**BLOCKCHAIN – BASED DRUG TRACEBILITY SYSTEM**

**NAAN MUDHALVAN**

(2023 – 2024)

**PROJECT REPORT**

Submitted By

NM ID : NM2023TMID02822

MAHINDRA JEYARAM.T(961820105019)

ARAVIND R.A (961820105302)

JENO JOSELIN.G (961820105303)

VISHWAG.K (961820105309)

In Partial Fulfilment for the Award of the Degree

Of

BACHELOR OF ENGINEERING

In

ELECTRICAL AND ELECTRONICS ENGINEERING

**PONJESLY COLLEGE OF ENGINEERING,**

**NAGERCOIL- 3**

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
| **1** | **INTRODUCTION** | 3 |
|  | 1.1 Project Overview | 3 |
|  | 1.2 Purpose | 3 |
| **2** | **LITERATURE SURVEY** | 4 |
|  | 2.1 Existing problem | 4 |
|  | 2.2 References | 4 |
|  | 2.3 Problem Statement Definition | 5 |
| **3** | **IDEATION & PROPOSED SOLUTION** | 6 |
|  | 3.1 Empathy Map Canvas | 6 |
|  | 3.2 Ideation & Brainstorming | 7 |
| **4** | **REQUIREMENT ANALYSIS** | 10 |
|  | 4.1 Functional requirement | 10 |
|  | 4.2 Non-Functional requirements | 11 |
| **5** | **PROJECT DESIGN** | 13 |
|  | 5.1 Data Flow Diagrams & User Stories | 13 |
|  | 5.2 Solution Architecture | 15 |
| **6** | **PROJECT PLANNING & SCHEDULING** | 16 |
|  | 6.1 Technical Architecture | 16 |
|  | 6.2 Sprint Planning & Estimation | 16 |
|  | 6.3 Sprint Delivery Schedule | 18 |
| **7** | **CODING & SOLUTIONS** | 19 |
|  | 7.1 Feature 1 | 22 |
|  | 7.2 Feature 2 | 23 |
|  | 7.3 Database Schema | 23 |
| **8** | **PERFORMANCE TESTING** | 24 |
|  | 8.1 Performance Metrics | 24 |
| **9** | **RESULTS** | 25 |
|  | 9.1 Output Screenshots | 25 |
| **10** | **ADVANTAGES & DISADVANTAGES** | 28 |
| **11** | **CONCLUSION** | 29 |
| **12** | **FUTURE SCOPE** | 30 |
| **13** | **APPENDIX** | 31 |
|  | Source Code | 31 |
|  | GitHub & Project Demo Link | 50 |

**CHAPTER 1**

**INTRODUCTION**

* 1. **PROJECT OVERVIEW**

Healthcare supply chains are complex structures spanning across multiple organizational and geographical boundaries, providing critical backbone to services vital for everyday life. The inherent complexity of such systems can introduce impurities including inaccurate information, lack of transparency and limited data provenance. Counterfeit drugs is one consequence of such limitations within existing supply chains which not only has serious adverse impact on human health but also causes severe economic loss to the healthcare industry. Consequently, existing studies have emphasized the need for a robust, endto-end track and trace system for pharmaceutical supply chains. Therein, an end-to-end product tracking system across the pharmaceutical supply chain is paramount to ensuring product safety and eliminating counterfeits. Most existing track and trace systems are centralized leading to data privacy, transparency and authenticity issues in healthcare supply chains. In this paper, we present an Ethereum blockchain-based approach leveraging smart contracts and decentralized off-chain storage for efficient product traceability in the healthcare supply chain. The smart contract guarantees data provenance, eliminates the need for intermediaries and provides a secure, immutable history of transactions to all stakeholders. We present the system architecture and detailed algorithms that govern the working principles of our proposed solution. We perform testing and validation, and present cost and security analysis of the system to evaluate its effectiveness to enhance traceability within pharmaceutical supply chains.

Healthcare supply chain is a complex network of several independent entities that include raw material suppliers, manufacturer, distributor, pharmacies, hospitals and patients. Tracking supplies through this network is non-trivial due to several factors including lack of information, centralized control and competing behaviour among stakeholders. Such complexity not only results in in-efficiencies such as those highlighted through COVID-19.

* 1. **PURPOSE**

The importance of drug traceability (track and trace) is increasingly emphasized and mandated by several countries across the world. For example, the U.S. Drug Supply Chain Security Act (DSCSA) has made it mandatory for the pharmaceutical industry to develop an electronic and interoperable system that identifies and tracks prescription drugs as they are distributed across the United States [. Similarly, over the last 8 years, China required all the stakeholders involved in the drugs supply chain to record information of individual pharmaceutical products in a specialized IT system whenever drugs are sent to/from their warehouses .

Therefore, drug traceability has become an integral part of the pharmaceutical supply chain as it establishes authenticity, and aims to track and trace chain of custody of the product across drug supply chain. Blockchain technology has introduced a new model of application development primarily based on the successful implementation of the data structure within the Bitcoin application. The fundamental concept of the blockchain data structure is similar to a linked list i.e. it is shared among all the nodes of the network where each node keeps its local copy of all the blocks (associated with the longest chain) starting from its genesis block . Recently, many real-world applications have been developed in diverse domains, such as the Internet of Things , e-Government and e-document management . These applications leverage benefits of blockchain technology due to its self-cryptographic validation structure among transactions (through hashes), and public availability of distributed ledger of transaction-records in a peer-to-peer network. Creating a chain of blocks connected by cryptographic constructs (hashes) makes it very difficult to tamper the records, as it would cost the rework from the genesis to the latest transaction in blocks as illustrated by [19]. Within the context of blockchain-based traceability for pharmaceutical supply chain, [20] presents one of the initial efforts. Although our solution has similarities with this effort due to the focus on pharmaceutical supply chain as well as the use of blockchains, we take a holistic view of 2 VOLUME 4, 2016 This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/ This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3049920, IEEE Access A. Musamih et al.: A Blockchain-based Approach for Drug Traceability in Healthcare Supply Chain the pharmaceutical supply chain, presenting an end-to-end solution for drug traceability whereas [20] only focused on a subset of these challenges. Firstly, our approach identifies and engages major stakeholders in the drug supply chain i.e. the FDA, supplier, manufacturer, distributor, pharmacy, and patient, whereas [20] is limited to the supplier, manufacturer, and wholesaler as the stakeholders. Consequently, the pharmacists are represented as an external entity which is not the case in a real drug supply chain. Secondly, we make explicit efforts to identify and define relationships among stakeholders, on-chain resources, smart contracts, and decentralized storage systems which is lacking in [20]. Furthermore, in view of the significance of interactions among stakeholders, we have included precise definitions to remove any ambiguity, whereas such interactions have not been defined as part of [20]. Thirdly, we use the smart contracts technology to achieve real-time, seamless traceability with push notifications so as to minimize human intervention and therefore undesired delays. Specifically, each drug Lot is assigned a unique smart contract that generates an event whenever a change in ownership occurs and a list of events is delivered to the DApp user. However, the smart contracts in [20] are programmed for specific roles such as supplier, manufacturer, and wholesaler which requires each participant to manually confirm which drugs are received. Such approach can introduce delays and inaccuracies in the immutable data stored on the ledger. Finally, we have conducted a cost and security analysis to evaluate the performance of the proposed solution including discussion on how the proposed solution can be generalized to other supply chains. The challenge of achieving traceability to mitigate against counterfeit drugs is well-established and several efforts have been made to address this within pharmaceutical industry. However, a careful review of literature presents several gaps and opportunities for a comprehensive application of blockchain technology for drug traceability. In this context, the primary contributions of this paper can be summarized as follows: • We propose a blockchain-based solution for the pharmaceutical supply chain that provides security, traceability, immutability, and accessibility of data provenance for pharmaceutical drugs. • We design a smart contract capable of handling various transactions among pharmaceutical supply chain stakeholders. • We present, implement and test the smart contract that defines the working principles of our proposed solution. • We conduct security and cost analysis to evaluate the performance of the proposed blockchain-based solution. The reminder of this paper is organized as follows. Section II presents a critical review of existing efforts with respect to traceability in the healthcare supply chain. This is followed by a description of the proposed blockchain-based track & trace system for phamaceutical products in section III. Section IV presents the implementation of the proposed system along with details of the testing and evaluation in section V. Section VI describes the efforts to evaluate the proposed system and analyzes the outcomes of evaluation. Section VII concludes the paper summarizing contributions and highlighting avenues for further work.

