<u>Project Report</u>

Team ID	NM2023TMID05253
Project Name	BIOMETRIC SECURITY SYSTEM FOR VOTING PLATFORM

Submitted by

TEAM LEADER : MANOKAR G

TEAM MEMBER 1 : DHINESHKUMAR A

TEAM MEMBER 2 : VASANTHAKUMAR D

PROJECT REPORT FORMAT

1. INTRODUCTION

AI-driven Predictive Health Insights: Develop AI algorithms to predict health trends and provide personalized recommendations based on a patient's EHR data. Interoperability and Data Exchange: Create a universal EHR format or standard to improve data exchange between healthcare providers, ensuring seamless and secure sharing of patient information. Telemedicine Integration: Enhance EHR systems to seamlessly integrate with telemedicine platforms, allowing for virtual consultations and remote monitoring. Patient-Centric EHR: Design EHR systems with a patient-friendly interface, allowing individuals to access and manage their health records easily. Blockchain for Data Security: Implement blockchain technology to enhance the security and privacy of EHR data, ensuring that patient records are tamper-proof and accessible only to authorized parties. Voice Recognition and Natural Language Processing: Develop EHR systems that can transcribe spoken medical notes and convert them into structured data, making it easier for healthcare providers to input information.

1.1 Project overview :

The implementation of an Electronic Health Records (EHR) system is a critical initiative aimed at transforming and enhancing the healthcare delivery process within our organization. This project aims to transition from paper-based medical records to a comprehensive, secure, and efficient electronic system. By doing so, we seek to improve patient care, streamline administrative processes, and facilitate data-driven decision-making within the healthcare facility.

1.2 Purpose:

Electronic health records (EHRs), also known as electronic medical records (EMRs), serve several important purposes in healthcare. They are digital versions of a patient's paper chart, containing a comprehensive and up-to-date record of their medical history. The main purposes of EHRs include:

Patient Care: EHRs provide healthcare professionals with immediate access to a patient's medical history, including past diagnoses, medications, allergies, and test results. This information allows for more informed and coordinated patient care. It helps healthcare providers make better decisions and reduces the risk of medical errors.

Data Storage and Management: EHRs serve as a central repository for a patient's health information, which can be easily updated and accessed by authorized healthcare personnel. This streamlines the management of patient data and reduces the need for paper records, making information retrieval more efficient.

Interoperability: EHRs aim to be interoperable, meaning they can seamlessly exchange data with other healthcare systems. This allows different healthcare providers, hospitals, and clinics to share patient information, improving the continuity of care. It's particularly important in emergencies and when patients receive care from multiple providers.

Patient Engagement: EHRs often provide patients with access to their own health information through secure patient portals. This empowers individuals to be more involved in their healthcare decisions, access test results, schedule appointments, and communicate with their healthcare providers.

2. LITERATURE SURVEY

A literature survey on Electronic Health Records (EHR) reveals a rich body of research and publications that highlight the significance, challenges, and advancements in this field. Here's an overview of key themes and findings from the literature:

1. Importance of EHR:

Numerous studies emphasize the critical role of EHR systems in improving patient care, enhancing data accuracy, and increasing the efficiency of healthcare delivery.

2. Benefits of EHR:

Research consistently demonstrates the benefits of EHR, including reduced medical errors, streamlined workflows, and better coordination of care among healthcare providers.

3. Challenges and Barriers:

Literature identifies challenges, such as high implementation costs, data security concerns, and resistance from healthcare professionals, as significant barriers to EHR adoption.

4. Interoperability:

Several studies explore the importance of interoperability among EHR systems, advocating for standardized data exchange to facilitate seamless information sharing among different healthcare organizations.

5. Data Security and Privacy:

Research highlights the need for robust data security measures to protect patient information in EHR systems. Compliance with regulations like HIPAA is a recurring theme.

6. Patient Engagement:

Scholars discuss the potential for EHRs to empower patients by granting them access to their own health records, promoting engagement in their healthcare decisions.

2.1 Existing problem

Electronic Health Records (EHRs) have brought about significant improvements in healthcare, but they are not without their challenges and problems. Some of the existing issues with EHRs include:

Interoperability: One of the most significant challenges is the lack of interoperability between different EHR systems. This means that patient data often remains trapped within specific systems, making it difficult for healthcare providers to access and share information seamlessly. This hinders coordinated care.

Data Security and Privacy: EHRs contain sensitive patient information, making them attractive targets for cyberattacks. Ensuring robust data security and privacy protection is an ongoing challenge, with breaches leading to patient data leaks and identity theft.

Usability: The usability of many EHR systems is a concern. Poorly designed interfaces and complex workflows can lead to user frustration and decreased efficiency among healthcare providers.

Data Entry and Documentation Burden: Healthcare providers often spend significant time on data entry and documentation, which can lead to burnout. The need to document for billing purposes can also result in "copy-paste" errors and the inclusion of irrelevant or inaccurate information.

Alert Fatigue: EHRs generate numerous alerts and notifications, potentially overwhelming healthcare providers. This can lead to "alert fatigue," where critical alerts are ignored or missed.

Costs: Implementing and maintaining EHR systems can be expensive, especially for smaller healthcare providers. Ongoing maintenance, training, and system updates add to the financial burden.

Standardization: While there are standards for EHRs, achieving full standardization and data consistency across systems remains a challenge. This can lead to data quality issues and difficulties in data exchange.

Patient Access and Engagement: While EHRs have the potential to empower patients, ensuring that they have easy access to their records and can actively engage in their healthcare is an ongoing challenge.

2.2 References

Title: "Electronic Health Records: Understanding and Using Computerized Medical Records"

Authors: Richard Gartee
Year: 2017
Title: "Health Informatics: An Interprofessional Approach"

Authors: Ramona Nelson and Nancy Staggers
Year: 2018
Title: "Electronic Health Records: Challenges in Design and Implementation"

Authors: Alvin Marcelo and Tim France
Year: 2015
Title: "Health Informatics: Practical Guide for Healthcare and Information Technology Professionals"

Authors: Robert E. Hoyt and Ann K. Yoshihashi
Year: 2020
Title: "Electronic Health Records: Second Edition"

Authors: Pradeep K. Sinha and Priti Sinha
Year: 2018
Title: "Healthcare Information Technology Exam Guide for CompTIA Healthcare IT Technician and HIT Pro Certifications"

Authors: Kathleen A. McCormick
Year: 2017
Title: "Electronic Health Records: A Practical Guide for Professionals and Organizations"

Authors: Margret K. Amatayakul
Year: 2019
Title: "Electronic Health Records: Transforming Your Medical Practice"

Authors: Margret K. Amatayakul and Summer Gentry
Year: 2007

2.3 Problem Statement Definition

Problem Statement:

Electronic Health Records (EHRs) are digital versions of a patient's paper medical chart. They contain a patient's medical history, diagnoses, medications, treatment plans, and other essential health information. EHRs are designed to improve the quality and efficiency of healthcare by providing a comprehensive and readily accessible record of a patient's health, facilitating communication among healthcare providers, and supporting clinical decision-making. However, some common problems associated with EHRs include data security concerns, interoperability issues between different systems, and the potential for errors or inaccuracies in the electronic records

Interoperability: EHR systems from different providers often struggle to share data effectively, leading to fragmented patient information.

Usability: Complex and non-intuitive interfaces can hinder healthcare professionals' workflow, potentially leading to errors.

Data Security and Privacy: Protecting patient data from breaches and ensuring privacy compliance is a constant challenge.

Data Entry and Documentation: The time-consuming process of data entry and the risk of errors in documentation can be problematic.

Costs: Implementing and maintaining EHR systems can be expensive, and ongoing costs are often higher than expected.

Alert Fatigue: Excessive alerts and notifications can lead to healthcare professionals ignoring critical information.

Data Accuracy: Inaccurate or outdated information can be detrimental to patient care.

