

Project name-Blockchain Technology For Food Tracking System

Nm team id-NM2023TMID06446

Date-25-10-2023

Project Report

Project Report

- 1. INTRODUCTION**
 - 1.1 Project Overview
 - 1.2 Purpose
- 2. LITERATURE SURVEY**
 - 2.1 Existing problem
 - 2.2 References
 - 2.3 Problem Statement Definition
- 3. IDEATION & PROPOSED SOLUTION**
 - 3.1 Empathy Map Canvas
 - 3.2 Ideation & Brainstorming
- 4. REQUIREMENT ANALYSIS**
 - 4.1 Functional requirement
 - 4.2 Non-Functional requirements
- 5. PROJECT DESIGN**
 - 5.1 Data Flow Diagrams & User Stories
 - 5.2 Solution Architecture
- 6. PROJECT PLANNING & SCHEDULING**
 - 6.1 Technical Architecture
 - 6.2 Sprint Planning & Estimation
 - 6.3 Sprint Delivery Schedule
- 7. CODING & SOLUTIONING**
 - 7.1 Feature 1
 - 7.2 Feature 2
 - 7.3 Database Schema (if Applicable)
- 8. PERFORMANCE TESTING**
 - 8.1 Performance Metrics
- 9. RESULTS**
 - 9.1 Output Screenshots
- 10. ADVANTAGES & DISADVANTAGES**
- 11. CONCLUSION**
- 12. FUTURE SCOPE**
- 13. APPENDIX**
 - Source Code
 - GitHub & Project Demo Link

1.Introduction

The recent advent in technology is affecting all parts of human life and is changing the way we use and perceive things previously. Just like the changes technology has offered in various other sectors of life, it is also finding new ways for improvement in healthcare sector. The main benefits that advancement in technology is offering are to improve security, user experience and other aspects of healthcare sector. These benefits were offered by Electronic Health Record (EHR) and Electronic Medical Record (EMR) systems. However, they still face some issues regarding the security of medical records, user ownership of data, data integrity etc. The solution to these issues could be the use of a novel technology, i.e., Blockchain. This technology offers to provide a secure, temper-proof platform for storing medical records and other healthcare related information.

Before the advent of modern technology, healthcare sector used paper based system to store the medical records, i.e., using handwritten mechanism. This paper-based medical record system was inefficient, insecure, unorganized and was not temper-proof. It also faced the issue of data- duplication and redundancy as all the institutions that patient visited had various copies of patient's medical records.

The healthcare sector faced a trend shift towards EHR systems that were designed to combine paper-based and electronic medical records (EMR). These systems were used to store clinical notes and laboratory results in its multiple components [1]. They were proposed to enhance the safety aspect of the patients by preventing errors and increasing information access [2]. The goal of EHR systems was to solve the problems faced by the paper-based healthcare records and to provide an efficient system that would transform the state of healthcare sector

The EHR systems have been implemented in a number of hospitals around the world due the benefits it provides, mainly the improvement in security and its cost-effectiveness. They are considered a vital part of healthcare sector as it provides much functionality to the healthcare [4]. These functionalities are electronic storage of medical records, patients' appointment management, billing and accounts, and lab tests. They are available in many of the EHR system being used in the healthcare sector. The basic focus is to provide secure, temper-proof, and shareable medical records across different platforms. Despite the fact that notion behind usage of EHR systems in the hospitals or healthcare was to improve the quality of healthcare, these systems faced certain problems and didn't meet the expectations associated with them [3]. A study was conducted in Finland to find the experiences of nursing staff with the EHR, it was concluded that EHR systems faced the problems related to them being unreliable and having a poor state of user-friendliness

1.1 Project overview

Blockchain have been an interesting research area for a long time and the benefits it provides have been used by a number of various industries. Similarly, the healthcare sector stands to benefit immensely from the blockchain technology due to security, privacy, confidentiality and decentralization. Nevertheless, the Electronic Health Record (EHR) systems face problems regarding data security, integrity and management. In this paper, we discuss how the blockchain technology can be used to transform the EHR systems and could be a solution of these issues. We present a framework that could be used for the implementation of blockchain technology in healthcare sector for EHR.

The aim of our proposed framework is firstly to implement blockchain technology for EHR and secondly to provide secure storage of electronic records by defining granular access rules for the users of the proposed framework. Moreover, this framework also discusses the scalability problem faced by the blockchain technology in general via use of off-chain storage of the records. This framework provides the EHR system with the benefits of having a scalable, secure and integral blockchain-based solution.

1.2 purpose

A. Interoperability

It is the way for different information systems to exchange information between them. The information should be exchangeable and must be usable for further purposes. An important aspect of EHR systems is its Health Information Exchange (HIE) or in general data sharing aspect. With a number of EHR systems being deployed in various hospitals they have a varying level of terminologies, technical and functional capabilities which makes it to have no universally defined standard [6]. Moreover, at technical level the medical records being exchanged should be interpretable, and that interpreted piece of information could be further used [6].

B. Information Asymmetry

Today the greatest problem in healthcare sector defined by the critics is information asymmetry which refers to one party having better access to information than the other party. In case of EHR systems, or in general healthcare sector is suffering from this problem as doctors or hospitals have access to the patient's records, thus making it central. If a patient wants to access his medical records he would have to follow a long and tedious process to access them. The information is centralized to only a single healthcare organization and its control is only provided to the hospitals or organizations.

C. Data Breaches

Data breaches in healthcare sector also calls for the need of a better platform. A study [7] was done for analyzing the data breaches in EHR systems and it depicted that 173 million data entries have been compromised in these systems since October 2009. Another study conducted by Argaw *et al.* [8], explains that 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 [9]–[11].

Moreover, many EHR systems are not designed to fulfill the needs and requirements of the patients and face the issues related to inefficiency and poor adaptation of these systems [12]. The literature also suggests that use of EHRs have introduced negative consequences to information processing [2]. These problems make it reasonable to find a platform that would be helpful in transforming healthcare sector to be patient-centered, i.e., Blockchain. A platform which is secure, transparent and it also provides data integrity to the medical records of the patients.

This paper proposes a framework that creates such a decentralized platform that would store patient's medical records and give access of those records to providers or concerned individuals, i.e., patient. We also intend to solve the scalability problem of blockchain, as it is not in the design of blockchain to store huge volumes of data on it. So, we would use off-chain scaling method that makes use of the underlying medium to solve the scalability problem by storing the data on that medium. Moreover, our proposed work is intending to solve the above mentioned information asymmetry and data breaches problem faced by the EHR system.