**CHAPTER 2**

**2.1 EXISTING PROBLEM:**

Traceability is defined as the ability to access any or all information relating to the object under consideration, throughout its life cycle, by means of recorded identifications. The object under consideration is referred to as Traceable Resource Unit (TRU) which is any traceable object within the supply chain. Traceability objectives are twofold; to track the history of transactions, and to track the real-time position of the TRU. In this context, a traceability system requires access to information related to the drug which is the TRU in the supply chain by using different identification techniques to record its identity and distinguish it from other TRUs. The components of a traceability system can be broadly identified by a mechanism for identifying TRUs, a mechanism for documenting the connections between TRUs, and a mechanism for recording the attributes of the TRUs

Existing solutions within supply chain management have traditionally used barcodes and RFID tags as identification techniques, Wireless Sensor Networks (WSN) to capture data, and Electronic Product Code (EPC) to identify, capture, and share product information to facilitate tracking of goods through different stages . In this context, Smart-Track utilizes GS1 standards barcodes containing unique serialized product identifier, Lot production and expiration dates. The information contained in the GS1 barcode is captured across various supply chain processes and used to maintain a continuous log of ownership transfers. As each stakeholder records the possession of the product, an end user (patient) can verify authenticity through central data repository maintained as Global Data Synchronization Network (GDSN) by using a smartphone app. In the downstream supply chain at the warehouse, pharmacy and hospital units can scan the barcode to verify the product and its characteristics. Similarly, Data-Matrix tracking system creates a Data-Matrix for each drug which includes the manufacturer ID, Product ID, Unique ID of the package, the authentication code, and an optional meta-data. This allows the patient to verify the origin of the drug by using the attached Data-Matrix

* 1. **REFERENCES:**

[1]Shortage of personal protective equipment endangering health workers worldwide" https://tinyurl.com/v5qauvp. [Accessed on: 3-June-2020]

[2] Chambliss W, Carroll W, Yelvigi M, el al. "Role of the pharmacist in preventing distribution of counterfeit medications". J Am pharm Assoc. 2012;52(2): 195-199.

[3] Ziance RJ. "Roles for pharmacy in combating counterfeit drugs". J Am pharm Assoc. 2008;48: e71-e88.

\[4] Toscan P. "The dangerous world of counterfeit prescription drugs". Available: http://usatoday30.usatoday.com/money/industries/health/ drugs/story/2011-10-09/cnbc-drugs/50690880/1.[Accessed on: 3-June2020]

[5] T.Adhanom, "Health is a fundamental human right",2017. Available:https://www.who.int/mediacentre/news/statements/ fundamental-human-right/en/. [Accessed: 26- May- 2020]. [6] World Health Organization, "Growing threat from counterfeit medicines", 2010.

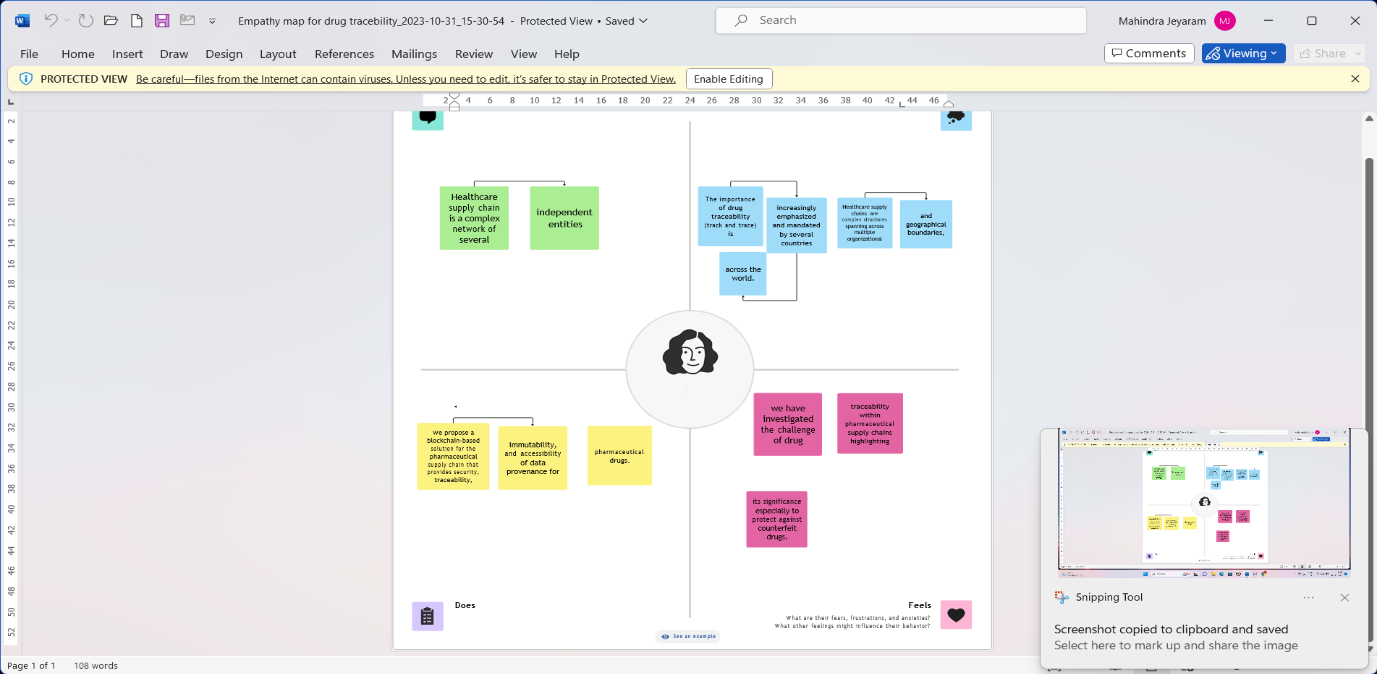
* 1. **PROBLEM STATEMENT DEFINITION:**

The context of blockchain-based traceability for pharmaceutical supply chain, presents one of the initial efforts. Although our solution has similarities with this effort due to the focus on pharmaceutical supply chain as well as the use of blockchains, we take a holistic view of 2 VOLUME 4, 2016 This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/ This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2021.3049920, IEEE Access A. Musamih et al.: A Blockchain-based Approach for Drug Traceability in Healthcare Supply Chain the pharmaceutical supply chain, presenting an end-to-end solution for drug traceability whereas only focused on a subset of these challenges. Firstly, our approach identifies and engages major stakeholders in the drug supply chain i.e. the FDA, supplier, manufacturer, distributor, pharmacy, and patient, whereas is limited to the supplier, manufacturer, and wholesaler as the stakeholders. Consequently, the phamacists are represented as an external entity which is not the case in a real drug supply chain. Secondly, we make explicit efforts to identify and define relationships among stakeholders, on-chain resources, smart contracts, and decentralized storage systems which is lacking in Furthermore, in view of the significance of interactions among stakeholders, we have included precise definitions to remove any ambiguity, whereas such interactions have not been defined as part of [Thirdly, we use the smart contracts technology to achieve real-time, seamless traceability with push notificaions so as to minimize human intervention and therefore undesired delays. Specifically, each drug Lot is assigned a unique smart contract that generates an event whenever a change in ownership occurs and a list of events is delivered to the DApp user. However, the smart contracts in are programmed for specific roles such as supplier, manufacturer, and wholesaler which requires each participant to manually confirm which drugs are received. Such approach can introduce delays and inaccuracies in the immutable data stored on the ledger. Finally, we have conducted a cost and security analysis to evaluate the performance of the proposed solution including discussion on how the proposed solution can be generalized to other supply chains.

**CHAPTER 3**

**IDEATION & PROPOSED SOLUTION**

**3.1 EMPATHY MAP CANVAS**

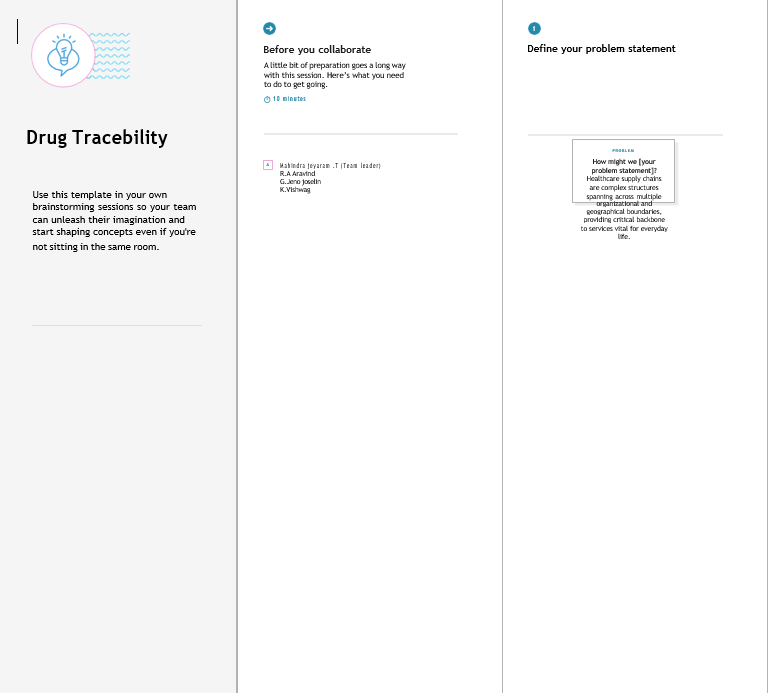
****

**3.2 IDEATION & BRAINSTORMING**

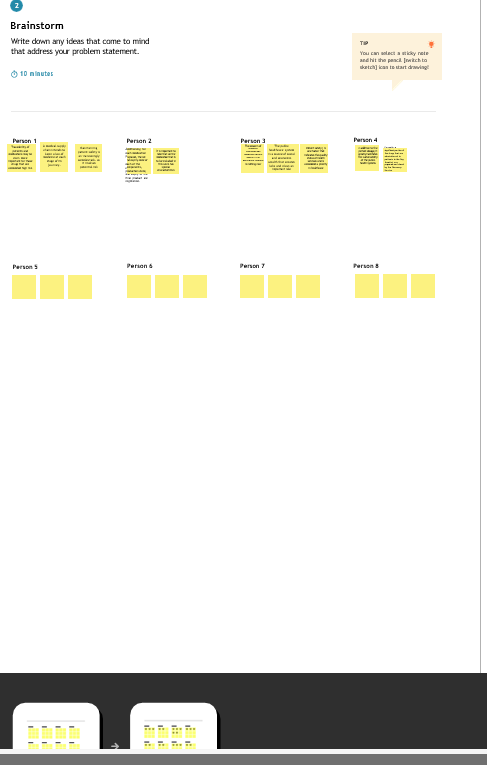
In the development of the " Blockchain – Based drug tracebility System," the project team engaged in a comprehensive ideation and brainstorming process to explore various potential solutions and features that can enhance the system's functionality. The following innovative ideas and strategies were considered:

supply chain are typically centralized and lack transparency across participants of the supply chain, which allows the central authority to modify information without notifying other stakeholders. On the other hand, a blockchain based solution offers data security, transparency, immutability, provenance and authenticated transaction records. Blockchain is a decentralized, immutable shared ledger that can be applied to a variety of business settings involving transaction processes. Transparency and traceability are used interchangeably however, they represent very different concepts. Transparency is usually used when referring to high-level information of a supply chain. For example, product’s components, facilities locations, names of suppliers, etc. with the objective to map the whole supply chain. However, traceability is related to granular information where it envisages choosing a specific component to trace, determines common standards to communicate with partners, implements methods to produce and gather accurate data, selects a platform to store traceability data, and determines how to share data on the platform. Although both terms represent different concepts, they rely on each other because accessing granular information requires full understanding of the supply chain. In this respect, a number of existing approaches leverage cryptographic properties of blockchain to achieve a decentralized, verifiable track and trace system for pharmaceutical drugs. Mettler et al. [32] have discussed the use of blockchain based approach for various issues in healthcare sector with no technical details or specific application. Kurki [33], presented the advantages of blockchain technology in pharmaceutical supply chain. However, similar to [32] only conceptual discussion was provided. Muniandy and Ong [20] proposed a traceability system using Ethereum for anticounterfeiting. The proposed solution employs smart contract however it lacks implementation or evaluation

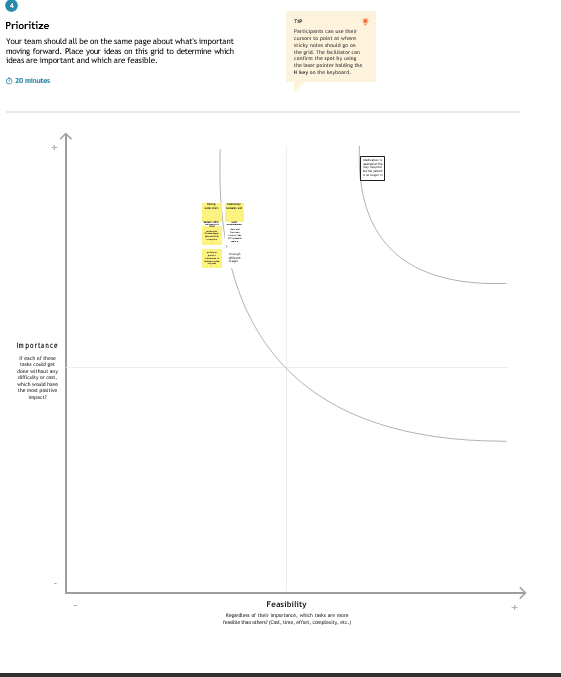
**Step-1: Team Gathering, Collaboration and Select the Problem Statement**

****

**Step-2: Brainstorm, Idea Listing and Grouping**

****

**Step-3: Idea Prioritization**

****

**CHAPTER 4**

**REQUIREMENT ANALYSIS**

**4.1 FUNCTIONAL REQUIREMENTS:**

|  |  |  |
| --- | --- | --- |
| **FR NO** | **FUNCTIONAL REQUIREMENT** | **SUB REQUIREMENT** |
| FR – 1 | User Registration and Authentication | Users should be able to create accounts with unique usernames and passwords.  Users should have the option for secure authentication methods (e.g., two-factor authentication). |
| FR – 2 | Drug Data Recording | Users can input drug product details, including drug information, livestock data, and supply chain information. - Data should be stored securely on the blockchain. |
| FR – 3 | Querying drug Data | Users should be able to query drug data, filter information by various parameters, and retrieve specific details from the blockchain. |
| FR – 4 | Data Update and Modification | Users should have the capability to update and modify data when necessary, ensuring that the blockchain remains accurate and up-to-date. |
| FR – 5 | Data Privacy and Security | Implement robust data security measures to protect sensitive information from unauthorized access and tampering. |
| FR – 6 | User Interface | Develop a user-friendly interface that allows users to interact with the smart contract and blockchain easily. |
| FR - 7 | Role Based Access Control | Implement role-based access control to manage user permissions for data modification. |
| R - 8 | Audit Trial | Maintain an audit trail of data changes to enhance transparency and traceability within the blockchain. |

These functional requirements are designed to ensure that the "Blockchain – Based Agriculture Data Management System" is capable of recording, querying, and securely managing agriculture data on the Ethereum blockchain.

**4.2 NON-FUNCTIONAL REQUIREMENTS:**

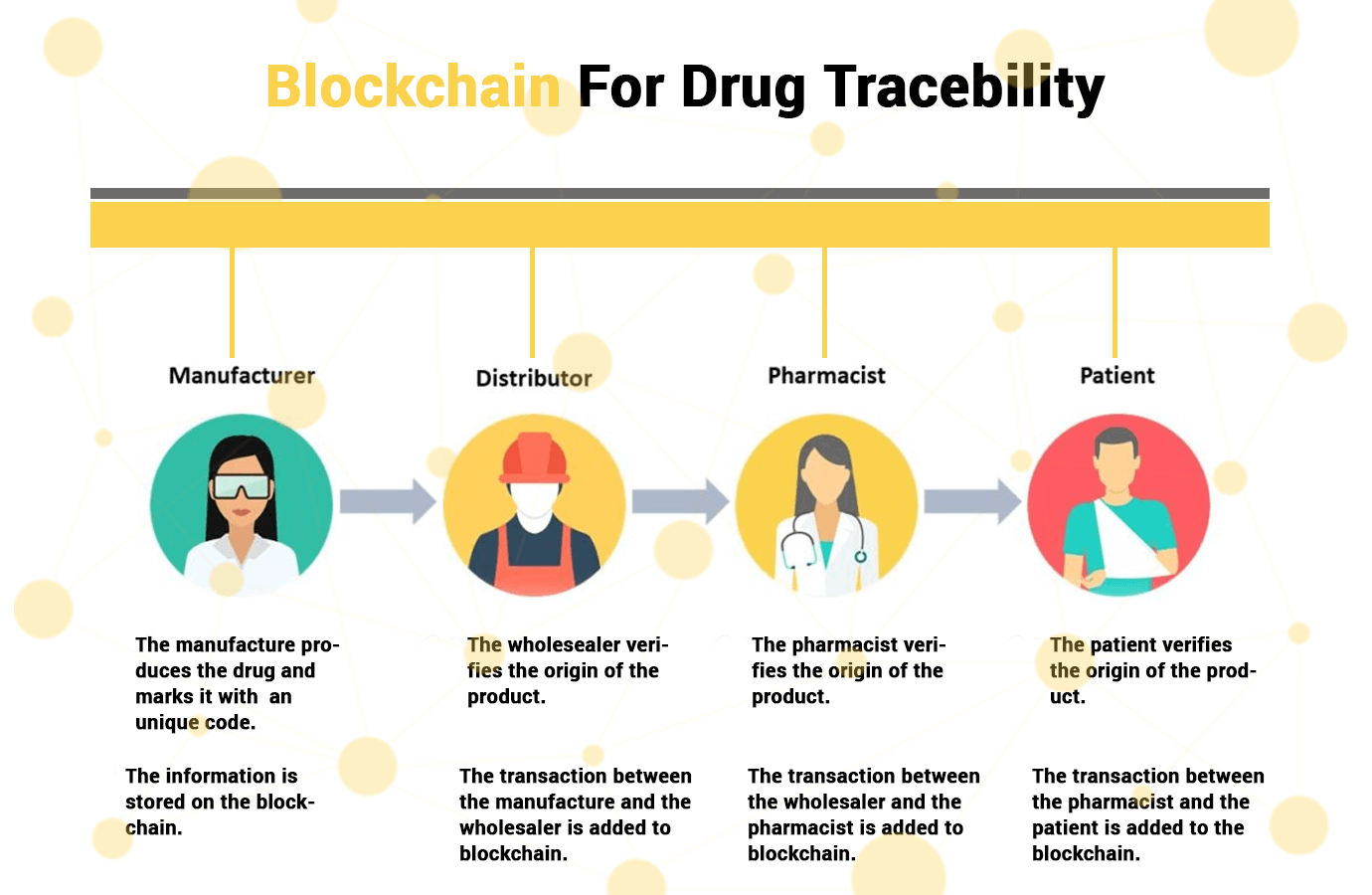
|  |  |  |
| --- | --- | --- |
| **NFR NO** | **NON-FUNCTIONAL REQUIREMENT** | **DESCRIPTION** |
| NFR – 1 | Scalability | The system should be designed to easily scale to accommodate an increasing user base and growing data. |
| NFR – 2 | Reliability | The system should be available and reliable, with minimal downtime or service interruptions. |
| NFR – 3 | Security | Data should be stored securely, with encryption to protect sensitive user information. |
| NFR – 4 | Data Backup and Recovery | Regular data backups should be performed to prevent data loss in the event of system failures. |
| NFR – 5 | Usability | The system should be user-friendly, with an intuitive and easy-to-navigate interface. |
| NFR – 6 | Accessibility | The system should comply with accessibility standards (e.g., WCAG) to make it usable by individuals with disabilities. |
| NFR – 7 | Response Time | Define acceptable response times for various system operations (e.g., loading a page, submitting data). |

These non-functional requirements outline the performance, reliability, security, and usability standards that the "Blockchain – Based Agriculture Data Management System" should adhere to in order to provide a robust and user-centric solution.

