Medical Records using Blockchain

Submitted in partial fulfilment of the requirements of the degree of

THIRD YEAR ENGINEERING IN

Computer Science and Engineering

(Internet of Things and Cyber Security Including Block Chain Technology)

By

NADAR KATHIR SHEWAG THANGIAH	**-**	**201111025**
NIKHIL RAMKUMAR BOURASI	**-**	**201111018**
MANSI DIPAK PISE	**-**	**201111001**
JATIN VASANT CHARNIA	**-**	**201111028**

Guide
Dr. Ulka Shirole

ACPCE
Where knowledge is second nature

Department of Computer Science and Engineering (Internet of Things and Cyber Security Including Block Chain Technology)
A. C. Patil College of Engineering, Kharghar, Navi Mumbai University of Mumbai
2022-2023

Jawahar Education Society's

A. C. Patil College of Engineering, Kharghar

CERTIFICATE

This is to certify that the Project entitled

"Medical Records using Blockchain"

is a bonafide work of

submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of Third Year Engineering in Computer Science and Engineering (Internet of Things and Cyber Security Including Block Chain Technology)

(Dr. Ulka Shirole)	**Dr. Ulka Shirole**
Guide	Head of Department

Dr.V.N.Pawar

Principal

Mini Project Approval

This Mini Project entitled “**Medical Records using Blockchain Technology”** by **Shewag Nadar (20), Nikhil Bourasi (03), Mansi Pise (24), Jatin Charania (04)** for the course Mini Project– 2 B is approved for the degree of **Third Year Engineering** in **Computer Science and Engineering (Internet of Things and Cyber Security Including Block Chain Technology)**

Examiners

1……………………………………..
(Internal Examiner Name & Sign)

2……………………………………..
(External Examiner name & Sign)

Date: /04/2023

Place:

Abstract

Our project deals with the medical records for storing information of the patient which consist of the medical reports. Medical records are entirely controlled by Hospitals instead of patients, which complicates seeking medical advices from different hospitals.

In the existing system of storing details of the patients are very dependent on the servers of the organization. In the proposed all the information of the patient are stored in the blockchain by using the Metamask and these details are stored in the block chain as a blocks of data. Each block consists of the data which is encrypted data. Medical Health Record system record health-related information on an individual so that it can be consulted by clinicians or staff for patient care. The data is encrypted by the algorithm known as SHA-256 which is used to encrypt all the data of the patients into a single line 256 bit encrypted text which will be stored in the block at etherscan. These records for not only useful for the consultation but also for creation of historic family health information tree that keeps track of genetic health issues and diseases it can also be used for any health service with the authorization from both the patient and medical organization

Contents

List of Abbreviations:

HR - Health Records
EHR - Electronic Health Records
RAM - Random Access Memory

List of Figures:

Chapter 1

Introduction

1.1 Introduction

The objective of this project is to provide the application which is user friendly and cost effective. The major advantage of this project is security. A securable system is more important to be reliable. Our project provide a convenient health record storage service, which promotes traditional patient medical records on paper to be electronically accessible on the web.

This system was designed to allow patients to possess the control of generating, managing and sharing HRs with respective Doctors. Moreover, provided that the healthcare researcher and providers of such service access these HRs across-the aboard, the transition program of healthcare solution is expected to be achieved. However, in the current situation, patients scatter their HRs across the different areas. During life events, causing the HRs to move from one service provider database to another.

Therefore, the patient may lose control of the existing healthcare data, while the service provider usually maintains the primary stewardship. Patient access permissions to HRs are very limited, and patients are typically unable to easily share these data with researchers or providers. Blockchain is a decentralized database whose data block is connected chronologically.

In the healthcare industry, there are many different parties who need to collaboratively manage personal HRs blockchain (in a model of consortium blockchain), such as medical specialists, hospitals, insurance departments, etc. Health Record Systems are proprietary that is centralized by design.
This means that, there's a single supplier that controls the code base, database and the

system outputs and supplies the monitoring tools at the same time.

It is difficult for centralized systems to gain trust from patients, doctors and hospital management. Open source, independently verifiable systems solves this issue. our project deals with the Health Records for storing information of the patient which consist of the medical reports. Health Records (HRs) are entirely controlled by Hospitals instead of patients, which complicates seeking medical advices from different hospitals.

These details are stored using the blockchain technology .In the existing system of storing details of the patients are very dependent on the servers of the organization. In the proposed all the information of the patient are stored in the blockchain by using the Metamask and these details are stored in the block chain as a blocks of data. Each block consist of the data which is encrypted data.

Health Record(HR) systems record health-related information on an individual so that it can be consulted by clinicians or staff for patient care. The data is encrypted by the algorithm known as SHA-256 which is used to encrypt all the data of the patients into a single line 256 bit encrypted text which will be stored in the block at Etherscan.

1.2 Motivation

There are several potential motivations for a project focused on using blockchain technology to store and manage medical records.

- Security and privacy: One of the main benefits of using blockchain technology for medical records is that it can provide a high degree of security and privacy. By using cryptographic techniques and distributed storage, blockchain can ensure that medical records are protected from unauthorized access, tampering, or deletion. This is particularly important for sensitive medical data that could be used for nefarious purposes if it fell into the wrong hands.

- Interoperability: Another potential benefit of using blockchain for medical records is that it could facilitate greater interoperability between different healthcare providers and systems. By using a shared, decentralized database, blockchain could allow different providers to access and update patient records in a secure and standardized way, without the need for complex data-sharing agreements or intermediaries.

- Patient empowerment: By giving patients greater control over their own medical records and allowing them to grant or revoke access to different doctors, blockchain could also empower patients to take a more active role in their own healthcare. This could help to improve patient outcomes and satisfaction, as well as reduce administrative overhead for healthcare providers.

- Research and analytics: Finally, using blockchain to store medical records could also facilitate greater research and analytics capabilities. By providing a standardized, secure, and verifiable database of medical data, blockchain could help researchers to gain new insights into disease patterns, treatment effectiveness, and other important healthcare topics. This could ultimately lead to better healthcare outcomes for patients.

1.3 Problem Statement & Objectives

In the proposed future system, the patient should have right to access his EHRs for managing and sharing them independently. The patient can be access his medical report directly and can use the digitalized report with anyone. By storing the data in the blockchain the user's data is encrypted and stored as blocks in the etherscan. The user stores data by two way authentication process such as getting secret key generated by the Metamask. Electronic Health Record Systems are proprietary that is centralized by design. This means that, there's a single supplier that controls the code base, database and the system outputs and supplies the monitoring tools at the same time. It is difficult for centralized systems to gain trust from patients and doctors and hospital management. Open source, independently verifiable systems solves this issue. This system was designed to allow patients to possess the control of generating, managing and sharing EHRs with family, friends, healthcare providers and other authorized data consumers. Moreover, provided that the healthcare researcher and providers of such service access these EHRs across-the aboard, the transition program of healthcare solution is expected to be achieved.

A blockchain is managed by a network of computers where there is no single computer is responsible for maintaining or storing the data, and any computers can enter or leave this network at any time Using Blockchain for records can make the whole process End to End verifiable and transparent. The stored data will be transactions, from which we can create a blockchain that will keep track of the database of the patient records. Using this approach, all the patients can make use of the records by themselves, and because of the blockchain they can use these records without any permission request from the organization directly by using the secret key given to them.

A decentralized system is a distributed network where no party has the full control over the data and the operations, but the decisions are made collectively through a consensus process. The parties forming the network are called nodes and communicate through message passing. Generally speaking, by sharing and replicating the information, the network provides availability and robustness especially in case of extensive failures. Moreover, Peer-to-peer systems (P2P) can even provide data ownership as the private information can be stored and requested only to the proprietary node. However, reaching consensus while preserving anonymity, security, and correctness despite failures has been a challenging problem studied in the literature. The introduction of blockchain made possible to achieve it while preserving anonymity and providing security and traceability.

Chapter 2

Literature Survey

2.1 Survey of Existing System

Starting from the concepts in, Yue et al. presented the architecture of so-called data gateway application for healthcare data based on the blockchain. They claim to be the first to propose a system based on the distributed ledger technology and address requirements like EHR sharing and patient control over the data. The architecture expects a private blockchain to run on the cloud, but neither they specify how it should be implemented nor provide performance tests. The firsts to introduce a fully functional prototype, applying blockchain technology to EHRs are Azaria, Ekblaw et al.

They propose a system called MedRec not only designed to control the access and authenticate the users but also to manage EMRs in a distributed fashion with the aim to address problems like health data fragmentation, slow access, system interoperability, patient agency and improved data quality and quantity for medical research. They attempt to achieve this by describing a system with a modular design meant for integration. In fact, for scalability issues and to facilitate the adoption, the actual medical record is not stored on the blockchain but is kept off-chain on the hospital, provider's relational database. The blockchain holds metadata and references to the EHR location. More precisely, a smart contract manages the interaction between actors and data and defines access rules and pointers to this data. The pointer is a tuple including a query string that shall be executed on the provider's database as well as the location (host port and credentials) where to access the EHR.

The prototype is developed using the Ethereum public blockchain: the access control is based on the user’s public keys that are Ethereum addresses and the stakeholders participate in the network as “miners” (they run a node). It implies that every party (patient included) must have a blockchain node to interact with it. The main drawback of this implementation is that every actor in the system must have a full copy of the data. Another disadvantage is the poor scalability caused by the consensus protocol. Even though the authors do not mention this possible limit, it is possible to set the upper bound to 60 transactions per second

Chapter 3

Proposed System

3.1 Introduction

An electronic health record (EHR) is the digital version of the patient's health record which includes highly sensitive private information on history, diagnosis, and treatment. Other stored EHR data typically include appointments, billing and accounts, and laboratory tests. Indiana University, the Mayo Clinic, and Vanderbilt University were among the early academic EHR pioneers. In the 1970s, the federal government developed the country's largest EHR system, VistA, for Veterans Affairs Health Care. By 1992, hardware had become cheaper and more powerful, and advancing computer technologies advanced the popularity of EHR. In 2004, EHR was incorporated into the Health Information Technology for Economic and Clinical Health Act (HITECH), reflecting the perceived need to convert medical records to EHR. New incentives and resources were available from the American Recovery and Reinvestment Act (ARRA or "ObamaCare") beginning in 2009. The proportion of American hospitals using EHR systems had reached 95% by 2017. EHR reduces the inefficiency, insecurity, disorganization, data redundancy, and duplicate records problems associated with traditional paper-based medical records. EHR is now regarded as an important part of the healthcare industry, though interoperability and privacy issues remain.

Interoperability may be defined as the degree to which computer software and/or systems share, interdigitate, and/or make use of data between programs and devices made by different manufacturers and/or engineers. For EHR, interoperability refers to the patients' information sharing among different healthcare software systems.

Privacy represents a second unresolved issue within EHR system deployment. Data privacy and security are increasingly important, given the ease with which digital data can be shared or stolen.

Blockchain is database storage using encrypted blocks of data organized in chains for access as a distributed ledger protocol. This means that all parties (nodes) participating in, contributing to, or accessing data in the blockchain decide together which blocks are validly accessed by different users through the use of consensus algorithms.

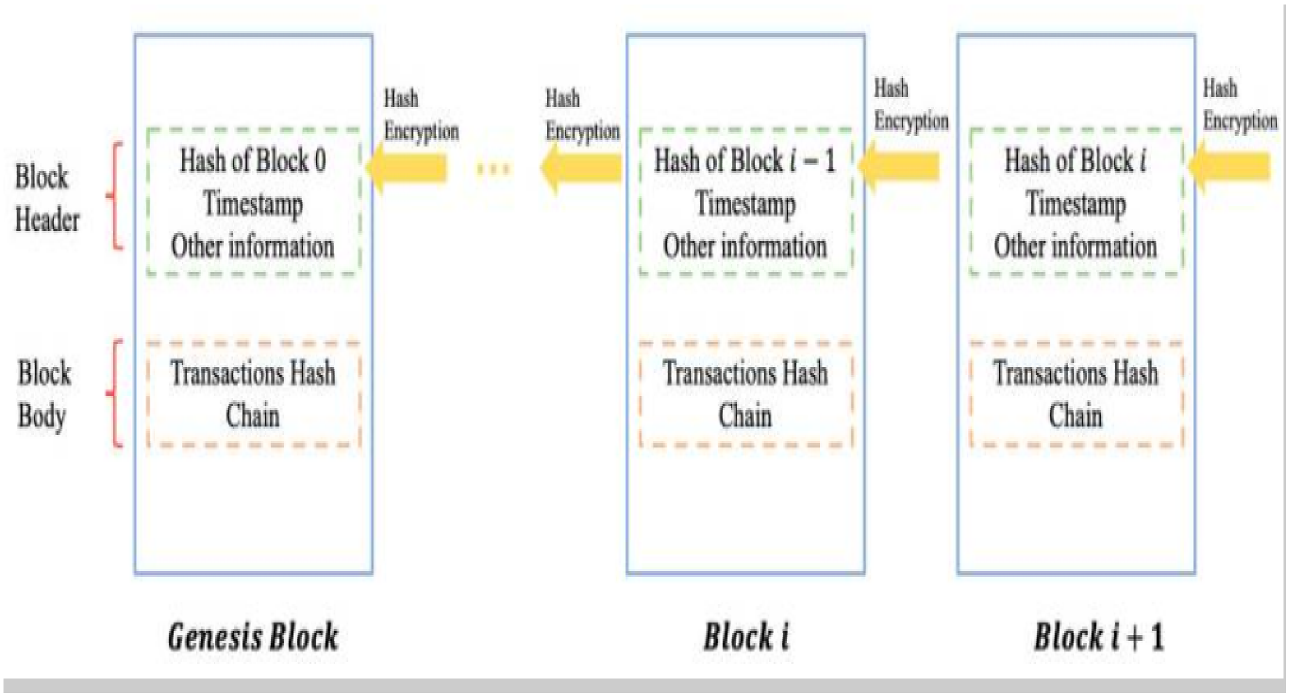


Fig 3.1 Blockchain Process

3.2 Architecture/ Framework

MODULES USED:

Modules present in the proposed system are:

• Ethereum

Ethereum is smart contract platform that is inspired by block chain technology. Its elemental unit is called ether. Ether, similarly to bitcoin is divisible up to 10-18, its smallest subunit is called wei. Due to the fee-by-computation18 policy, Ether (abbr. ETH) is sometimes referred as the fuel of Ethereum. The intention of Ethereum is to merge together enhanced scripting possibilities, meta protocol and time stamped database to allow development of an arbitrary application. The key difference from other block chain protocols is built-in programming language, various types of accounts and unlimited variation of application that can be built on top of it.

