

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY**  
**BELAGAVI-590018**



**A Project Report**  
**on**  
**Online Voting System using Blockchain**

*Submitted in partial fulfillment of the requirements for the final year degree in*  
**Bachelor of Engineering in Computer Science and Engineering**  
*of Visvesvaraya Technological University, Belagavi*

**Submitted by**

**Navaneeth N**                **1RN21CS098**  
**Prarthana R**                **1RN21CS111**  
**Rahul G Athreyas**        **1RN21CS118**

**Under the Guidance of**

**Ranjith V**  
**Assistant Professor**  
**Dept. of CSE**



**Department of Computer Science and Engineering**

**RNS Institute of Technology**

**Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi**  
**NACC 'A + Grade' Accredited, NBA Accredited (UG-CSE, ECE, ISE, EIE and EEE)**  
**Channasandra, Dr. Vishnuvardhan Road, Bengaluru-560098**  
**Ph:(080)28611880, 28611881 URL:www.rnsit.ac.in**

**2024-2025**

**RN SHETTY TRUST®**

**RNS INSTITUTE OF TECHNOLOGY**

Affiliated to VTU, Recognized by GOK, Approved by AICTE, New Delhi  
NACC 'A + Grade' Accredited, NBA Accredited (UG-CSE,ECE,ISE,EIE and EEE)

Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098

Ph:(080)28611880,28611881 URL:www.rnsit.ac.in

**DEPARTMENT OF COMPUTER SCIENCE ENGINEERING**



**CERTIFICATE**

Certified that the Project work entitled **Online Voting System using Blockchain** has been successfully carried out at **RNSIT** by **Navaneeth N** bearing **1RN21CS098**, **Prarthana R** bearing **1RN21CS111**, **Rahul G Athreyas** bearing **1RN21CS118** bonafide students of **RNS Institute of Technology** in partial fulfillment of the requirements of final year degree in **Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi** during academic year **2024-2025**. The Project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

\_\_\_\_\_  
**Ranjith V**

Assistant Professor  
Dept. of CSE, RNSIT

\_\_\_\_\_  
**Dr. Kavitha C**

Dean and Head  
Dept. of CSE , RNSIT

\_\_\_\_\_  
**Dr. Ramesh Babu H S**

Principal  
RNSIT

**External Viva**

**Name of the Examiners**

**Signature with Date**

1.

2.

# Acknowledgement

I extend my profound thanks to the **Management of RNS Institute of Technology** for fostering an environment that promotes innovation and academic excellence.

I want to express my gratitude to our beloved Director, **Dr. M K Venkatesha**, and Principal, **Dr. Ramesh Babu H S**, for their constant encouragement and insightful support. Their guidance has been pivotal in keeping me motivated throughout this endeavour.

My heartfelt appreciation goes to **Dr. Kavitha C**, Dean and HoD of Computer Science and Engineering, for her vital advice and constructive feedback, which significantly contributed to shaping this project.

I also thank, Project Coordinators, for continuous monitoring and ensuring the process was deployed as per schedule.

I am deeply grateful to project guide, **Ranjith V**, Assistant Professor, for their unwavering support, guidance, and valuable suggestions throughout the duration of this project. Lastly my thanks go to all the teaching and non-teaching staff members of the Computer Science and Engineering Department for their encouragement, cooperation and support have been invaluable during this journey.

Warm Regards,

**Navaneeth N (1RN21CS098)**

**Prarthana R (1RN21CS111)**

**Rahul G Athreyas (1RN21CS118)**

# **Abstract**

The "Online Voting System using Blockchain" project aims to revolutionize the traditional voting process by leveraging blockchain technology, ensuring secure, transparent, and tamper-proof elections. This decentralized system eliminates the risks of vote manipulation, as any tampering with the blockchain invalidates the entire chain and restores it to a valid state. A key feature of the system is the real-time voter authentication through face detection, ensuring that only authorized voters can access the platform. The system also incorporates multi-factor authentication for enhanced security and privacy. Designed to be scalable, it can be deployed for both small-scale and large-scale elections, offering a robust solution to the flaws of traditional voting systems. Future improvements may include Aadhaar card integration for biometric authentication, mobile voting features, and further scalability enhancements, aiming to provide an even more accessible, secure, and efficient voting experience.

# Contents

|   |           |
|---|-----------|
| <b>Acknowledgement</b>                              | <b>i</b>  |
| <b>Abstract</b>                                     | <b>ii</b> |
| <b>List of Figures</b>                              | <b>v</b>  |
| <b>1 INTRODUCTION</b>                               | <b>1</b>  |
| 1.1 Introduction about the project . . . . .        | 1         |
| 1.2 Existing System and its limitations . . . . .   | 2         |
| <b>2 LITERATURE SURVEY</b>                          | <b>4</b>  |
| 2.1 Relevant recent paper's summary . . . . .       | 5         |
| 2.2 Conclusion about literature survey . . . . .    | 6         |
| <b>3 PROBLEM STATEMENT</b>                          | <b>7</b>  |
| 3.1 Objectives . . . . .                            | 7         |
| <b>4 SYSTEM ARCHITECTURE</b>                        | <b>8</b>  |
| 4.1 Schema diagram . . . . .                        | 8         |
| 4.1.1 Voter Collection . . . . .                    | 9         |
| 4.1.2 Constituency Collection . . . . .             | 9         |
| 4.1.3 Candidate Collection . . . . .                | 10        |
| 4.1.4 Admin Collection . . . . .                    | 10        |
| 4.1.5 Time Collection . . . . .                     | 11        |
| 4.1.6 Relationships in the Schema Diagram . . . . . | 11        |
| 4.2 Functional Requirements . . . . .               | 12        |

|          |   |           |
|----------|---|-----------|
| 4.3      | Non-Functional Requirements . . . . .   | 12        |
| 4.4      | User Requirements . . . . .             | 12        |
| <b>5</b> | <b>IMPLEMENTATION</b>                   | <b>13</b> |
| 5.1      | Flow Diagram . . . . .                  | 13        |
| 5.2      | Libraries Used . . . . .                | 15        |
| 5.3      | Algorithms/Methods/Pseudocode . . . . . | 18        |
| <b>6</b> | <b>RESULT AND SNAPSHOTS</b>             | <b>22</b> |
| <b>7</b> | <b>CONCLUSION</b>                       | <b>27</b> |
| 7.1      | Conclusion . . . . .                    | 27        |
| 7.2      | Future Enhancements . . . . .           | 28        |
|          | <b>References</b>                       | <b>31</b> |

# List of Figures

|     |  |    |
|-----|--|----|
| 4.1 | Schema diagram . . . . .               | 8  |
| 5.1 | Schema diagram . . . . .               | 13 |
| 6.1 | Main page . . . . .                    | 22 |
| 6.2 | Login page . . . . .                   | 23 |
| 6.3 | Voting Page . . . . .                  | 23 |
| 6.4 | Successfully casted the vote . . . . . | 24 |
| 6.5 | Admin page . . . . .                   | 24 |
| 6.6 | Results page . . . . .                 | 25 |
| 6.7 | Tie condition . . . . .                | 25 |
| 6.8 | No winner condition . . . . .          | 26 |

# Chapter 1

## INTRODUCTION

Traditional voting systems often face challenges such as human errors, tampering, and logistical inefficiencies, which can compromise the integrity of elections. To address these issues, "Online Voting System Using Blockchain" offers a modern and secure alternative. By decentralizing the voting process and ensuring tamper-resistant vote recording, the system significantly reduces risks of manipulation and errors. Additionally, it enhances accessibility by allowing voters to cast their votes online, eliminating geographical barriers and improving participation rates.

