SECURE VOTING PLATFORM USING BLOCKCHAIN

A PROJECT REPORT

Submitted by

PRATHAP T (311420104051)
SANTHOSH KUMAR D (311420104063)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

MEENAKSHI COLLEGE OF ENGINEERING, WEST K.K NAGAR

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2024

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"SECURE VOTING PLATFORM USING BLOCKCHAIN"** is the bonafide work of **"PRATHAP T (311420104051), SANTHOSH KUMAR D (311420104063)"** who carried out the project work under my supervision.

SIGNATURE

Dr.CH. SARADA DEVI, M.Tech., Ph.D.,

HEAD OF THE DEPARTMENT

Computer Science and Engineering,

Meenakshi College Of Engineering,

West K.K. Nagar, Chennai-78.

SIGNATURE

Mr. ANANTHAKOOTHAN ,M.E.,

SUPERVISOR

Assistant professor

Computer Science and Engineering,

Meenakshi College Of Engineering,

West K.K. Nagar, Chennai-78.

Submitted for the Anna University Project VIVA-VOCE Examination held on

_____

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We sincerely express our gratitude to our chairman **Tmt. GOMATHI RADHAKRISHNAN,** Managing Director **Mrs. JAYANTHI PRABAKARAN** and the authorities of **MEENAKSHI AMMAL EDUCATIONAL TRUST** for the patronage and parental care showered on our welfare rooted in the academic career.

We express our thanks to our principal **Dr. R. GANESAN, Ph.D,** for his support and encouragement throughout our course of study.

We express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering, **Dr .CH. SARADA DEVI, M.Tech., Ph.D,** for her constant support, motivation, encouragement and being a source of inspiration throughout this project and during the course of this semester.

Our genuine thanks to **Ms. S. VIOLET, M.E.** Assistant Professor, Department of Computer Science and Engineering and **Mr. ANANTHAKOOTHAN ,M.E.,** Assistant Professor, Department of Computer Science and Engineering, internal project coordinators for their valuable guidance in modeling this project and for helping us in shaping this project to its best.

We thank our internal guide **Mr. ANANTHAKOOTHAN ,M.E.,** Assistant Professor, Department of Computer Science and Engineering for her suggestions, guidance and sustained interest in completing the project work successfully.

ABSTRACT

Building a secure electronic voting system that offers the fairness and privacy of current voting schemes, while providing the transparency and flexibility offered by electronic systems has been a challenge for a long time. In this work-in-progress paper, we evaluate an application of blockchain as a service to implement distributed electronic voting systems. The paper proposes a novel electronic voting system based on blockchain that addresses some of the limitations in existing systems and evaluates some of the popular blockchain frameworks for the purpose of constructing a blockchain-based e-voting system. In particular, we evaluate the potential of distributed ledger technologies through the description of a case study; namely, the process of an election, and the implementation of a blockchain-based application, which improves the security decreases the cost of hosting a nationwide election.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interface
UI	User Interface
HTTPS	Hyper Text Transfer Protocol Secure
HTML	Hyper Text Markup Language

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Blockchain is a decentralized and distributed digital ledger technology that records transactions in a secure, transparent, and immutable manner. It consists of a chain of blocks, with each block containing a set of transactions. Unlike traditional centralized systems, blockchain operates on a decentralized network of computers, often referred to as nodes. These nodes work together to maintain and update the ledger. The blockchain is built by various blocks. Different Transactions are grouped into blocks. Each block typically contains a collection of transactions that have occurred within a certain time frame. These blocks are linked together in a chronological order, forming a chain. Once a block is added to the chain, it cannot be altered or deleted, ensuring the integrity of the historical data. Blockchain's main significance is its security and validation. Transactions on the blockchain are secured using cryptographic techniques. Each block includes a reference to the previous block (hence the name "blockchain"), making it exceedingly difficult for anyone to tamper with past records. For validation, Blockchains rely on consensus mechanisms to validate and confirm transactions. Common mechanisms include Proof of Work (PoW) and Proof of Stake (PoS), which ensure that only valid transactions are added to the ledger. In most cases, blockchain networks are open and transparent, allowing anyone to view the transaction history. This transparency can enhance trust among participants. There are both public blockchains (open to anyone) and private blockchains (restricted to specific participants or organizations). Private blockchains offer more control over access and data privacy. Blockchain's decentralized and immutable nature makes it a promising technology for enhancing trust, transparency, and security in various applications and industries. It has the potential to

revolutionize the way transactions and data are recorded and verified. Blockchain technology has a wide range of applications beyond cryptocurrencies. It is used in finance for secure and efficient payment systems, in supply chain management for tracking the movement of goods, in healthcare for managing patient records, and in many other industries. There are both public blockchains (open to anyone) and private blockchains (restricted to specific participants or organizations). Private blockchains offer more control over access and data privacy.

1.2 OBJECTIVE

The objective of implementing a blockchain-based voting system is to enhance the integrity, security, and transparency of the electoral process. By leveraging the immutable and decentralized nature of blockchain technology, such a system aims to eliminate concerns related to tampering, fraud, and manipulation commonly associated with traditional voting systems. Additionally, the transparency provided by blockchain allows for real-time verification of voting results by stakeholders, enhancing trust in the electoral process and fostering greater participation and inclusivity.

1.3 SUMMARY

This project is about building a secure electronic voting system that offers the fairness and privacy of current voting schemes, while providing the transparency and flexibility offered by electronic systems has been a challenge for a long time. In this work-in-progress paper, we evaluate an application of blockchain as a service to implement distributed electronic voting systems.

CHAPTER 2

LITERATURE SURVEY

2.1 GENERAL

Literature survey gives the overall description of the reference papers that has been referred to design the application, using which the problems of the existing applications and technologies is identified. The methods to overcome such limitations are also recognized.

2.2 LITERATURE REVIEW

2.2.1 A. Mendon, B. Manoj Votavat and A. Lodh, "Blockchain based e-voting system," 2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT).

An innovative development in the field of information technology is distributed ledger technology. Blockchain technology provide a limitless number of sharing economy-friendly applications. This research attempts to assess how well distributed online voting systems may be implemented using blockchain technology. The project explains the specifications for developing online voting systems and outlines the technical constraints of using blockchain to implement such systems. A few of the well-familiar frameworks that uses blockchain as a service are first evaluated for the project. Then put forth a cutting-edge blockchain-based online voting system that fixes all the issues found. In a broader sense, this project assesses the possibilities of distributed ledger technologies by telling a case study, specifically the conduct of an election and the execution of a blockchain built application that improves privacy and reduces the cost of holding national elections.

2.2.2 Apoorva S. Drakshayani, U. Vijayalakshmi, S. R. Sri, A. Srivani and A. Vyshnavi, "Online Voting System Using Blockchain," 2022 International Conference on Electronics and Renewable Systems (ICEARS).

In an electronic information system, blockchain technology has already been touted as a way to support the need for trust between transactions. To a certain extent, electronic democracy has addressed the problems of helpless anonymity and low competency associated with conventional democracy. Electronic democracy. We can't ignore the problems it poses in overseeing the vote counting, or the fact that it necessitates the presence of a reliable third party. Blockchain technology's decentralisation, anonymity, and non-tampering properties have made it a better contender for overcoming electronic voting's concerns in the recent years. In this research, we propose a Blockchain-based online voting system. In contrast to traditional and online voting systems, we can cast our ballots anywhere in the world using a mobile device.

2.2.3 T. Vairam, S. Sarathambekai and R. Balaji, "Blockchain based Voting system in Local Network," 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS).

Building a secure electronic voting system that offers the fairness and privacy of current voting schemes, while providing the transparency and flexibility offered by electronic systems has been a challenge for a long time. In this work-in-progress paper, we evaluate an application of blockchain as a service to implement distributed electronic voting systems. The paper proposes a novel electronic voting system based on blockchain that addresses some of the limitations in existing systems and evaluates some of the popular blockchain frameworks for the purpose of constructing a blockchain-based e-voting system. In particular, we evaluate the potential of distributed ledger technologies through the description of a case study;

namely, the process of an election, and the implementation of a blockchain-based application, which improves the security and decreases the cost of hosting a nationwide.

2.3 SUMMARY

This chapter gives a brief description about each paper referred and also explains the concept which is extracted from them and is implemented in the proposed system.

CHAPTER 3

SYSTEM ANALYSIS

3.1 GENERAL

System Analysis is important because it provides an avenue for solutions in the system through the various tasks involved in doing the analysis. This chapter explains the existing system and how it works, and its disadvantages based on which the proposed system is designed.

3.2 EXISTING SYSTEM

- The existing paper-based voting system, while historically trusted, is prone to errors and logistical challenges.
- Digital voting presents an opportunity to modernize and streamline the electoral process. However, ensuring the security of digital voting systems is paramount due to the potential for cyber threats and manipulation.
- Current digital voting systems employ encryption, authentication, and audit trails to mitigate these risks. Yet, concerns persist about the vulnerability of centralized systems to cyberattacks.
- The transition to digital voting offers increased accessibility and efficiency but must prioritize the integrity and confidentiality of votes. As technology evolves, addressing security concerns will be crucial in maintaining trust and legitimacy in democratic elections.

3.2.1 EXISTING SYSTEM ARCHITECTURE

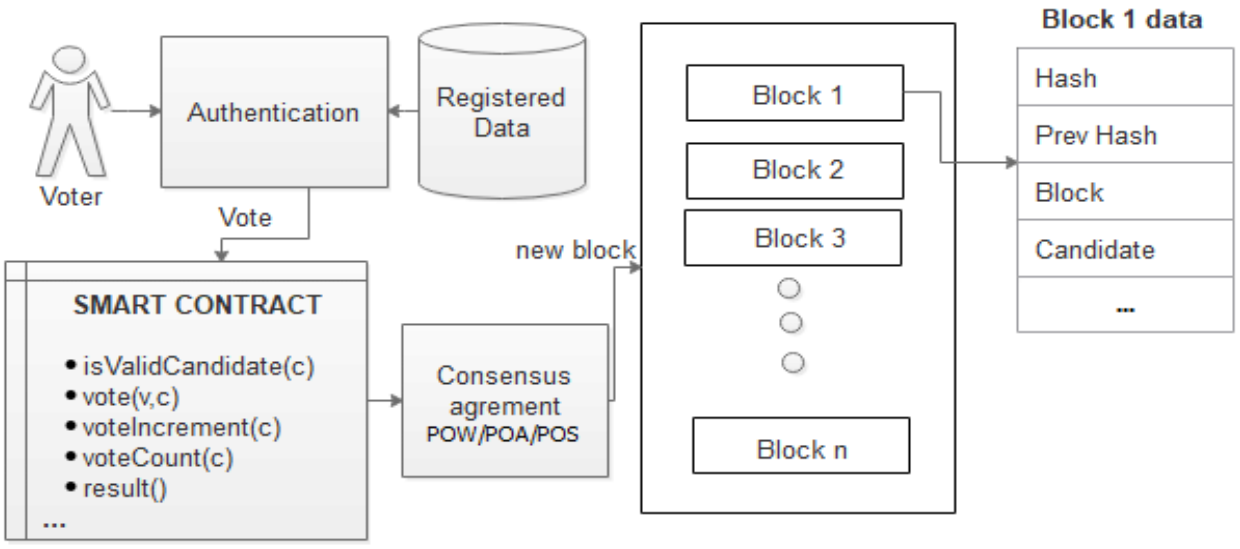

Figure 3.1 Existing system architecture

3.2.2 LIMITATIONS OF THE EXISTING SYSTEM

- The existing paper-based voting system faces significant limitations, including susceptibility to errors, logistical challenges, and vulnerability to fraud. Manual processes inherent in paper-based systems lead to inaccuracies in ballot handling, distribution, and counting, while managing large-scale elections introduces logistical hurdles and delays.
- The reliance on manual counting further extends the time required to announce results, fostering uncertainty and disputes. Transitioning to digital voting systems addresses these shortcomings, offering increased accuracy, efficiency, and accessibility. Despite facing cybersecurity and privacy concerns, the benefits of digital voting outweigh those of the paper-based

system, making its evolution imperative for modernizing and ensuring the integrity of democratic elections.