• Smart Contract

A smart contract is a self-executing contract that automatically enforces the terms of an agreement between two or more parties. It is a computer program that runs on a blockchain, a decentralized and distributed ledger technology that allows for secure and transparent transactions without the need for intermediaries.

• Truffle Framework

Truffle is a popular development framework for Ethereum smart contracts. It provides a suite of tools that simplifies the process of building, testing, and deploying smart contracts on the Ethereum blockchain.

• **Npm**

npm (short for "Node Package Manager") is a package manager for the JavaScript programming language. It is the default package manager for the Node.js runtime environment and is used to manage packages and dependencies for JavaScript projects.

With npm, developers can easily install, share, and manage packages of code that are available on the npm registry. These packages can include libraries, frameworks, and tools that can be used to build web applications, server-side applications, and other JavaScript-based projects.

• **Reactjs**

React is based on a component-based architecture, which means that UI elements are broken down into reusable and modular components. These components can be easily combined to create complex user interfaces. React also uses a "virtual DOM" approach, which allows it to efficiently update only the parts of the UI that need to be changed, rather than re-rendering the entire UI.

• **Web3js**

Web3.js is a JavaScript library that allows developers to interact with the Ethereum blockchain. It is a popular tool for building decentralized applications (dApps) and for integrating Ethereum into existing web applications.

• **Lite-server**

lite-server is a lightweight development server for web applications. It is designed to be simple and easy to use, while also providing powerful features for developing and testing web applications.

3.3 Process Design

The process design for electronic health records using blockchain technology involves several steps, including:

- Identification of stakeholders:
  - Identify the stakeholders involved in the EHR system, including patients, healthcare providers, hospitals, insurers, and regulatory bodies.
  - Define the roles and responsibilities of each stakeholder.
- Definition of requirements:
  - Define the requirements for the EHR system, including data security, privacy, interoperability, and accessibility.
  - Determine the data elements to be included in the EHR system.
- Design of the blockchain network:
  - Select the appropriate blockchain platform, such as Ethereum or Hyperledger.
  - Design the network architecture and consensus mechanism.
  - Define the data schema and storage structure.
- Development of smart contracts:
  - Develop the smart contracts that will govern the EHR system, including access control, data storage, and data sharing.
  - Define the workflows and business rules for the EHR system.
- Integration with existing systems:
  - Integrate the EHR system with existing healthcare systems, such as EMRs and billing systems.
  - Define the data exchange mechanisms and protocols.
- Testing and deployment:
  - Test the EHR system to ensure that it meets the requirements and is secure and reliable.
  - Deploy the EHR system to production.

- ❖ Ongoing maintenance and updates:
  - ➢ Maintain and update the EHR system regularly to ensure that it remains secure, reliable, and up-to-date with the latest technology and regulatory requirements.
  - ➢ Monitor the system for security breaches and take appropriate action if necessary.

Overall, the process design for electronic health records using blockchain technology involves careful planning, design, and implementation to ensure that the system meets the requirements of all stakeholders and provides a secure and reliable platform for managing healthcare data.

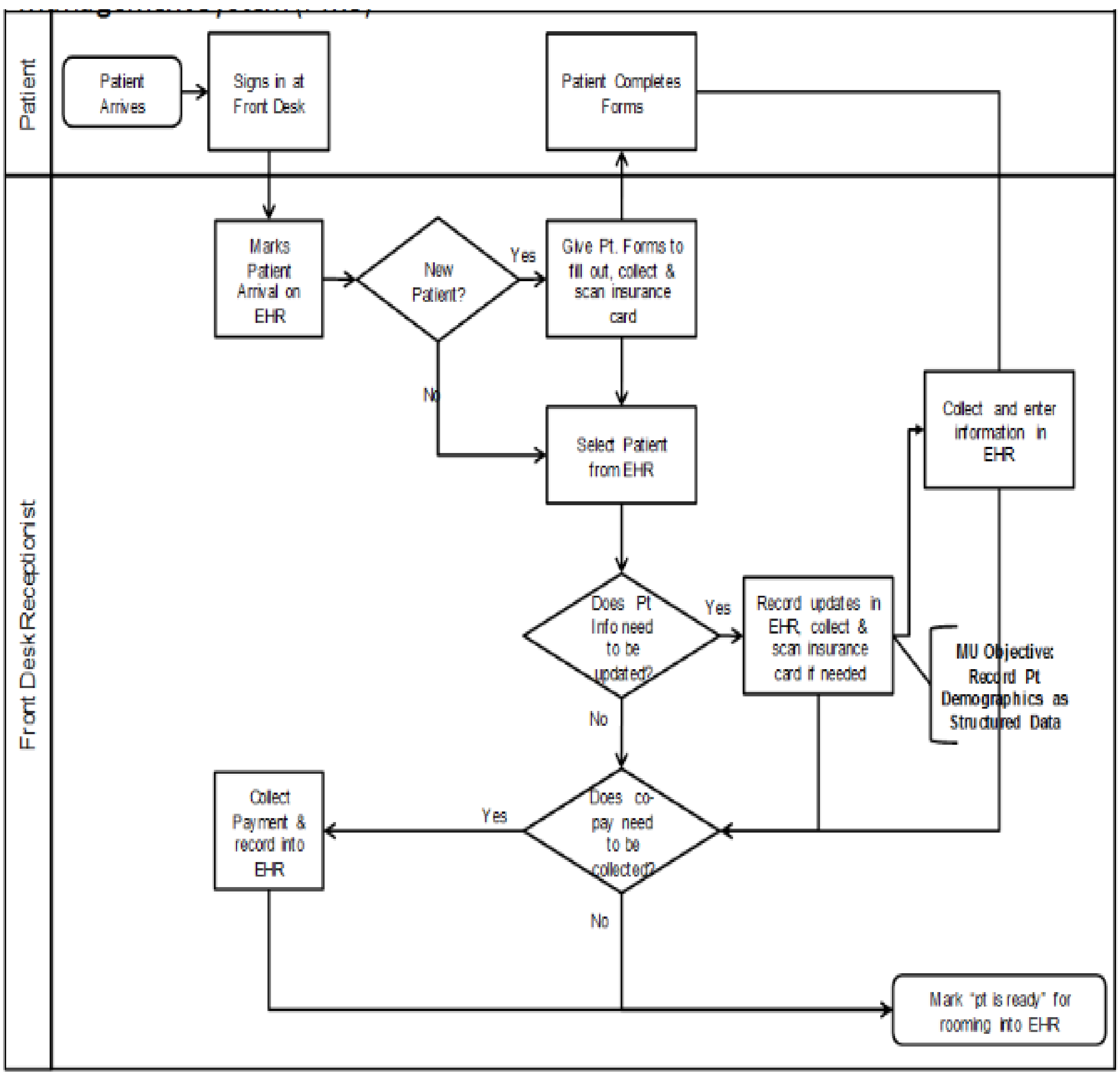

Patient
Front Desk Receptionist
Patient Arrives
Signs in at Front Desk
Patient Completes Forms
Marks Patient Arrival on EHR
New Patient?
Yes
No
Give Pt. Forms to fill out, collect & scan insurance card
Select Patient from EHR
Collect and enter information in EHR
Does Pt Info need to be updated?
Yes
Record updates in EHR, collect & scan insurance card if needed
MU Objective: Record Pt Demographics as Structured Data
No
Collect Payment & record into EHR
Yes
Does co-pay need to be collected?
No
Mark "pt is ready" for rooming into EHR

Fig 3.3 Flowchart For EHR

3.4 Details of Hardware & Software

 SOFTWARE USED:

- HTML

The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for its appearance. HTML elements are the building blocks of HTML pages. .

- CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.CSS is designed to enable the separation of content and presentation, including layout, colors, and fonts.This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.the use of CSS including XHTML, plain XML, SVG, and XUL. CSS is also used in GTK widget toolkit.

- JAVASCRIPT

JavaScript , often abbreviated as JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98% of websites use JavaScript on the client side for webpage behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices. JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard.It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

- SOLIDITY

Solidity is an object-oriented programming language for implementing smart contracts on various blockchain platforms, most notably, Ethereum. Solidity is licensed under GNU General Public License v3.0.Solidity was designed by Gavin Wood and developed by Christian Reitwiessner, Alex Beregszaszi, and several former Ethereum core contributors. Programs in Solidity run on Ethereum Virtual Machine or on compatible virtual machines.Solidity was proposed in August 2014 by Gavin Wood.The language was later developed by the Ethereum project's Solidity team, led by Christian Reitwiessner.

Solidity is the primary language on Ethereum as well as on other private blockchains, such as the enterprise-oriented Hyperledger Fabric blockchain. SWIFT deployed a proof of concept using Solidity running on Hyperledger Fabric.Solidity is a statically typed programming language designed for developing smart contracts that run on the Ethereum Virtual Machine (EVM) or compatible virtual machines.

Solidity uses ECMAScript-like syntax which makes it familiar for existing web developers; however unlike ECMAScript it has static typing and variadic return types. Solidity is different from other EVM-targeting languages such as Serpent and Mutan in some important ways. It supports complex member variables for smart contracts, including arbitrarily hierarchical mappings and structs.

Solidity smart contract support inheritance, including multiple inheritance with C3 linearization. Solidity introduces an application binary interface (ABI) that facilitates multiple type-safe functions within a single smart contract (this was also later supported by Serpent). The Solidity proposal also includes "Natural Language Specification", a documentation system for specifying user-centric descriptions of the ramifications of [Method (computer programming) | method]-calls.

- GANACHE

Ganache is used for setting up a personal Ethereum Blockchain for testing your Solidity contracts. It provides more features when compared to Remix. You will learn about the features when you work out with Ganache. Before you begin using Ganache, you must first download and install the Blockchain on your local machine.The biggest advantages of using Ganache smart contract development would refer to the facility for developing, testing, and deploying your smart contracts and dApp projects in a deterministic and safe environment.

You can access two different variants of Ganache, depending on the type of functionality you need. The Ganache UI is the desktop application that can offer support for Ethereum and Corda development tasks. On the other hand, you have the Ganache-CLI, which is the command-line tool and focuses specifically on Ethereum development. It is also important to note that both versions of Ganache are accessible on Linux, Mac, and Windows.

- IPFS

IPFS is a modular suite of protocols for organizing and transferring data, designed from the ground up with the principles of content addressing and peer-to-peer networking. Because IPFS is open-source, there are multiple implementations of IPFS. While IPFS has more than one use case, its main use case is for publishing data (files, directories, websites, etc.) in a decentralised fashion.

The Interplanetary File System (IPFS) is a distributed file storage protocol that allows computers all over the globe to store and serve files as part of a giant peer-to-peer network. Any computer, anywhere in the world, can download the IPFS software and start hosting and serving files.

The benefits of IPFS are revolutionizing the current technology sector, causing large companies to implement it in their processes. Netflix, Opera or Chrome are some companies that rely on this technology. But what does it really mean that we can safely store data on a distributed network? It means that we can optimize and improve the operation of all the processes that happen on the internet.

- METAMASK

MetaMask is a software cryptocurrency wallet used to interact with the Ethereum blockchain. It allows users to access their Ethereum wallet through a browser extension or mobile app, which can then be used to interact with decentralized applications. MetaMask is developed by ConsenSys Software Inc., a blockchain software company focusing on Ethereum-based tools and infrastructure.

MetaMask allows users to store and manage account keys, broadcast transactions, send and receive Ethereum-based cryptocurrencies and tokens, and securely connect to decentralized applications through a compatible web browser or the mobile app's built-in browser.Websites or other decentralized applications are able to connect, authenticate, and/or integrate other smart contract functionality with a user's MetaMask wallet (and any other similar blockchain wallet browser extensions) via JavaScript code that allows the website to send action prompts, signature requests, or transaction requests to the user through MetaMask as an intermediary.

The application includes an integrated service for exchanging Ethereum tokens by aggregating several decentralized exchanges (DEXs) to find the best exchange rate. This feature, branded as MetaMask Swaps, charges a service fee of 0.875% of the transaction amount.

- REACTJS

React (also known as React.js or ReactJS) is a free and open-source front-end JavaScript library for building user interfaces based on components. It is maintained by Meta (formerly Facebook) and a community of individual developers and companies.React can be used as a base in the development of single-page, mobile, or server-rendered applications with frameworks like Next.js.

However, React is only concerned with the user interface and rendering components to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

React adheres to the declarative programming paradigm (a programming paradigm). Developers design views for each state of an application, and React updates and renders components when data changes. This is in contrast with imperative programming.

- TRUFFLE

Truffle is a world-class development environment, testing framework and asset pipeline for blockchains using the Ethereum Virtual Machine (EVM), aiming to make life as a developer easier.

Truffle is widely considered the most popular tool for blockchain application development with over 1.5 million lifetime downloads. Truffle supports developers across the full lifecycle of their projects, whether they are looking to build on Ethereum, Hyperledger, Quorum, or one of an ever-growing list of other supported platforms.

Paired with Ganache, a personal blockchain, and Drizzle, a front-end dApp development kit, the full Truffle suite of tools promises to be an end-to-end dApp development platform.Truffle is the most popular development tooling for Ethereum programmers. Easily deploy smart contracts and communicate with their underlying state without heavy client side programming. An especially useful library for the testing and iteration of Ethereum smart contracts.

