Final Technical Report

On the

Project: PATIENT DATA MANAGEMENT SYSTEM USING BLOCKCHAIN

Submitted by,

BHAVIRISETTI CHANDRASEKHAR	– 16344254
BATTULA RASI HARSHINI	– 16341280
BOLLINENI SANDEEP SAI	– 16345740
MUKTA HARINI	- 16343090

Under the supervision of

A S M Touhidul Hasan, Ph.D

TABLE OF CONTENTS:

1.Abstract:

In this Project, we provide an abstract outline regarding the "Patient Data Management System Using Blockchain" initiative, which seeks to revolutionize medical data management & automate the process of filing insurance claims. Using blockchain technology, this initiative seeks to solve the fundamental issue of data security and trust in healthcare organizations. Insights into the project's goals, technological architecture, potential market impact, as well as advantages to patients, healthcare providers, as well as insurance companies are provided in this project. The suggested system provides a safe, transparent, and tamper-proof environment for managing patient data as well as medical bills, which is especially important considering recent data breaches and the prevalence of inaccurate medical records. The project guarantees the integrity & accessibility of vital healthcare data by utilizing blockchain's fundamental characteristics, including immutability and decentralized consensus. The project illustrates its dedication to transforming the healthcare landscape through collaboration, efficient governance, and a strong value offer. The advantages of this blockchain-based system over more conventional and different approaches are also highlighted in the paper.

2.Problem Statement:

Management of patient data as well as medical bills is at the core of a complicated problem facing the healthcare sector. Two major problems of current systems need to be addressed.

1. In the healthcare industry, trust is crucial. Nevertheless, conventional data management systems sometimes lack the honesty and security necessary to build and keep stakeholder trust. There must be complete trust in the reliability of health care records and billing among doctors, patients, and insurance.
2. Patients, hospitals, medical departments, as well as insurance companies are just a few of those involved in the healthcare system. Inefficiencies, misconceptions, and delays arise when trying to coordinate the sharing of patient data and billing between various organizations. Insurance claims are not processed smoothly due to a lack of communication, which can cause frustration and financial problems.

3.Solution:

The development of a patient data management system based on blockchain is the innovative approach that has been put up. The core of this system is blockchain technology, which is recognized for its transparency, security, & immutability. Here is how it addresses the problems that were identified:

- Data cannot be changed or removed after it has been recorded according to blockchain technology. Bills and patient data are kept in an immutable ledger that is only accessible to those with permission. Assuring stakeholders that the data they rely on is correct and unaffected, transparency as well as data integrity contribute to encouraging confidence among stakeholders.
- A shared, decentralized platform to manage patient data as well as bills is established via the blockchain-based system. The delays and miscommunication caused by traditional methods are gone because of to smooth access and updating of records by hospital departments, medical departments, including insurance companies. All those involved benefit from the streamlined cooperation that speeds up the insurance claim procedure.

4.Planning and Research:

The project aims to implement a secure and privacy-focused system for storing and retrieving sensitive data, such as patient records, using blockchain technology. The key components involve client-side encryption, a blockchain for immutable storage, and a third-party system for managing access keys and permissions. The scope includes developing and integrating encryption algorithms, blockchain protocols, and a centralized system for secure key management.

4.1. Scope of the Project:

The first stage in any market analysis should be determining the scope of the problem. It involves examining the prospective impact of the project based on:

a. Potential Revenue: Estimating how much money can be made with the blockchain. Making use of hospitals, medical departments, insurance companies, among others generate income through better data management and claims processing, which contributes to the overall.

b. Potential Users: Patients, healthcare professionals, insurance companies, and administrators should all be on the list of people who will be using the system. Learning about the user base is essential for designing functionality and interfaces that meet their needs.

c. potential Spending: Estimating how much cost healthcare organizations could have to spend to implement blockchain technology. The total includes everything from implementing the system to providing ongoing support.

4.2Timeline:

Quarterly Plan (6wks.) – Business

- Market research and product development
- Pilot testing and feedback
- Expansion of partnerships
- Full-scale deployment

Quarterly Plan (6 wks.) – Technical

- Development of smart contract functionality
- Data encryption enhancements
- Scalability improvements
- Security audits.

Quarterly Plan (6 wks.) – Financial

- Seed funding secured
- Initial product launch,
- Series A funding
- Revenue generation begins.

4.**3. Resource It will be required:**

a. Development Team:

Blockchain Developers

Encryption Specialists

Database Developers

System Architects

Quality Assurance/Testers

b. Infrastructure:

Servers for hosting blockchain nodes and the third-party system

Security measures (firewalls, encryption protocols, etc.)

c. Technology Stack:

Blockchain Platform (e.g., Ethereum, Hyperledger)

Encryption Libraries (e.g., OpenSSL, CryptoJS)

Database System (e.g., MongoDB, PostgreSQL)

Web/Application Framework (e.g., React, Node.js)

d. Estimated Cost

Amount Requested in Dollar: $3 million.

Industry Focus: Healthcare and blockchain technology.

Venture Stage: Early-stage.

Timeline to Achieve Profitability: Within 2 years.

Monthly Burn Rate: $150,000

Blockchain, originating in 2008 from an entity known as Satoshi Nakamoto, initially gained prominence for its association with cryptocurrency, particularly Bitcoin. However, its applications extend beyond finance. This exploration delves into the potential use of blockchain in managing electronic medical records, offering insights into its efficiency during emergencies. Much existing literature predates blockchain, focusing on traditional software frameworks and pre-blockchain techniques. Blockchain's introduction, notably through Ethereum, revolutionized data representation with its Turing-complete language, ushering in a new era of distributed and peer-to-peer communication.

In healthcare, blockchain offers promise for secure, transparent, and efficient management of patient records. Distinctions between Electronic Health Records (EHR) and Electronic Medical Records (EMR) are critical. While EMRs focus on individual healthcare providers' data, EHRs encompass a broader spectrum of health-related information.