This paper is organized as follows the section II of this paper summarizes the basics of blockchain technology and its dependencies; section III narrates the related work done in this domain. The section IV explains the design and architecture of the proposed framework and section V explains the performance of this framework. The last section provides the conclusion and references.

2.LITERATURE SURVEY

2.1 Existing problem

In most African countries like Nigeria, preservation and conservation of hospital documents and records has posed a serious problem. The deterioration of materials forms the basic problem of registries and gives rise to preservation and conservation of records. Therefore, knowledge of the causes of deterioration of materials and how to cater for these materials is essential for all librarians and others who are concerned about the preservation and conservation of information stored in books and non-book formats. Hence, it is in this premise that this paper is set to examine some challenges that affect the preservation and conservation of records in two health Institutions in Lagos State.

2.2Reference

Id.	Author	Year	Publisher	
A01	Linn and Koo ²³	2016	HealthIT	
A02	Ekblaw et al. ³	2016	IEEE	C

Id.	Author	Year	Publisher	
A03	Yue et al. ²	2016	Springer	
A05	Nichol and Brandt ¹⁰	2016	ResearchGate	
A16	Hashemi et al. ⁹	2016	IEEE	C
A04	Roehrs et al. ⁶	2017	Elsevier	
A07	Yang and Yang ²⁵	2017	NISK	C
A10	Smith and Dhillon ⁴³	2017	AMCIS	C
A11	Rabah ¹	2017	MRJOURNALS	
A12	Cheng et al. ⁴	2017	Taylor & Francis	
A13	Liu et al. ¹²	2017	IEEE	C
A15	Ichikawa et al. ¹³	2017	JMIR	
A19	Kshetri ¹⁴	2017	Elsevier	
A21	Shae and Tsai ¹⁵	2017	IEEE	C
A23	Karafiloski and Mishev ¹⁶	2017	IEEE	C
A24	Xia et al. ¹⁷	2017	IEEE	
A25	Thomason ¹⁸	2017	GHJ	
A26	Dubovitskaya et al. ¹⁹	2017	JAMIA	
A27	Lemieux ²⁰	2017	IEEE	C
A29	Priisalu and Ottis ²¹	2017	Springer	
A35	Kuo et al. ²²	2017	JAMIA	
A06	Cyran ²⁴	2018	Partners in Digital Health	
A08	Patel ¹¹	2018	SAGE	
A09	Dagher et al. ⁴²	2018	Elsevier	
A14	Ribitzky et al. ²⁶	2018	Partners in Digital Health	
A17	Zhang et al. ²⁷	2018	JNCA—Elsevier	
A18	Niranjanamurthy et al. ²⁸	2018	Springer	
A20	Guo et al. ²⁹	2018	IEEE	
A22	Fan et al. ³⁰	2018	IET Comm	
A28	Mannaro et al. ³¹	2018	IEEE	C
A30	Wang and Song ³²	2018	Springer	
A31	Kleinaki et al. ³³	2018	Elsevier	
A32	Banerjee et al. ³⁴	2018	Elsevier and KeAi	
A33	Gordon and Catalini ³⁵	2018	Elsevier	
A34	Grover et al. ³⁶	2018	Elsevier	
A36	Jiang et al. ³⁷	2018	IEEE	C
A37	Mamoshina et al. ³⁸	2018	Impact Journals	
A38	Roman-Belmonte et al. ³⁹	2018	MEDKNOW Publications	

Problem statement definition

It's no secret that many physicians are unhappy with their electronic health records (EHRs). They say they spend too much time keying in data and too

little making eye contact with patients. They say their electronic records are clunky, poorly designed, hard to navigate, and cluttered with useless detail that colleagues have cut and pasted to meet documentation requirements. Meanwhile, the data they really need are buried almost beyond retrieval.

Not all physicians feel this way. Two-thirds of primary care physicians say they are satisfied with their current EHRs, according to a 2018 survey by The Harris Poll. But the critics have a point. Current EHRs are not well-designed to meet the needs of users. And they don't do enough to make clinicians smarter and more efficient. This doesn't mean we would be better off in the paper world of 10 years ago. But it does mean that EHRs need improvement.

3. IDEATION & PROPOSED SOLUTION

Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals.




3.2 Brainstorm & Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Step-1: Team Gathering, Collaboration and Select the Problem Statement


Template



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

⌚ 10 minutes to prepare
🕒 1 hour to collaborate
👥 2-8 people recommended



Before you collaborate

search and collect information about the Electronic health records, collect the required information

⌚ 10 minutes

A

Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B

Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.

C

Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#)

1


Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

⌚ 5 minutes

PROBLEM

EHR providers may not update their systems



Key rules of brainstorming

To run a smooth and productive session

🗣️ Stay in topic.

💡 Encourage wild ideas.

⏸️ Defer judgment.

👂 Listen to others.

🗣️ Go for volume.

👁️ If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm
Write down any ideas that come to mind that address your problem statement.

🕒 10 minutes

TIP
You can select a sticky note and hit the pencil icon to sketch (don't start drawing!)

Person 1

Physician access to patient information

Computerized provider order entry

Secure electronic communication among providers and patients.

Person 2

Computerized administration processes

Standards-based electronic data storage

reporting for patient safety

Person 3

uninterrupted power supply

high security data encryption

cloud access to patient and do doctors only

Person 4

provide privacy and policy

continous upgrades

login id and password

3

Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

🕒 20 minutes

TIP
Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as business unfolds your results.

Computerized provider order entry.

Physician access to patient information, such as diagnoses, allergies, lab results, and medications.