**CHAPTER 5**

**PROJECT DESIGN**

**5.1 DATA FLOW DIAGRAMS & USER STORIES**

****

**DATA FLOW DIAGRAM:**

**BLOCKCHAIN – BASED AGRICULTURE DATA MANAGEMENT SYSTEM**

**USER STORIES:**

**User Story 1**: I want to upload and update information about my drug products in the blockchain database, so I can maintain accurate and up-to-date records.

**Requirements:**

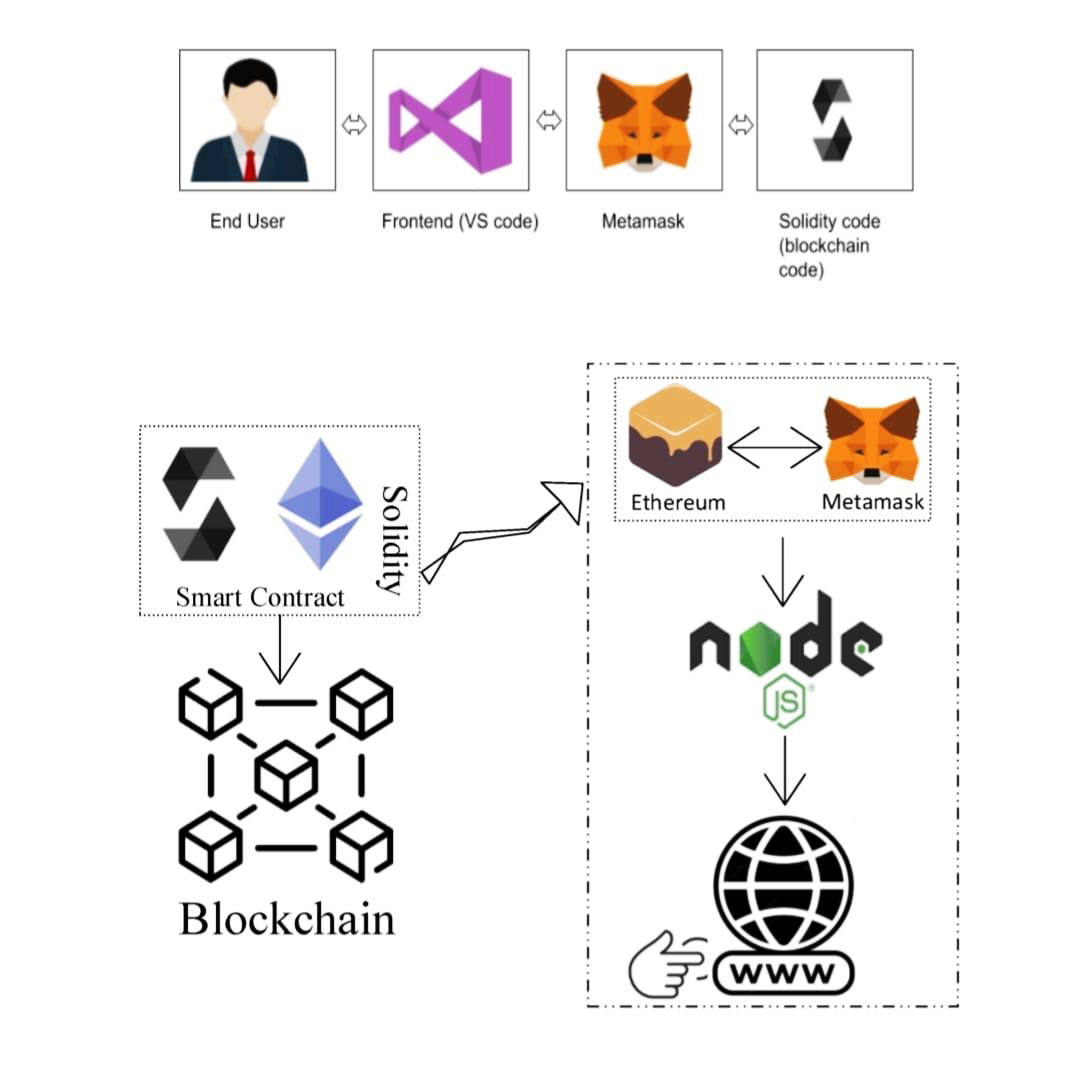
* The system should allow me to upload details of my drug products, including crop information, livestock data, and supply chain information.
* I should be able to easily update this information when there are changes, such as product details new livestock data, or supply chain updates.
* The system should maintain a secure and immutable record of the information I upload, ensuring data accuracy and traceability.

**User Story 2:** As a Stakeholder in the pharameautical Industry, I want to view specific details of an drug product stored in the blockchain database, so I can gather information about a particular product.

**Requirements:**

* The system should provide a search or query feature that allows me to specify the product I'm interested in.
* When I search for a specific drug product, the system should display detailed information, including livestock details, supply chain history, and any relevant data associated with the selected product.
* The information should be presented in a clear and organized manner, allowing me to quickly access the data I need for decision-making or verification.

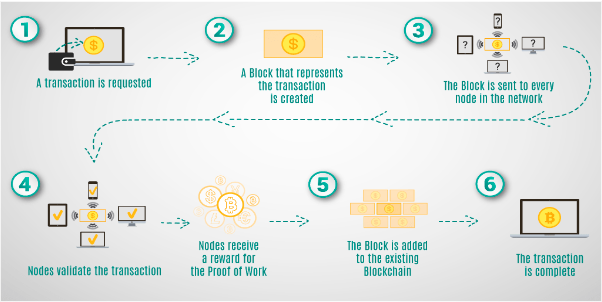
**SOLUTION ARCHITECTURE**



**Interaction between the Web and the Contract**

**CHAPTER 6**

**PROJECT PLANNING & SCHEDULING**

**6.1 TECHNICAL ARCHITECTURE**

**6.2 SPRINT PLANNING & ESTIMATION**

**Sprint Planning:**

Sprint planning is an integral part of our agile development process, ensuring that the project team selects and commits to completing specific work items from the product backlog during the upcoming sprint.

**Reviewing Product Backlog:** Our project team, composed of the Product Owner and the development team, conducts regular product backlog reviews. During these reviews, we evaluate user stories and technical tasks, taking into account the evolving needs and priorities of the project. This ongoing evaluation ensures that our work aligns with the project's overall goals.

**Setting Sprint Goals:** Based on the product backlog, our team establishes clear sprint goals. These goals serve as a guiding compass for the team throughout the sprint, ensuring alignment with the broader project objectives. Sprint goals help us maintain a focus on what needs to be achieved within the sprint.

**Breaking Down User Stories:** User stories and tasks are further decomposed into smaller, actionable sub-tasks. This detailed breakdown provides a comprehensive plan for the sprint, making it easier to understand the specific steps required to accomplish each task. Breaking down user stories helps ensure that we can tackle each part efficiently.

**Estimating Work:** Our development team employs agile estimation techniques, such as story points, to estimate the effort required for each task. These estimates help the team understand the scope and complexity of the work to be completed during the sprint. Accurate estimation is crucial for effective sprint planning and resource allocation.

**Sprint Backlog:** The selected user stories and tasks, along with their corresponding estimates, collectively form the sprint backlog. This backlog serves as the foundation for what the team will work on during the sprint. It includes a clear list of tasks to be completed and their associated estimates, ensuring that the team is aligned on the work to be accomplished.

This sprint planning and estimation process enables our team to effectively organize and execute the development of the "Blockchain-Enhanced Agriculture Data Management System." It ensures that we can allocate our resources efficiently and work towards achieving our project goals within each sprint.

**Estimation Techniques:**

In our project, we employ several estimation techniques to ensure that we can accurately gauge the complexity and effort required for tasks within each sprint. These techniques are vital for effective planning and resource allocation:

**Story Points:** Story points serve as a relative measure of the complexity and effort needed to complete a task. Tasks are assigned story point values based on their complexity compared to reference tasks. In the context of our project, story points help us assess the complexity of user stories and technical tasks related to agriculture data management. By using this technique, we can prioritize and allocate resources efficiently based on the estimated effort required for each task. Story points provide a common language for our development team to understand the relative complexities of different tasks.

