

DOCMEDSYNC

Enrollment Number(s)	20103062	20103333	20103078
Name of Student(s)	Archit Gupta	Rohit Saini	Sahil Jaiman
Name of Supervisor	Dr. Kapil Madan		



December- 2022

Submitted in partial fulfilment of the Degree of

**Bachelor of Technology
In
Computer Science Engineering**

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING &
INFORMATION TECHNOLOGY**

JAYPEE INSTITUTE OF INFORMATION TECHNOLOGY, NOIDA

TABLE OF CONTENT

1. Introduction	7-12
1.1 General Introduction	7
1.2 Problem Statement	9
1.3 Novelty of the problem	10
1.4 Brief description of the problem	10
1.5 Comparison of Existing approaches to the problem	11
1.6 Flowchart Depicting the course of Project	12
2. Literature Survey	13-16
2.1 Summary of paper Studied	13
2.2 Integrated summary of literature Studied	16
3. Requirement Analysis and Solution Approach	17-30
3.1 Overall Description of the project	17
3.2 Requirement Analysis	18
3.3 Solution Approach	19
4. Modelling and Implementation Details	31-36
4.1 Design Diagram	30
4.1.1 Use case Diagram	30
4.1.2 Sequence Diagram	31

4.1.3 Activity Diagram	31
4.2 Implementation Details	32
5. Conclusion and Future Work	39
6. References	40

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

PLACE: JIIT, NOIDA

SIGNATURE

DATE: December 5' 2022

NAME: Archit Gupta

Rohit Saini

Sahil Jaiman

ENROLLMENT NO:

20103062

20103333

20103078

CERTIFICATE

This is to certify that the work titled “**DOCMEDSYNC**” submitted by “**Archit Gupta, Rohit Saini, Sahil Jaiman**” in partial fulfilment for the award of degree of B. Tech of Jaypee Institute of Information Technology, Noida has been carried out under my supervision. This work has not been submitted partially or wholly to any other University or Institute for the award of this or any other degree or diploma.

Signature of Supervisor

Name of Supervisor

Designation

Date

ACKNOWLEDGEMENT

This minor project was a great chance for learning about Electronic Health Record system using Blockchain. It helped us in learning a lot about the smart contracts, Dapps and Interplanetary File System (IPFS).

We take up this opportunity to express our sincere gratitude and special thanks to our supervisors Dr. Kapil Madan for their insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project and also for their constant encouragement and advice throughout the project. We would also like to appreciate the invaluable help and support of the teaching and non-teaching staff of the Department of Computer Science throughout the entire course of this project. We are also grateful to all our teammates and classmates for their help, encouragement and invaluable suggestions.

We will definitely strive to use the gained skills and knowledge to the fullest extent as possible, and we will try to work on their expertise, in order to attain desired career objectives.

Signature of Supervisor

Name of Supervisor

Designation

Date

SUMMARY

The objective of this project is to implement an electronic health record system on the Ethereum Blockchain. Also, to investigate the possibility of using Ethereum Blockchain for EHR systems, and to identify some potential challenges and solutions that may arise from such an integration.

We introduce the Ethereum Blockchain as a potential solution to the current problems faced by electronic health record systems.

The system will be made up of three components: a ReactJS front end, a Solidity smart contract, and an IPFS back end.

The first component of the project is the ReactJS front end. This component will be responsible for displaying all the data on the user interface and will allow users to input data into the system. The second component is the Solidity smart contract which will contain all of our business logic and rules for how we want our database to function. It will also store all of our data on IPFS. Finally, we have our IPFS back end which stores all of our data and allows us to query it using API calls from both ReactJS as well as other applications that need access to it in the future such as a mobile application or website that needs information from our database without having direct access to it via API calls themselves.

SIGNATURE OF STUDENT(S)

NAME(S):

ARCHIT GUPTA

ROHIT SAINI

SAHIL JAIMAN

SIGNATURE OF SUPERVISOR

NAME:

DR. KAPIL MADAN

DATE: December 5' 2022

DATE: December 5' 2022

1. INTRODUCTION

1.1 General Introduction

I data sharing using blockchain introducing a novel framework for managing and sharing EMR data for cancer patient care. The authors report that EHRs are electronic, therefore easier to potentially share between healthcare entities such as pharmacies, insurance companies, patients' families, and other healthcare providers. One example use case for EMR blockchain included primary patient care, which would solve the problem of patients not having their records when moving from hospital to hospital. Another blockchain use case envisioned data aggregation for research since patients are often unwilling to participate in data sharing across organizations due to the current lack of appropriate data sharing mechanisms and distributed coordination efforts such as signing and sending consent forms to different entities. The authors suggested that blockchain may ultimately assist in connecting different healthcare systems for improved patient care. The authors proposed a novel framework on managing and sharing I data for cancer patient care. They implemented the framework as a prototype in collaboration with Stony Brook University reporting on their experiences. The prototype was developed to provide a chain for sharing data between oncology patients and their doctors, specifically patient history and physical exams, laboratory results, and delivered radiation doses. The chain was a solution developed to improve assurance that shared patient data are fast and convenient while complete, securely stored, and accessible only according to the patient's consent.

EHRs in a blockchain for building a blockchain taxonomy; examining known challenges; important principles, protocols, standards, and open questions; analysing architectures; and discussions on long-term blockchain needs for the "ever-growing" storage of patient medical records. Their synthesis identified and summarized high-level relevance of blockchain categories and underlying-terms within security, scalability, governance, interoperability, and privacy while noting which papers discussed the topics. Our research differs from research of Mayer et al and other similar reviews, in that we explore blockchain literature with different inclusion criteria and previously unpublished open practitioner questions regarding blockchain.

Blockchain is a peer-to-peer (P2P)-distributed ledger technology, developed in 2008, that requires validation from originators and organizers before being accepted. It is a newer technology that has potential to significantly improve the data exchange in the healthcare

sector, especially during the sudden development of the COVID-19 pandemic, which has exposed some limitations in healthcare systems to handle public health emergencies. By using a unique immutable architecture, blockchain can support characteristics of decentralization, exchange, anonymity, and accessibility. Examples can be found in finance, radiology, supply chain management, and government, among other venues. A major strength of blockchain is its related and supportive cryptographic and distributed architecture for sharing meta-information or specific components of EHRs, while maintaining a high degree of transfer accountability and providing data exchange interoperability among healthcare entities.

EHRs are being widely adopted by health systems and healthcare providers. Since EHRs contain identifiable and private patient information, data transfers between health systems and healthcare providers require a secure platform for exchange¹ hence, data sharing across disparate I systems remains a challenge. Blockchain could be a solution to enable secure data sharing. The overall objective of this scoping review was to determine if blockchain can improve interoperability and secure repetitive processes for efficient sharing and viewing of EHRs.

1.2 Problem Statement

The healthcare industry is facing many challenges when it comes to managing patient data. One of the biggest problems is that a lot of patient data is still paper-based, which is inefficient and leads to duplication of records. With the widespread adoption of electronic health records (EHRs). They have made it possible for healthcare providers to have quick and easy access to patient health information. This has improved the quality of care and led to better patient outcomes. It is important to consider their potential impact on the cost-effectiveness of health care. EHRs can lead to information overload. Errors in EHRs can lead to increased costs and decreased quality of care. Another challenge is that different healthcare organizations use different electronic medical record (EMR) systems, which makes it difficult to share data between them. Unfortunately, most I systems are not interoperable, the information should be exchangeable and must be usable for further purposes. Another big problem with EHRs is information asymmetry. Finally, EHRs are also vulnerable to data infringements. Hospitals have become a target of cyber-attacks and an increasing trend has been witnessed by the researchers while conducting this study that a lot of research work has been done in this domain. An electronic health record is defined as an electronic version of a medical history of the patient as kept by the health care provider. But it consists of some major security and privacy flaws.

Potential Cybersecurity Issues

The data of the patients lies on a centralized database, which are prone to Denial of Service (DoS) attacks and single point of failure

Privacy of Patients

If the database ever gets hacked. The data of the Patients can get leaked into the world which is unethical. Centralized systems are vulnerable to privacy attacks as well.

Inaccurate Data

If an EMR is not updated immediately, as soon as new information is known, such as after test results come in, anyone viewing that EMR could receive incorrect. This could lead to errors in diagnosis and treatment.

Time and Money

It also takes time to demo I products and negotiate with I system vendors to choose and implement the right system for your practice.

1.3 Significance/Novelty of the problem

The significance of our problem is that it can help the patients/users to not to hustle and panic in a situation of emergency about the important aspects such as medical records, test reports and past sickness details. If we are counting on the statistics available through extensive research, we get to know that there are not many projects available in the healthcare industry and those that are available and not successfully available to the users. As of now, India does not have any platform which can provide the same security and accessibility, and ease of access to its users and doctors. The other software available in the market is not inter communicable and data security is not ensured. Moreover, this project solves the problem of the displaced/lost records and tests which might be crucial for the treatment of the patient.

Users do not have the exact information about his/her health records. Our platform will ensure that the user account will be authenticated and well connected with the government-approved proof Id's such as Aadhar Card. Moreover, existing I systems operate on cloud services and DOCMEDSYNC is based on blockchain.

1.4 Brief Description of Solution Approach

Docmedsync provides a decentralized easy to use Electronic Medical Record system (EMR). It is a free to use web application providing a feature rich as well as interactive UI making it easy to use.

Minimal Security Risks

As previously mentioned, we use Ethereum Network for our computation making it very safe and secure. There cannot be a single point of failure.

Complete Privacy

The application used IPFS technology for storage of patient's data. Every patient can control who can access their data. Only registered Organizations and verified Medical Institute can access your data.

Verifies Admins

For a Medical Institute to participate in this shared system, it needs to be verified by one of the admins. Same goes for the Organizations. They require proper medical and identity license to be successfully registered.

Non-Profit

Docmedsync is a free to use, non-profit system. One does not need to buy this software. It is available for everyone. There are some public open features which can be accessed by anyone, be it admin, owner or someone visiting the website for the first time. All that is required is a crypto wallet.



Fig 1 : Solution Approach to build Dapp

1.5 Comparison of existing approaches to the problem

If we are counting on the statistics available through extensive research we get to know that there are not many projects available in the healthcare industry and those that are available and not successfully available to the users. The traditional systems at the hospitals around the world by time being do not provide the facility of storing the details of the user in such easy and secure ways providing the same security and accessibility, and ease of access to its users and doctors. Not many people around the world has given a thought about such approach in Healthcare Industry where we can build a whole infrastructure on the blockchain which will be beneficial for the Human Cause

The other software's which are available in the market is not inter-communicable and data security of the user is breached. Users do not have the exact information about his/her health records. Our platform will ensure that the user account will be authenticated and well connected with the government-approved proof Id's such as Aadhar Card. Moreover, existing I systems operate on cloud services and DOCMEDSYNC is based on blockchain which makes it even more transparent and secured.

1.6 Flowchart depicting the course of the project

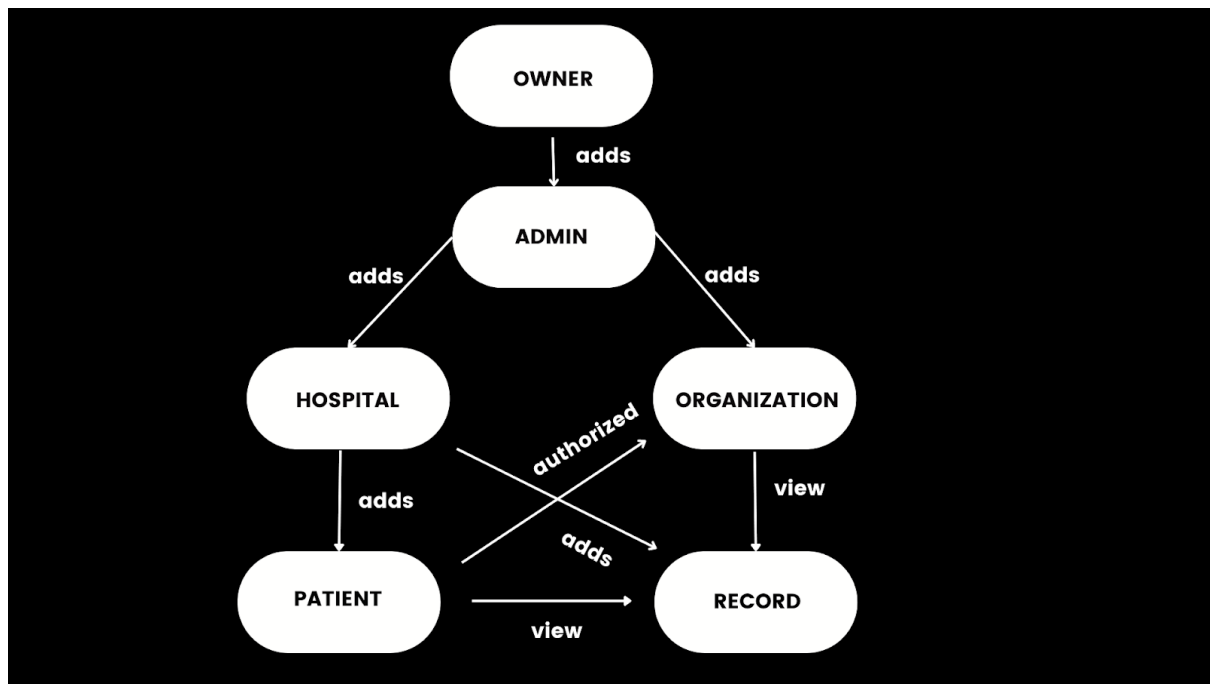


Fig 2: Flow chart of DOCMEDSYNC

Ethereum is a decentralized system that is temper-proof, secure and confidential blockchain based system for electronic health records. The proposed framework consists of Public, Hospital, Authorized, Owner, Admin, Patients. Hospitals and Presonal Clinics. The one who deployed the Smart Contract in the. Ethereum blockchain is considered as the Owner. Owner has the rights to add new Admins, remove an Admin and Transfer its ownership. To keep the Data Integrity and transparency of the data of the patients, Owner has not been given the right to view or delete any information about the Patients or Hospitals. Registered Hospitals and Registered Hospitals have the right to read, write and update the data of the patient they are working on with. Public section is a free space where anyone accessing the site can view the list of hospitals and organizations which are connected with the platform and check for the owner and the different admins of contract.

2. LITERATURE SURVEY

2.1 Summary of Paper Studied

[1] IEEE: Using Blockchain for Electronic Health Records

By -: Ayesha Shahnaz, Usman Qamar, Ayesha Khalid

By going through this research paper, we get to know that The Electronic Health Record (I) systems face problems regarding data security, integrity and management. This paper, discuss how the blockchain technology can be used to transform the I systems. It presents a framework that could be used for the implementation of blockchain technology in healthcare sector for I.

Blockchain is a chain of blocks that are connected together and are continuously growing by storing transactions on the blocks. This technology was introduced by Nakamoto for his popular work of digital currency or crypto-currency, i.e., bitcoin. A number of researchers have identified that using this technology in healthcare would be a feasible solution. A new transaction being sent by a user on the blockchain network suggests that a new block is created. A block in the blockchain is used for keeping transactions in them and these blocks are distributed to all of the connected nodes in the network. This whole process of the block being added on the blockchain is done by the nodes reaching upon a con-senses where they decide which blocks are valid to be added. A block contains three things in it which are data, hash of current block and hash of previous block. The hash that is stored in these blocks contains a SHA-256 cryptographic algorithm which is used for unique identification of a block on the chain. As in case of Bitcoin, the data consists of coins that are actually electronic. Each block that is added on the chain would need to follow some consensus rules for it to be added. For this purpose, the most common consensus algorithm is Proof of Work (PoW) algorithm and it was used by Satoshi Nakamoto, in bitcoin network. The basic working of this is that there are number of nodes or participants on the network. This process is called mining and the nodes that are performing these calculations are miners. The data that was before concentrated at one central point is now handled by many trusted entities across the entire network. The control of information to be distributed and handled by consensus is now reached by nodes on the network. Achieving data transparency in any technology is to have a trust-based relationship between entities. The data or record at stake should be secured and temper proof. Any data being stored on the blockchain is not concentrated at one place but is

instead distributed across the network.

This makes it to be transparent and secure from any third-party intervention. Blockchain technology uses cryptographic functions to provide security to the nodes connected on its network. The SHA-256 algorithm generates strong one-way checksum for digital data that cannot be used for data extraction.

System design is the most important and vital part of any framework as it is used for the development of the system. This section includes the modules, architecture and various elements that are combined together to form the whole system's framework. These entities or modules have further concepts that need to be explained as follows. Users of this system could be patients, doctors and administrative staff. The main task of these users would be to interact with the system and perform basic tasks such as create, update and delete the medical records. The system was implemented by using the Ethereum and its dependencies. This section explores system implementation in more detail. The GUI contains all the functions that could be accessed by a particular user. The user according to the assigned role could use this GUI for interacting with the other layer of the system, the blockchain layer.

These contracts are used for giving access to the users on the Dapp and performing CRUD operations on the records of a patient. The Roles smart contract is a pre-defined smart contract that defines all the operations that are being performed in it and various conditions associated with them.

The proposed framework is a combination of secure record storage and granular access rules for those records. This also solves the problem of information asymmetry of I system. For the future, the paper suggested to implement the payment module of the existing framework of information asymmetry of I system principles of the healthcare sector.

[2] IJARCCCE: Blockchain Technology in Electronic Health Record System

BY:- Malavika M.B , Richa Kumari , Nihara S.M

This paper suggested that Electronic Health Records (EHRs) are used to maintain the history of the Patient`s records. Patients face a critical need to focus on the details of their own healthcare and restore the rapid development of block chain technology promotes a secure healthcare system, including medical records as well as patient-related data. This technology provides patients with an extensive, validity of EHRs encapsulated in block chain, it presents an attribute-based signature scheme with multiple authorities.

In the existing system the patient may lose control of the existing healthcare data, while the service provider usually unable to easily share these data with researchers or providers. Interoperability challenges between different providers, add extra barriers to high-performance data sharing.

Standardization of problem lists in the healthcare industry is needed to enable more efficient exchange of information between health providers and especially to patients. Some forms of problem list preparation, such as auto-population of lists, represent significant compliance and patient.

Block chain is considered as a new technological revolution that was introduced. The benefits of the block chain technology are decentralized maintenance, data saving in the block-then-chain structure, secure transporting and accessing of data as well as antitamper and undeniable data security. Block chain enables the management of authentication, confidentiality, accountability and data sharing while handing.

Advantage of Proposed System:

Providing accurate, up-to-date, and complete information about patients at the point of care.
Enabling quick access to patient records for more coordinated, efficient care.

2.2 Integrated Summary of the Literature Studied

In this paper, we proposed situations of blockchain innovation utility in numerous social insurance settings: critical attention, restorative data inquire about, and associated wellness. We talked about how keeping up a permanent and easy document, which video display units every one of the occasions took place over the device, may want to improve and inspire the administration of restorative records. In view of the compels diagnosed with the social insurance placing, we defended the decision of the permissioned block chain innovation for the use of the proposed situations. We likewise exhibited a layout of the gadget for the unique wishes if there must be an occurrence of radiation oncology data sharing and accomplished a version that guarantees safety, security, accessibility, and best-grained get entry to electricity over exceedingly sensitive patients' information. As a characteristic of destiny paintings, we might need to develop the structure of an affected person record and its metadata, utilizing the semantics of social insurance facts, consisting of the chance of sharing radiology pictures, which is substantially more tough. Since we work in a joint effort with a health facility, we intend to test our framework with the facts of the real sufferers. Our long-haul objective is to investigate exclusive situations proposed in the paper, for example, associated wellbeing and therapeutic data check out and apply them via and by to upgrade the prevailing medicinal services records the board Dataset Analysis and Solution Approach

3. REQUIREMENT ANALYSIS AND SOLUTION APPROACH

3.1 Overall description of the project

Our project includes the work done in the domain of health care being implemented using blockchain technology. Moreover, Ethereum is used for the overall implementation of the proposed framework. Our framework is designed to create such a decentralized system that is temper-proof, secure and confidential blockchain based system for electronic health records.

The proposed framework consists of Public, Hospital, Authorized, Owner, Admin, Patients. They were given granular access as they should have varying level of authority on the system.

The one who deployed the Smart Contract in the Ethereum blockchain is considered as the Owner. Owner has the rights to add new Admins, remove an Admin and transfer its ownership. Owner has its own private key through which it verifies its ownership. To keep the Data Integrity and transparency of the data of the patients, Owner has not been given the rights to view, update or delete any information about the Patients or Hospitals. The major work of the Owner is to add multiple Admin.

Once an Admin is added, it has an access to add new hospitals which must include the details of the hospitals and the unique Ethereum address of the hospital for transactions. Furthermore, Admin has the access to add different organisations which are in the field of the health industry such as NGO's, Presonal Clinics. Similar to the hospitals, the bare minimum requirement includes the details of the organization and the unique Ethereum address of their organization

The Registered Hospitals have the authority to read, write and update the data of the patient they are working on with. To add a new record of a patient, the personal details of the patient are required and his/her unique Ethereum address to register him on the platform. Patient also need to provide the past health records and the medical history to add it to their account. As of Organisations, they only have the right to read and view the details of the patient.

Once a new patient is added, he/she has the access to get their details as well as their records. Patient has the authority on who is allowed to view their data by granting the requests of accessing their records and also to remove the granted requested as per their will.

Public section is a free space where anyone accessing the site can view the list of the hospitals and organizations which are connected with the platform and check for the owner and the different admins of contract.

3.2 Requirement Analysis

Functional Requirements:

Tokenization is one of the most important things you have to understand when talking about Blockchain. Tokens allows you to create digital representations for goods, services or rights. These mechanisms are highly relevant for all digitization of real-world goods use cases, because this allows trust and values to be exchanged between different parties without the need for a central intermediary.

Data security and privacy are very important requirements, especially for business use. The issue is that it must be possible to grant specific data access rights to individual users or roles. That's very important because in a public chain like Bitcoin, all data within a block is readable by everyone. The compliance with data privacy regulation. Especially in Europe we do have a very tough data privacy regulation (General Data Protection Regulation). So, we are faced with issues like for instance that there is a right to be forgotten within the GDPR or the requirement to have a data processing agreement with the data provider, which could be a big challenge in a decentralized system.

Decentralized data storage is a very core requirement of a distributed system. It must be possible to manage transactions and data storage in an efficient way. Sure, if decentralized data storage is your only requirement, there would be better solutions than Blockchain that solve that problem but obviously, the capability to manage data in a decentralized, distributed way is a fundamental requirement.

Smart Contracts are the most interesting part of Blockchain systems. While a standard contract defines only the terms of a business relationship, a Smart Contract is able to define also the conditions under which the contract is executed. A Smart Contract is a set of fixed rules, defined as a software program, executed on the Blockchain. Examples you can define with a Smart Contract are: The definition of criteria under which a payment is made, the definition of quality criteria and the rules if these are not observed. Because such coded and

tamper-proof executed rules can eliminate a central authority, Smart Contracts play an important role in automating cross-company collaboration.

Non-Functional Requirements:

UX Design: One unique obstacle to the creation of blockchain is the complexity of the user interface. The query is, how much of blockchain should you disclose to your consumers

Scalability: If the platform is connected to an implementation of a blockchain such as Ethereum then it is also connected to the scalability of the underlying network.

Development Operations: As blockchain is still emerging, the technology to build a blockchain-based system is still evolving. It takes time to integrate new functionality into the system and improve the DevOps experience. Therefore, there should be enough amount of time associated with the development phase

3.3 Solution Approach

Smart contracts are an important part of Dapps as they are used for performing basic operations. These contracts are used for giving access to the users on the Dapp and performing CRUD operations on the records of patient. The smart contract is made purely for implementing the functionality of the proposed framework.

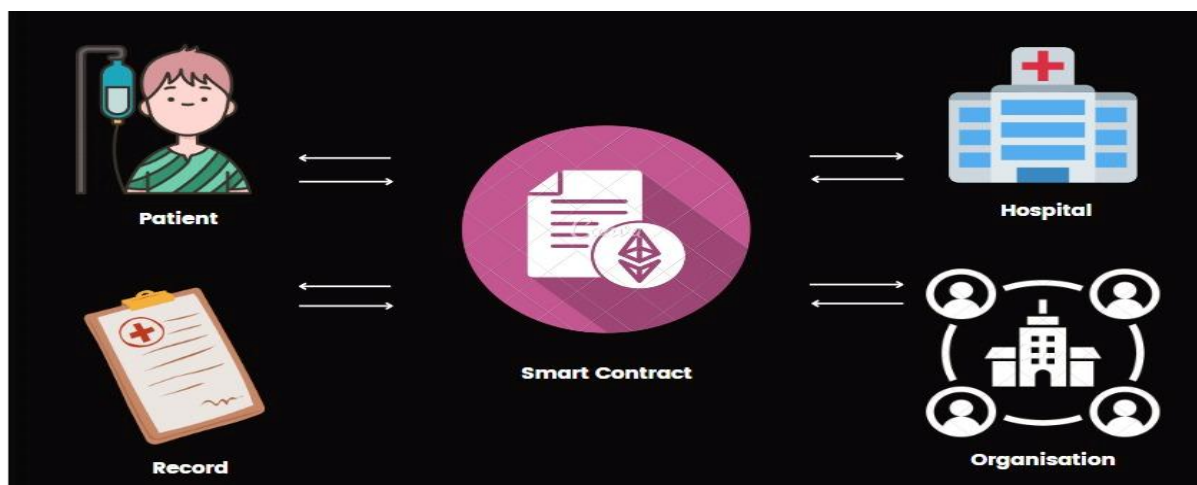


Fig 3: Smart Contract of DOCMEDSYNC

The Smart contract consists of the following structures -:

(i)Hospital

```
struct Hospital {  
    uint256 id;  
    string name;  
    string physicalAddress;  
    address walletAddress;  
    string License;  
}
```

(ii)Record

```
struct Record {  
    uint256 id;  
    uint256 hospitalId;  
    uint256 patientId;  
    string condition;  
    string description;  
    string allergies;  
    string Document;  
}
```

(iii)Patient

```
struct Patient {  
    uint256 id;  
    string name;  
    string gender;  
    string bloodGroup;  
    string DOB;  
    uint256 phoneNumber;  
    uint256[] records;  
    string physicalAddress;  
    string profilePicture;  
    address walletAddress;  
}
```

(iv)Organization

```
struct Organization {  
    uint256 id;
```

```

    string name;
    address walletAddress;
    string License;
}

```

Function created in Smart Contract are as follows:-

(i) -function to get the address of the owner

-no restriction in the access

-returns the address of the current owner

```

function getOwner() public view returns (address) {
    return owner;
}

```

(ii) – function to check if the given address is an admin or not

- no restriction in the access

- returns true or false.

```

Function isAdmin(address _admin) public view returns (bool) {
    return admin[_admin];
}

```

(iii) – function to add new hospitals.

- Callable by only admins.

- receive name of hospital, address of hospital, wallet of hospital, and IPFS Hash of document

- emits an event for adding new hospital

```

function addHospital(
    string memory name,
    string memory physicalAddress,
    address walletAddress,
    string memory license
) public onlyAdmin {
    require(
        hospitalToId[walletAddress] == 0,
        "Already registered as an hospital"
    );
}

```

```

        hospitals[hospitalId] = Hospital(
            hospitalId,
            name,
            physicalAddress,
            walletAddress,
            license
        );
        hospitalToId[walletAddress] = hospitalId;
        hospitalId++;
    }

```

(iv) – function to get hospital with the given id

- check if any hospital exists with the given Id
- If the hospital exists then we return it

```

function getHospitalById(uint256 _id)
    public
    view
    returns (Hospital memory)
{
    require(hospitals[_id].id != 0, "No such hospital exists");
    Hospital memory hosp = hospitals[_id];
    return hosp;
}

```

(v) – function to get hospital having the specified address

- firstly checks if the given address exists in the mapping from hospitalAddress to Id
- now we check if any hospital exists with the returned Id
- If the hospital exists then we return it

```

function getHospitalByAddress(address _address)
    public
    view
    returns (Hospital memory)
{
    require(hospitalToId[_address] != 0, "No such hospital exists");
    uint256 _id = hospitalToId[_address];
    require(hospitals[_id].id != 0, "No such hospital exists");
    Hospital memory hosp = hospitals[_id];
    return hosp;
}

```

(vi) – function to add new records of an existing patient.

- if the patient already exists then we make a new Record variable
- map the created record to its recordId
- add the recordId to the list of records of the patient
- increment the recordId

```
function addNewRecord(
    uint256 _hospitalId,
    uint256 _patientId,
    string memory _condition,
    string memory _description,
    string memory _allergies,
    string memory _document
) public onlyHospital {
    require(patients[_patientId].id != 0, "The patient does not exist");
    Record memory rec = Record(
        recordId,
        _hospitalId,
        _patientId,
        _condition,
        _description,
        _allergies,
        _document
    );
    records[recordId] = rec;
    patients[_patientId].records.push(recordId);
    recordId++;
}
```

(vii) – function to add new patients

- we check if the given patient id already exists
- if does not exist then we make a new Patient variable
- add the constructed variable to patients mapping
- add the patientId to the mapping of patient address to patient Id

```
function addNewPatient(
    uint256 _id,
    string memory _name,
    string memory _gender,
    string memory _bloodGroup,
```



```

    string memory _dob,
    uint256 _phoneNumber,
    string memory _physicalAddress,
    string memory _profilePicture,
    address _walletAddress
) public onlyHospital {
    require(
        patients[_id].id == 0 && patientToId[_walletAddress] == 0,
        "Patient already exists"
    );
    uint256[] memory _records;
    patientToId[_walletAddress] = _id;
    patients[_id] = Patient(
        _id,
        _name,
        _gender,
        _bloodGroup,
        _dob,
        _phoneNumber,
        _records,
        _physicalAddress,
        _profilePicture,
        _walletAddress
    );
}

```

(viii) – function to add new organizations.

- Callable by only admins.

- receive name of organisation, _wallet Address of organization, and IPFS Hash of document

```

function addOrganization(
    string memory _name,
    address _walletAddress,
    string memory _license
) public onlyAdmin {
    require(
        organizationToId[_walletAddress] == 0,
        "Organization already exists"
    );
    organizations[organizationId] = Organization(
        organizationId,
        _name,
        _walletAddress,
        _license
    );
}

```

```

    );
    organizationToId[_walletAddress] = organizationId;

    organizationId++;
}

```

(ix) – function to get organization with the given id

- check if any organization exists with the given Id
- If the organization exists then we return it

```

function getOrganizationById(uint256 _id)
    public
    view
    returns (Organization memory)
{
    require(organizations[_id].id != 0, "No such organization exists");
    return organizations[_id];
}

```

(x) – function to get organization having the specified address

- firstly checks if the given address exists in the mapping from organization Address to Id
- now we check if any organization exists with the returned Id
- If the organization exists then we return it

```

function getOrganizationByAddress(address _addr)
    public
    view
    returns (Organization memory)
{
    uint256 _id = organizationToId[_addr];
    require(
        _id != 0 && organizations[_id].id != 0,
        "No such organization exists"
    );
    return organizations[_id];
}

```

(xi) - function to give access authority to addresses

- message sender must be the patient

- if all condition pass add id of patient and address to authorised mapping

```
function addAuthByAddress(address addr) public onlyPatient {
    uint256 _id = patientToId[msg.sender];
    authorised[_id][addr] = true;
}
```

(xiii) – function to give access authority to an organization

- message sender must be the patient
- firstly, check if organization with the given id exist or not
- if all conditions pass add id of patient and address of the organization to the mapping

```
function addAuthById(uint256 _id) public onlyPatient {
    Organization memory org = organizations[_id];
    require(org.id != 0, "No such organization found");
    uint256 patientId = patientToId[msg.sender];
    authorised[patientId][org.walletAddress] = true;
}
```

(xiv) – function to remove access rights from some address

- message sender must be the patient
- check if the given address has the access authorities
- if all conditions check, remove the mapping of id and address

```
function revokeAuthByAddress(address addr) public onlyPatient {
    uint256 _id = patientToId[msg.sender];
    require(authorised[_id][addr] == true, "Already not authorised");
    authorised[_id][addr] = false;
}
```

(xv) – function to remove access authority from an organization

- message sender must be the patient
- check if the given organization exists or not

- check if the given organization has the access authorities
- if all conditions check, remove the mapping of id and address

```
function revokeAuthById(uint256 _id) public onlyPatient {
    Organization memory org = organizations[_id];
    require(org.id != 0, "No such organization found");
    uint256 patientId = patientToId[msg.sender];

    require(
        authorised[patientId][org.walletAddress] == true,
        "Already not authorised"
    );
    authorised[patientId][org.walletAddress] = false;
}
```

(xvi) – function to get the details of the patient

- this function is callable by either a hospital or an authorised address
- we check if the given id is a valid patient or not
- if all checks pass, we return all the data of the patient except list of its records

```
function getPatientDetails(uint256 _id)
    public
    view
    onlyAuthorised(_id)
    returns (
        uint256,
        string memory,
        string memory,
        string memory,
        string memory,
        uint256,
        string memory,
        string memory,
        address
    )
{
    require(_id != 0 && patients[_id].id != 0, "No such patient found");
    Patient memory pat = patients[_id];
    return (
        pat.id,
        pat.name,
        pat.gender,
        pat.bloodGroup,
        pat.DOB,
```

```

        pat.phoneNumber,
        pat.physicalAddress,
        pat.profilePicture,
        pat.walletAddress
    );
}

```

(xvii) – function to get all the records of the given patient

- this function is callable by only either a registered hospital, organization or patient
- it checks if the patient with the given id exists or not
- it checks if the number of records of the given patient is greater than 0 or not
- we create an array for each of the attributes
- we return all the attributes except patientId and records own id
- loop through all the records in the recordId array of the patient and fill the array as specified
- return all the filled arrays

```

function getPatientRecords(uint256 _id)
    public
    view
    onlyAuthorised(_id)
    returns (
        uint256[] memory,
        string[] memory,
        string[] memory,
        string[] memory,
        string[] memory
    )
{
    Patient memory patient = patients[_id];
    require(_id != 0 && patient.id != 0, "No such patient found");
    require(patient.records.length > 0, "No record found");

    uint256 len = patient.records.length;
    uint256[] memory rec = patient.records;

    uint256[] memory hosId = new uint256[](len);
    string[] memory cond = new string[](len);
    string[] memory desc = new string[](len);
}

```

```

string[] memory aller = new string[](len);
string[] memory docs = new string[](len);

for (uint256 i = 0; i < rec.length; i++) {
    Record memory curr = records[rec[i]];
    hosId[i] = curr.hospitalId;
    cond[i] = curr.condition;
    desc[i] = curr.description;
    aller[i] = curr.allergies;
    docs[i] = curr.Document;
}

```

Apart from creating contract for storing files **IPFS** is used. **IPFS** is a distributed system for storing and accessing files, websites, applications, and data. Built for speed and simplicity, Infura's IPFS API and dedicated gateway connects applications of all sizes to distributed secure storage, paving the way for a more resilient web. Infura is used as gateway to connect to ipfs node.