Step-3: Idea Prioritization

4

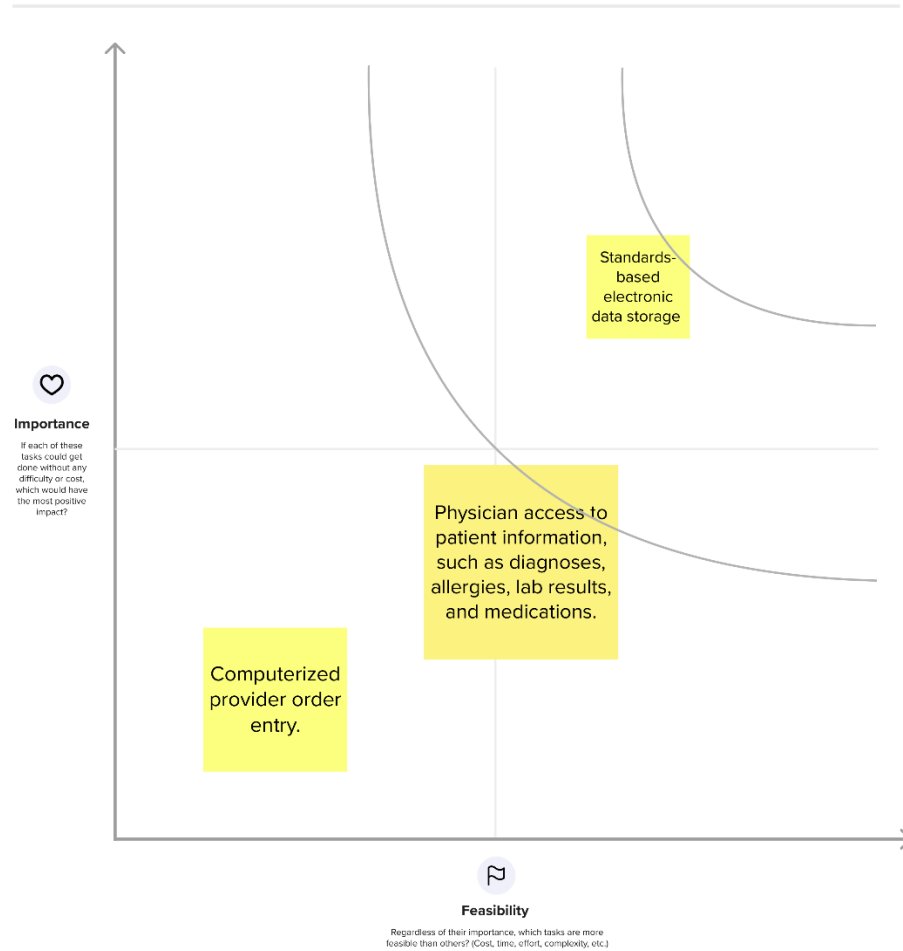
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the H key on the keyboard.



4.1 functional requirements

Visual studio code

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET). Begin your journey with VS Code with these introductory videos.

History

Visual Studio Code was first announced on April 29, 2015, by Microsoft at the 2015 [Build](#) conference. A [preview](#) build was released shortly thereafter.^[15]

On November 18, 2015, the source of Visual Studio Code was released under the [MIT License](#) and made available on [GitHub](#). Extension support was also announced.^[16] On April 14, 2016, Visual Studio Code graduated from the [public preview](#) stage and was [released to the Web](#).^[17] Microsoft has released most of Visual Studio Code's [source code](#) on [GitHub](#) under the permissive [MIT License](#),^{[6][18]} while the binary releases by Microsoft are [freeware](#),^[8] and include [proprietary](#) code.^[5] A community distribution, called VSCodium, is maintained, which provides MIT licensed binaries.^{[10][19][20]}

Features[edit]

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust, and Julia.^{[21][22][23][24][25]} It is based on the Electron framework,^[26] which is used to develop Node.js web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).^[27]

Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.^[28]



Visual Studio Code logo

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports many programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.[29]

Visual Studio Code can be extended via extensions,[30] available through a central repository. This includes additions to the editor[31] and language support.[29] A notable feature is the ability to create extensions that add support for new languages, themes, debuggers, time travel debuggers, perform static code analysis, and add code linters using the Language Server Protocol.[32]

Source control is a built-in feature of Visual Studio Code. It has a dedicated tab inside of the menu bar where users can access version control settings and view changes made to the current project. To use the feature, Visual Studio Code must be linked to any supported version control system (Git, Apache Subversion, Perforce, etc.). This allows users to create repositories as well as to make push and pull requests directly from the Visual Studio Code program.

Visual Studio Code includes multiple extensions for FTP, allowing the software to be used as a free alternative for web development. Code can be synced between the editor and the server, without downloading any extra software.

Visual Studio Code allows users to set the code page in which the active document is saved, the newline character, and the programming language of the active document. This allows it to be used on any platform, in any locale, and for any given programming language.[promotion?]

Visual Studio Code collects usage data and sends it to Microsoft, although this can be disabled.[33] Some of the telemetry code is accessible to the public,[34] but according to Visual Studio Code maintainers, some telemetry functionality is also added to the program before it is released with a proprietary license.[35][5]

Node.js

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.

Platform architecture[\[edit\]](#)

Node.js brings event-driven programming to web servers, enabling development of fast web servers in JavaScript. Developers can create scalable servers without using threading, by using a simplified model of event-driven programming that uses callbacks to signal the completion of a task. Node.js connects the ease of a scripting language (JavaScript) with the power of Unix network programming.

Node.js was built on top of Google's V8 JavaScript engine since it was open-sourced under the BSD license. It is proficient with internet fundamentals such as HTTP, DNS, and TCP. JavaScript was also a well-known language, making Node.js accessible to the web development community.

Console

The **console** is a module provided by Node.js that is akin to the JavaScript console in the browser when you inspect a webpage. The console has methods that are available for us to use for debugging purposes.

- `console.log()`: Frequently used to log some sort of output.
- `console.warn()`: Explicitly delivers a warning to the console.
- `console.error()`: Explicitly delivers an error message to the console. You can log an error as a string or as an object. If logged as a new `Error()`, a traceback will be included as part of the message.
- `console.trace()`: Logs a traceback when an error occurs in your code. Gives line number and column number of the file that the error probably occurred.

File System

The **file system (fs) module** allows us to interact with files in Node.js. There are synchronous and asynchronous methods that can be used to read or write to a file using the fs module. In contrast to using console or the Buffer class, we need to import the fs module into the file that we would like to use in order to get it to work.

The code example below shows how the `readFile`, an asynchronous method, works. Notice that the last argument in the method is a callback function whose first argument is an error. By definition this callback will always pass in the error first before the data.

Metamask

MetaMask is a popular cryptocurrency wallet and browser extension that allows users to manage their Ethereum-based assets and interact with decentralized applications (DApps) on the Ethereum blockchain. It was developed by ConsenSys, a blockchain technology company.

Here are some key features and functions of MetaMask:

1. **Wallet:** MetaMask serves as a secure digital wallet that can store Ethereum (ETH) and other Ethereum-based tokens (ERC-20 and ERC-721 tokens). Users can send, receive, and manage their cryptocurrency holdings through the wallet.

