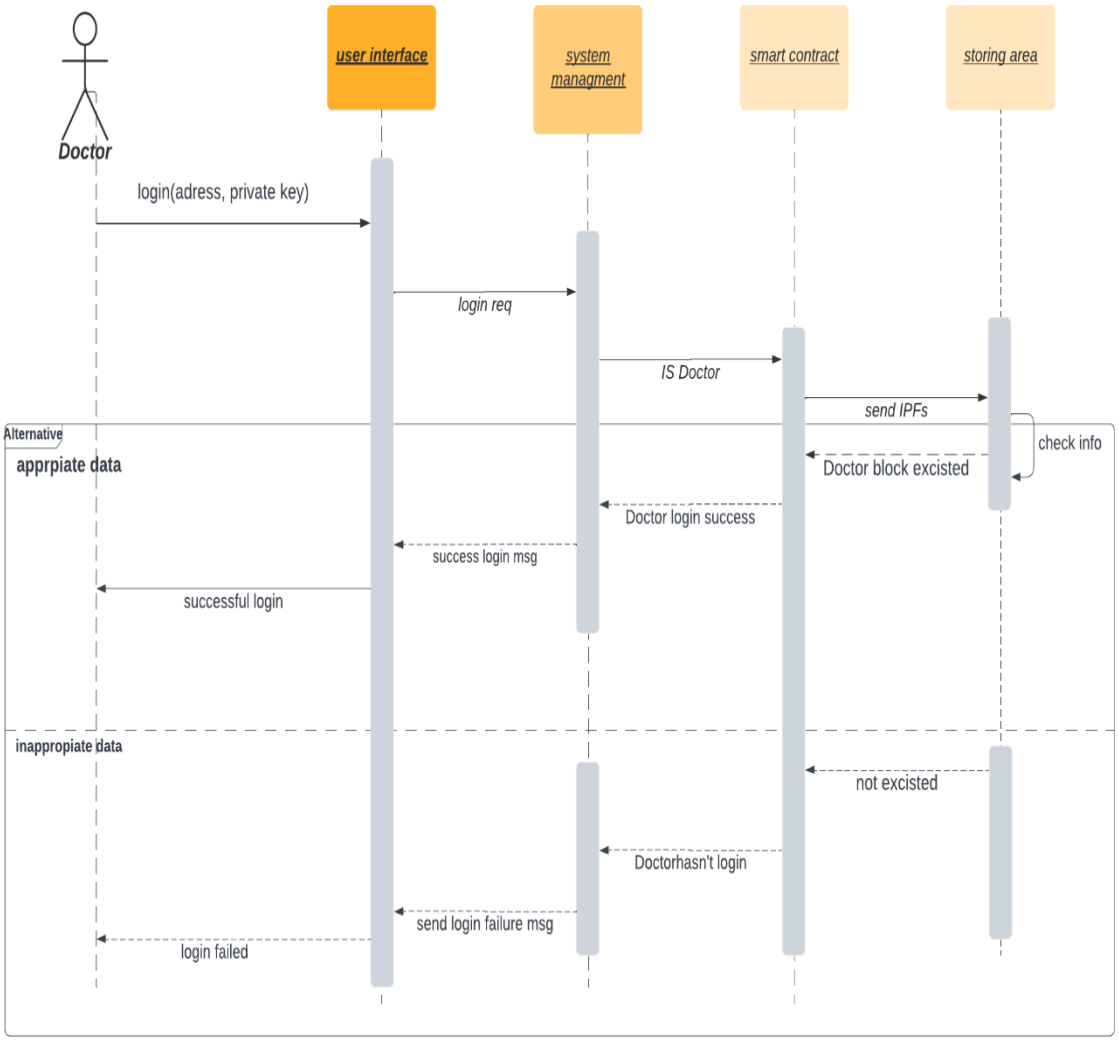
- **Patient signup**



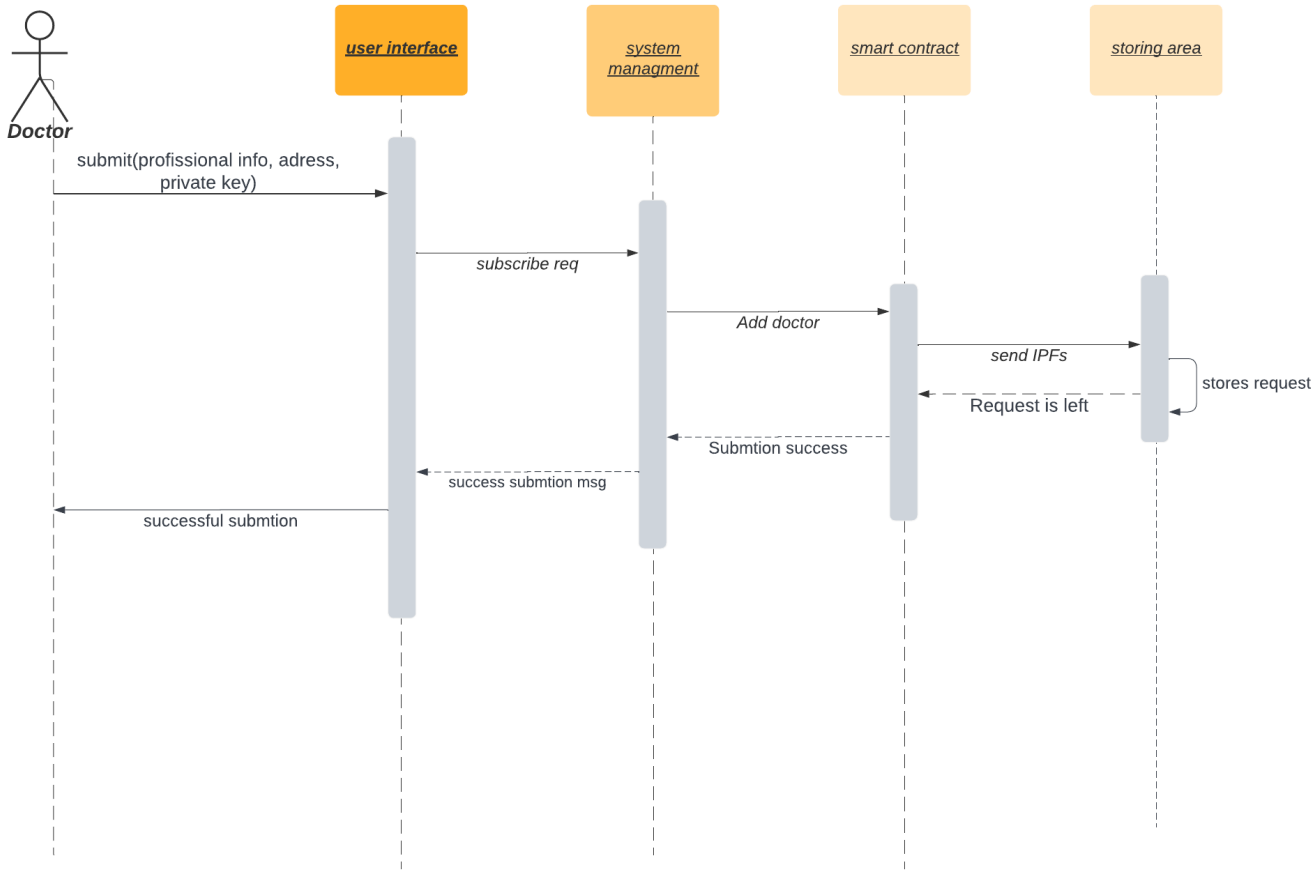
- **Patient login**



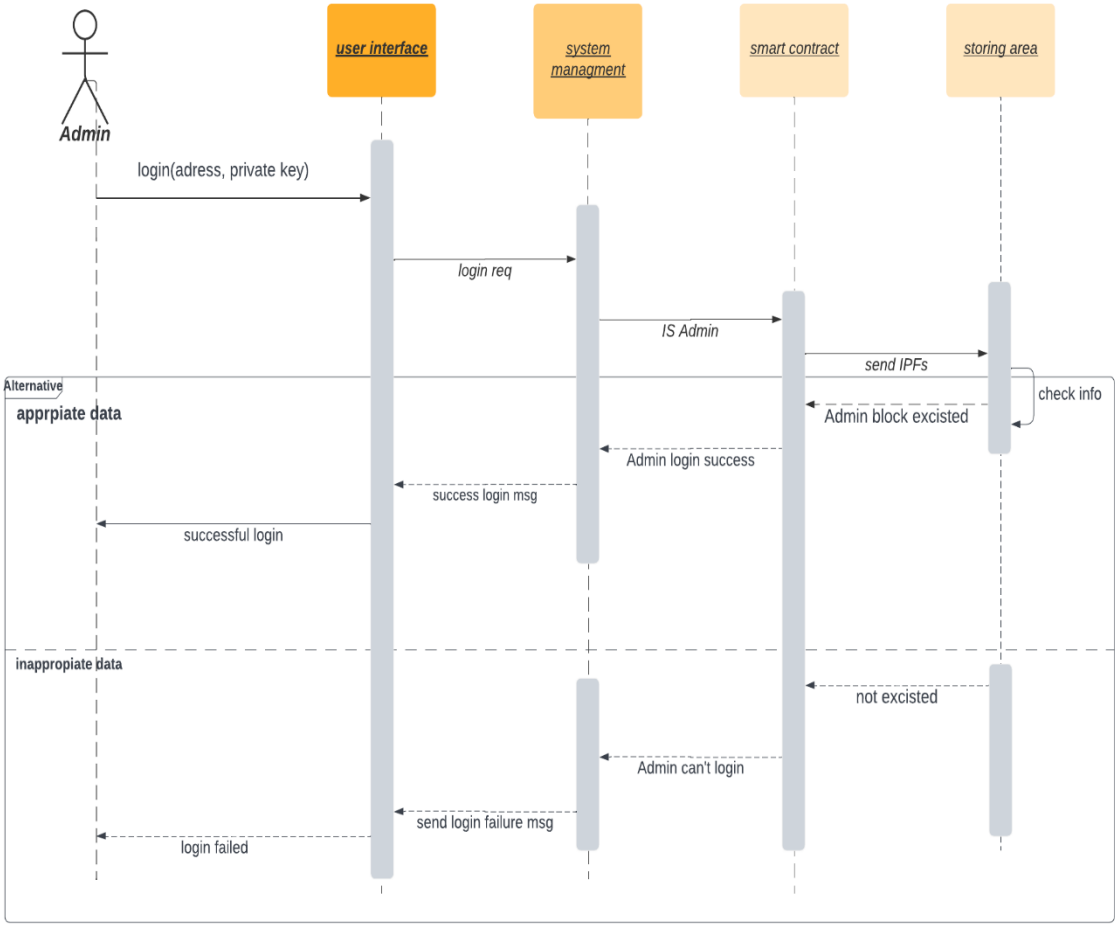
- **Doctor/Lab-Doctor login**



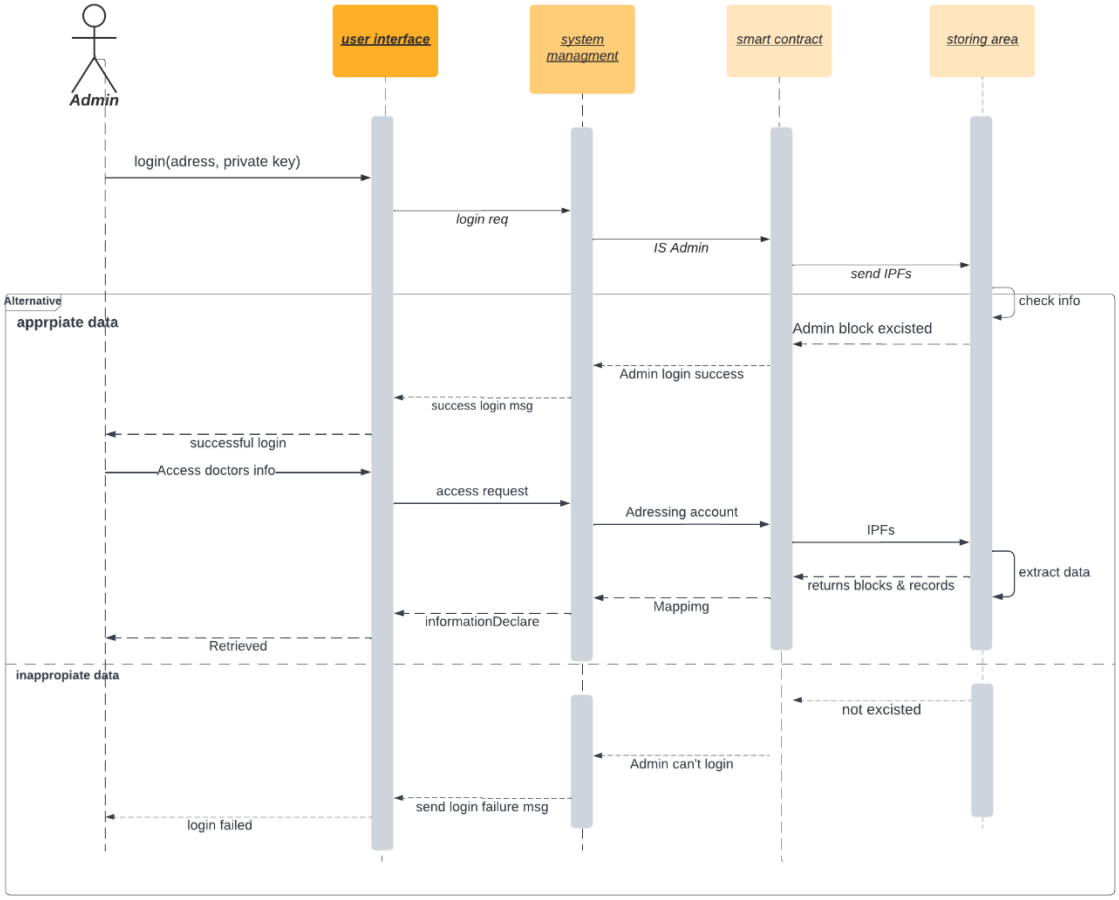
- **Doctor/Lab-doctor submits**



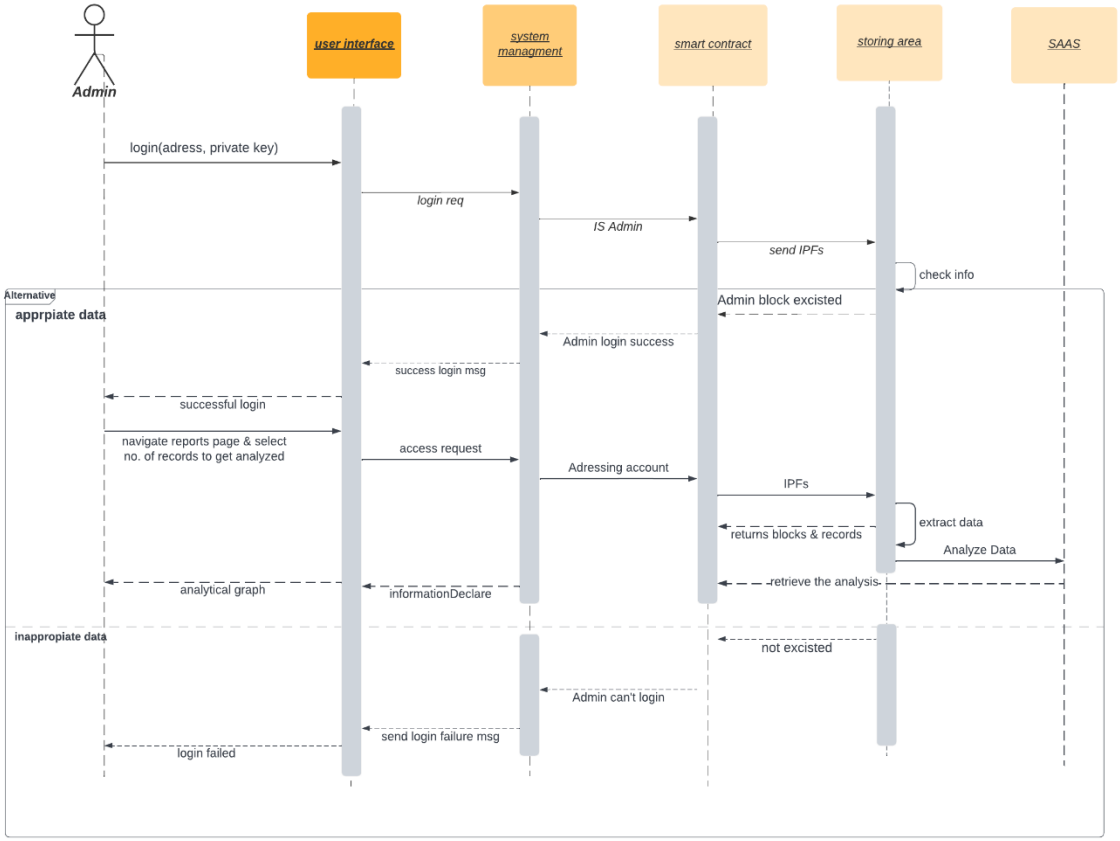
- **Admin login**


Admin
user interface
system managment
smart contract
storing area
login(adress, private key)
login req
IS Admin
send IPFs
check info
Admin block excisted
Admin login success
success login msg
successful login
Alternative
apprpiate data
inappropiate data
not excisted
Admin can't login
send login failure msg
login failed

