Software Engineering and Project Management (CSE-306)

Bachelor of Technology In

Computer Science and Engineering

Title of the Project :

College Voting System

Submitted by

Settipalli Manoj kumar (AP22110010573)

Nandem Sreeram reddy (AP22110010552)

Reddem Gowthami reddy (AP22110010541)

Regalagadda Mohith Varun (AP22110010529)

of **SEC - L**

Under the Guidance of

(Singarapu Pavan Kumar)

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

Table of Contents:

Certificate

Date: 28-4-2025

This is to certify that the work present in this Project entitled "**College Voting System**" has been carried out by **Group-12** under my/our supervision. The work is genuine, original, and suitable for submission to the SRM University – AP for the award of Bachelor of Technology/Master of Technology in **School of Engineering and Sciences**.

Supervisor

(Signature)

Prof.**Singarapu Pavan Kumar**

Designation,

Affiliation.

Co-supervisor

(Signature)

Designation,

Affiliation.

2. Acknowledgements

We would like to convey our deepest gratitude to everyone who assisted us in the successful execution of our project entitled **"College Voting System"**.

Above all, we offer our warm thanks to our project mentor,**Mr. Singarapu Pavan Kumar**, for his valuable suggestions, support, and motivation throughout the project. His timely responses and expert guidance helped us overcome challenges efficiently.

We also acknowledge the staff and teachers of the **Department of Computer Science and Engineering, SRM University AP**, for providing us with the necessary resources and facilities to complete our work.

We thank our team members **Mohith,Sreeram,Manoj,Gowthami** for their coordination, cooperation, and commitment during the development of the project.

This project gave us hands-on experience in developing a real-time voting system for college elections, enhancing our practical knowledge in web development, database management, and system security — skills that will support our future work in computer science.

3. Abstract

The **College Voting System** project aims to digitalize and streamline the election process within a college environment. Traditional voting methods are often time-consuming, error-prone, and lack transparency. This system provides a secure, user-friendly online platform where students can participate in various club and general elections. The system allows administrators to create and manage elections, register candidates, and monitor voting activities. Students can log in, view ongoing elections, explore candidates' details, and cast their votes efficiently. Features such as result computation and real-time result viewing are integrated to ensure a transparent and smooth election process.Developed using full-stack technologies, the project ensures secure authentication, accurate vote counting, and data privacy. Through this project, we learned key concepts of system design, web development, database integration, and user interaction, preparing us for real-world application development.

List of Diagrams

SRS

1. Introduction

1.1 Purpose

The purpose of this document is to describe the requirements for the development of a College Voting System that allows students to register, nominate, and vote for candidates across different clubs within the college. The system ensures fair elections and maintains integrity, transparency, and ease of use.

1.2 Document Conventions

Actors are denoted by bold.

System components are denoted in italics.

Functional requirements are listed with [FR] tags.

Non-functional requirements are listed with [NFR] tags.

1.3 Intended Audience

College Students (Users)

Admins (Faculty/Staff managing elections)

Developers and Designers

College Management/Authorities

1.4 Additional Information

This system will be accessible via college intranet or secured login from any device (laptop/mobile).

1.5 Team Members

Member 1: Gowthami reddy- Frontend developer

Member 2:sreeram reddy- Frontend Developer

Member 3:manoj kumar-Backend Developer

Member 4:mohithvarun- Database Designer

1.6 References

College Election Guidelines Document

SRMAP clubs portal

Mern document

2. Overall Description

2.1 Product Perspective

This system is a new, independent system designed to automate the college election process, replacing manual voting.

2.2 Product Functions

Admin manages user and candidate registrations.

Students can vote securely for club representatives.

Election details, candidate lists, and voting results are displayed.

Only authenticated users can access voting features.

2.3 User Classes and Characteristics

Admin: Creates elections, registers candidates, manages results.

User (Student Voter): Registers, logs in, votes for candidates.

Candidate: A user who has been approved by the admin to contest.

2.4 Operating Environment

Web Browser (Chrome, Firefox, Safari)

Server: Node.js backend

Database: MongoDB

Hosting: College Server / Cloud Platform

2.5 User Environment

Desktop, Laptop, Tablet, Smartphone access.