- NODE

Node.js is a cross-platform, open-source server environment that can run on Windows, Linux, Unix, macOS, and more. Node.js is a back-end JavaScript runtime environment, runs on the V8 JavaScript Engine, and executes JavaScript code outside a web browser.Node.js lets developers use JavaScript to write command line tools and for server-side scripting. The ability to run JavaScript code on the server is often used to generate dynamic web page content before the page is sent to the user's web browser.

Consequently, Node.js represents a "JavaScript everywhere" paradigm, unifying web-application development around a single programming language, as opposed to using different languages for the server- versus client-side programming.

Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimize throughput and scalability in web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games).The Node.js distributed development project was previously governed by the Node.js Foundation,and has now merged with the JS Foundation to form the OpenJS Foundation. OpenJS Foundation is facilitated by the Linux Foundation's Collaborative Projects program.

- NPM

npm is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the default package manager for the JavaScript runtime environment Node.js. It consists of a command line client, also called npm, and an online database of public and paid-for private packages, called the npm registry. The registry is accessed via the client, and the available packages can be browsed and searched via the npm website. The package manager and the registry are managed by npm, Inc.

npm is included as a recommended feature in the Node.js installer. npm consists of a command line client that interacts with a remote registry. It allows users to consume and distribute JavaScript modules that are available in the registry. Packages in the registry are in CommonJS format and include a metadata file in JSON format.Over 1.3 million packages are available in the main npm registry.

The registry does not have any vetting process for submission, which means that packages found there can potentially be low quality, insecure, or malicious. Instead, npm relies on user reports to take down packages if they violate policies by being low quality, insecure, or malicious.npm exposes statistics including number of downloads and number of depending packages to assist developers in judging the quality of packages.

In npm version 6, the audit feature was introduced to help developers identify and fix security vulnerabilities in installed packages.The source of security vulnerabilities were taken from reports found on the Node Security Platform (NSP) and has been integrated with npm since npm's acquisition of NSP.

- WEB3JS

web3.js is a collection of libraries that allow you to interact with a local or remote ethereum node using HTTP, IPC or WebSocket. Using this library, you can develop websites or clients that interact with the blockchain. This can be actions like sending Ether from one user to another, checking data from smart contracts, creating smart contracts, among other things. Ethereum nodes provide interfaces to users in order to complete transactions: of which, nodes receive this information through a JSON RPC interface.

This is an encoding format that allows running processes to receive new and verify existing data. Web3.js helps to make the process of running and selecting nodes participating in the Ethereum network simpler and easier to grasp.

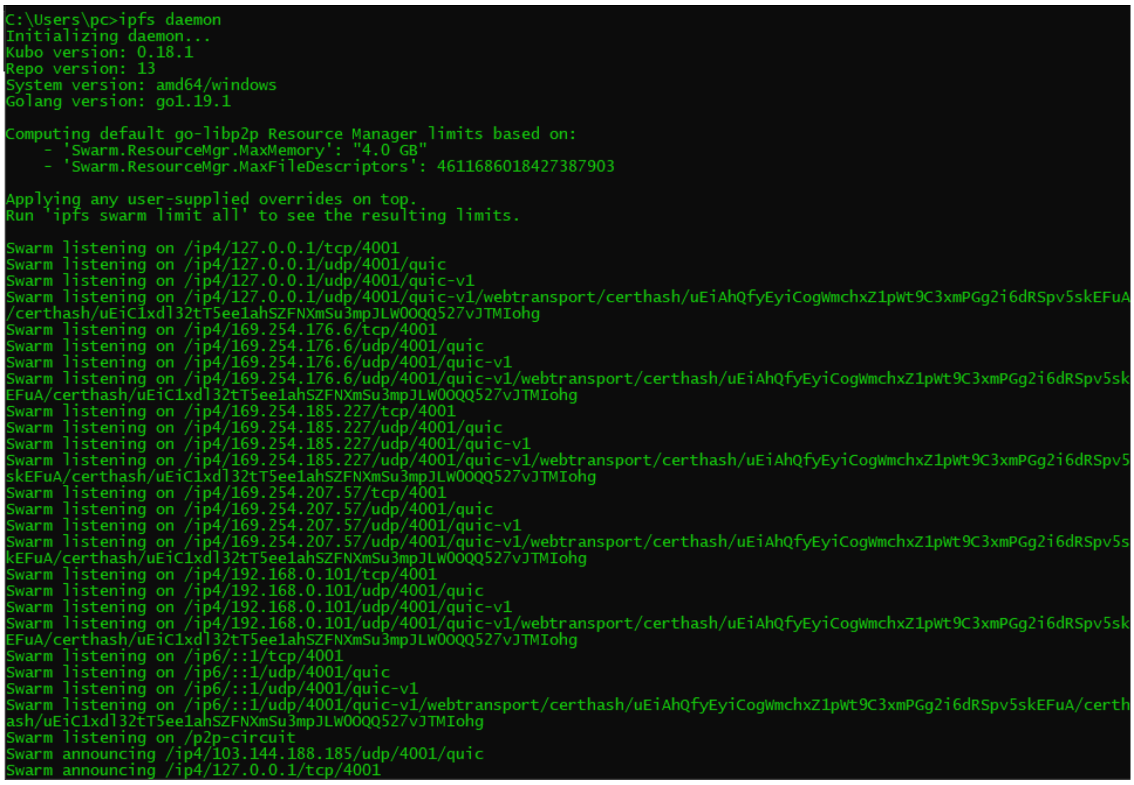
- LITE SERVER

Lite-server is a lightweight development server that serves a web application, opens it in the browser, and refreshes the page when HTML or JavaScript changes are made to the source code. This can help save time during development as the user does not manually have to refresh the page every time a change is made.

Chapter 4

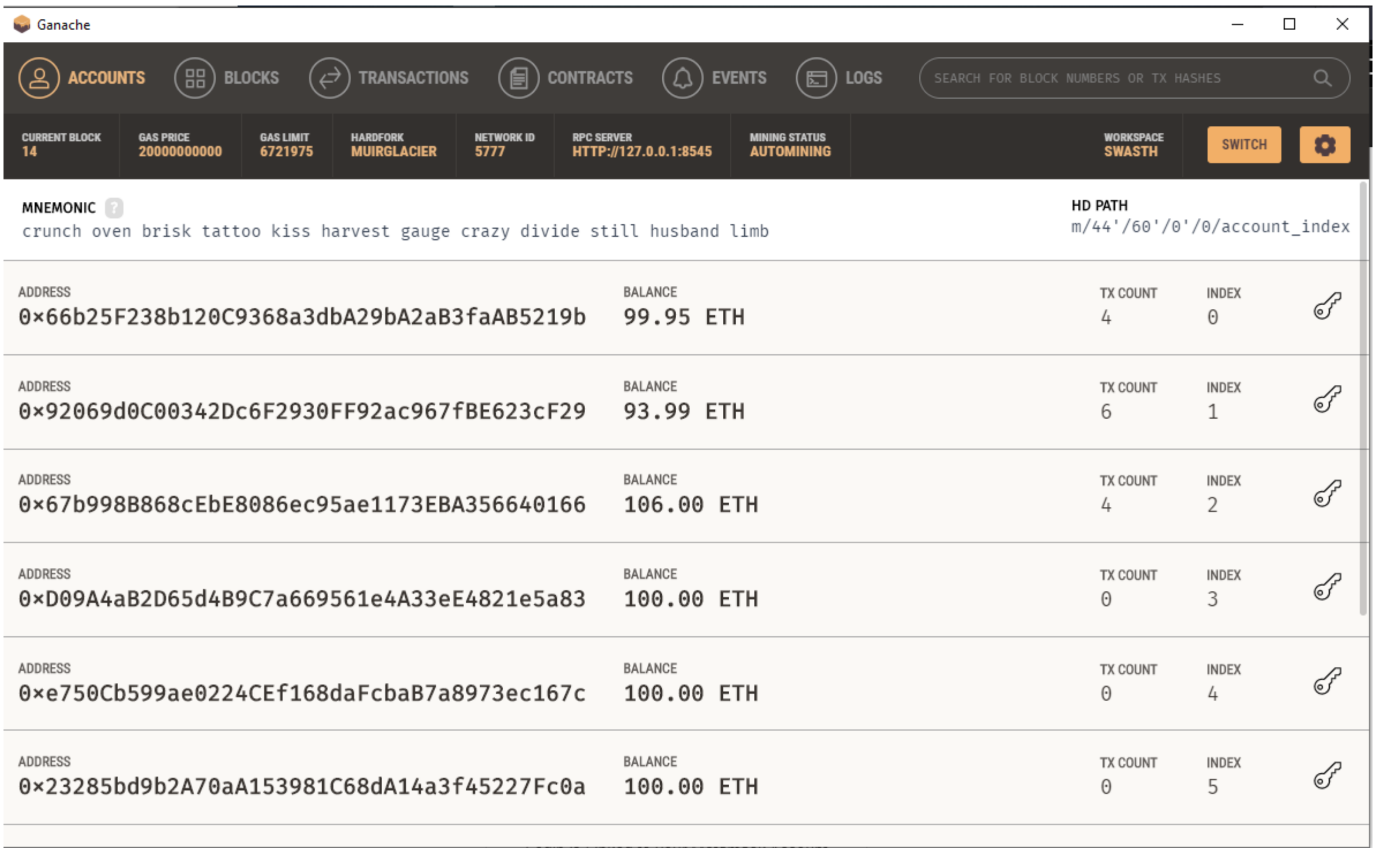
Result and Discussion

EHR is building the future of healthcare on blockchain. There are similar projects but EHR's unique vision stands out. It comes with the specialization of using the blockchain. The blockchain technology makes it easy to monitor population health, identify risk and trends in spread of any issues as it has updated medical report of the patient. This helps to promote effective treatment for the patients throughout the globe. As it is decentralized it in not owned by a single entity, the data is cryptographically stored and they are highly secured. The results of the system were asexpected.



```
C:\Users\pc>ipfs daemon
Initializing daemon...
Kubo version: 0.18.1
Repo version: 13
System version: amd64/windows
Golang version: go1.19.1

Computing default go-libp2p Resource Manager limits based on:
    - 'Swarm.ResourceMgr.MaxMemory': "4.0 GB"
    - 'Swarm.ResourceMgr.MaxFileDescriptors': 4611686018427387903

Applying any user-supplied overrides on top.
Run 'ipfs swarm limit all' to see the resulting limits.

Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/127.0.0.1/udp/4001/quic
Swarm listening on /ip4/127.0.0.1/udp/4001/quic-v1
Swarm listening on /ip4/127.0.0.1/udp/4001/quic-v1/webtransport/certhash/uEiAhQfyEyiCogWmchxZ1pWt9C3xmPGg2i6dRSpv5skEFuA
/certhash/uEiC1xdl32tT5ee1ahSZFNXmSu3mpJLW0OQQ527vJTMIohg
Swarm listening on /ip4/169.254.176.6/tcp/4001
Swarm listening on /ip4/169.254.176.6/udp/4001/quic
Swarm listening on /ip4/169.254.176.6/udp/4001/quic-v1
Swarm listening on /ip4/169.254.176.6/udp/4001/quic-v1/webtransport/certhash/uEiAhQfyEyiCogWmchxZ1pWt9C3xmPGg2i6dRSpv5sk
EFuA/certhash/uEiC1xdl32tT5ee1ahSZFNXmSu3mpJLW0OQQ527vJTMIohg
Swarm listening on /ip4/169.254.185.227/tcp/4001
Swarm listening on /ip4/169.254.185.227/udp/4001/quic
Swarm listening on /ip4/169.254.185.227/udp/4001/quic-v1
Swarm listening on /ip4/169.254.185.227/udp/4001/quic-v1/webtransport/certhash/uEiAhQfyEyiCogWmchxZ1pWt9C3xmPGg2i6dRSpv5
skEFuA/certhash/uEiC1xdl32tT5ee1ahSZFNXmSu3mpJLW0OQQ527vJTMIohg
Swarm listening on /ip4/169.254.207.57/tcp/4001
Swarm listening on /ip4/169.254.207.57/udp/4001/quic
Swarm listening on /ip4/169.254.207.57/udp/4001/quic-v1
Swarm listening on /ip4/169.254.207.57/udp/4001/quic-v1/webtransport/certhash/uEiAhQfyEyiCogWmchxZ1pWt9C3xmPGg2i6dRSpv5s
kEFuA/certhash/uEiC1xdl32tT5ee1ahSZFNXmSu3mpJLW0OQQ527vJTMIohg
Swarm listening on /ip4/192.168.0.101/tcp/4001
Swarm listening on /ip4/192.168.0.101/udp/4001/quic
Swarm listening on /ip4/192.168.0.101/udp/4001/quic-v1
Swarm listening on /ip4/192.168.0.101/udp/4001/quic-v1/webtransport/certhash/uEiAhQfyEyiCogWmchxZ1pWt9C3xmPGg2i6dRSpv5sk
EFuA/certhash/uEiC1xdl32tT5ee1ahSZFNXmSu3mpJLW0OQQ527vJTMIohg
Swarm listening on /ip6/::1/tcp/4001
Swarm listening on /ip6/::1/udp/4001/quic
Swarm listening on /ip6/::1/udp/4001/quic-v1
Swarm listening on /ip6/::1/udp/4001/quic-v1/webtransport/certhash/uEiAhQfyEyiCogWmchxZ1pWt9C3xmPGg2i6dRSpv5skEFuA/certh
ash/uEiC1xdl32tT5ee1ahSZFNXmSu3mpJLW0OQQ527vJTMIohg
Swarm listening on /p2p-circuit
Swarm announcing /ip4/103.144.188.185/udp/4001/quic
Swarm announcing /ip4/127.0.0.1/tcp/4001
```