2. **Browser Extension:** MetaMask is primarily a browser extension available for various web browsers like Chrome, Firefox, and Brave. It integrates with your browser, making it easy to interact with Ethereum-based DApps directly from your web browser.

3. **DApp Interaction:** MetaMask allows users to connect to decentralized applications (DApps) on the Ethereum blockchain. Users can access various DeFi (Decentralized Finance) platforms, NFT (Non-Fungible Token) marketplaces, games, and other blockchain-based services through the extension.

4. **Private Keys:** MetaMask uses a hierarchical deterministic (HD) wallet system and stores private keys locally on your device, enhancing security. Users are responsible for safeguarding their private keys and seed phrases to protect their funds.

5. **Cross-Platform Compatibility:** MetaMask is available as both a browser extension and a mobile app for iOS and Android devices. This allows users to access their wallet and interact with DApps on multiple platforms.

6. **Network Selection:** Users can choose which Ethereum network they want to interact with, such as the Ethereum mainnet, testnets (Ropsten, Rinkeby, Goerli, and Kovan), and custom networks.

7. **Security Features:** MetaMask includes various security features, such as password protection, biometric authentication, and a phishing detection system to help users avoid potential scams.

8. Token Swaps: MetaMask provides a token swap feature that allows users to exchange one cryptocurrency for another directly within the wallet.

9. Gas Fees: Users can customize gas fees for their transactions, allowing them to control the speed and cost of their Ethereum transactions.

MetaMask has gained widespread popularity due to its user-friendly interface and the convenience it offers for interacting with the Ethereum blockchain and various decentralized applications. However, users should always exercise caution and follow best security practices when managing their cryptocurrency assets.

Procedure to install metamask

To install MetaMask, you can follow these steps. Please note that the procedure may vary slightly depending on your web browser (commonly used on Chrome or Firefox). As of my last knowledge update in September 2021, MetaMask was available as a browser extension. Please ensure you are using a secure and up-to-date web browser. Here's a general procedure for installing MetaMask:

****1. Open your web browser:**** Launch your preferred web browser (e.g., Chrome, Firefox, or Brave).

****2. Go to the MetaMask website:****

- Type "MetaMask" in your browser's search bar or go directly to the MetaMask website: <https://metamask.io/>.

****3. Click on "Get Chrome Extension" (or the equivalent for your browser):**** You will see an option to install the MetaMask browser extension. Click on this option.

****4. Install the extension:****

- For Chrome: Click "Add to Chrome" or "Add to Brave" if you're using the Brave browser.
- For Firefox: Click "Add to Firefox."

****5. Confirm the installation:****

- A pop-up or prompt will appear asking you to confirm the installation. Click "Add Extension" to proceed.

****6. Wait for the installation:**** The extension will be downloaded and added to your browser. Once the installation is complete, you'll see the MetaMask fox icon in your browser's extensions area.

****7. Set up your wallet:****

- Click the MetaMask fox icon in your browser.
- Click "Get Started" to begin the wallet setup process.
- You will be guided through creating a new wallet or importing an existing one. Follow the instructions provided by MetaMask.

****8. Secure your wallet:****

- During the wallet setup, you'll be asked to create a password and back up a recovery seed phrase. Make sure to store this recovery phrase in a safe and secure place. This is essential for wallet recovery in case you forget your password or your device is lost.

****9. Complete the setup:****

- Once your wallet is set up and secured, you can start using MetaMask to interact with Ethereum DApps and manage your cryptocurrency assets.

****10. Optional:**** Customize your settings, including network selection, gas fees, and additional security features within the MetaMask extension.

Remember to keep your wallet password and recovery seed phrase secure and never share it with anyone. Additionally, ensure your browser and MetaMask extension are always up to date for the latest security features and enhancements.

Please note that the installation steps might be slightly different depending on your specific browser version and updates to MetaMask, so always refer to the latest instructions from MetaMask's official website or browser extension store.

4.2 Non functional requirements

Non-functional requirements in the context of a blockchain project refer to aspects of the system that are not directly related to its primary functionality but are crucial for its overall success. These requirements focus on characteristics like performance, security, scalability, and usability. Here are some non-functional requirements commonly associated with blockchain projects:

1. ****Security:****

- ****Immutability:**** Data on the blockchain should be tamper-proof.
- ****Privacy:**** Depending on the use case, some data should be private and accessible only to authorized parties.
- ****Authentication and Authorization:**** Secure and robust methods of authenticating users and granting appropriate permissions.
- ****Consensus Algorithm Security:**** The chosen consensus algorithm should be resistant to attacks.

2. ****Performance:****

- **Transaction Throughput:** The ability to handle a high volume of transactions per second (TPS) while maintaining low latency.
- **Scalability:** The system should be able to scale as the user base and transaction volume grow.
- **Network Latency:** Minimizing network latency is critical for real-time applications.
- **Resource Efficiency:** Efficient use of computational resources to reduce operating costs.

3. **Scalability:**

- **Horizontal and Vertical Scaling:** The ability to add more nodes (horizontal scaling) or resources (vertical scaling) to accommodate growth.
- **Sharding:** If applicable, support for sharding to divide the network into smaller, manageable segments.
- **Interoperability:** Compatibility with other blockchain networks or systems.

4. **Reliability:**

- **High Availability:** The system should be available and accessible as close to 100% of the time as possible.
- **Disaster Recovery:** Measures to recover the system in the event of a failure or a disaster.

5. **Regulatory Compliance:**

- **Data Retention and Compliance:** Compliance with data retention and privacy regulations applicable to the project.
- **KYC/AML:** Implementing Know Your Customer (KYC) and Anti-Money Laundering (AML) procedures when required.

6. **Usability:**

- **User-Friendly Interface:** An intuitive and user-friendly interface for users and administrators.
- **Documentation:** Comprehensive documentation for users and developers.
- **Training and Support:** Provision of training and support for users and administrators.

7. **Interoperability:**

- **Compatibility:** Ability to integrate with other systems, networks, or blockchains.
- **Smart Contract Interoperability:** The capability to interact with smart contracts on different blockchains or platforms.

8. **Compliance:**

- **Legal and Regulatory Compliance:** Ensuring that the project complies with relevant laws and regulations in the jurisdictions it operates in.
- **Licensing:** Complying with open-source licensing requirements if applicable.