3.3 PROPOSED SYSTEM

- I proposed electronic voting system leveraging blockchain technology, MetaMask is integrated to bolster security and user interaction.
- MetaMask serves as a digital wallet and browser extension, ensuring secure access to the voting platform.
- Through MetaMask's authentication mechanism, only authorized users can securely cast their votes, enhancing the system's integrity.
- MetaMask facilitates seamless interaction with the blockchain, enabling voters to securely submit their votes and ensuring their transparent and immutable recording on the distributed ledger.
- This integration not only enhances the security and privacy of the voting process but also provides voters with the ability to independently verify the integrity of the electoral process by accessing the blockchain directly through MetaMask.
- By incorporating MetaMask into our blockchain-based electronic voting system, we aim to provide a user-friendly and secure platform that maintains the privacy and fairness of traditional voting systems while leveraging the transparency and flexibility offered by blockchain technology.

3.3.1 PROPOSED SYSTEM ARCHITECTURE

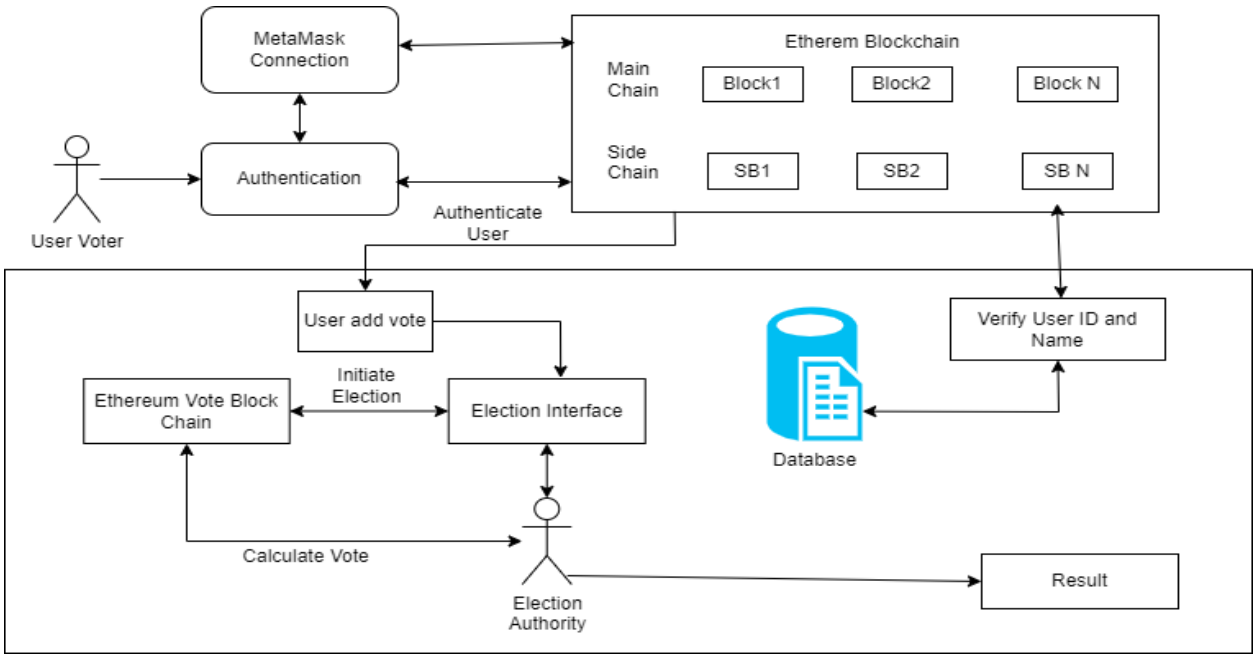

Figure 3.2 Proposed system architecture

3.3.2 ADVANTAGES OF PROPOSED SYSTEM

- Integrating MetaMask into our blockchain-based electronic voting system enhances security, transparency, and accessibility. MetaMask's authentication mechanism ensures only authorized users can securely cast votes, bolstering system integrity.
- Seamless interaction with the blockchain allows for transparent and immutable recording of votes, enabling voters to independently verify the electoral process.
- Additionally, MetaMask's user-friendly interface simplifies the voting process, increasing accessibility for all voters. Overall, this integration improves the fairness and reliability of the electronic voting system while leveraging the benefits of blockchain technology.

USE CASE DIAGRAM FOR PROPOSED SYSTEM

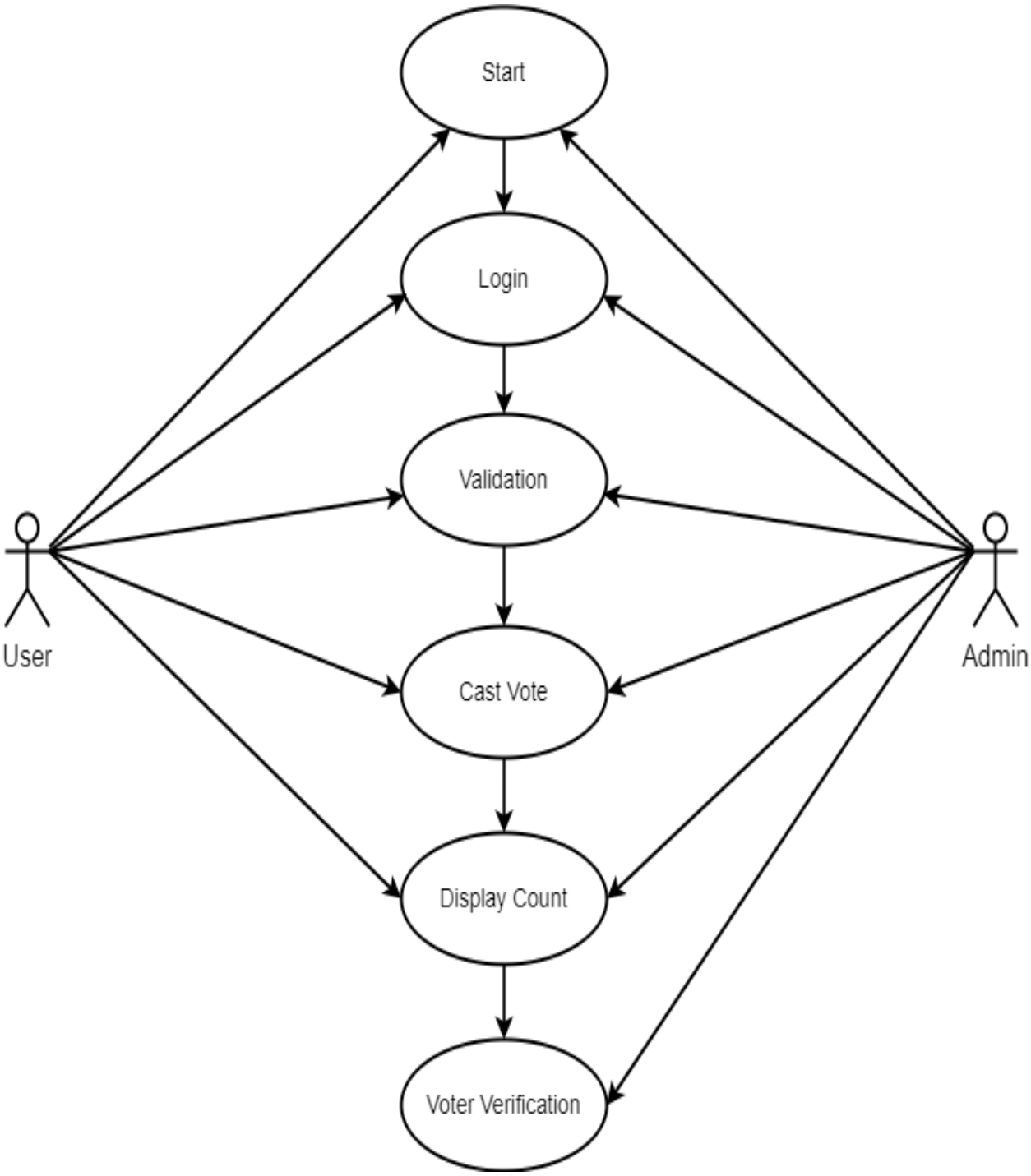

Figure 3.3 Use Case Diagram

Describes how the user performs various functions from start to finish in the System.