**Adjustment for Uncertainty:** Recognizing the inherent uncertainty in complex projects, we incorporate an adjustment for uncertainty in our estimation process. Complex projects, like the development of the "Blockchain-Enhanced Agriculture Data Management System," may encounter unforeseen challenges, changing requirements, or unexpected issues. To accommodate such uncertainties, we establish buffers and contingencies within our sprint planning. These buffers allow us to address any unexpected deviations from the original plan while maintaining our commitment to sprint goals and timelines. This adjustment ensures that we can adapt to changing circumstances and still deliver a high-quality product within the defined project constraints.

**6.3 SPRINT DELIVERY SCHEDULE**

Sprint Delivery Schedule for Food Tracking System Project

|  |  |  |
| --- | --- | --- |
| **Sprint Delivery Schedule and its Objectives** | | |
| **Sprint 1** | Project Initiation | Set project scope and team roles. |
| **Sprint 2** | Data Modeling | Define data models and prioritize user stories. |
| **Sprint 3** | Blockchain Integration | Begin integrating blockchain technology. |
| **Sprint 4** | User Interfaces and Mobile Apps | Develop user interfaces and mobile apps. |
| **Sprint 5** | Quality Control and Testing | Implement quality control and initiate testing. |
| **Sprint 6** | Compliance and Security | Ensure regulatory compliance and enhance security. |
| **Sprint 7** | Deployment Preparation | Prepare for system deployment. |
| **Sprint 8** | Pilot Testing and Optimization | Conduct pilot tests and optimize the system. |
| **Sprint 9** | Full System Deployment | Roll out the fully operational system to a wider audience. |

This sprint delivery schedule outlines the specific objectives and focus areas for each sprint of the "Blockchain-Enhanced Agriculture Data Management System" project. It ensures that the project progresses systematically, with clear goals for each sprint.

**CHAPTER 7**

**CODING & SOLUTIONING**

**Smart Contract (Solidity)**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

struct DrugInfo {

string drugName;

string manufacturer;

uint256 manufacturingDate;

address trackingHistory;

}

contract Drug {

address public owner;

mapping(uint256 => DrugInfo) public drugs;

uint256 public drugCount;

constructor() {

owner = msg.sender;

}

modifier \_onlyOwner() {

require(msg.sender == owner, "Only the owner can perform this action");

\_;

}

modifier onlyOwner() {

require(msg.sender == owner, "Only the owner can perform this action");

\_;

}

event DrugManufactured(uint256 indexed drugId, string drugName, string manufacturer, uint256 manufacturingDate);

event DrugTransferred(uint256 indexed drugId, address indexed from, address indexed to, uint256 transferDate);

function manufactureDrug(uint256 drugId, string memory \_drugName, string memory \_manufacturer, uint256 \_manufacturingDate) external onlyOwner {

address initialHistory;

initialHistory = owner;

The key features related to drug tracking in this contract are:

**7.1 FEATURE 1**

* **Product Registration and Update:**

The smart contract allows for the registration and updating of products. Users can add information about an drug product, including its name, description, and quantity. This feature is essential for tracking and recording data about various drug products within the blockchain. Users, such as suppliers, can update the product information as needed, ensuring that the data remains accurate and up to date.

**7.2 FEATURE 2**

* **Ownership Control:**

The contract includes an ownership control mechanism. Only the owner of a particular product (as specified by the ‘**owner’** field) can perform actions such as updating the product's information. This feature enhances data security and ensures that only authorized users can modify the information associated with specific products.

**7.3 DATABASE SCHEMA**

In the context of the "Blockchain-Enhanced Agriculture Data Management System," the database schema is primarily defined by the structure of the smart contract deployed on the Ethereum blockchain. The Ethereum blockchain, being a decentralized and distributed ledger, serves as the underlying data storage mechanism for the project, eliminating the need for a traditional relational database schema.

The core data structure within the smart contract is the ‘**drug Product’** struct, which is used to organize and store information about agricultural products.

Here is the structure of the ‘**drug Product’** and how data is stored within the contract:

struct drug Product {

string name;

string description;

uint256 quantity;

address owner;

}

* **name**: Represents the name or title of the agricultural product.
* **description**: Contains a description or additional details about the product.
* **quantity**: Indicates the quantity or amount of the product.
* **owner**: Records the Ethereum address of the user who owns the product.

The data for each agricultural product is stored within the **products** mapping using a unique product identifier as the key. The Ethereum blockchain provides the underlying storage for this data, and it is inherently decentralized and immutable. There is no need for traditional relational database tables, as the blockchain itself acts as the distributed ledger for maintaining the integrity and security of the agricultural data.

This decentralized and blockchain-based database schema ensures that agricultural data is transparent, tamper-proof, and securely managed within the system. Users can access and update data with confidence, and the blockchain's architecture guarantees data immutability and traceability.Top of Form

**CHAPTER 8**

**PERFORMANCE TESTING**

**8.1 PERFORMANCE METRICS**

In the evaluation of the "Blockchain-Enhanced Agriculture Data Management System," several performance metrics are crucial for assessing the system's effectiveness and efficiency. These metrics are instrumental in ensuring the system's ability to meet the requirements of a modern and secure agricultural data management solution:

**1. Transaction Throughput:**

* **Metric:** Measure the number of transactions the blockchain can process per second.
* **Importance:** Higher throughput is essential for handling a large volume of data in real-time, ensuring the system can accommodate the demands of the agricultural supply chain.

**2. Data Accuracy:**

* **Metric:** Assess the accuracy of the data recorded on the blockchain.
* **Importance:** Ensure that the system minimizes errors and discrepancies in the supply chain data, enhancing trust and reliability in the recorded information.

**3. Data Integrity:**

* **Metric:** Measure the immutability of data stored on the blockchain.
* **Importance:** Verify that once data is recorded, it cannot be altered or deleted, ensuring data integrity and preventing unauthorized modifications.

**4. Security:**

* **Metric:** Monitor the system's ability to protect sensitive data and prevent unauthorized access or tampering.
* **Importance:** Maintaining robust security is crucial for safeguarding confidential agricultural data and ensuring the privacy of stakeholders.

**5. Response Time:**

* **Metric**: Evaluate the time it takes for the system to respond to inquiries or requests, such as tracking a product's origin.
* **Importance:** A prompt response time is critical for users to access timely information, enhancing the system's usability and efficiency.

**6. Supply Chain Efficiency:**

* **Metric:** Assess the system's impact on the overall efficiency of the food supply chain.
* **Importance:** Evaluate how the system contributes to reducing waste, optimizing inventory management, and minimizing delays in the agricultural supply chain.

**7. Feedback and Improvement:**

* **Metric:** Collect feedback from users and stakeholders to continuously improve the system, including the user interface and features.
* **Importance:** Ongoing feedback and improvements are essential for ensuring the system aligns with user needs and evolving industry requirements.

**8. Cost Efficiency:**

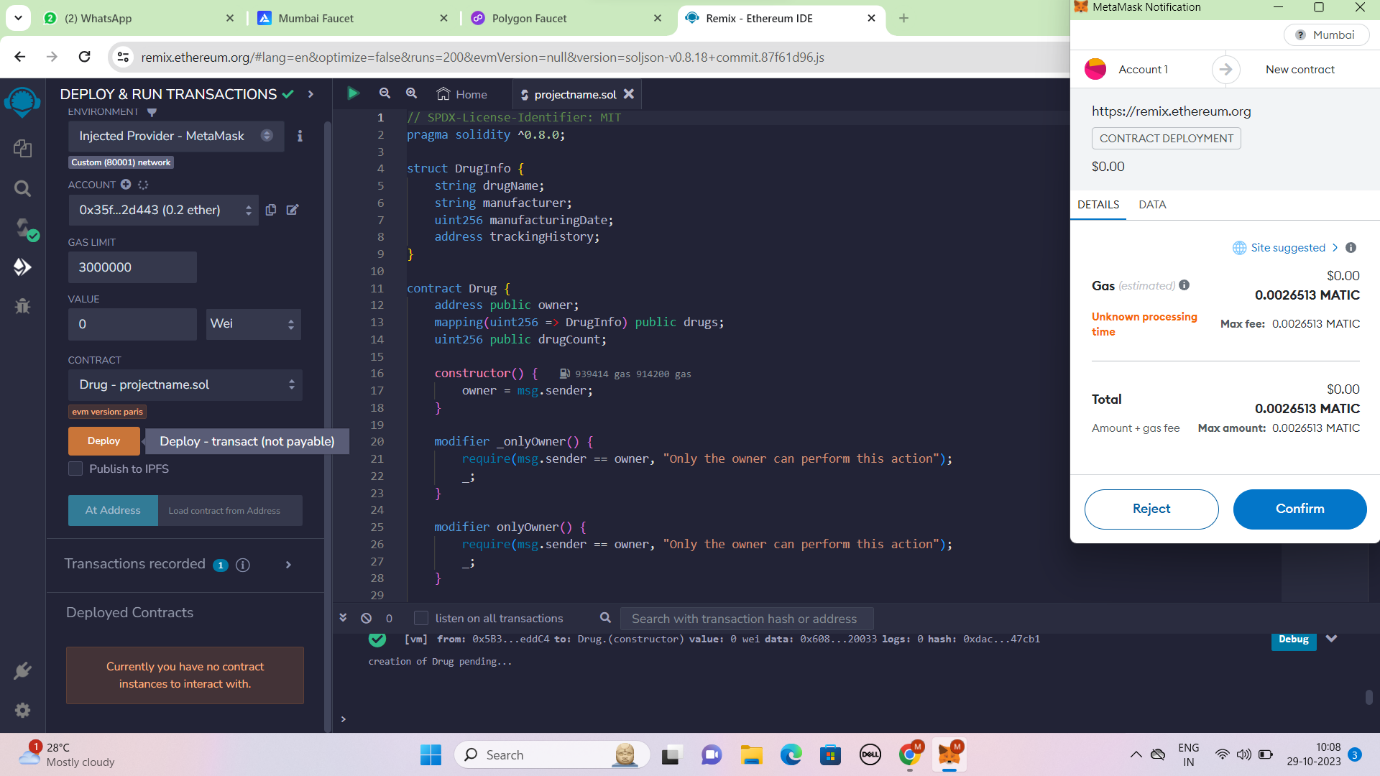
* **Metric:** Assess the system's cost-effectiveness, including the cost of blockchain technology, IoT sensors, and maintenance compared to the benefits it provides.
* **Importance:** Evaluating cost efficiency helps ensure that the system provides value for its investment and operational costs.