9. **Environmental Impact:**

- **Energy Efficiency:** Reducing the environmental impact by optimizing energy consumption, especially in Proof of Work (PoW) blockchains.

10. **Data Storage and Retention:**

- **Data Storage Costs:** Managing the long-term storage of data on the blockchain and associated costs.
- **Data Pruning:** Implementing data pruning strategies to manage blockchain bloat.

11. **Auditability and Transparency:**

- **Public Ledger:** Ensuring that transactions are transparent and can be audited by authorized parties.
- **Auditing Capabilities:** Supporting the auditing of transactions, contracts, and system performance.

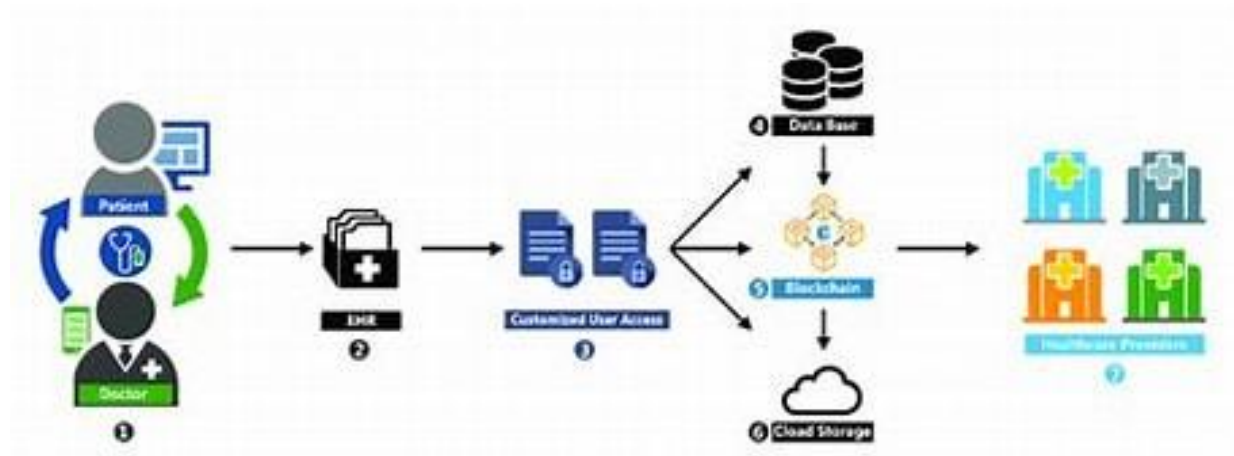
12. **Upgradability:**

- **Hard and Soft Forks:** Preparing for network upgrades through hard forks and soft forks, ensuring minimal disruptions.

Non-functional requirements are essential to the success of a blockchain project. The specific requirements will vary depending on the project's use case, technology stack, and intended audience. It's crucial to define and prioritize these requirements early in the project's planning and development stages.

5. Project design

5.1 data flow diagram



User stories

User stories are a way to describe the functionality and features of a system from the perspective of its end-users. In the context of electronic health records (EHR) systems, various stakeholders have different needs and use cases. Here are some user stories for different types of users who interact with EHR systems:

1. ****As a Patient:****

- As a patient, I want to be able to access my electronic health records online, so I can review my medical history and treatment plans.
- As a patient, I want to receive automated reminders for upcoming appointments and medication refills.
- As a patient, I want to request prescription refills and appointments online to save time and reduce administrative hassles.

2. ****As a Healthcare Provider:****

- As a healthcare provider, I want an EHR system that allows me to easily access and update patient records to provide the best possible care.

- As a healthcare provider, I want to receive real-time alerts and notifications for critical patient information, such as allergies and contraindications.

- As a healthcare provider, I want a user-friendly system for documenting patient encounters, including clinical notes, diagnoses, and treatment plans.

3. **As a Nurse:**

- As a nurse, I want the ability to scan patient wristbands and medications to ensure medication administration accuracy.

- As a nurse, I want a system that notifies me of any changes in a patient's condition, so I can respond promptly.

- As a nurse, I want to easily access a patient's care plan and medication history to provide consistent care.

4. **As a Healthcare Administrator:**

- As a healthcare administrator, I want to track and manage user access and permissions to ensure data security and privacy.

- As a healthcare administrator, I want a system that provides comprehensive reporting and analytics for decision-making and compliance purposes.

- As a healthcare administrator, I want the system to be scalable and support multiple healthcare facilities within our organization.

5. **As a Specialist Consultant:**

- As a specialist consultant, I want to have secure access to relevant patient records when consulting on cases from other facilities.

- As a specialist consultant, I want to provide feedback and recommendations within the EHR system to collaborate with the primary care team.

- As a specialist consultant, I want to view images and test results for in-depth analysis and diagnosis.

6. **As a Pharmacist:**

- As a pharmacist, I want to access patient medication histories to prevent potential drug interactions or duplications.

- As a pharmacist, I want to receive e-prescriptions from healthcare providers for accuracy and efficiency.

- As a pharmacist, I want to communicate with healthcare providers through the EHR system regarding patient medication concerns.

7. **As a Researcher:**

- As a researcher, I want access to de-identified patient data for medical research and studies.

- As a researcher, I want to track patient demographics, conditions, and treatments for epidemiological studies.

- As a researcher, I want to have the ability to export and analyze EHR data for research purposes.

These user stories represent a range of EHR system users and their specific needs. They serve as a basis for developing features and functionality that cater to the diverse requirements of stakeholders in the healthcare ecosystem. Each user story can be further refined and prioritized during the development process.

5.2 solution architecture

Designing a solution architecture for blockchain-based electronic health records (EHR) requires careful consideration of various factors, including data security, privacy, interoperability, and regulatory compliance. Here is a high-level solution architecture for a blockchain EHR system:

****1. Blockchain Network:****

- **Blockchain Type:** Choose an appropriate blockchain type, such as a permissioned blockchain (e.g., Hyperledger Fabric) or a public blockchain (e.g., Ethereum), depending on the specific use case and regulatory requirements.**
- **Consensus Mechanism:** Select a consensus mechanism that aligns with the desired level of decentralization and security. Options include Proof of Work (PoW), Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), and more.**

****2. Identity and Access Management:****

- **User Authentication:** Implement secure user authentication mechanisms, including multi-factor authentication (MFA) for healthcare providers, patients, and other stakeholders.**

- ****Identity Verification:**** Verify the identity of users, including healthcare providers, patients, and administrators, to ensure data integrity and compliance with regulations.