CLASS DIAGRAM FOR PROPOSED SYSTEM

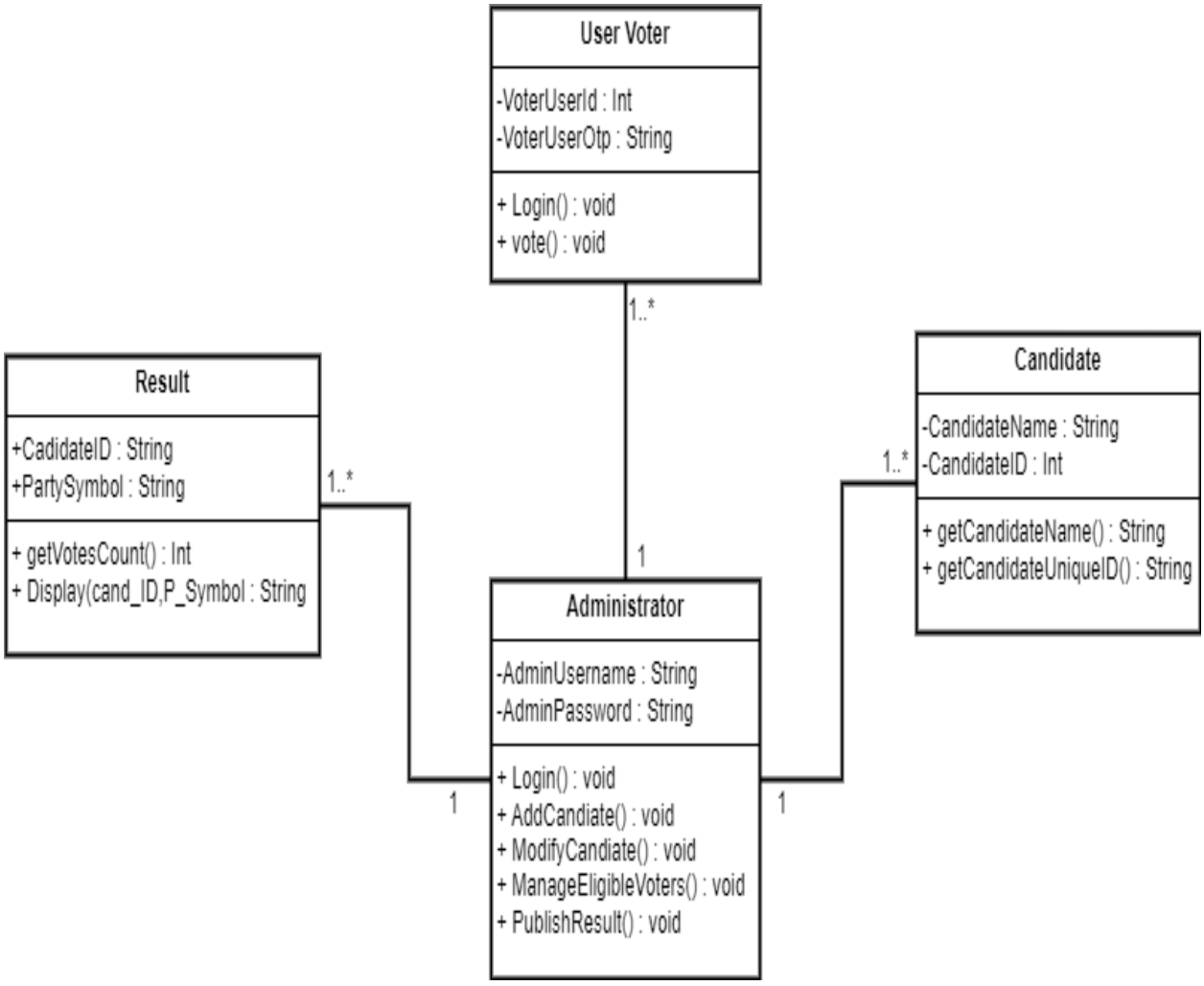

Figure 3.4 Class Diagram

Figure 3.4 depicts the Class Diagram which describes the user's details and service requests as opposed to how the System responds to these requests with the help of the database.

SEQUENCE DIAGRAM FOR PROPOSED SYSTEM

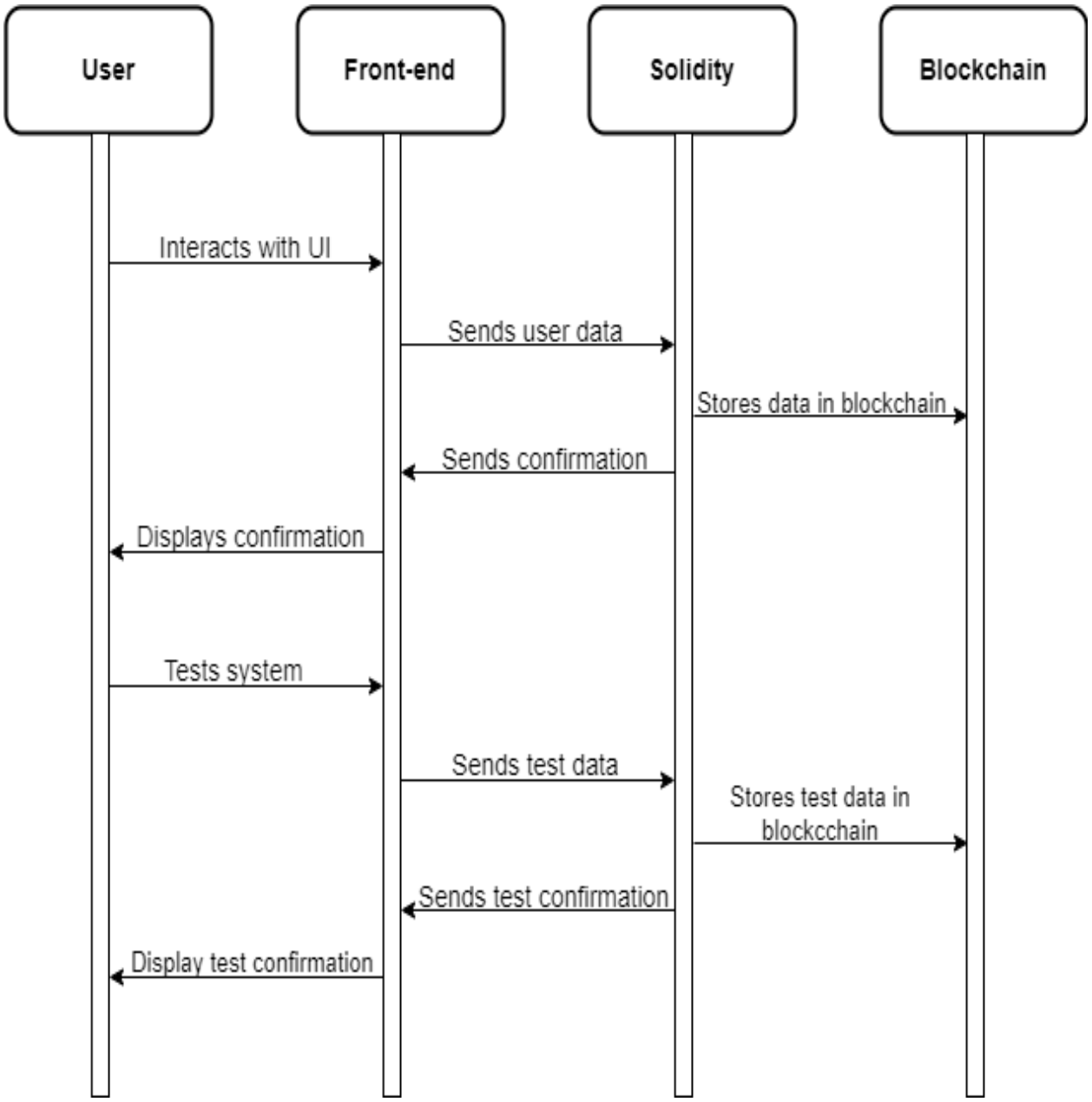

Figure 3.5 Sequence diagram

Figure 3.5 depicts the Sequence Diagram which describes the sequence of events occurring within the System with details as to how the System utilizes the database to deliver the various functionalities to the user.

3.4 SUMMARY

The UML Diagrams are an ideal representation of the design work that describes the working of the System in a general perspective. In that, the various categories of UML Diagrams describe the System in a unique manner such that all the aspects of the System are described, analyzed, rectified and finalized before it is translated in the form of working code. The purpose of representing the requirements in the form of such diagrams is to make sure that all the necessary changes can be made before the implementation process as doing so here will be effective in terms of cost and time.

CHAPTER 4

SYSTEM DESIGN AND IMPLEMENTATION

4.1 GENERAL

This chapter describes the design phase of a project which is divided into various modules. It briefs about system design of each module along with its functionalities.

4.2 LIST OF MODULES

- Blockchain Network Emulator (Ganache)
- Blockchain Network Interface (MetaMask)
- Voter
- Administrator
- Email OTP Dispatcher (Nodemailer)
- Front-end Development (HTML, CSS, JavaScript)
- Back-end Implementation (Next.js)

4.3 MODULE DESCRIPTION

The modules are performed using different techniques and they are described below with the help of illustrations.

4.3.1 BLOCKCHAIN NETWORK EMULATOR (Ganache) MODULE

Ganache is a personal blockchain for Ethereum development that allows developers to create a private blockchain network on their local machine. It is part of the Truffle Suite, a popular set of tools for Ethereum development. Ganache provides a local environment where developers can deploy contracts, develop

decentralized applications (DApps), and run tests without the need for a live Ethereum network connection. It offers features such as quick blockchain creation, account management, built-in block explorer, and customizable blockchain settings. Smart contracts governing the decentralized voting system's logic and candidate data, is deployed and tested within this local environment. Overall, Ganache plays a crucial role in accelerating the decentralized voting project's development lifecycle by providing a reliable and flexible blockchain simulation environment.

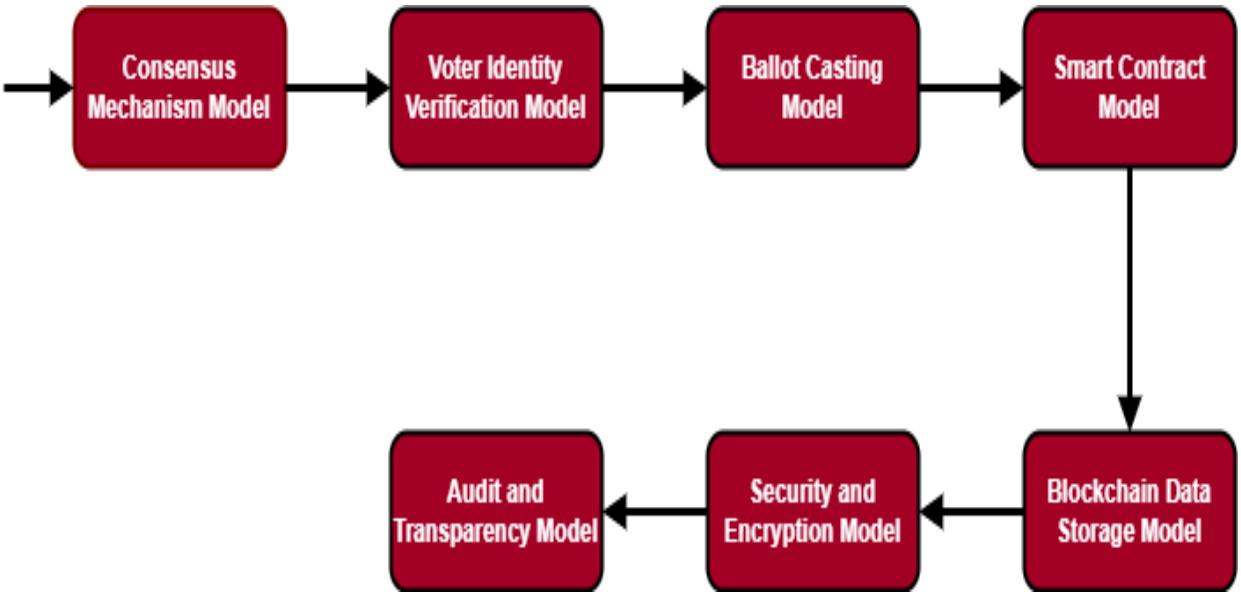

Fig 4.1 Blockchain Network Emulator module

4.3.1.1 To start Blockchain Network Emulator module for Secure Voting Platform using Blockchain, we'll break down Module 1 into several steps:

- **Consensus Mechanism Model:** This ensures agreement among the network participants on the validity of transactions (votes). Common consensus mechanisms include Proof of Work (PoW), Proof of Stake (PoS), Delegated Proof of Stake (DPoS), etc.

