

# Blockchain based patient health record management system

Jaysheel Shah (2020A7PS0083P)  
Shreyas Ketkar (2020A7PS0075P)

Aryan Desai (2020A7PS0123P)  
Samyak Jain (2020A7PS0089P)

## INTRODUCTION

Currently, we use a centralized system for storing and sharing sensitive details of the Patient among doctors, hospitals, laboratories, physicians, and pharmacies. Such systems are prone to threats, data misuse, and data loss, and thus, there is a requirement for maintaining a decentralized system for storing health-related records. Such a decentralized system uses the Blockchain network in the e-healthcare system. Such systems help in seamless data sharing among different healthcare organizations upholding patient privacy through the anonymization of sensitive details.

### **Benefits of using Decentralized System:**

#### **1) Security:**

The decentralized system ensures the integrity and confidentiality of The patient data as the data is shared over every node in the blockchain network and a simple change can be viewed by every node thus ensuring security.

#### **2) User-Centric Interfaces**

The decentralized system are designed for intuitive interaction with simple navigation and easy retrieval of data.

#### **3) Simplified Patient Accessibility**

The patient convenient and efficient access to its own data. The accessibility of patient data is independent of their healthcare Provider.

#### **4) Efficient Data Exchange:**

The implementation of a blockchain network enables smooth exchange of medical data among different healthcare entities.

#### **5) Reduced Administrative Burden:**

The implementation of a blockchain network replaces the traditional, expensive and tedious paperwork technique of storing healthcare data. This reduces the administrative burden on both healthcare providers and patients.

In this project, we have designed a decentralized web application that is secure for patient in such a way that the doctor can see the patients record only when the patient will grant access to that doctor otherwise the doctor will not get access of the patient data.

### **METHODOLOGY**

The various Technologies that are used in our project:

Blockchain Network:

- 1) Polygon Mumbai Testnet: Test network of ethereum which simulates the real Ethernet network.

Cryptocurrency Wallet:

- 1) Metamask: To enable the payment of gas fees when storing data in the Testnet

IDE:

- 1) Remix IDE: Used for initial development and testing of Smart Contract.
- 2) VS Code: Used to code the rest of the development web application from frontend to backend.

Ethereum API:

- 1) Infura: To connect the blockchain web application to the web browser seamlessly and easily.

Programming Language:

- 1) Solidity: To develop the smart contracts.
- 2) Next.js: To code the entire frontend and backend of the web application.

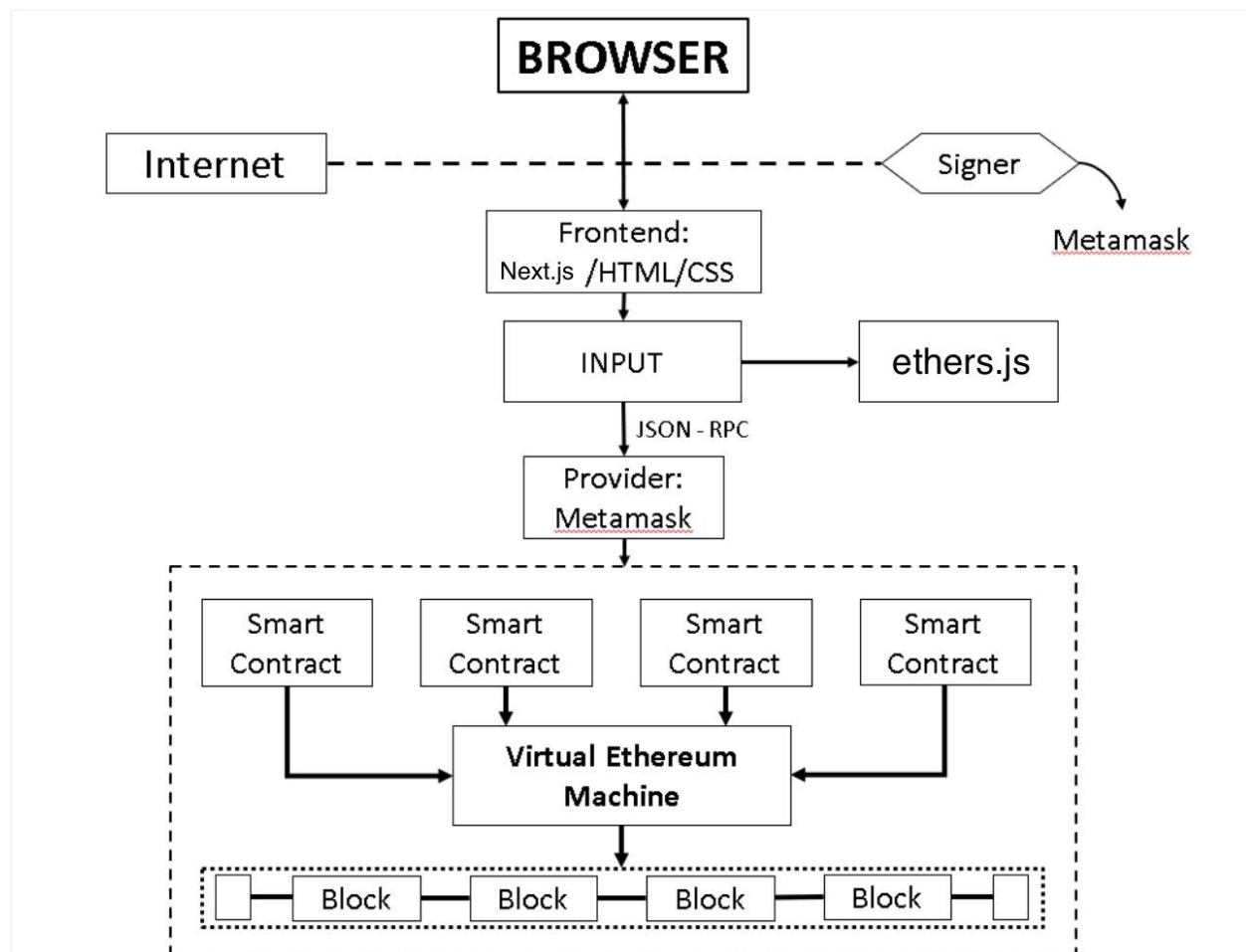
## FrontEnd Frameworks:

- 1) React: To create interaction between the front end and the smart contract of the application. It is used to create the html web pages with styling.