- **Admin Assign patient /Add/modify doctors**

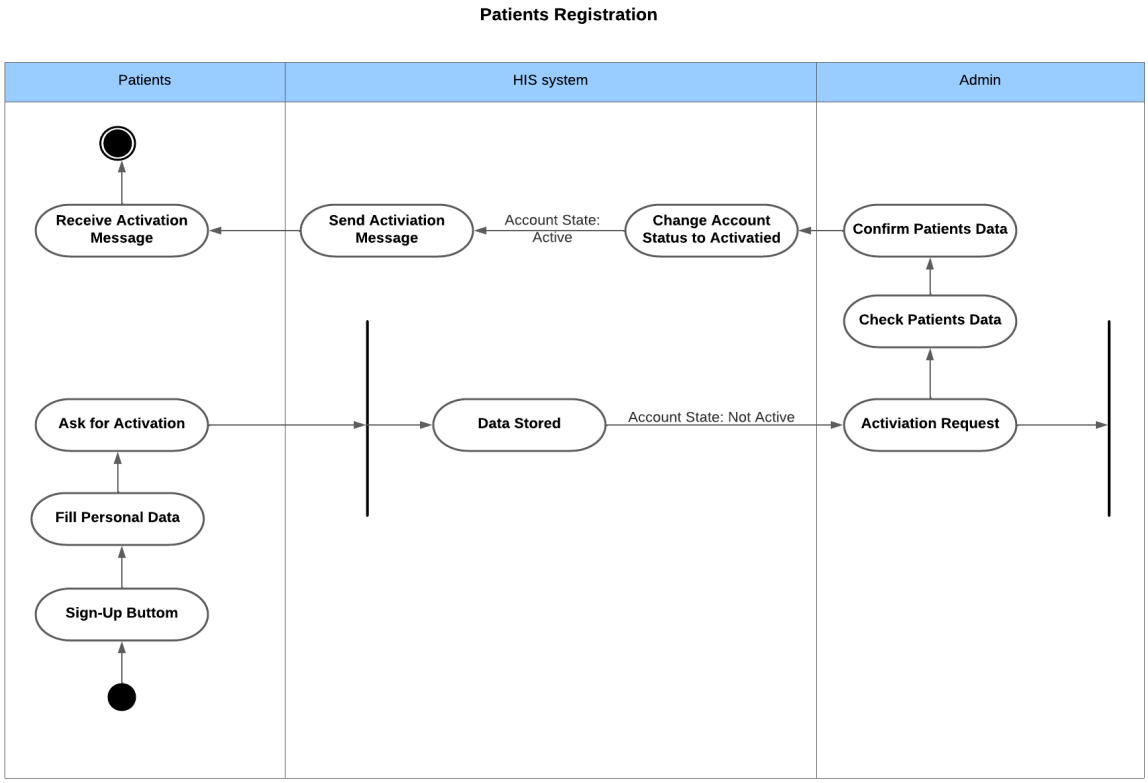


- **Admins generate report**

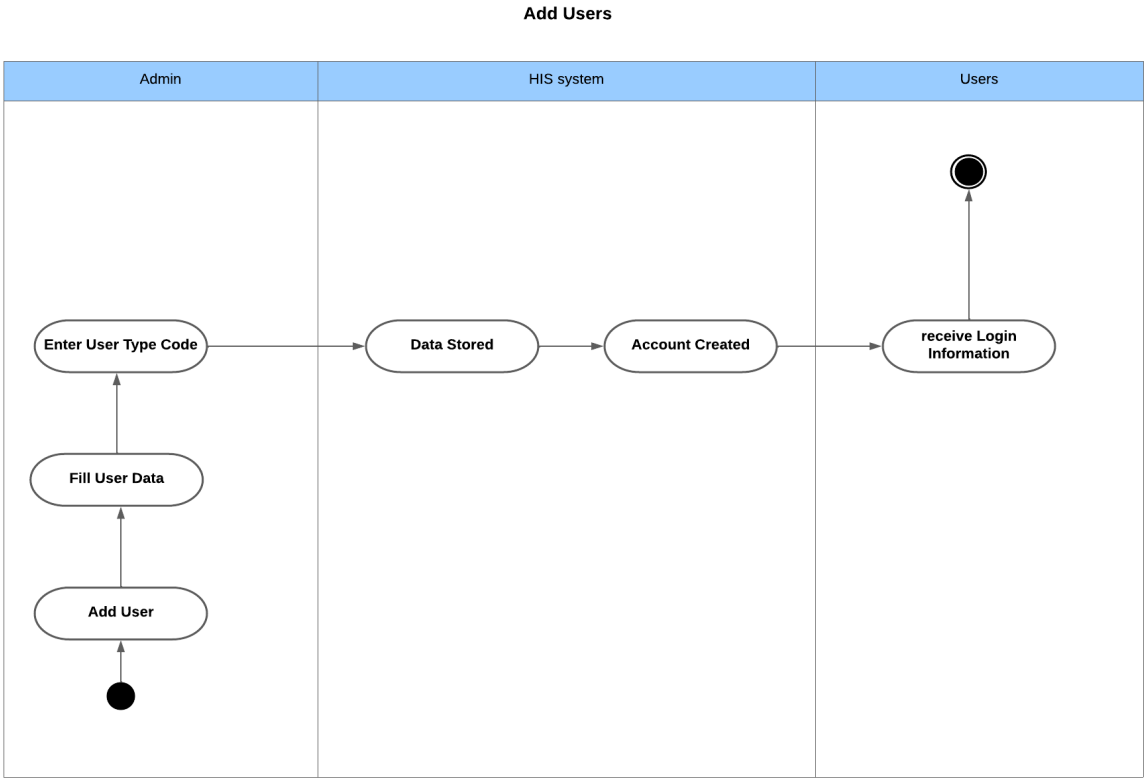


USE-CASES Activity Diagrams

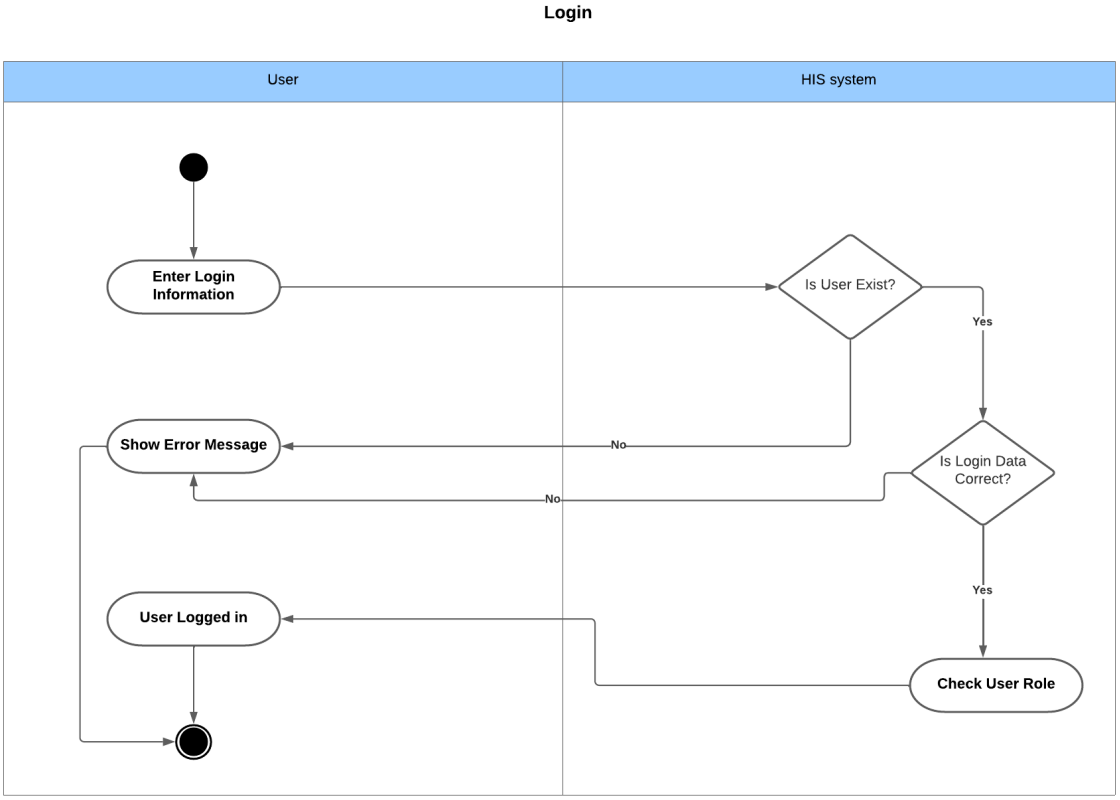
Patients Registration:



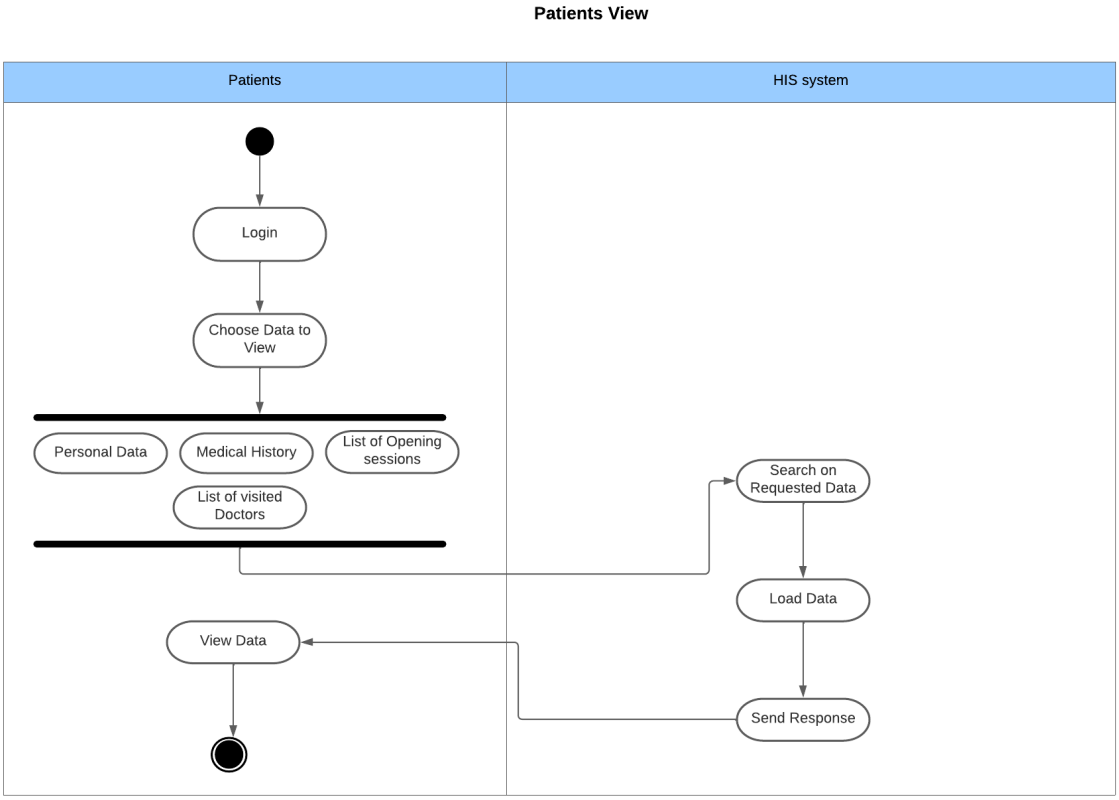
Add Users [Doctors/Lab-Doctors]:



Login:

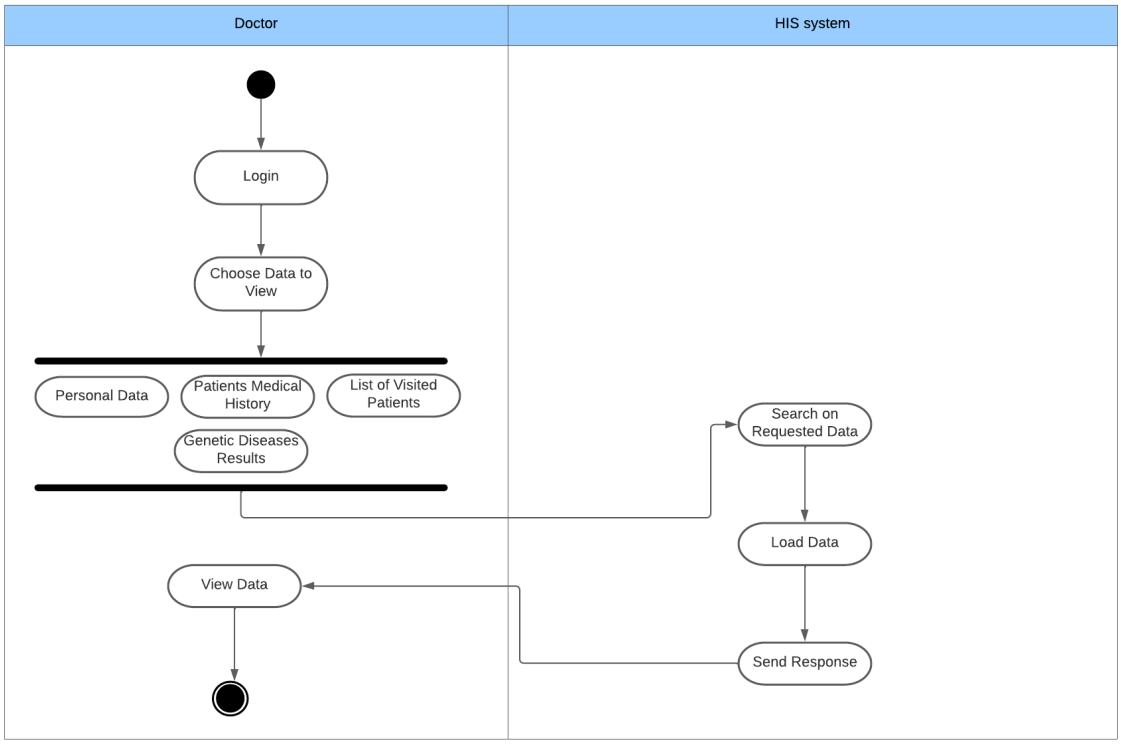

Login
User
HIS system
Enter Login Information
Is User Exist?
Yes
Show Error Message
No
Is Login Data Correct?
No
Yes
User Logged in
Check User Role

Patients View:

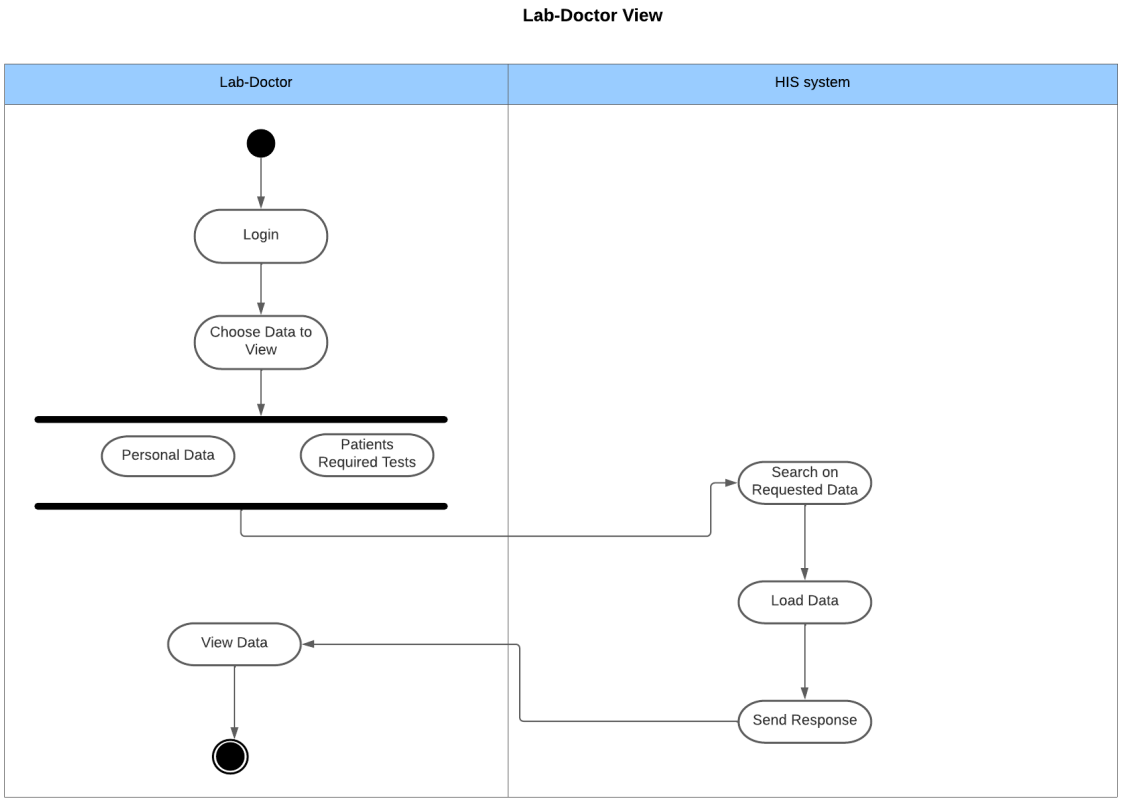


Doctor View:

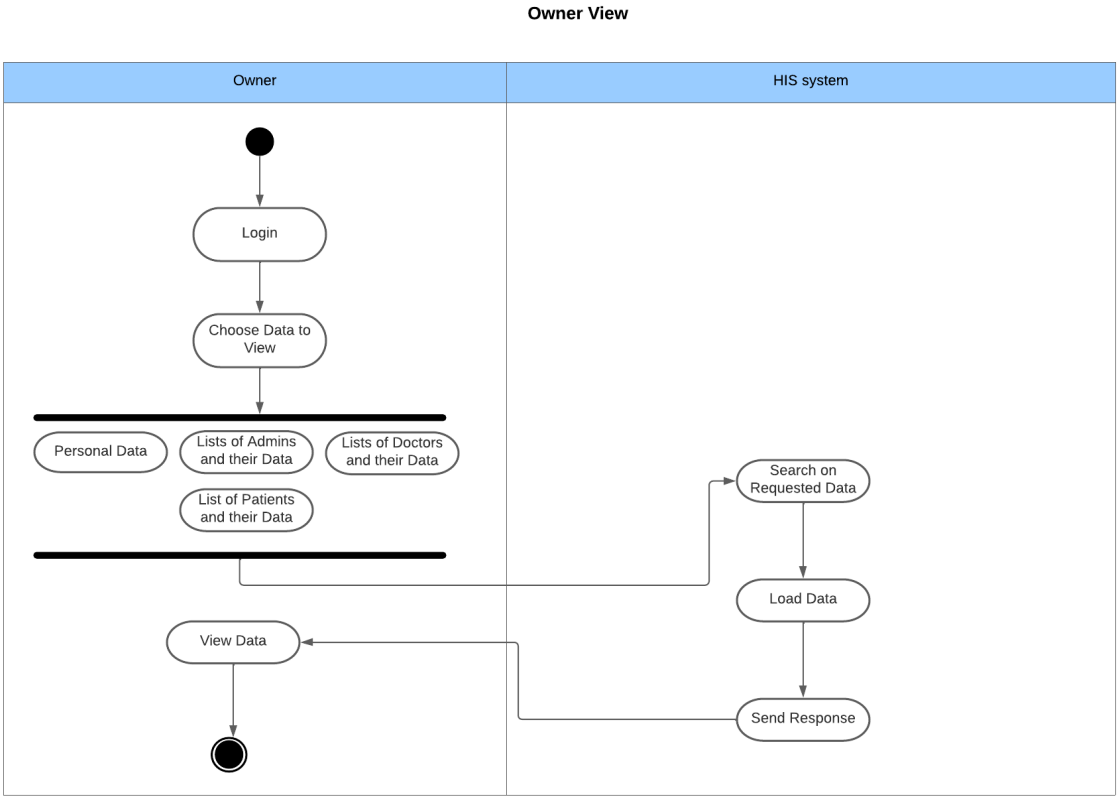
Doctor View



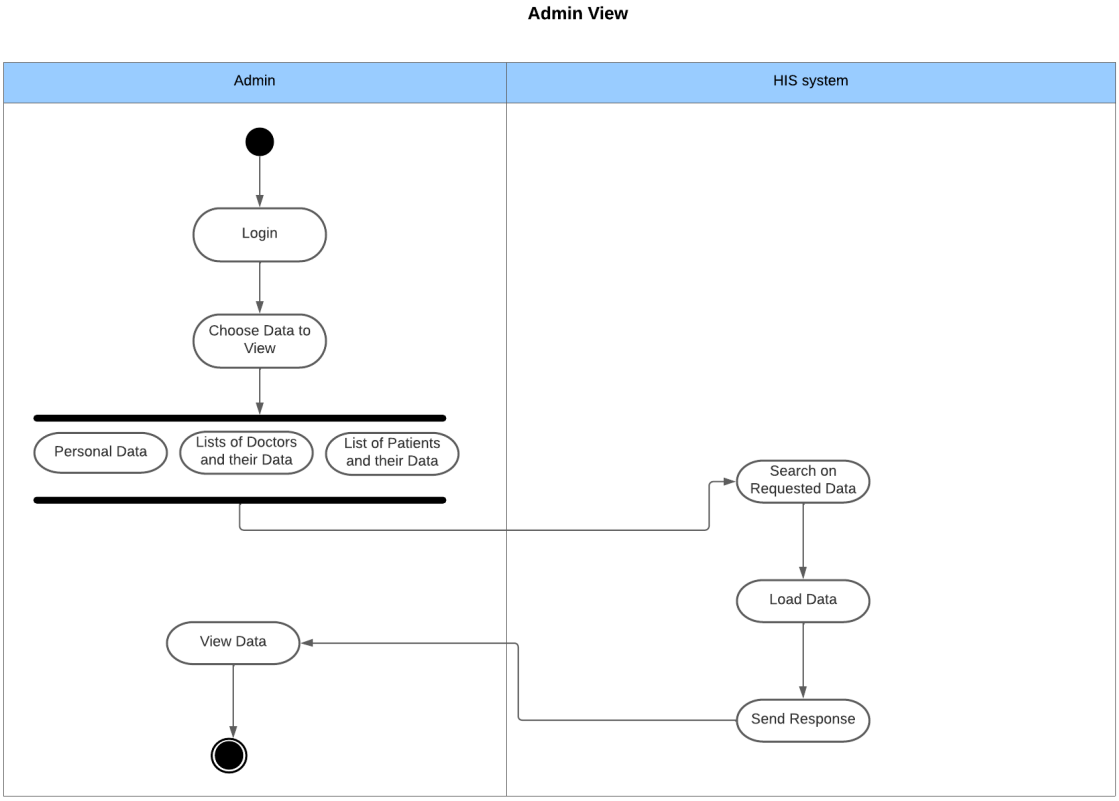
Lab-Doctor View:



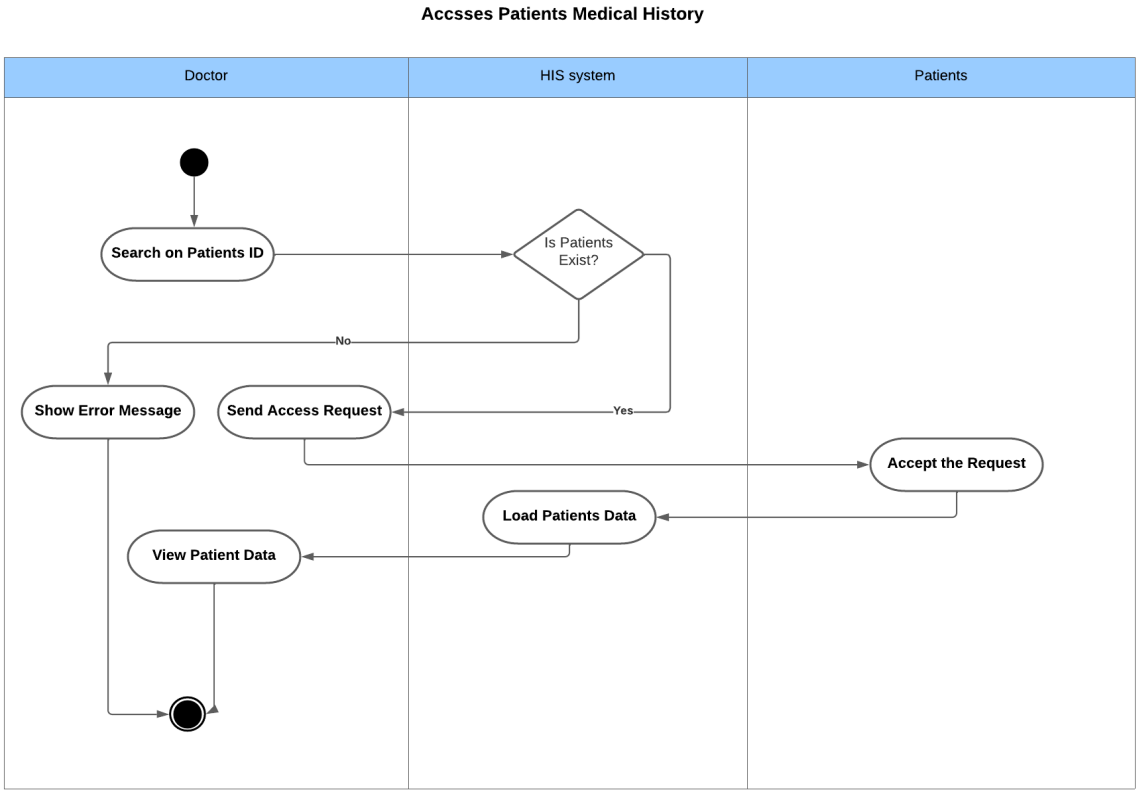
Owner View:


Owner View
Owner
HIS system
Login
Choose Data to View
Personal Data
Lists of Admins and their Data
Lists of Doctors and their Data
List of Patients and their Data
Search on Requested Data
Load Data
View Data
Send Response

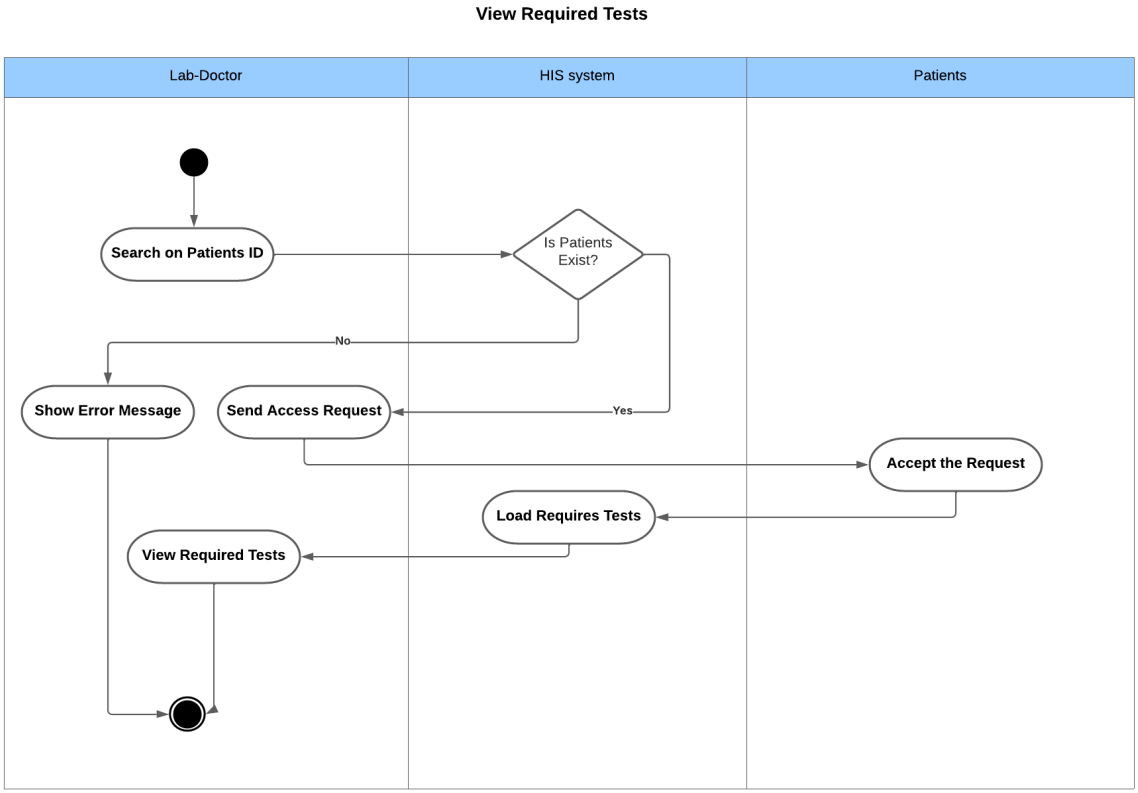
Admin View:


Admin View
Admin
HIS system
Login
Choose Data to View
Personal Data
Lists of Doctors and their Data
List of Patients and their Data
Search on Requested Data
Load Data
View Data
Send Response

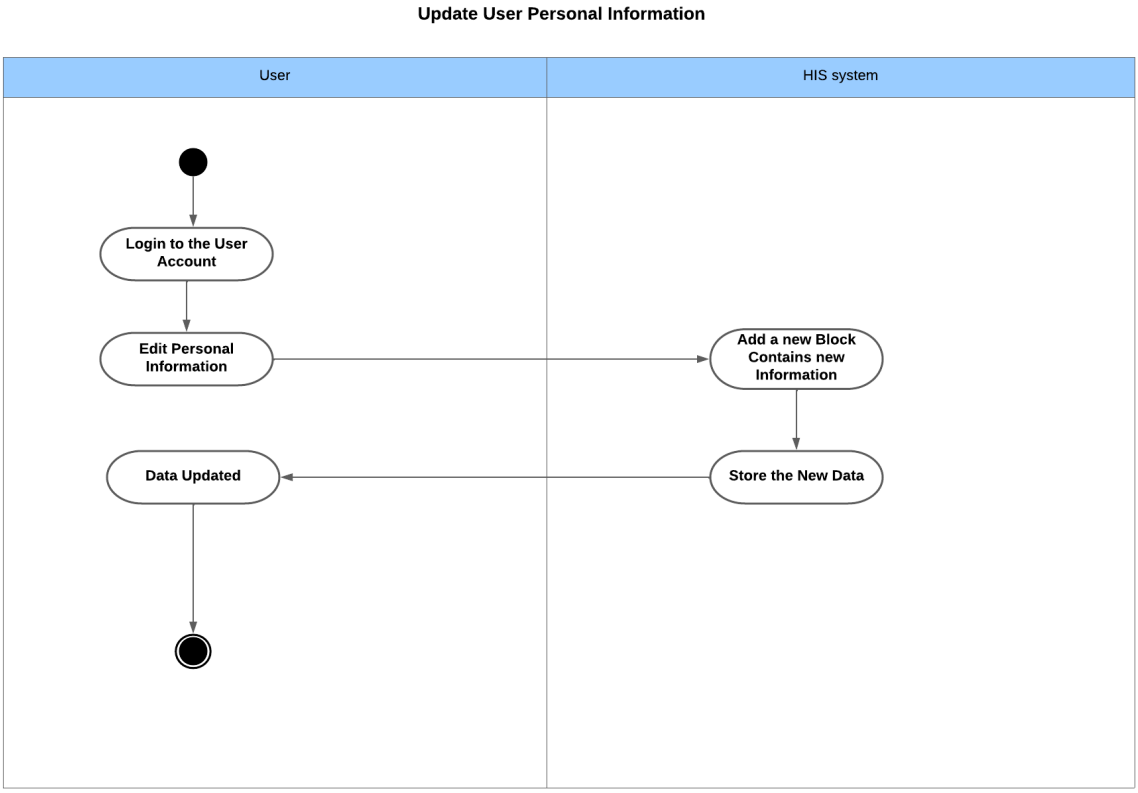
Accesses Patients' Medical History:



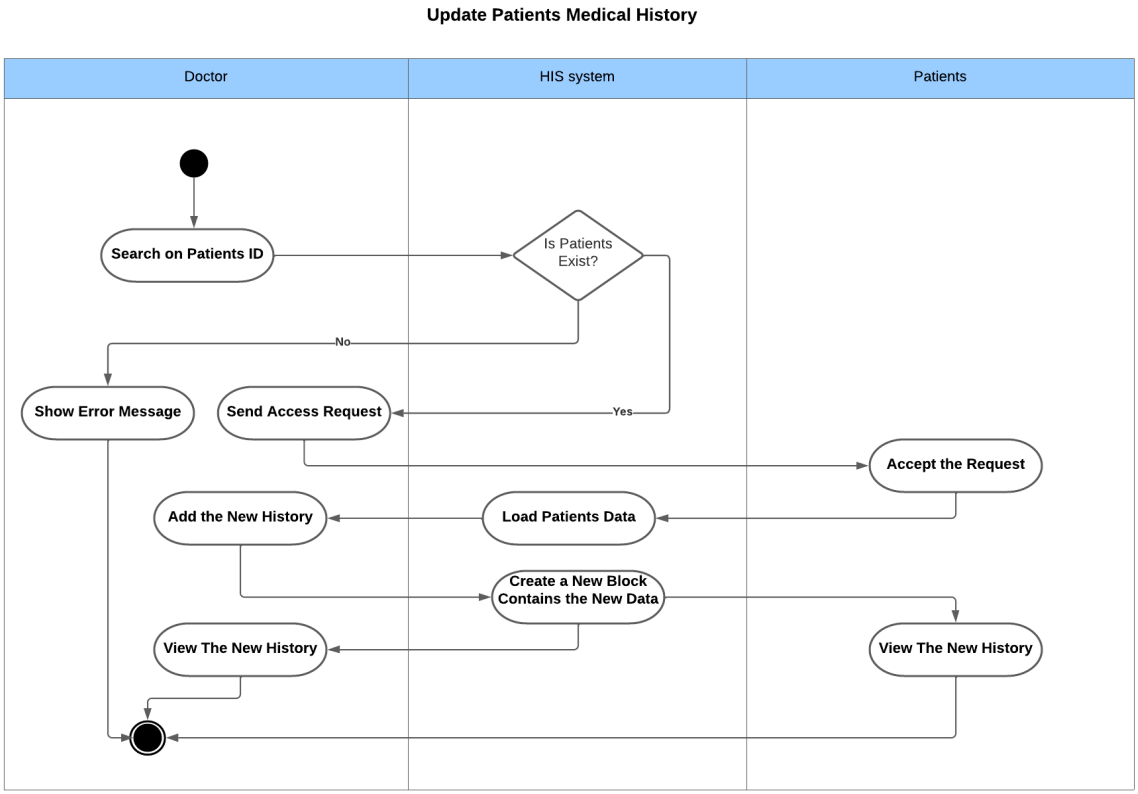
View Required Tests:



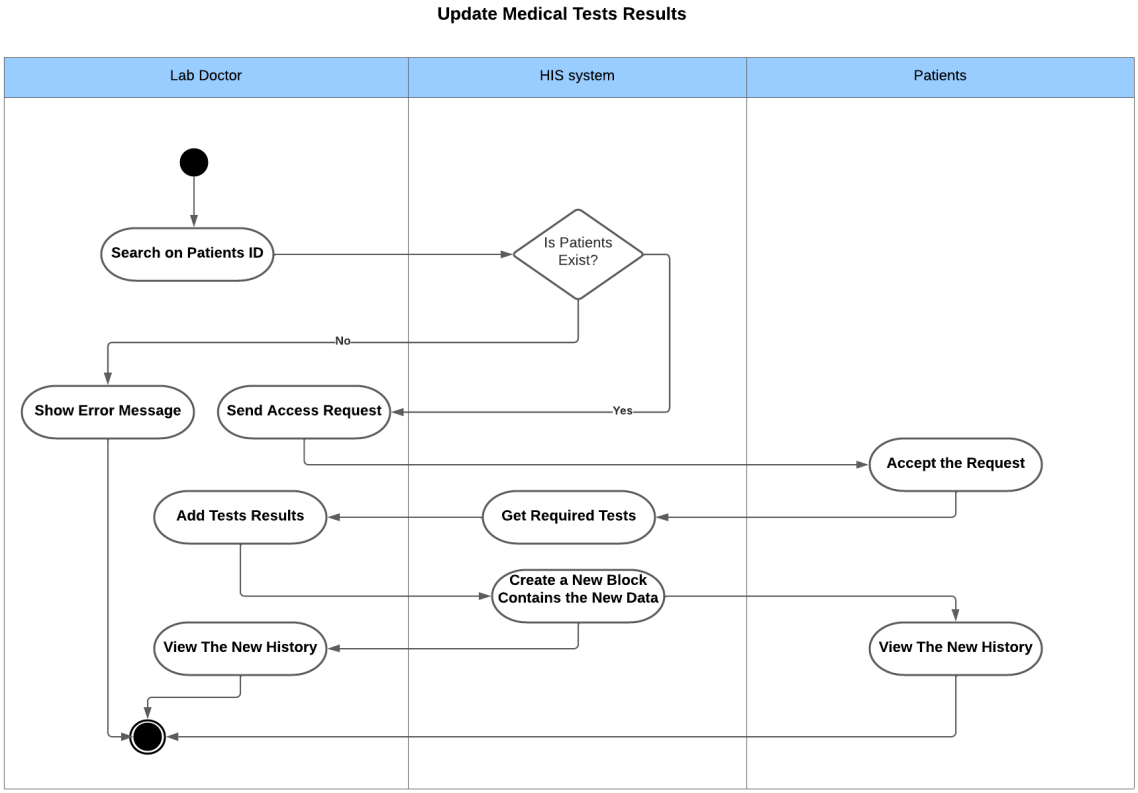
Update User Personal Information:



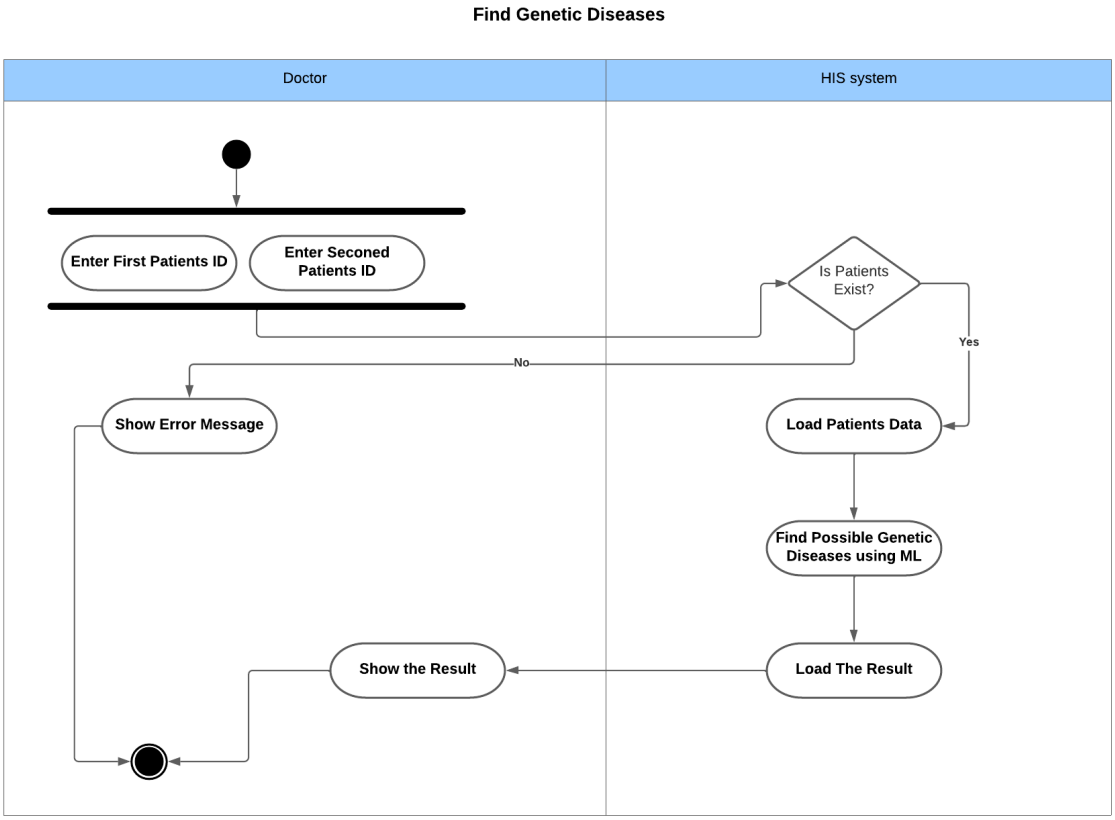
Update Patients Medical History:



Update Medical Tests Results:



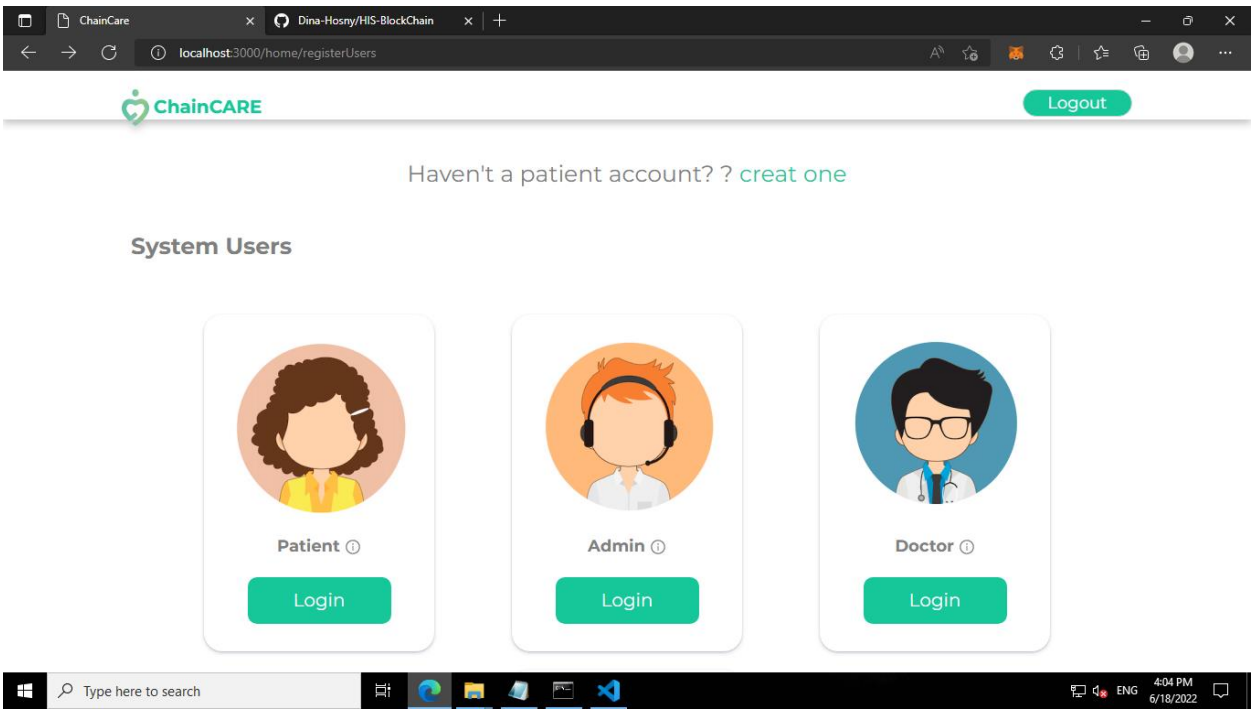
Find Genetic Diseases:



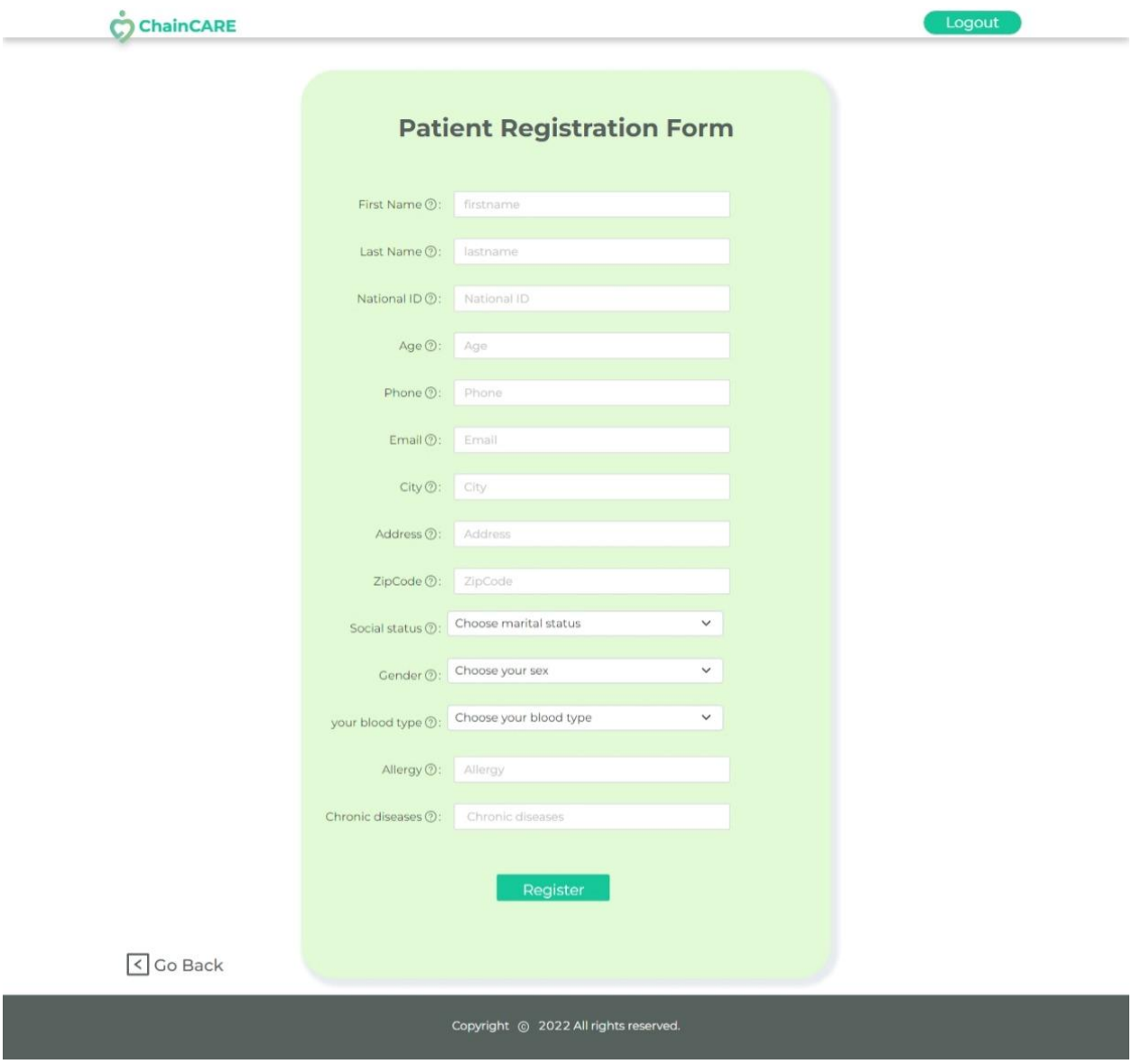
Chapter 4: Implementation

In this chapter, we are going to discuss and go deeper in ChainCare health system implementation and present its code and the algorithms used to build it.