- **Voter Identity Verification Model:** To ensure that only eligible voters can cast their votes, this model might include identity verification mechanisms such as biometric authentication, government-issued ID verification, or cryptographic methods.
- **Ballot Casting Model**: This model manages the process by which voters cast their votes securely and anonymously. It ensures that each vote is recorded accurately and cannot be tampered with.
- **Smart Contract Model**: Smart contracts can be utilized to enforce the rules of the voting process, such as ensuring that only eligible voters can cast a vote, and automatically tallying the votes in a transparent and tamper-proof manner.
- **Blockchain Data Storage Model:** This model defines how the data (votes) is stored on the blockchain in a secure, immutable, and decentralized manner.
- **Security and Encryption Model:** This model ensures that the voting process is secure from various threats, including hacking, tampering, and unauthorized access. It may include encryption techniques to protect sensitive data.

- **Audit and Transparency Model:** To ensure transparency and accountability, this model provides tools for auditing the voting process, verifying the integrity of the votes, and ensuring that the results are accurate.

4.3.1.1 To start Blockchain Network Emulator module for Secure Voting Platform using Blockchain, we'll brak down Module 1 Algorithm into several steps:

- **Step 1:** Network participants agree on transaction validity. Utilize PoW, PoS, DPoS, etc., for agreement.
- **Step 2:** Verify voter eligibility through biometrics, IDs, cryptography.
- **Step 3:** Enforce voting rules using smart contracts.Automate tallying in a transparent manner.
- **Step 4:** Store vote data securely, immutably, and decentralized on the blockchain.
- **Step 5:** Verify vote integrity and ensure accurate results.

4.3.2 BLOCKCHAIN NETWORK INTERFACE (MetaMask) MODULE

MetaMask simplifies the process of interacting with Ethereum-based blockchain network by providing a user-friendly interface for managing Ethereum accounts and transactions. Beyond its core functionality as an Ethereum wallet and transaction signer, MetaMask offers advanced features such as custom network configuration, token management, and decentralized finance (DeFi) integrations. It supports multiple Ethereum networks, including main net, test nets, and custom networks, enabling users to access a wide range of decentralized services and applications. MetaMask also prioritizes user privacy and security through features like password protection, encrypted data storage, and seed phrase backup, ensuring

a secure and seamless experience for users.

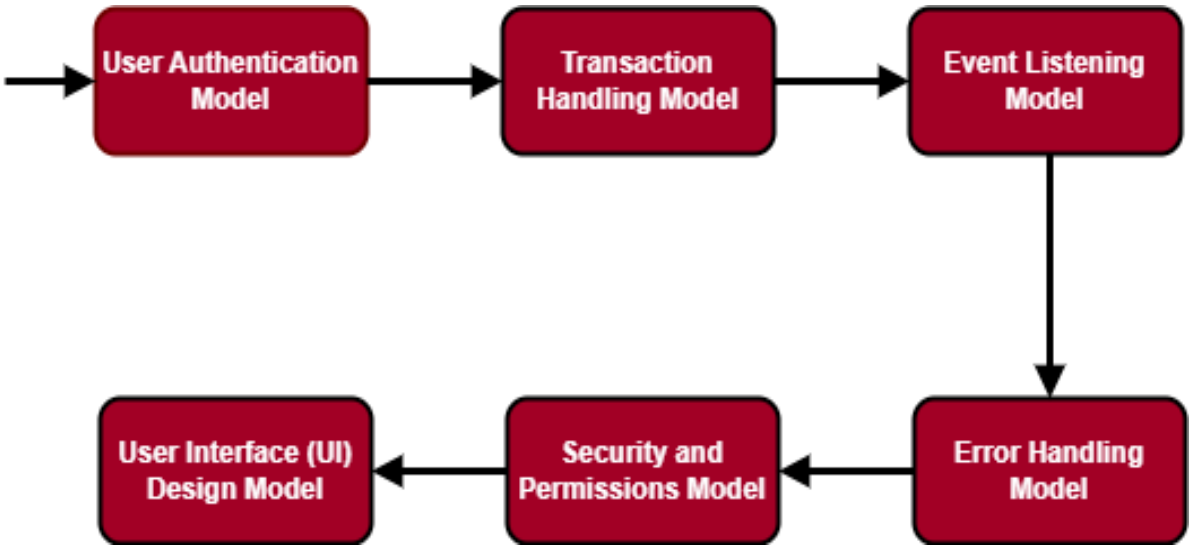

Figure 4.2 Blockchain Network Interface (MetaMask) Model

4.3.2.1 To do an Blockchain Network Interface (MetaMask) Module for Secure Voting Platform using Blockchain, we'll break down Module 2 into several steps:

- **User Authentication Model:** This model manages the authentication process for users accessing the voting system through MetaMask or similar blockchain interfaces. It ensures secure login and identity verification.
- **Transaction Handling Model:** Handles the creation, signing, and submission of transactions (votes) to the blockchain network through MetaMask. It ensures that transactions are properly formatted, signed with the user's private key, and broadcast to the network.
- **Event Listening Model:** Monitors blockchain events related to the voting process (e.g., new votes, tally updates) and provides real-time feedback to users through the MetaMask interface.
- **Error Handling Model:** Manages errors and exceptions that may occur during the voting process, providing clear and informative messages to users

through the MetaMask interface.

- **Security and Permissions Model:** Ensures that only authorized users with the appropriate permissions can access and interact with the voting system through MetaMask. It also includes measures to protect users' private keys and sensitive data.
- **User Interface (UI) Design Model:** Focuses on designing a user-friendly and intuitive interface within MetaMask for accessing and participating in the voting process. This model ensures that users can easily understand and navigate the voting interface.

4.3.2.1 To do an Blockchain Network Interface (MetaMask) Module for Secure Voting Platform using Blockchain, we'll brak down Module 2 Algorithm into several steps:

- **Step 1**: Ensure secure login and identity verification.
- **Step 2:** Handle creation, signing, and submission of transactions (votes) via MetaMask.
- **Step 3:** Provide clear and informative messages to users via MetaMask interface.

- **Step 4:** Ensure only authorized users access the voting system via MetaMask.
- **Step 5:** Design user-friendly interface within MetaMask for voting.

4.3.3 VOTER MODULE

The user module in our project serves as the interface through which individuals interact with the voting system. Users, typically voters, utilize this module to cast their votes and view election-related information such as candidate lists, election results, and other pertinent details. Through the user module, voters can access the voting platform, select their preferred candidates, submit their votes securely, and verify the integrity of the electoral process by viewing transparent and accurate election outcomes.

Additionally, the user module includes features such as user authentication, ensuring that only authorized individuals can participate in the voting process, and user-friendly interfaces to enhance the overall voting experience. Users play a central role in the decentralized voting system, participating in the electoral process through secure and transparent voting mechanisms. The user interface provides intuitive navigation, informative candidate profiles, and real-time updates on election results, fostering engagement and transparency.

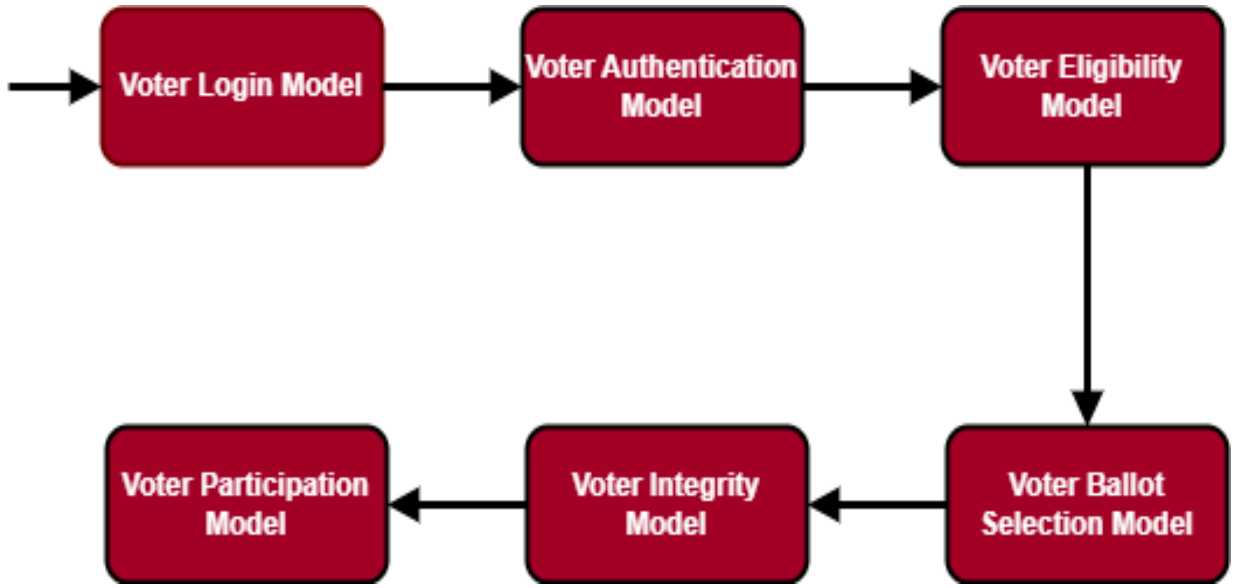

Figure 4.3 VOTER MODULE

4.3.3.1 To do an Voter Module for Secure Voting Platform using Blockchain, we'll break down Module 3 into several steps:

- **Voter Login Model:** This model manages the registration process for voters, ensuring that only eligible individuals are allowed to participate in the voting process. It may include functionalities such as identity verification, registration validation, and voter database management.
- **Voter Authentication Model:** This model handles the authentication of voters when they attempt to access the voting system. It verifies the identity of voters through various means such as biometric authentication, government-issued ID verification, or login credentials.
- **Voter Eligibility Model:** Ensures that voters meet the eligibility criteria to participate in specific elections or decision-making processes. This model may include age requirements, citizenship status, residency, or other criteria defined by the voting system.
- **Voter Ballot Selection Model:** Manages the process by which voters select their choices on the ballot. It ensures that voters can make their selections accurately and securely, whether through traditional paper ballots or electronic voting systems.
- **Voter Integrity Model:** This model focuses on maintaining the integrity of the voting process and preventing fraudulent activities such as double voting or coercion. It includes measures to protect the anonymity of voters and prevent unauthorized access to the voting system.
- **Voter Participation Model:** Encourages and facilitates voter participation in the electoral process through various means, such as voter education initiatives, outreach campaigns, and accessibility measures for individuals with disabilities.

4.3.3.1 To do an Voter Module for Secure Voting Platform using Blockchain, we'll brak down Module 3 Algorithm into several steps:

- **Step 1:** Validate registrations and ensure only eligible individuals participate.
- **Step 2:** Authenticate voters accessing the system using various methods.
- **Step 3:** Verify eligibility before allowing participation in elections.
- **Step 4:** Protect voter anonymity and prevent unauthorized access to the system.