****3. Smart Contracts:****

- ****Smart Contract Development:**** Develop smart contracts to handle EHR-related transactions, such as record creation, access permissions, and updates. Smart contracts should be written securely and audited for vulnerabilities.
- ****Access Control:**** Use smart contracts to manage access control to EHR data, specifying who can view or update patient records.

****4. Data Storage and Encryption:****

- ****Off-Chain Data Storage:**** Store large EHR files (e.g., medical images) off-chain for scalability, using IPFS or similar technologies.
- ****On-Chain Metadata:**** Store metadata, such as record pointers and access permissions, on the blockchain.

- ****Data Encryption:**** Implement robust data encryption for both data at rest and data in transit to protect patient information.

****5. Interoperability:****

- ****FHIR Standards:**** Ensure interoperability by adhering to Fast Healthcare Interoperability Resources (FHIR) standards, enabling data exchange with other healthcare systems.

- ****APIs:**** Provide APIs to connect with legacy EHR systems, medical devices, and healthcare applications, allowing data sharing and integration.

****6. User Interfaces:****

- ****Patient Portal:**** Develop a user-friendly patient portal for patients to access and manage their EHR data, including appointment scheduling, prescription refills, and medical history review.

- ****Provider Dashboard:**** Create a comprehensive dashboard for healthcare providers to view patient records, update information, and communicate securely.

****7. Regulatory Compliance:****

- ****HIPAA Compliance:**** Ensure compliance with the Health Insurance Portability and Accountability Act (HIPAA) or other applicable data protection regulations, depending on the jurisdiction.
- ****Audit Trail:**** Implement an immutable audit trail to track all interactions with patient records for compliance and accountability.

****8. Privacy and Consent Management:****

- ****Patient Consent:**** Develop a system for patients to control who can access their EHR data and under what conditions.
- ****Data De-Identification:**** Implement data de-identification techniques to protect patient privacy while still enabling research and analytics.

****9. Data Ownership and Portability:****

- ****Data Ownership:**** Clearly define data ownership and control, allowing patients to take their EHR data with them when switching healthcare providers.

****10. Scalability and Performance:****

- ****Load Balancing:**** Implement load balancing to ensure the system can handle a high volume of transactions and data storage.
- ****Sharding:**** Consider sharding the blockchain to distribute data across multiple nodes for improved scalability.

****11. Disaster Recovery and Redundancy:****

- ****Data Backup:**** Regularly back up blockchain data to ensure data recovery in case of system failure or disaster.

****12. Monitoring and Analytics:****

- ****Real-Time Monitoring:**** Implement real-time monitoring and analytics to detect anomalies, suspicious activities, and performance issues.

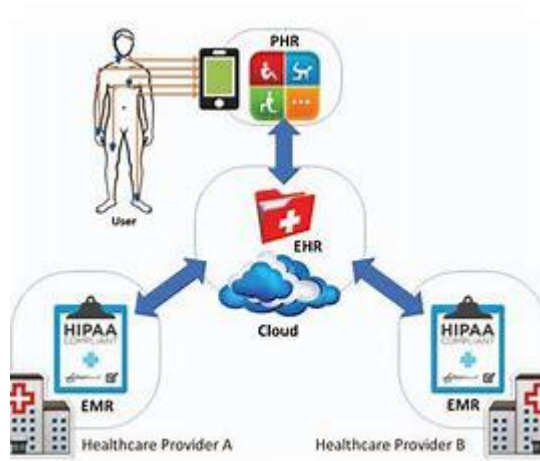
****13. Training and Support:****

- ****User Training:**** Provide training for healthcare providers, administrators, and patients to ensure they can effectively use the blockchain EHR system.

****14. Integration with External Systems:****

- ****Integration with Labs and Imaging Systems:**** Allow integration with external systems for seamless access to test results and medical images.

This solution architecture is a starting point for designing a blockchain-based EHR system. It should be adapted to meet the specific requirements of the healthcare organization and the regulatory environment in which it operates. Additionally, regular security audits and compliance assessments are essential to maintain the system's integrity and trustworthiness.



6. project planning and solution

6.1 Technical architecture

Designing a solution architecture for blockchain-based electronic health records (EHR) requires careful consideration of various factors, including data security, privacy, interoperability, and

regulatory compliance. Here is a high-level solution architecture for a blockchain EHR system:

****1. Blockchain Network:****

- ****Blockchain Type:**** Choose an appropriate blockchain type, such as a permissioned blockchain (e.g., Hyperledger Fabric) or a public blockchain (e.g., Ethereum), depending on the specific use case and regulatory requirements.
- ****Consensus Mechanism:**** Select a consensus mechanism that aligns with the desired level of decentralization and security. Options include Proof of Work (PoW), Proof of Stake (PoS), Practical Byzantine Fault Tolerance (PBFT), and more.

****2. Identity and Access Management:****

- ****User Authentication:**** Implement secure user authentication mechanisms, including multi-factor authentication (MFA) for healthcare providers, patients, and other stakeholders.
- ****Identity Verification:**** Verify the identity of users, including healthcare providers, patients, and administrators, to ensure data integrity and compliance with regulations.

****3. Smart Contracts:****

- ****Smart Contract Development:**** Develop smart contracts to handle EHR-related transactions, such as record creation, access permissions, and updates. Smart contracts should be written securely and audited for vulnerabilities.

- ****Access Control:**** Use smart contracts to manage access control to EHR data, specifying who can view or update patient records.

****4. Data Storage and Encryption:****

- ****Off-Chain Data Storage:**** Store large EHR files (e.g., medical images) off-chain for scalability, using IPFS or similar technologies.

- ****On-Chain Metadata:**** Store metadata, such as record pointers and access permissions, on the blockchain.

- ****Data Encryption:**** Implement robust data encryption for both data at rest and data in transit to protect patient information.

****5. Interoperability:****

- ****FHIR Standards:**** Ensure interoperability by adhering to Fast Healthcare Interoperability Resources (FHIR) standards, enabling data exchange with other healthcare systems.

- ****APIs:**** Provide APIs to connect with legacy EHR systems, medical devices, and healthcare applications, allowing data sharing and integration.

****6. User Interfaces:****

- ****Patient Portal:**** Develop a user-friendly patient portal for patients to access and manage their EHR data, including appointment scheduling, prescription refills, and medical history review.
- ****Provider Dashboard:**** Create a comprehensive dashboard for healthcare providers to view patient records, update information, and communicate securely.