Our project, "Online Voting System Using Blockchain," leverages the MERN stack to develop a web-based e-voting platform that is efficient, reliable, and user-friendly. By combining decentralized principles with internet technologies, it ensures transparency, privacy, and security in the voting process. This solution overcomes traditional voting flaws while fostering trust among voters and stakeholders, paving the way for a more modern and trustworthy electoral process.

### 1.1 Introduction about the project

Our platform goes beyond the traditional one-size-fits-all approach, recognizing that every learner is distinct. Through advanced data analytics, the system continuously evaluates user progress, identifying strengths and areas that require additional focus. Real-time feedback and adaptive assessments guide learners on their educational journey, promoting a dynamic and responsive learning environment.

The platform also facilitates collaborative learning by connecting users with similar interests or



---

learning objectives. Discussion forums, virtual study groups, and peer-to-peer interactions enhance the overall learning experience, fostering a sense of community among users. Accessibility is a key priority, with the platform being designed to accommodate various devices, ensuring that learning can take place anytime, anywhere. Furthermore, the incorporation of artificial intelligence enables predictive analytics, anticipating future learning needs and providing tailored recommendations.

## **1.2 Existing System and its limitations**

- **Issues in Traditional Voting Systems:**

- Traditional voting systems, including paper-based and electronic systems, suffer from fraud, manipulation, and lack of transparency.
- These systems are vulnerable to tampering, vote-rigging, and human errors, raising concerns about the integrity of the election process.
- Centralized control of the process creates a single point of failure, increasing the risk of malicious activities.

- **Security Challenges in Electronic Voting:**

- Existing electronic voting systems face significant security challenges, such as susceptibility to hacking and cyber-attacks.
- Encryption and authentication methods, although employed in some systems, do not provide the necessary level of security for such sensitive processes.
- Lack of transparency in vote tallying and the risk of voter coercion or manipulation add to the security concerns.

- **Scalability Issues:**

- Current e-voting systems often struggle with handling large-scale elections due to their limited scalability.
- These systems are not always equipped to manage the volume of data and the distributed nature of the election process.

- 
- Many systems lack mechanisms to ensure vote count verifiability, making it difficult for voters and election authorities to validate results.

- **Infrastructure and Implementation Challenges:**

- Implementing e-voting systems requires significant infrastructure and technical expertise, which may not be available in regions with limited digital infrastructure.
- The complexity and cost of implementing and maintaining these systems create barriers, particularly in developing countries.

- **Blockchain-based Solutions:**

- Blockchain offers a decentralized, transparent, and immutable solution that can address many of the limitations of traditional and electronic voting systems.
- However, blockchain introduces its own challenges, including scalability issues, complexity of implementation, and reliance on specialized technologies such as smart contracts and encryption.
- These challenges may make blockchain solutions unfeasible for some regions, particularly those with limited technological resources.

# Chapter 2

## LITERATURE SURVEY

The literature survey explores existing approaches to blockchain-based e-voting systems, focusing on the strengths and limitations of each system. Various papers have highlighted the advantages of blockchain in enhancing the security, transparency, and immutability of the voting process. Some key contributions include:

- **"Online Voting System"** (2024) by Noor Ahmed and Prof. Anupama Pattanasetty, proposes a secure blockchain voting system using smart contracts and adaptable consensus algorithms. While enhancing security and voter accessibility, it faces challenges related to digital barriers and voter confusion.
- **"BlockchainEnabledOnline-VotingSystem"** (2020) by Akhil Shah, Nishita Sodhia, and Shruti Saha, presents an Android-based voting application integrating blockchain with AES-128 encryption, SHA-256 hashing, and voter verification through OTPs. However, it is resource-intensive and requires sophisticated setup.
- **"Decentralized E-Voting Portal Using Blockchain"** (2019) by Dr. Swapnil Jain and Kriti Patidar, utilizes smart contracts deployed on the Ethereum blockchain for secure vote management. The limitations include technical dependencies and scalability challenges.
- **"Blockchain-Based E-Voting System"** (2018) by Fririk . Hjalmarsson and Gunnlaugur K. Hreiðarsson, explores blockchain as a service for e-voting systems. Although it enhances transparency, implementing such a system at a national scale faces technical challenges.

- 
- **"E-voting systems: A tool for e-democracy"** (2010) by Emad Abu-Shanab, Michael Knight, and Heba Refai, discusses the adoption of e-voting systems to improve accuracy and convenience. However, issues regarding security and inconsistent adoption across regions remain significant hurdles.

These systems demonstrate the potential of blockchain to revolutionize voting by ensuring data integrity and improving the election process. However, limitations such as scalability, complexity, and digital divides must be addressed for widespread adoption.

## **2.1 Relevant recent paper's summary**

Blockchain technology has been widely explored as a solution for securing and enhancing the transparency of online voting systems. Several studies highlight the potential of blockchain to address the challenges faced by traditional voting systems, such as tampering, security breaches, and lack of transparency. A key advantage of blockchain-based e-voting systems is the ability to offer immutable and verifiable records of votes, ensuring that votes cannot be altered or erased. This is achieved through the use of smart contracts and decentralized ledgers that guarantee vote integrity and eliminate the need for intermediaries, thus reducing the risks of manipulation and fraud.

Many blockchain-based e-voting systems incorporate advanced encryption methods, such as AES-128 and SHA-256, and multi-factor authentication techniques like OTPs, fingerprints, and biometric data to verify voter identities securely. These systems are designed to enhance accessibility and convenience for voters while providing a transparent and auditable process. However, the adoption of blockchain for e-voting is not without challenges. The integration of blockchain technology requires sophisticated infrastructure, and the technical complexity of implementing such systems can be resource-intensive. The use of Ethereum and other blockchain platforms may also create scalability issues, particularly in large-scale national elections, due to the limitations of transaction throughput and network congestion.

Furthermore, there are concerns about the digital divide, as many voters may not have the necessary access to the technology required for participation, especially in regions with limited internet connectivity or low levels of digital literacy. Security concerns, such as protecting voter privacy while ensuring transparency, remain a critical issue. Despite the promising potential of

---

blockchain to transform the electoral process, the system's success will depend on overcoming these technical, social, and economic challenges. The research suggests that while blockchain can offer a more secure and transparent voting process, additional work is needed to ensure that these systems are scalable, user-friendly, and widely accepted by voters and governments alike.