Good internet connectivity within college premises.

2.6 Design/Implementation Constraints

Development using MERN Stack.

Security and authentication required.

Election voting once done, should be immutable.

2.7 Assumptions and Dependencies

Students will have a valid college ID to register.

Admin will manage the timing of elections properly.

System will be deployed on a secure server.

3. External Interface Requirements

3.1 User Interfaces

Login/Register Pages

Candidate Registration Page

Voting Page

Results Display Page

Admin Dashboard

3.2 Hardware Interfaces

Devices: PC, Mobile, Tablet

Servers for backend processing

3.3 Software Interfaces

Frontend: React.

Backend: Node.js, Express.js

Database: MongoDB

Authentication Middleware (JWT)

3.4 Communication Protocols and Interfaces

HTTPS for secure communication

REST APIs between frontend and backend (by postman)

4. System Features

4.1 User Registration and Authentication

4.1.1 Description and Priority

High priority. Users must register and authenticate before voting.

4.1.2 Action/Result

User enters credentials → System verifies → Login Success

4.1.3 Functional Requirements

[FR1] System must allow new users to register with student ID.

[FR2] System must verify login credentials.

4.2 Candidate Registration

4.2.1 Description and Priority

High priority. Admin registers candidates manually.

4.2.2 Action/Result

Admin fills candidate details → Candidate is listed in election.

4.2.3 Functional Requirements

[FR3] System must allow Admin to add candidates with position and club.

[FR4] System must prevent duplicate candidate entries.

4.3 Voting System

4.3.1 Description and Priority

Critical. Students cast votes for candidates only once.

4.3.2 Action/Result

User selects candidate → Vote is submitted → Confirmation shown.

4.3.3 Functional Requirements

[FR5] System must allow voting only once per user.

[FR6] System must record votes securely and immutably.

4.4 Result Publication

4.4.1 Description and Priority

Medium priority.

4.4.2 Action/Result

Admin publishes results → Users view winners.

4.4.3 Functional Requirements

[FR7] System must show vote counts after election ends.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

[NFR1] The system should support 500+ concurrent users.

[NFR2] Vote submission should take less than 2 seconds.

5.2 Safety Requirements

[NFR3] System backup should occur daily to avoid data loss.

5.3 Security Requirements

[NFR4] Passwords must be encrypted.

[NFR5] Votes must be securely recorded (No vote tampering).

5.4 Software Quality Attributes

Maintainability: Modular design for easy updates.

Usability: Simple and clean UI for students.

Portability: Accessible from various devices.

5.5 Project Documentation

Technical Documentation for maintenance.

User Manual for students and admin.

5.6 User Documentation

Guide for voting process.

Guide for admin operations.

6. Other Requirements

Data must be stored according to college privacy policies.

Election results must be preserved for audit purposes.

Glossary/Definitions

User: Student eligible to vote.

Admin: Faculty member managing the system.

Candidate: Student contesting the election.

Vote: Action of selecting a preferred candidate.

Club: An organizational unit for elections.

Appendix B: To Be Determined

College server deployment environment.

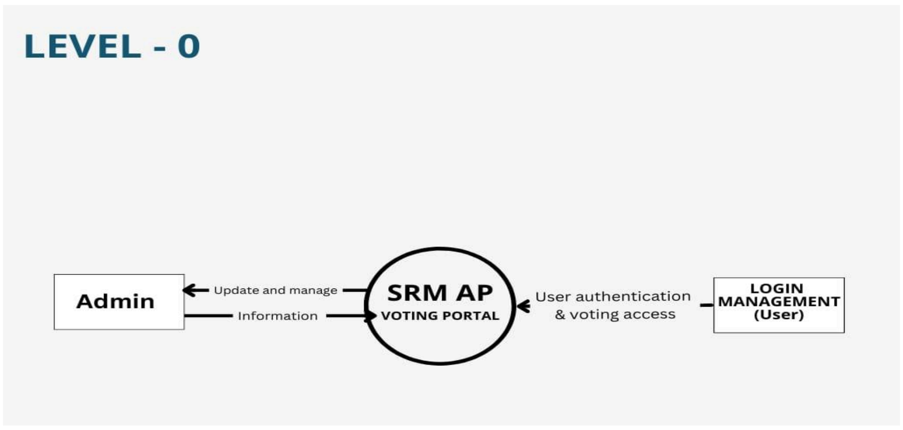
Election date-time settings.