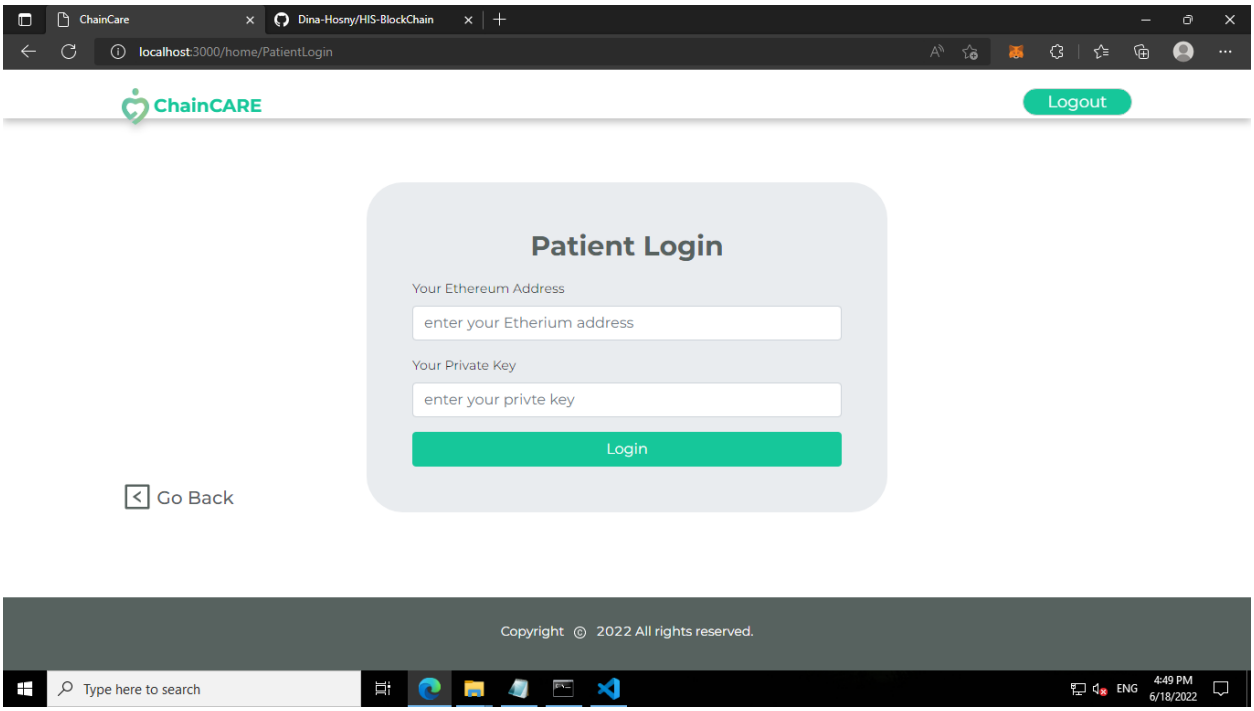

ChainCARE
About
OverView
Contact
Connect
ChainCARE
Let's make your life happier with Healthy Living
Connect
Medical Staff Registration
About
Read to know some INFO about US
WELCOME TO YOUR HEALTH CARE
Learn More
Demo
This video explains the technology behind this application and walk through for how to use it.
Blockchain Technology for EHRs
App Presentation
describing the project and the technology behind it
Contact Us
name
email
message
Send
ChainCare
Copyright © 2022 All rights reserved


ChainCare
Dina-Hosny/HIS-BlockChain
localhost:3000/home/registerUsers
ChainCARE
Logout
Haven't a patient account? ? creat one
System Users
Patient
Login
Admin
Login
Doctor
Login
Type here to search
ENG
4:04 PM
6/18/2022