## **2.2 Conclusion about literature survey**

The exploration of blockchain technology for e-voting systems highlights its potential to significantly enhance the security, transparency, and accessibility of the voting process. Blockchain's inherent features, such as decentralization, immutability, and the use of smart contracts, provide a robust framework to prevent vote tampering and ensure the integrity of election results. Additionally, advanced encryption methods and authentication techniques offer greater protection for voter data, making the system more reliable and trustworthy.

However, the literature also reveals several challenges in the implementation of blockchain-based e-voting systems. Technical complexities, scalability issues, and the high resource demands of blockchain infrastructure remain significant hurdles, especially when dealing with large-scale elections. Furthermore, concerns around voter privacy, digital accessibility, and the adoption of the technology in regions with limited digital literacy or infrastructure need to be addressed. Despite these challenges, the potential benefits of blockchain in providing a secure, transparent, and efficient voting process make it a promising solution for modernizing electoral systems worldwide. Future research and technological advancements will likely help overcome these challenges, paving the way for the broader adoption of blockchain-based e-voting systems in the future.

# Chapter 3

## PROBLEM STATEMENT

Existing e-voting systems often lack security, transparency, and scalability, undermining trust in the voting process. The proposed solution leverages blockchain technology to address these challenges by providing immutable vote records, ensuring voter privacy, and enabling real-time tracking for enhanced transparency. The integration of a responsive frontend, developed using the MERN stack, delivers an intuitive user experience, while MongoDB handles the secure storage of non-sensitive election data. This combination of technologies creates a robust, scalable, and trustworthy e-voting platform, paving the way for more secure and transparent electoral processes.

### 3.1 Objectives

This project aims to develop an intelligent, adaptable e-learning platform that utilizes machine learning models to:

- To build a secure and transparent e-voting system using blockchain technology.
- To create an intuitive and user-friendly interface for the voting process using the MERN stack.
- To ensure the integrity of votes by decentralizing the storage of votes and the status of the voting process.
- To provide real-time monitoring and secure, anonymous data storage.
- To overcome limitations of existing e-voting systems, particularly in terms of scalability and security.

# Chapter 4

## SYSTEM ARCHITECTURE

### 4.1 Schema diagram

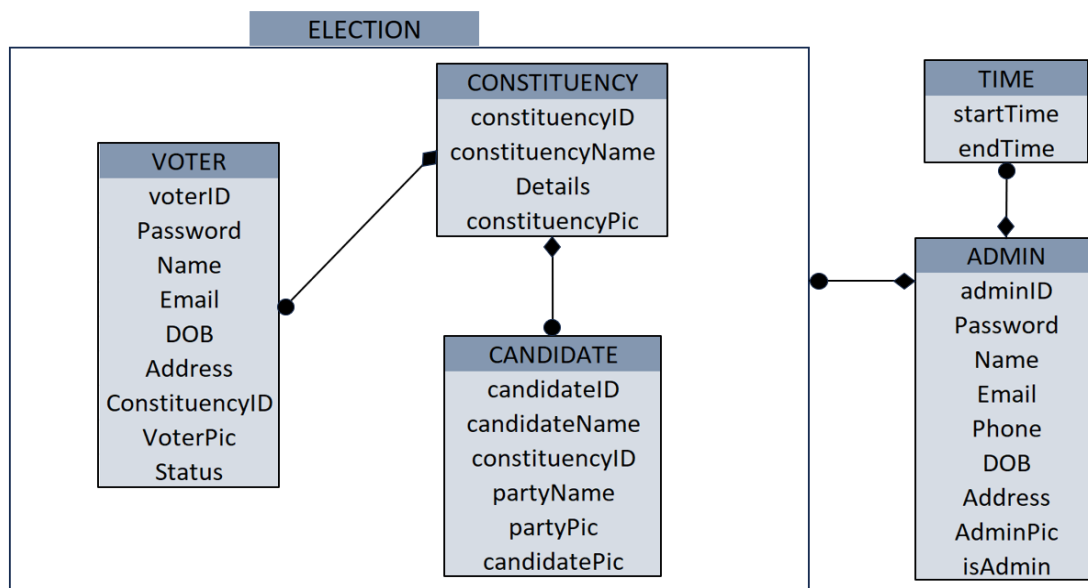


Figure 4.1: Schema diagram

---

### 4.1.1 Voter Collection

**Purpose:** Stores details about the registered voters.

**Fields:**

- **voterID:** A unique identifier for the voter.
- **Password:** Used for authentication during login.
- **Name, Email, DOB, Address:** Personal details of the voter.
- **ConstituencyID:** Links the voter to a specific constituency (foreign key relationship).
- **VoterPic:** Stores a picture of the voter for identity verification.
- **Status:** Tracks whether the voter has cast their vote or not (e.g., "voted" or "not voted").

**Relationships:**

- Linked to the **Constituency** collection through **ConstituencyID**.
- Indirectly linked to the **Time** collection as voters are bound by the voting schedule.

### 4.1.2 Constituency Collection

**Purpose:** Represents geographical areas or regions where elections take place.

**Fields:**

- **constituencyID:** A unique identifier for the constituency.
- **constituencyName:** The name of the constituency (e.g., "Downtown District").
- **Details:** Additional information about the constituency (optional field).
- **constituencyPic:** A visual representation or logo of the constituency.

**Relationships:**

- Connected to both **Voter** and **Candidate** through **ConstituencyID**.



---

### 4.1.3 Candidate Collection

**Purpose:** Stores information about candidates contesting in elections.

**Fields:**

- **candidateID:** A unique identifier for the candidate.
- **candidateName:** The candidate's name.
- **ConstituencyID:** Links the candidate to a specific constituency (foreign key relationship).
- **partyName:** The name of the political party the candidate belongs to.
- **partyPic:** A logo or symbol of the political party.
- **candidatePic:** A picture of the candidate.

**Relationships:**

- Linked to the **Constituency** collection through **ConstituencyID**.

### 4.1.4 Admin Collection

**Purpose:** Manages system operations and oversees the election process.

**Fields:**

- **adminID:** A unique identifier for the admin.
- **Password:** Used for admin authentication.
- **Name, Email, Phone, DOB, Address:** Personal details of the admin.
- **AdminPic:** A picture of the admin (optional, for identification purposes).
- **isAdmin:** A flag indicating the role of the admin (e.g., super admin or regular admin).

**Relationships:**

- Linked to the **Time** collection as admins are responsible for scheduling elections.

---

### 4.1.5 Time Collection

**Purpose:** Defines the voting session duration (start and end times).

**Fields:**

- **startTime:** The date and time when the voting session begins.
- **endTime:** The date and time when the voting session ends.

**Relationships:**

- Linked to the **Admin** collection because admins set up the voting schedule.

### 4.1.6 Relationships in the Schema Diagram

- **Voter → Constituency:**
  - Each voter belongs to a specific constituency.
  - This is represented by the **ConstituencyID** field in the **Voter** collection.
- **Candidate → Constituency:**
  - Each candidate is associated with a particular constituency where they are contesting for votes.
  - This is represented by the **ConstituencyID** field in the **Candidate** collection.
- **Admin → Time:**
  - Admins schedule the voting sessions, as shown by their connection to the **Time** collection.