These performance metrics guide the evaluation and optimization of the "Blockchain-Enhanced drug tracebility System," ensuring that it meets the requirements of a modern and efficient data management solution.

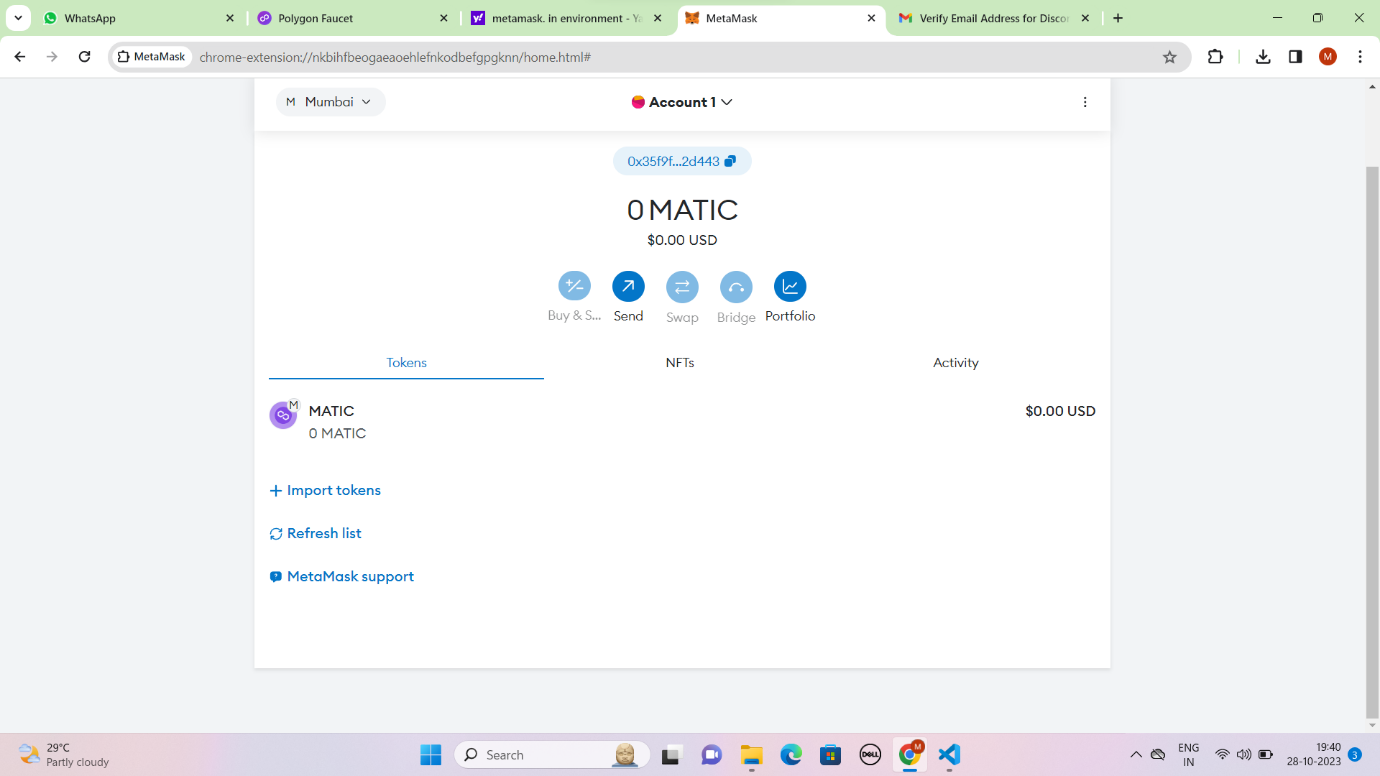
**CHAPTER 9**

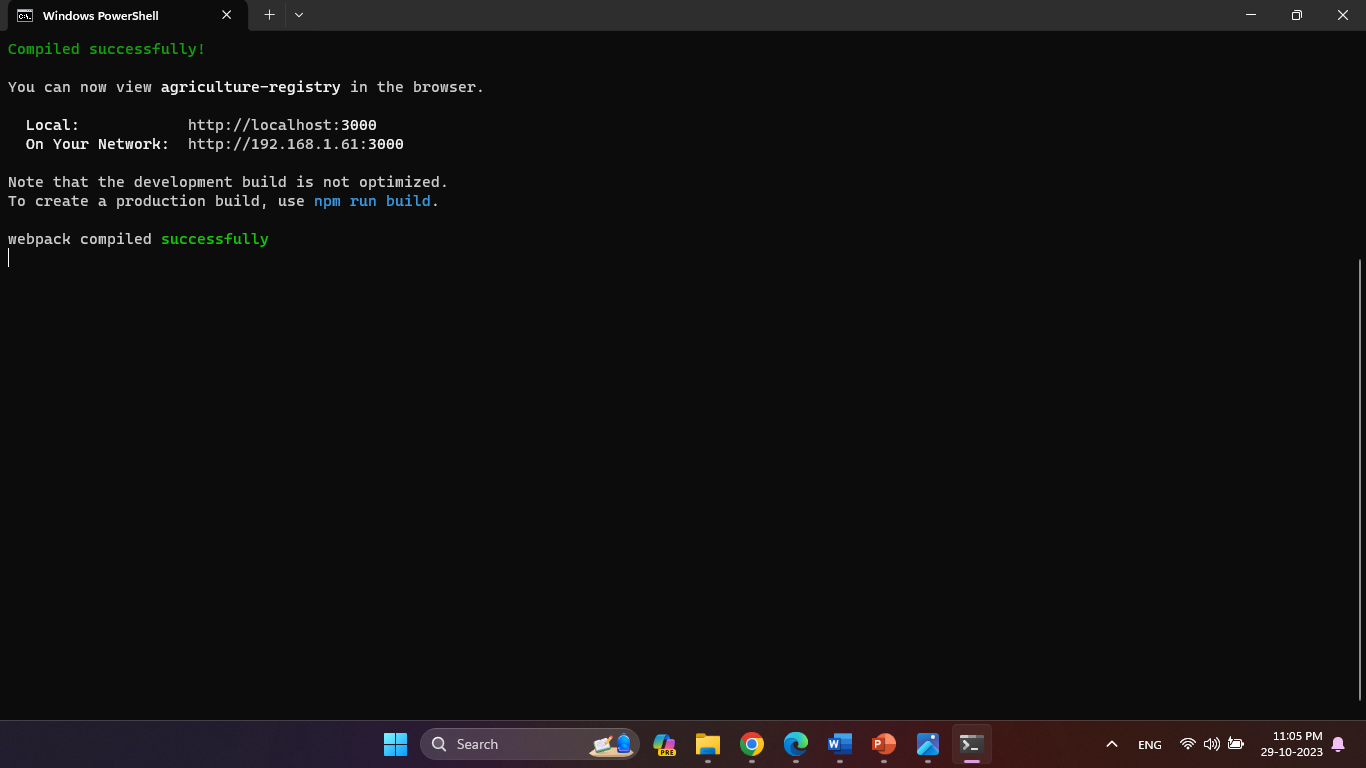
**RESULTS**

**9.1 OUTPUT SCREENSHOTS**

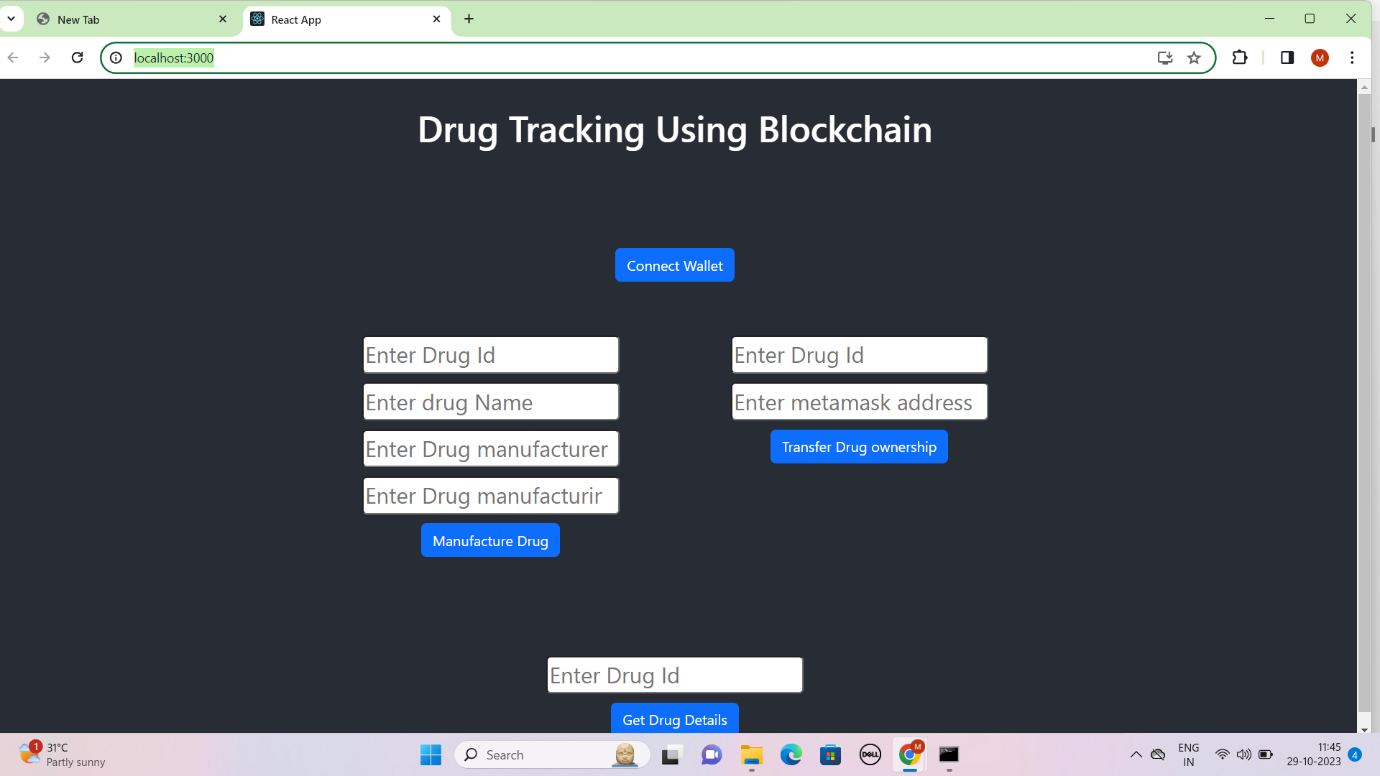
****

**METAMASK ACCOUNT**

****

**REMIX IDE – DEPLOYED CONTRACT**

**TERMINAL OUTPUT SCREEN**

****

**FINAL FRONTEND OUTPUT SCREEN**

**CHAPTER 10**

**ADVANTAGES & DISADVANTAGES**

**ADVANTAGES:**

1. **Data Transparency:** Blockchain ensures transparent and immutable records, enhancing trust and transparency in the agricultural supply chain.
2. **Traceability:** Users can trace the origin and journey of agricultural products, promoting food safety and quality assurance.
3. **Data Security:** The decentralized nature of blockchain enhances data security, protecting sensitive information from unauthorized access or tampering.
4. **Efficiency and Reduced Waste:** Improved supply chain efficiency reduces waste and optimizes inventory management, resulting in cost savings.
5. **User Engagement:** The system's user-friendly interface and features promote user engagement and participation in data management.

**DISADVANTAGES:**

1. **Cost of Implementation:** The initial setup, including blockchain technology and potential IoT sensors, can be costly, impacting the project's budget.
2. **Technological Barriers:** Some stakeholders may face challenges in adopting and adapting to blockchain technology and IoT sensors.
3. **Regulatory Compliance:** Ensuring compliance with relevant regulations and standards can be complex and time-consuming.
4. **Scalability:** Scaling the system to accommodate a growing volume of data and users may pose challenges that require ongoing maintenance and optimization.
5. **Data Accuracy:** Relying on user-input data can lead to inaccuracies or inconsistencies in the recorded information, affecting data reliability.

**CHAPTER 11**

**CONCLUSION**

**CONCLUSION:**

we have investigated the challenge of drug traceability within pharmaceutical supply chains highlighting its significance especially to protect against counterfeit drugs. We have developed and evaluated a blockchain-based solution for the pharmaceutical supply chain to track and trace drugs in a decentralized manner. Specifically, our proposed solution leverages cryptographic fundamentals underlying blockchain technology to achieve tamper-proof logs of events within the supply chain and utilizes smart contracts within Ethereum blockchain to achieve automated recording of events that are accessible to all participating stakeholders. We have demonstrated that our proposed solution is cost efficient in terms of the amount of gas spent in executing the different functions that are triggered within the smart contract. Moreover, the conducted security analysis has shown that our proposed solution achieves protection against malicious attempts targeting is integrity, availability and nonrepudiation of transaction data which is critical in a complex multi-party settings such as the pharmaceutical supply chain