Fig 4.1 - IPFS daemon



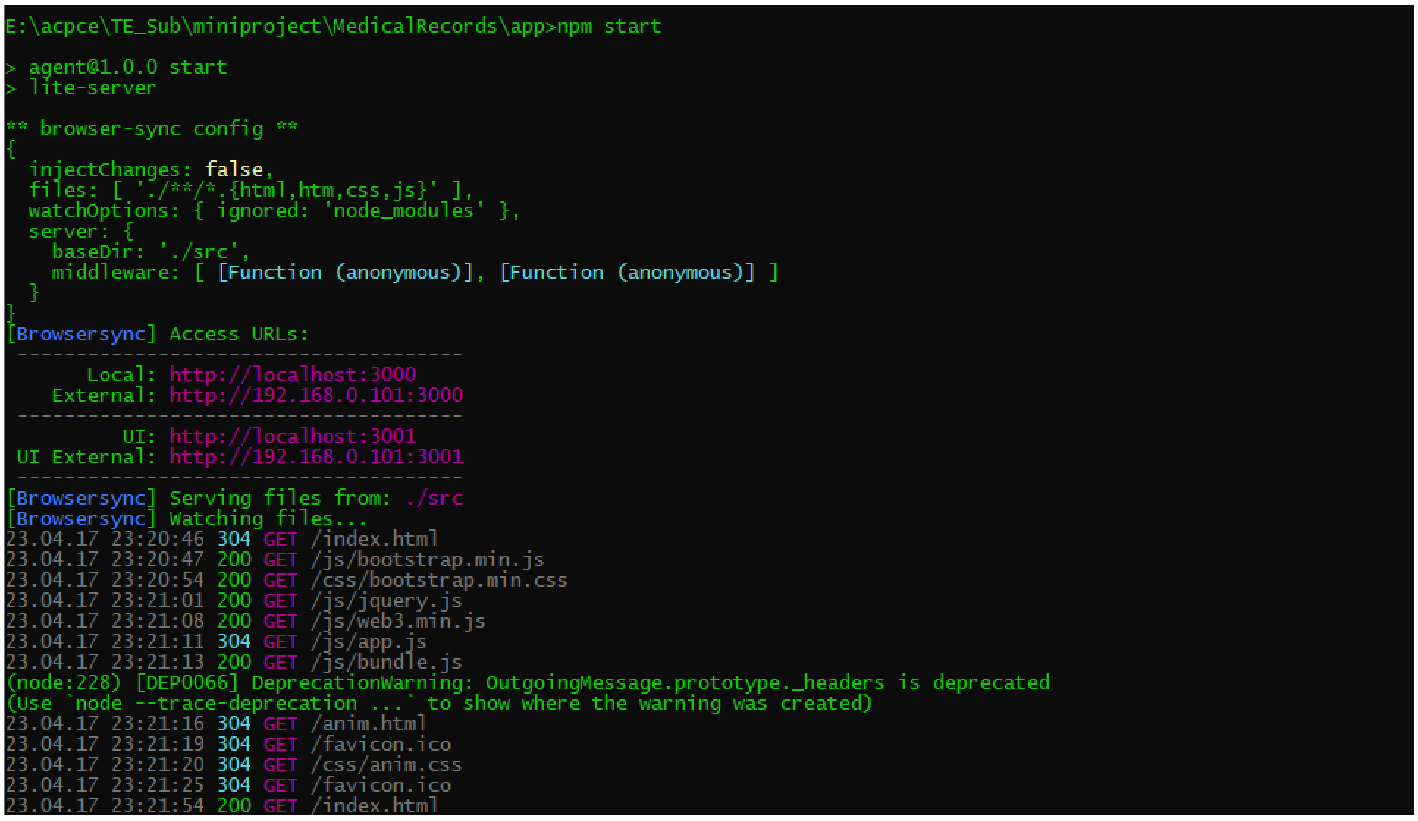
CURRENT BLOCK 14 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:8545 | MINING STATUS AUTOMINING | WORKSPACE SWASTH

MNEMONIC
crunch oven brisk tattoo kiss harvest gauge crazy divide still husband limb

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

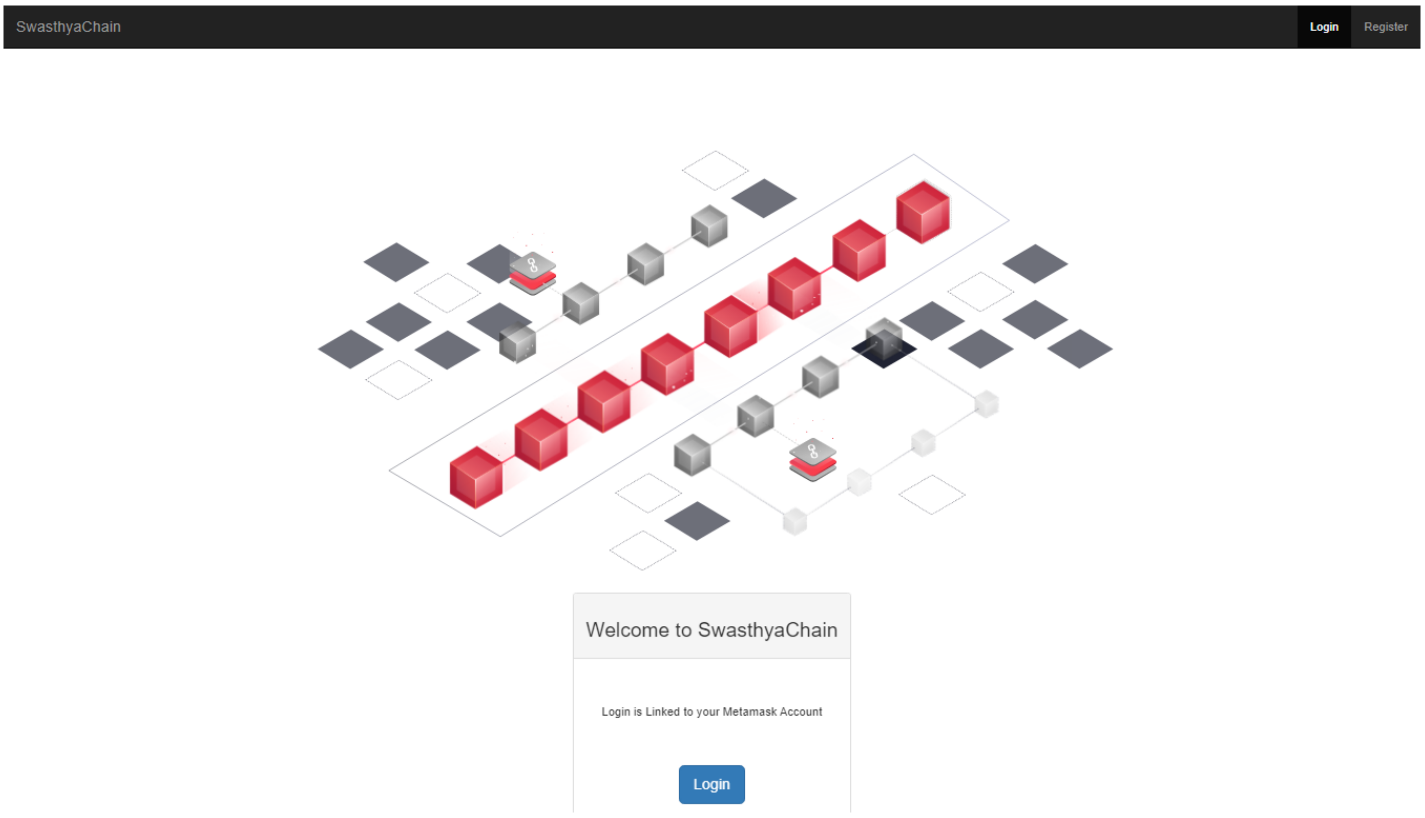
ADDRESS	BALANCE	TX COUNT	INDEX
0×66b25F238b120C9368a3dbA29bA2aB3faAB5219b	99.95 ETH	4	0
0×92069d0C00342Dc6F2930FF92ac967fBE623cF29	93.99 ETH	6	1
0×67b998B868cEbE8086ec95ae1173EBA356640166	106.00 ETH	4	2
0×D09A4aB2D65d4B9C7a669561e4A33eE4821e5a83	100.00 ETH	0	3
0×e750Cb599ae0224CEf168daFcbaB7a8973ec167c	100.00 ETH	0	4
0×23285bd9b2A70aA153981C68dA14a3f45227Fc0a	100.00 ETH	0	5

Fig 4.2 - Ganache Accounts



```
E:\acpce\TE_Sub\miniproject\MedicalRecords\app>npm start

> agent@1.0.0 start
> lite-server

** browser-sync config **
{
  injectChanges: false,
  files: [ './**/*.{html,htm,css,js}' ],
  watchOptions: { ignored: 'node_modules' },
  server: {
    baseDir: './src',
    middleware: [ [Function (anonymous)], [Function (anonymous)] ]
  }
}
[Browsersync] Access URLs:
 --------------------------------------
       Local: http://localhost:3000
    External: http://192.168.0.101:3000
 --------------------------------------
          UI: http://localhost:3001
 UI External: http://192.168.0.101:3001
 --------------------------------------
[Browsersync] Serving files from: ./src
[Browsersync] Watching files...
23.04.17 23:20:46 304 GET /index.html
23.04.17 23:20:47 200 GET /js/bootstrap.min.js
23.04.17 23:20:54 200 GET /css/bootstrap.min.css
23.04.17 23:21:01 200 GET /js/jquery.js
23.04.17 23:21:08 200 GET /js/web3.min.js
23.04.17 23:21:11 304 GET /js/app.js
23.04.17 23:21:13 200 GET /js/bundle.js
(node:228) [DEP0066] DeprecationWarning: OutgoingMessage.prototype._headers is deprecated
(Use `node --trace-deprecation ...` to show where the warning was created)
23.04.17 23:21:16 304 GET /anim.html
23.04.17 23:21:19 304 GET /favicon.ico
23.04.17 23:21:20 304 GET /css/anim.css
23.04.17 23:21:25 304 GET /favicon.ico
23.04.17 23:21:54 200 GET /index.html
```

Fig 4.3 NPM start


SwasthyaChain

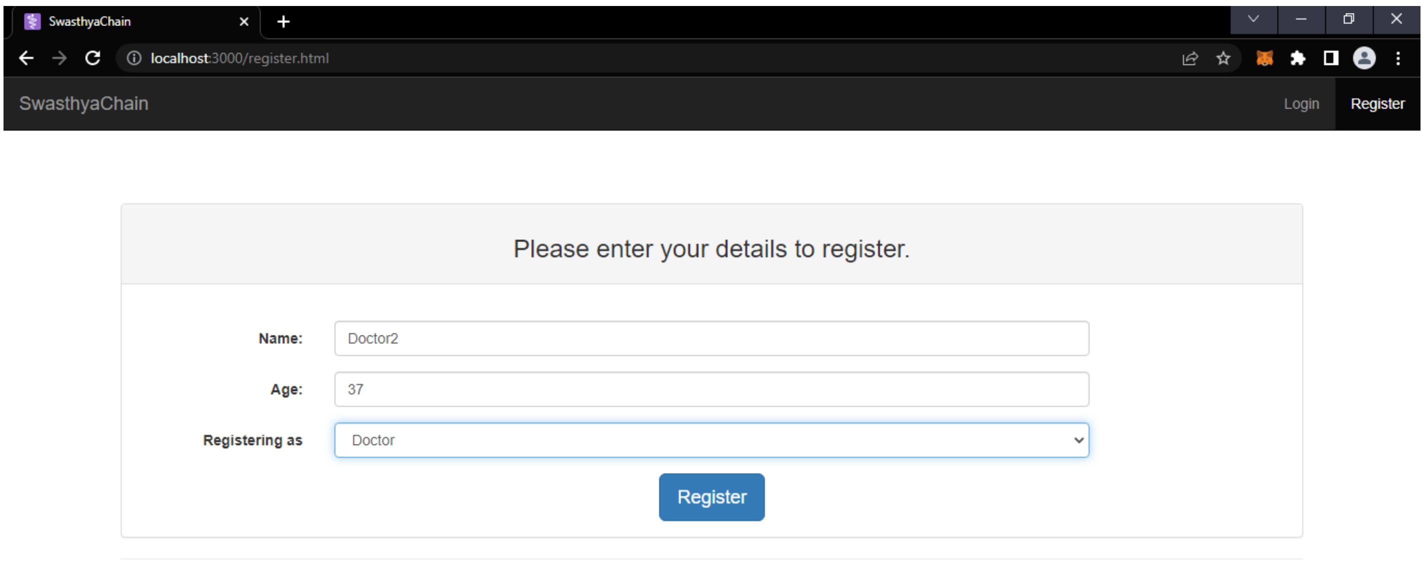
Login Register

Welcome to SwasthyaChain

Login is Linked to your Metamask Account

Login

Fig 4.4- Home Page



SwasthyaChain

Login Register

Please enter your details to register.