A critical aspect explored is the integration of blockchain with patient record systems. The pioneering use involves a modular approach, storing actual records off-chain for scalability.

Smart contracts, managing interactions and access rules, play a pivotal role. However, challenges like every participant maintaining a data copy and scalability concerns arise due to the consensus mechanism.

Projects concentrated on data-sharing, access controls, and integration mechanisms. Patient-side security during data aggregation was also a focal point. Frameworks can be categorized as permissioned and permissionless, with the former ideal for organizations familiar with each other. Unlike cryptocurrency-centric models, these systems are incentivized differently, making them suitable for secure information transfer among trusted entities.

5.Feasibility Analysis:

Feasibility analysis for a Patient Data Management System involves assessing the practicality and viability of implementing a secure and efficient healthcare data solution. This comprehensive examination considers technical, financial, and operational aspects. Technically, the analysis explores the integration of blockchain technology, encryption, and decentralized data storage. Resource assessment evaluates the availability of skilled personnel, necessary infrastructure, and potential training requirements. Financial considerations delve into the costs associated with development, maintenance, and potential upgrades. The timeline and schedule assessment establish realistic project timelines, considering dependencies and regulatory approvals. Stakeholder alignment ensures that the system meets the expectations of healthcare professionals, administrators, and patients. The ultimate goal is to make informed decisions regarding the successful and responsible implementation of a Patient Data Management System.

5.1. Project Purpose and Goals:

The primary purpose of implementing a Patient Data Management System using blockchain is to revolutionize the secure storage, accessibility, and integrity of patient data in the healthcare ecosystem. Goals include enhancing data security, ensuring privacy compliance, streamlining data sharing among authorized parties, and ultimately improving patient care and outcomes.

5.2. Technical Exploration:

Technical exploration involves a detailed examination of the technologies and protocols associated with blockchain. This includes understanding how the system will leverage decentralized and distributed ledger technologies, implement smart contracts for data access control, and integrate encryption mechanisms for enhanced security.

5.3. Resource Assessment:

It depends on the availability and proficiency of resources required for implementing and maintaining the blockchain-based Patient Data Management System. This includes assessing the expertise of IT professionals in blockchain development, ensuring access to necessary hardware infrastructure, and determining if additional training is needed.

5.4. Tools and Technologies:

Identify and assess the tools and technologies essential for implementing the blockchain solution. This includes selecting a suitable blockchain platform (e.g., Ethereum, Hyperledger), encryption libraries for data security, and integration tools for connecting the blockchain system with existing healthcare IT infrastructure.

5.5. Risk Identification:

Anticipate potential risks and challenges associated with implementing a blockchain-based system. Risks may include data breaches, interoperability issues, regulatory compliance challenges, and resistance to technology adoption. Develop strategies to mitigate these risks, such as implementing robust security measures and conducting thorough compliance checks.

5.6. Financial Considerations:

The financial aspects of developing and maintaining a blockchain-based Patient Data Management System. Consider the costs associated with blockchain technology implementation, software development, hardware infrastructure, ongoing maintenance, and potential training programs for staff.

5.7. Timeline and Schedule:

Assess the proposed timeline for the project, considering the complexities of blockchain development, regulatory approvals, and integration with existing healthcare systems. Establish realistic schedules and milestones, accounting for potential delays and dependencies on external factors.

5.8. Stakeholder Alignment:

Ensure alignment with the expectations and needs of stakeholders in the healthcare ecosystem. Engage with healthcare professionals, administrators, and patients to understand their requirements and concerns. Align system goals with the broader objectives of improving patient care, data security, and overall healthcare efficiency.

5.9. Decision-Making:

Based on the comprehensive analysis, make informed decisions regarding the feasibility of implementing a blockchain-based Patient Data Management System. Consider factors such as technical viability, resource availability, financial feasibility, and alignment with stakeholder expectations when deciding whether to proceed with the development.

In summary, by conducting a feasibility analysis for a Patient Data Management System using blockchain involves thorough examination and evaluation of technical, financial, and stakeholder-related aspects to ensure the successful and impactful implementation of a secure and efficient healthcare data management solution.

6.System Requirements and Specifications:

The System Requirements Specification (SRS) stands as a pivotal document, serving as the cornerstone for the entire software development process. It not only outlines the essential needs of a system but also provides a detailed description of its key features. Essentially, an SRS captures an organization's written interpretation of a client's or potential customer's system requirements and expectations at a specific moment, typically before any actual design or development work commences. It functions as a mutually beneficial communication tool, ensuring a shared understanding between the client and the organization regarding each other's requirements at a particular juncture. This two-way safeguarding approach is vital for fostering clear communication and alignment throughout the development lifecycle.

6.1.Specific Requirement:

Hardware Specification:

Processor Type: Intel Core™ – i7

Speed: 2.4 GHz

RAM: 16 GB RAM

Hard Disk: 80 GB HDD

Software Specification:

Operating System: Windows 64-bit

Technology: JAVA

IDE: Eclipse

Tools: Visual Studio.

Frond and back end Frameworks: NodeJS, Reactjs, web3js

Blockchain Development Tools (Ethereum):

MetaMask

Solidity

Truffle

Ganache

These specific hardware and software requirements provide a detailed outline for the system's infrastructure. The specified processor, RAM, and hard disk details define the necessary hardware components, while the software requirements encompass the operating system, technology stack, integrated development environment (IDE), and specific tools. Adhering to these specifications ensures the system's optimal performance and compatibility with the designated technology stack.

6.2Functional Requirements:

1. User-Centric Web Application:

- Develop a user-friendly web application accommodating three distinct roles: data owner, doctor, and admin.

