**Decentralized Patient Record Storage**

**A Usecase of Application of Blockchain**

**In The Field Of Medical Science**

**Problem Statement:**

The patient record is one of the important assets that is currently centralized, maintained, and managed by hospitals.

Some countries have moved to Electronic Health Record (EHR), and many countries still follow the old method to store the patient and medical details. Around 80 to 90% of the hospitals in the United States have adopted basic EHR systems, which have a lot of problems related to privacy, security, and ownership of these health records.

Even after moving to EHR, the patient record is completely owned and controlled by hospitals, and being the owner of that record, a patient does not have access to the data and has to always reach out to hospitals to get it.

Data consolidation is one of the biggest challenges as every hospital stores these EHRs on their server. The patients need to access different EHRs to consolidate their complete medical history.

Since hospitals own the EHRs, they have full control and can manipulate the data at any moment. They can even sell the data to research organization and earn a good amount of money.

Along with the patient record, doctor authenticity is another challenge that is faced as there is no direct way to check if the doctor is real or not.

**Solution**:

Blockchain is a technology designed to manage patient data that has the potential to support transparency and accountability. A blockchain is a ledger of transactions where an identical copy is visible to all the members of a computer network. Network members validate the data entered into the ledger, and once entered, the data is immutable.

With the help of Blockchain technology, a solution can be developed to store the EHR of any patient in a distributed and decentralized network. With blockchain solutions, patients can own their data and can approve hospitals, doctors, and research organizations to access their data.

**Roles**:

In this solution, we are going to focus on two roles:

* Patient
* Doctor

A patient can do the following functions:

* Register to the portal
* Add a disease
* View their records
* Update their data

A doctor can do the following things:

* Register to the portal
* Access and add the records of their patients
* Prescribe medicine to a patient

**Features of the application:**

1. Register a new doctor

This function is used to register a new doctor to the ledger. It takes the below parameters:

/\*\*

@param \_name Name of the doctor

@param \_qualification Which degree he/she holds as a doctor

@param \_workPlace Address of his/her hospital/clinic \*/

1. Register a new patient

This function is used to register a new patient to the ledger. It takes the below parameters:

/\*\*

@param \_name Name of the user

@param \_age Age of user

\*/

1. Add a patient's disease

This function is used to add a patient's disease. It takes the below parameters:

/\*\*

@param \_disease Name of the disease

\*/

1. Add medicine

This function is used to add medicines. It takes the below parameters:

/\*\*

@param \_id Id of the medicine

@param \_name name of the medicine

@param \_expiryDate ExpiryDate of the medicine

@param \_dose Dose prescribed to the patient

@param \_price Price of the medicine

\*/

1. Prescribe medicine

This function is used by doctors to prescribe medicine to a patient. It takes the below parameters:

/\*\*

@param \_id Medicine Id

@param \_patient address of the patient

\*/

1. Update patient details by patient

This function helps patients to update their age. It takes the below parameters:

/\*\*

@param \_age New age of the patient

\*/

1. View patient data

This function helps to view patient data stored in Blockchain. It takes the below parameters:

/\*\*

@return id Id of the patient

@return age Age of the patient

@return name Name of the patient

@return disease The disease of the patient

\*/

1. View medicine details

This function helps to fetch medicine details. This function below input parameters and return the details about the medicine.

/\*\*

@param id Id of the medicine

@return name Name of the medicine

@return expiryDate Expiry date of the medicine

@return dose Dose prescribed for the medicine

@return price Price of the medicine

\*/

1. View patient data by a doctor

This function helps a doctor to view patient data. It takes the below parameters:

/\*\*

@param \_id ID of the patient

@return id ID of the patient

@return age Age of the patient

@return name Name of the patient

@return disease The diseases of the patient

\*/

1. View prescribed medicine to the patient

This function helps the doctor to view patient data. It takes the below parameters:

/\*\*

@dev View prescribed medicines to the patient

@param \_patient address of the patient

@return ids list of medicine id's

\*/

1. View doctor details

This function helps to view doctor details. It takes the below parameters:

/\*\*

@param \_id ID of the doctor

@return id ID of the doctor

@return name Name of the doctor

@return qualification Name of degree he/she holds as a doctor

@return add Address of his/her hospital/clinic

\*/

**Recommended technologies:**

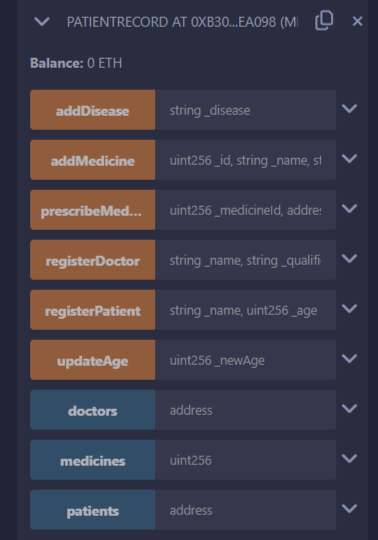
1. Smart Contract development: Solidity
2. IDE Tool: Remix
3. Blockchain: Ethereum
4. Server: Ganache Blockchain

**Project development guidelines:**

* The project will be delivered within four sprints with every sprint delivering a minimal viable product.
* It is mandatory to do proper sprint planning with user stories to develop all the components of the project.
* The learner can use any technology from the above-mentioned technologies for different layers of the project.
* The learner must maintain the version of the application over GitHub and every new change should be sent to the repository.
* The learner should also deploy and host the application on any blockchain instance.

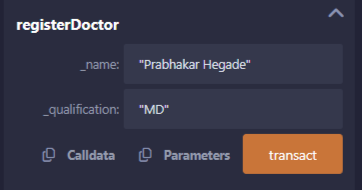
**Project outputs:**

Once you deploy the contract using remix IDE, you can see all the below functions:

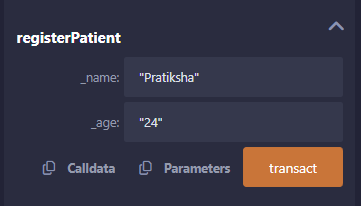


**Function Details:**

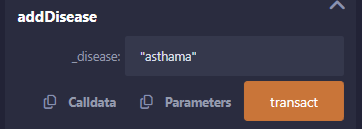
Register a new doctor using the below function:



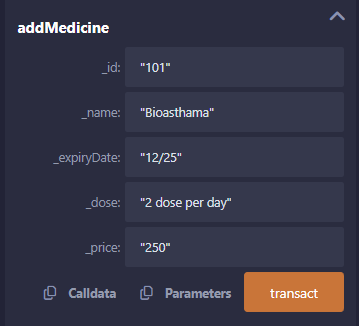
Add a new patient:



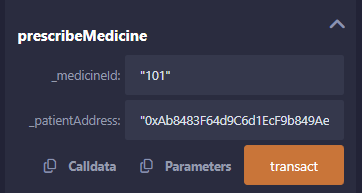
Add a disease for the patient:



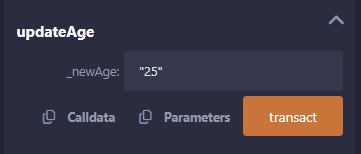
Add medicine to the ledger:



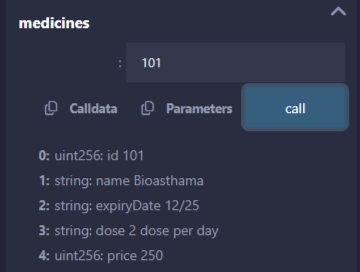
Prescribe medicine to the patient:



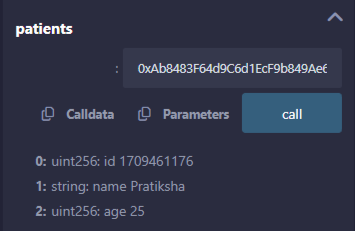
Update the age of the patient:



View medicine details:



View patient data:



View doctor details:

