ABSTRACT

Blockchain innovation is quickly picking up consideration towards the security of secret information. The healthcare industry is one of the areas of organization where tall hazard includes and it’s pulled in consideration of numerous innovative organizations, so this field required the security for securing their information. By utilizing the Blockchain innovation there are various openings for healthcare industry to accomplish and pick up. Such as decreased exchange costs, expanded straightforwardness for administrative detailing, proficient healthcare information administration and health-care records all-inclusiveness as well as able to get to information from any area. The proposed healthcare system looks for to overcome protection challenges related with the centralized capacity of therapeutic information by presenting a dispersed off-chain demonstrate utilizing IPFS (Interplanetary File System) and blockchain innovation. In the current centralized framework, putting away delicate understanding data postures dangers of unauthorized get to and information abuse. The proposed arrangement includes decentralizing information capacity over a organize of hubs utilizing IPFS, upgrading security and openness. The integration of blockchain guarantees a tamper-resistant and straightforward record for recording therapeutic exchanges, giving an unchanging record of understanding information. Smart contracts are utilized to oversee get to control, permitting as it were authorized substances such as healthcare suppliers get to particular data. This system addresses security concern, moreover guarantees consistency, keenness, and accessibility of restorative information, eventually cultivating a more secure and proficient healthcare data administration framework.

Keywords: Blockchain, Interplanetary File System (IPFS), Healthcare, Decentralized, Privacy

CHAPTER 1 INTRODUCTION

Large amounts of medical data are produced by the healthcare sector, and they must be stored, distributed, and accessed basis. Medical information is created for an event when a patient has imaging tests like CT-SCAN, X-ray, and computed tomography. When a professional prescribes medication, it is another source of this information. Medical records for patients must be stored so that, should the need arise, physicians at other hospitals within the company can access them. In any case, this information needs to be kept confidential. Furthermore, it should be constant. The need for an easy-to-handle system for storing medical records necessitates a structure that makes data efficiently maintained and accessible. Blockchain technology provides a decentralized capacity plot that enables peers (healing centers or specialists) in a healthcare framework to easily communicate a patient's medical record, including symptoms, persistent individual data, prescription drugs, and so on [1].

This innovation ensures a number of aspects, including protection, stability, integrity, and consistency—all essential components of the contemporary healthcare system. A blockchain is composed of a series of blocks with transactions that are connected by a cryptographic hash to guarantee the transactions' durability (medical records). It is possible to expertly preserve and distribute among peers the healing records of visual disorders and a persistent counting history within the blockchain framework [2]. Every record is efficiently available, transportable, and permanently accessible due to the structure of the blockchain. In any case, to manage these massive amounts of medical records, a peer-to-peer, distributed capacity (off-chain) architecture is needed. The framework must also have a specific record capacity.

1.1. SCOPE

We introduce a distributed off-chain storage solution utilizing IPFS and blockchain technology for patient diagnostic reports. Content-addressable and immutable storage is the fundamental model. The protection of patient privacy is the main goal of this framework. We separated our system into three distinct modules: data upload, mining process, and data storage in order to achieve privacy.

The medical professional uses a Web-UI (Web User Interface) to upload each patient's information. Subsequently, the mining procedure is carried out to verify the transactions and maintain network consistency. Finally, patient diagnostic report privacy is ensured by hash-based data storage.

CHAPTER 2 PROBLEM DEFINITION

STATEMENT: Electronic medical records, including patient personal information, diagnostic results, and prescriptions from doctors, are kept by the healthcare industry. Currently, this sensitive data is stored using a centralized storage strategy. The difficulty in maintaining patient privacy is a significant disadvantage of this centralized method. Risks to user privacy include the possibility of patient data and medical records being misused, as well as unauthorized access to sensitive information such as medical conditions and personal identifiers.

SOLUTION: The need for a secure, efficient, and privacy-preserving system for storing and accessing patient data is evident. The proposing system aims to address these issues by implementing a distributed off-chain storage solution using IPFS and blockchain technology. This system will ensure the privacy of patient reports while enabling authorized healthcare providers to access the data easily and securely. The primary problem to be solved is the lack of a secure and efficient method for storing and accessing patient diagnostic reports in the current centralized healthcare data storage model.

CHAPTER 3 LITERATURE REVIEW

The decentralized Interplanetary File System (IPFS)-based blockchain technology for storing and accessing medical records. The main security issues with medical data that this system tackles are data breaches, illegal access, and data integrity. A secure and unbreakable environment for keeping sensitive medical data is provided by the system, which makes use of the distributed nature of IPFS and the immutability of blockchain. This study, "Implementation of Ethereum Blockchain in Healthcare Using IPFS," claims that users are in complete control of their data and can allow authorized users to access it only on a need-to-know basis. The study addresses the possible advantages of the suggested system for enhancing healthcare data security and provides a thorough examination of its security aspects [1].

A unique Blockchain and IPFS based platform for storing patient diagnostic reports. The solution takes advantage of IPFS's distributed and content-addressed characteristics to store the actual reports off-chain. In contrast, blockchain technology ensures data integrity and immutability by storing the cryptographic hashes of the reports. This method reduces the hazards and addresses privacy issues related to centralized storage. This paper, "Distributed off-chain storage of patient diagnostic reports in healthcare system using IPFS and blockchain," describes the specific features, functionalities, and benefits of the suggested system, which include user-controlled access, enhanced data security, and improved accessibility [2].

The healthcare system manages data securely by leveraging IPFS, blockchain, and IoT. It tackles privacy and security issues in current healthcare systems, especially those with client/server architectures. The system's primary function is to monitor patients remotely, particularly for long-term illnesses. To ensure data integrity and tamper-proof records, it uses a private Ethereum blockchain for data security and IPFS for decentralized storage. In this study, "Blockmedcare: a Healthcare System Based on IOT, Blockchain, and IPFS for Data Management Security," the potential for this system to revolutionize healthcare data security and patient privacy is addressed. Proxy re-encryption is used to strengthen data access control [3]

The "Patient-Centric Healthcare Data Management (PCHDM)" a system for storing health records using Hyperledger Fabric (blockchain) and IPFS (decentralized storage). PCHDM addresses scalability limitations of blockchain by storing only hashes of records on-chain, while actual data resides encrypted in IPFS. This ensures immutability and privacy. Patients retain control over their data, granting access to authorized individuals through smart contracts. The paper “Hyperledger Health-Chain: Patient- Centric IPFS-Based Storage of Health Records”, analyzes PCHDM's security and performance, demonstrating its potential for secure and patient-centric healthcare data management [4]

The potential of combining blockchain and IPFS for secure and decentralized storage of medical records. It compares existing solutions, dissecting their architectures to guide future research and development. The paper highlights the benefits of blockchain for creating tamper-proof audit trails and ensuring data integrity. It then explores IPFS's role in efficient and secure off-chain storage of actual medical data. By analyzing strengths and weaknesses of various implementations, the paper “Decentralized secure storage of medical records using blockchain and IPFS: A comparative analysis with future directions, offers valuable insights and paves the way for future advancements in secure healthcare data management [5]

The MedLink, a permissioned blockchain-based system for secure sharing and integration of electronic health records (EHRs). Leveraging IPFS for decentralized storage, MedLink addresses concerns about performance and privacy associated with traditional EHR systems. Patients and authorized healthcare professionals can securely share and access EHRs using a web-based interface. The system utilizes public key infrastructure for encryption and digital signatures, ensuring data confidentiality and integrity. Additionally, blockchain-based smart contracts regulate access permissions, enabling granular control over EHR data sharing. The paper “Secure: Sharing of Medical Records using IPFS”, presents a detailed analysis of MedLink's architecture and its potential to revolutionize secure and efficient healthcare data management [6].

IPFS and blockchain technology application for the protection of data in medical records. Medical record privacy, medical record data traceability, and medical record data integrity verification are all accomplished via a number of apps that are currently accessible. Nevertheless, these applications have a number of problems, some of which are inherent to IPFS and blockchain technology, and some of which are the result of design errors. Improvements to the blockchain-based system and IPFS itself, including hash indexing, encryption algorithms, and consensus mechanisms, should be the main emphasis of future development. In conclusion, this study offers an IPFS and blockchain-based medical health record storage system in "A survey: medical health record data security based on interplanetary file system and blockchain technologies"[7]

CHAPTER 4 PROJECT DESCRIPTION

4.1 WEB 3.0 ARCHITECTURE

Ethereum smart contracts are created in Solidity and deployed to the local blockchain that Ganache provides for testing and development in the project design that makes use of Ganache. To communicate with these smart contracts, an HTML, CSS, and JavaScript user interface is created. The Ganache blockchain and user interface can communicate more easily thanks to the integration of Web3.js. The project specifications are met by Ganache, and thorough unit tests are created and run on the local blockchain. The finished smart contracts can be implemented on the active Ethereum network, but Ganache's controlled environment can be used for continued testing and development. To guarantee clarity and continuity in the project, security considerations—including best practices for the construction of smart contracts—are put into practice, and documentation is kept up to date.[1], [4]

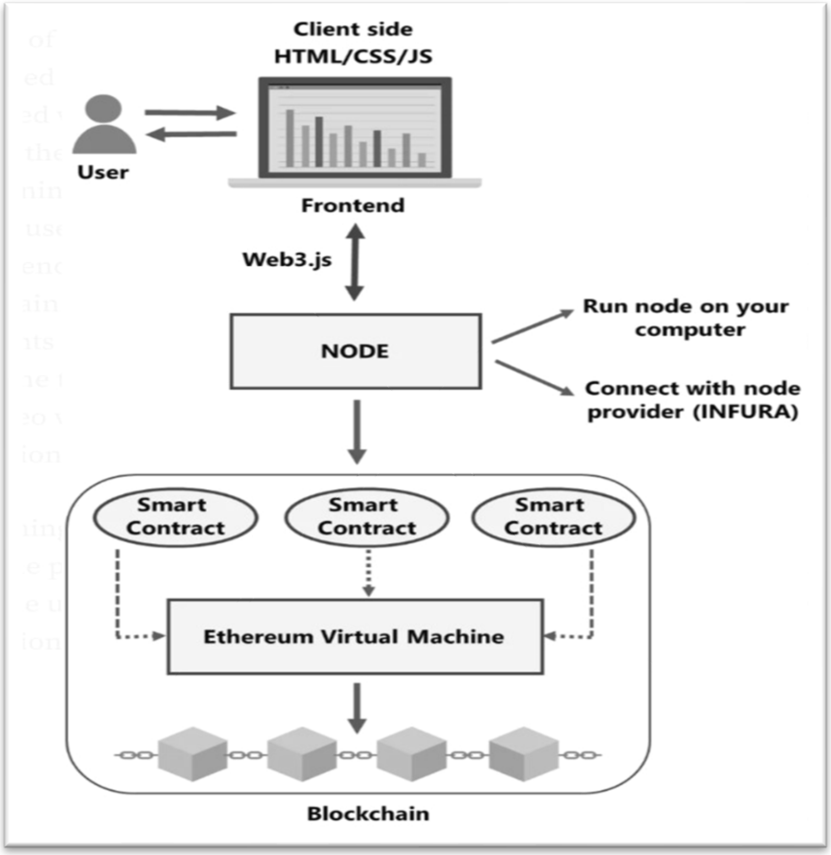


Fig 4.1. Web 3.0 Architecture

4.2 SYSTEM ARCHITECTURE

Building a framework's underlying structure is a step in the architectural configuration process. It involves figuring out the framework's real parts as well as how they interact with one another. The first step in configuring these subsystems and creating a framework for subsystem control and communication is called construction modeling outline. This method results in an illustration of the structural planning of the product. Below is a possible architecture for the system. It provides an illustration of the system's architecture and fundamental functionality.

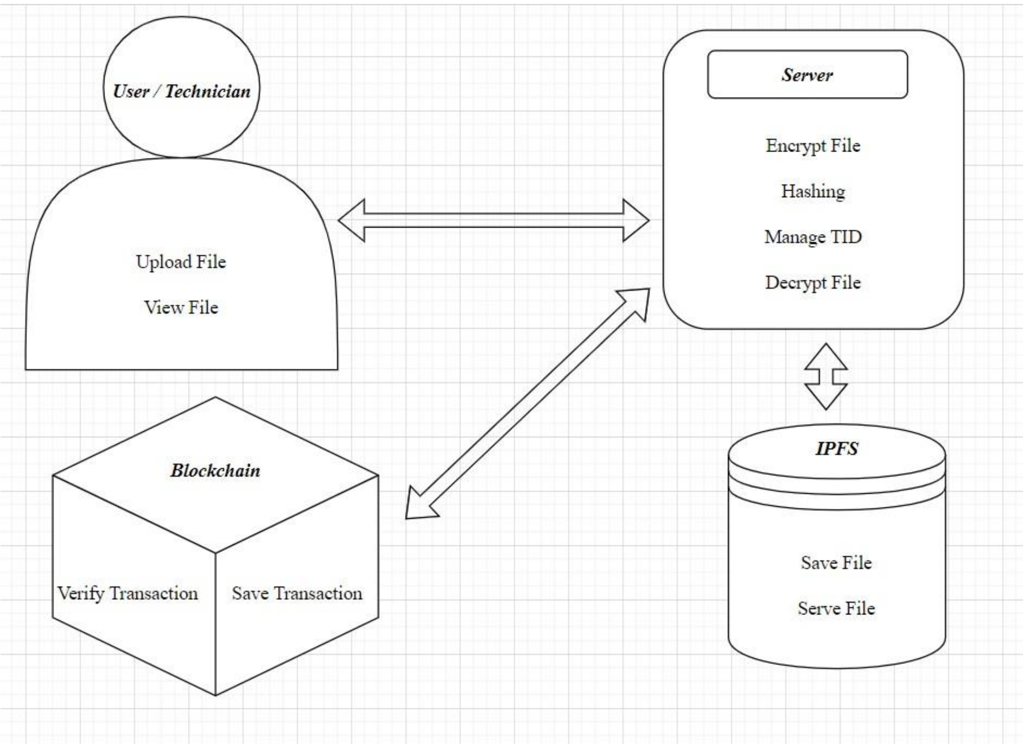


Fig 4.2. System Architecture

CHAPTER 5 REQUIREMENTS

5.1. FUNCTIONAL REQUIREMENTS

1. **User registration and authentication:** User registration: healthcare providers (e.g., hospitals, doctors) must be able to register on the consortium network to obtain a proof-of-identity (poi) or registration-id. User authentication: The system should authenticate registered users to ensure that only authorized entities can access patient data.

2. **Data upload:** Healthcare providers should have the capability to upload patient diagnostic reports using a web user interface. The system must support the secure and efficient uploading of medical data.

3. **Mining process:** The system should implement a mining process using proof-of-work (PoW) to validate transactions and maintain network consistency. Miners should be able to validate uploaded reports and transactions.

4. **Transaction verification and block creation:** Miners should be able to verify transactions by comparing them with their local copies. Verified transactions should be added to the blockchain as blocks.

5. **IPFS integration:** The system must integrate with the Interplanetary File System (IPFS) for distributed off-chain storage. It should store the content-addressed hash of patient reports in the IPFS distributed file storage system.

6. **Data privacy and access control:** Patient data must be securely stored to ensure data privacy. Only authorized healthcare providers who have registered with the network should be able to access patient reports. Implement access control mechanisms to restrict data access to authorized users.

7. **Data retrieval:** Authorized healthcare providers (doctors, nurses) should be able to retrieve patient diagnostic reports securely and efficiently.

8. **Hashed data integrity:** Hashes of the stored reports should be calculated and compared with stored values to ensure data integrity when retrieved.

9. **Cache management:** The system should implement an effective cache management mechanism to store frequently accessed reports, reducing retrieval latency.

10. **Hashed access control:** Hashed access tokens should be used for controlling access to reports, ensuring secure and controlled access to sensitive medical data.

11. **Selective data sharing:** Patients should have the ability to selectively share their diagnostic reports with specific authorized users, enhancing data privacy.

5.2 HARDWARE REQUIREMENTS

System	:	intel I5
Memory	:	8 GB.
Hard Disk	:	160 GB.

5.3 SOFTWARE REQUIREMENTS

Operating System	:	Windows 10/11
Language	:	Python, Java, Solidity
Tools	:	Python IDE, Remix, NetBeans, Ganache
Database	:	MySQL

CHAPTER 6 METHODOLOGY

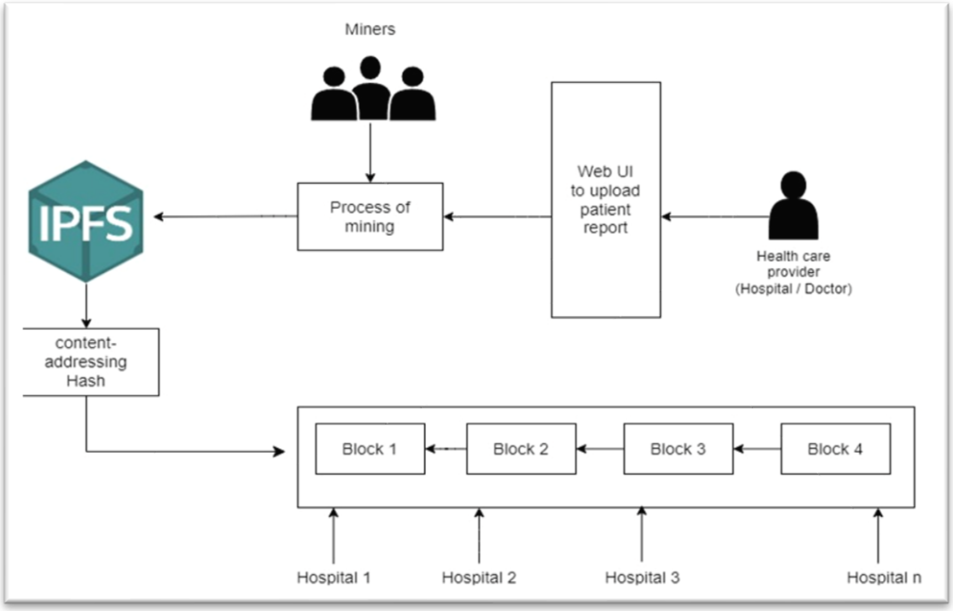


Fig 6.1 High Level Design

High Level Design explains as follows:

- This is the initial overview of the project, Healthcare provider was expected to take the privilege of adding the reports of a particular patient in that healthcare department.
- Once the report is initiated to upload, the process begins with AES-128 encryption, followed by storage of the file in Interplanetary file system. The hash value will be returned with SHA-256 algorithm that will be stored in the block.
- In the same way each and every patient will be having a block in the storage format of Hospital id, Doctor id, and followed by hash value that is pointing to the actual file in the IPFS
- The blocks of multiple patients will be forming a blockchain network

6.1 FRAMEWORK

Ensuring patient record privacy is the primary objective of this approach. The suggested structure makes it easy for authorized entities, including healthcare providers (like doctors and nurses), to access medical data while maintaining patient confidentiality.

The following steps are part of the proposed framework:

- Healthcare providers are required to register in the network. So that the providers will be provided with access to add doctors and patients of that health care department.
- Participants in healthcare such as doctors then utilize a web user interface to upload patient diagnostic reports.
- The uploaded report will be encrypted using AES-128 algorithm and stores in the IPFS, which will return a hash value using SHA-256 algorithm.
- Ganache will simulate the blockchain resources, which are used by Remix.etherium.org to Deploy smart contracts
- IPFS is connected to the block chain using python scripts, and the retrieval of the data is also done using python script which will be connecting the block chain network and IPFS.
- As the Proof of work with real time miners is not that optimistic in healthcare, proof of work (PoW) will be taken care by an algorithm and the python scripts to add the block into the block chain.
- Once the report is uploaded into the IPFS, The Patient will be receiving an email-id with the key for the further authentication purposes.
- Patient authentication is needed to doctor who is willing to view the report.
- Once the doctor sends the request to a patient to view the report, Patient need to enter the 16-byte key from the “Key-request” page, which will result in sending the key to doctor’s email-id, so that doctor can get through the authentication
- After doctor entering the key, decryption of the file using AES-128 will be done and file will be available for the doctor to view. So that Confidentiality is maintained.

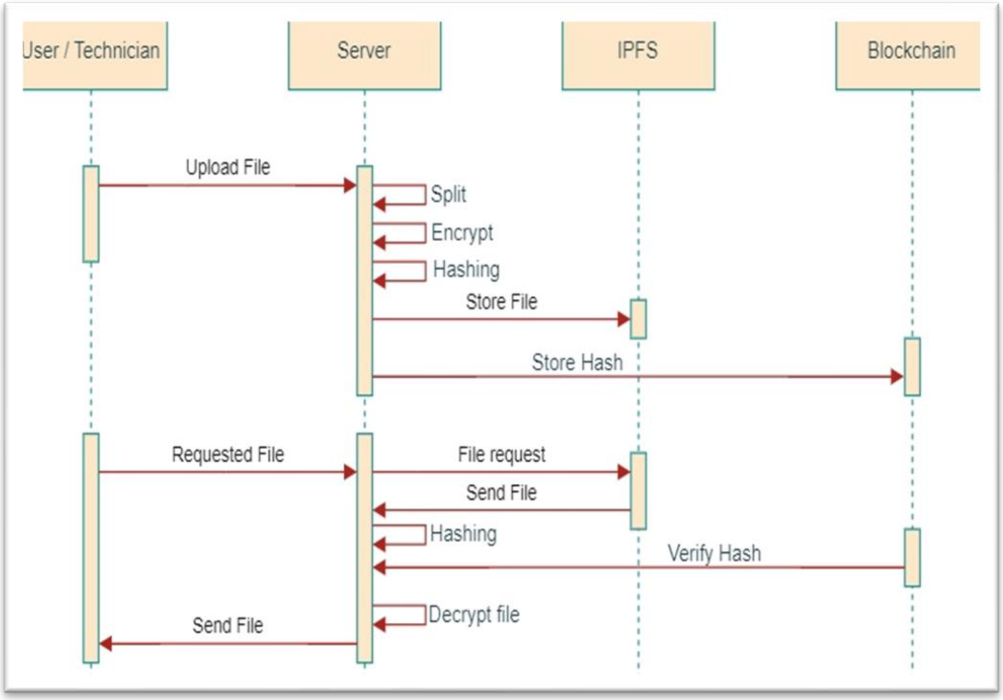


Fig 6.2 Sequence Diagram

In the process of File Management and Retrieval, a secure and reliable system is imperative to ensure data integrity and confidentiality. This sequence diagram outlines the steps involved in both uploading and requesting a file, employing encryption, hashing, and blockchain technology for enhanced security.

While Uploading a File:

- The user initiates the upload process by utilizing the web user interface to submit the report.
- Upon receiving the file, the server undertakes encryption of the report utilizing the AES-128 algorithm, ensuring data security as shown in Fig 6.3 below.
- Following encryption, the server stores the encrypted file securely within IPFS (InterPlanetary File System), a distributed file system designed to maintain data integrity and availability.
- Within the IPFS, a cryptographic hash value is generated for the encrypted file using the SHA-256 algorithm. This hash value serves as a unique identifier for the file's contents.
- Furthermore, the hash value is securely stored within a blockchain, providing an immutable record of the file's existence and integrity.

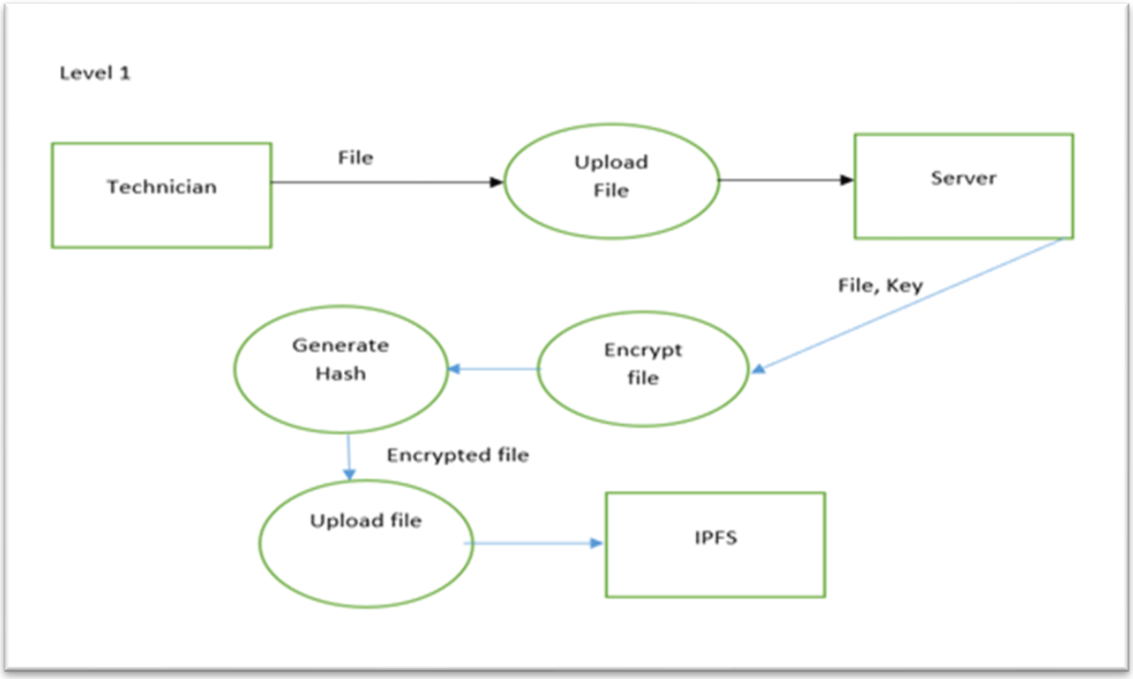


Fig 6.3 Uploading a File into IPFS

While Requesting the File:

- In the event of a file request as shown in Fig 6.4 below, the user communicates with the IPFS to retrieve the desired file.
- Upon receiving the file, the user performs a hashing technique on the retrieved file, generating a hash value locally.
- This locally generated hash value is then compared with the hash value stored within the blockchain, serving as a means of verifying the integrity of the retrieved file.
- Following successful verification, the encrypted file is decrypted by the user, utilizing an AES-128 decryption method.
- Finally, the decrypted file is delivered to the user, ensuring the completion of the file retrieval process.

The described sequence ensures the secure and reliable transfer of files, maintaining data integrity throughout the process.

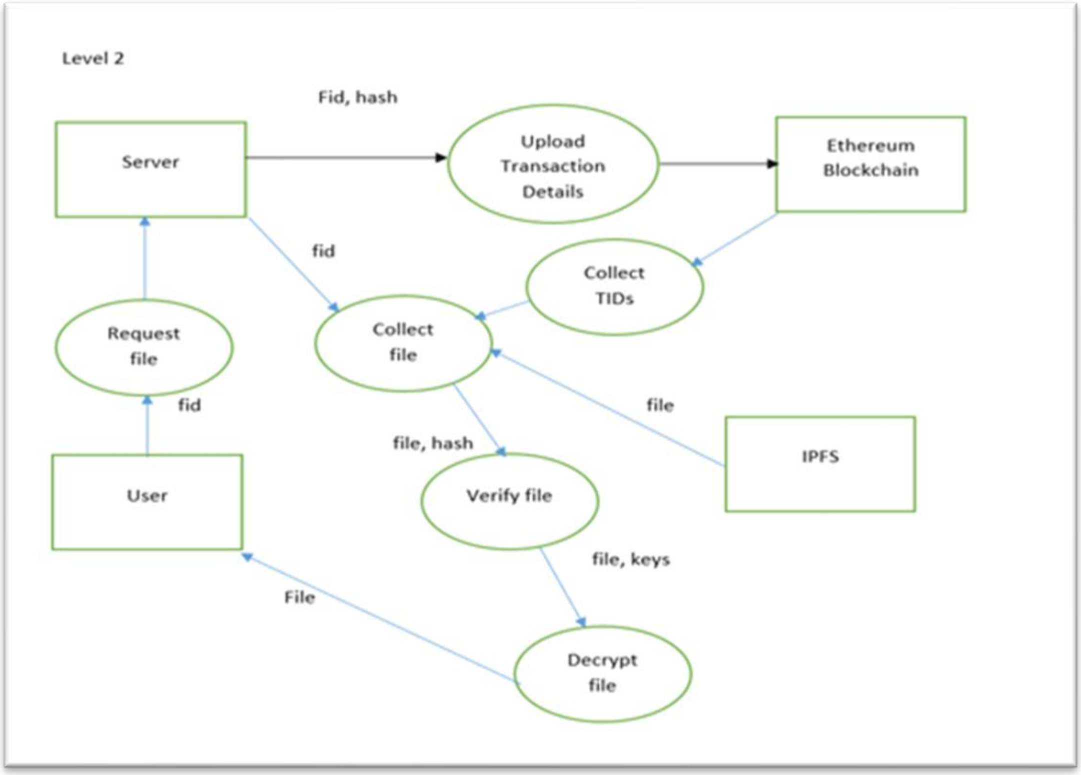


Fig 6.4 Requesting a File from IPFS

CHAPTER 7 EXPERIMENTATION

Step 1: Install and Configure Software Requirements

Ganache: Ganache provides a local blockchain for Ethereum development. Install it and ensure it's running.

Netbeans: Netbeans is an Integrated Development Environment (IDE) for Java development. Install it and set it up for the project.

Navicat: Navicat is a database management tool. Install it to manage your database.

Remix.ethereum.org: This is an online Solidity compiler and IDE. You can configure it by accessing the settings and pasting the IP address from Ganache.

IPFS: An open way for storing and exchanging hypermedia between peers in a distributed file system is achieved using the IPFS protocol and network. For the project, install and setup it.

Step 2: Open Ganache and Configure Remix

Open Ganache and copy the provided IP address. Access remix.ethereum.org, go to settings, and paste the Ganache IP address in the Remix section to connect Remix to our local blockchain.

Step 3: Run and Deploy Smart Contracts

Write the required smart contracts in Remix and deploy them to our local blockchain using Ganache. Make sure to test them thoroughly.

Step 4: Copy the Checksum Address

After deploying the smart contracts, copy the checksum address of the deployed block. This address will be used to interact with the deployed contracts.

Step 5: Configure and Run IPFS

Set up and run IPFS. This involves configuring the IPFS node and ensuring it's running correctly to store and retrieve files.

Step 6: Modify Upload.py and Download.py

Update the checksum address obtained earlier in the upload.py and download.py files. These files likely contain code to interact with the deployed smart contracts.

Step 7: Run Project in Netbeans and ensure it compiles without errors

Step 8: Create or Login Using Doctor Credentials

In our application, provide functionality for doctors to create accounts or login using existing credentials.

Step 9: Select the Patient from the List

Implement a feature where doctors can select patients from a list to manage their medical records.

Step 10: Add Report by Choosing File

Enable doctors to add reports by uploading files through the application's interface, and also generate the authentication key to the patient registered email-id so that patient acknowledgement is concerned.

Step 11: Upload Report into IPFS

Implement functionality to upload the report file to IPFS. The uploaded report is then encrypted and stored in IPFS. Encryption is performed using AES-128 algorithm, a symmetric cryptographic technique that utilizes the same key for both encryption and decryption. IPFS generates a hash value for the encrypted file using SHA-256 algorithm and store the hash-value in the blockchain using the deployed smart contracts. The 16 bytes authentication key will be sent to the patient email-id for further authentication.

Step 12: Login Using Patient Credentials

Provide a login interface for patients to access their medical records.

Step 13: Click on View Report

Implement functionality for patients to view their uploaded reports. The reports are encrypted, to decrypt them we are using AES-128 algorithm and allow patients to download the decrypted file.

Step 14: Patient to View Report

Finally, the patient's medical report will display in a readable format within our application.

Step 15: Doctor to View report with Patient Authentication

Login using doctor credentials and in the home page there will be an option to view the patient records. The Key request will be sent to patient which helps patients to authenticate.

Step 16: Patient authentication

In the patient web interface, there will be an icon called Key Request, which will let patient know if there are any Authentications need to be addressed.

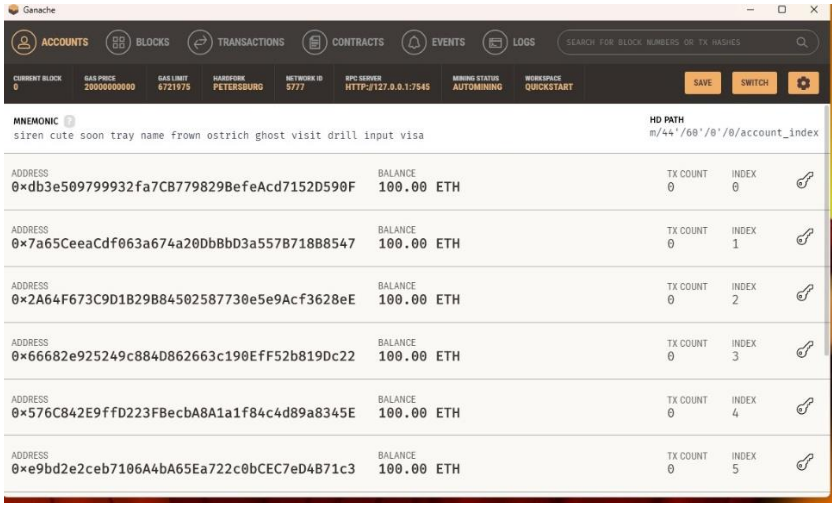
Step 17: The Authentication Addressing

After reviewing the key-requests, the key will be sent to the doctor email-id, so that the authentication will be successful and integrity is maintained.

Step 18: Doctor to view report

Finally, the patient's medical report will display in a readable format within our application. These steps provide a comprehensive guide for setting up, configuring, and using the system. Each step plays a crucial role in ensuring the functionality and security of the application

CHAPTER 8 TESTING AND RESULTS



CURRENT BLOCK 0 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK PETERSBURG | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:7545 | MINING STATUS AUTOMINING | WORKSPACE QUICKSTART

MNEMONIC
siren cute soon tray name frown ostrich ghost visit drill input visa

HD PATH
m/44'/60'/0'/0/account_index

ADDRESS	BALANCE	TX COUNT	INDEX
0x db3e509799932fa7CB779829BefeAcd7152D590F	100.00 ETH	0	0
0x7a65CeeaCdf063a674a20DbBbD3a557B718B8547	100.00 ETH	0	1
0x2A64F673C9D1B29B84502587730e5e9Acf3628eE	100.00 ETH	0	2
0x66682e925249c884D862663c190EfF52b819Dc22	100.00 ETH	0	3
0x576C842E9ffD223FBecbA8A1a1f84c4d89a8345E	100.00 ETH	0	4
0xe9bd2e2ceb7106A4bA65Ea722c0bCEC7eD4B71c3	100.00 ETH	0	5

Fig 8.1 Ganache Environment

As application needs to run on Ethereum, using real Ethereum for the experimental purposes is highly impractical. Ganache is the tool that helps us to use virtual Ethereum. This is how ganache looks like once it is quick started.

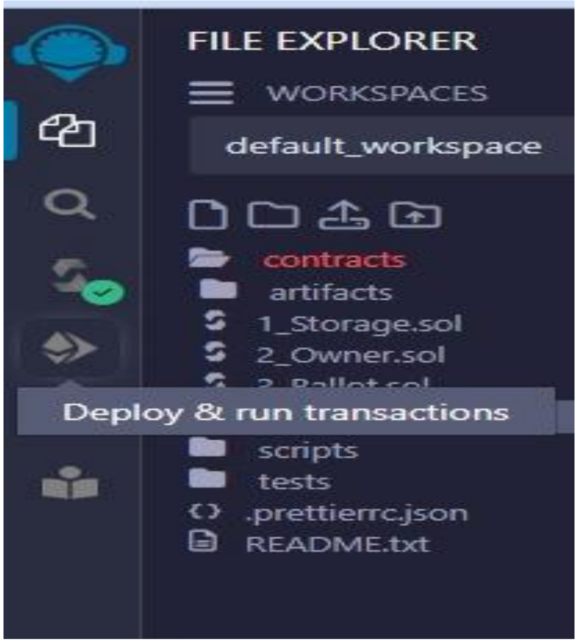


Fig 8.2 Run and Deploy Smart contracts

It is complex for the web application to interact with block chain network without any intervention, so Remix.ethereum.org will be acting as an interpreter which will be responsible to add the block into the network using smart contracts. This is interface of remix.ethereum.org allowing to run and deploy smart contracts

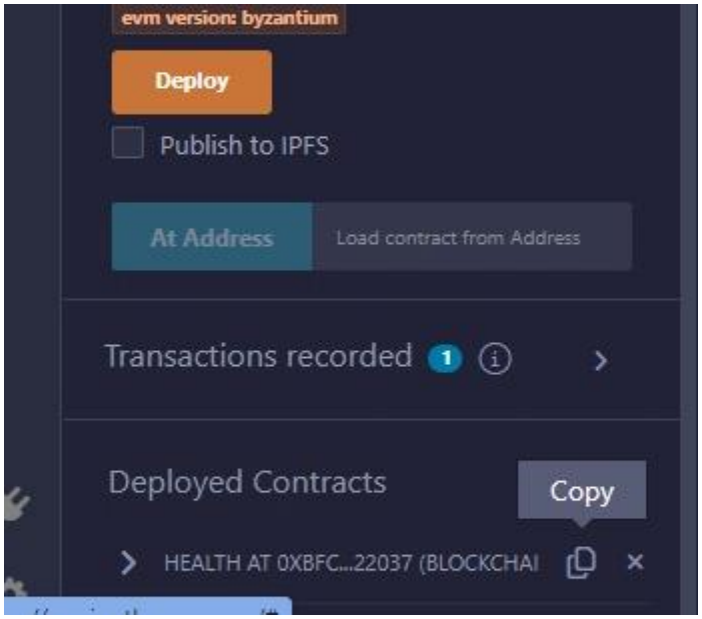
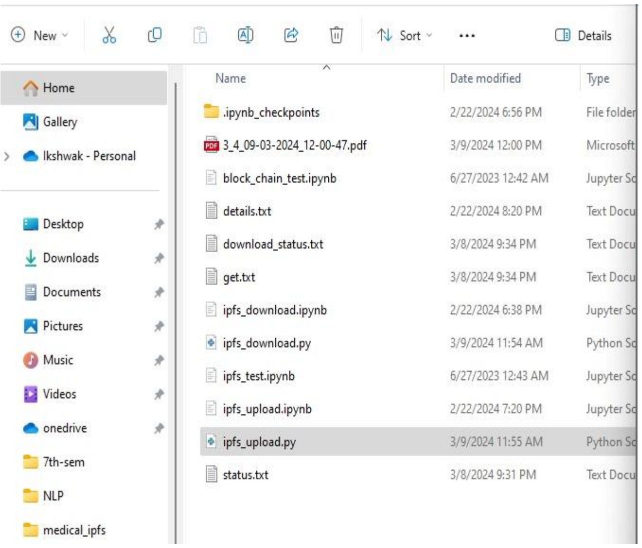
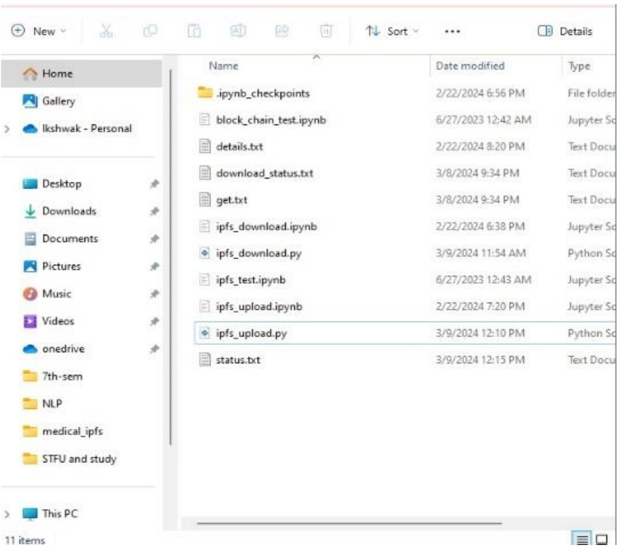


Fig 8.3 Checksum Address

After deploying the smart contract, Remix.etherum.org will be providing a checksum address acknowledging the location to add block into the block chain. This address needs to be updated in the scripts, that are responsible to connect IPFS and blockchain.



Name	Date modified	Type
.ipynb_checkpoints	2/22/2024 6:56 PM	File folder
3_4_09-03-2024_12-00-47.pdf	3/9/2024 12:00 PM	Microsoft
block_chain_test.ipynb	6/27/2023 12:42 AM	Jupyter Sc
details.txt	2/22/2024 8:20 PM	Text Docu
download_status.txt	3/8/2024 9:34 PM	Text Docu
get.txt	3/8/2024 9:34 PM	Text Docu
ipfs_download.ipynb	2/22/2024 6:38 PM	Jupyter Sc
ipfs_download.py	3/9/2024 11:54 AM	Python Sc
ipfs_test.ipynb	6/27/2023 12:43 AM	Jupyter Sc
ipfs_upload.ipynb	2/22/2024 7:20 PM	Jupyter Sc
ipfs_upload.py	3/9/2024 11:55 AM	Python Sc
status.txt	3/8/2024 9:31 PM	Text Docu

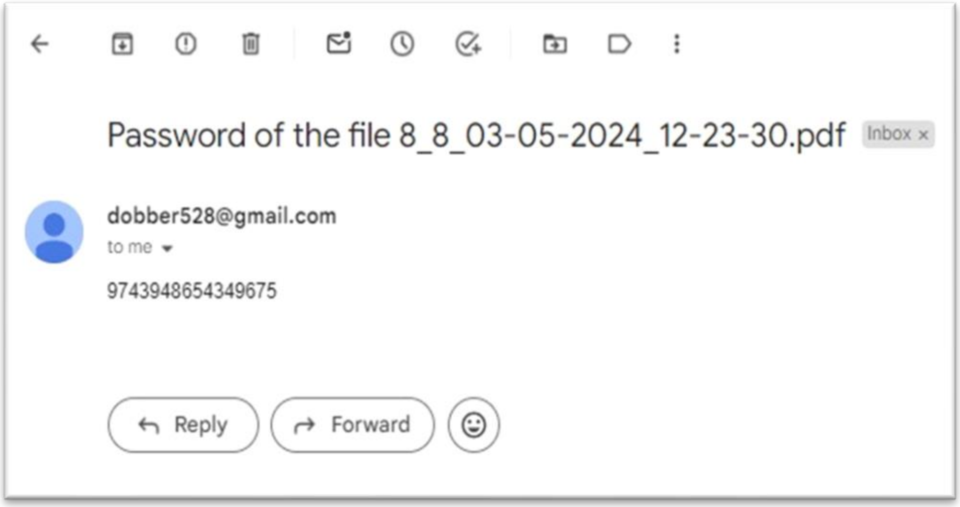


Name	Date modified	Type
.ipynb_checkpoints	2/22/2024 6:56 PM	File folder
block_chain_test.ipynb	6/27/2023 12:42 AM	Jupyter Sc
details.txt	2/22/2024 8:20 PM	Text Docu
download_status.txt	3/8/2024 9:34 PM	Text Docu
get.txt	3/8/2024 9:34 PM	Text Docu
ipfs_download.ipynb	2/22/2024 6:38 PM	Jupyter Sc
ipfs_download.py	3/9/2024 11:54 AM	Python Sc
ipfs_test.ipynb	6/27/2023 12:43 AM	Jupyter Sc
ipfs_upload.ipynb	2/22/2024 7:20 PM	Jupyter Sc
ipfs_upload.py	3/9/2024 12:10 PM	Python Sc
status.txt	3/9/2024 12:15 PM	Text Docu

Fig 8.4 Uploading the Report

Fig 8.5 Encrypting and Hashing the File

Once the report is uploaded it will get encrypted as the process mentioned above, after encryption of the file the hash value will be generated and then the file, hash value stored in the IPFS, and block with format respectively



Password of the file 8_8_03-05-2024_12-23-30.pdf

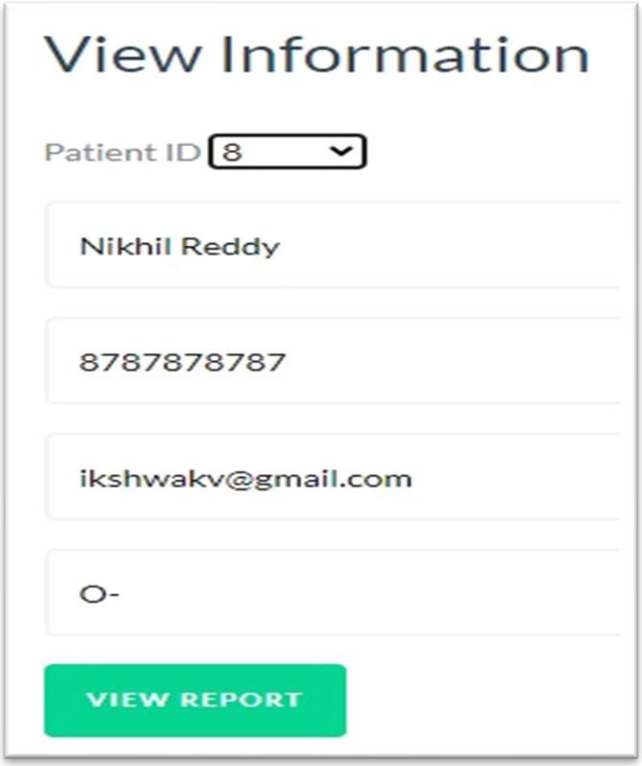
dobber528@gmail.com

to me

9743948654349675

Fig 8.6 16-byte key

Patient will be receiving a 16-byte key through mail soon after the report is uploaded.