---

## 4.2 Functional Requirements

- **User Authentication:** Authenticates voters securely using methods like multi-factor authentication, providing an authentication token or error.
- **Voting:** Allows voters to cast their votes securely and privately, confirming the vote cast.
- **Vote Counting:** Automatically tallies votes from the blockchain, ensuring security.
- **Result Display:** Shows real-time election results with visual representations like charts and graphs.

## 4.3 Non-Functional Requirements

- **Performance:** Handle a specified number of concurrent users with response times under 2 seconds.
- **Scalability:** Support scaling to accommodate increasing voter numbers during peak periods.
- **Security:** Ensure data encryption in transit and at rest, with protection against vulnerabilities like SQL injection, DDoS, and unauthorized access.
- **Usability:** Provide an intuitive and accessible user interface for all demographics.
- **Reliability:** Implement strong backup and disaster recovery solutions.

## 4.4 User Requirements

- **Voter Experience:** Voters should easily register, authenticate, and cast their votes via a user-friendly interface with immediate confirmation.
- **Transparency:** Provide real-time updates on election status and results, allowing voters to verify their vote was counted.
- **Security:** Ensure voters feel confident their votes are secure, private, and tamper-proof. This is achieved by hashing their passwords

# Chapter 5

## IMPLEMENTATION

### 5.1 Flow Diagram

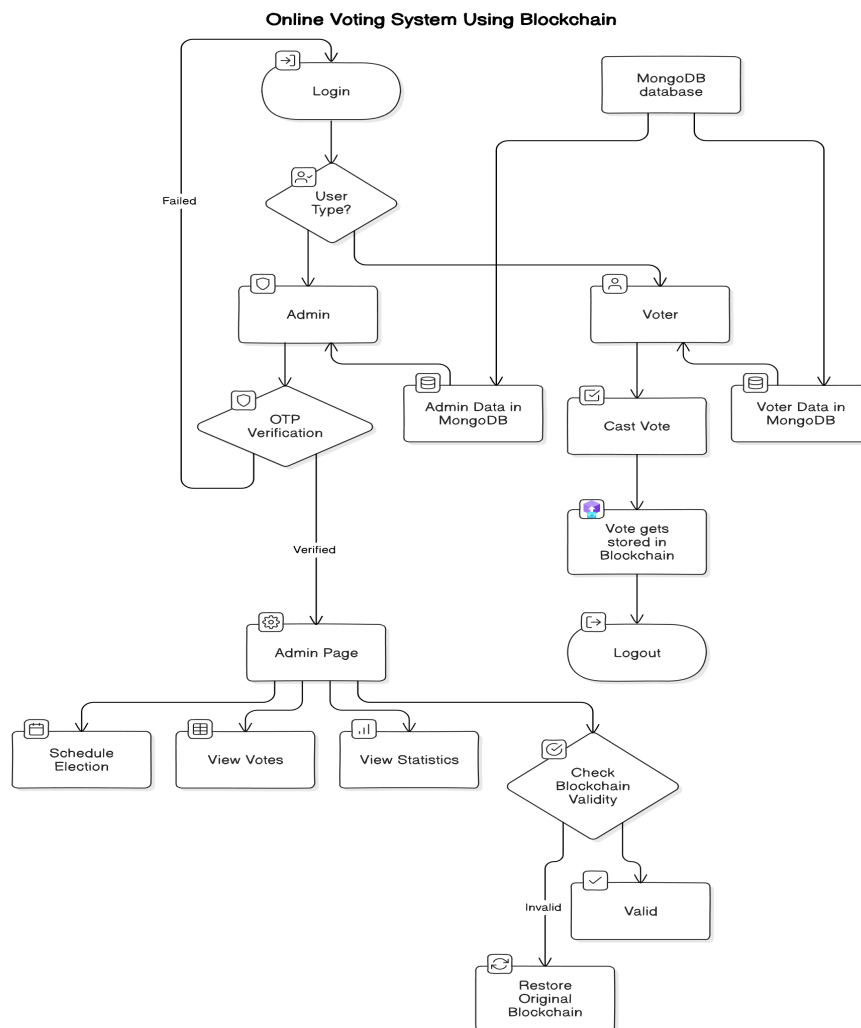


Figure 5.1: Schema diagram

---

The flowchart outlines the workflow of the **Online Voting System using Blockchain**. Below is the step-by-step explanation:

1. **Login:**

- The process begins with the user attempting to log in to the system using their credentials.

2. **User Type:**

- After login, the system determines whether the user is an **Admin** or a **Voter** based on their credentials.

3. **Admin Workflow:**

(a) **Admin Verification:**

- Admins undergo **OTP Verification** to ensure secure access to the system.
- If verification fails, the admin is redirected back to the login process.

(b) **Admin Page:**

- Upon successful verification, the admin gains access to the **Admin Page**, which provides several features:
  - **Schedule Election:** The admin sets the start and end times for elections.
  - **View Votes:** Allows the admin to view the votes cast in the system.
  - **View Statistics:** Displays overall election data and voting patterns.

(c) **Check Blockchain Validity:**

- The admin can verify the integrity of the blockchain storing the votes:
  - If the blockchain is **valid**, the process ends successfully.
  - If the blockchain is **invalid**, the system provides the option to restore the original, uncorrupted blockchain.

4. **Voter Workflow:**

(a) **Face Detection:**

- After the voter logs in, the system performs **face detection** for identity verification.

- 
- If the detected face matches the registered face, the voter is directed to the voting page.
  - If the face does not match, the voter is redirected back to the login page.

(b) **Access Voter Data:**

- The voter accesses their corresponding data stored in the **MongoDB database**.

(c) **Cast Vote:**

- The voter casts their vote for a candidate of their choice.

(d) **Store Vote in Blockchain:**

- Once a vote is cast, it is securely stored in the **blockchain** for transparency and immutability.

(e) **Logout:**

- After casting a vote, the voter logs out of the system, concluding their session.

5. **System Data Storage:**

- The **MongoDB database** serves as the backend, securely storing admin and voter details.
- Votes are recorded on the **blockchain**, ensuring tamper-proof data storage.

## 5.2 Libraries Used

### Frontend (React Libraries)

- **react:** React is a JavaScript library for building user interfaces, particularly single-page applications. It allows developers to create reusable UI components and manage application state efficiently. React uses a virtual DOM for optimized rendering performance, making UI updates faster and more efficient.
- **react-dom:** ReactDOM is the package responsible for rendering React components into the DOM (Document Object Model). It provides methods that allow React components to interact with the browser's DOM, enabling the update of the UI when the state of the application changes.