4.3.4 ADMINISTRATOR MODULE

The admin module within the project serves as the central authority responsible for overseeing and managing various administrative functions related to the voting system. The admin dashboard offers comprehensive tools for managing candidate registration, voter authentication, and election monitoring. Advanced features such as real-time analytics, audit trails, and role-based access control (RBAC) empower administrators to make informed decisions and maintain the integrity of the voting process. Administrators can configure voting parameters, set election timelines, and generate detailed reports on voter turnout and election outcomes.

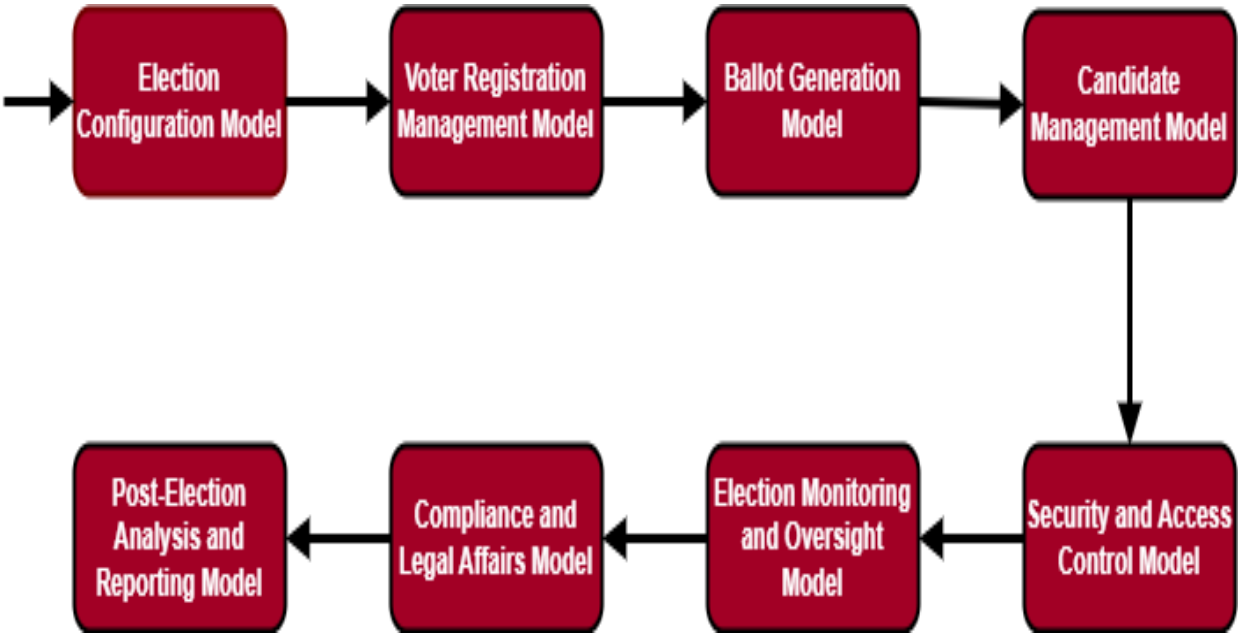

Figure 4.4 ADMINISTRATOR MODULE

4.3.4.1 To do an Administrator Module for Secure Voting Platform using Blockchain, we'll break down Module 4 into several steps:

- **Election Configuration Model:** This model manages the configuration of elections, including setting up the parameters such as voting dates, eligible candidates or options, ballot formats, and any special rules or requirements.
- **Voter Registration Management Model:** Handles the registration of voters, including verifying their eligibility, maintaining voter databases, processing registration applications, and ensuring voter rolls are accurate and up-to-date.
- **Ballot Generation Model:** Generates the official ballots for the election based on the configured options and candidates. It ensures that the ballots are correctly formatted and include all necessary information for voters to make their selections.
- **Candidate Management Model:** Manages information about candidates or

options participating in the election, including candidate registration, verification, and monitoring campaign activities to ensure compliance with election regulations.

- **Security and Access Control Model:** Implements security measures to protect the integrity of the voting system, including user authentication, authorization, and access control mechanisms to prevent unauthorized access to sensitive data or administrative functions.
- **Election Monitoring and Oversight Model:** Monitors the election process in real-time to detect and respond to any irregularities or security threats. This model may include auditing capabilities, logging of administrative actions, and reporting mechanisms for transparency and accountability.
- **Compliance and Legal Affairs Model:** Ensures that the election process complies with relevant laws, regulations, and ethical standards. This model may involve legal review of election procedures, compliance monitoring, and addressing legal challenges or disputes.
- **Post-Election Analysis and Reporting Model:** Conducts analysis of election data, evaluates the effectiveness of election procedures, and prepares reports on election outcomes and any recommendations for improvements in future elections.

4.3.4.1 To do an Administrator Module for Secure Voting Platform using Blockchain, we'll brak down Module 4 Algorithm into several steps:

- **Step 1:** Set up parameters such as voting dates, eligible candidates, and ballot formats.
- **Step 2:** Process registration applications and ensure accuracy of voter rolls.
- **Step 3:** Manage candidate registration, verification, and campaign activities.
- **Step 4:** Analyze election data and evaluate effectiveness of procedures.

4.3.5 EMAIL OTP DISPATCHER (Nodemailer) MODULE

Nodemailer enables the decentralized voting system to send secure, transactional emails to users for account verification, password reset, and important notifications. The email OTP sender module integrates seamlessly with the user authentication system, generating unique one-time passwords (OTPs) or verification links that users can use to authenticate their identities securely. Nodemailer supports customization of email templates, branding, and localization, allowing the voting system to deliver personalized and contextually relevant messages to users. Additionally, Nodemailer provides robust error handling, delivery tracking, and SMTP server configuration options, ensuring reliable and timely delivery of emails across diverse email providers and networks.

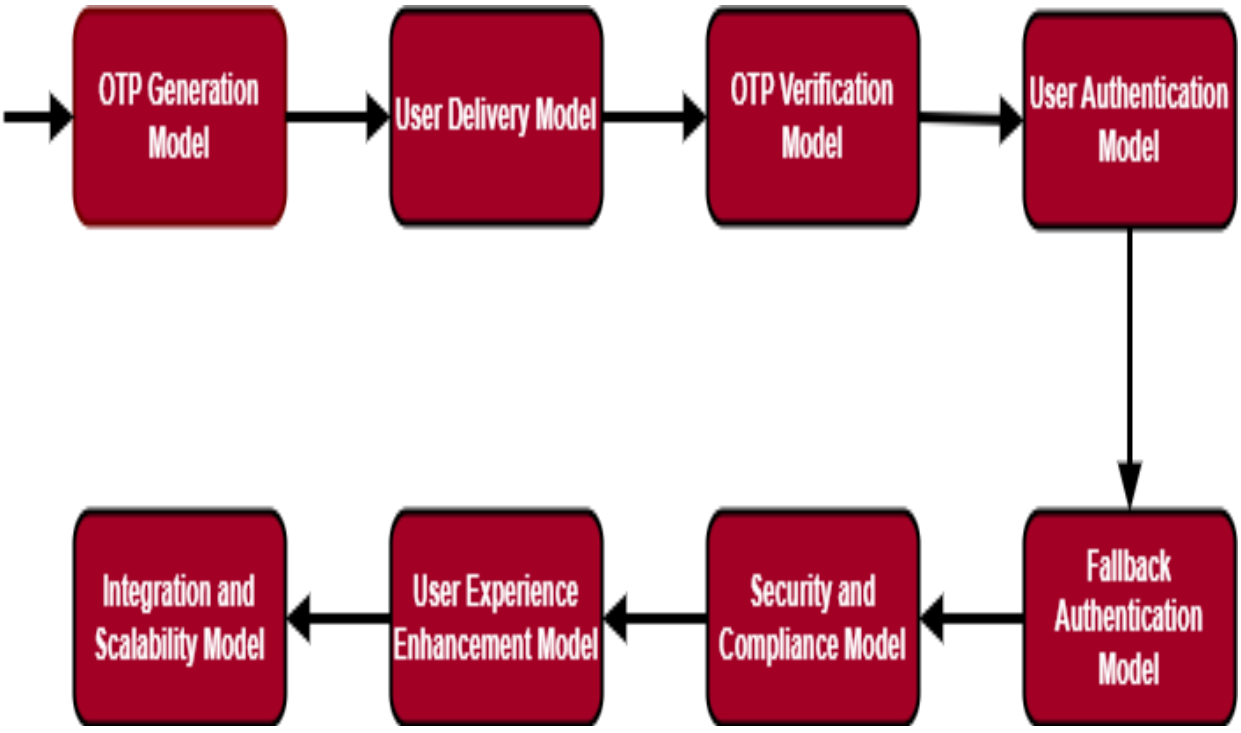

Figure 4.5 EMAIL OTP DISPATCHER (Nodemailer) MODULE

4.3.5.1 To do an Email OTP Dispatcher (Nodemailer) Module for Secure Voting Platform using Blockchain, we'll break down Module 5 into several steps:

- **OTP Generation Model:** This model is responsible for generating the one-time passwords that are sent to users via email. It ensures that each OTP is unique, securely generated, and has a limited validity period.
- **User Delivery Model:** Manages the delivery of OTPs to users' addresses. This model includes functionalities for composing and sending emails containing the OTPs, as well as handling any errors or issues that may arise during the delivery process.
- **OTP Verification Model:** Handles the verification of OTPs submitted by users to authenticate themselves. This model validates the OTPs against the ones generated and sent to users, ensuring that they match and are within the valid time window.
- **User Authentication Model:** Integrates OTP verification as part of the user authentication process within the voting system. This model ensures that users are authenticated securely before they can access or participate in the voting process.
- **Fallback Authentication Model:** Provides alternative authentication methods in case users encounter issues with receiving or using OTPs via User. This model may include backup options such as SMS OTP, security questions, or manual verification by election officials.
- **Security and Compliance Model:** Ensures that the implementation of User OTP follows best practices for security and compliance with relevant regulations (e.g., GDPR). This model includes measures to protect users' email addresses, secure transmission of OTPs, and adherence to data privacy

standards.

- **User Experience Enhancement Model:** Focuses on optimizing the user experience of receiving and using OTPs via User. This model includes features such as clear instructions, user-friendly interfaces for entering OTPs, and providing feedback on the authentication status.
- **Integration and Scalability Model:** Ensures seamless integration of User OTP functionality into the voting system architecture, with considerations for scalability to handle large numbers of users during peak voting periods.

4.3.5.1 To do an Email OTP Dispatcher (Nodemailer) Module for Secure Voting Platform using Blockchain, we'll brak down Module 7 Algorithm into several steps:

- **Step 1:** Generate unique, securely generated OTPs with limited validity periods.
- **Step 2:** Compose and send emails containing OTPs to users' email addresses.
- **Step 3:** Validate OTPs against those generated and sent to users.
- **Step 4:** Protect users' email addresses and data privacy.
- **Step 5:** Ensure scalability to handle large numbers of users during peak voting periods.

4.3.6 FRONTEND DEVELOPMENT (HTML, CSS, JavaScript) MODULE

The frontend of the decentralized voting system is crafted with modern web technologies to deliver a responsive, visually appealing, and intuitive user experience. HTML provides the structural framework for organizing content elements, while CSS enhances the presentation layer with styling, animations, and layout adjustments. JavaScript adds interactivity and dynamic behaviour to the frontend, enabling features such as real-time updates, form validation, and client-side data processing. The frontend design prioritizes accessibility, mobile responsiveness, and cross-browser compatibility to ensure seamless access and usability across different devices and platforms.