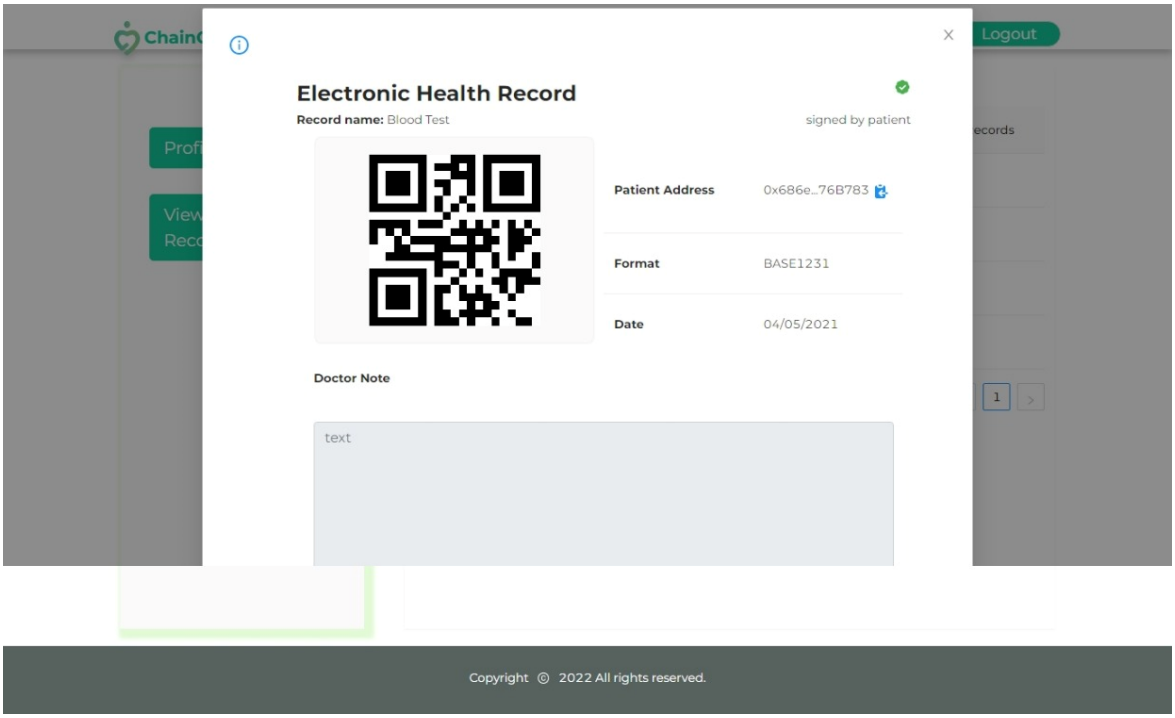
Patient registration



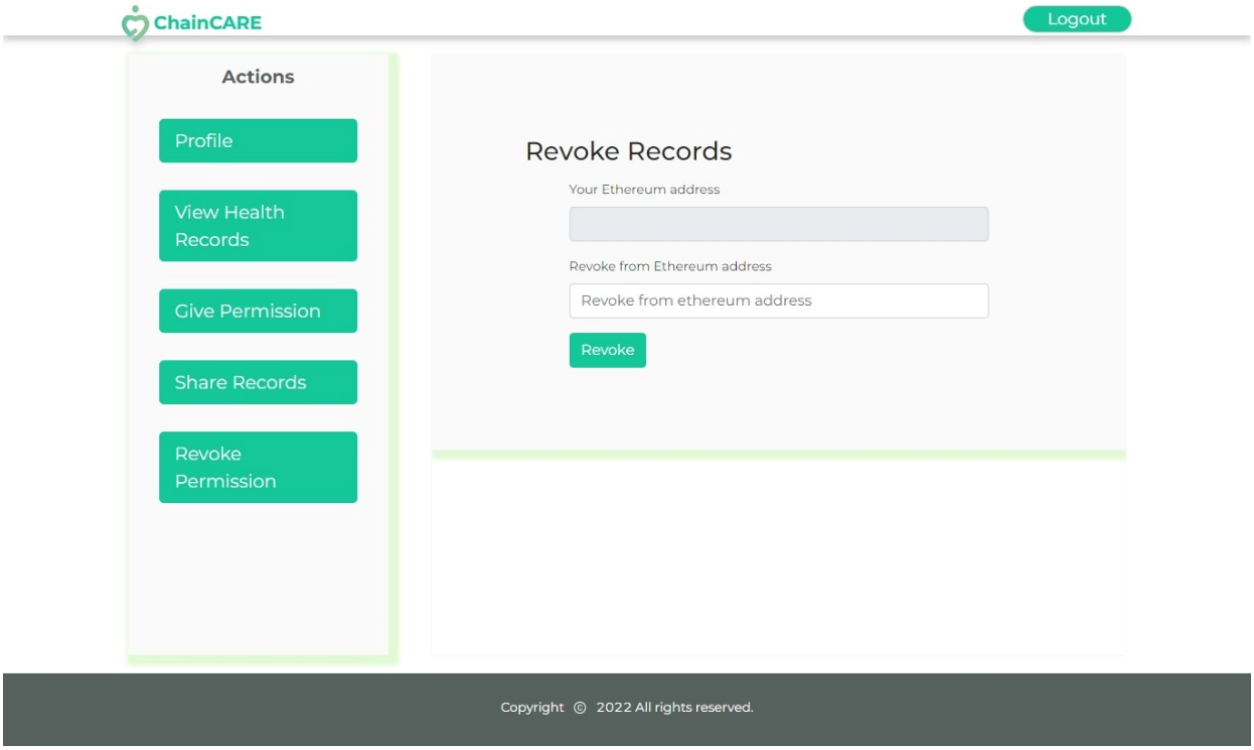
Patient login



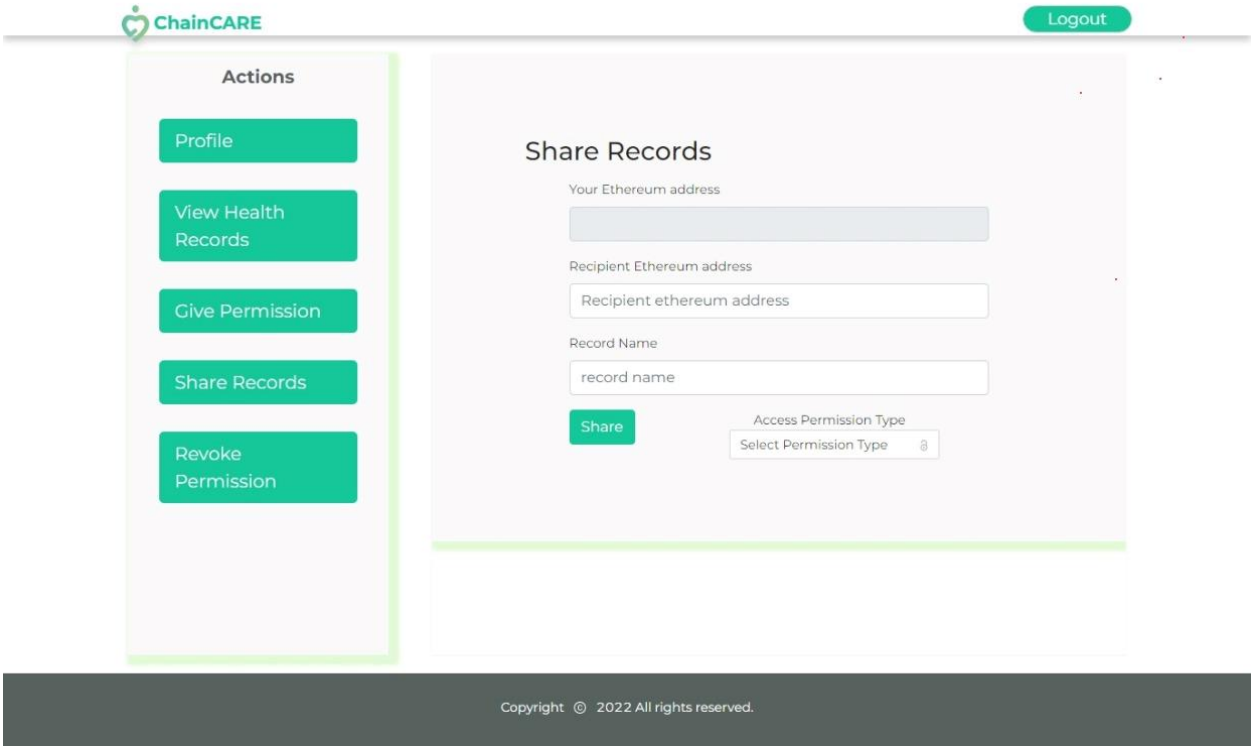
Pateint medical record



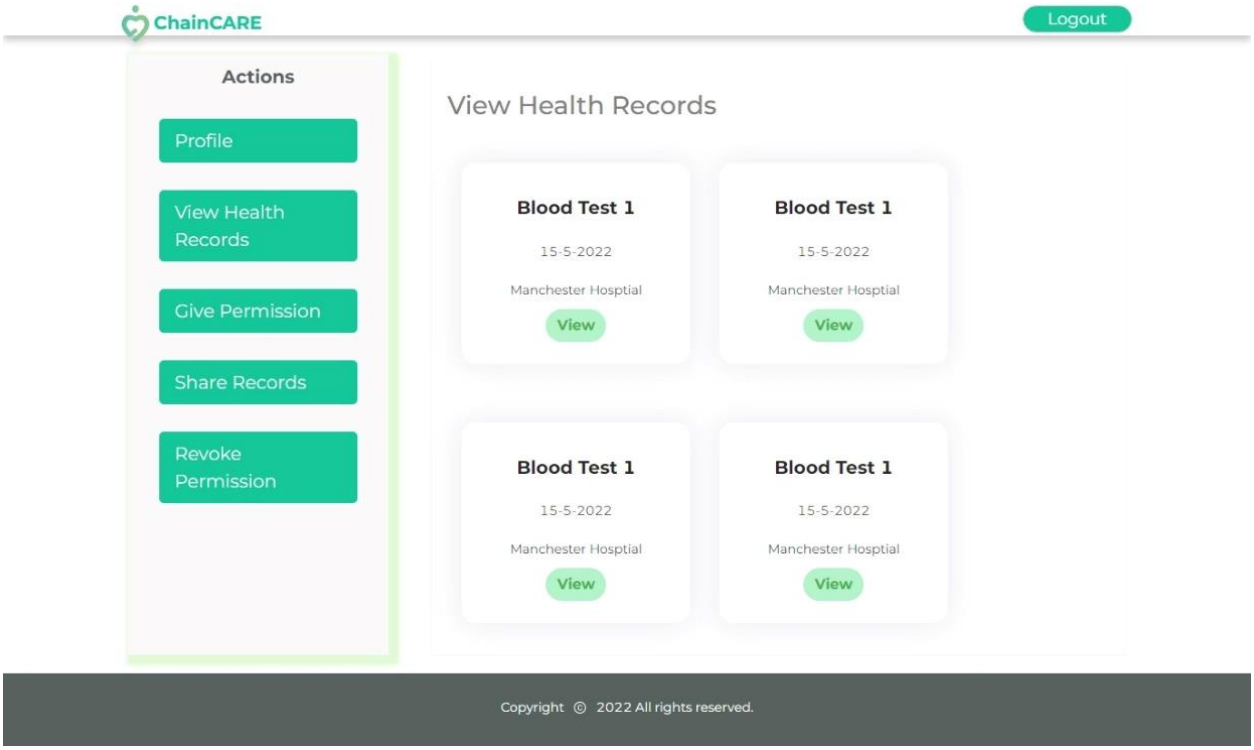
Patient revoke record



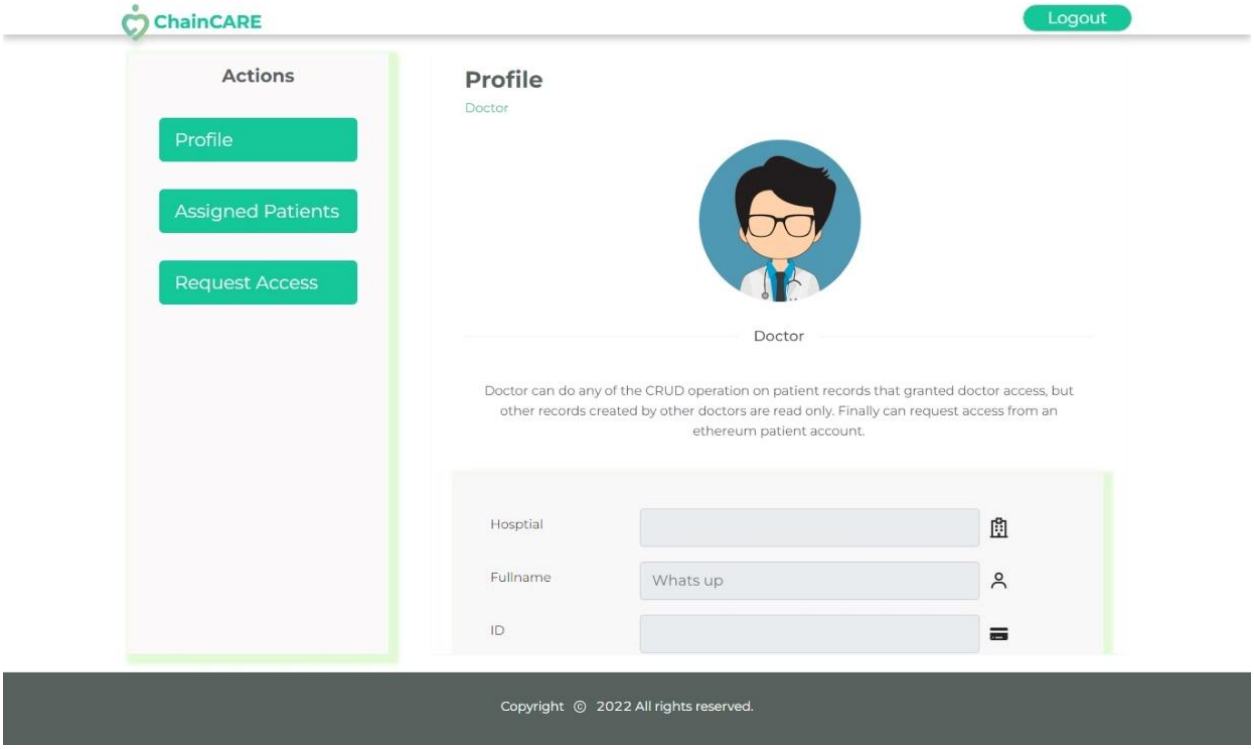
Patient shares record



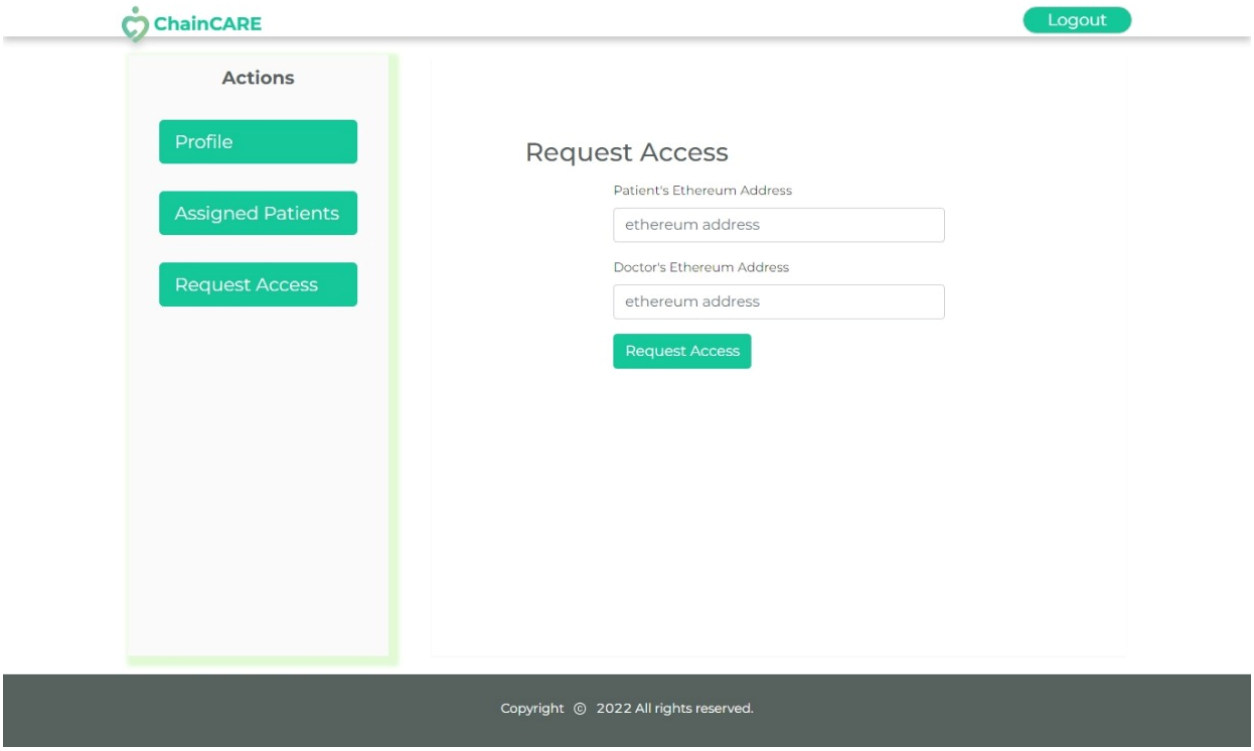
Patient views Record



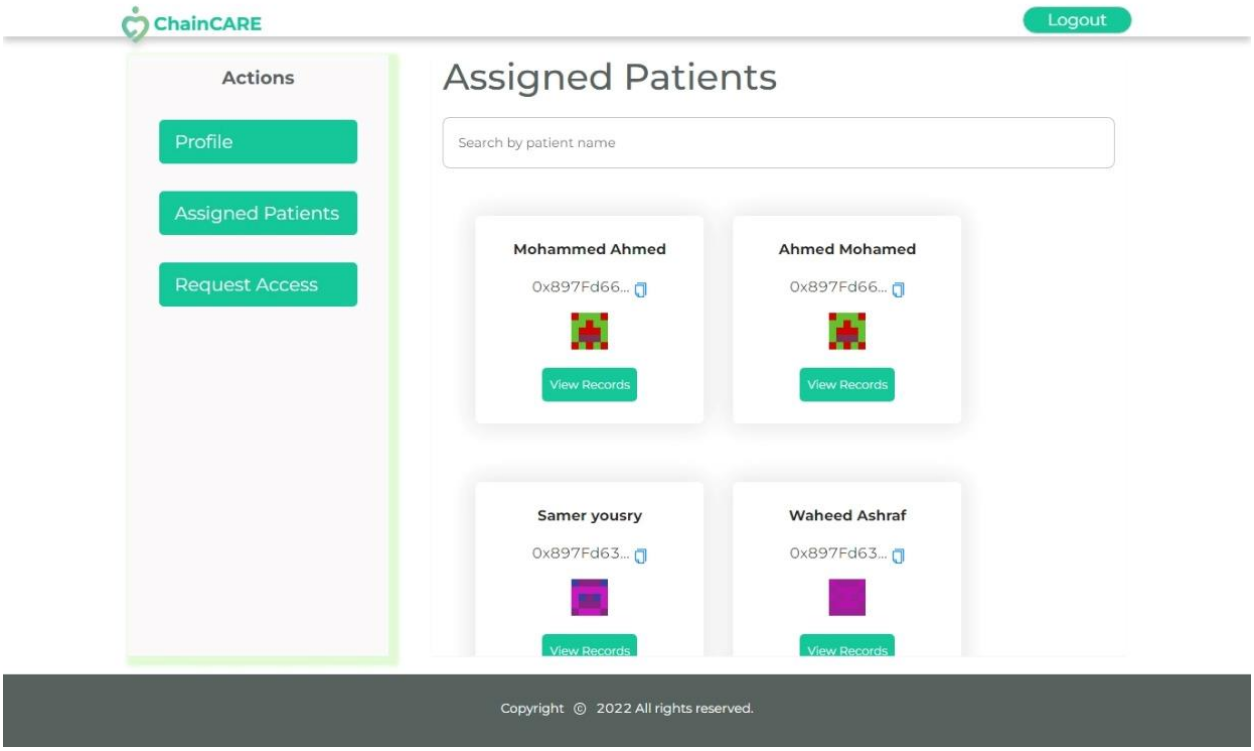
Doctor's profile



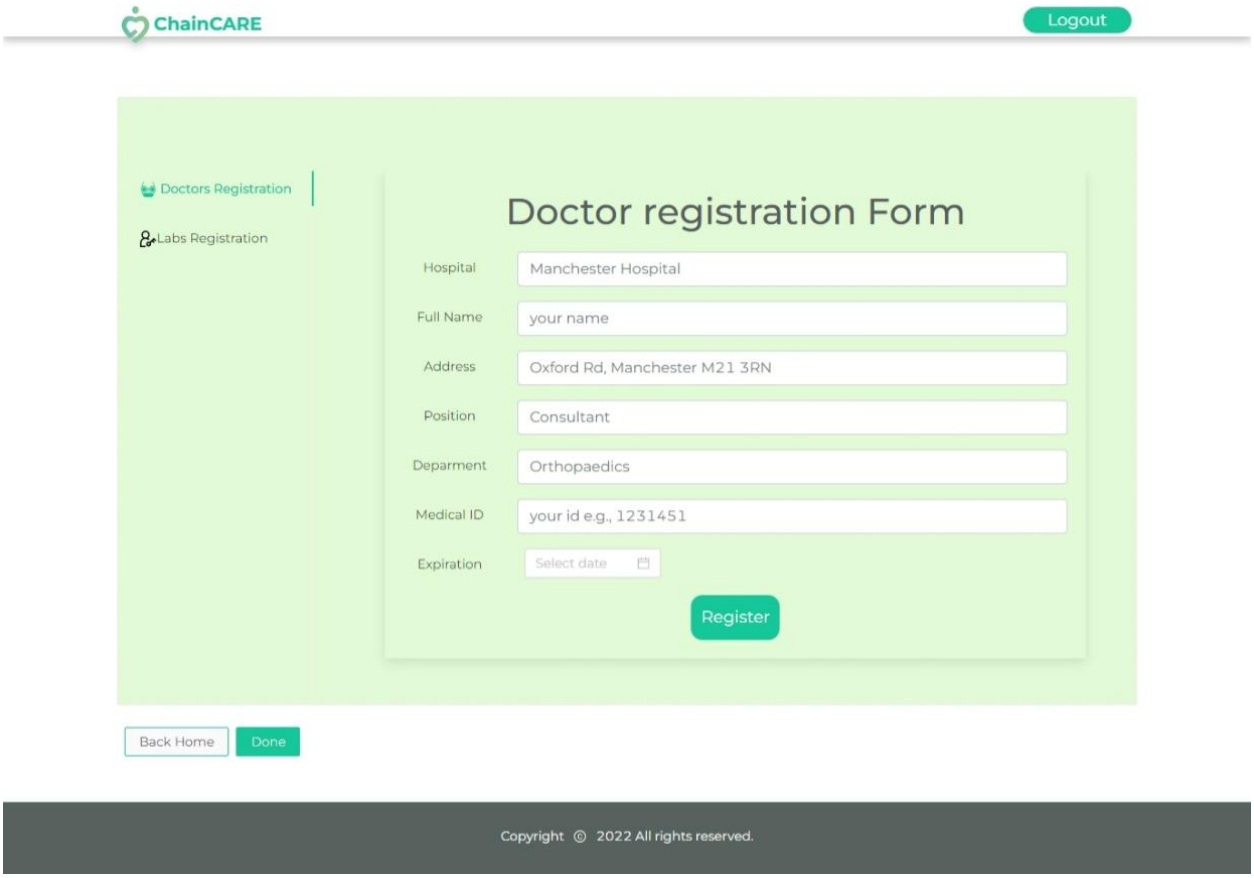
Doctors request access



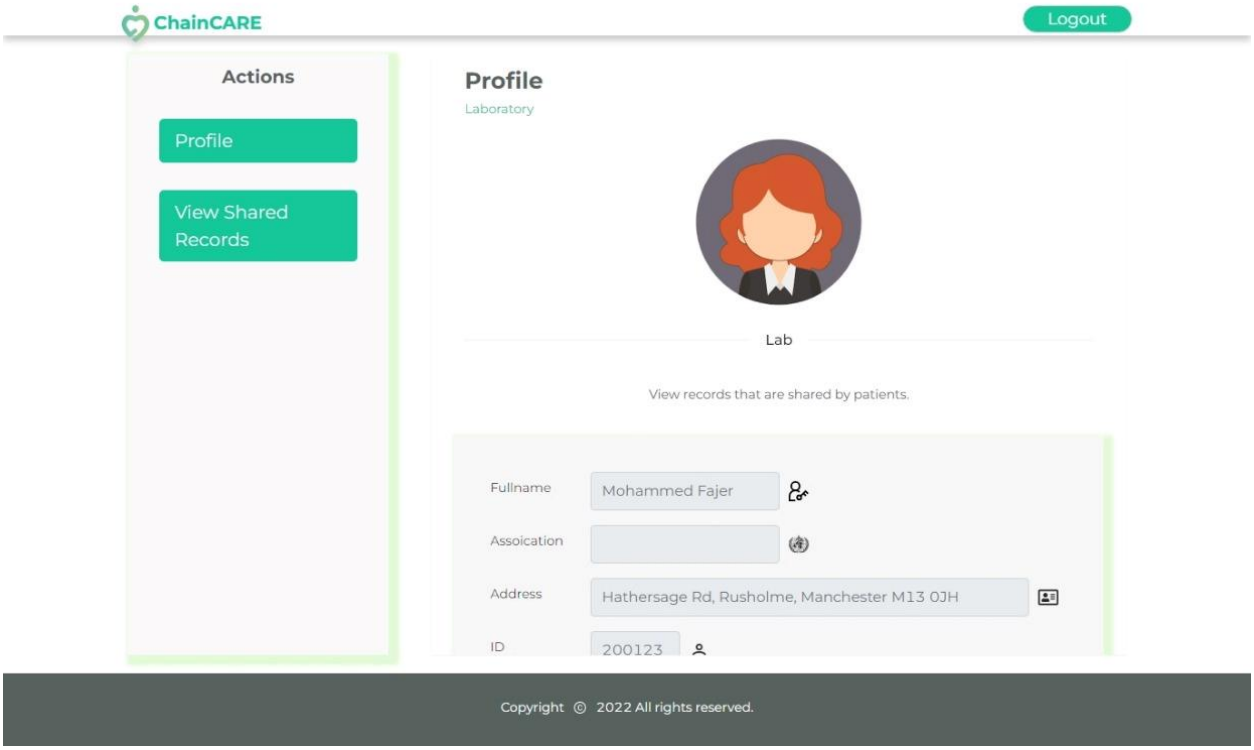
Doctors Assigned patients



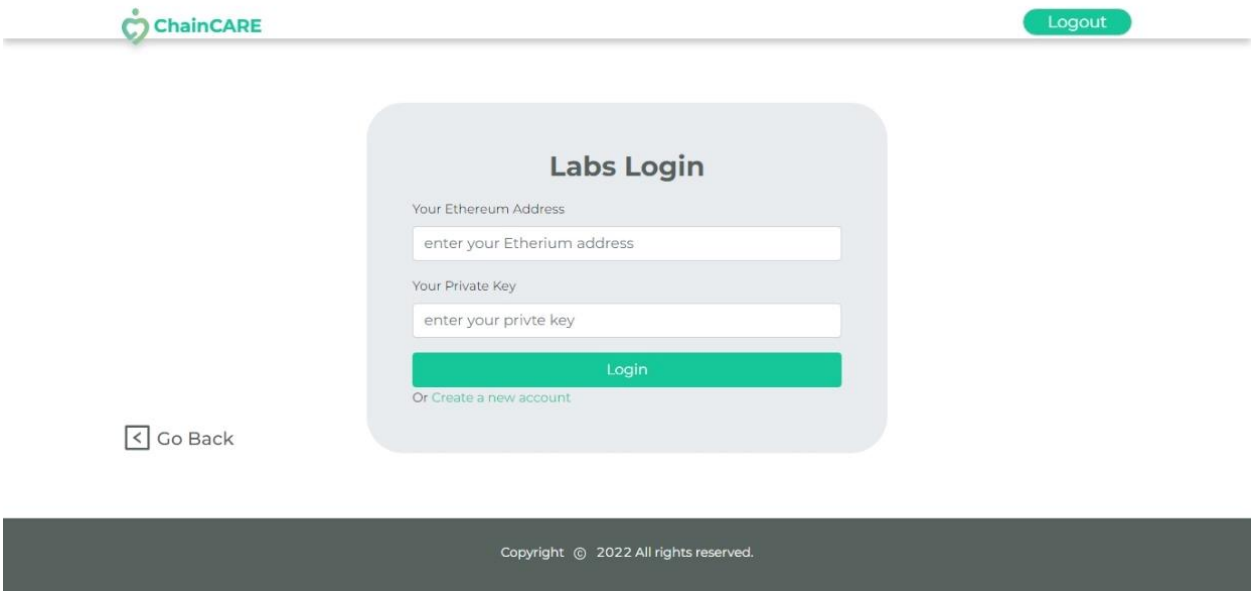
Doctors/Laboratories registration form



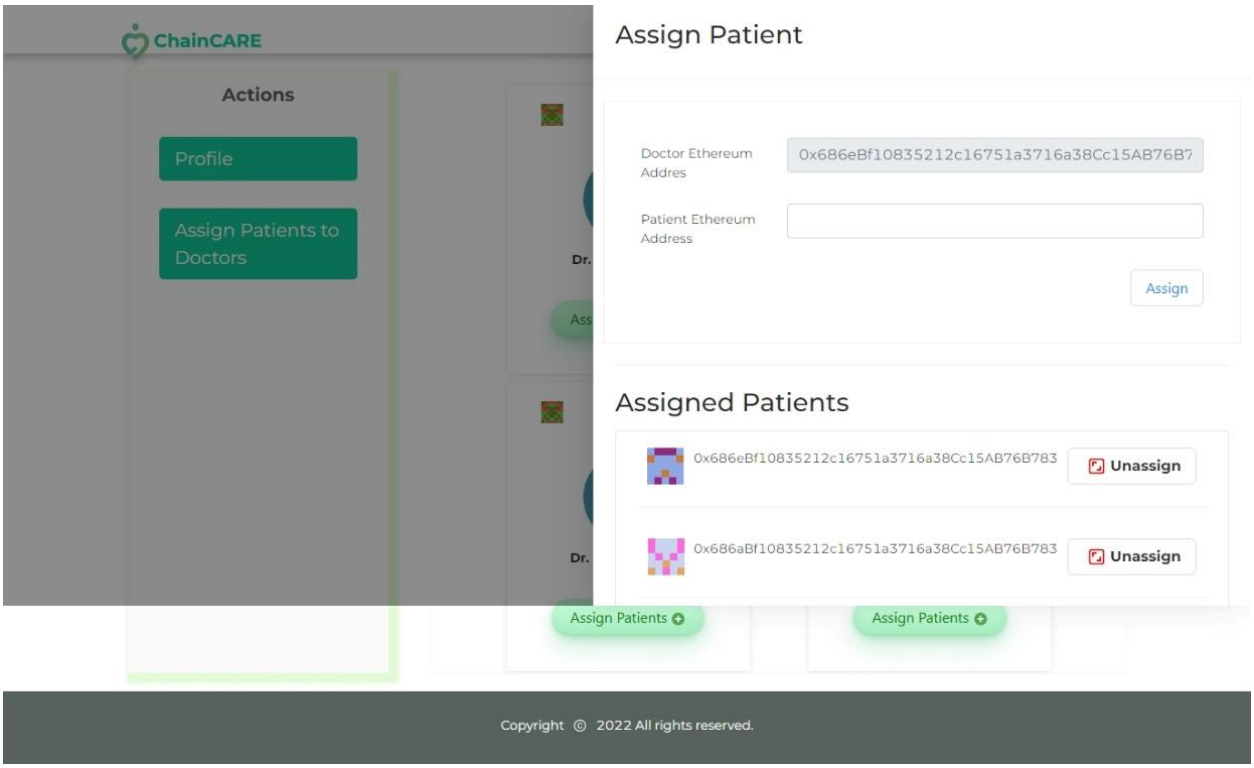
Lab-doctor’s profile



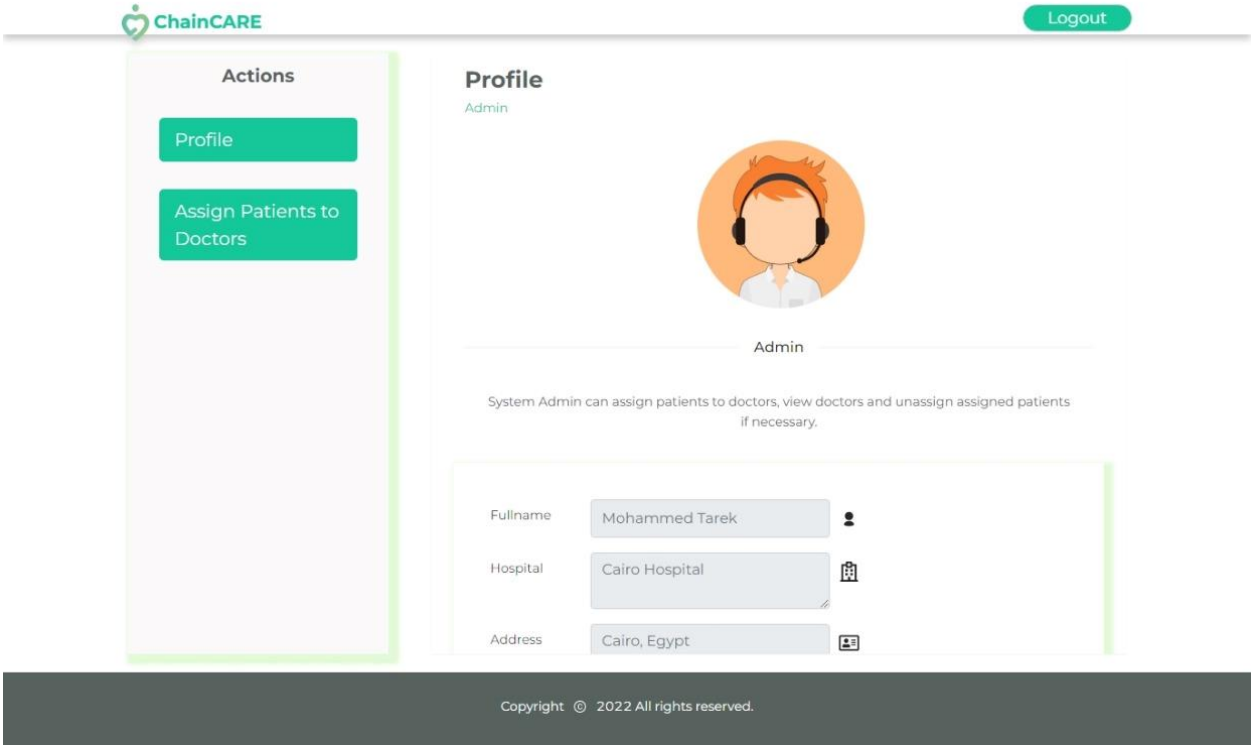
Laboratory login



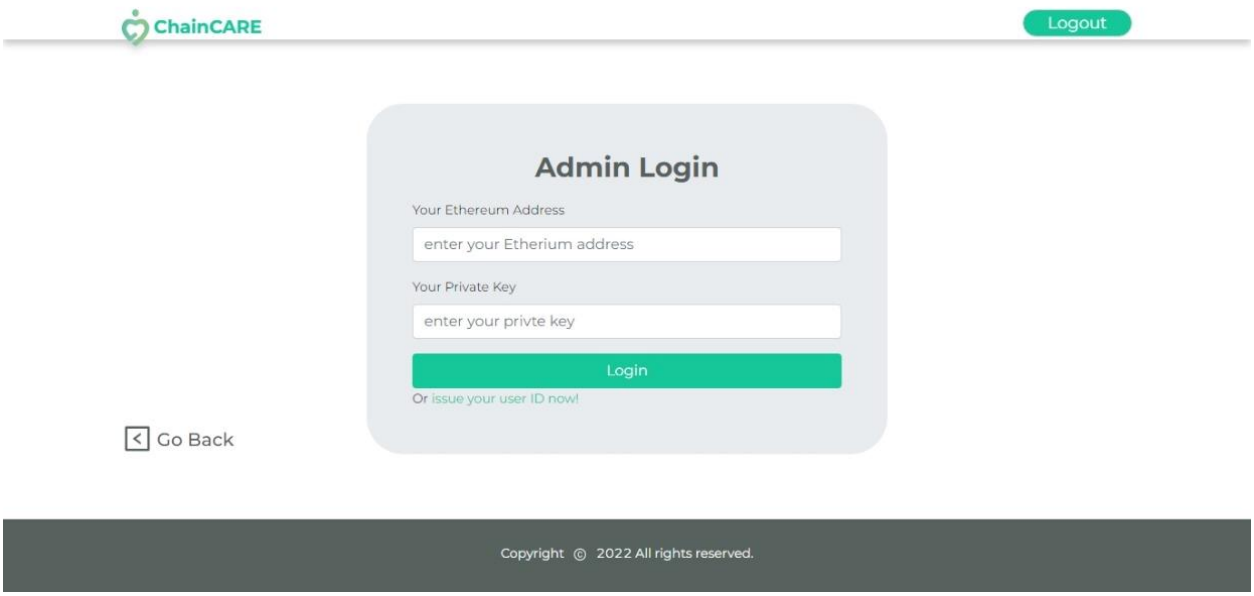
Doctors/Labs Assign patient



Admin profile

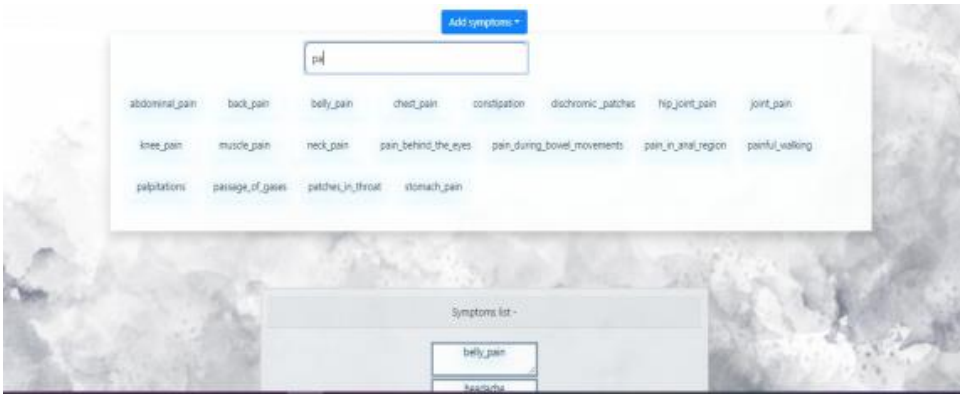


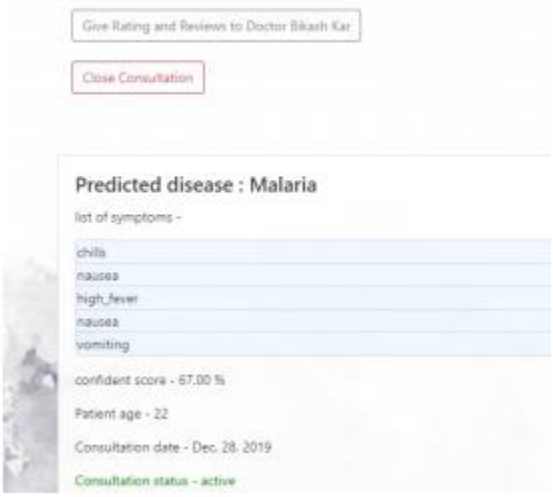
Admin login


ChainCARE
Logout
Admin Login
Your Ethereum Address
enter your Etherium address
Your Private Key
enter your privte key
Login
Or issue your user ID now!
Go Back
Copyright © 2022 All rights reserved.

Admin view medical reports

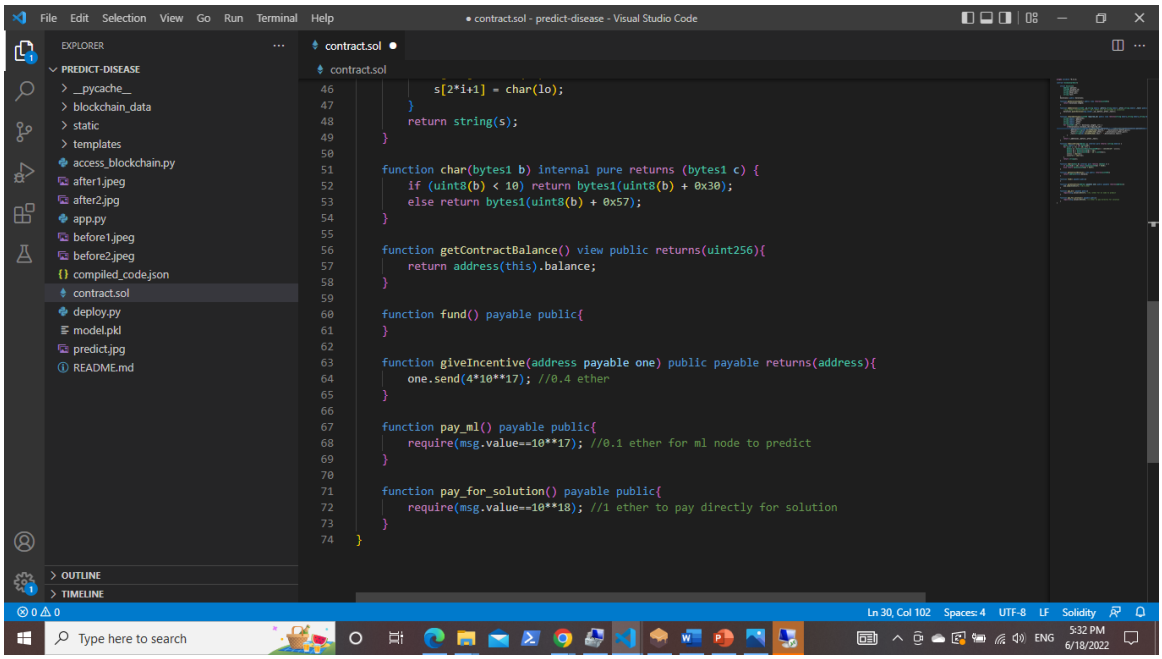
Predict of diseases

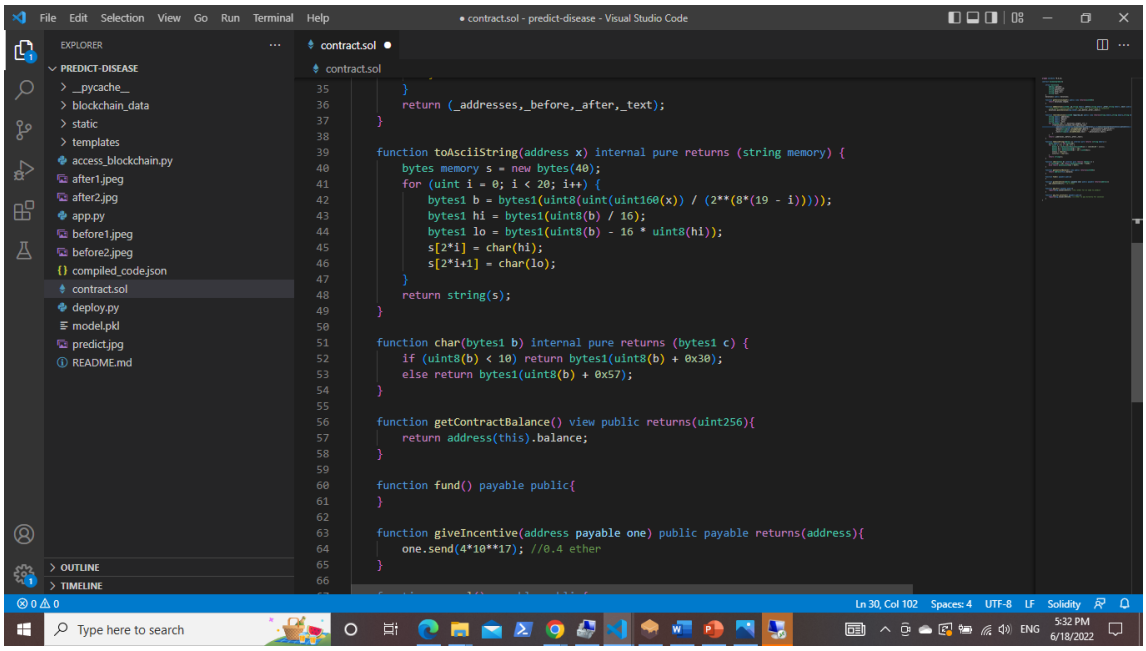


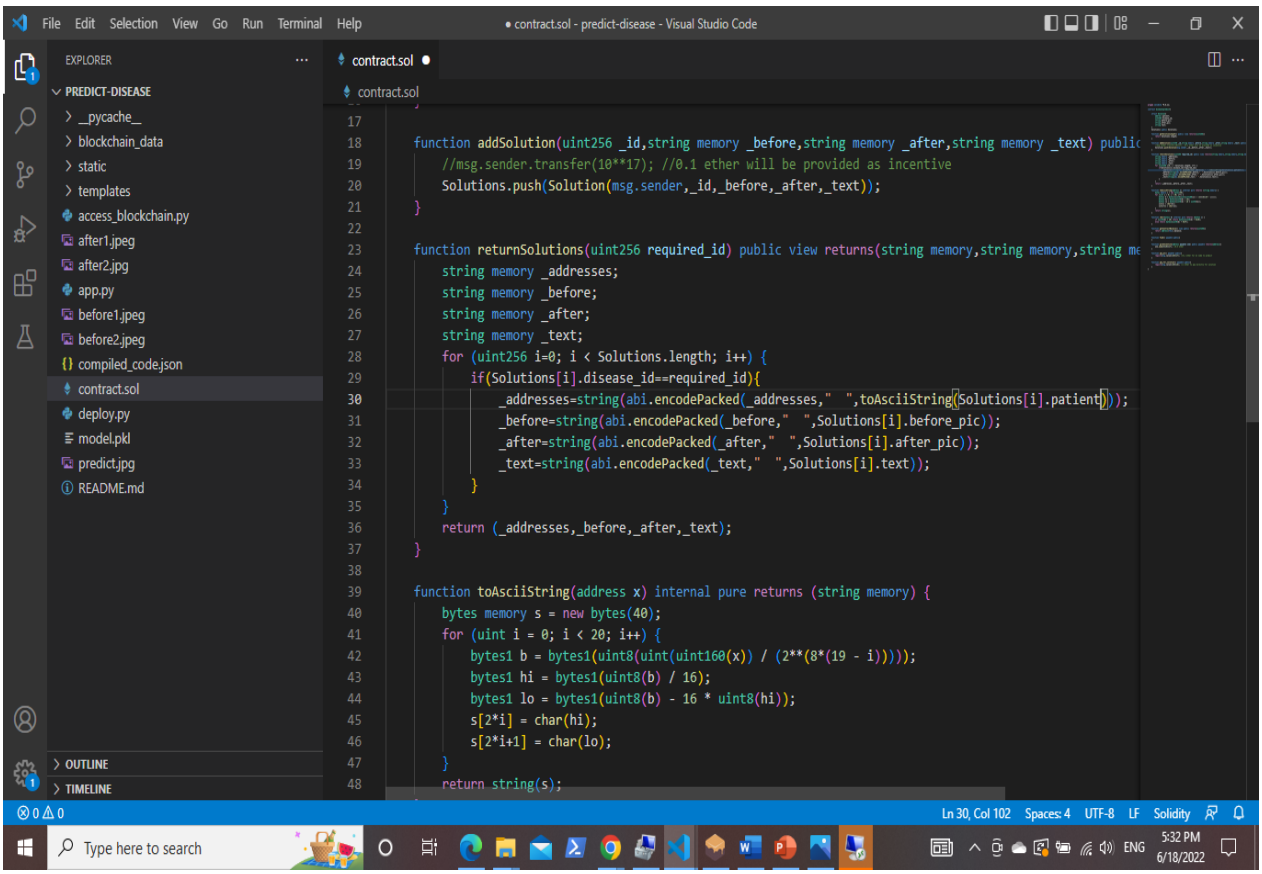


Back-end snippets

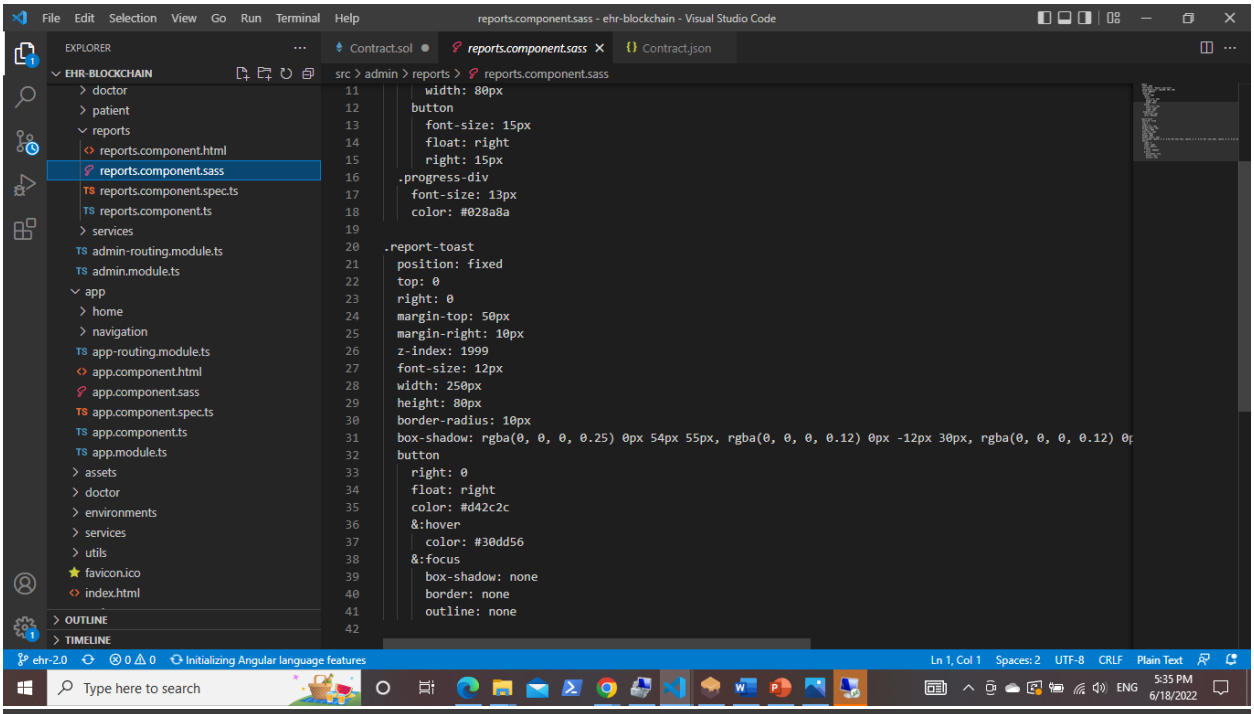
Disease prediction transactions

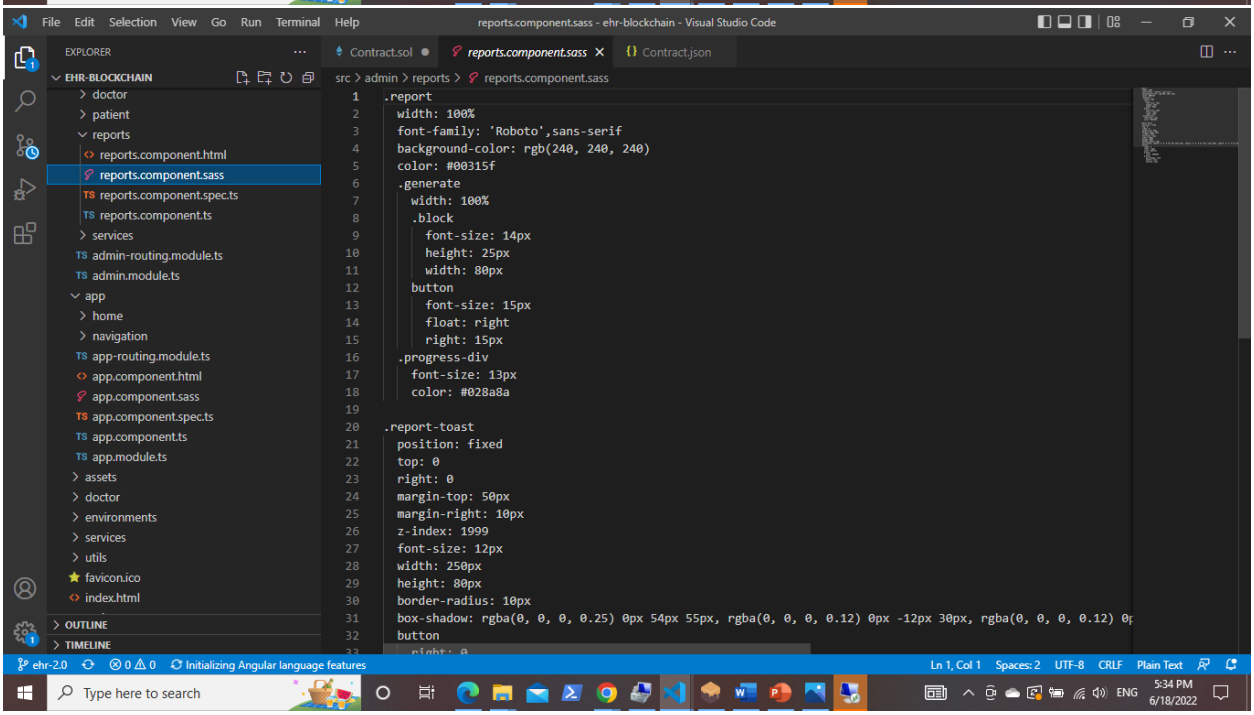




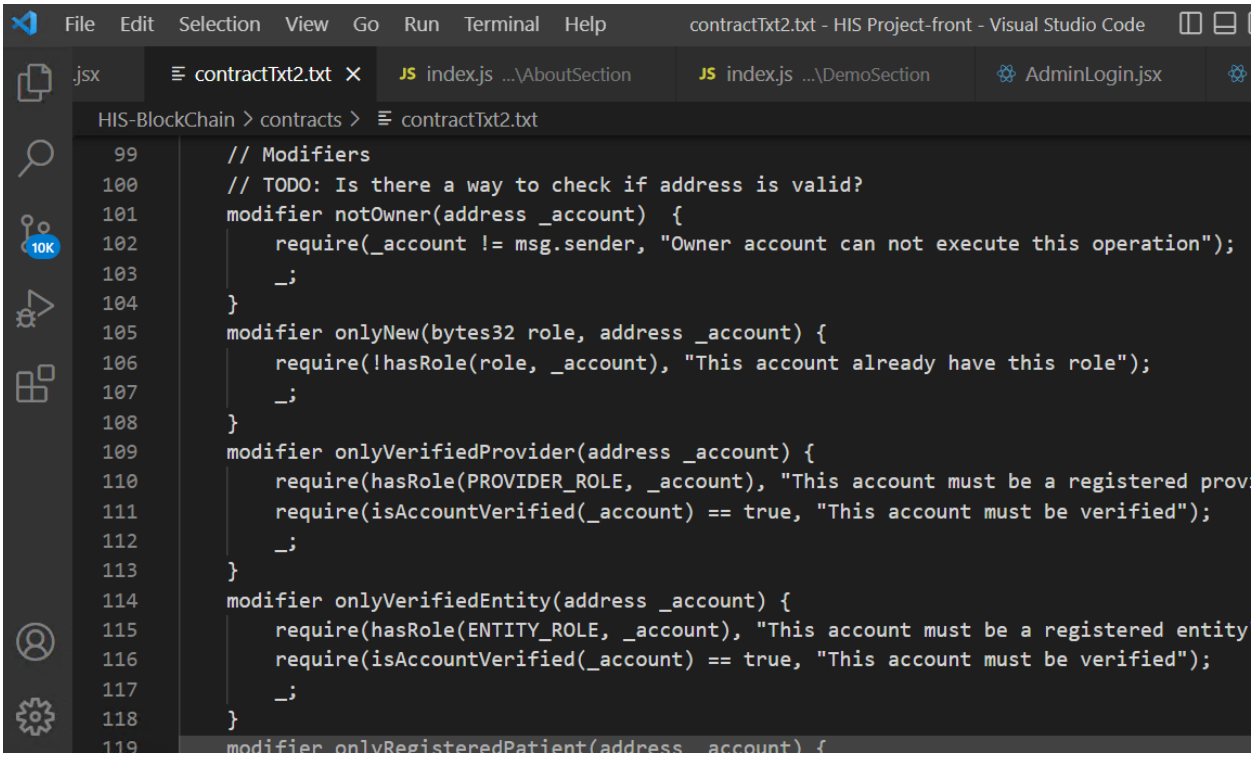

File Edit Selection View Go Run Terminal Help