2. Medical Record Uploading:

- Enable data owners to seamlessly upload their medical records onto the platform.

3. Blockchain Integration:

- Implement a robust system to convert medical records into blocks, storing them securely within a distributed environment

4. Ethereum Blockchain Storage:

- Utilize Ethereum blockchain technology to store and manage medical data securely, ensuring tamper-proof and transparent record-keeping.

5. Doctor Authentication and Record Selection:

- Establish a secure login mechanism for doctors to access the application, allowing them to efficiently select and retrieve patients' medical records.

6. KNN Algorithm Implementation:

- Integrate the K-nearest neighbors (KNN) algorithm, enhancing the efficiency of access control for medical data. This algorithm intelligently determines the proximity of entities, contributing to robust data security.

7. Efficient Secure Data Access Control:

- Design the application to seamlessly provide efficient and secure data access control, ensuring that authorized individuals, specifically doctors, have exclusive access to patients' medical records. This emphasis on secure access enhances overall data privacy and confidentiality.

These functional requirements collectively define the system's capabilities, emphasizing user-centricity, secure data handling, and the integration of advanced technologies for enhanced healthcare data management.

6.3Non-Functional Requirements:

1.Portability and Self-Containment:

- The program must be self-contained, allowing easy transfer between computers. While network connections are assumed, the system should function seamlessly on various computers.

2. Capacity, Scalability, and Availability:

- The system must achieve 100% availability at all times. It should be scalable to support additional clients and volunteers, ensuring efficient performance as the user base expands.

3. Maintainability and Optimization:

- Emphasize maintainability by optimizing the system for supportability. This includes adhering to coding standards, naming conventions, class libraries, and abstraction to facilitate ease of maintenance

4. Randomness, Verifiability, and Load Balancing:

- The system should incorporate randomness for checking nodes and ensuring a level of unpredictability. It should be designed with verifiability in mind, allowing stakeholders to verify system processes. Additionally, the system should implement load balancing to distribute computational tasks evenly, optimizing performance and preventing resource bottlenecks.

These non-functional requirements establish the operational constraints and expectations for the healthcare data management system, addressing aspects such as portability, scalability, maintainability, and system optimization.

7.Design and Prototyping:

Prior to the introduction of Blockchain and Smart Contracts came into play, the predominant discourse in Electronic Health Records (EHR) revolved around the practical dilemma of "Where and how to store EMRs?" This deliberation often pitted the advantages of cloud-based platforms against the utilization of localized systems. The outcome of this deliberation typically leaned towards centralization, signifying that each healthcare provider and hospital took on the responsibility of housing all patient records within their own premises, relying on locally managed storage and databases.

However, the centralized model presented notable challenges:

1.Lack of Patient Control:

Patients lacked control over their data, as it was owned and managed by healthcare providers. The absence of patient ownership hindered data security and privacy.

Shifting towards a patient-centric model could disrupt the centralized storage of sensitive healthcare data, enhancing patient control.

2. Scattering of Records:

Diverse treatment structures led to the scattering of patient records across various systems, potentially causing duplication.

A more cohesive approach is needed to streamline record-keeping and prevent unnecessary replication.

3. Interoperability Challenges:

Different hospitals and healthcare providers employed varied systems, hindering seamless data sharing and viewing.

Addressing interoperability issues is crucial to facilitate efficient data exchange, especially when patients transition between different healthcare providers.

4. Inconvenient Secure Sharing:

Traditional methods of sharing healthcare data, such as using secure email standards like Direct, were deemed complicated and time-consuming.

Exploring more user-friendly and efficient ways of secure data sharing is essential for enhancing communication within the healthcare ecosystem.

By recognizing these challenges, the evolution towards Blockchain and its Smart Contract capabilities seeks to overcome these limitations, offering a decentralized, patient-driven approach to Electronic Health Records that addresses issues of ownership, scattering, interoperability, and secure sharing.

The attempted solutions to the early challenges indeed addressed some issues, but they weren't without vulnerabilities. This prompted a renewed exploration of centralized approaches, yet many drawbacks like privacy concerns, data ownership issues, and transparency limitations persisted. In disaster scenarios, the centralized model revealed weaknesses, as its response tended to be disorganized, and any damage to the central storage could compromise crucial data.

While natural calamities are infrequent, their potential impact on healthcare, through data replication and sharing, underscores the need for a robust network. Peer-to-peer networks emerged as a solution, allowing data ownership by specific system-nodes. A multitude of such nodes enhances accessibility. However, achieving consensus while maintaining privacy, security, and anonymity poses considerable challenges.

Blockchain technology has emerged as a transformative solution, addressing these challenges and enhancing transparency and reliability. Blockchain structures data in a linked list, forming an immutable chain where breaching the system requires altering the entire chain with consensus, a nearly insurmountable task. This revolutionary approach significantly improves the resilience and security of healthcare data management.

7.1Ethereum:

Ethereum stands as the second-largest cryptocurrency platform globally, trailing only the Bitcoin network. This open-source and decentralized platform serves as a foundational infrastructure for the creation and execution of smart contracts. Miners within the Ethereum network are incentivized with ether, the native cryptocurrency, as a reward for validating transactions. Presently, Ethereum supports a vast ecosystem, hosting numerous cryptocurrencies, with three of them ranking among the top globally.

The Ethereum network incorporates an Ethereum Virtual Machine (EVM), a global network of nodes that execute scripts. It operates on an internal transaction unit known as "gas," which is essential for allocating resources on the network. Vitalik Buterin, a prominent cryptocurrency researcher, developed the Ethereum platform. In 2016, Ethereum underwent a significant event when an exploitation in the smart contract of the DAO (Decentralized Autonomous Organization) project resulted in a theft of millions of dollars. This incident led to a split in the Ethereum blockchain, creating two separate entities: ETH and ETC. The theft was reversed on the ETH blockchain, while the ETC blockchain continued its course.