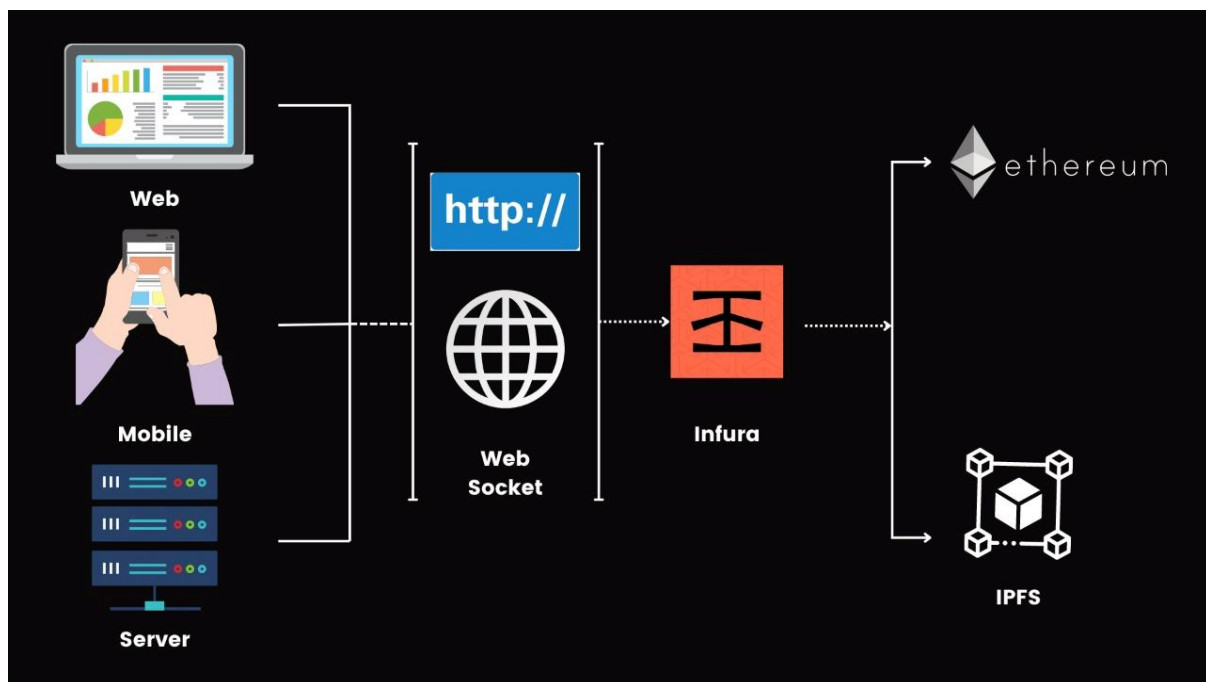


Fig 4 : Files transferring Path to IPFS through Infura

4. MODELING AND IMPLEMENTATION DETAILS

4.1 Design Diagrams

4.1.1 USE CASE DIAGRAM

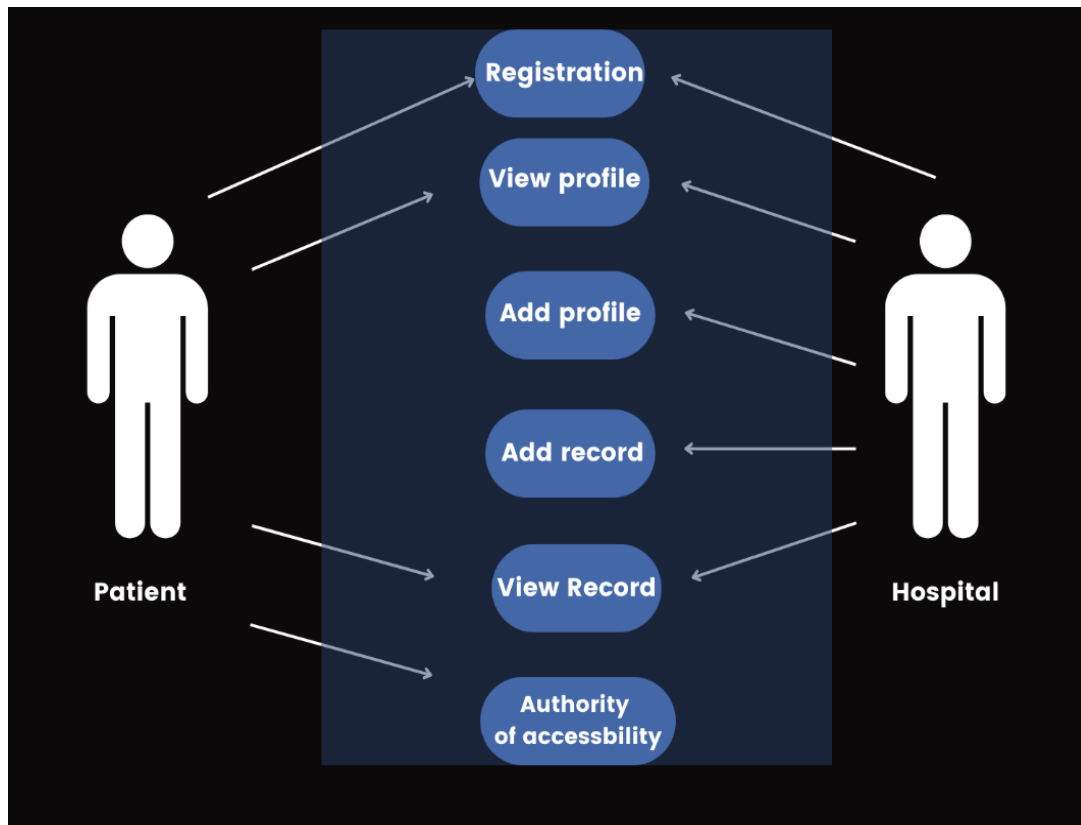
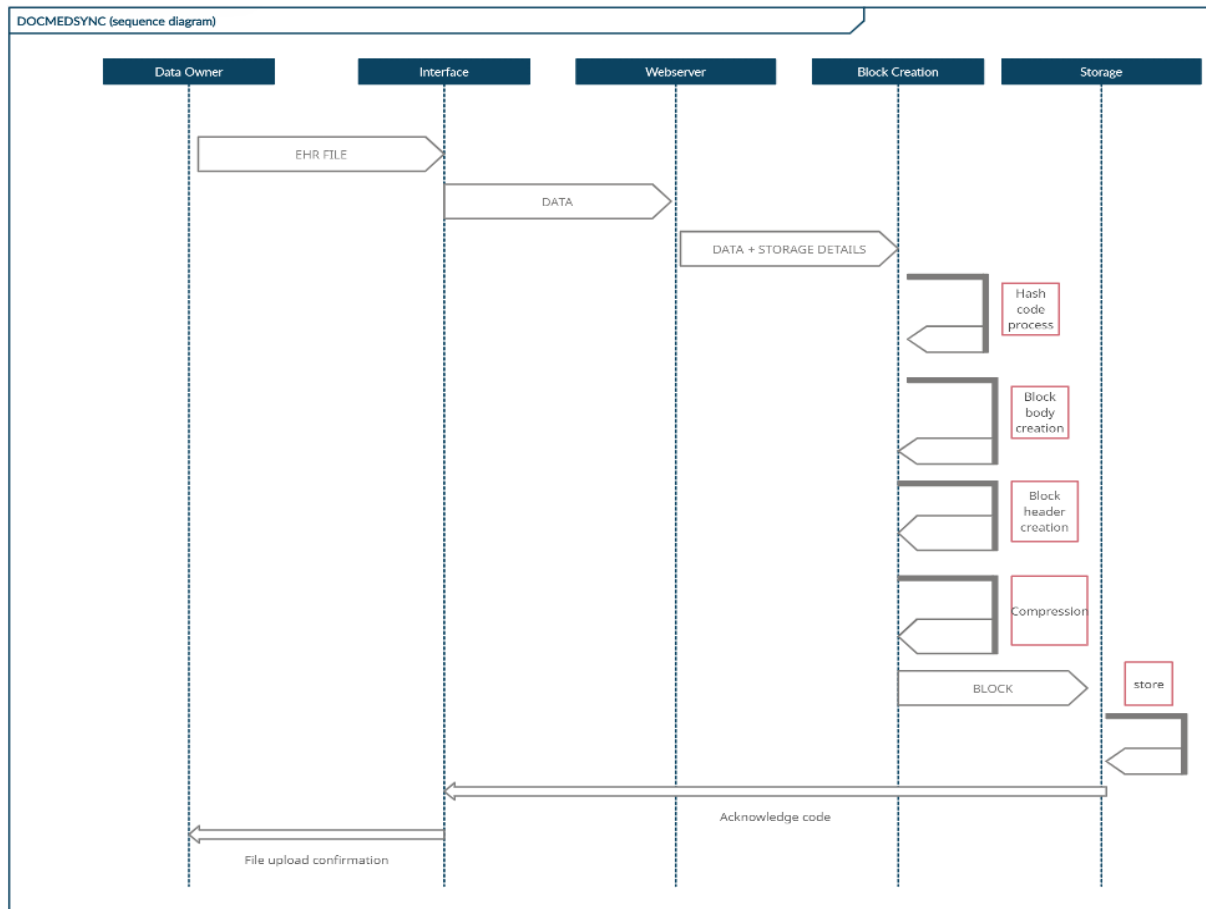