## Other Technologies:

- 1) Web3: To enable interaction between Metamask with the Smart Contract that has been deployed to that Polygon Mumbai Testnet.
- 2) Node.js: To npm install packages that are crucial to the development of the blockchain application.

## PROPOSED SYSTEM



## Implementation of Smart Contract

We have created a Solidity smart contract, named "HealthDetails," which serves as a decentralized health records management system. The contract facilitates the storage and retrieval of health-related information for both patients and doctors on the Ethereum blockchain. Here are the key components and functionalities of the smart contract:

### 1. Patient and Doctor Structs:

- The contract defines two main structures, Patient and Doctor, encapsulating relevant details such as personal identification, contact information, and timestamps of creation.

### 2. Owner and Permissions:

- The contract has an owner, initialized during deployment, who has special privileges.
- Patients and doctors are identified by their Ethereum addresses, and permission to access patient records is explicitly managed through the `isAllowed` mapping.

### 3. Record Creation and Editing:

- Patients can register and update their health records using the `setPatient` and `editPatient` functions.
- Similarly, doctors can register and modify their profiles through `setDoctor` and `editDoctor`.

### 4. Access Control:

- The contract implements a permission system where the owner (presumably a healthcare provider or the patient) can grant or revoke access to specific doctors. This is done through the `grant_Access` and `revoke_Access` functions.

### 5. Lists and Counts:

- Lists of patient and doctor addresses are maintained in `p_List` and `d_List` respectively.
- The contract keeps track of the number of patients (`p_Cnt`), doctors (`d_Cnt`), and permissions granted (`perm_Granted_Cnt`).

### 6. Information Retrieval:

- The contract provides functions (`findPatient` and `findDoctor`) to retrieve health information based on the Ethereum address.

Access control is enforced, ensuring that only authorized individuals can access the data.

7. Date of Record Creation:

- Additional functions (findPatientRecordDate and findDoctorRecordDate) allow the retrieval of the creation timestamp for patient and doctor records.

8. Summary Counts:

- The contract includes functions (no\_of\_Patients, no\_of\_Doctors, and no\_of\_Perm\_Granted) to retrieve counts of patients, doctors, and granted permissions, respectively.

## Functionality that is Implemented:

1) Add Patient:

This adds patient records in the block. A deduction of fees in terms of gas fee is made from the corresponding patient account

Patient ID	<input type="text"/>
Patient Name	<input type="text"/>
Patient Address	<input type="text"/>
Patient Sex	<input type="text"/>
Patient Phone	<input type="text"/>
Patient BirthDate	<input type="text"/>
Patient Height	<input type="text"/>
Patient Weight	<input type="text"/>
Patient BloodGroup	<input type="text"/>

2) Add Doctor

This adds doctor records in the block. A deduction of fees in terms of gas fees is made from the corresponding doctor account.

Doctor ID	<input type="text" value="Doc ID"/>	<b>Save Doctor Record</b>
Doctor Name	<input type="text" value="Doc Name"/>	
Doctor Sex	<input type="text" value="Doc Sex"/>	
Doctor Phone	<input type="text" value="Doc Phone"/>	
Doctor BirthDate	<input type="text" value="Doc BirthDate"/>	
Doctor Qualification	<input type="text" value="Doc Qualification"/>	

### 3) Find Patient

This functionality takes the patient account public address to retrieve Patient Details. This functionality is used by the patient and the doctor( who has been granted access) to find the respective patient details.

Other patients do not have access to the details.

Patient Address	<input type="text" value="Patient Address"/>	<b>Retrieve Patient Details</b>
-----------------	----------------------------------------------	---------------------------------

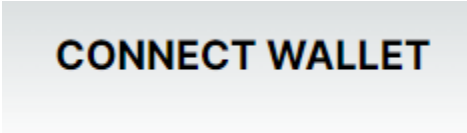
### 4) Grant Access to Doctor:

This functionality is used by the patient to grant doctor access to its information.It takes the doctor account public address as input to grant access to the respective doctor.

Doctor Address	<input type="text" value="Doctor Address"/>	<b>Grant Access</b>
----------------	---------------------------------------------	---------------------

#### 5) Connect Wallet:

This connects the respective account with Metamask.

A rectangular button with a light gray background and a thin black border. The text "CONNECT WALLET" is centered in a bold, black, sans-serif font.

### **CONSENSUS MECHANISM**

We have used Polygon testnet which uses Ethereum consensus mechanism which is based on Proof of Stake (PoS). They have validators who create new blocks by putting up some of their own cryptocurrency as a kind of collateral. Ethereum handle more transactions without using up as much energy. The selection of validators to generate new blocks follows a deterministic approach, typically influenced by factors such as the quantity of cryptocurrency staked and the duration for which it has been staked.

### **INDIVIDUAL CONTRIBUTION**

Jaysheel Shah and Shreyas Ketkar: Built the smart contract and deployed on the ethereum smart contract.

Samyak Jain and Aryan Desai: Built Frontend and connect it with the blockchain Network.