Ethereum employs two major consensus algorithms: Proof of Work (PoW) and Proof of Stake (PoS). PoW involves miners solving complex mathematical problems to validate transactions, while PoS assesses a participant's stake in the cryptocurrency to determine their role in validating transactions. Although PoS introduces the risk of monopoly, Ethereum mitigates this through a mechanism that randomly selects stakeholders in successive rounds, fostering a more balanced and secure network.

7.2Web3:

Efficient communication within the Ethereum network relies on the validation of transactions within the chain. To enable participants in an offline framework to create and validate transactions, a relay through the peer-to-peer (p2p) network becomes essential. The underlying network hosts a library collection that facilitates communication between Ethereum nodes and in-chain components. This library is specifically designed for server-side applications developed in Node.js.

Connecting to the Ethereum network is accomplished through an Ethereum node utilizing an HTTP connection. This node can either be locally provided by Infura or an HD wallet. The integration of Ethereum with web applications is streamlined through tools like Metamask—an in-browser extension. Metamask serves as an Ethereum wallet within the browser, introducing a Web3 provider object to the browser. A Web3 provider establishes a connection to publicly accessible Ethereum nodes, serving as a crucial data structure.

Metamask further allows users to operate from Ethereum accounts, manage public and private keys, and store unique account information. This integration, complemented by web3js and a web interface, simplifies the communication between the backend and frontend components. The combination of Ethereum, Metamask, and web3js forms a cohesive framework that enhances the user experience and facilitates secure communication within the Ethereum network.

7.3Ganache:

Ganache is a popular and powerful tool in the realm of blockchain development, particularly for Ethereum-based projects. It serves as a personal blockchain emulator, offering a local and private Ethereum blockchain environment for testing, development, and debugging. Here are some key aspects of Ganache's utility:

1. Local Blockchain Environment:

- Ganache allows developers to set up a local and private Ethereum blockchain on their machines. This local blockchain is isolated from the main Ethereum network, providing a controlled environment for testing without the need for real cryptocurrency transactions.

2. Testing Smart Contracts:

- Developers can deploy and test their smart contracts on the Ganache blockchain. This facilitates a rapid and iterative development process, enabling developers to quickly identify and fix issues in their smart contracts before deploying them on the actual Ethereum network.

3. Fast and Deterministic Blockchain:

- Ganache provides a fast and deterministic blockchain, meaning that the state of the blockchain is consistent and predictable. This determinism is crucial for ensuring that tests and experiments produce consistent results, enhancing the reliability of the development process.

4. Mock Accounts with Ether:

- Ganache comes pre-loaded with a set of mock accounts, each having a balance of ether. Developers can use these accounts for testing various functionalities, such as sending transactions, interacting with smart contracts, and observing how changes in the blockchain impact account balances.

5. User-Friendly Interface:

- Ganache offers a user-friendly graphical interface that allows developers to visualize the blockchain state, inspect transactions, and monitor events. This ease of use makes it accessible to both experienced blockchain developers and those who are new to the technology.

6. Integration with Development Tools:

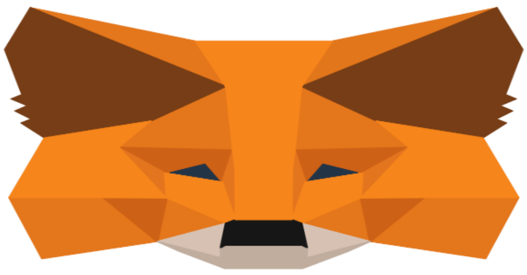
- Ganache seamlessly integrates with popular development tools and frameworks such as Truffle and Remix. This makes it a versatile choice for developers who prefer different tools for their blockchain development workflow.

7. Gas Price and Limit Configuration: - Developers can configure gas prices and transaction limits on Ganache to simulate various scenarios. This helps in understanding how

transactions behave under different gas conditions, providing valuable insights into real-world deployment scenarios.

In summary, Ganache is a valuable tool that significantly simplifies the development and testing processes for blockchain applications, especially those built on the Ethereum platform. Its features make it an essential component in the toolkit of blockchain developers, offering a reliable and efficient environment for building and testing decentralized applications.

7.4MetaMask:



Metamask, a pivotal browser extension in the blockchain realm, serves as an Ethereum wallet seamlessly integrated into popular browsers like Chrome and Firefox. As a Web3 provider, it enables decentralized applications (DApps) to interact with the Ethereum blockchain directly from the browser. Users can manage Ethereum accounts, initiate transactions, and switch between various networks, including testnets. Metamask simplifies token management, supporting popular Ethereum-based tokens. Its secure key management encrypts private keys locally, and users confirm transactions within the browser, ensuring a smooth and secure experience. Open source and backed by a vibrant community, Metamask plays a crucial role in fostering user-friendly and secure interactions with decentralized applications and blockchain assets, contributing significantly to the Ethereum ecosystem's accessibility and adoption.

7.5.Truffle:

Truffle stands as a versatile Ethereum development environment, functioning as both a blockchain asset pipeline and an extensive testing network for the Ethereum Virtual Machine (EVM). Streamlining the smart contract development lifecycle, Truffle handles tasks like compilation, linking, and binary dependency management, offering a seamless coding experience. Automation is a key feature, providing a robust testing environment to enhance the reliability of Ethereum-based applications by identifying and addressing potential issues.

Truffle introduces a scriptable and extensible framework for deploying and migrating smart contracts, offering flexibility for managing complex applications on the Ethereum network. With efficient package management, direct contract communication, configurable build pipelines, and a script execution environment, Truffle serves as a comprehensive toolkit. It empowers Ethereum developers by simplifying development processes, ensuring reliable testing, and providing flexibility in deployment, package management, and script execution.

7.6.Smart Contracts:

The integration of smart contracts in the Medicare project brings a patient-centric approach, empowering individuals to have control over their medical records. The smart contract ensures that patients make critical decisions regarding who can access and modify their health records. Five key operations on the patient's end define this control: Permissioning and Revoking of View and Write Permissions for medical practitioners.

When medical practitioners add record details, the smart contract ensures the secure storage of this information in the blockchain, serving as irrefutable proof of the data's existence. Crucially, only validated practitioners possess the authority to add recorded data; other network participants are restricted from this operation.

The smart contract serves as the backbone of access control functionality, emphasizing that only patients have the authority to determine who can access their health records. This patient-centric model enhances privacy, security, and transparency in healthcare data management, aligning with the broader goals of the Medicare project.

The architecture of the "Patient Data Management System Using Blockchain" project is built on the seamless integration of blockchain with modern database and framework technologies.

8.Architecture:

8.1.Blockchain Platform Selection:

The core of our project's design is MongoDB, a powerful and versatile database management system:

Database Technology: NoSQL database MongoDB holds information in documents similar to JSON. Our project requires constantly changing healthcare data, and the platform's adaptability in this area is ideal.

8.2.Framework and Middleware Technologies:

Our core technology is built on the powerful server-side environment and web application framework that is Node.js, Express.js, and Mongoose.

a. Node.js allows us to execute JavaScript on the server, a crucial part of developing web applications, because it serves as a server-side JavaScript runtime environment.
b. We use Express.js, a Node.js web application framework, because of its extensive feature set, middleware for routing, and request handling skills.

8.3.Data Management Excellence:

The success of our initiative depends on our ability to effectively manage data while protecting the privacy and security of patient data.

a. On-Chain Data: Due to its secure cryptography and immutability, the blockchain is used to store sensitive patient data, medical records, including billing information.
b. Off-Chain Data: We use MongoDB for off-chain storage to solve the scalability problem while maintaining data accessible. When on-chain and off-chain data management are used together, efficiency and legality are both improved.

8.4.Data Encryption: To ensure conformance with data security standards like HIPAA and GDPR, our design makes use of strong encryption techniques to protect patient data and billing information.

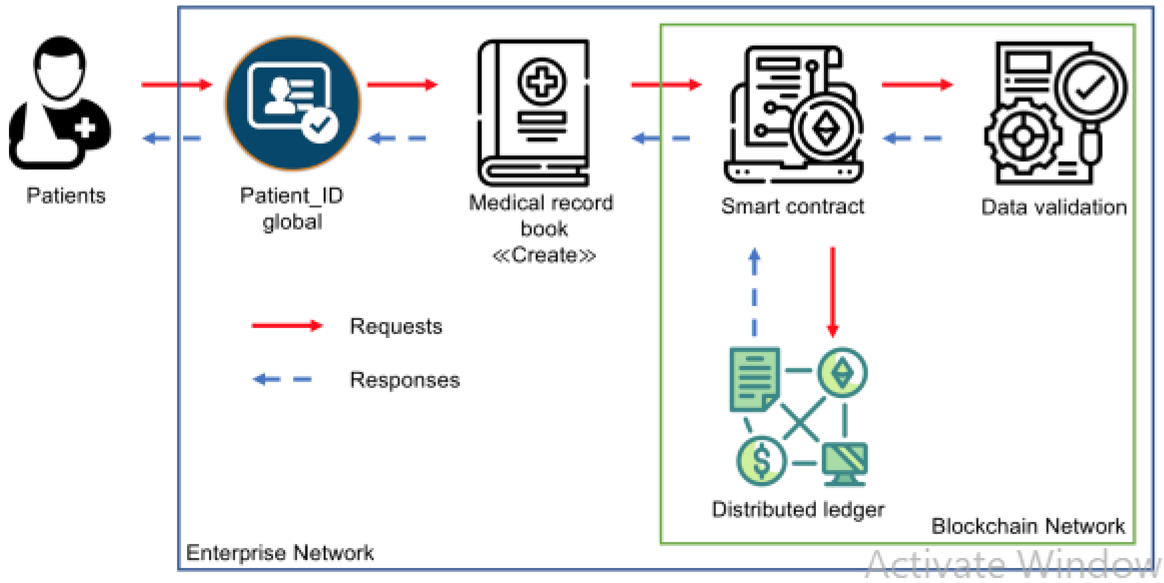
8.5.Protecting Patient Rights:

The protection of patient privacy and rights is fundamental to our design:

a. We use cutting-edge digital identification solutions to ensure the privacy of our patients' sensitive data through strong authentication and encryption.
b. The GDPR, HIPAA, and other industry-specific criteria are all met by our design. It has protections in place to ensure patients' permission and limit unauthorized access to their information.

8.6.System Architecture Diagrams:

System architecture diagrams vividly depict our architecture to improve understanding.



9.Development:

Once the groundwork in the planning and research phase is complete, the project transitions to the development stage. This pivotal phase is where developers actively engage in programming, transforming the conceptualized software into a tangible and functional product. The development stage is typically the most time-intensive, as it involves crafting a nearly finalized version of the software.

In the continuum of development, the software undergoes several key stages:

1. Pre-Alpha:

- In this initial stage, developers work on foundational coding, creating the software's core structure.

2. Alpha:

- The software progresses to the alpha stage, where basic functionalities are implemented for internal testing.

3. Beta:

- During the beta phase, the software is expanded to incorporate additional features and undergoes more extensive testing.

4. Release Candidate:

- The release candidate marks a near-final version, ready for comprehensive testing and potential deployment. It represents a refined and stable iteration poised for release.

Each stage involves iterative development, testing, and refinement, gradually shaping the software into a robust and functional solution.

Encryption, Hashing, and Storage Process:

9.1.Client's Key Generation:

The client initiates the process by creating a key using the data as key-text through the SHA-256 cryptographic hashing function.