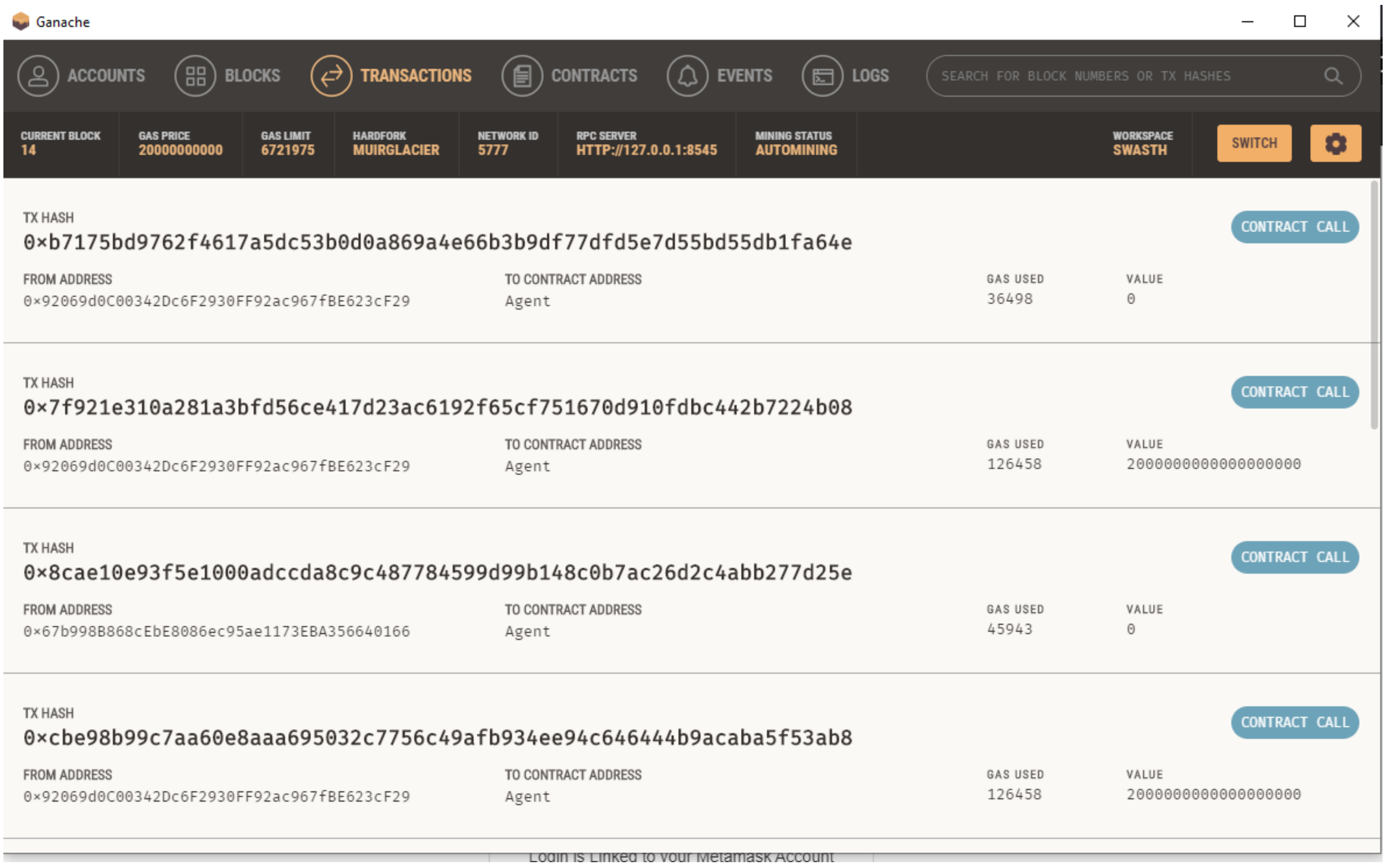
Name: Doctor2

Age: 37

Registering as Doctor

Register

Fig 4.5- Register

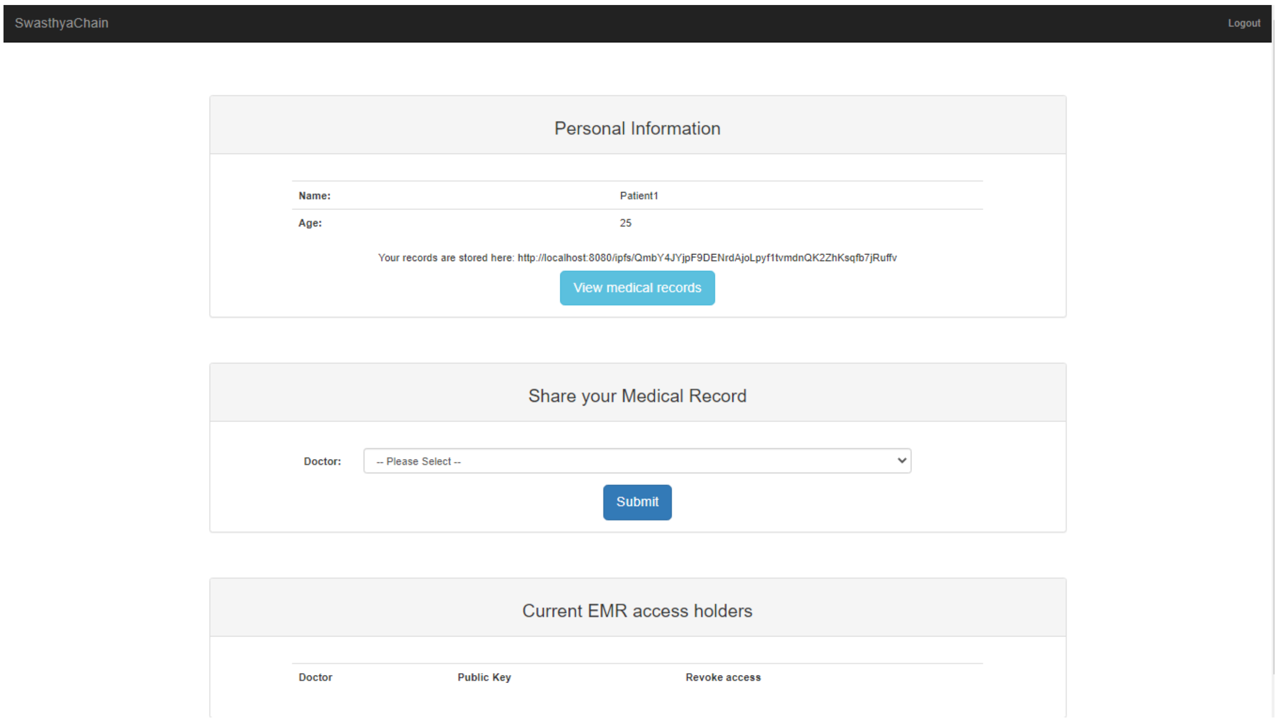


CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS	WORKSPACE
14	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:8545	AUTOMINING	SWASTH

```
TX HASH
0×b7175bd9762f4617a5dc53b0d0a869a4e66b3b9df77dfd5e7d55bd55db1fa64e               CONTRACT CALL
FROM ADDRESS                                 TO CONTRACT ADDRESS     GAS USED    VALUE
0×92069d0C00342Dc6F2930FF92ac967fBE623cF29   Agent                   36498       0

TX HASH
0×7f921e310a281a3bfd56ce417d23ac6192f65cf751670d910fdbc442b7224b08               CONTRACT CALL
FROM ADDRESS                                 TO CONTRACT ADDRESS     GAS USED    VALUE
0×92069d0C00342Dc6F2930FF92ac967fBE623cF29   Agent                   126458      2000000000000000000

TX HASH
0×8cae10e93f5e1000adccda8c9c487784599d99b148c0b7ac26d2c4abb277d25e               CONTRACT CALL
FROM ADDRESS                                 TO CONTRACT ADDRESS     GAS USED    VALUE
0×67b998B868cEbE8086ec95ae1173EBA356640166   Agent                   45943       0

TX HASH
0×cbe98b99c7aa60e8aaa695032c7756c49afb934ee94c646444b9acaba5f53ab8               CONTRACT CALL
FROM ADDRESS                                 TO CONTRACT ADDRESS     GAS USED    VALUE
0×92069d0C00342Dc6F2930FF92ac967fBE623cF29   Agent                   126458      2000000000000000000
```

Login is Linked to your Metamask Account

Fig 4.6- Ganache transaction



Personal Information

Name:	Patient1
Age:	25

Your records are stored here: http://localhost:8080/ipfs/QmbY4JYjpF9DENrdAjoLpyf1tvmdnQK2ZhKsqfb7jRuffv