2. Data Flow Diagram (DFD) for College Voting System

A Data Flow Diagram (DFD) represents the flow of data within a system. It helps in visualizing the processes, the data inputs and outputs, and the interactions between different entities in the system. Here's a breakdown of the DFD for the College Voting System built using the MERN stack:

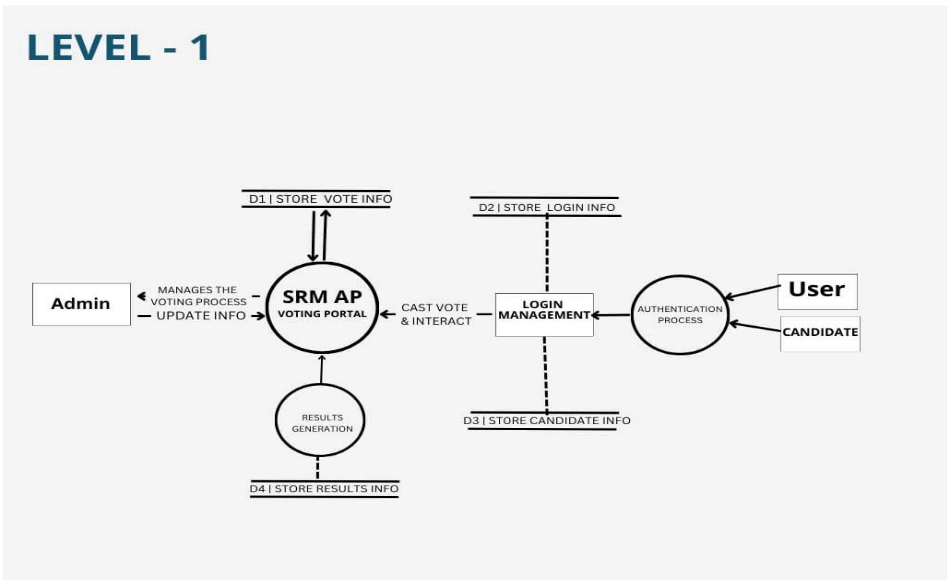
Level 0 DFD:

This is the highest-level DFD that shows the system as a single process and how it interacts with external entities.



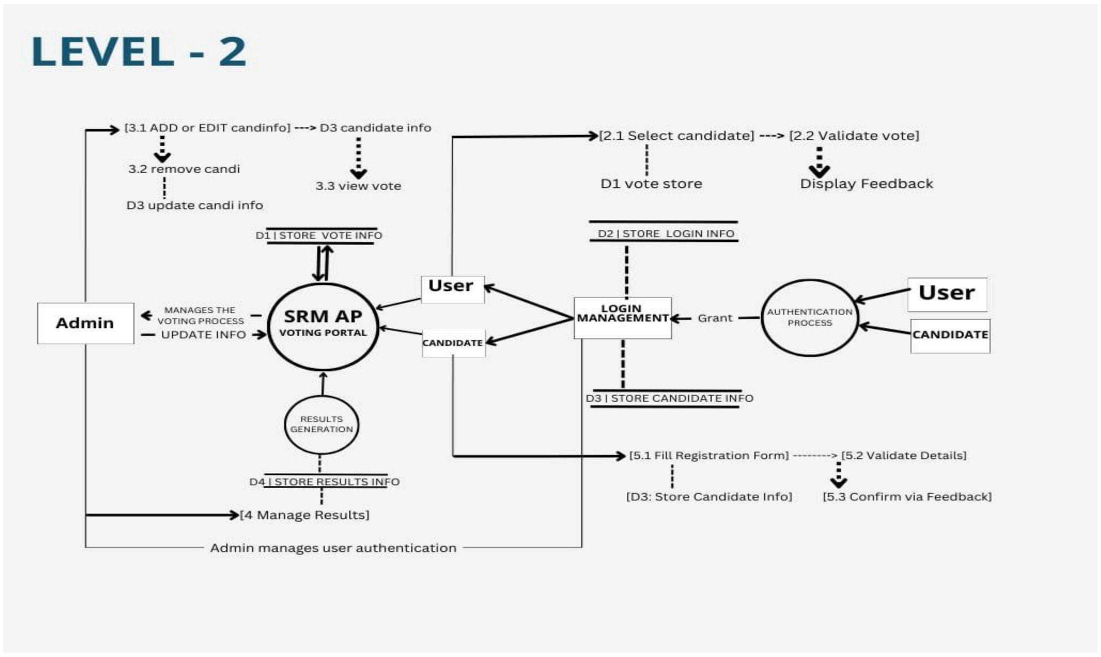
Level 1 DFD:

This diagram breaks the system into smaller processes for a more detailed view.



Level 2 DFD:

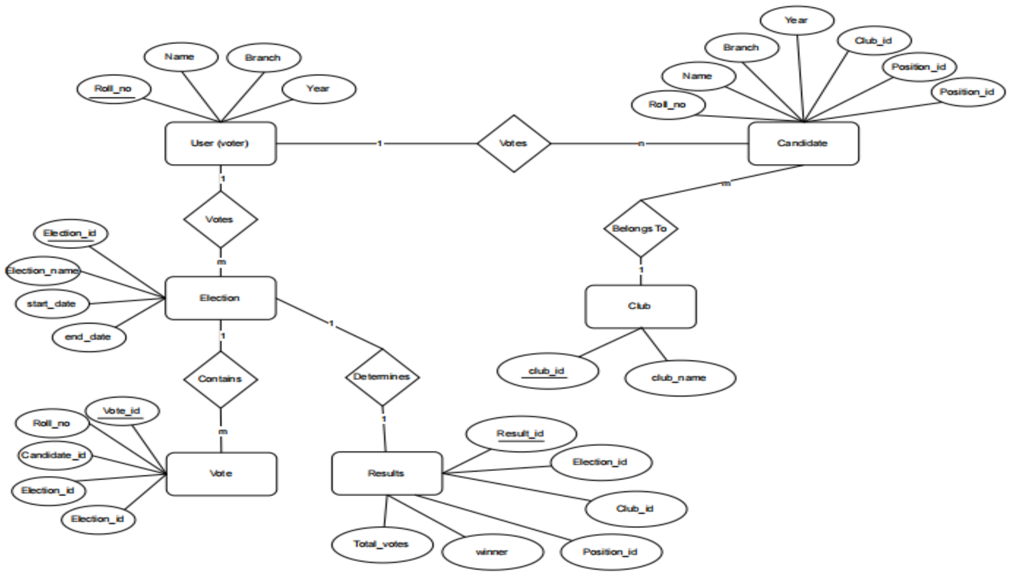
This diagram further breaks down the processes into more granular levels.



LEVEL - 2
[3.1 ADD or EDIT candinfo] ---> D3 candidate info
3.2 remove candi
D3 update candi info
3.3 view vote
[2.1 Select candidate] ---> [2.2 Validate vote]
D1 vote store
Display Feedback
D1 | STORE VOTE INFO
D2 | STORE LOGIN INFO
Admin
MANAGES THE VOTING PROCESS
UPDATE INFO
SRM AP
VOTING PORTAL
User
CANDIDATE
LOGIN MANAGEMENT
Grant
AUTHENTICATION PROCESS
User
CANDIDATE
D3 | STORE CANDIDATE INFO
RESULTS GENERATION
D4 | STORE RESULTS INFO
[4 Manage Results]
[5.1 Fill Registration Form] --------> [5.2 Validate Details]
[D3: Store Candidate Info]
[5.3 Confirm via Feedback]
Admin manages user authentication

3.ER DIAGRAM:

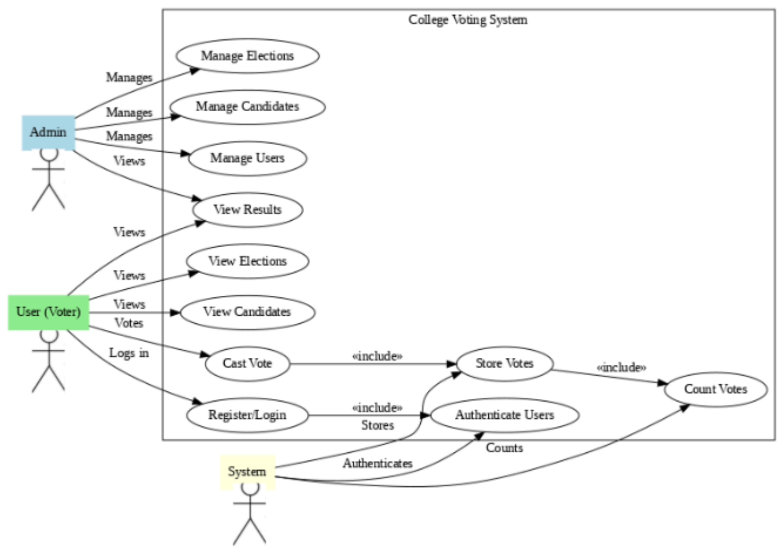
The Entity–Relationship (ER) diagram provides a high–level, conceptual view of the data model for the College Voting System. It identifies all the key entities, their attributes, and the relationships between them. This diagram serves as the blueprint for designing the database schema, ensuring that all data requirements are captured and that referential integrity is maintained.



4.USE CASE DIAGRAM:

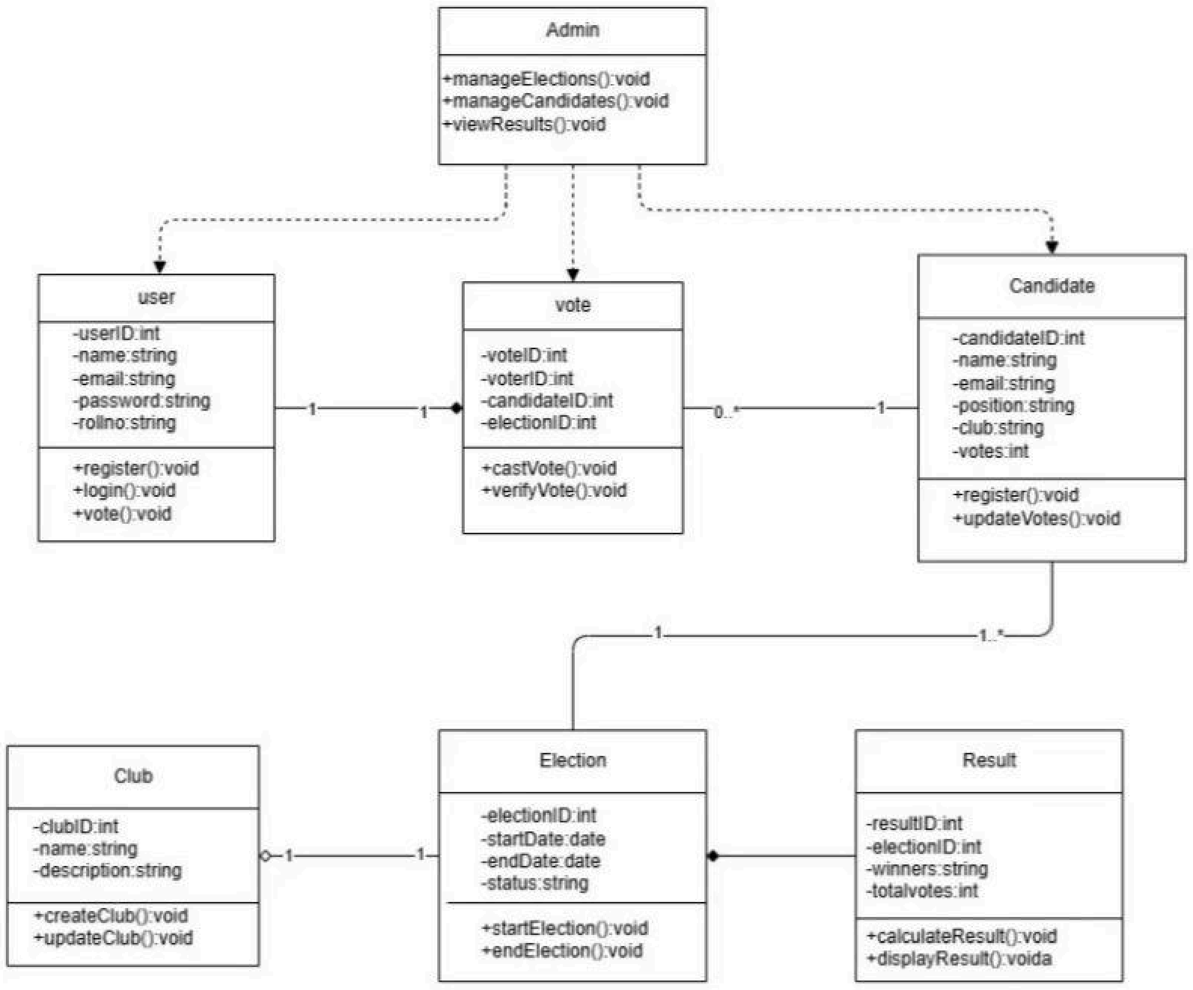
The use case diagram captures the functional requirements of the College Voting System by illustrating the interactions between external actors (students and administrators) and the system’s major functions. It provides a high–level view of who can do what, and serves as a guide for both design and testing.