Fig 5 : Use Case Diagram of scenario between Hospital and Patient



4.1.2 Sequence Diagram

Fig 6 : Sequence Diagram

4.1.3 Activity Diagram

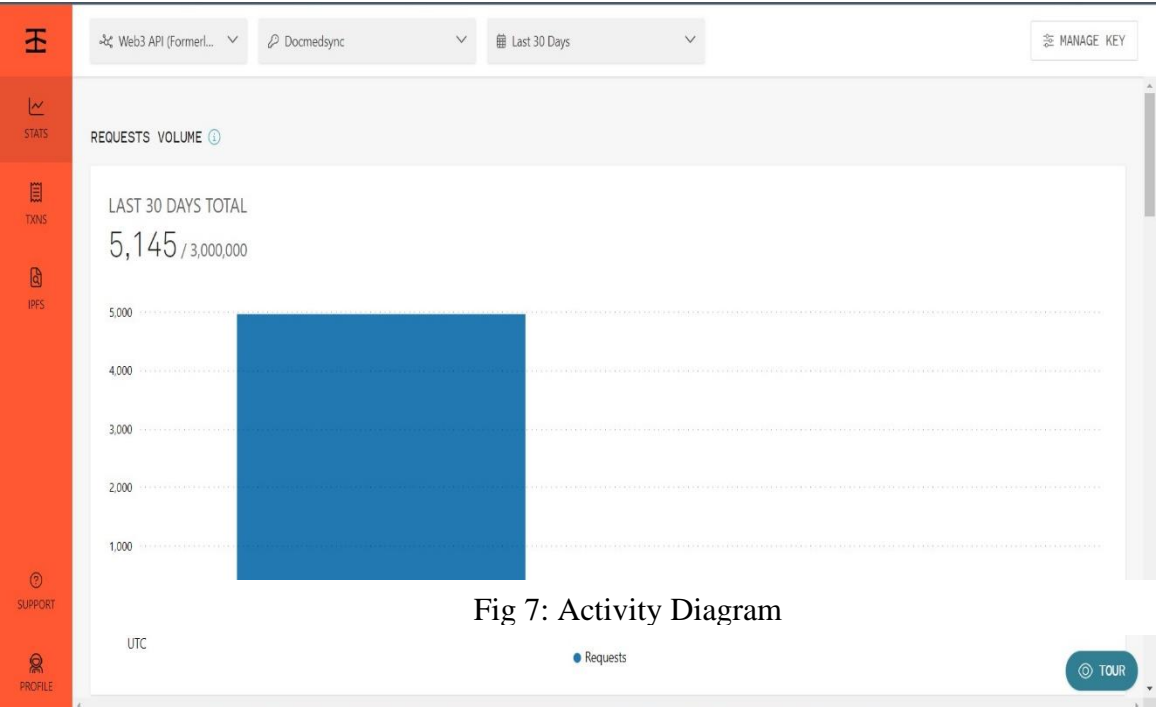


Fig 7: Activity Diagram

4.2 Implementation Details

Fig 8 :Request Volume

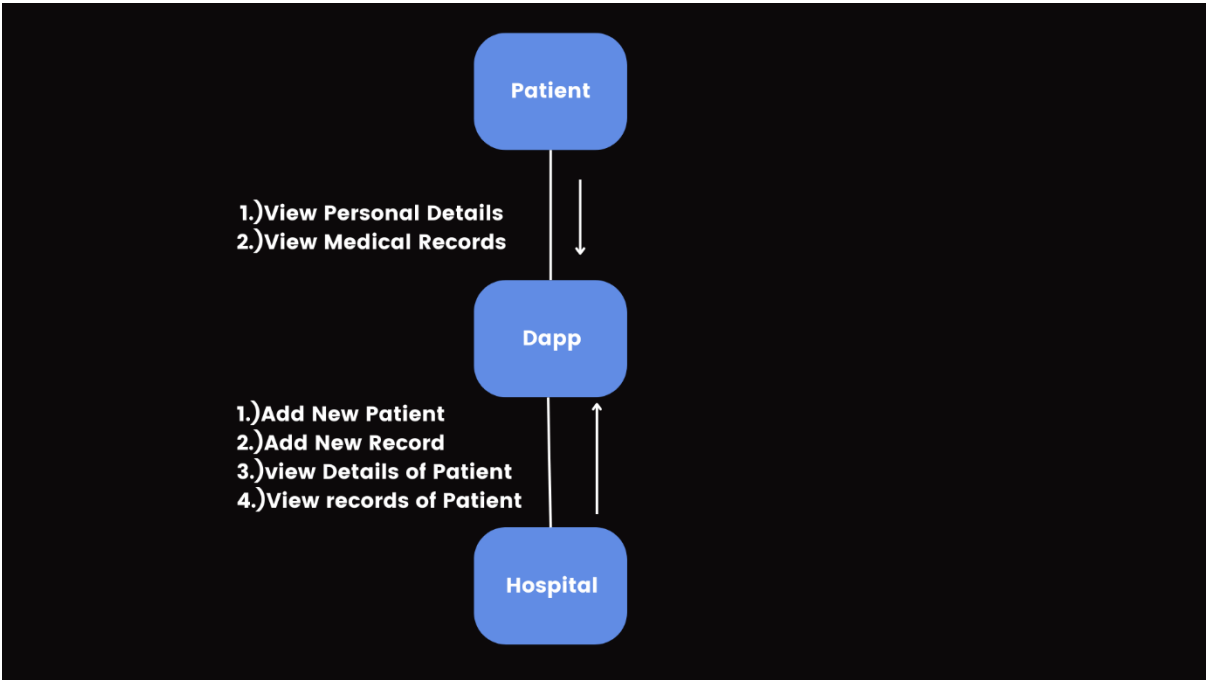


Fig 9: Data Transfer

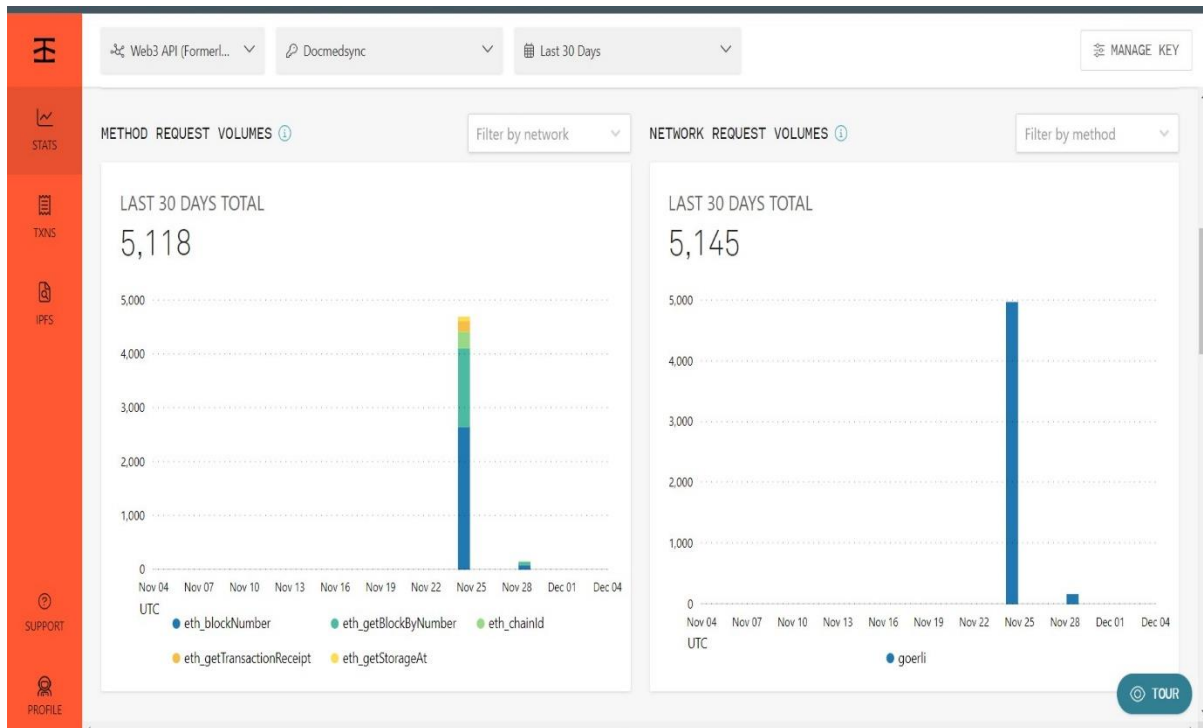


Fig 10: Method Request and network request volumes

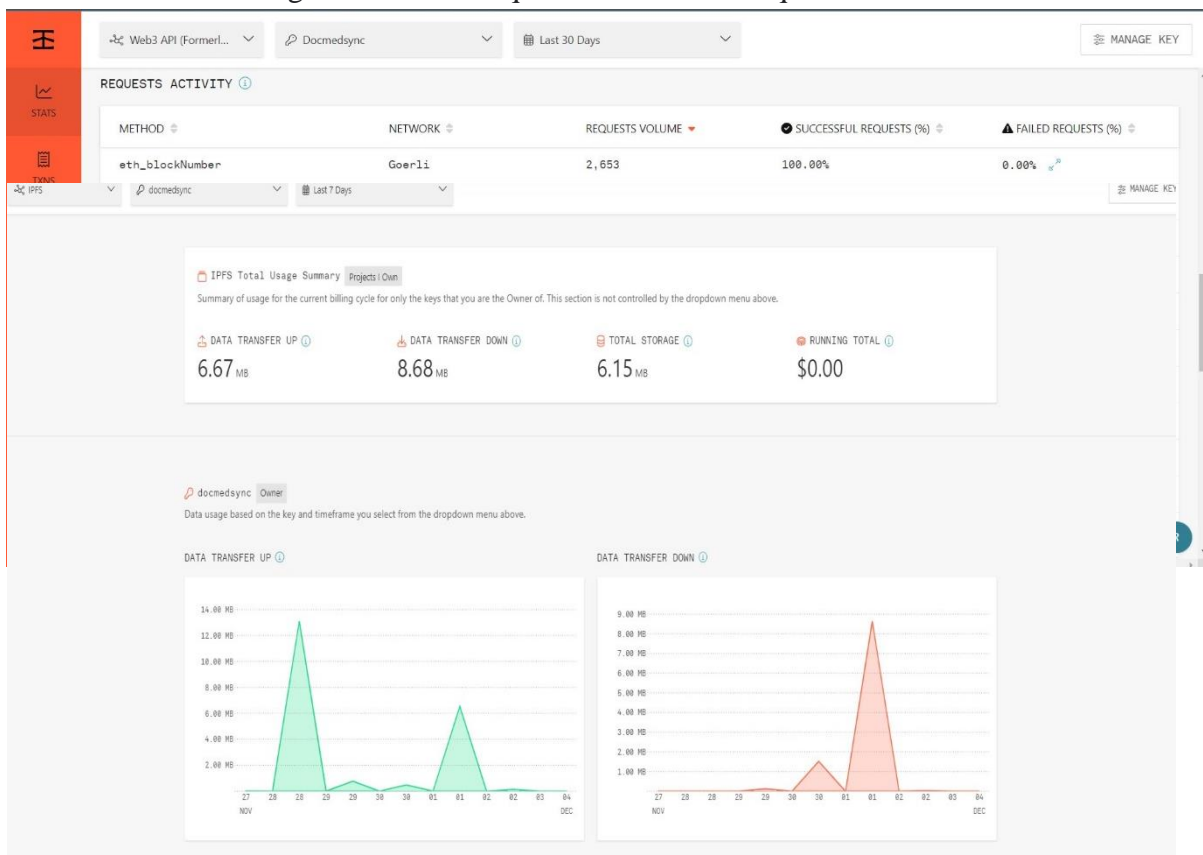


Fig 11: Requests Activity

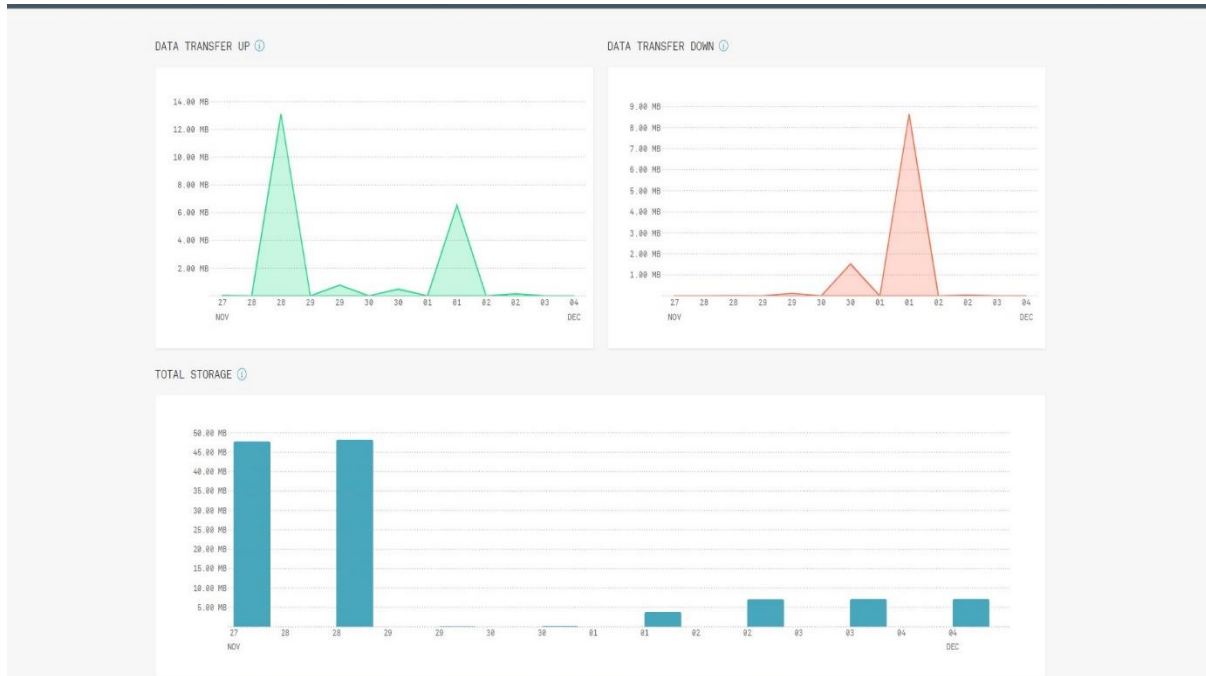


Fig 12: Total Storage

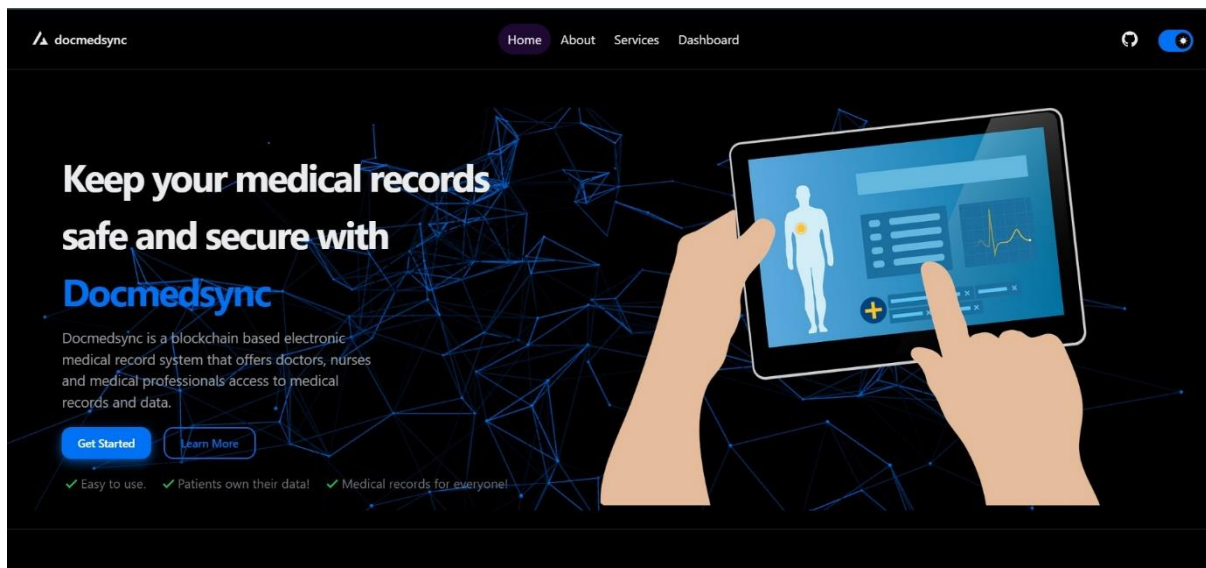


Fig 13: Home Page

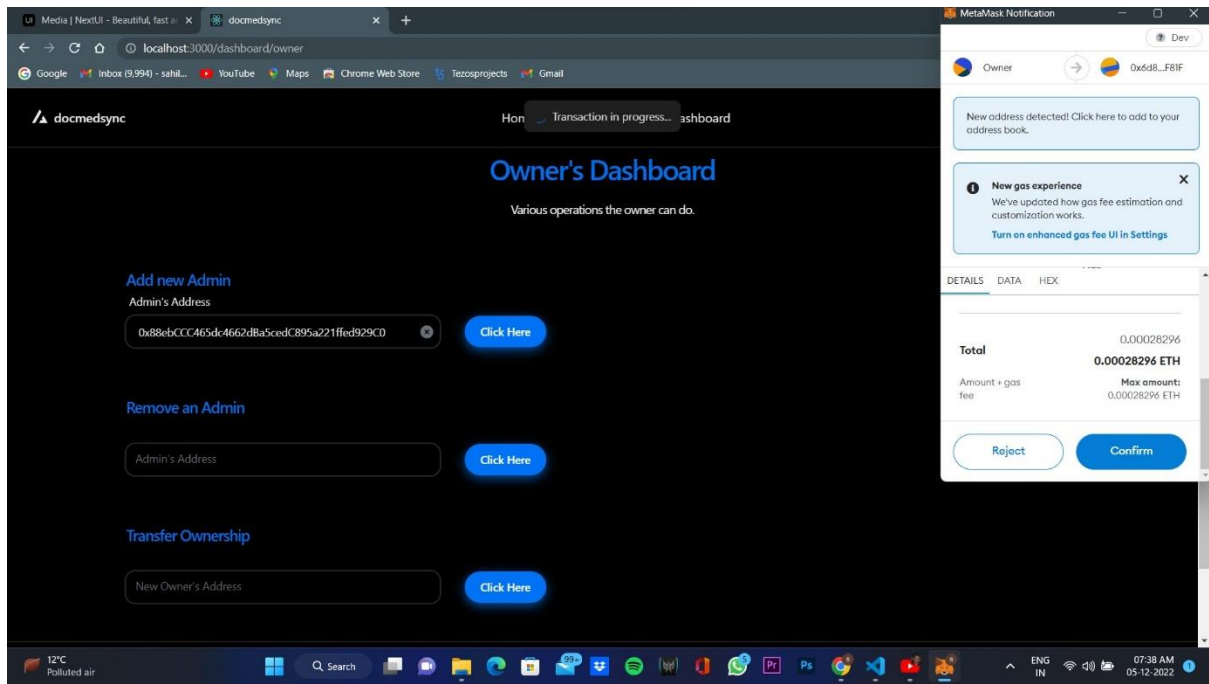


Fig 14: Owner's Dashboard

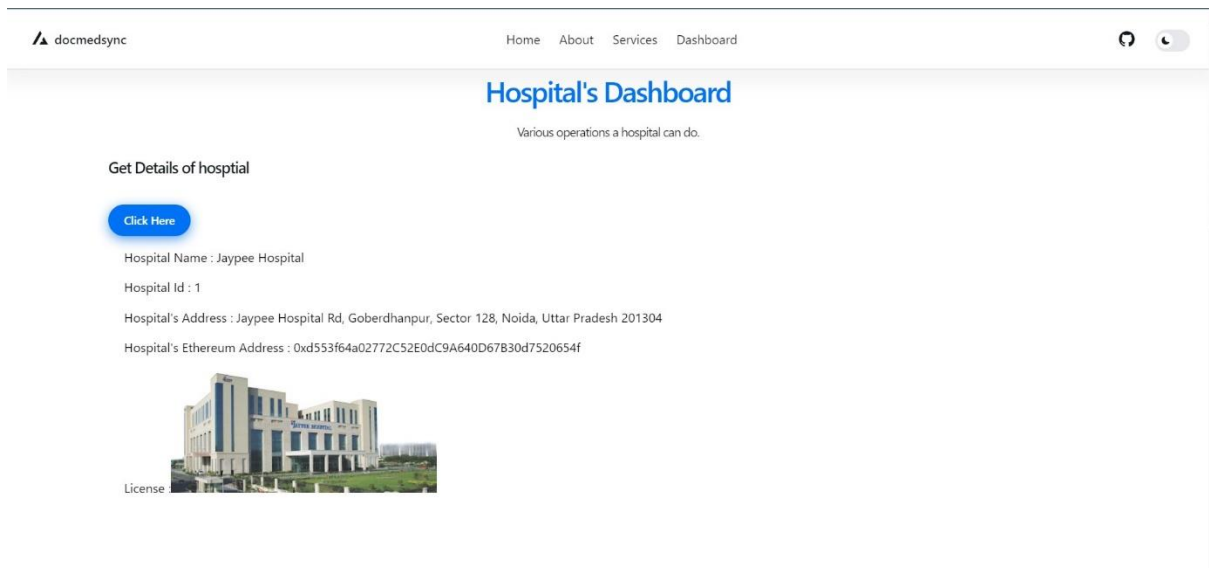


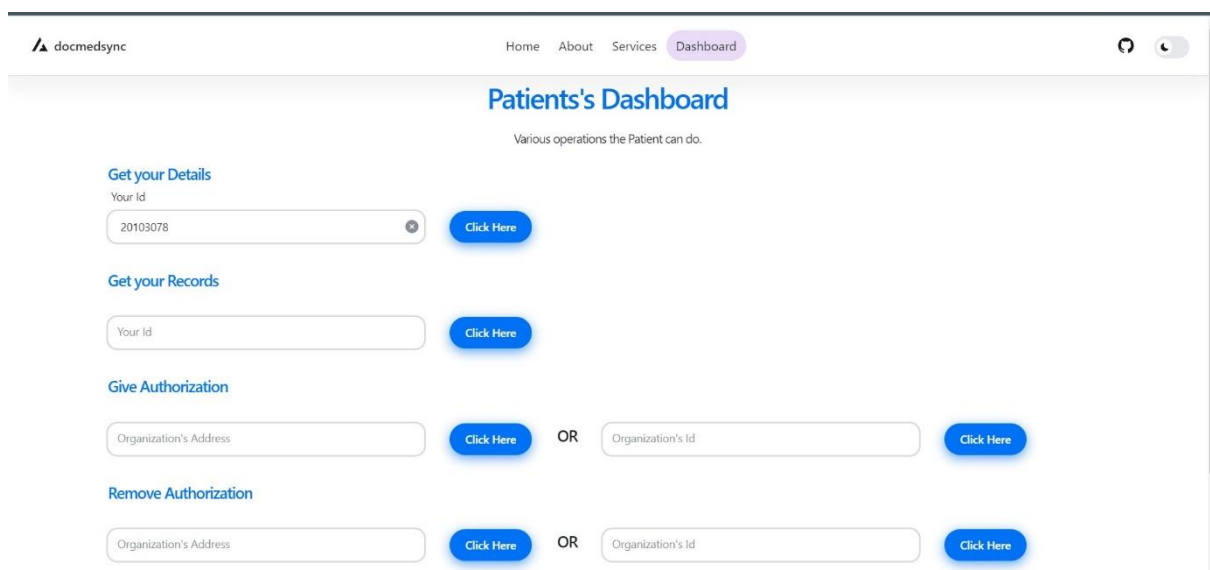
Fig 15: Hospital's Dashboard



The screenshot shows the Docmedsync website's dashboard. At the top, there is a navigation bar with the Docmedsync logo, links for Home, About, Services, and Dashboard (which is highlighted), and a user profile icon. Below the navigation bar, there is a table listing hospitals. The table has three columns: NAME, ADDRESS, and ETHEREUM ADDRESS. Two hospitals are listed: Jaypee Hospital and Fortis Hospital Noida. Below the table, there are social media icons for GitHub, LinkedIn, and Telegram, and a copyright notice: © Copyright Docmedsync. All Rights Reserved.

NAME	ADDRESS	ETHEREUM ADDRESS
Jaypee Hospital	Jaypee Hospital Rd, Gobarhanpur, Sector 128, Noida, Uttar Pradesh 201304	0xd553f64a02772C52E0dC9A640D67B30d7520654f
Fortis Hospital Noida	B-22, Rasoolpur Nawada, D Block, Sector 62, Noida, Uttar Pradesh 201301	0x59B304aa8ca785419BEF5628a6cEeB5d0C64019E

Fig 16: Hospitals list



The screenshot shows the 'Patients' Dashboard on the Docmedsync website. The dashboard has a title 'Patients' Dashboard' and a subtitle 'Various operations the Patient can do.' Below the title, there are four sections: 'Get your Details', 'Get your Records', 'Give Authorization', and 'Remove Authorization'. Each section has a text input field and a 'Click Here' button. The 'Get your Details' section has a dropdown menu for 'Your Id' with the value '20103078'. The 'Get your Records' section has a text input field for 'Your Id'. The 'Give Authorization' section has two text input fields: 'Organization's Address' and 'Organization's Id', with 'Click Here' buttons next to each. The 'Remove Authorization' section has two text input fields: 'Organization's Address' and 'Organization's Id', with 'Click Here' buttons next to each.