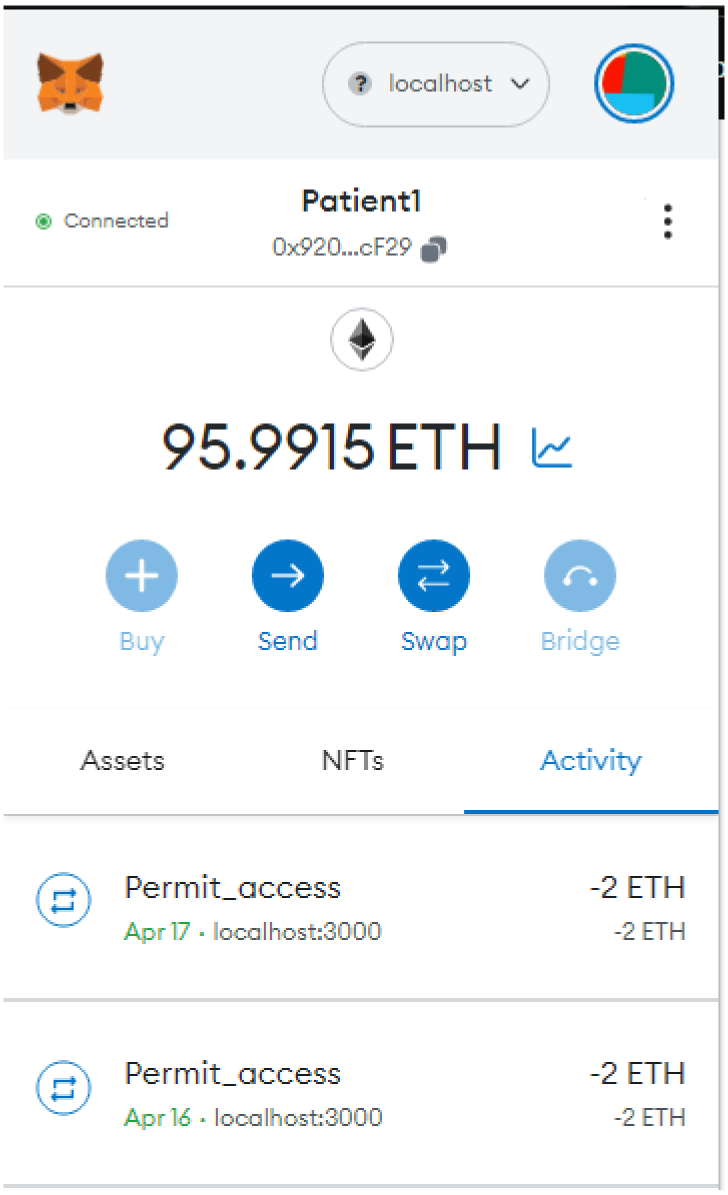
Share your Medical Record

Doctor: -- Please Select --

Current EMR access holders

Doctor	Public Key	Revoke access

Fig 4.7 Patient



localhost

Connected

Patient1

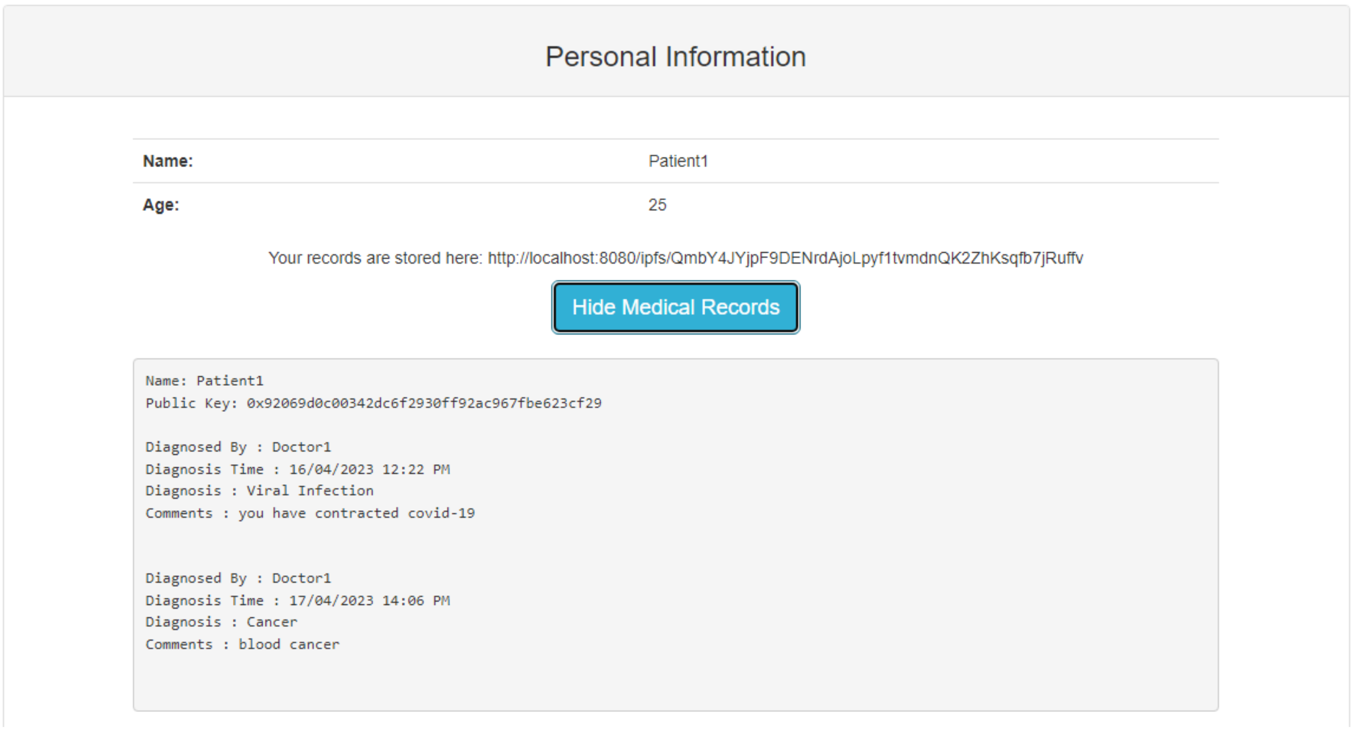
0x920...cF29

95.9915 ETH

Permit_access -2 ETH
Apr 17 · localhost:3000 -2 ETH

Permit_access -2 ETH
Apr 16 · localhost:3000 -2 ETH

Fig 4.8 - Patients Wallet

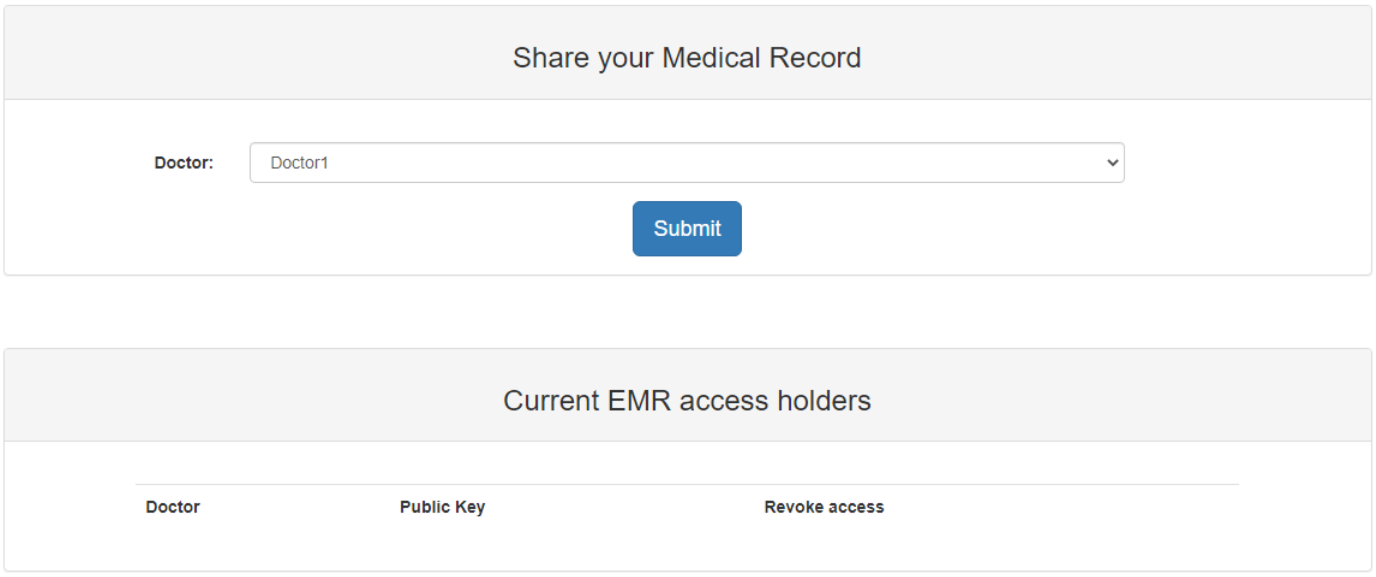

Personal Information

Name:	Patient1
Age:	25

Your records are stored here: http://localhost:8080/ipfs/QmbY4JYjpF9DENrdAjoLpyf1tvmdnQK2ZhKsqfb7jRuffv

```
Name: Patient1
Public Key: 0x92069d0c00342dc6f2930ff92ac967fbe623cf29

Diagnosed By : Doctor1
Diagnosis Time : 16/04/2023 12:22 PM
Diagnosis : Viral Infection
Comments : you have contracted covid-19


Diagnosed By : Doctor1
Diagnosis Time : 17/04/2023 14:06 PM
Diagnosis : Cancer
Comments : blood cancer
```

Fig 4.9 - Patient View Diagnosis

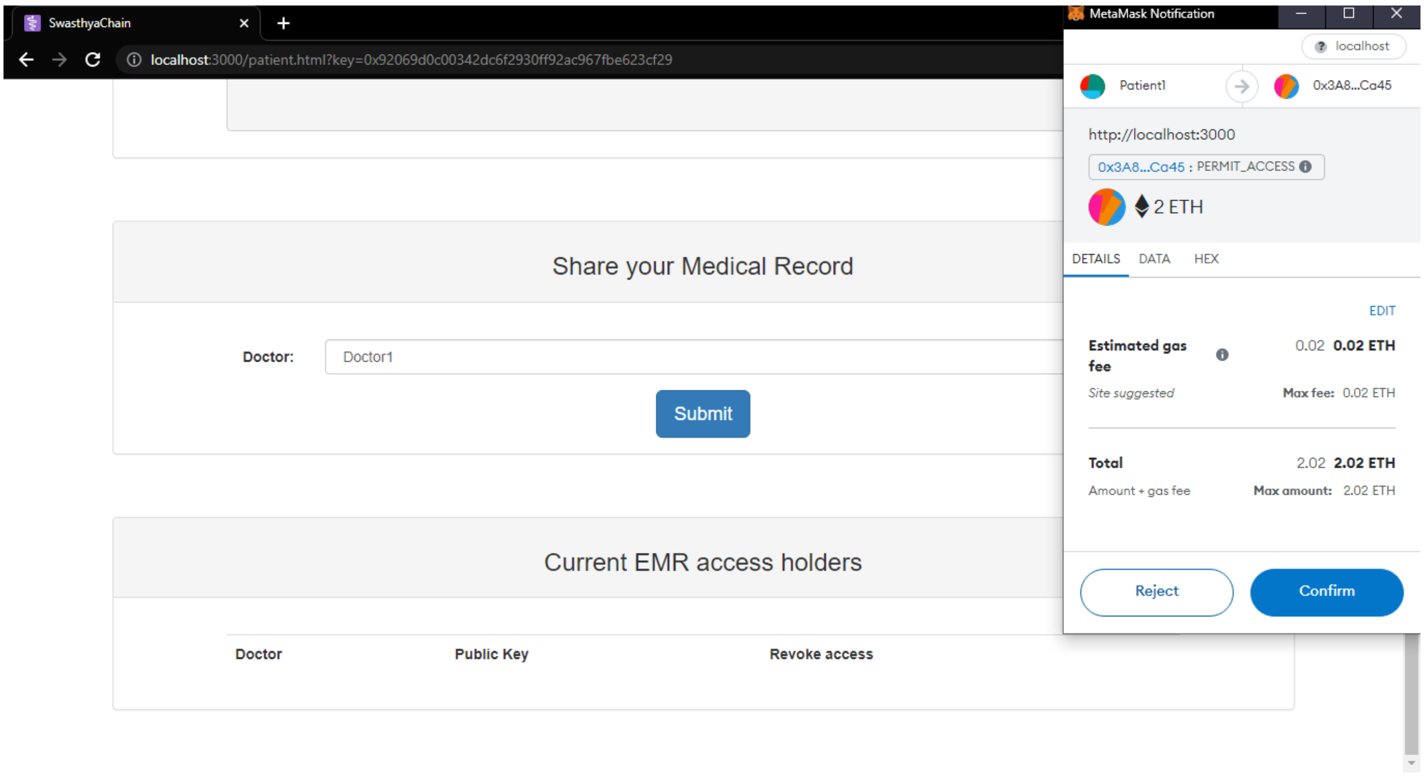

Share your Medical Record

Doctor: Doctor1

Current EMR access holders

Doctor	Public Key	Revoke access

Fig 4.10- Patient Share access


SwasthyaChain

localhost:3000/patient.html?key=0x92069d0c00342dc6f2930ff92ac967fbe623cf29

Share your Medical Record

Doctor: Doctor1

Submit

Current EMR access holders

Doctor	Public Key	Revoke access

MetaMask Notification

localhost

Patient1 → 0x3A8...Ca45

http://localhost:3000

0x3A8...Ca45 : PERMIT_ACCESS

2 ETH

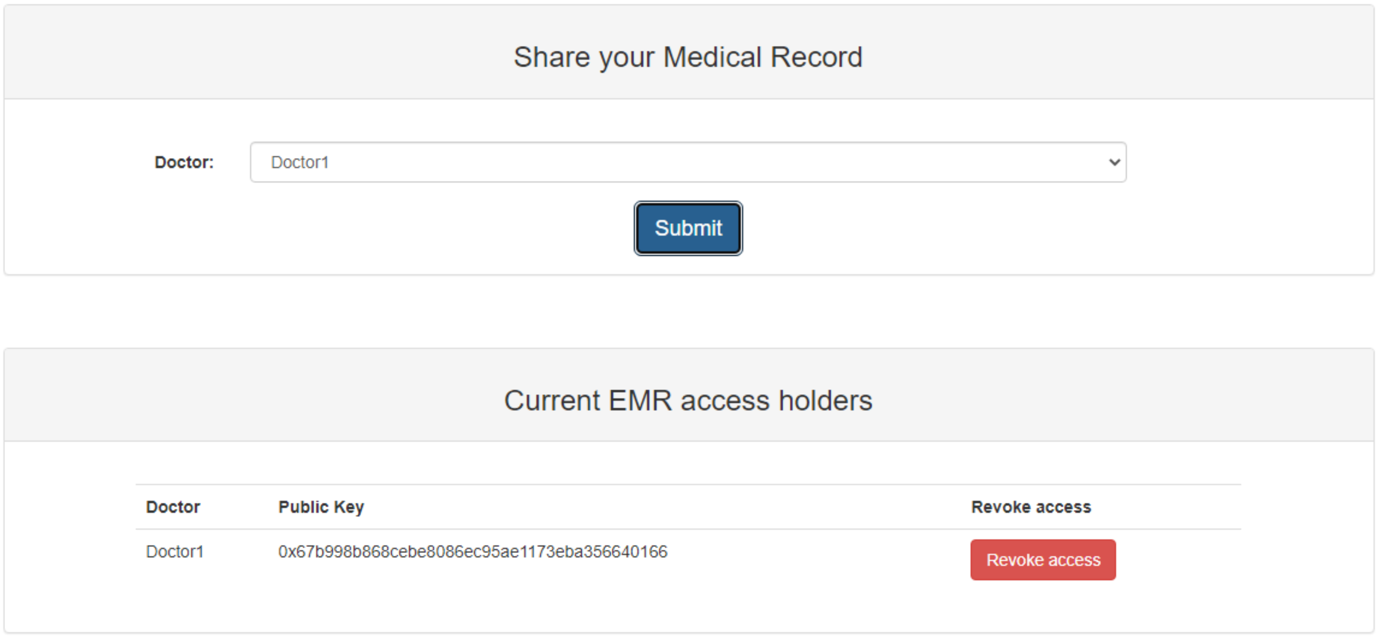
DETAILS DATA HEX

EDIT

Estimated gas fee	0.02	0.02 ETH
Site suggested	Max fee:	0.02 ETH
Total	2.02	2.02 ETH
Amount + gas fee	Max amount:	2.02 ETH

Reject Confirm

Fig 4.11- Patient Share Transaction access


Share your Medical Record

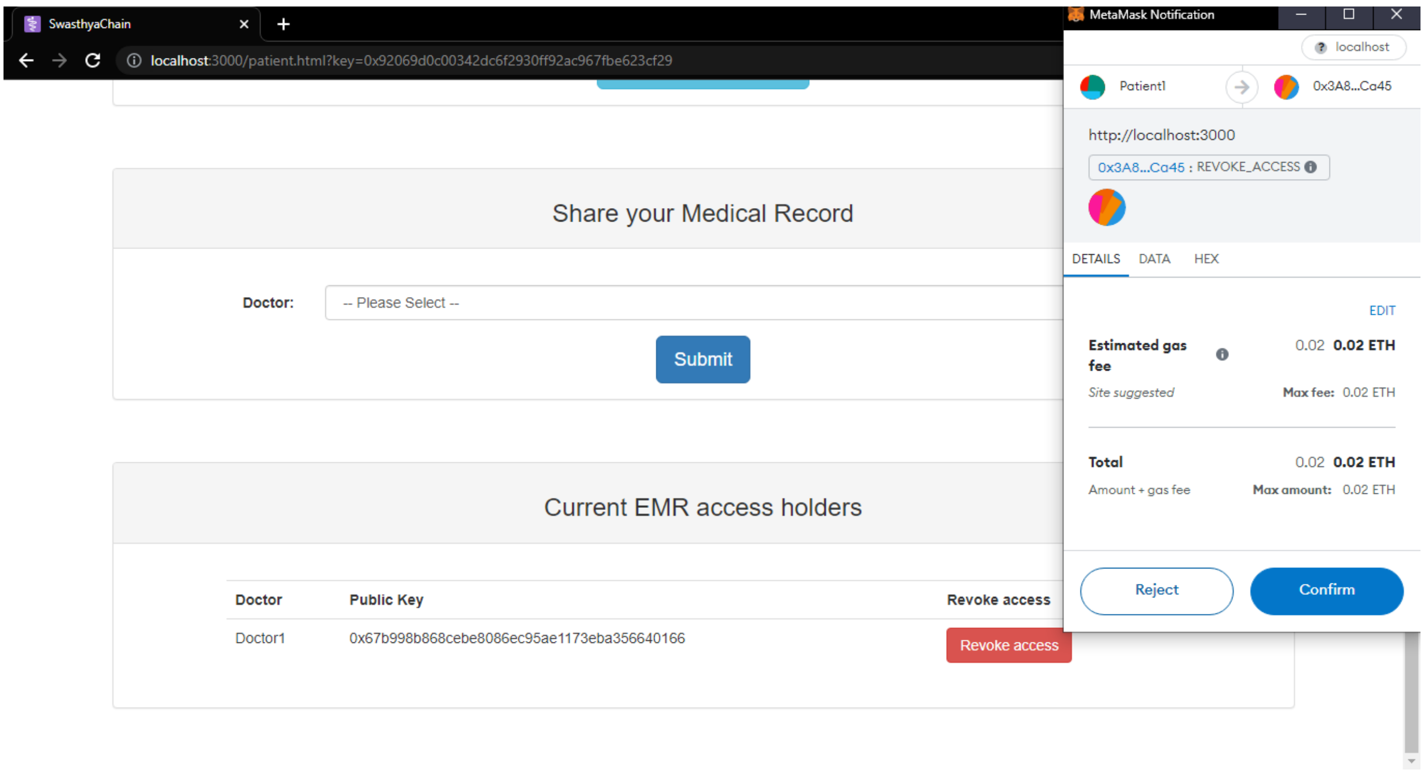
Doctor: Doctor1

Submit

Current EMR access holders

Doctor	Public Key	Revoke access
Doctor1	0x67b998b868cebe8086ec95ae1173eba356640166	Revoke access