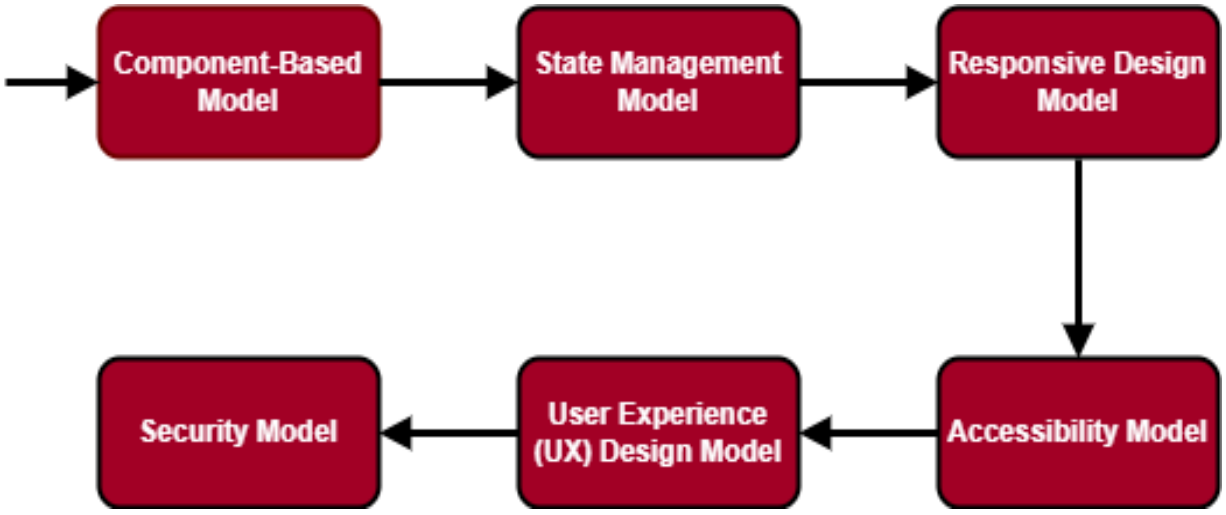

Figure 4.6 FRONTEND DEVELOPMENT (HTML, CSS, JavaScript) MODULE

4.3.6.1 To do an Frontend Development (HTML, CSS, JavaScript) Module for Secure Voting Platform using Blockchain, we'll break down Module 6 into several steps:

- **Component-Based Model:** Organizes the front-end codebase into reusable and modular components, such as header, footer, ballot, candidate list, and voting confirmation modal. This model promotes code reusability, maintainability, and scalability.
- **State Management Model:** Manages the state of the front-end application, including user interactions, data fetching, and application state transitions. This model may utilize state management libraries such as Redux, MobX, or React Context API to centralize and synchronize application state.
- **Responsive Design Model:** Ensures that the front-end application is accessible and functional across various devices and screen sizes, including desktops, laptops, tablets, and smartphones. This model utilizes responsive design techniques such as media queries and flexible layouts.
- **Accessibility Model:** Focuses on making the front-end application accessible to users with disabilities, including those with visual, auditory, motor, or cognitive impairments. This model includes implementing keyboard navigation, semantic HTML, ARIA roles, and other accessibility best practices.
- **User Experience (UX) Design Model:** Focusing on enhancing the overall user experience of the front-end application contributes to user satisfaction and engagement. Incorporating user research, usability testing, and iterative design processes helps identify user needs and preferences, resulting in a more intuitive and enjoyable voting experience.
- **Security Model:** Ensuring the security of the front-end application is essential

for protecting user data and preventing unauthorized access or malicious activities. Implementing security measures such as input validation, HTTPS encryption, and protection against common web vulnerabilities (e.g., XSS, CSRF) helps safeguard the application and user information.

4.3.6.1 To do an Frontend Development (HTML, CSS, JavaScript) Module for Secure Voting Platform using Blockchain, we'll brak down Module 6 Algorithm into several steps:

- **Step 1:** Implement responsive design techniques for accessibility across devices.
- **Step 2:** Ensure accessibility features for users with disabilities.
- **Step 3:** Enhance overall user satisfaction and engagement.
- **Step 4:** Implement security measures like input validation and HTTPS encryption.

4.3.7 BACKEND IMPLEMENTATION (Next.js) MODULE

Next.js powers the backend infrastructure of the decentralized voting system, orchestrating server-side logic, data management, and communication with external services and APIs. As a versatile server-side rendered (SSR) React framework, Next.js enables efficient rendering of dynamic web pages, optimizing performance and SEO visibility. The backend architecture leverages Next.js API routes, middleware, and serverless functions to handle authentication, authorization, and data validation, ensuring secure and reliable interactions between the frontend and backend components. Next.js also supports database integration, session management, and caching mechanisms for scalable and resilient backend operations, empowering us to build robust and highperformance decentralized voting systems.

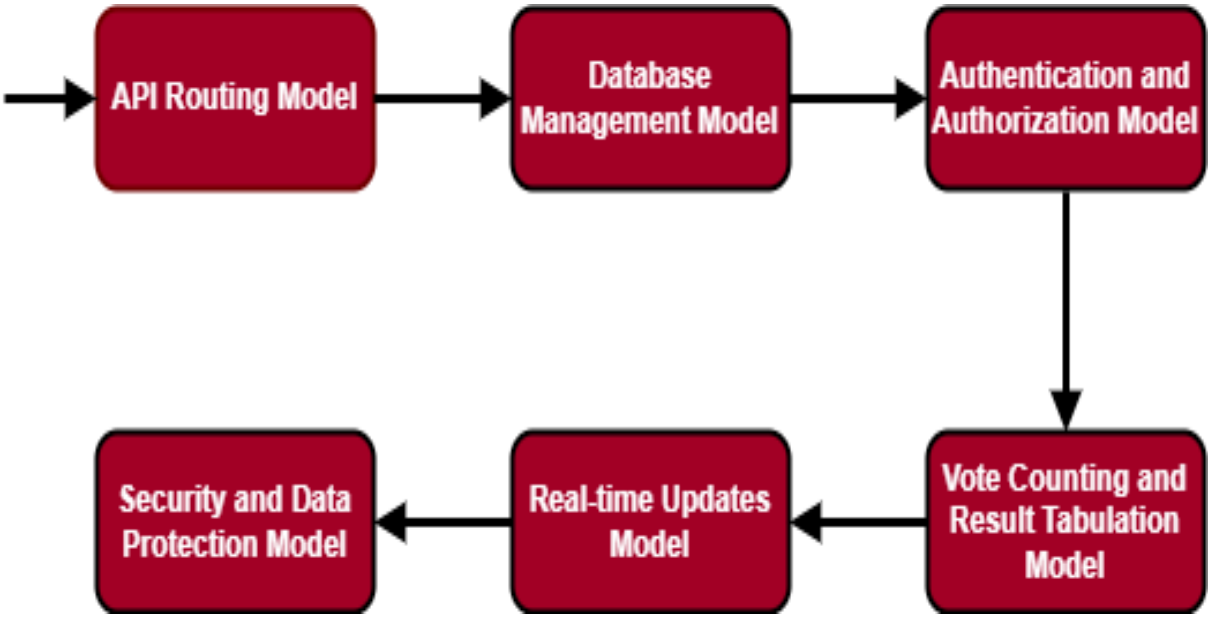

Figure 4.7 BACKEND IMPLEMENTATION (Next.js) MODULE

4.3.7.1 To do an Backend Implementation (Next.js) Module for Secure Voting Platform using Blockchain, we'll break down Module 7 into several steps:

- **API Routing Model:** Next.js provides a convenient way to create API routes that handle server-side logic for processing requests from the front end. This model involves defining routes for various functionalities such as user authentication, vote submission, candidate information retrieval, and result retrieval.
- **Database Management Model:** Managing the database is crucial for storing and retrieving data related to users, candidates, ballots, votes, and election configurations. This model involves setting up database schemas, defining database models, and implementing CRUD (Create, Read, Update, Delete) operations using libraries like Prisma or Mongoose.
- **Authentication and Authorization Model:** Implementing authentication and authorization mechanisms ensures that only authorized users can access certain functionalities of the voting system. This model involves verifying

user credentials, generating and validating JSON Web Tokens (JWT), and enforcing access control rules based on user roles and permissions.

- **Vote Counting and Result Tabulation Model:** This model handles the logic for counting votes and tabulating election results based on the ballots cast by voters. It involves aggregating and analyzing vote data stored in the database, applying any relevant election rules (e.g., proportional representation), and generating final election results.
- **Real-time Updates Model:** Providing real-time updates to users about the status of the election, such as live vote counts or result announcements, enhances the user experience. This model involves implementing real-time communication between the server and client using technologies like WebSockets or Server-Sent Events.
- **Security and Data Protection Model:** Ensuring the security and integrity of the voting system's data is paramount to prevent tampering, manipulation, or unauthorized access. This model involves implementing security measures such as input validation, data encryption, HTTPS encryption for network communication, and protection against common web vulnerabilities (e.g., SQL injection, Cross-Site Scripting).

4.3.7.1 To do an Backend Implementation (Next.js) Module for Secure Voting Platform using Blockchain, we'll brak down Module 7 Algorithm into several steps:

- **Step 1 :** Verify user credentials and validate JSON Web Tokens (JWT) for authentication.
- **Step 2 :** Provide users with live updates on election status, vote counts, and result announcements.
- **Step 3 :** Implement security measures such as input validation, data encryption, and HTTPS encryption for network communication.

4.4 SUMMARY

The project integrates Ganache for blockchain emulation, MetaMask for network interaction, and Nodemailer for OTP dispatch. Users, divided into Voters and Administrators, engage through HTML/CSS/JavaScript front-end, with Next.js for backend. This comprehensive setup ensures a secure, transparent, and user-friendly online voting system.

CHAPTER 5

SYSTEM REQUIREMENT

5.1 GENERAL

This chapter explains about the software and hardware requirements and their specifications. The requirements listed here are used to satisfy system design and other implementation design.

5.2 SYSTEM REQUIREMENTS

5.2.1 HARDWARE REQUIREMENTS

The hardware components are used to develop the systems and to achieve the objectives of the system.

- System : Intel i3 5th Generation or Higher
- Hard Disk : 500 GB
- Monitor : 14” LCD monitor display
- Ram : 8 GB

5.2.2 SOFTWARE REQUIREMENTS

The software requirements of the system can also be enlisted in terms of that is used to achieve the objectives of the System and those that will assist the former.