Use Case Diagram for SRM AP Voting Portal Actors:



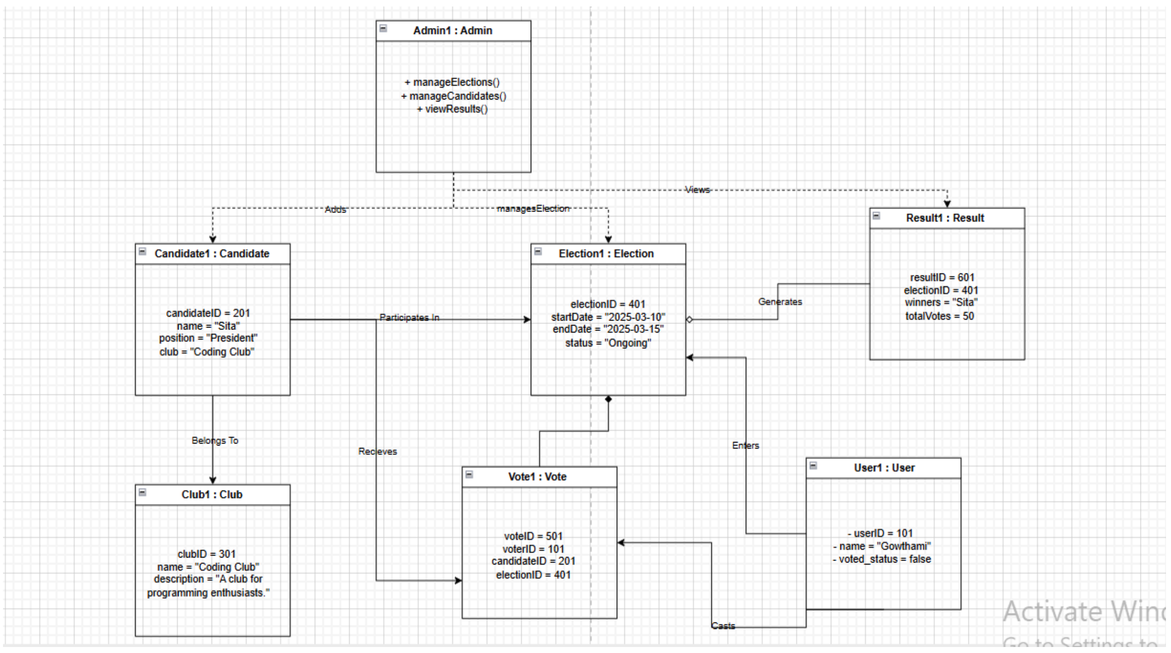
5. Class Diagram

A **Class Diagram** is a static structure diagram that shows the system's classes, their attributes, methods, and the relationships among objects. It is primarily used to represent the object-oriented design of the system. Each class is shown with its name, attributes, and methods. Relationships like inheritance, associations, and dependencies are depicted through various lines and arrows.