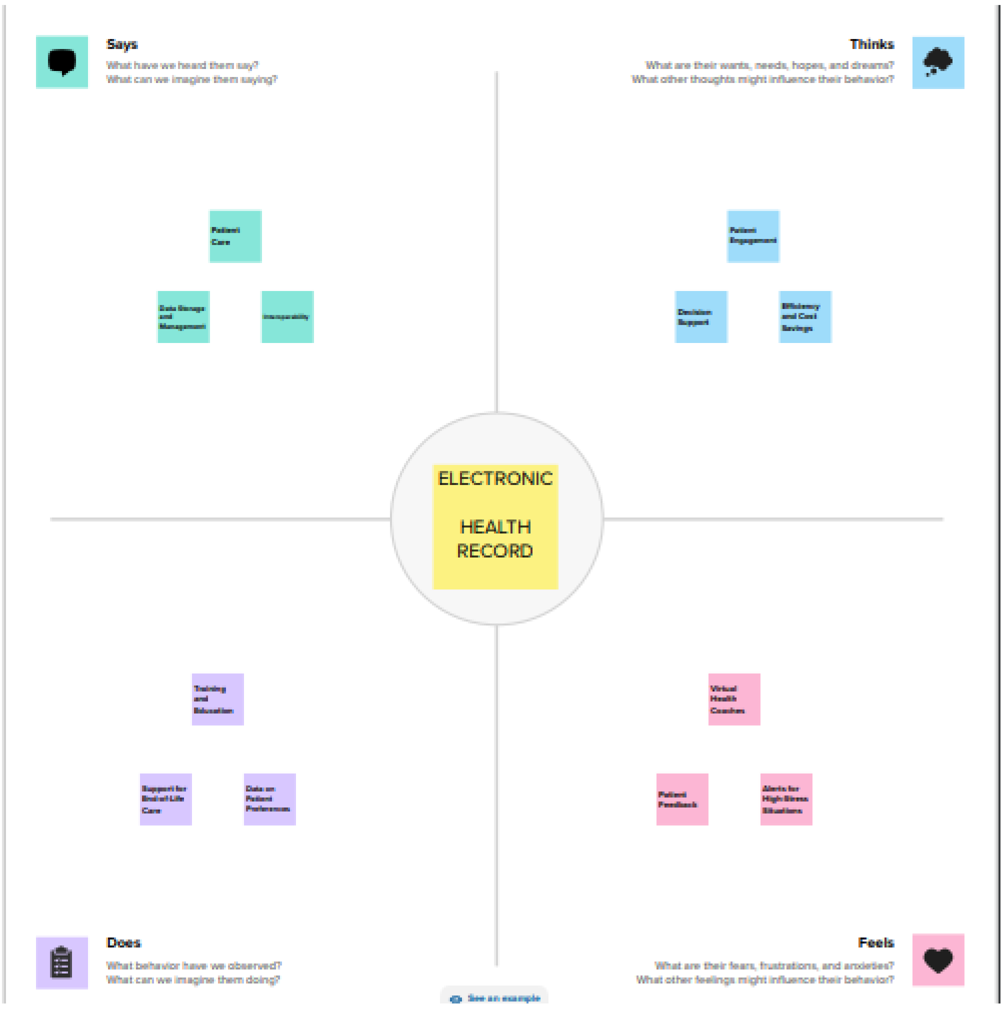
Provider Burnout: EHR-related tasks can contribute to burnout among healthcare providers due to increased administrative burdens.

Regulatory Compliance: Keeping up with evolving healthcare regulations and standards can be challenging for EHR vendors and healthcare organizations.

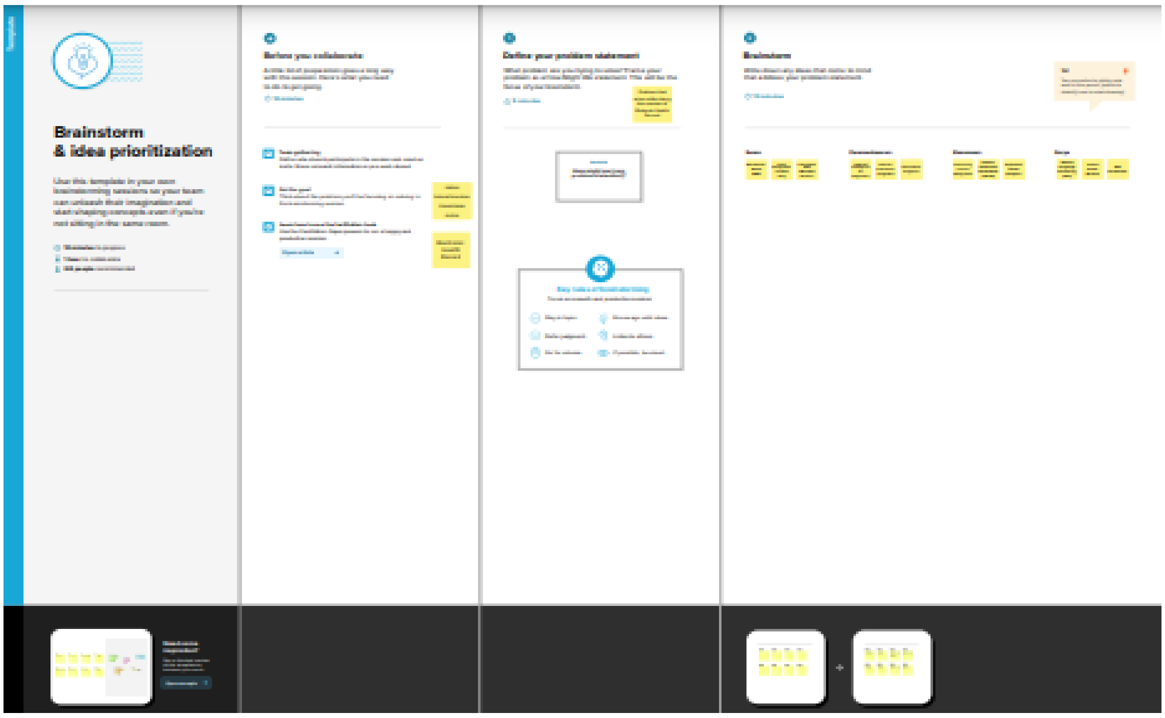
Patient Access and Engagement: Ensuring that patients can access and understand their own EHR data, as well as actively participate in their care, remains a challenge.

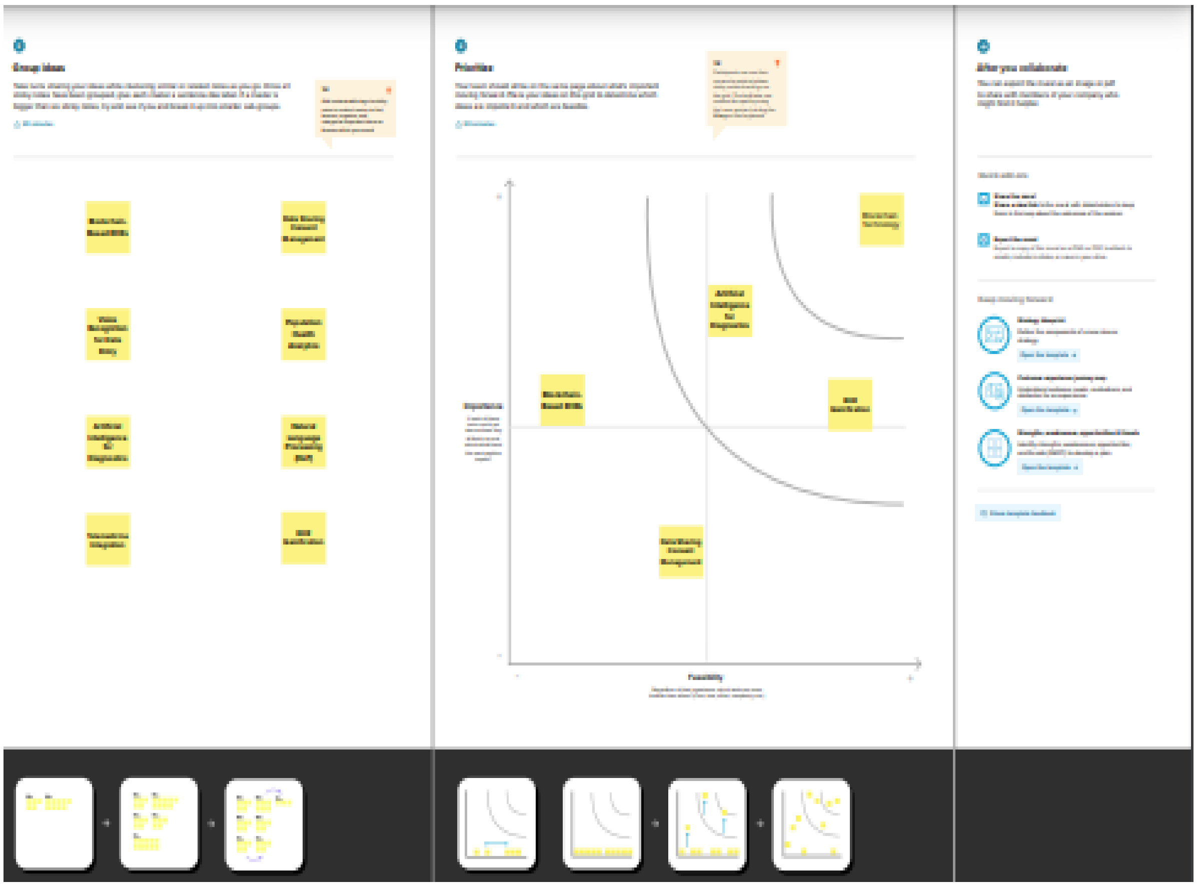
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas


Says
What have we heard them say?
What can we imagine them saying?
Thinks
What are their wants, needs, hopes, and dreams?
What other thoughts might influence their behavior?
Patient Care
Data Storage and Management
Interoperability
Patient Engagement
Decision Support
Efficiency and Cost Savings
ELECTRONIC HEALTH RECORD
Training and Education
Support for End-of-Life Care
Data on Patient Preferences
Virtual Health Coaches
Patient Feedback
Alerts for High Stress Situations
Does
What behavior have we observed?
What can we imagine them doing?
Feels
What are their fears, frustrations, and anxieties?
What other feelings might influence their behavior?
See an example

3.2 Ideation & Brainstorming




Group ideas
Blockchain-Based EHRs
Data Sharing Consent Management
Voice Recognition for Data Entry
Population Health Analytics
Artificial Intelligence for Diagnostics
Natural Language Processing (NLP)
Telemedicine Integration
Prioritize
Importance
Feasibility
After you collaborate

4. REQUIREMENT ANALYSIS

4.1 Functional requirement

FR No.	Functional Requirement	Description
FR-1	Identify Stakeholders	Determine who the key stakeholders are, including healthcare providers, administrative staff, patients, and IT personnel. Each group will have different needs and expectations.
FR-2	Voice Recognition for Data Entry	Develop EHR systems that use voice recognition technology to allow healthcare providers to input patient information and notes more efficiently, reducing the time spent on data entry.