- Operating system : Windows 10 or Higher
- Programming Language : React js, Meta Mask, Ganache, Solidity Compiler
- Tool : Visual Studio Code , Anaconda

5.3 TECHNICAL SPECIFICATION

5.3.1 Node.js

Node.js, a powerful runtime environment built on Chrome's V8 JavaScript engine, has revolutionized the landscape of server-side programming since its inception in 2009. Developed by Ryan Dahl, Node.js provides an event-driven architecture and a non-blocking I/O model, making it particularly well-suited for building scalable and high-performance applications. At its core, Node.js enables developers to write server-side code using JavaScript, a language traditionally associated with client-side scripting in web browsers. This unification of client and server-side development simplifies the development process, allowing developers to leverage their existing JavaScript skills across the entire stack.

One of the key features of Node.js is its event-driven architecture. Unlike traditional server-side environments that use a multi-threaded model, Node.js operates on a single-threaded, event-driven paradigm. This means that instead of creating a new thread for each incoming request, Node.js employs a single event loop that efficiently manages asynchronous operations. As a result, Node.js can handle a large number of concurrent connections without incurring the overhead associated with thread creation and context switching. This event-driven model is particularly advantageous for applications that require real-time communication or handle a high volume of I/O operations, such as chat applications, streaming services, or data-intensive APIs.

This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Under the Python release. Node.js v20.12.2 (October 2010), Python v16.0.x and 18.4.x versions, Node.js 20.5.3 (December 2020).

Features of Nodejs

- **Asynchronous and Event-Driven:** Node.js uses a non-blocking, event-driven architecture, allowing it to handle multiple concurrent operations efficiently. This makes it particularly suitable for building scalable network applications.
- **Single-Threaded, Non-Blocking I/O Model:** Node.js operates on a single-threaded event loop, which enables it to handle multiple requests simultaneously without getting blocked by I/O operations. This architecture makes Node.js highly efficient for handling I/O-heavy tasks.
- **V8 JavaScript Engine:** Node.js is built on top of Google's V8 JavaScript engine, known for its high performance and speed. V8 compiles JavaScript code directly into machine code, resulting in fast execution.
- **NPM (Node Package Manager):** Node.js comes with npm, a package manager that provides access to a vast ecosystem of open-source libraries and modules. npm allows developers to easily install, manage, and share reusable code, making development faster and more efficient.
- **Cross-Platform:** Node.js is cross-platform and can run on various operating systems, including Windows, macOS, and Linux. This allows developers to write code once and run it anywhere, making Node.js applications highly portable.
- **Scalability:** Node.js is highly scalable and well-suited for building real-time, data-intensive applications. Its event-driven architecture and non-blocking I/O model enable it to handle large numbers of concurrent connections with low overhead, making it ideal for building scalable web servers and applications.
- **Extensive Ecosystem:** Node.js has a vibrant and active ecosystem with a wide range of frameworks, libraries, and tools. Frameworks like Express.js provide a lightweight and flexible environment for building web applications, while

libraries like Socket.IO enable real-time communication between clients and servers.

- **Easy-to-Learn:** Node.js is relatively easy to learn, especially for developers familiar with JavaScript. Since both the frontend and backend can be written in JavaScript, developers can leverage their existing skills to build full-stack applications with Node.js.

Apart from the above-mentioned features, Nodejs has a big list of good features, few are listed below –

- Node.js utilizes an event-driven architecture, allowing developers to build highly responsive and scalable applications by handling events asynchronously.
- Node.js enables developers to write code that can run seamlessly across different operating systems, reducing development time and effort.
- Node.js supports data streaming, making it efficient for handling large files and real-time data processing.
- Node.js includes a built-in HTTP module that simplifies the creation of web servers, making it easy to build fast and scalable web applications.
- Node.js integrates seamlessly with CI/CD pipelines, enabling automated testing, building, and deployment of applications, leading to faster release cycles and improved productivity.

5.4 SUMMARY

This chapter summarizes about the usage of technologies. It also explains about the associated entities which are used for the implementation.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

In conclusion, the blockchain-based voting system we've developed represents a significant step forward in making elections more secure, transparent, and accessible. By using advanced technologies like blockchain, we've created a system that ensures each vote is recorded accurately and cannot be tampered with. This enhances trust in the electoral process and gives voters confidence that their voices are being heard. With features like real-time result tracking for users and comprehensive management tools for administrators, our system provides a seamless voting experience for all stakeholders. By incorporating user-friendly interfaces and reliable communication channels like email, we've made it easy for voters to participate and for administrators to manage the process efficiently. Overall, our project aims to modernize the way elections are conducted, promoting democracy and civic engagement. By harnessing the power of technology, we're working towards a future where elections are fair, transparent, and accessible to all.