Fig 4.12- Patient Revoke


SwasthyaChain

localhost:3000/patient.html?key=0x92069d0c00342dc6f2930ff92ac967fbe623cf29

Share your Medical Record

Doctor: -- Please Select --

Submit

Current EMR access holders

Doctor	Public Key	Revoke access
Doctor1	0x67b998b868cebe8086ec95ae1173eba356640166	Revoke access

MetaMask Notification

localhost

Patient1 → 0x3A8...Ca45

http://localhost:3000

0x3A8...Ca45 : REVOKE_ACCESS

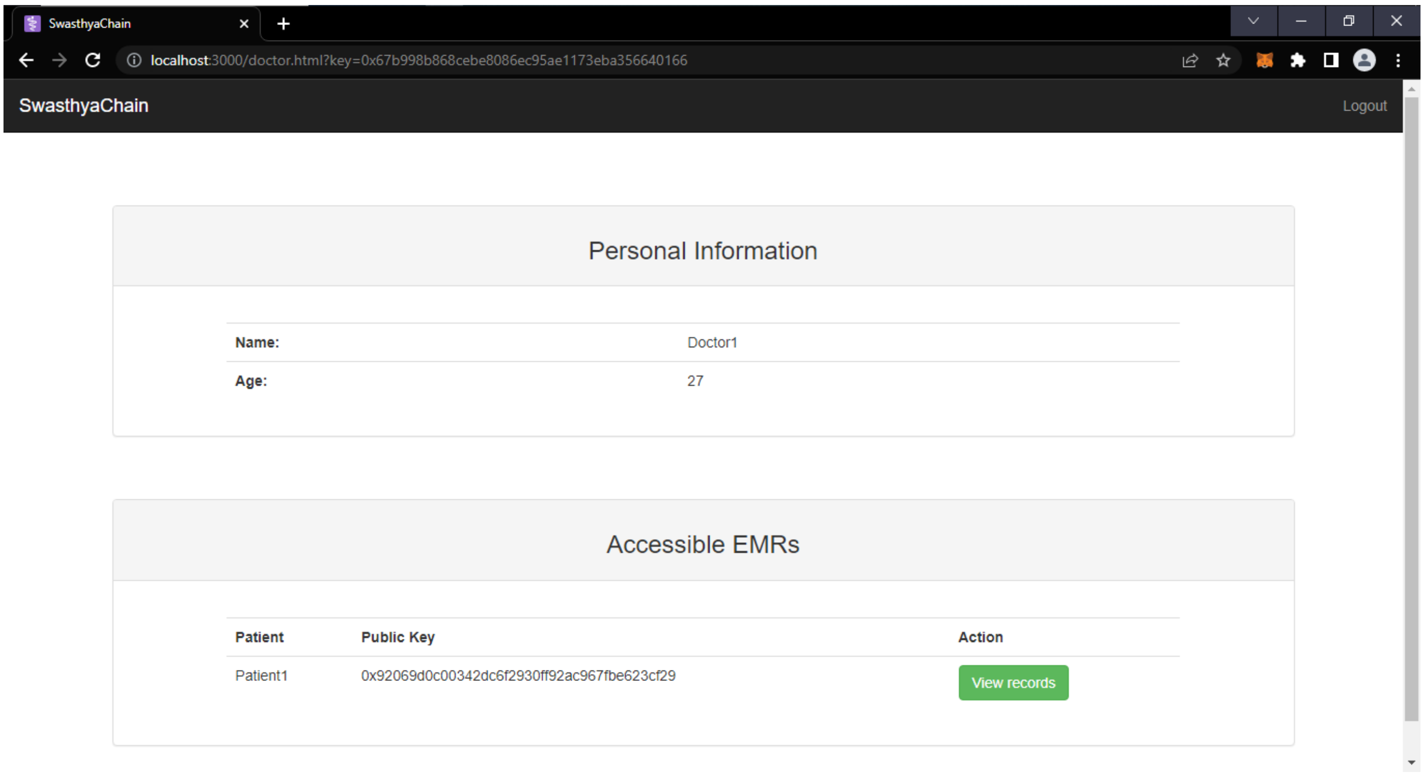
DETAILS DATA HEX

EDIT

Estimated gas fee	0.02	0.02 ETH
Site suggested	Max fee:	0.02 ETH
Total	0.02	0.02 ETH
Amount + gas fee	Max amount:	0.02 ETH

Reject Confirm

Fig4.13- Patient Transaction Revoke


SwasthyaChain

localhost:3000/doctor.html?key=0x67b998b868cebe8086ec95ae1173eba356640166

SwasthyaChain Logout

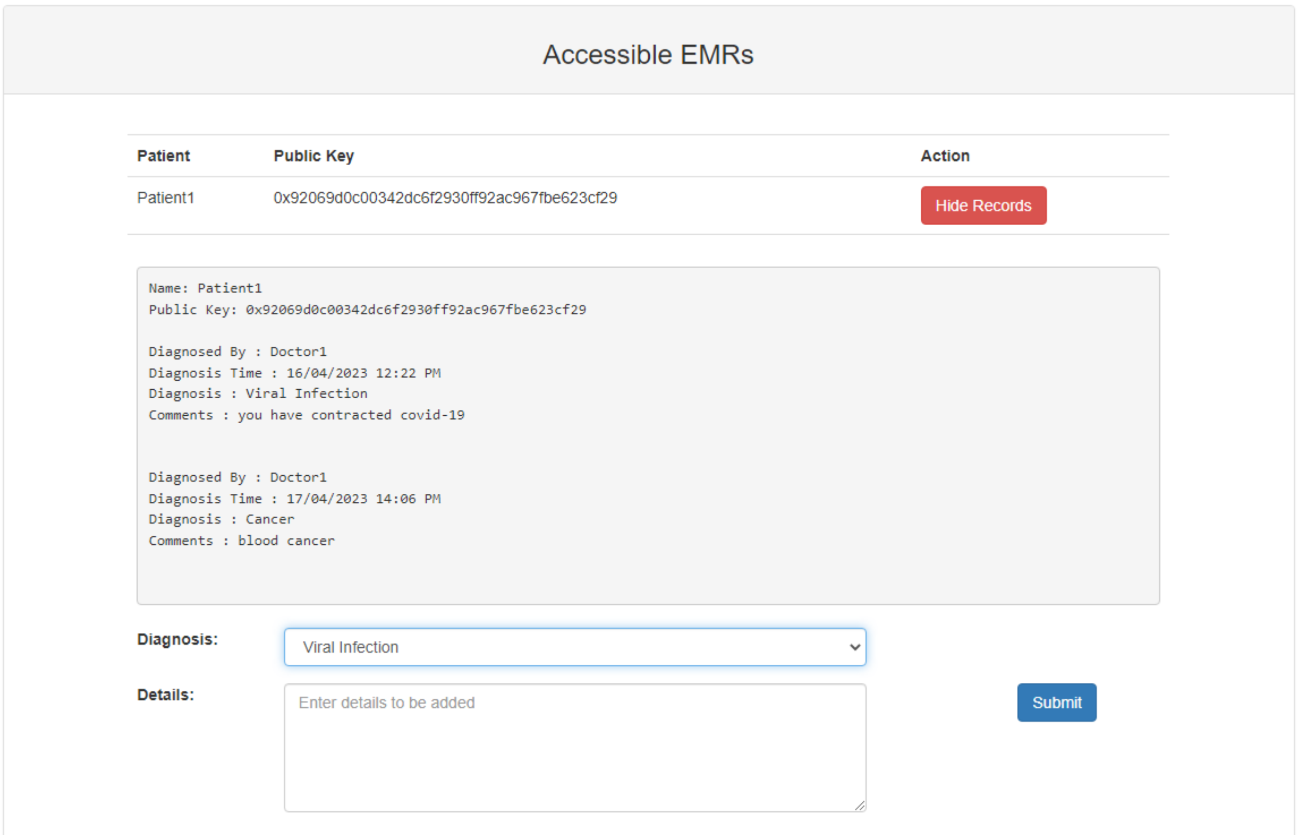
Personal Information

Name:	Doctor1
Age:	27

Accessible EMRs

Patient	Public Key	Action
Patient1	0x92069d0c00342dc6f2930ff92ac967fbe623cf29	View records

Fig 4.14 - Doctor


Accessible EMRs

Patient	Public Key	Action
Patient1	0x92069d0c00342dc6f2930ff92ac967fbe623cf29	Hide Records

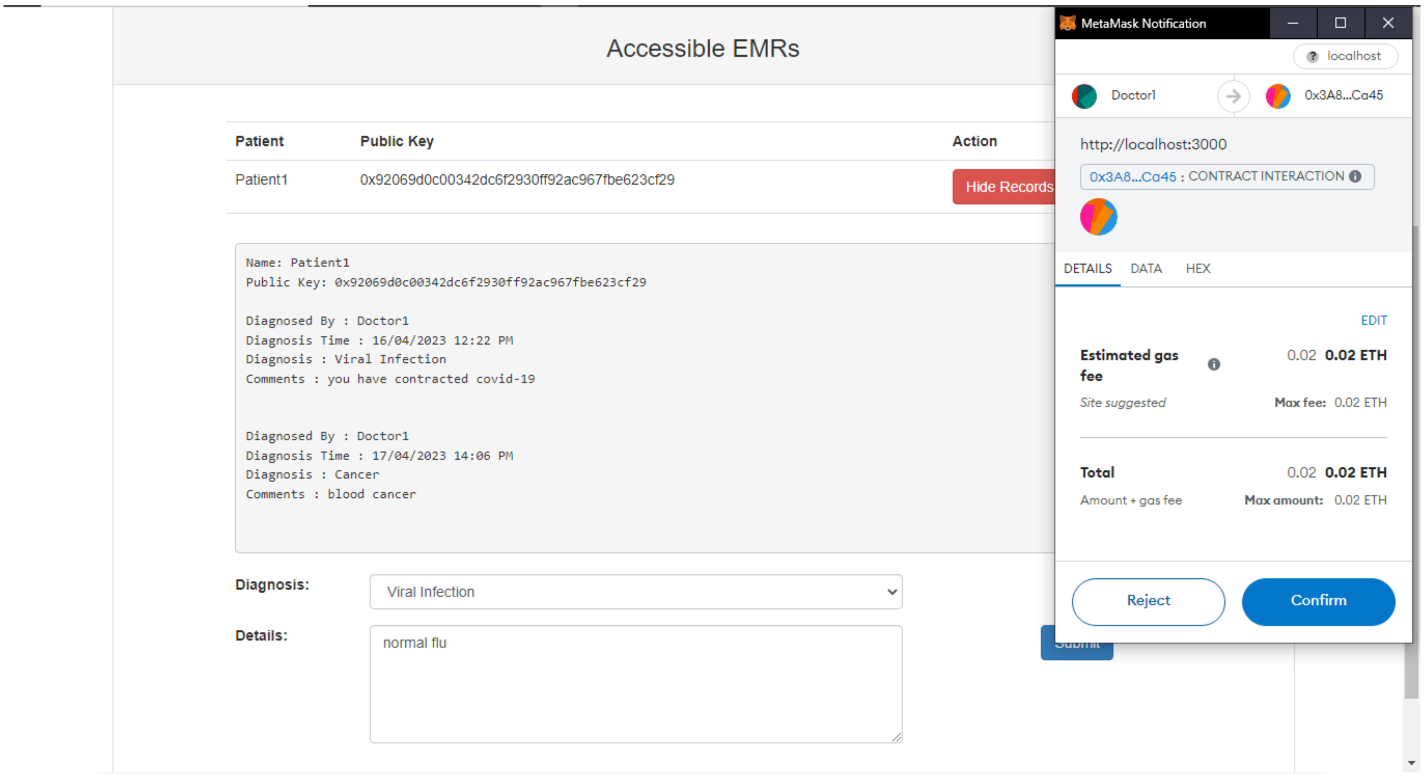
```
Name: Patient1
Public Key: 0x92069d0c00342dc6f2930ff92ac967fbe623cf29

Diagnosed By : Doctor1
Diagnosis Time : 16/04/2023 12:22 PM
Diagnosis : Viral Infection
Comments : you have contracted covid-19


Diagnosed By : Doctor1
Diagnosis Time : 17/04/2023 14:06 PM
Diagnosis : Cancer
Comments : blood cancer
```

Diagnosis: Viral Infection

Details: Enter details to be added

Submit

Fig 4.15 - Doctor Diagnosis


Accessible EMRs

Patient	Public Key	Action
Patient1	0x92069d0c00342dc6f2930ff92ac967fbe623cf29	Hide Records

```
Name: Patient1
Public Key: 0x92069d0c00342dc6f2930ff92ac967fbe623cf29

Diagnosed By : Doctor1
Diagnosis Time : 16/04/2023 12:22 PM
Diagnosis : Viral Infection
Comments : you have contracted covid-19


Diagnosed By : Doctor1
Diagnosis Time : 17/04/2023 14:06 PM
Diagnosis : Cancer
Comments : blood cancer
```