**CHAPTER 12**

**FUTURE SCOPE**

**FUTURE SCOPE:**

The use of blockchain technology for drug traceability has the potential for significant future scope and impact in the pharmaceutical and healthcare industries. Drug traceability using blockchain can address various challenges such as counterfeit drugs, supply chain inefficiencies, and ensuring patient safety. Here are some aspects of its future scope:

1. **Enhanced Security and Authenticity**: Blockchain can provide a tamper-proof and transparent ledger for tracking drugs from their manufacturing origin to the end user. This enhances the security and authenticity of pharmaceutical products, reducing the risk of counterfeit or substandard drugs.

2. **Supply Chain Transparency**: Blockchain can provide real-time visibility into the pharmaceutical supply chain. This can help in tracking the movement of drugs, ensuring proper handling and storage, and reducing the risk of diversion or theft. It can also help in identifying the source of any contamination or quality issues.

3. **Reduced Counterfeit Drugs**: The ability to verify the authenticity of drugs at every stage of the supply chain can significantly reduce the circulation of counterfeit drugs. Patients and healthcare providers can easily verify the origin and authenticity of medications.

4. **Regulatory Compliance**: Blockchain can facilitate compliance with regulatory requirements. It can automate and streamline the reporting of drug transactions and help in ensuring that pharmaceutical companies adhere to safety and quality standards.

5. **Improved Recall Management**: In the event of a product recall, blockchain can enable rapid and accurate identification of affected batches, reducing the impact on patients and healthcare providers.

6**. Data Integrity**: Blockchain's immutability ensures that once data is recorded, it cannot be altered or deleted. This is crucial for maintaining the integrity of drug traceability records.

7. **Efficiency and Cost Savings**: By automating various processes and reducing the need for intermediaries, blockchain can lead to cost savings and efficiency improvements in the pharmaceutical supply chain.

8. **Patient Empowerment**: Patients can have more confidence in the medications they receive when they can verify the authenticity and source of the drugs. This can empower them to make more informed decisions about their healthcare.

9. **Global Adoption**: As pharmaceutical supply chains are global in nature, blockchain can facilitate international cooperation and standardization in drug traceability. It can provide a common platform for stakeholders worldwide.

10. **Integration with Emerging Technologies**: Blockchain can be integrated with other emerging technologies like IoT (Internet of Things) and AI (Artificial Intelligence) for enhanced monitoring and data analysis in the pharmaceutical supply chain.

While the future scope of drug traceability using blockchain is promising, there are also challenges and considerations to address.