View Information

Patient ID 8

Nikhil Reddy

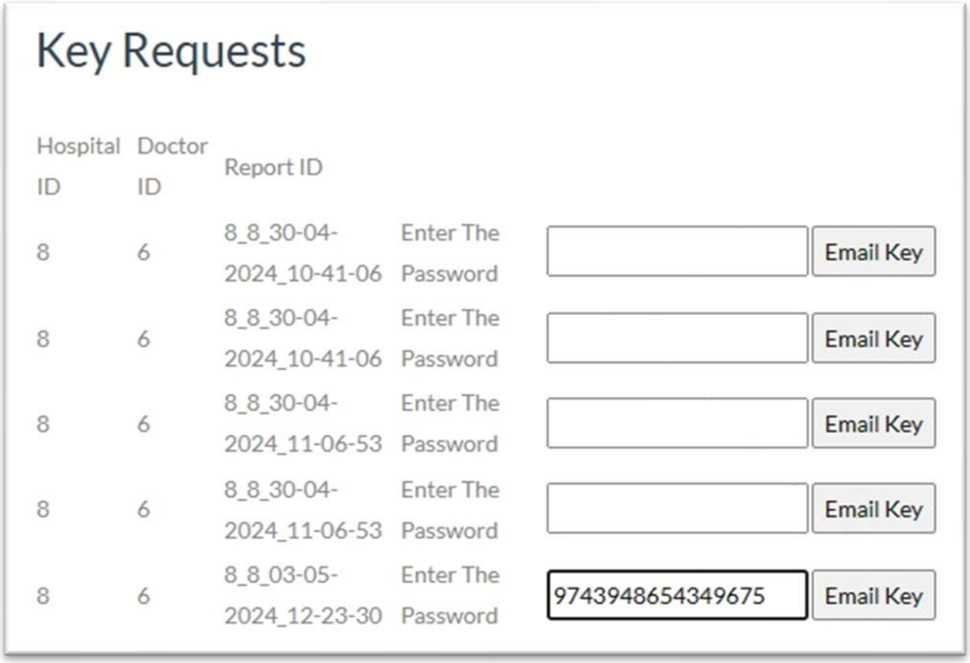
8787878787

ikshwakv@gmail.com

O-

Fig 8.7 Request approval sent to the patient

When doctor wants to view the patient information, it will send a key request to that patient

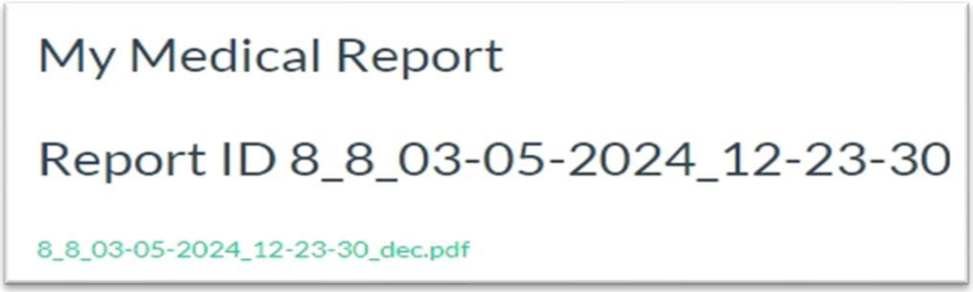


Key Requests

Hospital ID	Doctor ID	Report ID			
8	6	8_8_30-04-2024_10-41-06	Enter The Password		Email Key
8	6	8_8_30-04-2024_10-41-06	Enter The Password		Email Key
8	6	8_8_30-04-2024_11-06-53	Enter The Password		Email Key
8	6	8_8_30-04-2024_11-06-53	Enter The Password		Email Key
8	6	8_8_03-05-2024_12-23-30	Enter The Password	9743948654349675	Email Key

Fig 8.8 Key Request

Patient will be receiving a key request whenever doctor request to view a report. To allow patient will be approving the request by sending the key through mail.



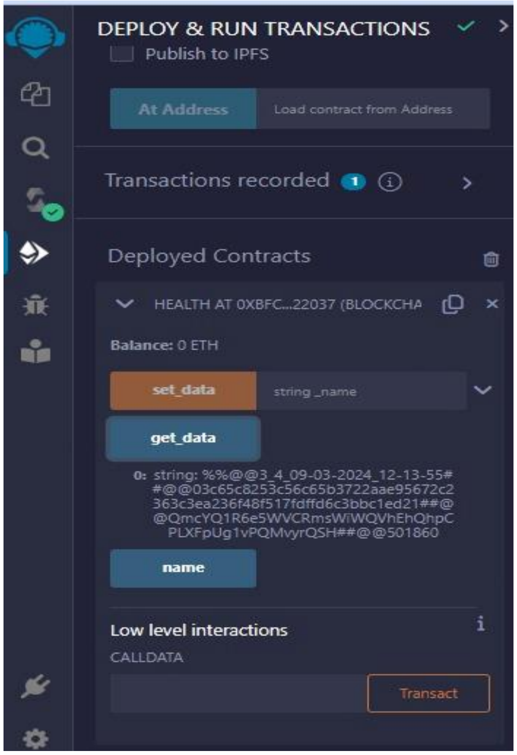
My Medical Report

Report ID 8_8_03-05-2024_12-23-30

8_8_03-05-2024_12-23-30_dec.pdf

Fig 8.9 View Report

After doctor receiving the key. Doctor will be able to decrypt and view the file



HEALTH AT 0XBFC...22037 (BLOCKCHA

Balance: 0 ETH

```
0: string: %%@@3_4_09-03-2024_12-13-55#
   #@@03c65c8253c56c65b3722aae95672c2
   363c3ea236f48f517fdffd6c3bbc1ed21##@
   @QmcYQ1R6e5WVCRmsWiWQVhEhQhpC
     PLXFpUg1vPQMvyrQSH##@@501860
```

Fig 8.10 Block created with Hash value

The block is created in the format of Hospital -ID, Doctor -ID, date, time and followed by the hash value of the uploaded file.

REFERENCES

[1] N. Rauta and K. Shah, “Implementation of Ethereum Blockchain in Healthcare Using IPFS,” *International Journal of Intelligent Communication, Computing and Networks*, May 2021, doi: 10.51735/ijiccn/001/17.

[2] IEEE Communications Society and Institute of Electrical and Electronics Engineers, *2020 International Conference on Communication Systems & Networks (COMSNETS).*

[3] K. Azbeg, O. Ouchetto, and S. Jai Andaloussi, “BlockMedCare: A healthcare system based on IoT, Blockchain and IPFS for data management security,” *Egyptian Informatics Journal*, vol. 23, no. 2, pp. 329–343, Jul. 2022, doi: 10.1016/j.eij.2022.02.004.

[4] V. Mani, P. Manickam, Y. Alotaibi, S. Alghamdi, and O. I. Khalaf, “Hyperledger healthchain: Patient-centric ipfs-based storage of health records,” *Electronics (Switzerland)*, vol. 10, no. 23, Dec. 2021, doi: 10.3390/electronics10233003.

[5] S. Kumar, A. K. Bharti, and R. Amin, “ Decentralized secure storage of medical records using Blockchain and IPFS : A comparative analysis with future directions ,” *SECURITY AND PRIVACY*, vol. 4, no. 5, Sep. 2021, doi: 10.1002/spy2.162.

[6] R. Sathya, G. Govind, M. Athif, M. Zaid, and N. Kumar, “Issue 6 www.jetir.org (ISSN-2349-5162),” 2022. [Online]. Available: www.jetir.org

[7] R. A. Al-Kaabi and A. A. Abdullah, “A survey: medical health record data security based on interplanetary file system and blockchain technologies,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 30, no. 1, pp. 586–597, Apr. 2023, doi: 10.11591/ijeecs.v30.i1.pp586-597.