Diagnosis: Viral Infection

Details: normal flu

MetaMask Notification

localhost

Doctor1 → 0x3A8...Ca45

http://localhost:3000

0x3A8...Ca45 : CONTRACT INTERACTION

DETAILS DATA HEX

EDIT

Estimated gas fee 0.02 **0.02 ETH**

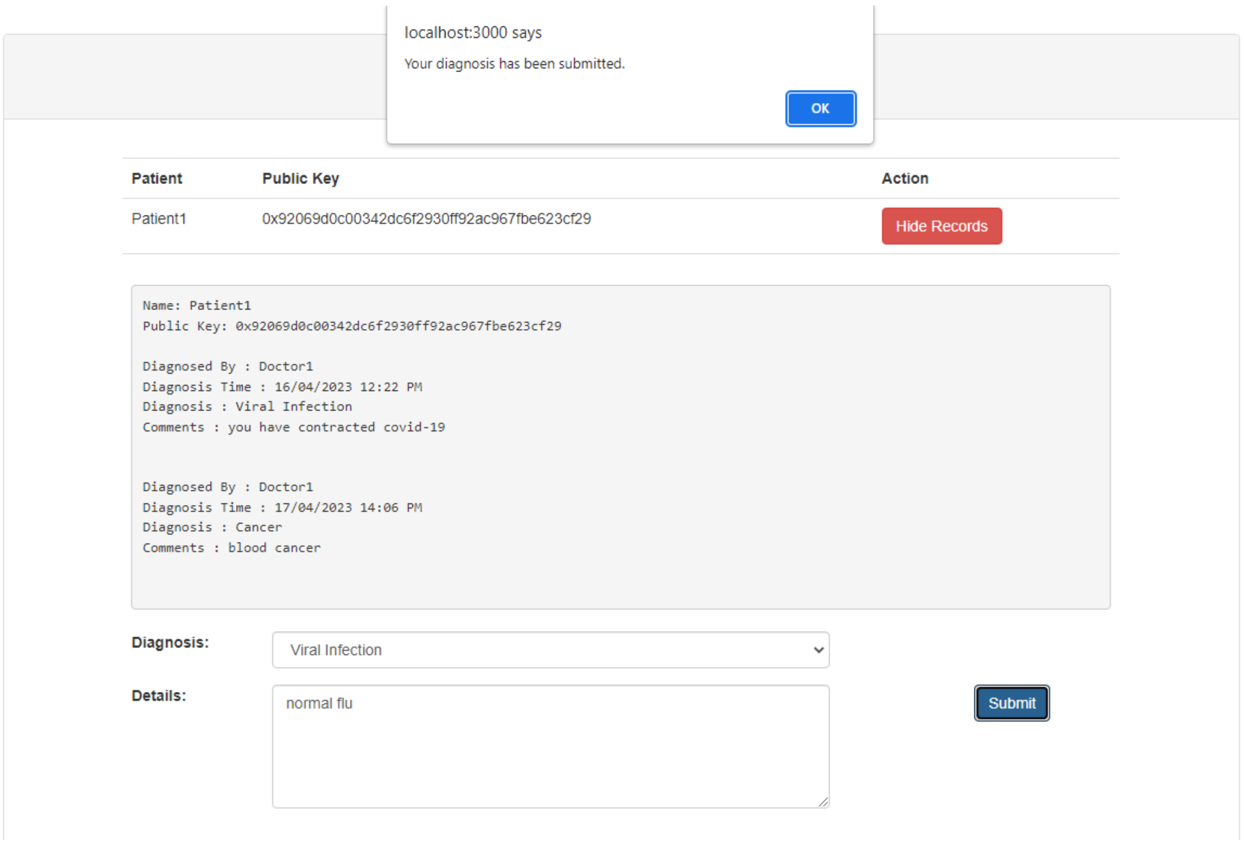
Site suggested **Max fee:** 0.02 ETH

Total 0.02 **0.02 ETH**

Amount + gas fee **Max amount:** 0.02 ETH

Reject Confirm

Fig 4.16 - Doctor Transaction Diagnosis



localhost:3000 says

Your diagnosis has been submitted.

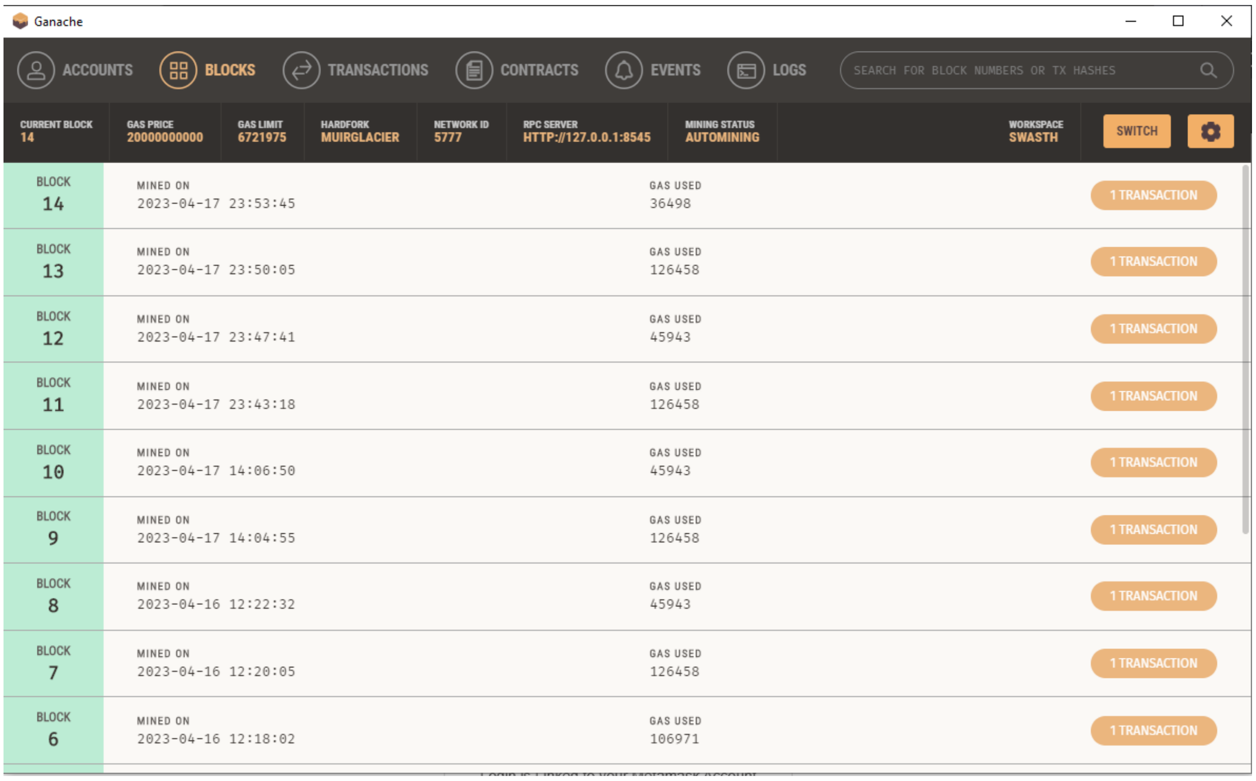
Patient	Public Key	Action
Patient1	0x92069d0c00342dc6f2930ff92ac967fbe623cf29	Hide Records

```
Name: Patient1
Public Key: 0x92069d0c00342dc6f2930ff92ac967fbe623cf29

Diagnosed By : Doctor1
Diagnosis Time : 16/04/2023 12:22 PM
Diagnosis : Viral Infection
Comments : you have contracted covid-19


Diagnosed By : Doctor1
Diagnosis Time : 17/04/2023 14:06 PM
Diagnosis : Cancer
Comments : blood cancer
```

Diagnosis: Viral Infection

Details: normal flu

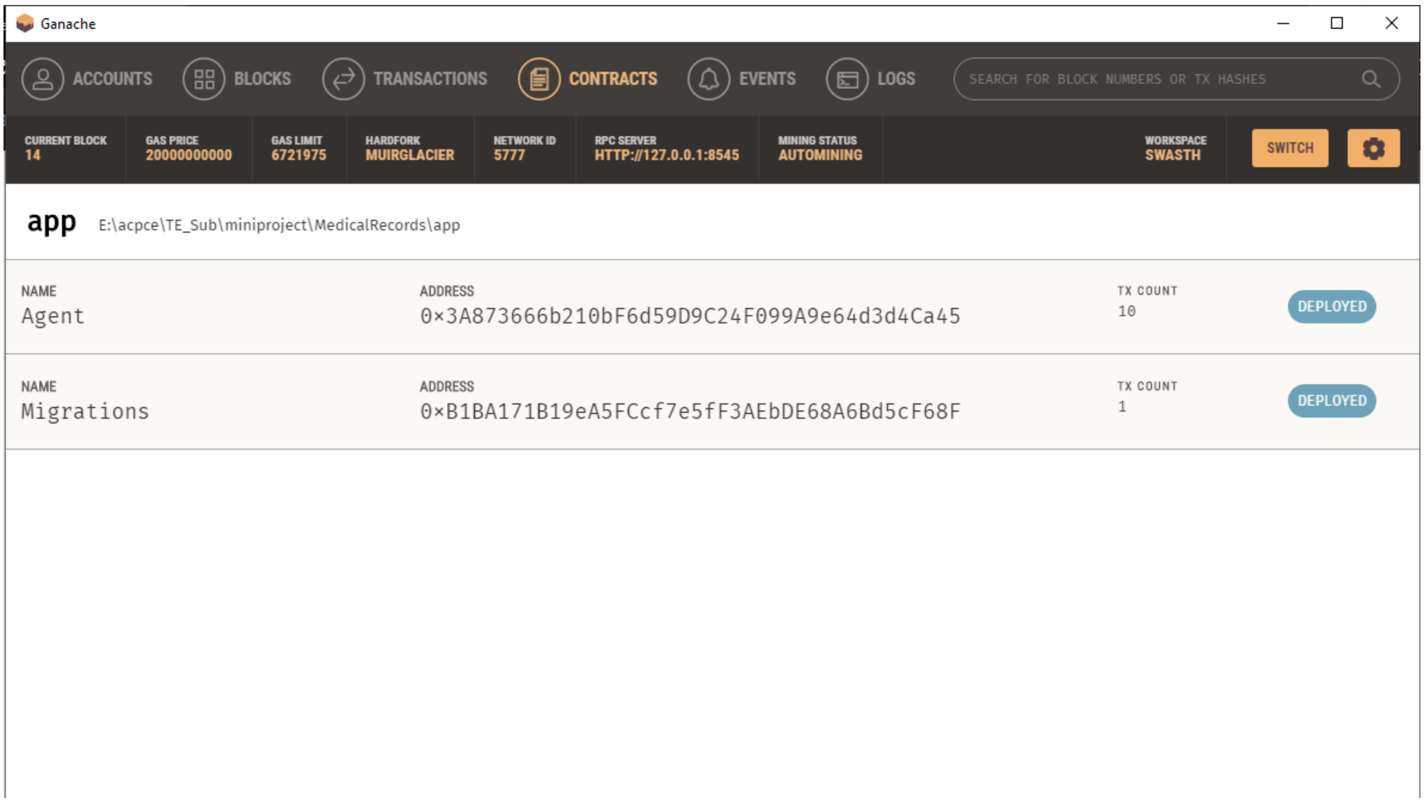
Fig 4.17- Doctor Diagnosis confirm



CURRENT BLOCK 14 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:8545 | MINING STATUS AUTOMINING | WORKSPACE SWASTH

BLOCK	MINED ON	GAS USED	
14	2023-04-17 23:53:45	36498	1 TRANSACTION
13	2023-04-17 23:50:05	126458	1 TRANSACTION
12	2023-04-17 23:47:41	45943	1 TRANSACTION
11	2023-04-17 23:43:18	126458	1 TRANSACTION
10	2023-04-17 14:06:50	45943	1 TRANSACTION
9	2023-04-17 14:04:55	126458	1 TRANSACTION
8	2023-04-16 12:22:32	45943	1 TRANSACTION
7	2023-04-16 12:20:05	126458	1 TRANSACTION
6	2023-04-16 12:18:02	106971	1 TRANSACTION

Fig 4.18 - Ganache Blocks



Ganache

CURRENT BLOCK 14 | GAS PRICE 20000000000 | GAS LIMIT 6721975 | HARDFORK MUIRGLACIER | NETWORK ID 5777 | RPC SERVER HTTP://127.0.0.1:8545 | MINING STATUS AUTOMINING | WORKSPACE SWASTH

app E:\acpce\TE_Sub\miniproject\MedicalRecords\app

NAME	ADDRESS	TX COUNT	
Agent	0×3A873666b210bF6d59D9C24F099A9e64d3d4Ca45	10	DEPLOYED
Migrations	0×B1BA171B19eA5FCcf7e5fF3AEbDE68A6Bd5cF68F	1	DEPLOYED

Fig 4.19 - Ganache Contracts

Conclusion:

In the EHR system the patient can access their report and can use the report for their lifetime with security. The private key is used for the patient which can be used for the further use of the reports. The one who don’t have the private key cannot involve in the process of retrieving data. Hence the Health Records of the patients are more secured with the BlockChain and can used with their own private key as well as they can make use of that for further reference. Currently, there is a huge obstacle for using ethereum platform, which lays in difficulty of obtaining ETH units. The units can be obtained either by mining or purchased for fiat or cryptocurrency like Bitcoin. Once ETH units are available, the publishing and execution of smart contract is really smooth. At this stage, the solidity code has been deployed and verified. The deployed contract is accessed with the help of metamask injected on web3. Following the port from plain JS to Truffle Framework, we have developed a web application user interface for the health record system. For increased security, we can add account authentication features using some unique identity features.

References:

- https://github.com/

- K. D. Mandl, P. Szolovits, and I. S.Kohane, Public standards and patients' control: How to keep electronic medical records accessible but private, BMJ, 2001, vol. 322, no. 7281, pp. 283_287.

- M. Weinger, Dangers of postoperative opioids, APSF Newslett., 2007, vol. 21,no. 4, pp. 61– 68.

- S.D. Cannoy and A.F. Salam, A Framework for Health Care Information Assurance Policy and Compliance, communications of the ACM, 2010 ,vol. 53, no. 3.
- https://www.youtube.com

- J Liu la Protecting mobile health records in cloud computing secure, efficient, and anonymous design | ACM Trans. Embed. Comput Syst., Apr. 2017, vol. 16. 10. 2.

- G. Irving and J. Holden, How blockchain-timestamped protocols could improve the trustworthiness of medical science ,F1000Research, 2016, vol. 5, p. 22

Acknowledgments

I thank my college Principal Dr. V. N. Pawar sir for providing the required resources for the development of the project. I would also like to thank HOD Prof. S. P. Pawar for suggesting such a great project topic for departmental purposes. My sincere thanks to my Project Guide (**Dr. Ulka Shirole**) for helping, suggesting new ideas and guiding me throughout the semester. I am also grateful to all the faculty members for their support and encouragement.

Date :