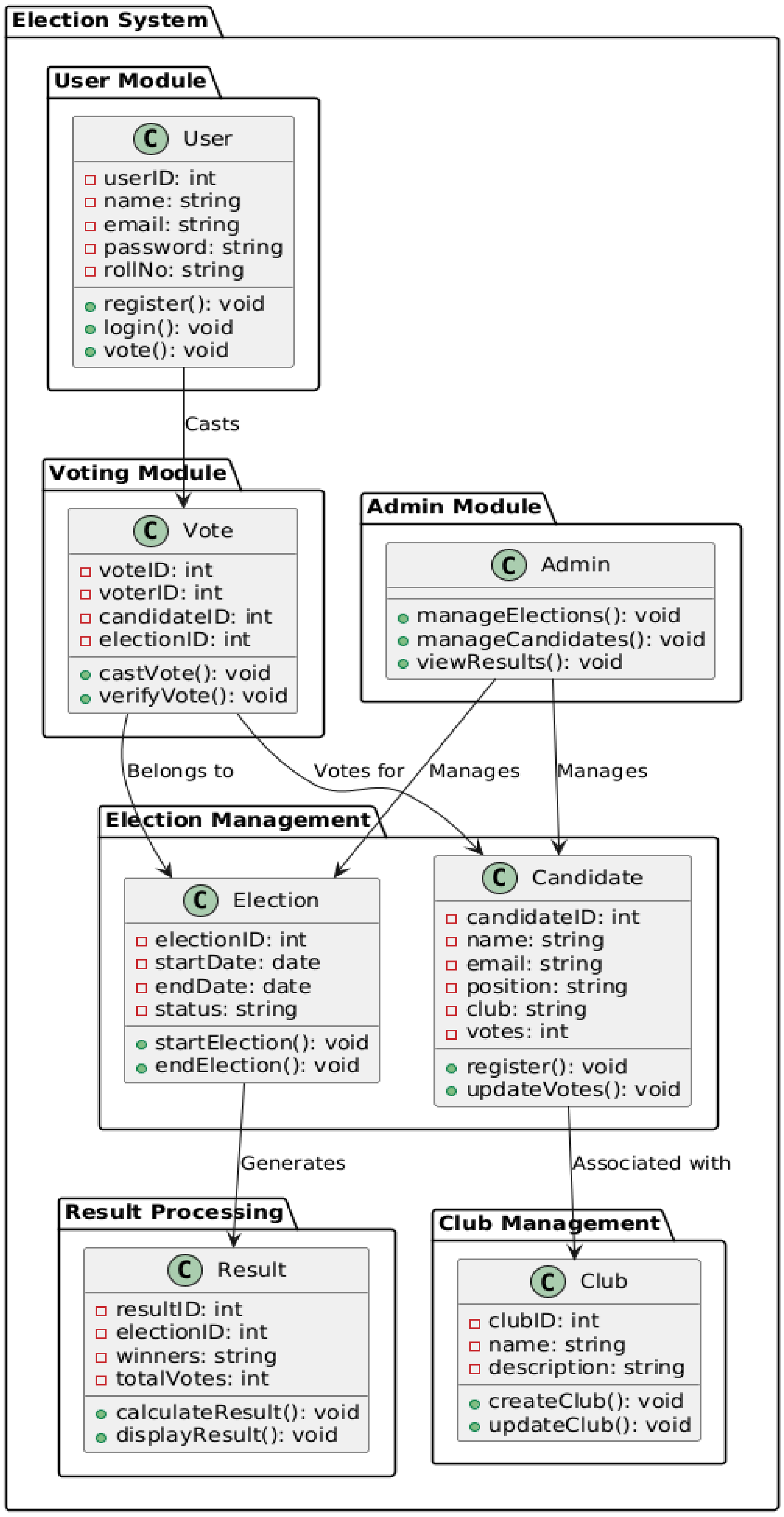
6. Object Diagram

An **Object Diagram** represents a snapshot of the objects in a system at a particular point in time. It shows instances of classes (objects), their attributes, and the relationships between them. This diagram is particularly useful to understand the state of the system at a given time.