9.2.Local Encryption:

The generated key encrypts the data locally using the AES encryption algorithm.

9.3.Third-Party System Encryption:

The encrypted data is sent to the Third-Party System, where it undergoes another layer of encryption. The encrypted data serves as key-text for SHA-256, generating a key used to encrypt the locally encrypted data. This key is stored in the server's database with the owner's data ID.

9.4.Blockchain Transaction:

The encrypted data from the Third-Party System, along with the locally generated key, is sent back to the client. The client then sends this encrypted data to the blockchain, where it is written into a block. This transaction is stored in the blockchain's database, ensuring immutability.

9.5.Fetching Encrypted Data from Blockchain:

Data Retrieval:

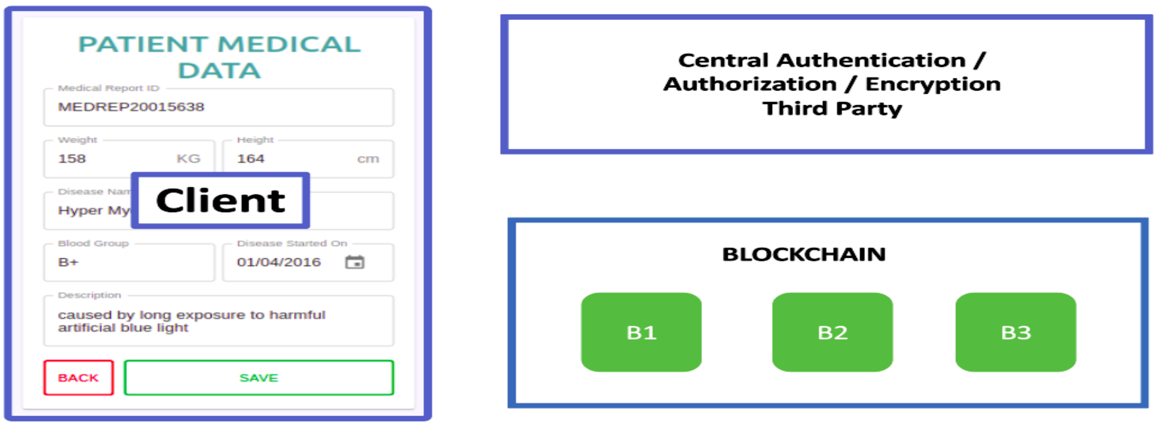
The encrypted data and the locally generated key stored in the blockchain are fetched.

Third-Party Decryption:

The encrypted data is sent to the Third-Party System, which decrypts the data using the key stored in its database with the user ID. Decryption occurs only if the data owner grants permission to the requesting client. The decrypted data is then sent to the client.

Final Decryption:

The decrypted data from the Third-Party System is further decrypted to the original data using the key fetched from the blockchain. The final, original data is then presented to the client for access and viewing.



PATIENT MEDICAL DATA

Medical Report ID: MEDREP20015638	
Weight: 158 KG	Height: 164 cm
Disease Nam: Hyper My	
Blood Group: B+	Disease Started On: 01/04/2016
Description: caused by long exposure to harmful artificial blue light	

BACK
SAVE
Client
Central Authentication / Authorization / Encryption Third Party
BLOCKCHAIN
B1
B2
B3

Fig: Three major components of our application's architecture to ensure privacy in blockchain

Blockchain:

A blockchain is an interconnected and secure ledger of records, or blocks, each containing a hash of the previous block, a timestamp, and transaction data. The structure ensures data integrity by making it resistant to retroactive modifications. The chain reinforces itself with each new block, forming a tamper-resistant sequence.

Smart Contracts:

Smart contracts are self-executing protocols designed to automatically execute, control, or document events and actions according to contract terms. They aim to reduce the reliance on trusted intermediaries, lower enforcement costs, and minimize fraudulent activities.

Ethereum:

Ethereum is an open-source blockchain with smart contract functionality and its native cryptocurrency, Ether (ETH). As the second-largest cryptocurrency after Bitcoin, Ethereum facilitates decentralized applications and transactions.

Third-Party System:

A centralized off-chain system for storing keys, the third-party system encrypts data using Advanced Encryption Standard (AES) and generates keys using the SHA-256 cryptographic hash function

Advanced Encryption Standard (AES):

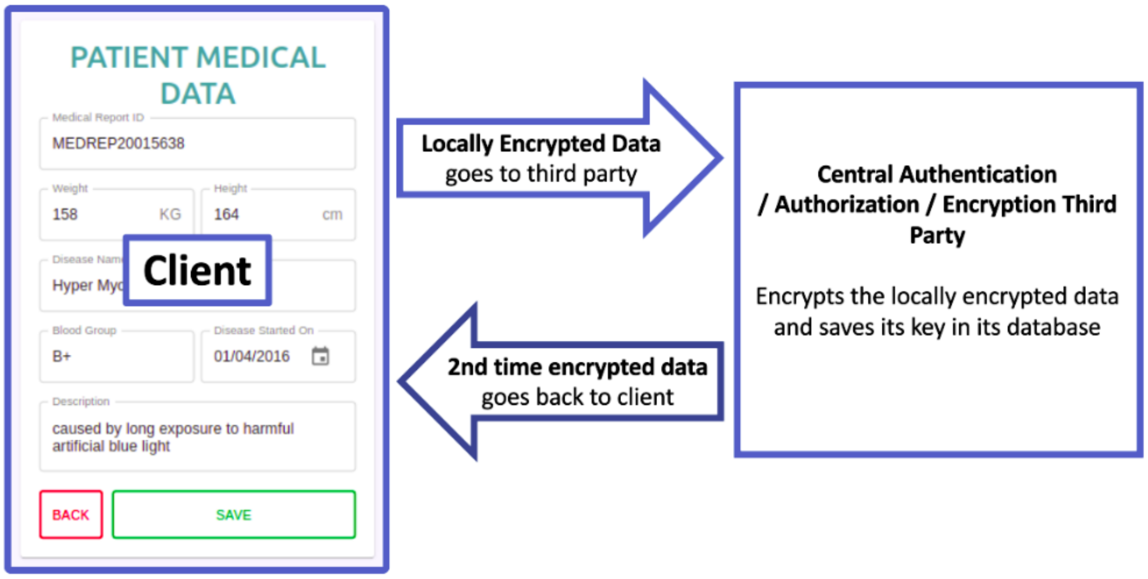
AES, also known as the Rijndael algorithm, is a symmetrical block cipher converting 128-bit plain text into ciphertext using keys of 128, 192, or 256 bits. Its widespread adoption attests to its security.