- 
- **react-router-dom:** React Router is a library used for handling routing in a React application. It enables navigation between different components or views without reloading the page. This provides a seamless, single-page application (SPA) experience for the user.
  - **fetch:** The Fetch API provides an interface for making HTTP requests in JavaScript. It supports promises and enables asynchronous requests to be made from the client-side, such as retrieving data from a server or sending data to a server without reloading the page.
  - **react-icons:** React Icons is a library that provides a collection of popular icons in React components. It makes it easy to include scalable vector icons in React applications without having to deal with image files or SVGs directly.
  - **react-bootstrap:** React Bootstrap is a library that provides pre-built Bootstrap components as React components. It allows developers to create responsive and mobile-first web applications using the Bootstrap design framework but within a React ecosystem.

## Backend (Express Flask Libraries)

- **express:** Express is a minimal and flexible Node.js web application framework. It provides robust features for web and mobile applications, including routing, middleware integration, and handling HTTP requests and responses.
- **mongoose:** Mongoose is an ODM (Object Document Mapper) for MongoDB, a library that provides a straightforward API for working with MongoDB databases in Node.js. It simplifies querying, data modeling, and schema creation for MongoDB collections.
- **flask:** Flask is a micro web framework for Python, used for building web applications and APIs. It is lightweight and easy to use, providing the essential tools for routing, request handling, and building RESTful APIs.
- **cors:** CORS (Cross-Origin Resource Sharing) is a mechanism that allows web applications to make requests to domains other than their own. Flask-CORS is a Flask extension that handles CORS headers, enabling secure communication between the frontend and backend even when hosted on different domains.

- 
- **requests:** Requests is a simple and elegant HTTP library for Python. It is designed to make sending HTTP requests more accessible and more human-friendly, offering methods like GET, POST, PUT, and DELETE to interact with REST APIs or web resources.
  - **pymongo:** PyMongo is a Python library that provides tools for working with MongoDB databases. It offers a straightforward API for interacting with MongoDB, including operations such as querying, updating, and inserting documents into collections.

## Cryptography and Data Handling Libraries

- **hashlib:** hashlib is a Python module that provides various algorithms for securely hashing data, including SHA-256, MD5, and others. It is commonly used for hashing passwords, files, or other sensitive data in applications.
- **json:** The json module in Python allows for parsing JSON data and converting Python objects into JSON format. It is useful for working with JSON-encoded data in APIs or for storing data in a human-readable format.

## Machine Learning Libraries

- **face-api.js:** face-api.js is a JavaScript library that provides real-time face detection and recognition capabilities. Built on TensorFlow.js, it can detect facial landmarks, recognize faces, and analyze emotions directly in the browser without relying on a server.
- **tensorflow.js:** TensorFlow.js is an open-source JavaScript library for training and deploying machine learning models in the browser or on Node.js. It enables running pre-trained models or training new models directly within the web application.



---

## 5.3 Algorithms/Methods/Pseudocode

### Blockchain Algorithms

#### SHA-256 (Hashing Algorithm)

- **Usage:** SHA-256 is a cryptographic hashing function that generates a fixed-size hash value from any input data. In your system, it ensures the integrity of the blockchain by producing unique, irreversible hash values for each block.
- **How it fits:** Each vote or block is hashed using SHA-256, creating a unique fingerprint for each piece of data. If any vote or block is altered, the hash value will change, making it easy to detect tampering and ensuring the immutability of the blockchain.

```
import hashlib  
  
def sha256_hash(data):  
    return hashlib.sha256(data.encode()).hexdigest()  
  
# Example usage  
print(sha256_hash("Vote data"))
```

#### Proof of Work (PoW)

- **Usage:** Proof of Work (PoW) is a consensus algorithm in which participants (miners) must solve a computationally difficult puzzle to add new blocks to the blockchain. It ensures security by requiring significant computational resources to add a block.
- **How it fits:** In your voting system, PoW ensures that blocks (containing votes) are only added after solving a complex puzzle. This makes it computationally expensive to alter any existing block, adding a layer of security and immutability to the voting data.

```
import hashlib  
  
def proof_of_work(block_data, difficulty=4):  
    nonce = 0  
    while True:  
        block_hash = hashlib.sha256(f"{block_data}{nonce}".encode()).hexdigest()
```

---

```
        if block_hash[:difficulty] == '0000':
            return nonce, block_hash
        nonce += 1

# Example usage
print(proof_of_work("Vote data"))
```

## Voting Algorithms

### Majority Voting Algorithm

- **Usage:** The majority voting algorithm is used to determine the winner in a voting system by counting the votes for each candidate. The candidate with the highest number of votes wins.
- **How it fits:** In your blockchain-based system, the majority voting algorithm is used to calculate the votes stored in the blockchain. After all the votes are cast, the candidate with the highest recorded votes is declared the winner.

```
def majority_voting(votes):
    return max(set(votes), key=votes.count)

# Example usage
votes = ['Alice', 'Bob', 'Alice']
print(majority_voting(votes))
```

## Security Algorithms

### One-Time Password (OTP) Authentication

- **Usage:** OTP is used to authenticate users by generating a time-sensitive, one-time password sent to the user's phone or email.
- **How it fits:** In your voting system, OTP is used to authenticate both voters and election administrators. The OTP is required before allowing access to vote or perform any administrative tasks, ensuring the identity of the user and adding an extra layer of security.

```
import random
```

---

```
def generate_otp():
    return random.randint(100000, 999999)

# Example usage
print(generate_otp())
```

### Face Detection (Facial Recognition)

- **Usage:** Face detection and recognition are used to verify the identity of voters by comparing their live facial image with a stored database.
- **How it fits:** Before casting a vote, voters are required to verify their identity using facial recognition. This ensures that only the registered voter can vote, preventing identity fraud and ensuring secure voting.

```
import cv2

# Load pre-trained face detection model
face_cascade = cv2.CascadeClassifier(cv2.data.harcascades + 'haarcascade_f

# Capture image and detect faces
cap = cv2.VideoCapture(0)
ret, frame = cap.read()
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_cascade.detectMultiScale(gray)

# Show detected faces
for (x, y, w, h) in faces:
    cv2.rectangle(frame, (x, y), (x+w, y+h), (255, 0, 0), 2)
cv2.imshow('Face Detection', frame)
cv2.waitKey(0)
cap.release()
cv2.destroyAllWindows()
```

---

## Consensus Algorithm

### Proof of Work (PoW) (Repeated for Consensus)

- **Usage:** Proof of Work (PoW) is used to validate new blocks in the blockchain. It requires miners to solve a difficult puzzle to add a new block, ensuring that the blockchain remains secure.
- **How it fits:** In your voting system, PoW is used to secure the blockchain. Each new block added to the blockchain must be validated by solving a computational puzzle, ensuring that malicious users cannot alter the vote data.

```
import hashlib

def proof_of_work(block_data, difficulty=4):
    nonce = 0
    while True:
        block_hash = hashlib.sha256(f"{block_data}{nonce}".encode()).hexdigest()
        if block_hash[:difficulty] == '0000':
            return nonce, block_hash
        nonce += 1

# Example usage
print(proof_of_work("Vote data"))
```