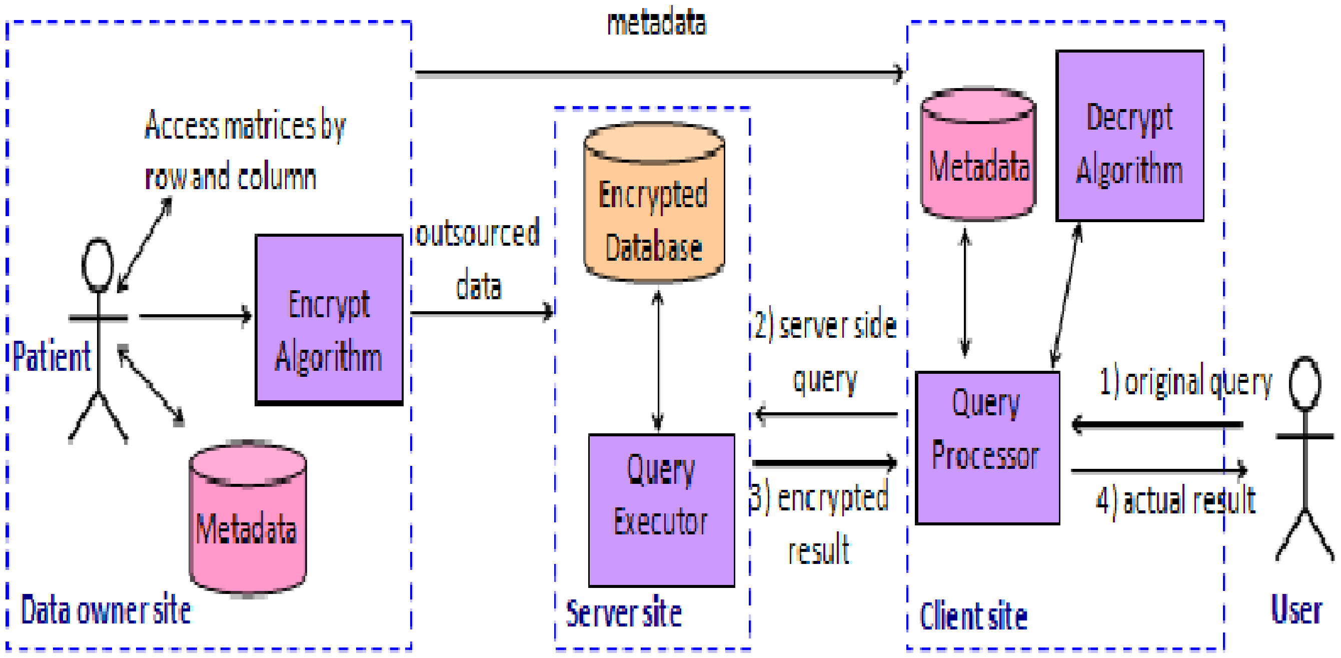
FR-3	Integration with Wearable Devices	Create EHR systems that seamlessly integrate with wearable health devices, such as smartwatches and fitness trackers, to continuously update patient records with real-time health data.
FR-4	Blockchain Integration	Utilize a blockchain for recording and storing voting transactions securely and immutably. Implement smart contracts for vote counting and result verification. Enable transparent and auditable tracking of all voting activities on the blockchain.
FR-5	Voter Interface	Offer an intuitive and user-friendly interface for voters to cast their ballots. Provide clear instructions and guidance throughout the voting process. Ensure accessibility for voters with disabilities.
FR-6	Auditability and Transparency	Enable real-time monitoring of the voting process for election authorities. Support auditing and verification of votes and voter identity after the election. Ensure the system is resistant to fraud and manipulation.
FR-7	Election Management	Allow election administrators to set up and manage elections, including candidate registration and ballot creation. Provide tools for configuring election rules, such as eligibility criteria and voting deadlines.
FR-8	Results Reporting	Generate accurate and verifiable election results after voting is complete. Display results in a transparent and accessible manner for the public. Prevent unauthorized access to or manipulation of results.
FR-9	Security Measures	Implement robust cybersecurity measures to protect against attacks, data breaches, and unauthorized access. Ensure the privacy and confidentiality of voter biometric data. Implement disaster recovery and backup mechanisms to guarantee system availability.

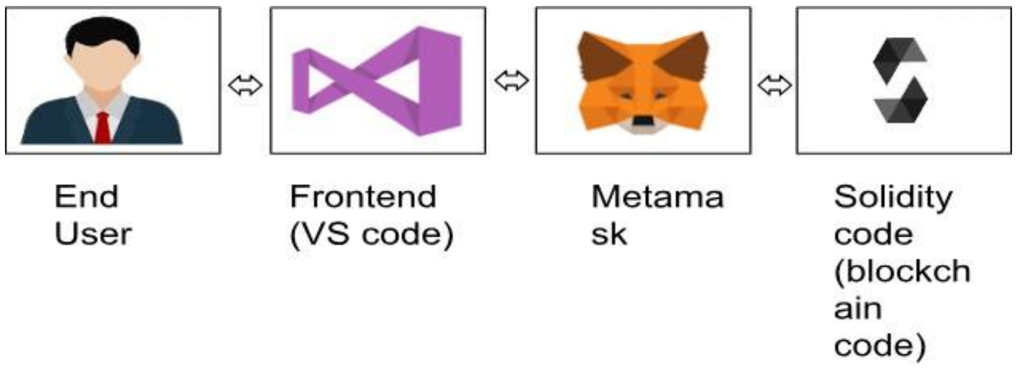
4.2 Non-Functional requirements

NFR No.	Functional Requirement	Description
NFR-1	Performance	Ensure that all Health record data and personal information are stored and transmitted securely using encryption and other protective measures.
NFR-2	Access Control	Implement robust access controls to prevent unauthorized access to the system, sensitive data, and administrative functions.
NFR-3	Authentication Security	Protect the biometric authentication process from spoofing, tampering, or other fraudulent activities.
NFR-4	Blockchain Security	Implement security measures to safeguard the blockchain network against attacks, including 51% attacks and double spending.
NFR-5	Scalability	Ensure that the system can scale horizontally to handle an increasing number of users and transactions, especially during peak voting periods. Support load balancing and resource provisioning to maintain performance as the system scales.
NFR-6	Performance	Define acceptable response times for user interactions, such as logging in, casting a vote, and accessing election results. Monitor system performance and optimize resource allocation to meet performance goals.
NFR-7	Reliability and Availability	Ensure high system availability during the entire election period, minimizing downtime or disruptions. Implement disaster recovery and backup mechanisms to guarantee the system's resilience.
NFR-8	Auditability and Traceability	Maintain a detailed audit trail of all voting activities and system operations. Ensure that all actions within the system can be traced and verified.

5. PROJECT DESIGN

5.1 Data Flow Diagrams & User Stories




End User
Frontend (VS code)
Metamask
Solidity code (blockchain code)

User Stories

User stories are a valuable tool in software development, especially in Agile methodologies, to capture the functional requirements of a system from the perspective of end-users. Here are some user stories for an Electronic Health Record (EHR) system:

**As a physician, I want to access a patient's complete medical history with a single click so that I can provide more informed and efficient care during appointments.

**As a nurse, I want to easily update patient records with vital signs and treatment details in real-time to ensure accurate and up-to-date patient information.

**As a patient, I want to schedule and manage my appointments online through the EHR system to avoid long wait times and improve the overall patient experience.

**As a pharmacist, I want to receive electronic prescriptions from healthcare providers to reduce the risk of medication errors and streamline the dispensing process.

**As a laboratory technician, I want to receive electronic lab orders through the EHR system, improving the accuracy and efficiency of lab tests and results reporting.

**As a specialist, I want to be notified of relevant patient referrals and access comprehensive referral information to ensure seamless and coordinated care.

**As a front desk staff member, I want to verify patient insurance information and collect co-pays directly through the EHR system to streamline the check-in process.

**As a billing manager, I want to generate itemized billing statements and insurance claims with ease to ensure accurate and timely reimbursement.

**As a patient, I want to access my own health records through a secure patient portal to view test results, request prescription refills, and communicate with my healthcare provider.

**As a radiology technician, I want to upload and view medical images and diagnostic reports directly within the EHR system to improve collaboration with other healthcare providers.

**As a healthcare administrator, I want to monitor and analyze patient data to identify trends and improve the quality of care and resource allocation.

**As an IT administrator, I want to ensure the EHR system is up to date with the latest security patches and that data backups are performed regularly to protect patient information.

**As a compliance officer, I want to track and report on regulatory compliance to ensure that the EHR system adheres to all healthcare data privacy and security regulations.