APPENDIX 1

SCREENSHOTS

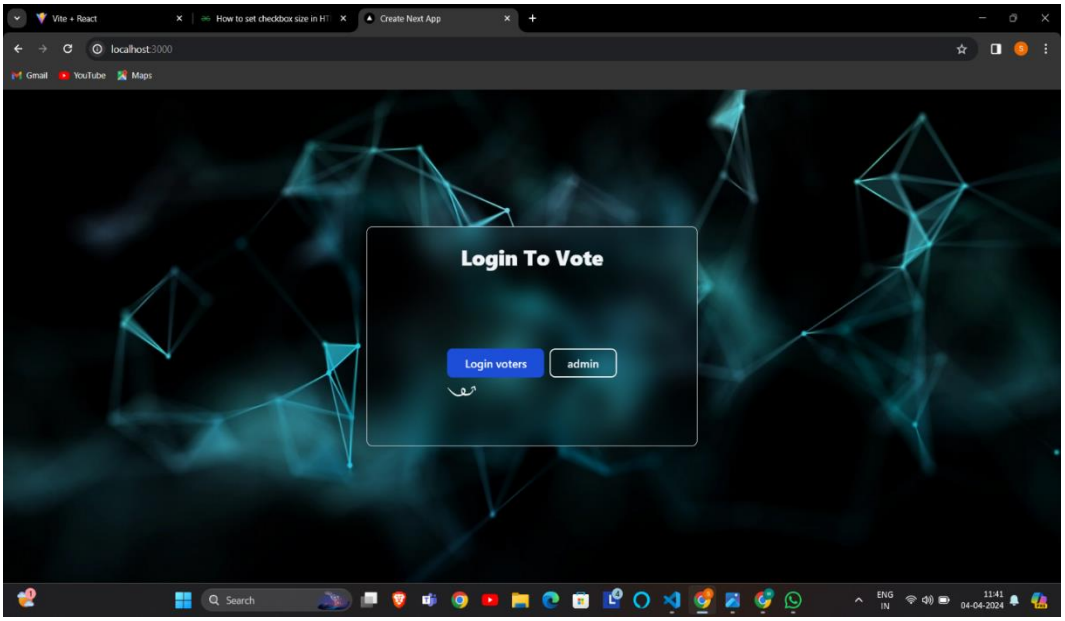

Figure A1.1 Login to Vote From

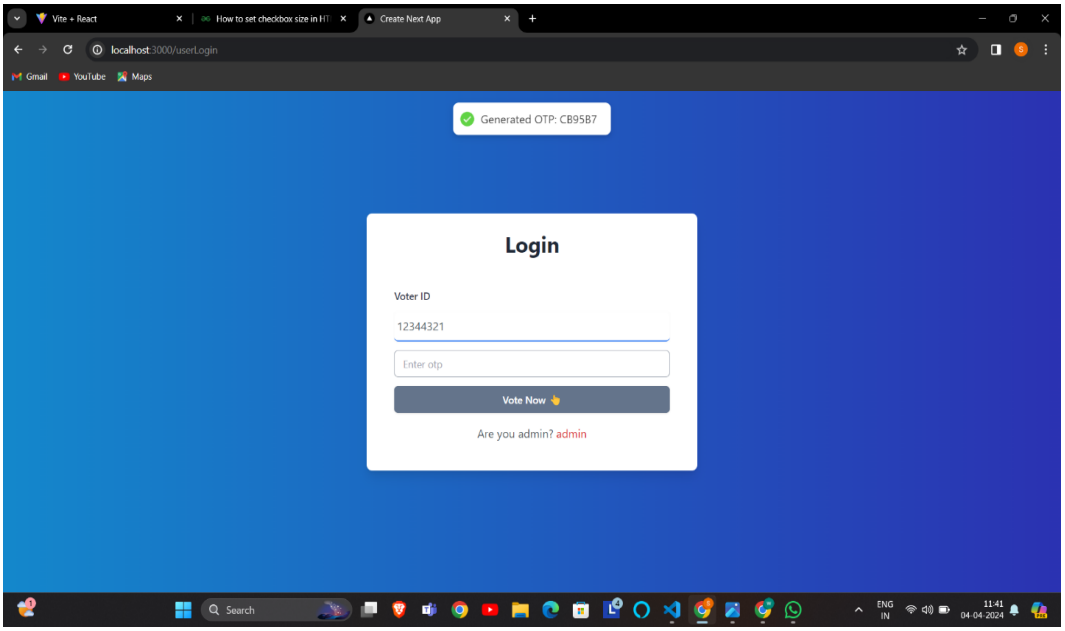

Figure A1.2 User Login From

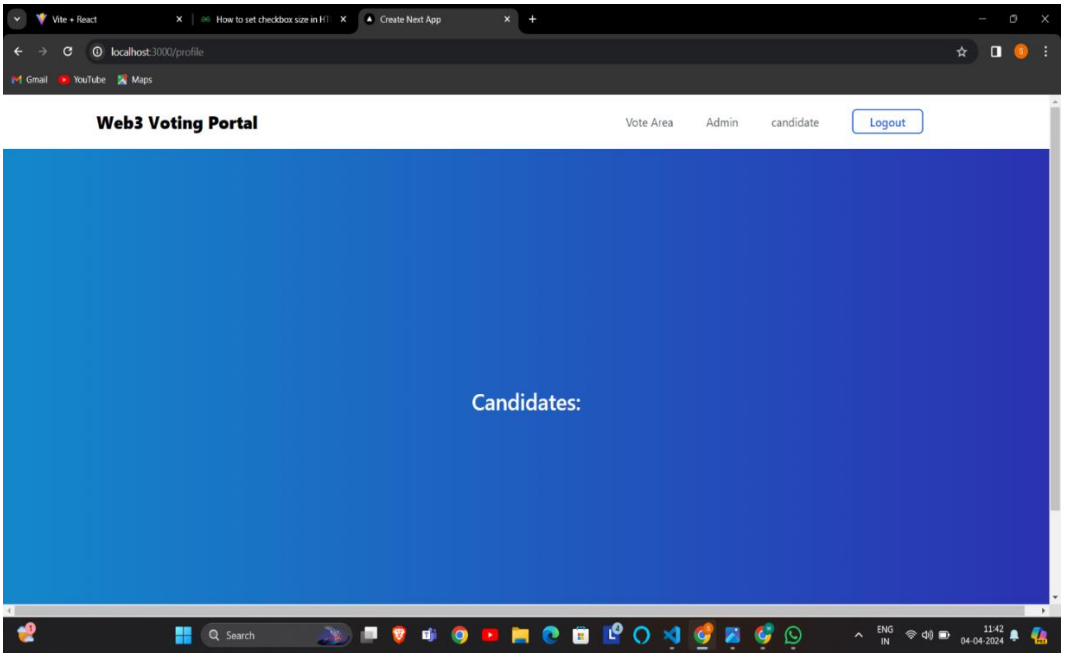

Figure A1.3 User Voting portal

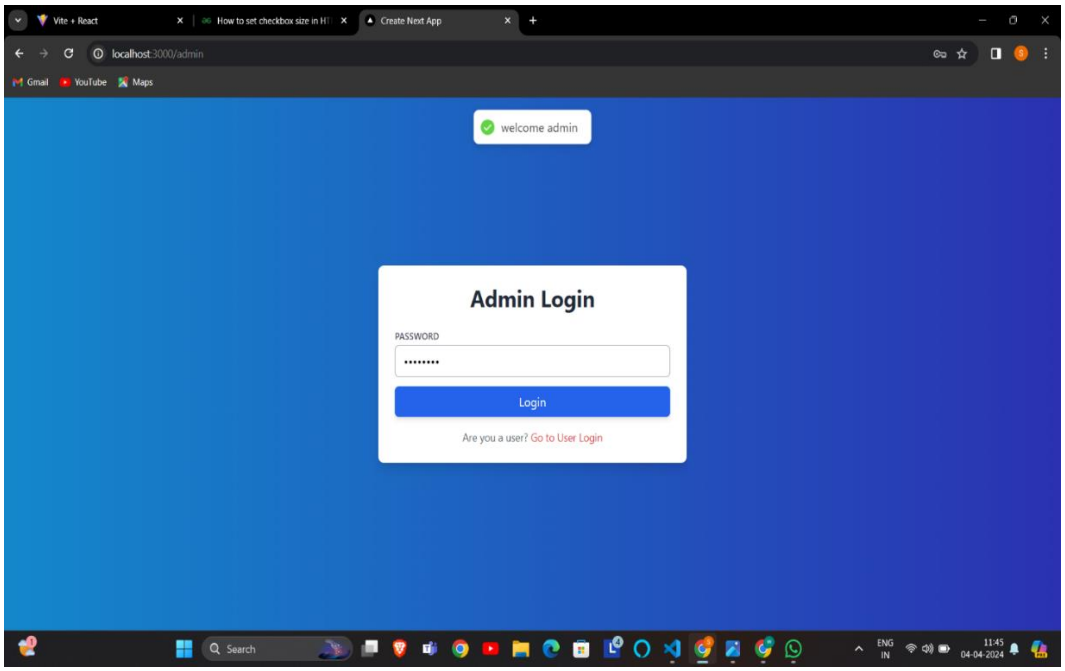

Figure A1.4 Admin Login From

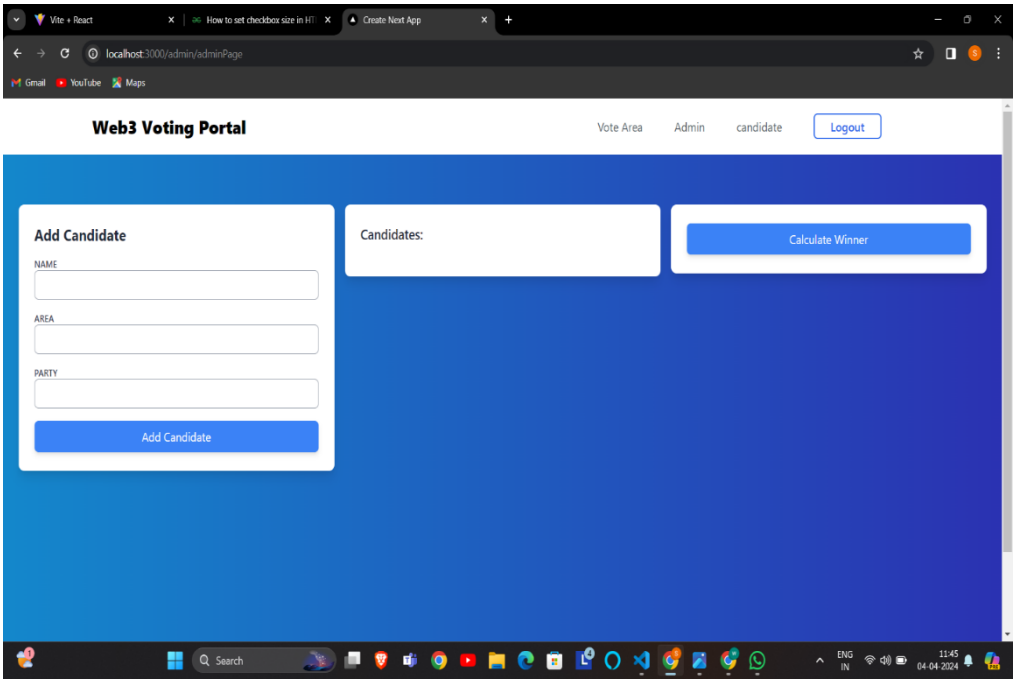

Figure A1.5 Admin Voting Portal

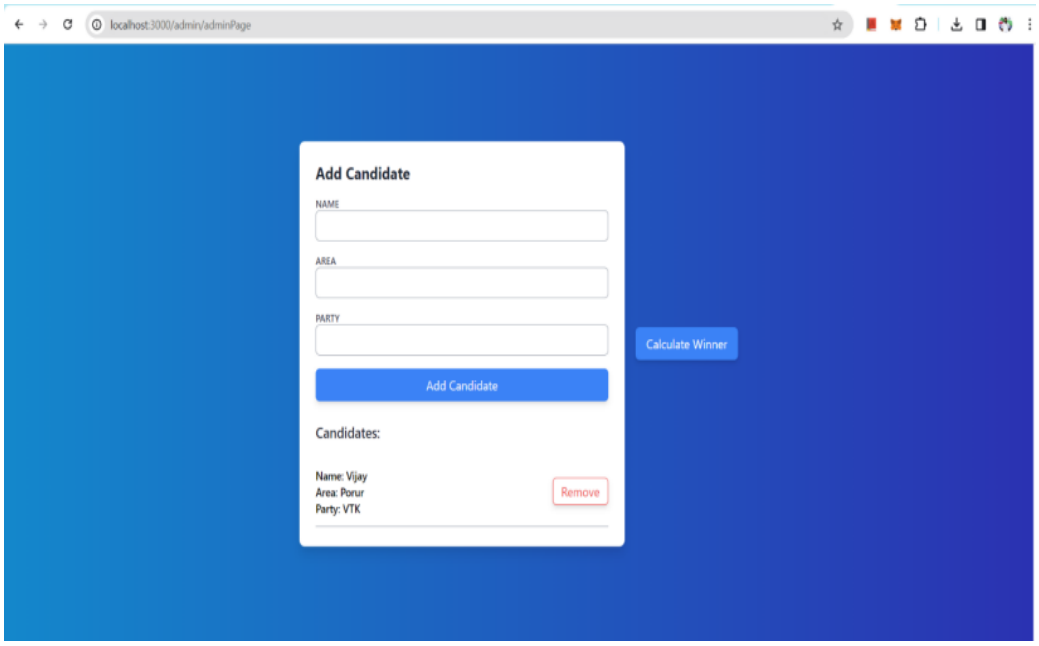

Figure A1.6 Admin Dashboard

APPENDIX 2

SAMPLE CODING

CODE SNIPPETS

VotingSystem.sol

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.5.0 <0.9.0;
contract VotingSystem{
 address public admin;
 string public winningParty;
 uint public Seats = 0;
 struct Candidate {
 uint id;
 string name;
 string area;
 string party;
 uint voteCount;
 }
 string[] public Ar;
 struct Area {
 string name;
 string leadingParty;
26
 uint totalVote;
 }
 string[] public Pr;
 struct Party{
 string name;
 uint totalSeatsWon;
 }
 mapping(uint => Candidate) public candidates;
 mapping (string => Area) public areas;
 mapping (string => Party) public parties;
 uint public candidatesCount;
 uint public areasCount;
 uint public partiesCount;
 mapping(address => bool) public voters;
```

```
 modifier onlyAdmin() {
 require(msg.sender == admin, "Only admin can perform this action");
 _;
 }
 constructor() {
 admin = msg.sender;
 }
 function addCandidate(string memory _name, string memory _area, string
memory
_party) public onlyAdmin {
 candidatesCount++;
 candidates[candidatesCount] = Candidate(candidatesCount, _name, _area, _party,
0);

if(keccak256(abi.encodePacked(_area))==keccak256(abi.encodePacked(areas[_are
a].na
me))){
 //
 }
 else{
 areasCount++;
 areas[_area]=Area(_area,"",0);
 Ar.push(_area);
 }

if(keccak256(abi.encodePacked(_party))==keccak256(abi.encodePacked(parties[_
party]
.name))){
 //
 }
 else{
 partiesCount++;
 parties[_party]=Party(_party,0);
 Pr.push(_party);
 }
 }
 // remove candidates
 function removeCandidate(uint _candidateId) public onlyAdmin {
 require(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid
candidate
```

```
ID");
 delete candidates[_candidateId];
 // You might want to implement logic here to reorganize candidate IDs if needed
 }
 function vote(uint _candidateId) public {
28
 require(!voters[msg.sender], "You have already voted");
 require(_candidateId > 0 && _candidateId <= candidatesCount, "Invalid
candidate
ID");
 Candidate storage candidate = candidates[_candidateId];
 candidate.voteCount++;
 voters[msg.sender] = true;
 if(areas[candidate.area].totalVote<candidate.voteCount){
 areas[candidate.area].totalVote=candidate.voteCount;
 areas[candidate.area].leadingParty=candidate.party;
 }
 }
 function calculateWinner() public{
 for(uint i=0;i<areasCount;i++){
 parties[areas[Ar[i]].leadingParty].totalSeatsWon++;
 }
 for(uint i=0;i<partiesCount;i++){
 if(parties[Pr[i]].totalSeatsWon>Seats){
 Seats=parties[Pr[i]].totalSeatsWon;
 winningParty=Pr[i];
 }
 }
 }
 function getOverallWinningParty() public view returns (string memory, uint) {
 return(winningParty,Seats);
}
}
```

REFERENCES

1. Apoorva S. Drakshayani, U. Vijayalakshmi, S. R. Sri, A. Srivani and A. Vyshnavi, "Online Voting System Using Blockchain," 2022 International Conference on Electronics and Renewable Systems (ICEARS).

2. T. Dimitriou, "Efficient coercion-free and universally verifiable blockchain-based voting", *Comput. Netw.*, vol. 174, Jun. 2020.

3. J. Huang, D. He, M. S. Obaidat, P. Vijayakumar, M. Luo and K.-K.-R. Choo, "The application of the blockchain technology in voting systems: A review", ACM Comput. Surv., vol. 54, no. 3, pp. 1-28, Apr. 2022.

4. T. Vairam, S. Sarathambekai and R. Balaji, "Blockchain based Voting system in Local Network," 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS).

5. Voting Using Blockchain," 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA).


ANNA UNIVERSITY
PROGRESS THROUGH KNOWLEDGE


ANNA UNIVERSITY
PROGRESS THROUGH KNOWLEDGE


MEENAKSHI
COLLEGE OF ENGINEERING