**CHAPTER 13**

**APPENDIX**

**SOURCE CODE**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract Registry {

struct Product {

string name;

string description;

uint256 quantity;

address owner;

}

mapping(uint256 => foodProduct) public products;

uint256 public productCount;

event ProductAdded(uint256 productId, string name, string description, uint256 quantity, address owner);

event ProductUpdated(uint256 productId, string name, string description, uint256 quantity);

modifier onlyOwner(uint256 \_productId) {

require(products[\_productId].owner == msg.sender, "Only the owner can perform this action");

\_;

}

function addProduct(uint256 ProductId, string memory \_name, string memory \_description, uint256 \_quantity) external {

products[ProductId] = foodProduct(\_name, \_description, \_quantity, msg.sender);

productCount++;

emit ProductAdded(productCount, \_name, \_description, \_quantity, msg.sender);

}

function updateProduct(uint256 \_productId, string memory \_name, string memory \_description, uint256 \_quantity) external onlyOwner(\_productId) {

foodProduct storage product = products[\_productId];

product.name = \_name;

product.description = \_description;

product.quantity = \_quantity;

emit ProductUpdated(\_productId, \_name, \_description, \_quantity);

}

function getProductDetails(uint256 \_productId) external view returns (string memory name, string memory description, uint256 quantity, address owner) {

foodProduct memory product = products[\_productId];

return (product.name, product.description, product.quantity, product.owner);

}

}

**Connector.js**

const { ethers } = require("ethers");

const abi = [

{

"anonymous": false,

"inputs": [

{

"indexed": false,

"internalType": "uint256",

"name": "productId",

"type": "uint256"

},

{

"indexed": false,

"internalType": "string",

"name": "name",

"type": "string"

},

{

"indexed": false,

"internalType": "string",

"name": "description",

"type": "string"

},

{

"indexed": false,

"internalType": "uint256",

"name": "quantity",

"type": "uint256"

},

{

"indexed": false,

"internalType": "address",

"name": "owner",

"type": "address"

}

],

"name": "ProductAdded",

"type": "event"

},

{

"anonymous": false,

"inputs": [

{

"indexed": false,

"internalType": "uint256",

"name": "productId",

"type": "uint256"

},

{

"indexed": false,

"internalType": "string",

"name": "name",

"type": "string"

},

{

"indexed": false,

"internalType": "string",

"name": "description",

"type": "string"

},

{

"indexed": false,

"internalType": "uint256",

"name": "quantity",

"type": "uint256"

}

],

"name": "ProductUpdated",

"type": "event"

},

{

"inputs": [

{

"internalType": "uint256",

"name": "ProductId",

"type": "uint256"

},

{

"internalType": "string",

"name": "\_name",

"type": "string"

},

{

"internalType": "string",

"name": "\_description",

"type": "string"

},

{

"internalType": "uint256",

"name": "\_quantity",

"type": "uint256"

}

],

"name": "addProduct",

"outputs": [],

"stateMutability": "nonpayable",

"type": "function"

},

{

"inputs": [

{

"internalType": "uint256",

"name": "\_productId",

"type": "uint256"

}

],

"name": "getProductDetails",

"outputs": [

{

"internalType": "string",

"name": "name",

"type": "string"

},

{

"internalType": "string",

"name": "description",

"type": "string"

},

{

"internalType": "uint256",

"name": "quantity",

"type": "uint256"

},

{

"internalType": "address",

"name": "owner",

"type": "address"

}

],

"stateMutability": "view",

"type": "function"

},

{

"inputs": [],

"name": "productCount",

"outputs": [

{

"internalType": "uint256",

"name": "",

"type": "uint256"

}

],

"stateMutability": "view",

"type": "function"

},

{

"inputs": [

{

"internalType": "uint256",

"name": "",

"type": "uint256"

}

],

"name": "products",

"outputs": [

{

"internalType": "string",

"name": "name",

"type": "string"

},

{

"internalType": "string",

"name": "description",

"type": "string"

},

{

"internalType": "uint256",

"name": "quantity",

"type": "uint256"

},

{

"internalType": "address",

"name": "owner",

"type": "address"

}

],

"stateMutability": "view",

"type": "function"

},

{

"inputs": [

{

"internalType": "uint256",

"name": "\_productId",

"type": "uint256"

},

{

"internalType": "string",

"name": "\_name",

"type": "string"

},

{

"internalType": "string",

"name": "\_description",

"type": "string"

},

{

"internalType": "uint256",

"name": "\_quantity",

"type": "uint256"

}

],

"name": "updateProduct",

"outputs": [],

"stateMutability": "nonpayable",

"type": "function"

}

]

if (!window.ethereum) {

alert('Meta Mask Not Found')

window.open("https://metamask.io/download/")

}

Export const provider = new ethers.providers.Web3Provider(window.ethereum);

export const signer = provider.getSigner();

export const address = "0xE3E79289C9dcD24DD6F591C15bcD74FBA532d3A3"

export const contract = new ethers.Contract(address, abi, signer)

**home.js**

import React, { useState } from "react";

import { Button, Container, Row, Col } from 'react-bootstrap';

import 'bootstrap/dist/css/bootstrap.min.css';

import { contract } from "./connector";

function Home() {

const [Id, setId] = useState("");

const [Name, setName] = useState("");

const [Desc, setDesc] = useState("");

const [Qty, setQty] = useState("");

const [Ids, setIds] = useState("");

const [Names, setNames] = useState("");

const [Descs, setDescs] = useState("");

const [Qtys, setQtys] = useState("");

const [Wallet, setWallet] = useState("");

const [gId, setGIds] = useState("");

const [Details, setDetails] = useState("");

const handleId = (e) => {

setId(e.target.value)

}

const handleName = (e) => {

setName(e.target.value)

}

const handleDesc = (e) => {

setDesc(e.target.value)

}

const handleQty = (e) => {

setQty(e.target.value)

}

const handleAddProduct = async () => {

try {

let tx = await contract.addProduct(Id.toString(), Name, Desc, Qty)

let wait = await tx.wait()

alert(wait.transactionHash)

console.log(wait);

} catch (error) {

alert(error)

}

}

const handleIds = (e) => {

setIds(e.target.value)

}

const handleNames = (e) => {

setNames(e.target.value)

}

const handleDescs = (e) => {

setDescs(e.target.value)

}

const handleQtys = (e) => {

setQtys(e.target.value)

}

const handleUpdate = async () => {

try {

let tx = await contract.updateProduct(Ids.toString(), Names, Descs, Qtys)

let wait = await tx.wait()

console.log(wait);

alert(wait.transactionHash)

} catch (error) {

alert(error)

}

}

const handleGetIds = async (e) => {

setGIds(e.target.value)

}

const handleGetDetails = async () => {

try {

let tx = await contract.getProductDetails(gId.toString())

let arr = []

tx.map(e => {

arr.push(e)

})

console.log(tx);

setDetails(arr)

} catch (error) {

alert(error)

console.log(error);

}

}

const handleWallet = async () => {

if (!window.ethereum) {

return alert('please install metamask');

}

const addr = await window.ethereum.request({

method: 'eth\_requestAccounts',

});

setWallet(addr[0])

}

return (

<div>

<h1 style={{ marginTop: "30px", marginBottom: "80px" }}>Agriculture Registry</h1>

{!Wallet ?

<Button onClick={handleWallet} style={{ marginTop: "30px", marginBottom: "50px" }}>Connect Wallet </Button>

:

<p style={{ width: "250px", height: "50px", margin: "auto", marginBottom: "50px", border: '2px solid #2096f3' }}>{Wallet.slice(0, 6)}....{Wallet.slice(-6)}</p>

}

<Container>

<Row>

<Col style={{marginRight:"100px"}}>

<div>

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleId} type="number" placeholder="Product Id" value={Id} /> <br />

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleName} type="string" placeholder="Product Name" value={Name} /> <br />

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleDesc} type="string" placeholder="product description" value={Desc} /> <br />

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleQty} type="number" placeholder="product quantity" value={Qty} /> <br />

<Button onClick={handleAddProduct} style={{ marginTop: "10px" }} variant="primary"> Add Product</Button>

</div>

</Col>

<Col style={{ marginRight: "100px" }}>

<div>

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleIds} type="number" placeholder="Product Id" value={Ids} /> <br />

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleNames} type="string" placeholder="Product Name" value={Names} /> <br />

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleDescs} type="string" placeholder="product description" value={Descs} /> <br />

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleQtys} type="number" placeholder="product quantity" value={Qtys} /> <br />

<Button onClick={handleUpdate} style={{ marginTop: "10px" }} variant="primary"> Update Product</Button>

</div>

</Col>

</Row>

<Row>

<Col >

<div style={{ margin: "auto" , marginTop:"100px"}}>

<input style={{ marginTop: "10px", borderRadius: "5px" }} onChange={handleGetIds} type="number" placeholder="Enter Id" value={gId} /><br />

<Button onClick={handleGetDetails} style={{ marginTop: "10px" }} variant="primary">Get Product Details</Button>

{Details ? Details?.map(e => {

return <p>{e.toString()}</p>

}) : <p></p>}

</div>

</Col>

</Row>

</Container>

</div>

)

}

export default Home;

**App.js**

import './App.css';

import Home from './Page/Home'

function App() {

return (

<div className="App">

<header className="App-header">

<Home />

</header>

</div>

);

}

export default App;

**index.js**

import React from 'react';

import ReactDOM from 'react-dom/client';

import './index.css';

import App from './App';

import reportWebVitals from './reportWebVitals';

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(

<React.StrictMode>

<App />

</React.StrictMode>

);

// If you want to start measuring performance in your app, pass a function

// to log results (for example: reportWebVitals(console.log))

// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals

reportWebVitals();

**reportWebVitals.js**

const reportWebVitals = onPerfEntry => {

if (onPerfEntry && onPerfEntry instanceof Function) {

import('web-vitals').then(({ getCLS, getFID, getFCP, getLCP, getTTFB }) => {

getCLS(onPerfEntry);

getFID(onPerfEntry);

getFCP(onPerfEntry);

getLCP(onPerfEntry);

getTTFB(onPerfEntry);

});

}

};

export default reportWebVitals;

**setupTests.js**

import '@testing-library/jest-dom';

**GITHUB & PROJECT DEMO LINK**

* GitHub Link: https://github.com/jeyaram1811/NM2023TMID02822.git
* Demo link:https://drive.google.com/file/d/1DY7a\_l\_6XN1IxbtsYJubBZLxcqzDy3IU/view?usp=drive\_link