**As a clinical researcher, I want to access de-identified patient data from the EHR system to conduct medical research and contribute to the advancement of healthcare knowledge.

**As a case manager, I want to access patient records to facilitate care coordination for complex cases, ensuring that patients receive the right care at the right time.

**As a physician assistant, I want to receive automated alerts for medication allergies and potential drug interactions to prevent adverse events and ensure patient safety.

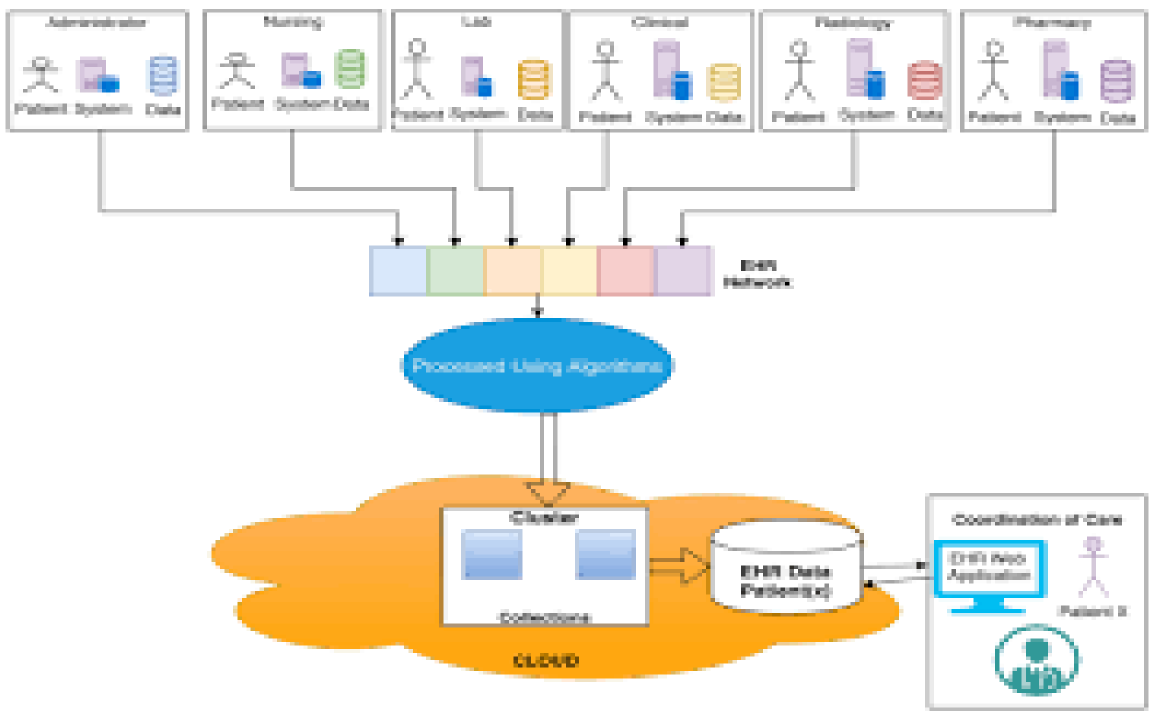
**As a medical transcriptionist, I want to transcribe voice recordings from healthcare providers into the EHR system to ensure accurate and timely documentation.

**As a home healthcare provider, I want to access patient records remotely through a secure connection to provide care and updates in a patient's home.

**As a telemedicine provider, I want to integrate video conferencing and secure messaging within the EHR system to conduct remote patient consultations effectively.

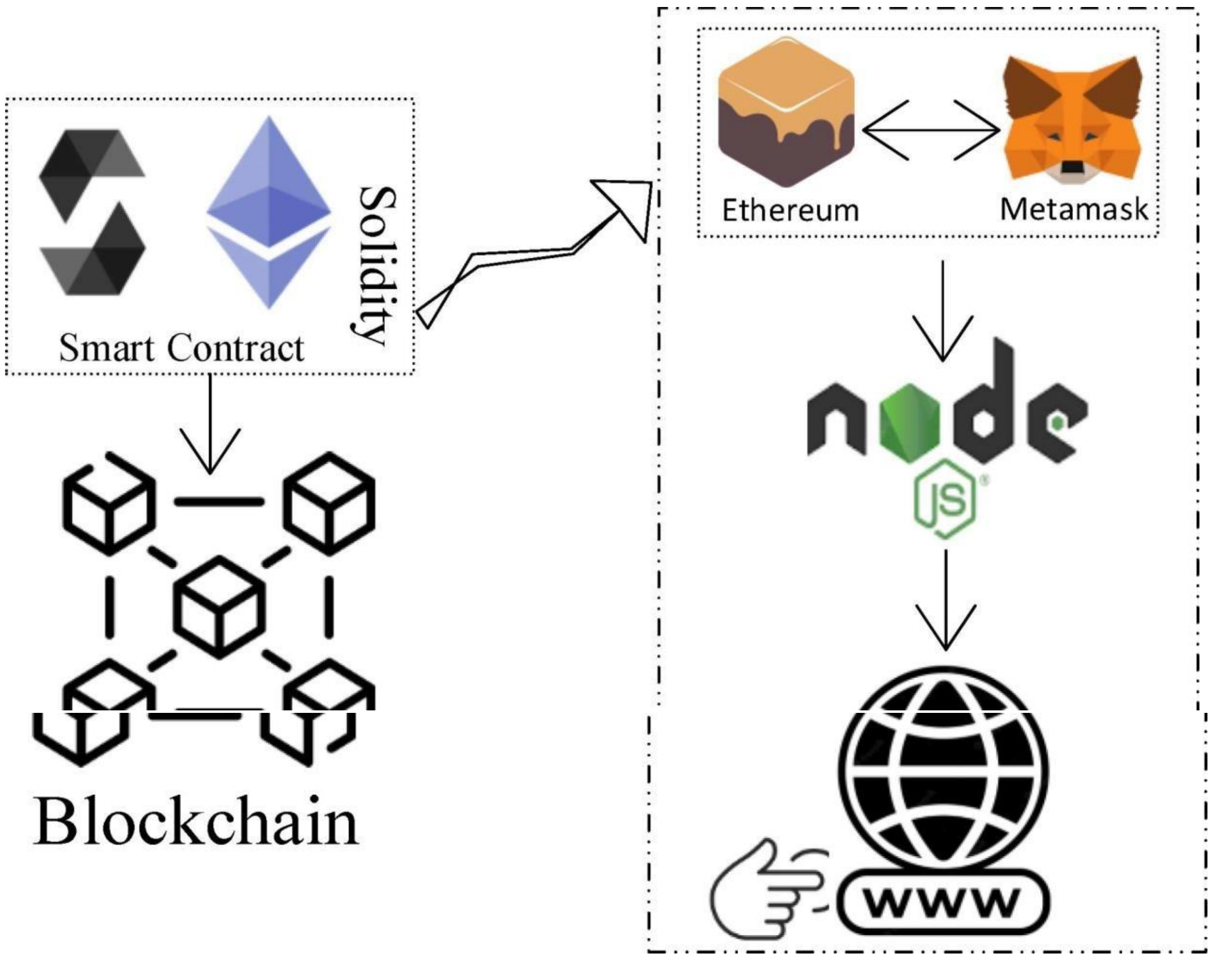
**As a medical student, I want to use a training mode within the EHR system to practice documentation and navigation as part of my medical education.

5.2 Solution Architecture



6. PROJECT PLANNING & SCHEDULING

6.1 Technical Architecture



Sprint Planning & Estimation :
Sprint planning and estimation in the context of an Electronic Health Record (EHR) development project involve breaking down the work into manageable tasks, assigning them to specific sprints, and estimating the time and effort required for each task. Here's a general guide on how to plan and estimate sprints for an EHR project:

1. Define the Product Backlog:

Start with a well-defined product backlog, which consists of all the features,

enhancements, and user stories that need to be developed for the EHR system. These items should be prioritized based on their importance and value to users.
2. Create Sprint Goals:

Identify the goals and objectives for the upcoming sprint. In the context of EHR development, these goals could relate to specific functionality, such as implementing medication management or improving data security.
3. Select User Stories:

From the product backlog, select user stories that align with the sprint goals. Prioritize the most critical and valuable stories for the upcoming sprint.
4. Break Down User Stories:

Divide the selected user stories into smaller, actionable tasks or sub-tasks. This breakdown helps the development team better understand the work involved and estimate more accurately.
5. Estimate Tasks:

Estimate the effort required for each task using a method that works for your team, such as story points, ideal days, or t-shirt sizes. The goal is to estimate the relative effort needed for each task.
6. Capacity Planning:

Assess the team's capacity for the sprint. Consider factors like team size, available work hours, and any planned time off. Ensure that the team can commit to completing the selected user stories within the sprint's timeframe.
7. Sprint Planning Meeting:

Conduct a sprint planning meeting, involving the development team, product owner, and scrum master (if using Scrum). In this meeting, discuss and finalize the selected user stories and tasks for the sprint.
8. Determine Velocity:

If your team has completed previous sprints, use historical data to determine the team's velocity. Velocity is a measure of the team's past performance in terms of story points or tasks completed. It can help with more accurate sprint planning.