SHA-256:

SHA-256 is a cryptographic hash function generating a nearly unique 256-bit (32-byte) signature for a given text. It serves as a secure 'signature' for data files.

Client:

The client, utilizing front-end technology like a web app, stores patient data. Before transmitting data to the blockchain or the third-party system, local encryption with AES takes place, and a key is generated using SHA-256 for added security.


PATIENT MEDICAL DATA

Medical Report ID	MEDREP20015638
Weight	158 KG
Height	164 cm
Disease Name	Hyper My
Blood Group	B+
Disease Started On	01/04/2016
Description	caused by long exposure to harmful artificial blue light

BACK
SAVE
Client
Locally Encrypted Data goes to third party
Central Authentication / Authorization / Encryption Third Party
Encrypts the locally encrypted data and saves its key in its database
2nd time encrypted data goes back to client

Fig: showing transmission of data between the Client and the Third-Party System write the same in the different words

10.Quality Assurance:

After the completion and approval of the release candidate version, the software undergoes a crucial phase known as Quality Assurance (QA). The QA process is essential for ensuring that the software meets predefined standards and functions as intended. The QA stages include:

Go through each QA step with an example using a hypothetical patient data management system developed with blockchain technology.

10.1.Integration Testing:

Objective: Ensure that the integrated components (Client, Blockchain, and Third-Party System) work together seamlessly.

Test whether the encrypted data generated by the client is successfully transmitted to the Third-Party System and then recorded on the blockchain without data loss or corruption.

10.2.System Testing:

Objective: Evaluate the entire patient data management system's functionality and behaviour.

Verify that the end-to-end process, including data encryption, storage in the blockchain, and retrieval through the Third-Party System, works as expected without errors.

10.3.Performance Testing:

Objective: Assess the software's speed, responsiveness, and stability under various conditions.

Measure the time it takes for the system to encrypt and store data, ensuring that it meets performance benchmarks even during peak usage.

10.4.Security Testing:

Objective: Identify and address vulnerabilities to ensure robust protection.

Attempt to access encrypted data without proper authorization, ensuring that the system denies unauthorized access attempts.

10.5.User Acceptance Testing (UAT):

Objective: Validate that the software aligns with end-users' needs and expectations.

Have actual users (medical practitioners and data owners) interact with the system, checking if they can easily upload, access, and manage patient data securely.

10.6.Regression Testing:

Objective: Ensure that new changes or additions do not negatively impact existing functionalities.

After introducing a new feature for enhanced encryption, verify that it doesn't introduce errors or affect the core encryption and storage processes.

10.7.Load Testing:

Objective: Assess the system's capability to handle expected user loads.

Simulate a scenario where multiple data owners simultaneously upload encrypted patient data, ensuring the system maintains performance and responsiveness.

10.8.Stress Testing:

Objective: Evaluate the system's stability under extreme conditions.

Subject the system to a sudden surge in data upload requests to identify its breaking points and understand how it recovers.

10.9.QA Report:

Summarizes findings from each testing phase, detailing any issues discovered and providing recommendations for improvement.

Report might indicate that the system performed well under normal load but experienced delays during stress testing, suggesting the need for optimization for extreme usage scenarios.

11.Deployment:

Certainly! Let's customize the deployment procedure and licensing terms for the Patient Data Management System using Blockchain:

11.1.Pre-Deployment Checklist:

- Ensure successful completion of development and QA phases.
- Confirm the availability of blockchain infrastructure (nodes, network).
- Verify implementation of encryption and security measures.

11.2.Software Package Preparation:

- Compile the software package, including the blockchain application, smart contracts, and necessary configurations.
- Ensure that the package aligns with the latest version featuring enhanced data privacy.

11.3.Testing in Staging Environment:

- Deploy the software in a staging environment with a replicated blockchain network.

- Conduct final checks, including testing encryption/decryption processes and blockchain interactions.

11.3.Backup and Rollback Plan:

- Implement a backup plan for encrypted data and blockchain configurations.

- Establish a rollback strategy using version control in case of unexpected issues.

11.3.Deployment to Production:

- Schedule deployment during a low-impact period to minimize user disruption.

- Use automated deployment scripts to ensure consistency.

- Monitor real-time deployment to catch and address any issues promptly.

11.3.Post-Deployment Verification:

- Perform extensive testing post-deployment to ensure proper encryption and blockchain functionality.

- Address any issues, if identified, to maintain a seamless user experience.

11.3.User Communication:

- Notify users about the deployment schedule, emphasizing enhanced data security.

- Share information on new features, such as improved access control through blockchain.

11.3.Monitoring and Performance Tuning:

- Implement continuous monitoring of blockchain performance.

- Fine-tune configurations for optimal data storage and retrieval.

12.Licensing Terms:

12.1. User Licenses:

Define user licenses based on roles: data owner, doctor, admin.

Each hospital using the Patient Data Management System is granted a specified number of admin, doctor, and data owner licenses based on their requirements.