• contract.sol - predict-disease - Visual Studio Code
EXPLORER
PREDICT-DISEASE
__pycache__
blockchain_data
static
templates
access_blockchain.py
after1.jpeg
after2.jpg
app.py
before1.jpeg
before2.jpeg
compiled_code.json
contract.sol
deploy.py
model.pkl
predict.jpg
README.md
OUTLINE
TIMELINE
contract.sol
function addSolution(uint256 _id,string memory _before,string memory _after,string memory _text) public
//msg.sender.transfer(10**17); //0.1 ether will be provided as incentive
Solutions.push(Solution(msg.sender,_id,_before,_after,_text));
function returnSolutions(uint256 required_id) public view returns(string memory,string memory,string me
string memory _addresses;
string memory _before;
string memory _after;
string memory _text;
for (uint256 i=0; i < Solutions.length; i++) {
if(Solutions[i].disease_id==required_id){
_addresses=string(abi.encodePacked(_addresses," ",toAsciiString(Solutions[i].patient)));
_before=string(abi.encodePacked(_before," ",Solutions[i].before_pic));
_after=string(abi.encodePacked(_after," ",Solutions[i].after_pic));
_text=string(abi.encodePacked(_text," ",Solutions[i].text));
}
}
return (_addresses,_before,_after,_text);
}
function toAsciiString(address x) internal pure returns (string memory) {
bytes memory s = new bytes(40);
for (uint i = 0; i < 20; i++) {
bytes1 b = bytes1(uint8(uint(uint160(x)) / (2**(8*(19 - i)))));
bytes1 hi = bytes1(uint8(b) / 16);
bytes1 lo = bytes1(uint8(b) - 16 * uint8(hi));
s[2*i] = char(hi);
s[2*i+1] = char(lo);
}
return string(s);
Ln 30, Col 102 Spaces: 4 UTF-8 LF Solidity
Type here to search
ENG
5:32 PM
6/18/2022