6.2 Sprint Delivery Schedule

Sprint planning and estimation in the context of an Electronic Health Record (EHR) development project involve breaking down the work into manageable tasks, assigning them to specific sprints, and estimating the time and effort required for each task. Here's a general guide on how to plan and estimate sprints for an EHR project:

1. Define the Product Backlog:

Start with a well-defined product backlog, which consists of all the features,

enhancements, and user stories that need to be developed for the EHR system. These items should be prioritized based on their importance and value to users.
2. Create Sprint Goals:

Identify the goals and objectives for the upcoming sprint. In the context of EHR development, these goals could relate to specific functionality, such as implementing medication management or improving data security.
3. Select User Stories:

From the product backlog, select user stories that align with the sprint goals. Prioritize the most critical and valuable stories for the upcoming sprint.
4. Break Down User Stories:

Divide the selected user stories into smaller, actionable tasks or sub-tasks. This breakdown helps the development team better understand the work involved and estimate more accurately.
5. Estimate Tasks:

Estimate the effort required for each task using a method that works for your team, such as story points, ideal days, or t-shirt sizes. The goal is to estimate the relative effort needed for each task.
6. Capacity Planning:

Assess the team's capacity for the sprint. Consider factors like team size, available work hours, and any planned time off. Ensure that the team can commit to completing the selected user stories within the sprint's timeframe.
7. Sprint Planning Meeting:

Conduct a sprint planning meeting, involving the development team, product owner, and scrum master (if using Scrum). In this meeting, discuss and finalize the selected user stories and tasks for the sprint.
8. Determine Velocity:

If your team has completed previous sprints, use historical data to determine the team's velocity. Velocity is a measure of the team's past performance in terms of story points or tasks completed. It can help with more accurate sprint planning.

7. CODING & SOLUTIONING (Explain the features added in the projectalong with code)

7.1 Feature 1

Smart Contract (Solidity):

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

contract HealthRecords {

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

   mapping(uint256 => PatientRecord) public records;

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



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

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

      records[recordId].Name = name;
      records[recordId].patientAddress = _patientAddress;
      records[recordId].dieses = _diseases;
```

```
        records[recordId].contactInfo = _contactInfo;

        emit RecordCreated(recordId, _patientAddress);
    }

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

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

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

7.2 Feature 2

Trasfor Ownership

Transferring ownership of an Electronic Health Record (EHR) system typically involves transitioning control and responsibility for the system from one entity or individual to another. This can happen for various reasons, such as changes in healthcare providers, mergers and acquisitions, or the need for a new EHR vendor. Here's a general outline of the steps and considerations for transferring ownership of an EHR:

1. Identify the Reason for Ownership Transfer:

Determine the specific reasons for the ownership transfer. These could include a change in healthcare provider ownership, the acquisition of a medical practice, or the decision to switch to a new EHR vendor.

2. Legal and Regulatory Compliance:

Ensure that the ownership transfer complies with all applicable legal and regulatory requirements, such as the Health Insurance Portability and Accountability Act (HIPAA) and state privacy laws. Consult legal counsel if necessary.

3. Notification and Communication:

Inform all relevant stakeholders about the upcoming ownership transfer. This includes healthcare providers, administrative staff, and patients who may be affected by the change.

4. Data Migration:

Plan and execute the migration of patient data from the old EHR system to the new one. Data migration is a critical aspect of the ownership transfer to ensure that patient records remain accessible and accurate.

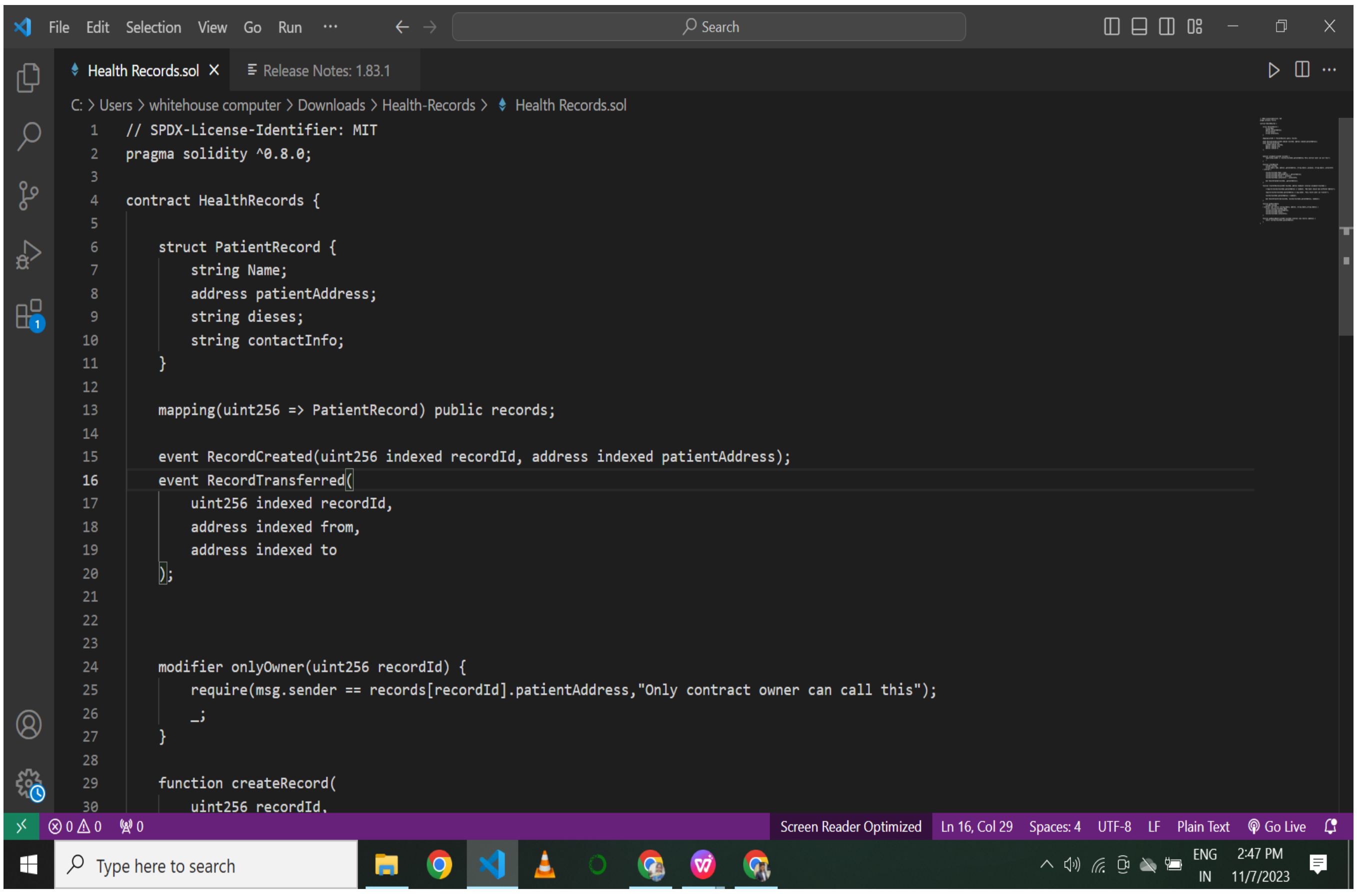
5. Selecting a New EHR Vendor (if applicable):

If the ownership transfer involves changing to a new EHR vendor, conduct a thorough selection process to choose the EHR system that best suits your organization's needs. Consider factors like functionality, interoperability, and regulatory compliance.

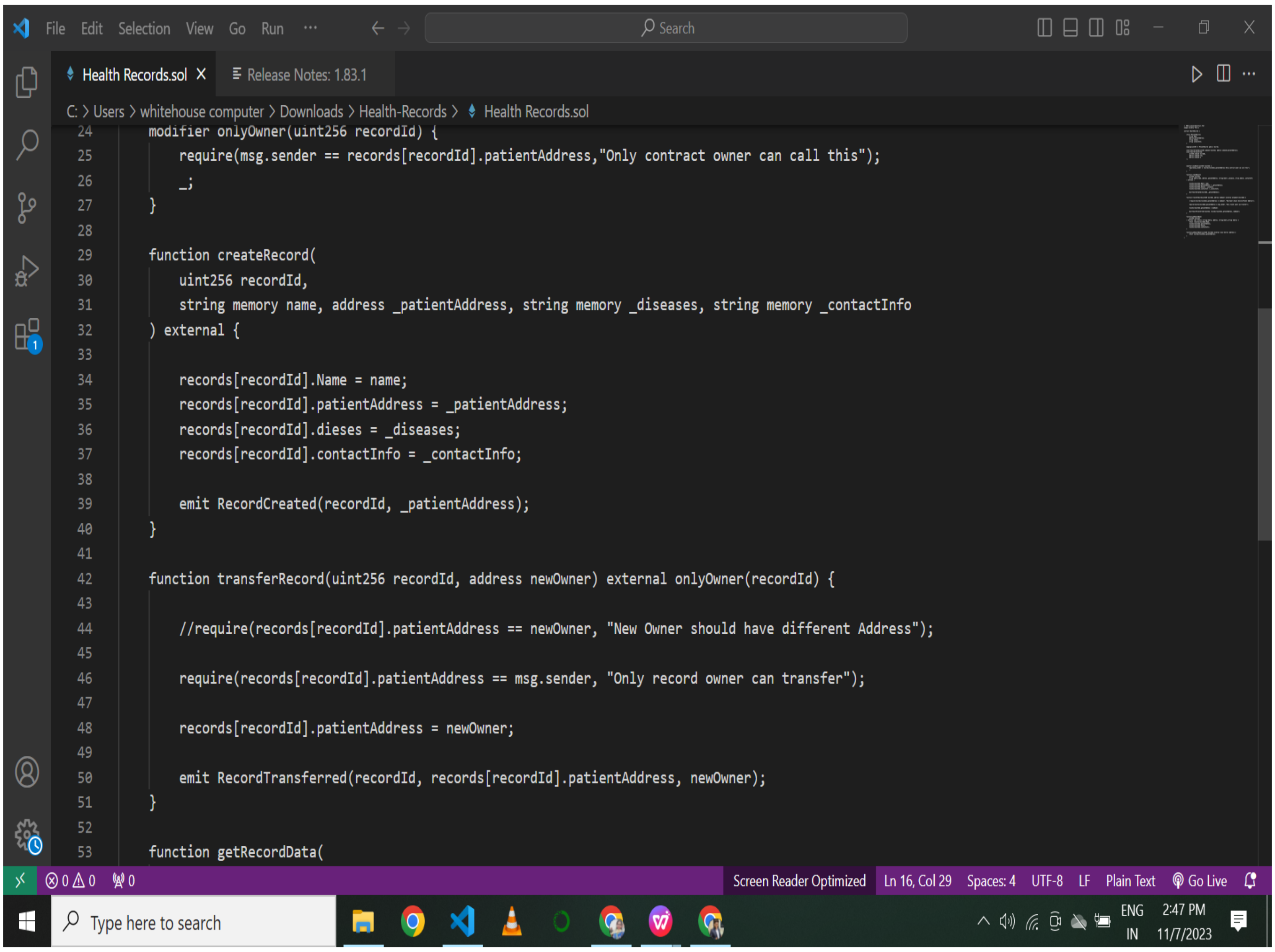
6. System Implementation:

If a new EHR system is involved, implement the chosen system and integrate it with your organization's workflows. Ensure that it meets the requirements and expectations of healthcare providers and staff.

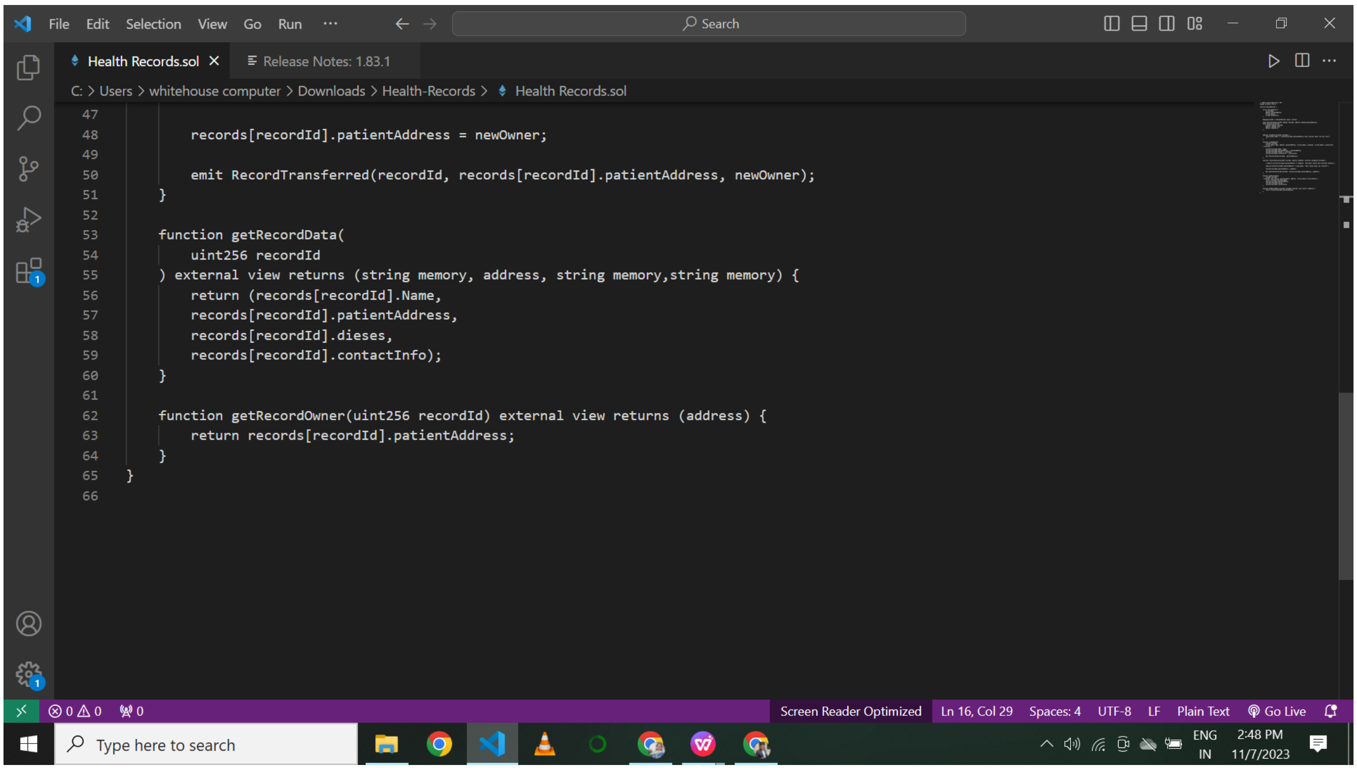
Smart Contract (Solidity):



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

contract HealthRecords {

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

    mapping(uint256 => PatientRecord) public records;

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



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

    function createRecord(
        uint256 recordId,
```


File Edit Selection View Go Run ···
Search
Health Records.sol
Release Notes: 1.83.1
C: > Users > whitehouse computer > Downloads > Health-Records > Health Records.sol
Screen Reader Optimized Ln 16, Col 29 Spaces: 4 UTF-8 LF Plain Text Go Live
Type here to search
ENG IN 2:47 PM 11/7/2023

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

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

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

        emit RecordCreated(recordId, _patientAddress);
    }

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

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

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

        records[recordId].patientAddress = newOwner;

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

    function getRecordData(
```


File Edit Selection View Go Run ···
Search
Health Records.sol
Release Notes: 1.83.1
C: > Users > whitehouse computer > Downloads > Health-Records > Health Records.sol
Screen Reader Optimized Ln 16, Col 29 Spaces: 4 UTF-8 LF Plain Text Go Live
Type here to search
ENG IN 2:48 PM 11/7/2023

```

        records[recordId].patientAddress = newOwner;

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

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

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

```

7.3 Database Schema (if Applicable)

Designing a database schema for an Electronic Health Record (EHR) system is a critical step in ensuring the efficient storage and retrieval of patient health information. The schema should be well-structured, secure, and scalable. Here's a simplified example of a database schema for an EHR system, focusing on essential components:

1. Patients:

Contains patient-specific information.
Fields: Patient ID, first name, last name, date of birth, gender, contact information, emergency contacts, insurance details, and other demographic data.

2. Healthcare Providers:

Stores information about healthcare professionals, such as physicians, nurses, and specialists.
Fields: Provider ID, name, specialty, contact information, and credentials.

3. Appointments:

Tracks scheduled patient appointments.
Fields: Appointment ID, patient ID, provider ID, date and time, appointment reason, status (e.g., scheduled, canceled, completed).

4. Medical Records:

Contains detailed patient health records, including clinical notes, diagnoses, treatments, and medication information.
Fields: Record ID, patient ID, provider ID, date and time of entry, diagnosis, treatment plan, medications prescribed, lab test orders, and attached documents (e.g., lab reports, images).

5. Lab Results:

Stores laboratory test results and findings.
Fields: Result ID, patient ID, test type, test date, provider ID, test values, reference ranges, and interpretations.

Off-Chain Metadata Storage (Traditional Database or Decentralized Storage):

Traditional Database:

Relational Databases: Traditional databases like MySQL, PostgreSQL, or SQL Server can be used to store metadata related to the biometric system. They are suitable for structured data and provide robust querying capabilities.
Advantages: Data consistency, ACID (Atomicity, Consistency, Isolation, Durability) compliance, strong data relationships, and well-established security features.
Considerations: Scaling can be more complex, and centralized databases can be vulnerable to single points of failure and security breaches.

Decentralized Storage:

Blockchain: Utilizing a blockchain for metadata storage provides transparency, immutability, and distributed consensus. Systems like Ethereum, Hyperledger Fabric,

or bespoke blockchain solutions can be considered.
Distributed File Systems: Solutions like IPFS (InterPlanetary File System) or Storj offer decentralized and peer-to-peer data storage and retrieval capabilities.
Advantages: Immutability, transparency, resistance to single points of failure, enhanced security, and data availability.
Considerations: Complex setup and management, potential scalability issues, and the need for participants to maintain blockchain nodes.