****7. Regulatory Compliance:****

- ****HIPAA Compliance:**** Ensure compliance with the Health Insurance Portability and Accountability Act (HIPAA) or other applicable data protection regulations, depending on the jurisdiction.
- ****Audit Trail:**** Implement an immutable audit trail to track all interactions with patient records for compliance and accountability.

****8. Privacy and Consent Management:****

- ****Patient Consent:**** Develop a system for patients to control who can access their EHR data and under what conditions.

- ****Data De-Identification:**** Implement data de-identification techniques to protect patient privacy while still enabling research and analytics.

****9. Data Ownership and Portability:****

- ****Data Ownership:**** Clearly define data ownership and control, allowing patients to take their EHR data with them when switching healthcare providers.

****10. Scalability and Performance:****

- ****Load Balancing:**** Implement load balancing to ensure the system can handle a high volume of transactions and data storage.

- ****Sharding:**** Consider sharding the blockchain to distribute data across multiple nodes for improved scalability.

****11. Disaster Recovery and Redundancy:****

- ****Data Backup:**** Regularly back up blockchain data to ensure data recovery in case of system failure or disaster.

****12. Monitoring and Analytics:****

- ****Real-Time Monitoring:**** Implement real-time monitoring and analytics to detect anomalies, suspicious activities, and performance issues.

****13. Training and Support:****

- ****User Training:**** Provide training for healthcare providers, administrators, and patients to ensure they can effectively use the blockchain EHR system.

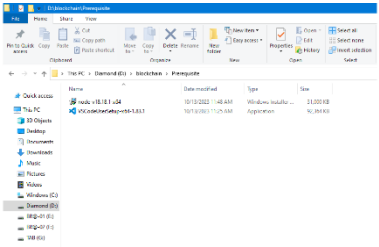
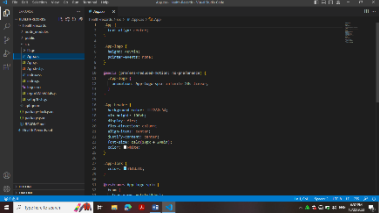
****14. Integration with External Systems:****

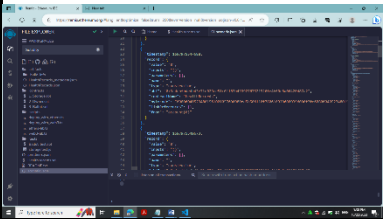
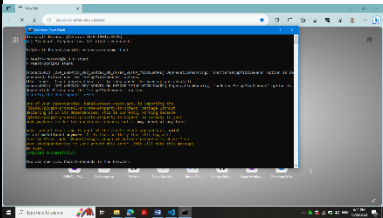
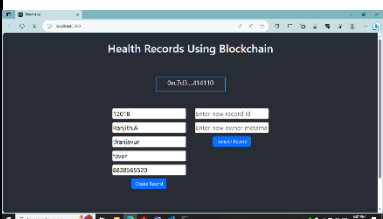
- ****Integration with Labs and Imaging Systems:**** Allow integration with external systems for seamless access to test results and medical images.

This solution architecture is a starting point for designing a blockchain-based EHR system. It should be adapted to meet the specific requirements of the healthcare organization and the regulatory environment in which it operates. Additionally, regular security audits and compliance assessments are essential to maintain the system's integrity and trustworthiness.

Model Performance Testing:

Project team shall fill the following information when working for blockchain.

S.No.	Parameter	Values	Screenshot
1.	Information gathering	Setup all the Prerequisite:	
2.	Extract the zip files	Open to vs code	

3.	Remix Ide platform exploring	<p>Deploy the smart contract code</p> <p>Deploy and run the transaction. By selecting the environment - inject the MetaMask.</p>	
4	Open file explorer	<p>Open the extracted file and click on the folder.</p> <p>Open src, and search for utilities.</p> <p>Open cmd enter commands</p> <ol style="list-style-type: none"> 1.npm install 2.npm bootstrap 3. npm start 	
5	{LOCALHOST ADDRESS IP	<p>copy the address and open it to chrome so you can see the front end of your project.</p>	

Results

The screenshot shows a web browser window with the title 'Health Records Using Blockchain'. The page has a dark blue background. At the top, there is a text input field containing the hexadecimal string '0e7d3...414110'. Below this, there are two columns of input fields. The left column contains five text inputs with the following values: '12018', 'Ranjith A', 'Chanjavur', 'Power', and '8838565520'. The right column contains two text inputs: 'Enter new record id' and 'Enter new owner meta', both of which are empty. Below the right column's inputs is a blue button labeled 'Create Record'. At the bottom of the left column, there is another blue button labeled 'Create Record'. The browser's address bar shows 'localhost:3000'. The Windows taskbar is visible at the bottom of the screen.

Health Records Using Blockchain

0e7d3...414110

12018

Ranjith A

Chanjavur

Power

8838565520

Create Record

Enter new record id

Enter new owner meta

Create Record

10. Advantages and disadvantages

Blockchain technology has several advantages and disadvantages. It's important to understand both sides to make informed decisions about its use. Here's a summary of some of the key advantages and disadvantages of blockchain technology:

****Advantages of Blockchain:****

1. ****Security:****

- ****Immutability:**** Once data is recorded on the blockchain, it is extremely difficult to alter, providing a high level of data integrity.
- ****Cryptography:**** Strong cryptographic techniques are used to secure transactions, making it highly resistant to fraud and unauthorized access.

2. ****Transparency:****

- ****Public Ledgers:**** Public blockchains are transparent and allow anyone to view transactions, promoting trust and accountability.
- ****Immutable History:**** Historical data is permanently recorded and can be audited, making it useful for various applications like supply chain and voting systems.

3. ****Decentralization:****

- ****Removal of Intermediaries:**** Blockchain can eliminate the need for intermediaries, reducing costs and increasing efficiency in various processes.
- ****Resilience:**** Distributed data is less vulnerable to single points of failure or attacks, making the system more resilient.

4. ****Trust and Consensus:****

- ****Decentralized Trust:**** Trust is established through consensus mechanisms, reducing the need for centralized authorities.
- ****Permissioned Systems:**** In private blockchains, participants are known and trusted, further enhancing trust.