12.2. Subscription Model:

Specify subscription terms.

The Patient Data Management System operates on an annual subscription model, renewable upon expiry.

12.3. Usage Limitations:

Clarify restrictions on software use.

This software is licensed for internal use within the healthcare institution and should not be shared with external entities.

12.4. Support and Updates:

Detail support services.

Subscribers are entitled to technical support and receive regular updates containing security enhancements and new features.

12.5. Termination Conditions:

Outline conditions for license termination.

The license may be terminated if there is a breach of the terms outlined in this agreement, subject to a notice period.

12.6. Compliance and Auditing:

Emphasize user responsibility for compliance.

Example: "Users must comply with all licensing terms, and periodic audits may be conducted to ensure adherence."

12.7. Intellectual Property:

- Specify ownership rights.

The intellectual property rights for the Patient Data Management System, including the blockchain application and smart contracts, are owned by company,

12.8. Confidentiality:

Address the handling of user data.

User data stored in the Patient Data Management System is treated with utmost confidentiality, and [Your Company] ensures rigorous data protection measures.

12.9. Legal Disclaimers:

Include legal disclaimers.

Users acknowledge that [Your Company] is not liable for any indirect, incidental, or consequential damages arising from the use of the Patient Data Management System.

12.10. License Agreement Acceptance:

Require explicit acceptance.

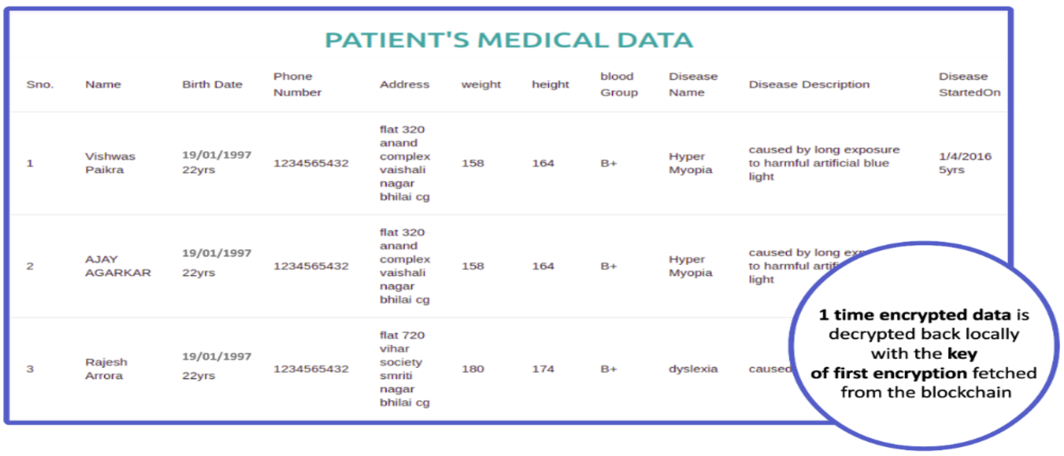
Users must accept the terms of this license agreement before accessing and using the Patient Data Management System.

13.Summary:

The inherent immutability of blockchain data poses significant challenges, particularly when there's a need for data deletion or selective access. In response to these privacy concerns, we've devised a robust solution: a two-tier encryption approach. Firstly, the data undergoes encryption locally, and subsequently, it undergoes another layer of encryption within the Third-Party System. When transmitted to the blockchain, the data remains encrypted, rendering any unauthorized attempts to decipher it futile. This double-encryption strategy ensures that even with blockchain's transparency, sensitive information remains secure and inaccessible to those without the proper decryption keys.

Moreover, we've addressed the concern of data deletion from the blockchain. If a user desires to expunge their data, the process involves the removal of associated decryption keys stored in the Third-Party System. This strategic deletion renders the encrypted data on the blockchain indecipherable and effectively useless. By implementing these measures, we've not only fortified the privacy aspects within the blockchain environment but also provided users with a mechanism to control the permanence of their data, striking a balance between security and user autonomy.

Our future vision for this project involves continuous improvement in encryption techniques, exploration of alternative consensus mechanisms, and enhanced interoperability. We aim to develop decentralized identity management, user-friendly interfaces, and integrate AI for analytics. Staying compliant with healthcare regulations, fostering community engagement, and promoting global impact are integral. Education initiatives will ensure informed adoption, making the project a benchmark for secure and robust healthcare blockchain applications.



PATIENT'S MEDICAL DATA

Sno.	Name	Birth Date	Phone Number	Address	weight	height	blood Group	Disease Name	Disease Description	Disease StartedOn
1	Vishwas Paikra	19/01/1997 22yrs	1234565432	flat 320 anand complex vaishali nagar bhilai cg	158	164	B+	Hyper Myopia	caused by long exposure to harmful artificial blue light	1/4/2016 5yrs
2	AJAY AGARKAR	19/01/1997 22yrs	1234565432	flat 320 anand complex vaishali nagar bhilai cg	158	164	B+	Hyper Myopia	caused by long ex to harmful arti light	
3	Rajesh Arrora	19/01/1997 22yrs	1234565432	flat 720 vihar society smriti nagar bhilai cg	180	174	B+	dyslexia	caused	

1 time encrypted data is decrypted back locally with the **key of first encryption** fetched from the blockchain

14.Reference:

“Blockchain” https://en.wikipedia.org/wiki/Blockchain

“Ethereum” https://ethereum.org/en/

“Smart Contracts” https://ethereum.org/en/developers/docs/smart-contracts/

“SHA-256” https://en.wikipedia.org/wiki/SHA-2

“AES” https://www.educative.io/edpresso/what-is-the-aes-algorithm

"How to Build Ethereum Dapp with React.js · Complete Step-By-Step Guide" https://www.dappuniversity.com/articles/ethereum-dapp-react-tutorial