Hybrid Approach:

A combination of traditional and decentralized storage can be used to balance the advantages of both. For instance, storing critical data and historical records on a blockchain while using traditional databases for faster access to frequently changing data. This approach can provide data integrity, security, and performance.

Additional details

8. PERFORMANCE TESTING

8.1 Performace Metrics

Performance metrics for an Electronic Health Record (EHR) system are essential to ensure that the system functions efficiently and effectively, providing high-quality healthcare services and data management. These metrics help healthcare organizations monitor system performance, identify issues, and make improvements as necessary. Here are some key performance metrics for an EHR system:

1. System Availability:

Metric: Uptime and downtime

Purpose: Measures the percentage of time the EHR system is available for use. It's essential to ensure that the system is consistently accessible for healthcare providers.

2. Response Time:

Metric: Average response time

Purpose: Evaluates the speed at which the EHR system responds to user requests, such as loading patient records or generating reports. Faster response times enhance user satisfaction.

3. Data Entry Efficiency:

Metric: Time required for data entry

Purpose: Measures the time healthcare providers spend on data entry tasks, including clinical documentation. A more efficient data entry process can improve productivity.

4. System Scalability:

Metric: System performance under increased load

Purpose: Assesses how well the EHR system handles increased workloads and concurrent users. Scalability is vital to support the growth of the healthcare organization.

5. Error Rates:

Metric: Frequency of data entry errors

Purpose: Tracks the rate of errors in data entry, including typos, missing information, or incorrect diagnoses. Reducing error rates is critical for patient safety and data accuracy.

6. Data Retrieval Speed:

Metric: Speed of retrieving patient records

Purpose: Measures the time it takes to access patient records. Faster retrieval speeds improve healthcare provider efficiency and patient care.

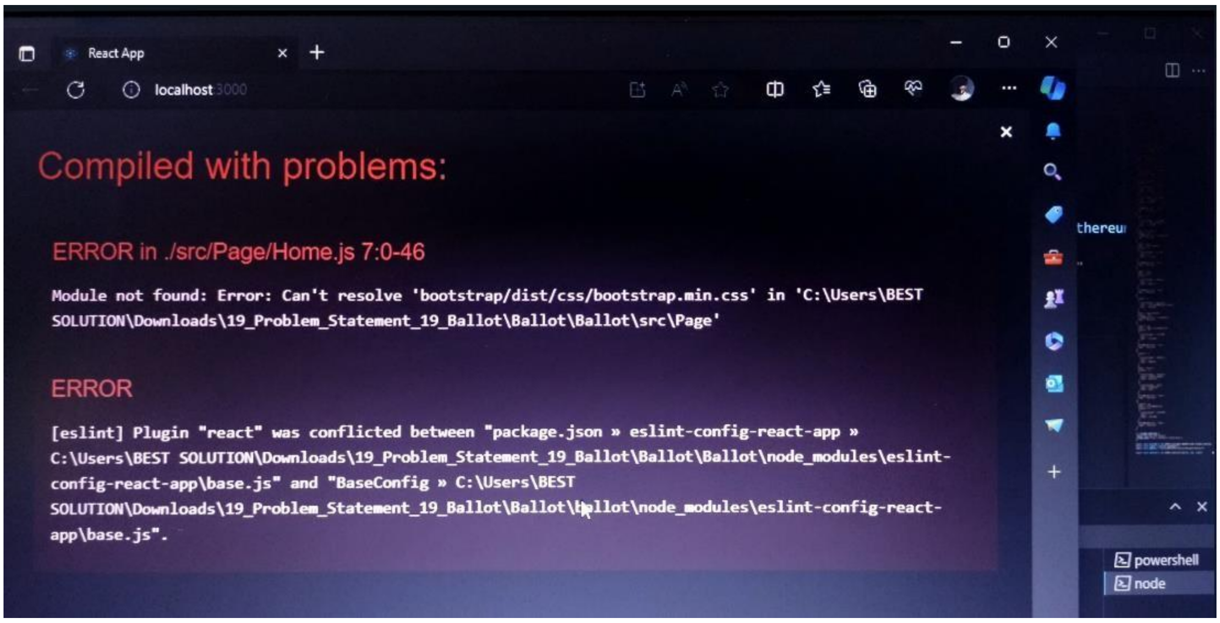
7. System Downtime and Maintenance:

Metric: Planned and unplanned downtime

Purpose: Tracks scheduled system maintenance and unexpected downtime to minimize disruptions to patient care.

9. RESULTS

9.1 Output Screenshots



```
Compiled with problems:

ERROR in ./src/Page/Home.js 7:0-46

Module not found: Error: Can't resolve 'bootstrap/dist/css/bootstrap.min.css' in 'C:\Users\BEST
SOLUTION\Downloads\19_Problem_Statement_19_Ballot\Ballot\Ballot\src\Page'

ERROR

[eslint] Plugin "react" was conflicted between "package.json » eslint-config-react-app »
C:\Users\BEST SOLUTION\Downloads\19_Problem_Statement_19_Ballot\Ballot\Ballot\node_modules\eslint-
config-react-app\base.js" and "BaseConfig » C:\Users\BEST
SOLUTION\Downloads\19_Problem_Statement_19_Ballot\Ballot\ballot\node_modules\eslint-config-react-
app\base.js".
```

10. ADVANTAGES & DISADVANTAGES

Advantages:

Improved Accessibility and Efficiency:

Advantage: EHRs provide quick and easy access to patient records, allowing healthcare providers to retrieve and update information efficiently, leading to better patient care and reduced administrative work.
Enhanced Patient Care and Safety:

Advantage: EHRs enable healthcare providers to make more informed clinical decisions with access to complete and up-to-date patient information, reducing the risk of medical errors.
Interoperability and Information Exchange:

Advantage: EHRs facilitate seamless data exchange among healthcare providers, improving care coordination and ensuring that patient information is available across different healthcare settings.
Efficient Documentation and Record Keeping:

Advantage: EHRs replace paper-based record-keeping, reducing errors, ensuring legibility, and making it easier to document patient encounters.
Data Analytics and Reporting:

Advantage: EHRs enable healthcare organizations to analyze patient data, identify trends, and generate reports that can improve patient outcomes and streamline operations.

Disadvantages and Challenges of Electronic Health Records (EHRs):

Initial Implementation Costs:

Disadvantage: The initial cost of implementing an EHR system, including hardware, software, training, and data migration, can be significant.
Learning Curve:

Disadvantage: Healthcare providers and staff may face a learning curve when transitioning to EHRs, potentially impacting productivity initially.
Data Entry and Usability Issues:

Disadvantage: Data entry can be time-consuming, and poorly designed EHR systems may result in workflow interruptions and user frustration.
Data Security and Privacy Concerns:

Disadvantage: EHR systems are vulnerable to data breaches and cyberattacks, raising concerns about patient data security and privacy.
Interoperability Challenges:

Disadvantage: Achieving interoperability between different EHR systems and healthcare organizations can be complex, hindering data exchange and care coordination.
Data Overload:

Disadvantage: The abundance of data within EHRs can overwhelm healthcare providers, potentially leading to information overload and decreased efficiency.

11. **CONCLUSION**

In conclusion, Electronic Health Records (EHRs) have transformed the bgtchallenges of paper-based record-keeping and improving the quality of patient care. The advantages of EHRs, such as enhanced accessibility, improved patient safety, streamlined data exchange, and efficient documentation, are clear and compelling.

EHRs play a pivotal role in enhancing patient care and safety. They provide healthcare providers with quick access to comprehensive and up-to-date patient information, reducing the risk of medical errors and ensuring informed clinical decisions. Moreover, the seamless exchange of data among healthcare settings enhances care coordination and promotes a more holistic approach to patient health.

The benefits extend beyond patient care to include cost savings, data analytics, patient engagement, and support for regulatory compliance. EHRs enable healthcare organizations to reduce administrative overhead, analyze patient data to improve outcomes, engage patients in their care, and meet regulatory requirements, such as HIPAA.

12. FUTURE SCOPE

The future scope of Electronic Health Records (EHRs) is highly promising and evolving as healthcare continues to embrace digital transformation. Several trends and areas of development suggest a bright future for EHR systems:

1. Interoperability and Data Exchange:

EHRs will increasingly focus on achieving seamless interoperability, allowing different EHR systems and healthcare organizations to exchange data efficiently. This will enhance care coordination, reduce duplicated tests, and improve patient outcomes.

2. Patient-Centered Care:

Future EHRs will prioritize patient engagement and empowerment, allowing patients to have more control over their health information, access to telemedicine, and the ability to participate actively in their care decisions.

3. Artificial Intelligence (AI) and Machine Learning:

EHRs will incorporate AI and machine learning to help healthcare providers make more accurate diagnoses, predict patient outcomes, and assist in treatment recommendations based on large datasets and patient history.

4. Mobile and Remote Access:

Mobile EHR applications will become more prevalent, enabling healthcare providers to access patient records securely from smartphones and tablets. This will be especially valuable in remote or home healthcare settings.

5. Data Analytics and Population Health:

EHRs will continue to enhance data analytics capabilities to support population health management, enabling healthcare organizations to identify trends and implement proactive health interventions.