5. ****Data Ownership and Control:****

- ****Personal Data Control:**** Users can have more control over their data and decide who can access it.

- **Data Portability:** Data can be easily transferred between applications or services.

6. **Efficiency and Speed:**

- **Quick Settlement:** Blockchain can facilitate faster settlement of transactions, particularly in financial services.
- **Reduced Intermediaries:** Eliminating intermediaries can streamline processes and reduce delays.

7. **Cost Reduction:**

- **Operational Efficiency:** Lowering costs by reducing the need for middlemen and streamlining operations.
- **Cross-Border Transactions:** Reducing fees associated with cross-border transactions in finance.

Disadvantages of Blockchain:

1. **Scalability:**

- **Network Congestion:** Public blockchains may suffer from slow transaction processing times and high fees during network congestion.
- **Scaling Challenges:** Scaling to accommodate a high volume of transactions can be technically challenging.

2. **Energy Consumption:**

- **Proof of Work (PoW):** PoW blockchains, like Bitcoin and Ethereum, consume substantial energy, which is an environmental concern.

3. **Lack of Regulation:**

- **Legal and Regulatory Challenges:** The evolving nature of blockchain technology presents legal and regulatory challenges, particularly in relation to cryptocurrencies.

4. **User Experience:**

- **Complexity:** Blockchain can be complex for non-technical users, making mass adoption a challenge.
- **Lost Private Keys:** Users risk losing access to their assets if they lose their private keys.

5. **Data Privacy Concerns:**

- **Public Ledgers:** Public blockchains may expose sensitive information, leading to concerns about data privacy.
- **Privacy Technology:** Privacy-enhancing technologies like zero-knowledge proofs are still evolving.

6. **Interoperability:**

- **Isolated Blockchains:** Different blockchains may not easily communicate or interoperate, limiting their potential for widespread use.

7. **Security Risks:**

- **Smart Contract Vulnerabilities:** Smart contracts can have coding vulnerabilities that, if exploited, may lead to financial losses.
- **51% Attacks:** In PoW blockchains, a malicious actor with majority computational power could compromise the network.

8. **Legal and Ethical Issues:**

- **Use in Illicit Activities:** Blockchain technology can be used for illegal activities, such as money laundering and black-market transactions.

The advantages and disadvantages of blockchain technology vary depending on the use case, blockchain type, and the specific implementation. It's important to carefully consider these factors when evaluating whether blockchain is the right solution for a particular application or organization.

11.Conclusion

In conclusion, blockchain technology offers a promising solution for the management of electronic health records (EHR) in the healthcare industry. It presents several key advantages, such as enhanced security, transparency, and data integrity. Blockchain can enable patients to have more control over their data while maintaining privacy and compliance with regulations like HIPAA.

The decentralized nature of blockchain reduces the need for intermediaries, potentially streamlining processes and reducing costs. It also provides a resilient and tamper-resistant platform for managing EHR data, which is vital for patient care and medical research.

However, blockchain technology is not without its challenges. Scalability and energy consumption, particularly in Proof of Work (PoW) blockchains, are concerns that need to be addressed. User adoption and the complexity of the technology pose obstacles, and privacy issues must be carefully managed.

Despite these challenges, the potential benefits of blockchain in electronic health records are significant, offering the possibility of secure, efficient, and patient-centric EHR management. The successful implementation of blockchain in healthcare will depend on ongoing technological advancements, regulatory frameworks, and the commitment of stakeholders to overcome the associated challenges. In the coming years, as blockchain continues to evolve and mature, it has the potential to revolutionize how EHR data is managed, leading to improved patient care and healthcare processes.

12. future scope

In conclusion, blockchain technology offers a promising solution for the management of electronic health records (EHR) in the healthcare industry. It presents several key advantages, such as enhanced security, transparency, and data integrity. Blockchain can enable patients to have more control over their data while maintaining privacy and compliance with regulations like HIPAA.

The decentralized nature of blockchain reduces the need for intermediaries, potentially streamlining processes and reducing costs. It also provides a resilient and tamper-resistant platform for managing EHR data, which is vital for patient care and medical research.

However, blockchain technology is not without its challenges. Scalability and energy consumption, particularly in Proof of Work (PoW) blockchains, are concerns that need to be addressed. User adoption and the complexity of the technology pose obstacles, and privacy issues must be carefully managed.

Despite these challenges, the potential benefits of blockchain in electronic health records are significant, offering the possibility of secure, efficient, and patient-centric EHR management. The successful implementation of blockchain in healthcare will depend on ongoing technological advancements, regulatory frameworks, and the commitment of stakeholders to overcome the associated challenges. In the coming years, as blockchain continues to evolve and mature, it has the potential to revolutionize how EHR data is managed, leading to improved patient care and healthcare processes.

13. Appendix

Source code

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract HealthRecords {

    struct PatientRecord {
        string Name;
        address patientAddress;
        string diseases;
        string contactInfo;
    }

    mapping(uint256 => PatientRecord) public records;

    event RecordCreated(uint256 indexed recordId, address indexed
patientAddress);
    event RecordTransferred(
        uint256 indexed recordId,
        address indexed from,
        address indexed to
    );

    modifier onlyOwner(uint256 recordId) {
        require(msg.sender == records[recordId].patientAddress, "Only contract
owner can call this");
        _;
    }

    function createRecord(
        uint256 recordId,
        string memory name, address _patientAddress, string memory _diseases,
string memory _contactInfo
    ) external {

        records[recordId].Name = name;
        records[recordId].patientAddress = _patientAddress;
        records[recordId].diseases = _diseases;
        records[recordId].contactInfo = _contactInfo;

        emit RecordCreated(recordId, _patientAddress);
    }
}
```

```

    function transferRecord(uint256 recordId, address newOwner) external
onlyOwner(recordId) {

        //require(records[recordId].patientAddress == newOwner, "New Owner
should have different Address");

        require(records[recordId].patientAddress == msg.sender, "Only record
owner can transfer");

        records[recordId].patientAddress = newOwner;

        emit RecordTransferred(recordId, records[recordId].patientAddress,
newOwner);
    }

    function getRecordData(
        uint256 recordId
    ) external view returns (string memory, address, string memory, string
memory) {
        return (records[recordId].Name,
records[recordId].patientAddress,
records[recordId].dieses,
records[recordId].contactInfo);
    }

    function getRecordOwner(uint256 recordId) external view returns (address)
{
        return records[recordId].patientAddress;
    }
}

```