Report generator

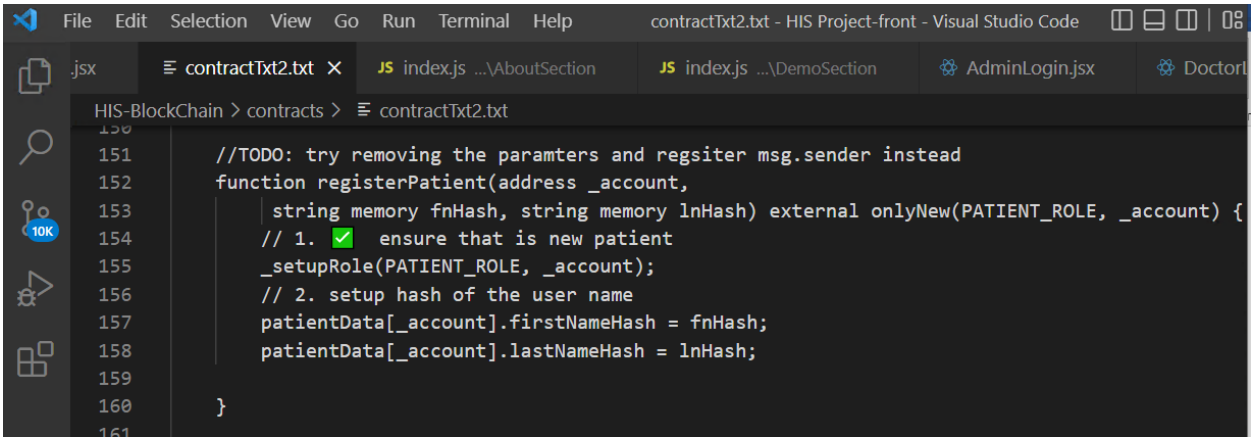




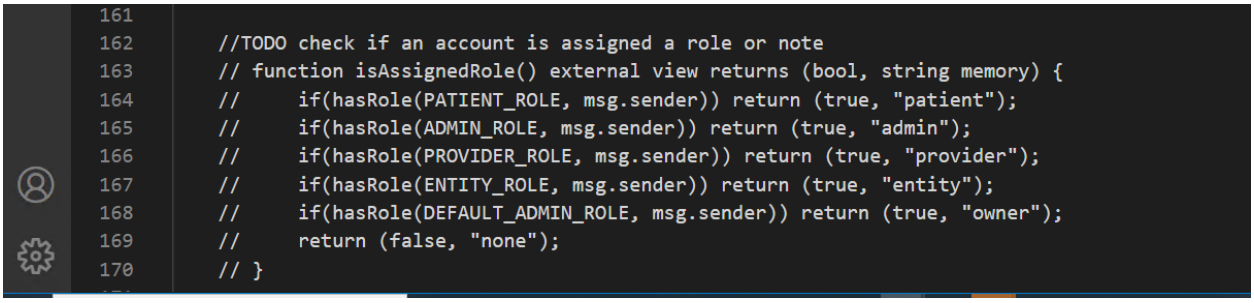
Check account validation



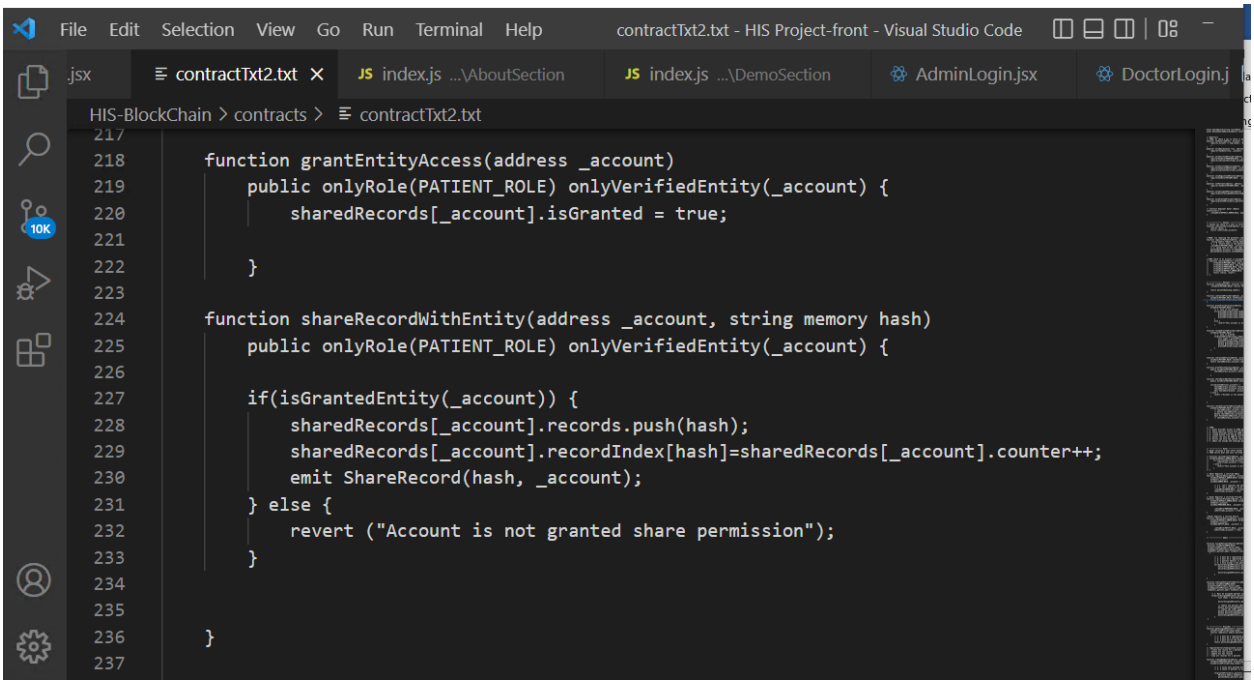
Check if account is assigned



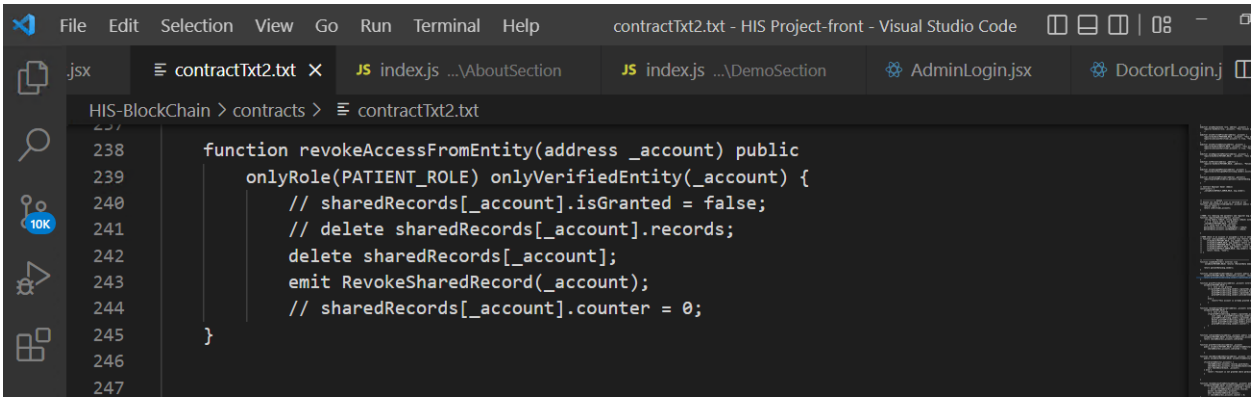
Check if account is assigned a role or not



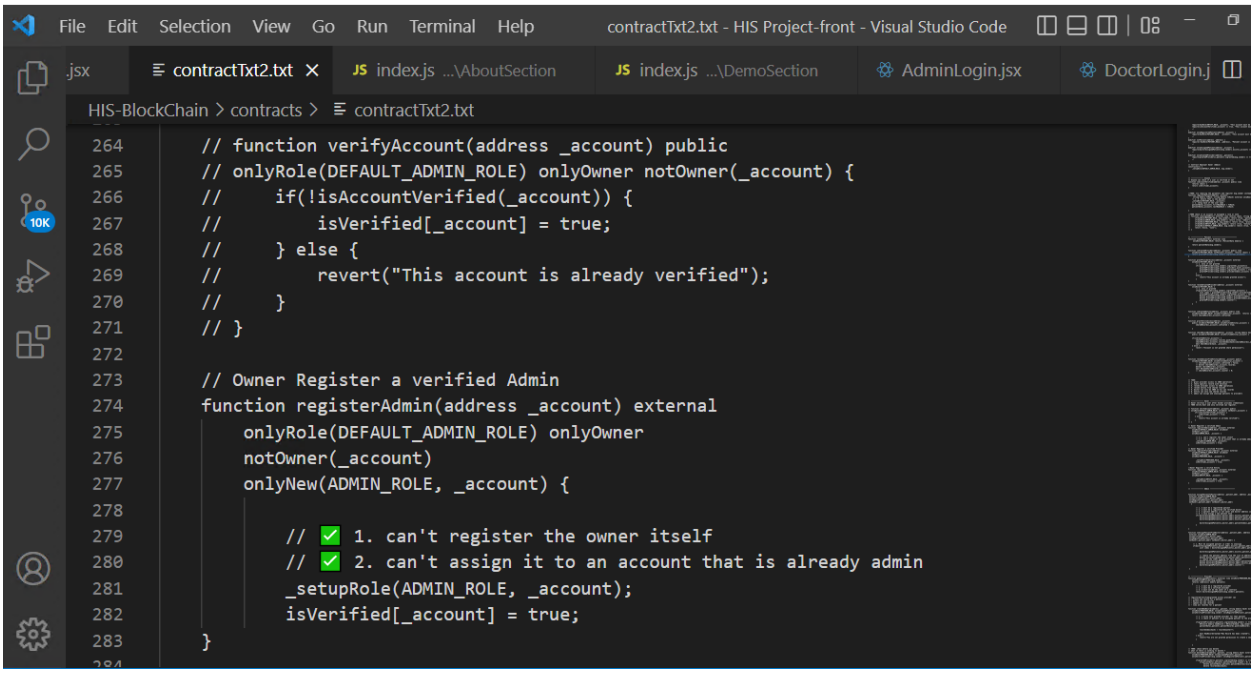
Check if account is granted to share records