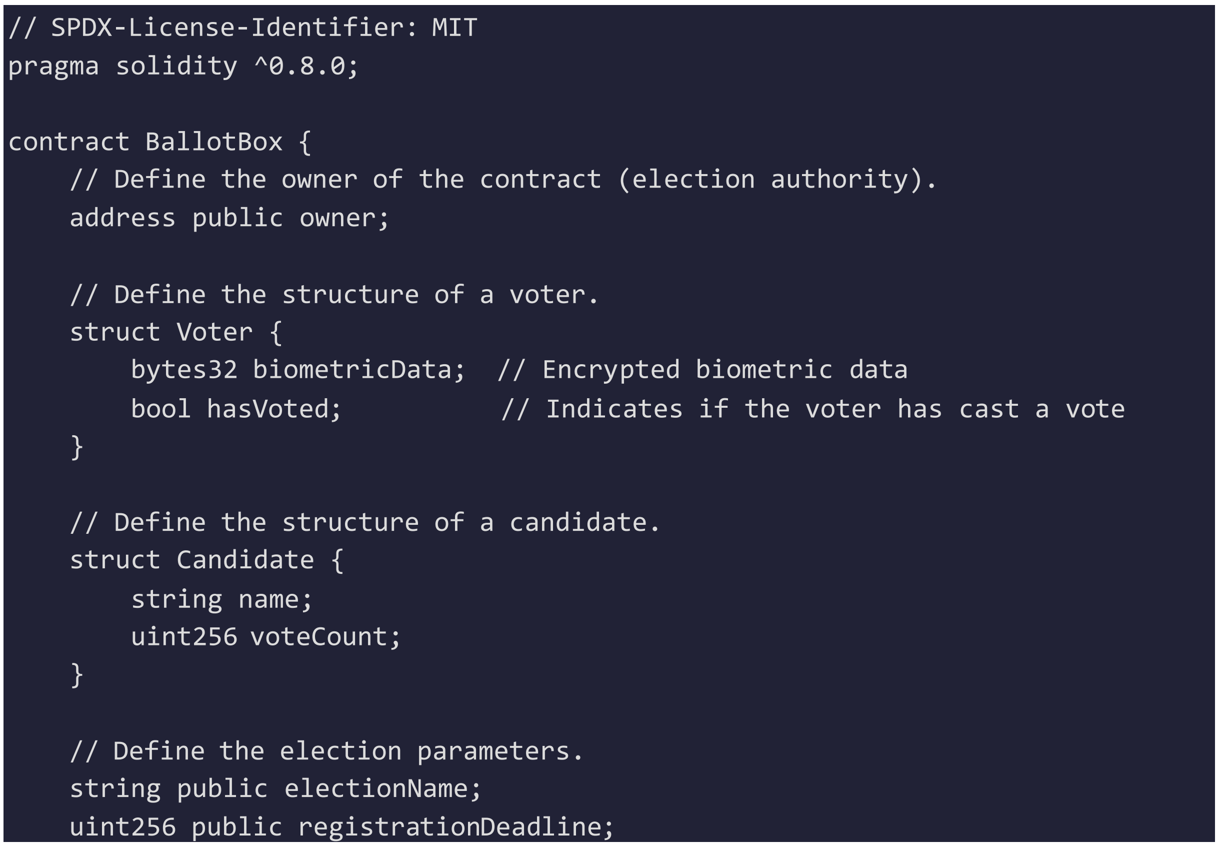
6. Telehealth Integration:

EHR systems will further integrate with telehealth platforms, offering comprehensive telemedicine solutions that allow for remote consultations, patient monitoring, and virtual care delivery.

13. APPENDIX

Source Code

Solidity coding :



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

contract BallotBox {
    // Define the owner of the contract (election authority).
    address public owner;

    // Define the structure of a voter.
    struct Voter {
        bytes32 biometricData;  // Encrypted biometric data
        bool hasVoted;           // Indicates if the voter has cast a vote
    }

    // Define the structure of a candidate.
    struct Candidate {
        string name;
        uint256 voteCount;
    }

    // Define the election parameters.
    string public electionName;
    uint256 public registrationDeadline;
    uint256 public votingDeadline;

    // Store the list of candidates.
    Candidate[] public candidates;

    // Store the mapping of voters.
    mapping(address => Voter) public voters;

    // Event to announce when a vote is cast.
    event VoteCast(address indexed voter, uint256 candidateIndex);

    // Modifiers for access control.
    modifier onlyOwner() {
        require(msg.sender == owner, "Only the owner can call this
function.");
        _;
    }

    modifier canVote() {
        require(block.timestamp < votingDeadline, "Voting has ended.");
        require(block.timestamp < registrationDeadline, "Registration has
ended.");
        require(!voters[msg.sender].hasVoted, "You have already voted.");
        _;
    }

    // Constructor to initialize the contract.
    constructor(
        string memory _electionName,
        uint256 _registrationDeadline,
        uint256 _votingDeadline,
        string[] memory _candidateNames
    ) {
        owner = msg.sender;
        electionName = _electionName;
        registrationDeadline = _registrationDeadline;
        votingDeadline = _votingDeadline;

        // Initialize the list of candidates.
        for (uint256 i = 0; i < _candidateNames.length; i++) {
            candidates.push(Candidate({
                name: _candidateNames[i],
                voteCount: 0
            }));
        }
    }

    // Function to register a voter and store their encrypted biometric data.
    function registerVoter(bytes32 _encryptedBiometricData) public canVote {
        voters[msg.sender] = Voter({
            biometricData: _encryptedBiometricData,
            hasVoted: false
        });
    }

    // Function to cast a vote for a candidate.
    function castVote(uint256 _candidateIndex) public canVote {
        require(_candidateIndex < candidates.length, "Invalid candidate
index.");
        require(voters[msg.sender].biometricData != 0, "You must register
first.");

        // Mark the voter as having voted.
        voters[msg.sender].hasVoted = true;

        // Increment the candidate's vote count.
        candidates[_candidateIndex].voteCount++;

        // Emit a VoteCast event.
        emit VoteCast(msg.sender, _candidateIndex);
    }
}
```

Java script :

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

const abi = [
 {
  "inputs": [
   {
    "internalType": "string",
    "name": "_electionName",
    "type": "string"
   },
   {
    "internalType": "uint256",
    "name": "_registrationDeadline",
    "type": "uint256"
   },
   {
    "internalType": "uint256",
    "name": "_votingDeadline",
    "type": "uint256"
   },
   {
    "internalType": "string[]",
    "name": "_candidateNames",
    "type": "string[]"
   }
  ],
  "stateMutability": "nonpayable",
  "type": "constructor"
 },
 {
  "anonymous": false,
  "inputs": [
   {
    "indexed": true,
    "internalType": "address",
    "name": "voter",
    "type": "address"
   },
   {
    "indexed": false,
    "internalType": "uint256",
    "name": "candidateIndex",
    "type": "uint256"
   }
  ],
  "name": "VoteCast",
  "type": "event"
 },
 {
  "inputs": [
   {
    "internalType": "uint256",
    "name": "",
    "type": "uint256"
   }
  ],
  "name": "candidates",
  "outputs": [
   {
    "internalType": "string",
    "name": "name",
    "type": "string"
   },
   {
    "internalType": "uint256",
    "name": "voteCount",
    "type": "uint256"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
  "inputs": [
   {
    "internalType": "uint256",
    "name": "_candidateIndex",
    "type": "uint256"
   }
  ],
  "name": "castVote",
  "outputs": [],
  "stateMutability": "nonpayable",
  "type": "function"
 },
 {
  "inputs": [],
  "name": "electionName",
  "outputs": [
   {
    "internalType": "string",
    "name": "",
    "type": "string"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
  "inputs": [],
  "name": "owner",
  "outputs": [
   {
    "internalType": "address",
    "name": "",
    "type": "address"
   }
  ],
  "stateMutability": "view",
  "type": "function"
 },
 {
  "inputs": [
   {
    "internalType":  "bytes32",
    "name": "_encryptedBiometricData",
   "type": "bytes32"
  }
 ],
 "name": "registerVoter",
 "outputs": [],
 "stateMutability": "nonpayable",
 "type": "function"
},
{
 "inputs": [],
 "name": "registrationDeadline",
 "outputs": [
  {
   "internalType": "uint256",
   "name": "",
   "type": "uint256"
  }
 ],
 "stateMutability": "view",
 "type": "function"
},
{
 "inputs": [
  {
   "internalType": "address",
   "name": "",
   "type": "address"
  }
 ],
 "name": "voters",
 "outputs": [
  {
   "internalType": "bytes32",
   "name": "biometricData",
   "type": "bytes32"
  },
  {
   "internalType": "bool",
   "name": "hasVoted",
   "type": "bool"
  }
 ],
 "stateMutability": "view",
 "type": "function"
},
{
 "inputs": [],
 "name": "votingDeadline",
  "outputs": [
   {
    "internalType": "uint256",
    "name": "",
    "type": "uint256"
   }
  ],
  "stateMutability": "view",
  "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 = "0x3c3aAB9D955f0c1eba7db7E9beE044740b2Bad79"

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

GitHub :

https://github.com/Paramesh451/naanmudhalvan

PROJECT DEMO LINK:

https://youtu.be/RrL6FhktlKo?si=Mp5H2tQpSjQ6VDJ