## Chapter 6

# RESULT AND SNAPSHOTS

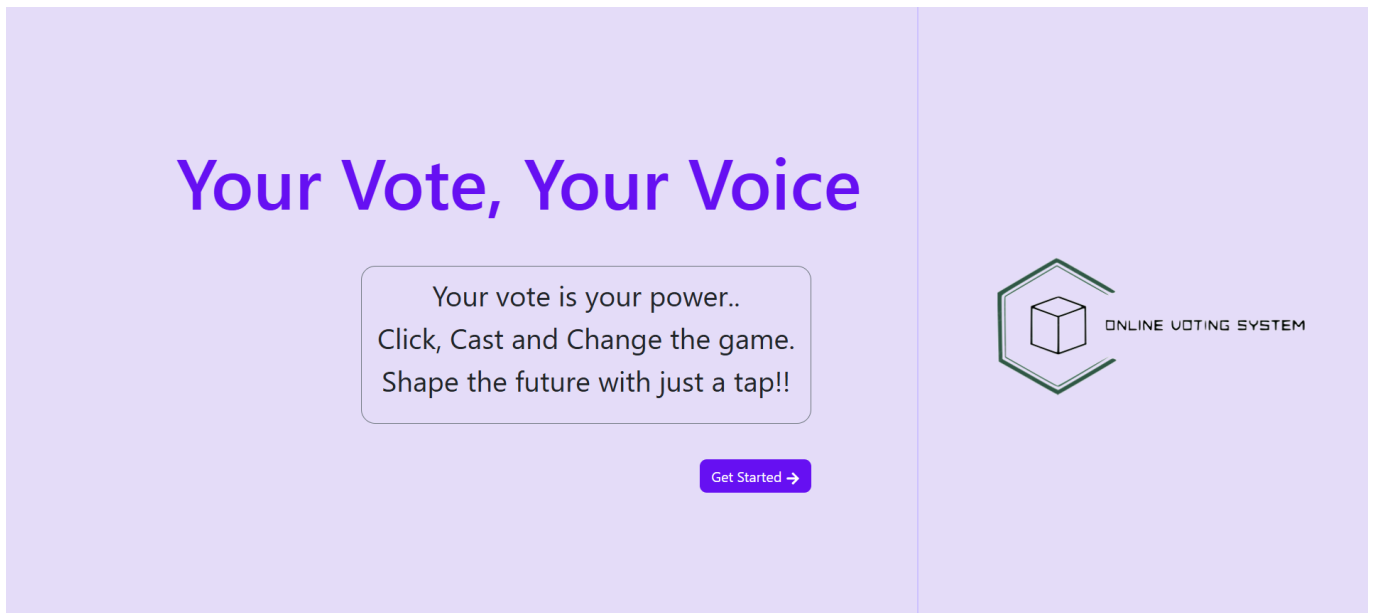



Figure 6.1: Main page



ONLINE VOTING SYSTEM

### Voter

Voter ID

Password


---


### Admin

Unique ID

Password

Figure 6.2: Login page


Logout



**Voter ID** : YER914804











**Name** : Navaneeth N

**Email** : navaneethnaren6@gmail.com


**Date of Birth** : 2003-10-15

**Address** : 1, 1st Main, M M Hills Road, Uttarahalli, Bengaluru, Karnataka 560061, India

### ELECTION COMMISSION

| Party Logo  | Candidate Image  | Party Name           | Candidate Name | Candidate ID                     |
|---|--|----------------------|----------------|----------------------------------|
|  |  | Unity Front          | Amit Sharma    | <input checked="" type="radio"/> |
|  |  | Progressive Alliance | Neha Verma     | <input type="radio"/>            |
|  |  | People's Voice       | Rahul Mehta    | <input type="radio"/>            |
|  |  | Nation Builders      | Priya Nair     | <input type="radio"/>            |
|  |  | Future Vision Party  | Anjali Gupta   | <input type="radio"/>            |