Get your Details
Your Id
20103078 [Click Here](#)

Get your Records
Your Id [Click Here](#)

Give Authorization
Organization's Address [Click Here](#) OR Organization's Id [Click Here](#)

Remove Authorization
Organization's Address [Click Here](#) OR Organization's Id [Click Here](#)

Fig 17: Patient's Dashboard

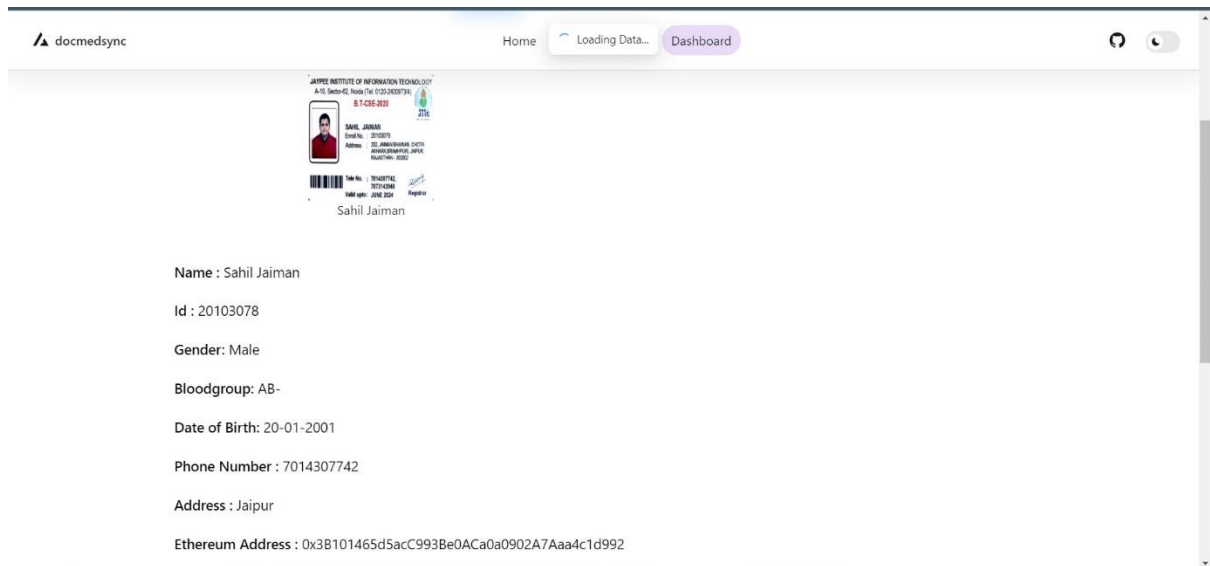


Fig 18: Patient's List

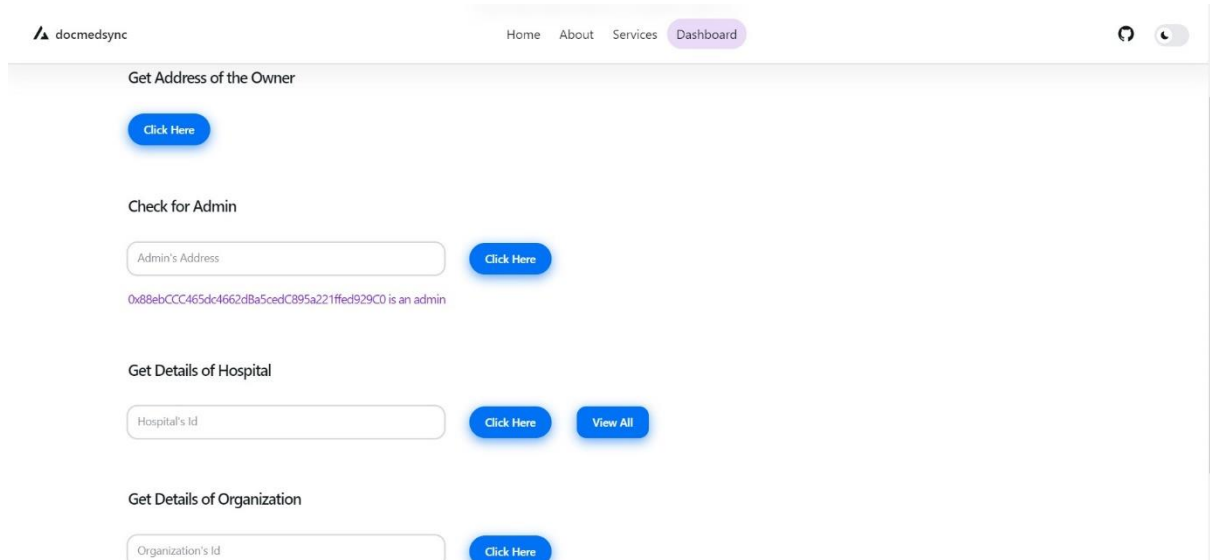


Fig 19: Public's Dashboard

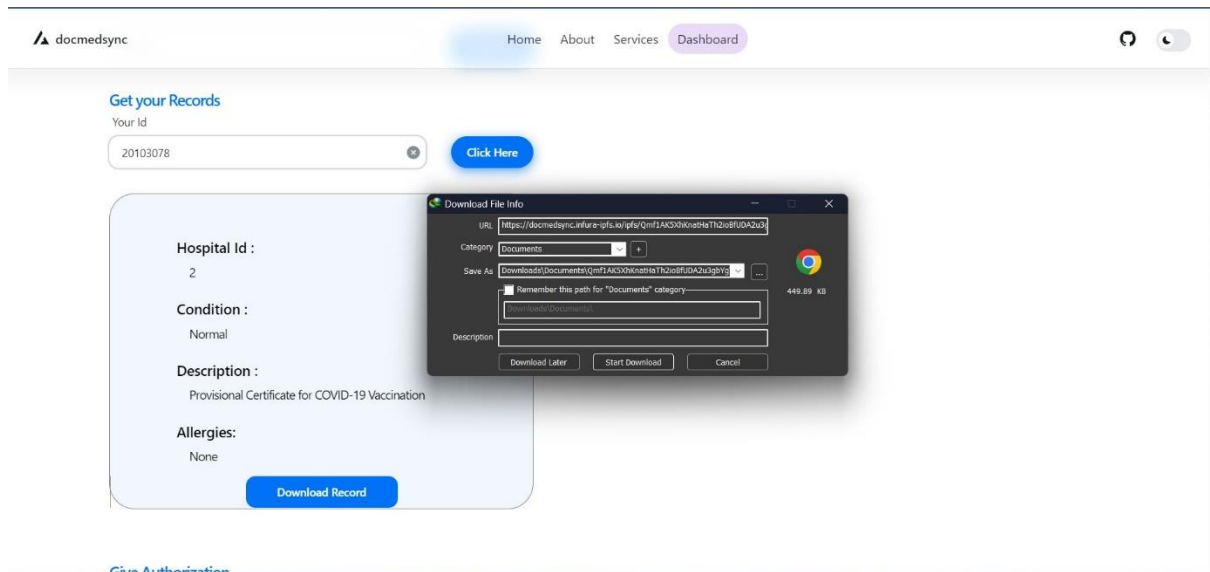


Fig 20: Record

5. Conclusion and Future Work

Blockchain is a newer technology that is already being deployed in healthcare settings for many reasons. This scoping review synthesized blockchain EHR use cases and compelling architectural features and identified historical standards and security advantages and challenges. We identified areas for future research including historical health informatics challenges, such as proper semantic analysis of blockchain data transfer and potential patient-matching concerns.

Future work includes collecting additional analytics on blockchain implementation successes in healthcare settings as blockchain case studies begin multiyear deployment. With longer running case studies and the introduction of newer novel architectures, additional studies could examine more up-to-date analytics, additional data components, and different system architectural designs. For example, collecting institutional review board-approved interviews with healthcare practitioners supporting different job functionalities could produce a comprehensive set of open practitioner questions. Furthermore, as all deployed technology likely leads to data breaches or other adverse challenges with time, future work should analyse adversarial attacks or related challenges to deployment over time.

6. References

1. Dubovitskaya A, Xu Z, Ryu S, Schumacher M, Wang F. Secure and trustable electronic medical records sharing using blockchain. In: AMIA annual symposium proceedings. Vol. 2017. American Medical Informatics Association; 2018: 650–9. [[PMC free article](#)] [[PubMed](#)] [[Google Scholar](#)]
2. G. Jetley and H. Zhang, “Electronic health records in IS research: Quality issues, essential thresholds and remedial actions,” *Decis. Support Syst.*, vol. 126, pp. 113–137, Nov. 2019.
3. G. Jetley and H. Zhang, “Electronic health records in IS research: Quality issues, essential thresholds and remedial actions,” *Decis. Support Syst.*, vol. 126, pp. 113–137, Nov. 2019.
4. S. T. Argaw, N. E. Bempong, B. Eshaya-Chauvin, and A. Flahault, “The state of research on cyberattacks against hospitals and available best practice recommendations: A scoping review,” *BMC Med. Inform. Decis. Making*, vol. 19, no. 1, p. 10, Dec. 2019.
5. Ming Li, Shucheng Yu, and Wenjing Lou, “Scalable and Secure Sharing of Personal Health Records in Cloud Computing using Attribute based Encryption”, *IEEE Transactions On Parallel And Distributed Systems* 2012.
6. IEEE 2012 paper on “Improving the interoperability of healthcare information system through HL7 CDA and CCD standards”
7. S. Yu, C. Wang, K. Ren, and W. Lou, “Achieving secure, scalable, and fine-grained data access control in cloud computing,” in *IEEE INFOCOM’10*, 2010
8. C. Dong, G. Russell, and N. Dulay, “Shared and searchable encrypted data for untrusted servers,” in *Journal of Computer Security*, 2010.
9. “Privacy-preserving personal health record system using attributebased encryption,” Master’s thesis, WORCESTER POLYTECHNIC INSTITUTE, 2011.