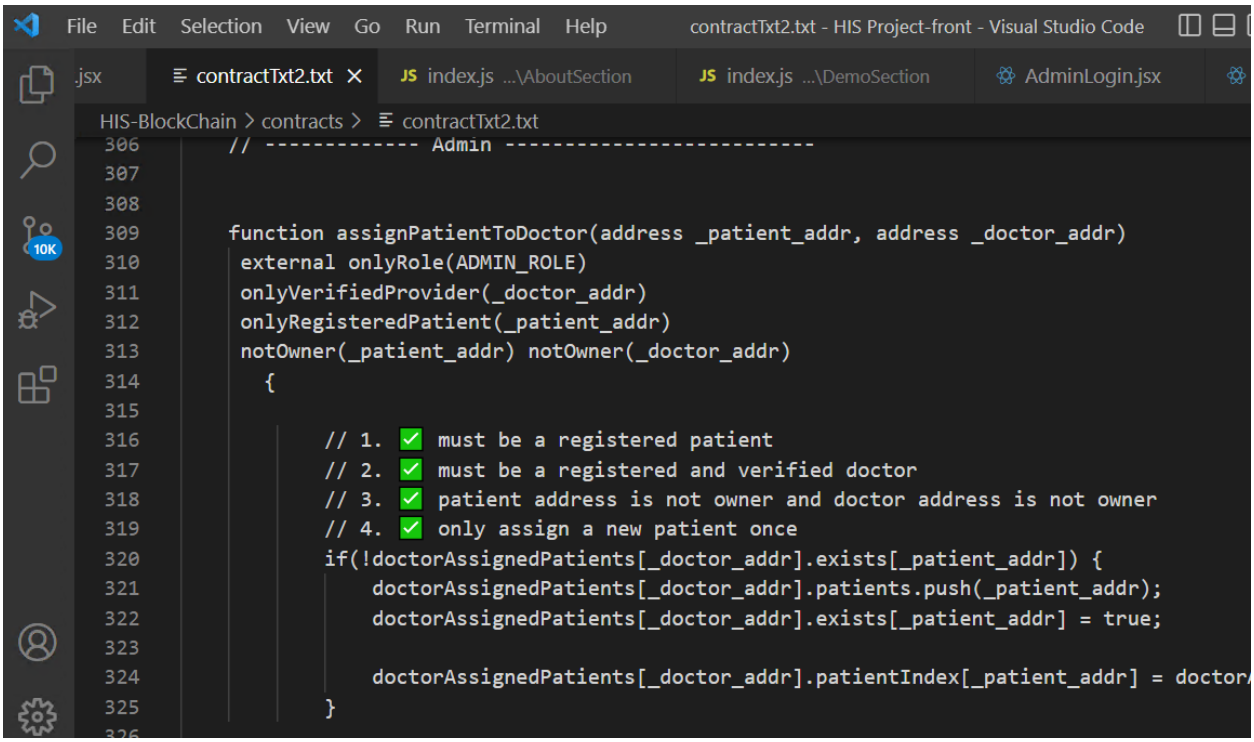
Owner revoke account



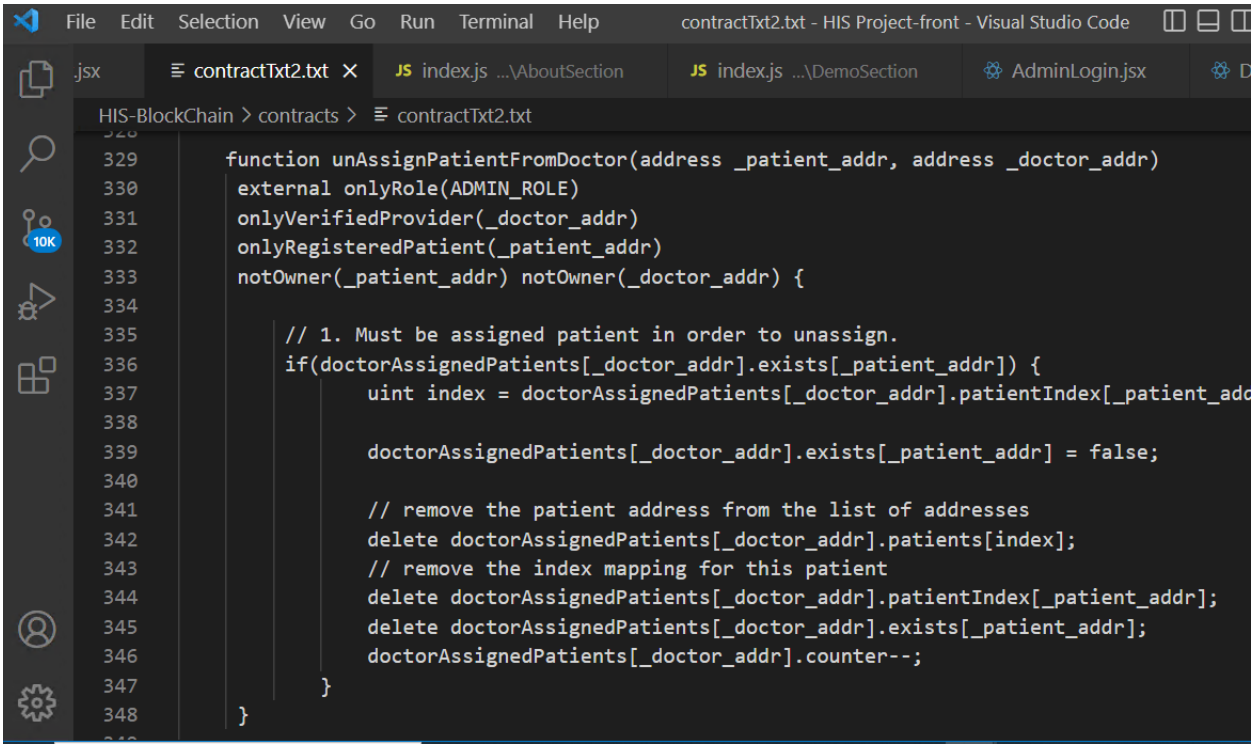
Owner adds Admin



Admin assigns patient to a doctor

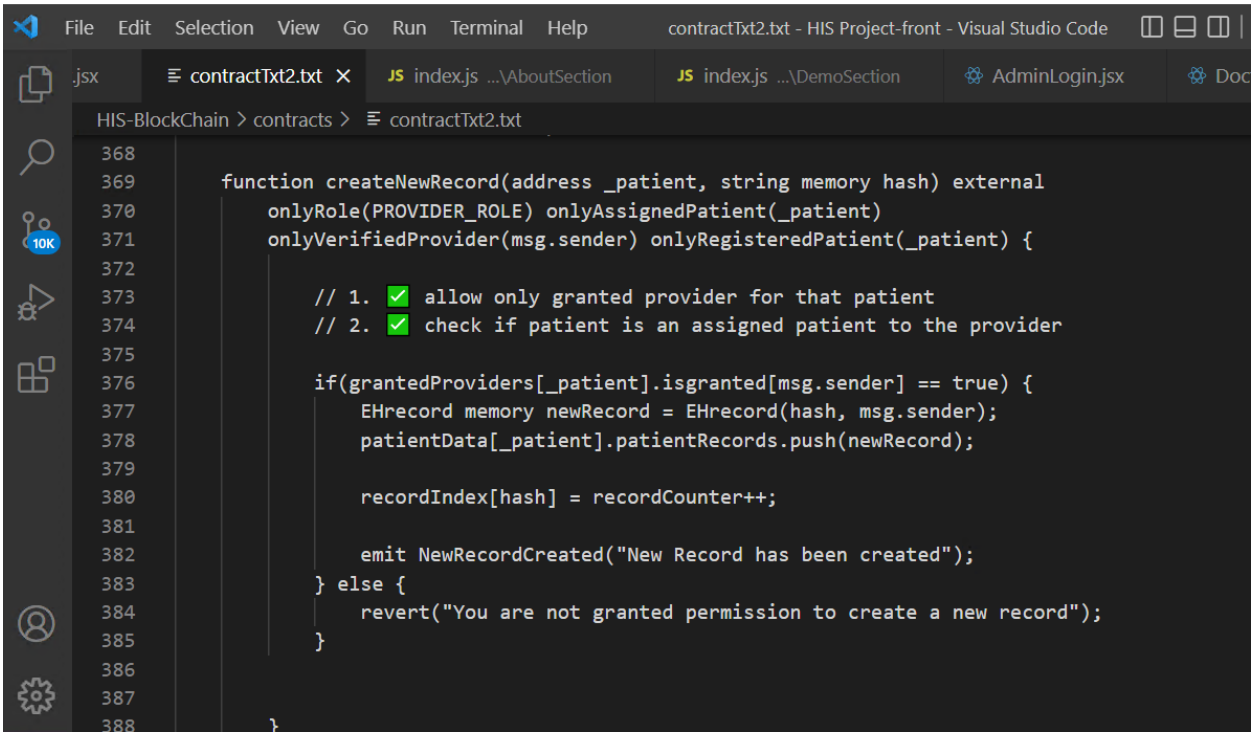


Admin unassigns patients from a doctor



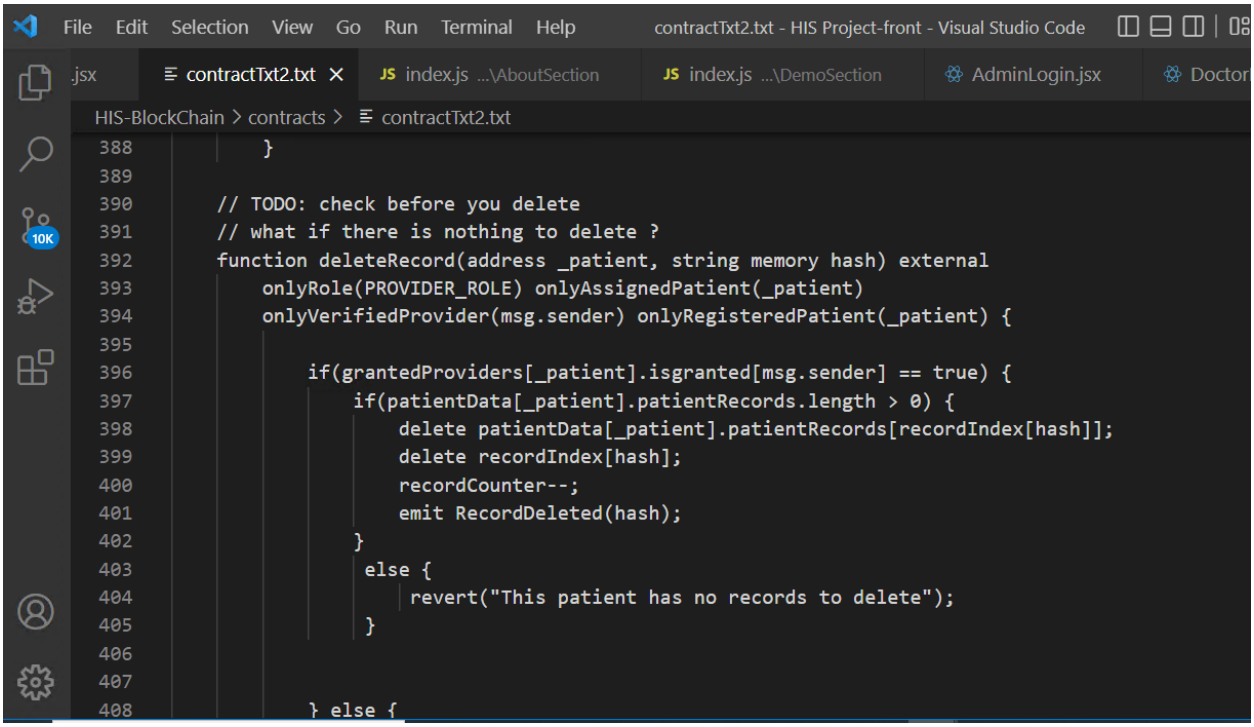

```
function unAssignPatientFromDoctor(address _patient_addr, address _doctor_addr)
 external onlyRole(ADMIN_ROLE)
 onlyVerifiedProvider(_doctor_addr)
 onlyRegisteredPatient(_patient_addr)
 notOwner(_patient_addr) notOwner(_doctor_addr) {

     // 1. Must be assigned patient in order to unassign.
     if(doctorAssignedPatients[_doctor_addr].exists[_patient_addr]) {
            uint index = doctorAssignedPatients[_doctor_addr].patientIndex[_patient_add

            doctorAssignedPatients[_doctor_addr].exists[_patient_addr] = false;

            // remove the patient address from the list of addresses
            delete doctorAssignedPatients[_doctor_addr].patients[index];
            // remove the index mapping for this patient
            delete doctorAssignedPatients[_doctor_addr].patientIndex[_patient_addr];
            delete doctorAssignedPatients[_doctor_addr].exists[_patient_addr];
            doctorAssignedPatients[_doctor_addr].counter--;
        }
 }
```

Doctor creates new record for patient

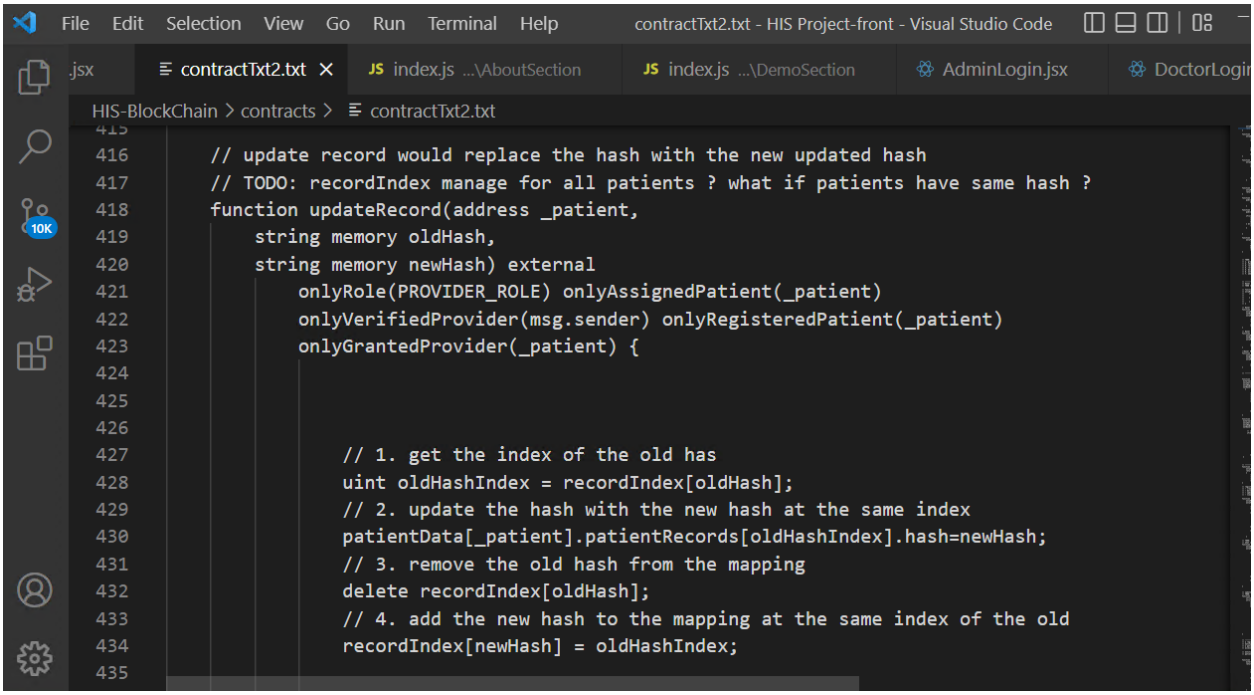



```
function createNewRecord(address _patient, string memory hash) external
    onlyRole(PROVIDER_ROLE) onlyAssignedPatient(_patient)
    onlyVerifiedProvider(msg.sender) onlyRegisteredPatient(_patient) {

        // 1. ✅ allow only granted provider for that patient
        // 2. ✅ check if patient is an assigned patient to the provider

        if(grantedProviders[_patient].isgranted[msg.sender] == true) {
            EHrecord memory newRecord = EHrecord(hash, msg.sender);
            patientData[_patient].patientRecords.push(newRecord);

            recordIndex[hash] = recordCounter++;

            emit NewRecordCreated("New Record has been created");
        } else {
            revert("You are not granted permission to create a new record");
        }

    }
```

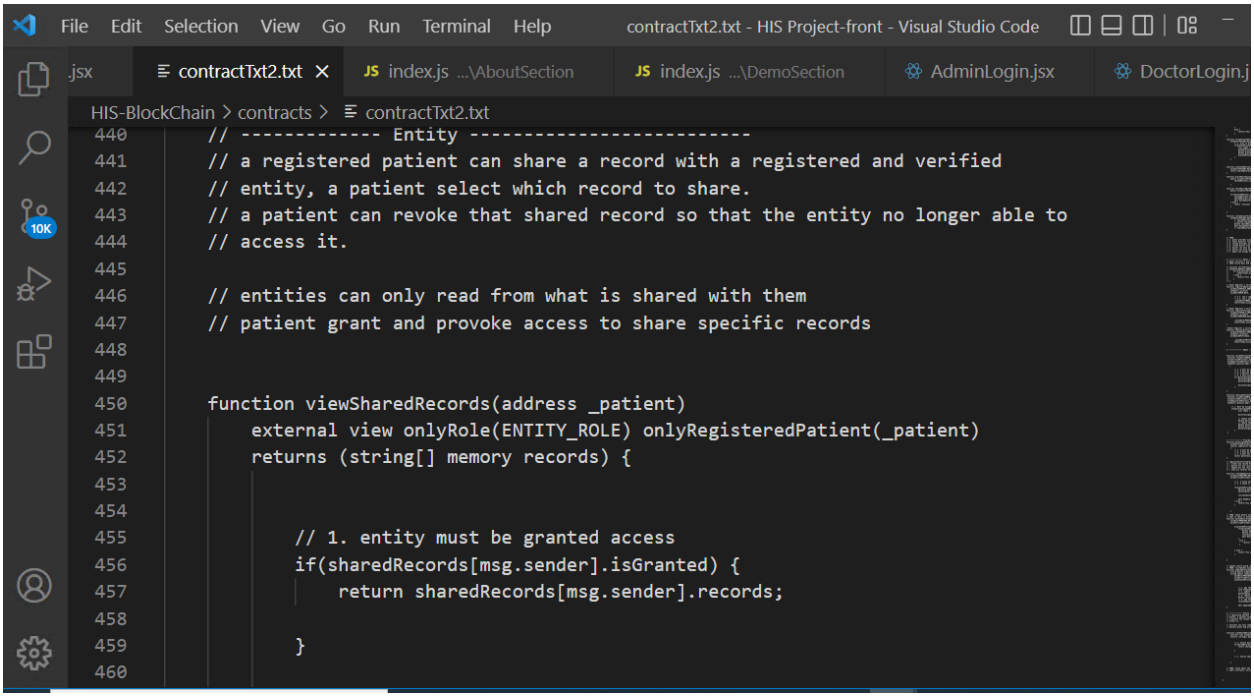
Doctors deletes patient record



Doctor updates patient record

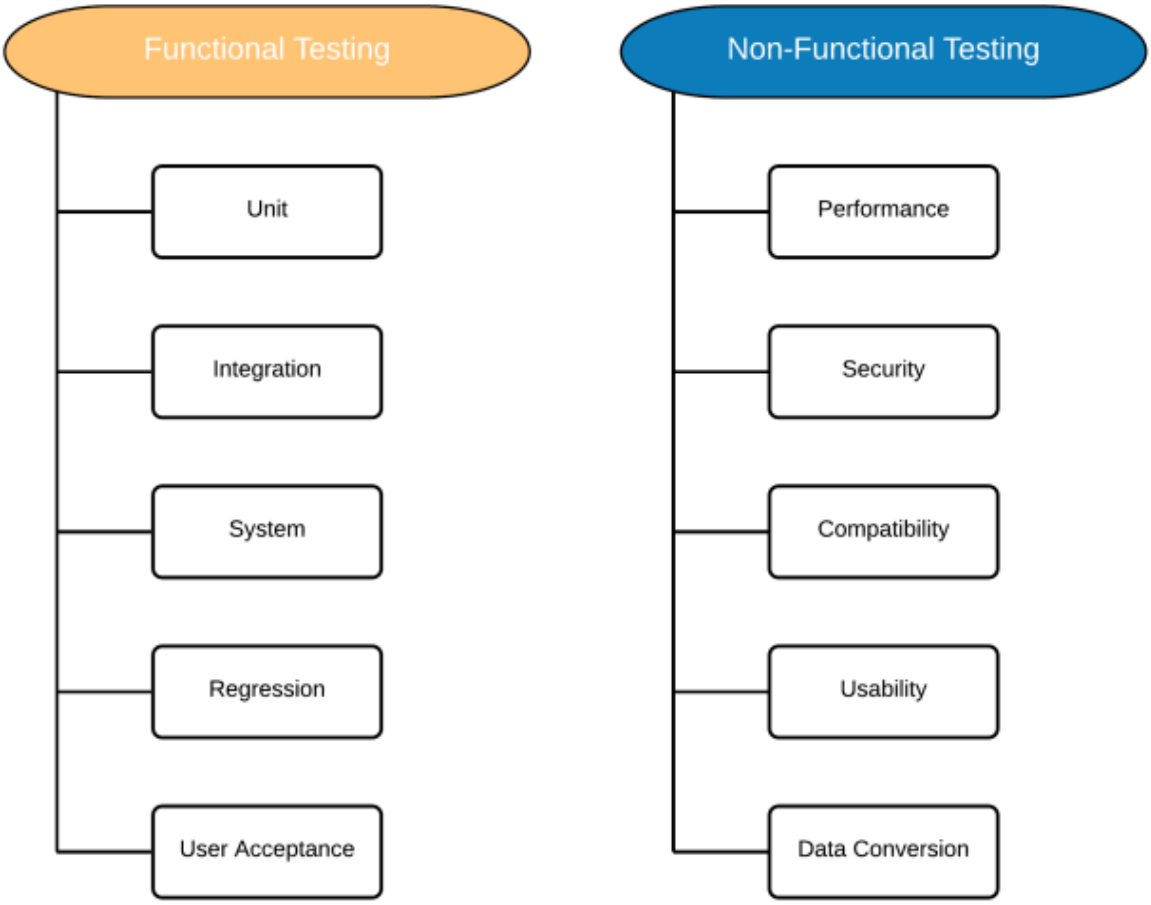


Patient views shared records



Chapter 5: Testing

In this chapter, we are going to discuss and go deeper in ChainCare health system testing and present the types of testing to be used and test cases we examined our application through.



Functional Testing:

Unit Testing

Testing of individual items (e.g., modules, programs, objects, classes, etc.) Usually as part of the coding phase, in isolation from other development items and the system as a whole.

Integration testing

Testing the interfaces between major (e.g., systems level application modules) and minor (e.g., individual programs or components) items within an application which must interact with each other.

Additional Testing

System Testing

Testing a system behavior as a whole when development is finished, and the system can be tested as complete entity.

Regression Testing

To check older functionality after integrating new functionality.

Acceptance testing

Testing to ensure that a development is ready to be deployed into the business, operational or production environment.

Non-Functional Testing

Performance Testing

Accomplished a designated function regarding processing time and through put rate.

Load Testing

Measuring the behavior of within creasing load which can be handled by the component or system.

Stress Testing

Evaluate a system or component at or beyond the limits of its specified requirements.

Security Testing

How well the system protects against unauthorized internal or external access.

Test cases:

Test scenario objective:

Verify patient registration with correct Address and private key.

Assumptions/dependencies:

Valid account: 0x8e0bfE022315efb106C9Fe311B05A6063b8B5f62

Valid private key:
565c767282fdf87b32876fd3c058068cdefc148fdfe7c73001432a4d9fe41373

Step No.	Description	Expected results	Actual results	Error type
1	Navigate to patient registration form	Users navigate to registration form		
2	Enter full name	User can enter full name		
3	Press tap	Move to last name text field		
4	Enter last name	User can enter last name		
5	Press tap	Move to national ID text field		
6	Enter national ID	User can enter national ID		
7	Press tap	Move to age text field		
8	Enter age	User can enter age		
9	Press tap	Move to phone text field		
10	Enter phone	User can enter phone		
11	Press tap	Move to Email text field		
12	Enter Email	User can enter Email		
13	Press tap	Move to city text field		
14	Enter city	User can enter City		
15	Press tap	Move to Address text field		
16	Enter Address	User can enter Address		
17	Press tap	Move to Zip code text field		

18	Enter Zip code	User can enter Zip code		
19	Press tap	Move to social status text field		
20	Enter social status	User can enter social status		
21	Press tap	Move to cinder text field		
22	Enter cinder	User can enter cinder		
23	Press tap	Move to blood type text field		
24	Enter blood type	User can enter blood type		
25	Press tap	Move to Allergy text field		
26	Enter Allergy	User can enter Allergy		
27	Press tap	Move to chronic diseases text field		
28	Enter chronic diseases	User can enter chronic diseases		
29	Press tap	View message of successful registration with Address and private key (0x8e0bfE022315efb106C 9Fe311B05A6063b8B5f62/565c767282fdf87b3287 6fd3c058068cdefc148fdfe 7c73001432a4d9fe41373)	Passed	

Test scenario objective:

Verify doctor's login with correct Address and private key.

Assumptions/dependencies:

Valid account: 0x5Dc31FEd2Abd7c1B1368b5C3Cd75035F3BBf0554

Valid private key:
565c767282fdf87b32876fd3c058068cdefc148fdfe7c73001432a4d9fe41373

Step No.	Description	Expected results	Actual results	Error type
1	Navigate to patient login form	Users navigate to login form		
2	Enter personal Address (0x5Dc31FEd2Abd7c1B1368b5C3Cd75035F3BBf0554)	User can enter the address		
3	Press tap	Move to private key field		
4	Enter private key (565c767282fdf87b32876fd3c058068cdefc148fdfe7c73001432a4d9fe41373)	User can enter last name		
5	Press tap	Doctor access doctor's pages	passed	

Test scenario objective:

Verify doctor's request a permission to access patients medical records after login in successfully.

Assumptions/dependencies:

Valid account: 0x5Dc31FEd2Abd7c1B1368b5C3Cd75035F3BBf0554

Valid private key: 565c767282fdf87b32876fd3c058068cdefc148fdfe7c73001432a4d9fe41373

Step No.	Description	Expected results	Actual results	Error type
1	Navigate to Doctors profile	Users navigate to doctors' profile		
2	Press on request patient access	Users navigate to request form		
3	Enter patients Ethereum	User can enter patients Ethereum		
4	Press tab	Move to Doctors Ethereum text field		
5	Enter Doctors Ethereum	User can enter Doctors Ethereum		
6	Press tap	Request s successfully sent	passed	

Chapter 6: Results and Discussion

In this chapter, we are going to talk about the results of our project whether they have been achieved or not also the differences between the desired result and the actual one.

Results

Expected result

- The patient's medical records are saved securely in one place.
- The patient has easily access to his medical records including medical history, radiology images, diagnoses, treatment plans, allergies, laboratory results.
- The patient gives access to his data to the doctors and removes it whenever he wants.
- Doctors have easy access to the patient's health records used in decision making related to the patient's care
- Doctor can view his patients
- Lap stuff gets the required tests from the patient account and loud it after releases it
- Detect the possible future genetic diseases of the patient he would have.
- Prepare accurate information about the health situation all over the country.
- The patient gives the access to the doctor using Card authentication devices.
- The parents have access and can Monitor the accounts of their children.

Actual Results

- The patient's medical records are saved securely in one place.
- The patient has easily access to his medical records including medical history, radiology images, diagnoses, treatment plans, allergies, laboratory results.
- The patient gives access to his data to the doctors and removes it whenever he wants.
- Doctors have easy access to the patient's health records used in decision making related to the patient's care
- Doctor can view his patients
- Lap stuff gets the required tests from the patient account and loud it after releases it
- Detect the possible future genetic diseases of the patient he would have.
- Prepare accurate information about the health situation all over the country.
- The patient can give an access to another patient to view his records

Discussion

As a conclusion for the previous points, we have managed to meet most of the expectations we planned for except for some points such as building an authentication method using Wi-Fi cards devises, instead we made the authentication method using the wallet address and the private key.

Due to shortage of time, we cancelled the feature that make the parents have access and can monitor the accounts of their children, but we managed this by making any patient can gives an access to another patient to view his records.

Chapter 7: Conclusion

In this chapter, we sum up the whole things and summarize the things mentioned before.

Conclusion

Finally, here we are writing the very last words and putting the last lines of our story and adventure at FCAI-HU, working in this project was really different from any other project we worked on during the last 4 years, this project was full of the feeling of responsibility towards our society and ourselves, that why we tries to do our best in it, regardless the poor resources we had and the limited time.

We consider this project as a thanks to all our professors and teacher assistance and to our country as well, trying to improve the health care industry in Egypt through what we learned and know was a real challenge, a cool challenge that was worth trying.

We hope that ChainCare will play its role in society as we are expecting. We are looking forward to being the first local solution improve the health care industry and make it way easier and effective.

Finally, Information saves lives. This is especially true in the medical field. Health care administrators, doctors, and nurses who access patient and population health data can make crucial decisions about care that could make all the difference in the world to their patients.

Chapter 8: Future Work

In this chapter, we will propose what are the future plans we have concerning our application as it will not end by the end of the discussion.

ChainCare in Future Work

Here is our plan for the coming versions of ChainCare:

- Develop a machine learning model to suggest the optimal diseases that the patient have from his medical tests.
- Make the authentication easier using Wi-Fi cards devises
- Globalize the app and make the patient account valid all over the world like his bank account.