**Your vote:**



**Candidate Name:** Amit Sharma

**Party Name:** Unity Front

**Candidate ID:** 12435

Figure 6.3: Voting Page

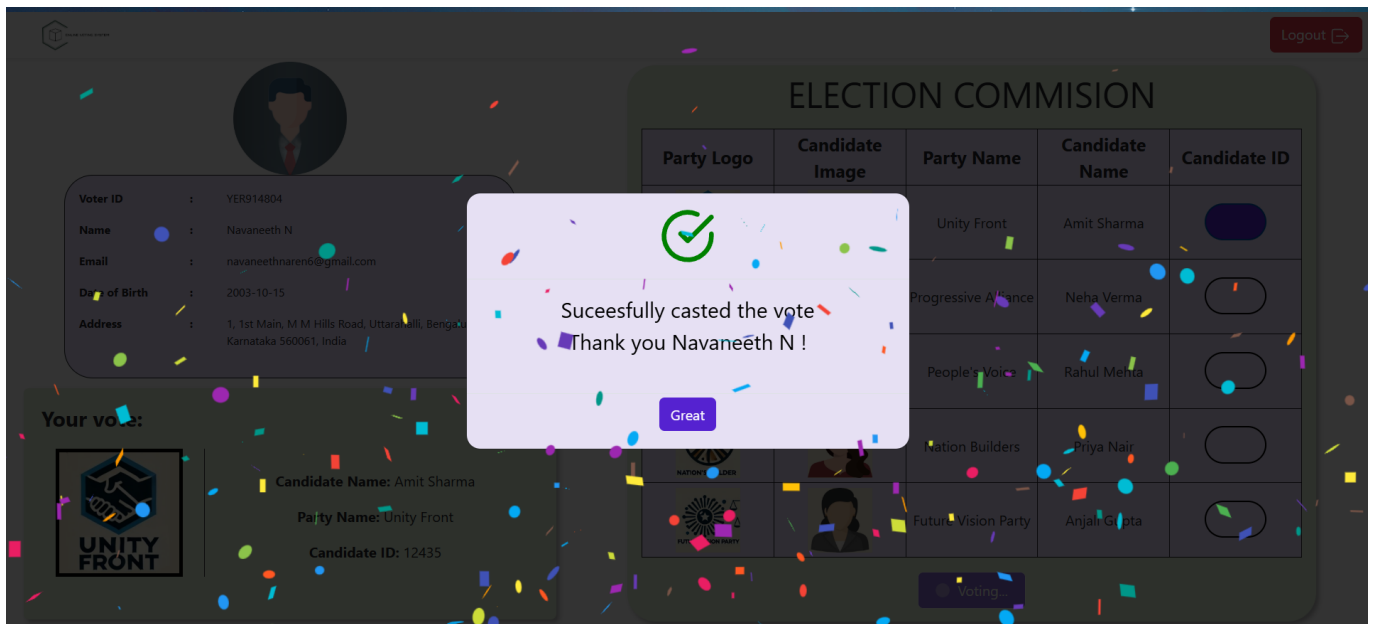


Figure 6.4: Successfully casted the vote

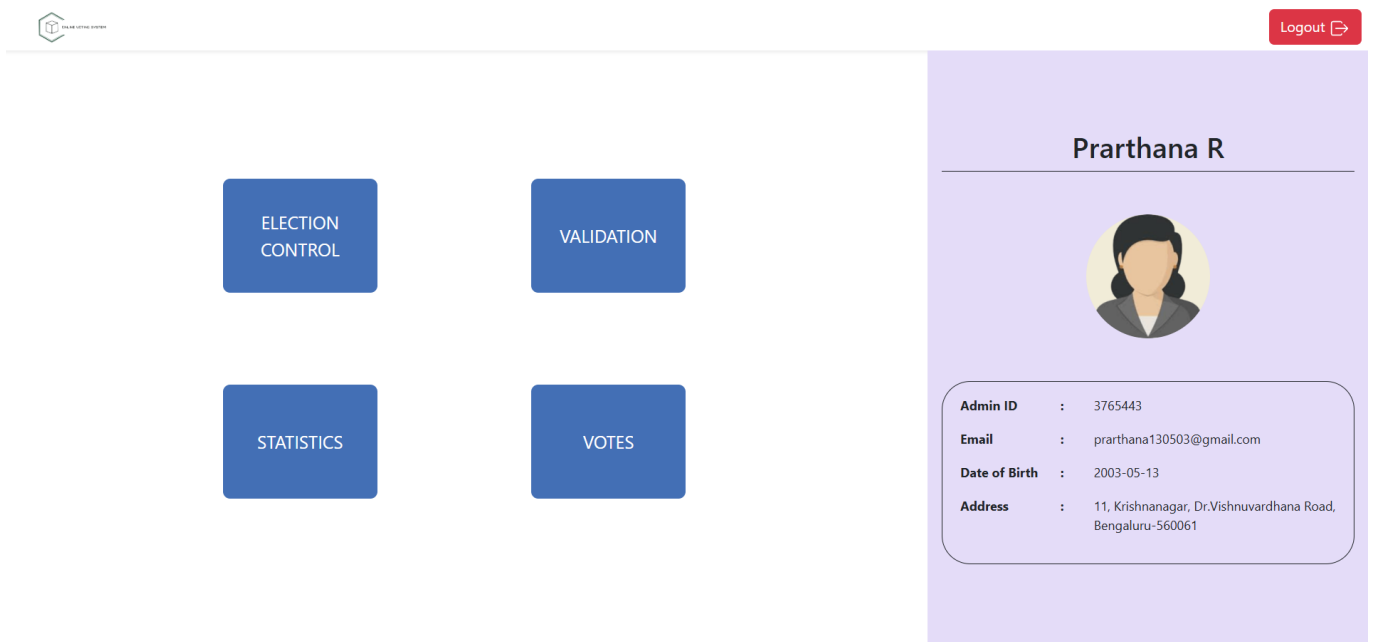


Figure 6.5: Admin page



## Results

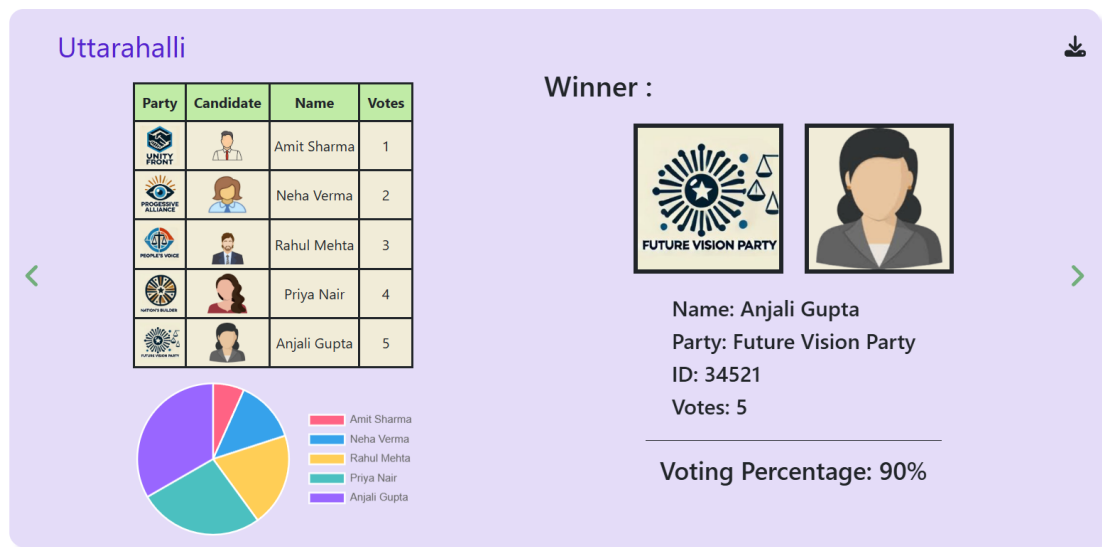


Figure 6.6: Results page



## Results

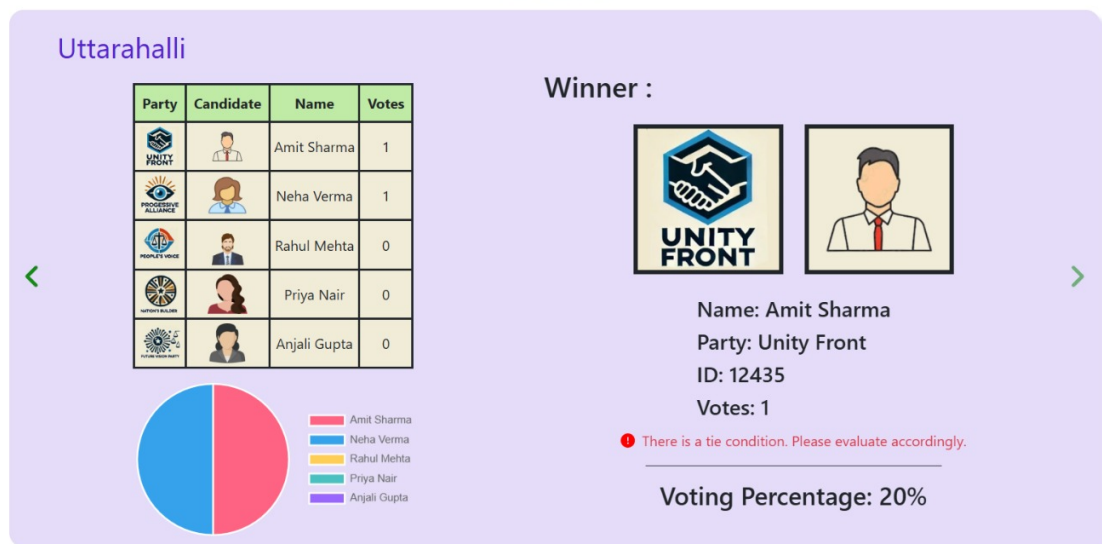


Figure 6.7: Tie condition





## Results

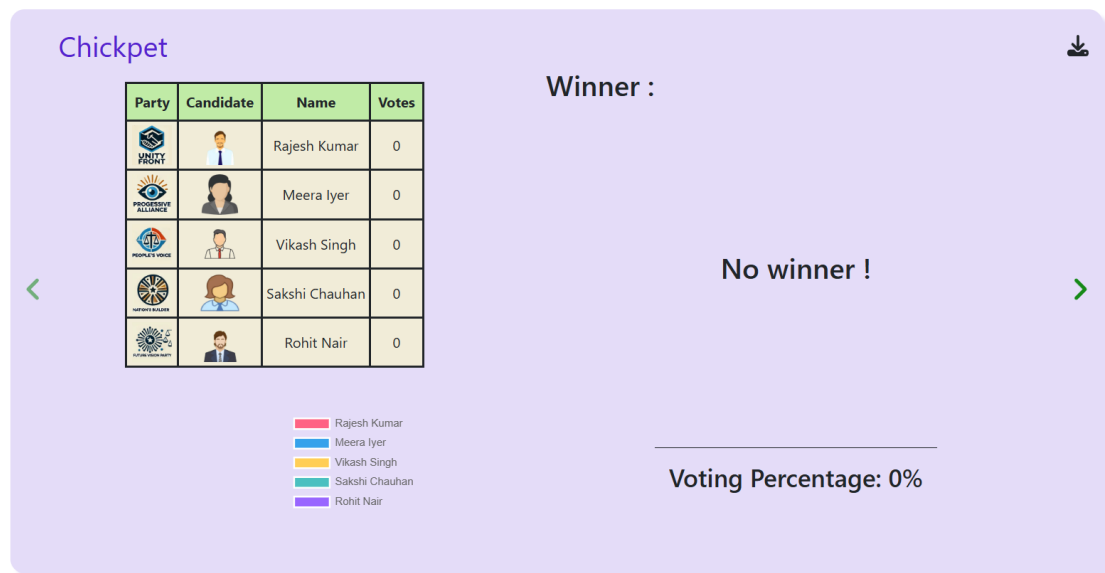


Figure 6.8: No winner condition

# Chapter 7

## CONCLUSION

### 7.1 Conclusion

The project involves the development of an "Online Voting System using Blockchain" that addresses the limitations of traditional voting systems and incorporates advanced security features. Below are the key points of the project:

- **Objective:** The platform provides a secure, transparent, and tamper-proof online voting system using blockchain technology to ensure the integrity of the electoral process.
- **Core Features:**
  - *Blockchain-based Security:* The system leverages blockchain to ensure that voting records are decentralized and immutable, preventing tampering or unauthorized alterations.
  - *Tamper-proof Blockchain:* Any tampering with the blockchain causes it to become invalid, and the system automatically restores the valid chain, preserving the integrity of the voting data.
  - *Real-time Vote Tallying:* The system offers transparent and real-time vote counting, ensuring immediate access to election results while maintaining security and privacy.
  - *Voter Authentication and Privacy:* In addition to OTP verification and encryption, the system incorporates face detection technology to authenticate voters in real-time before granting access to the voting platform, ensuring that only authorized voters can cast their ballots.

- 
- *Decentralization:* The blockchain-based design eliminates single points of failure, enhancing system security and resilience by storing election data across multiple nodes.
  - **Technology Stack:** The project utilizes the MERN stack, which includes:
    - *MongoDB:* For secure and scalable data storage, including user information, votes, and blockchain records.
    - *Express.js:* For creating a RESTful API that manages voting operations, voter registration, and election result management.
    - *React.js:* For building the frontend interface, providing users with a seamless voting experience.
    - *Node.js:* For handling server-side operations, including blockchain interaction and secure data handling.
  - **Methodology:** The platform integrates blockchain with the MERN stack to provide a secure, decentralized, and user-friendly voting system. The face detection feature enhances security by ensuring that only authorized users are allowed to vote, eliminating the possibility of fraudulent votes.
  - **Impact:** This system ensures secure, transparent, and accessible elections, offering a reliable and fraud-resistant voting process that can be conducted remotely. It enhances voter confidence and prevents election-related fraud.
  - **Technological Integration:** By integrating blockchain with face detection and the MERN stack, the system offers a modern, scalable, and secure solution to the challenges of traditional and existing e-voting systems.

## 7.2 Future Enhancements

The "Online Voting System using Blockchain" is a significant step in enhancing the security, transparency, and efficiency of the voting process. However, there are several potential future enhancements that can improve the system's functionality and address evolving challenges. Some of these enhancements include:

---

- **Integration with Biometric Authentication (Aadhaar Integration):**

- The current system uses face detection for real-time voter authentication. Future versions could integrate biometric authentication methods such as fingerprint, iris scanning, or Aadhaar card integration for voter verification. Aadhaar, being a widely used national identity system in India, could enhance the security and ease of voter identification.

- **Multi-layer Security Measures:**

- While blockchain ensures data immutability and tamper-proof features, adding additional layers of security, such as two-factor authentication (2FA) and end-to-end encryption, would further strengthen the system and prevent unauthorized access.

- **Support for Multiple Blockchain Platforms:**

- Future improvements could include support for various blockchain platforms like Ethereum, Hyperledger, or other emerging blockchain solutions. This would offer greater flexibility and scalability, depending on the election's size and requirements.

- **Improved User Experience (UX):**

- The platform's user interface could be enhanced to be more intuitive and user-friendly, including multilingual support and accessibility features, ensuring that voters of all ages and backgrounds can easily navigate the system.

- **Smart Contracts for Election Rules:**

- Incorporating smart contracts into the election process could automate aspects such as vote counting, eligibility checks, and result validation, minimizing human error and ensuring a fair election.

- **Blockchain Interoperability:**

- Enhancing the interoperability between different blockchain networks could streamline election data transfer across jurisdictions or multiple election bodies, facilitating large-scale election processes.

---

- **Integration with Government Databases:**

- Integrating the voting system with government databases (such as voter registries) could automatically verify voter eligibility, prevent fraud, and ensure only authorized individuals participate in the election.

- **Mobile Voting:**

- Enabling mobile voting would allow voters to securely cast their votes from anywhere, increasing accessibility and convenience, especially for people with disabilities or those living in remote areas.

These enhancements would help improve the system's reliability, security, and scalability, ensuring that it remains adaptable to the demands of future elections at various levels.

# References

- [1] Kaliyamurthie, K. P., et al. "Highly secured online voting system over network." Indian Journal of Science and Technology 6.6 (2013): 1-6
- [2] Jayson Falkner, Ben Galbraith, Romin Irani, Casey Kochmer, Sathya Narayana Panduranga, Krishnaraj
- [3] Sos.ca.gov. (2007). Top-to-Bottom Review — California Secretary of State. Available at: <http://www.sos.ca.gov/elections/voting-systems/oversight/top-bottom-review>. Supervisor: John Pearson, SIUC, USA.
- [4] Patidar, Kriti, and Swapnil Jain. "Decentralized e-voting portal using blockchain." 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, 2019
- [5] Hjálmarsson, Fririk ., et al. "Blockchain-based e-voting system." 2018 IEEE 11th international conference on cloud computing (CLOUD). IEEE, 2018.