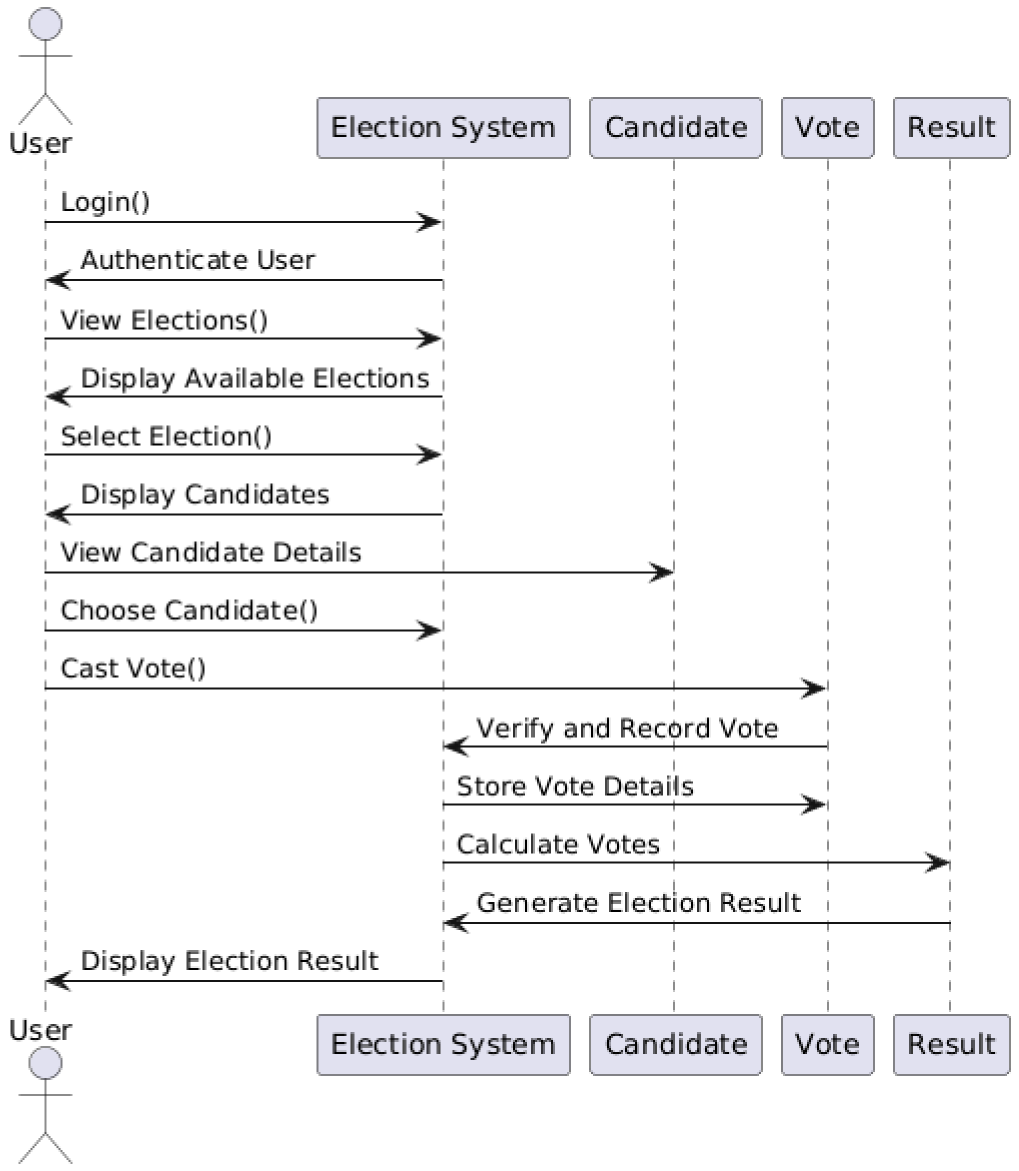
7. Package Diagram

A **Package Diagram** organizes the system into packages or modules, showing how different classes and components are grouped together to form logical units. This diagram emphasizes the organization and structure of the system, making it easier to manage large projects by breaking them down into smaller subsystems.



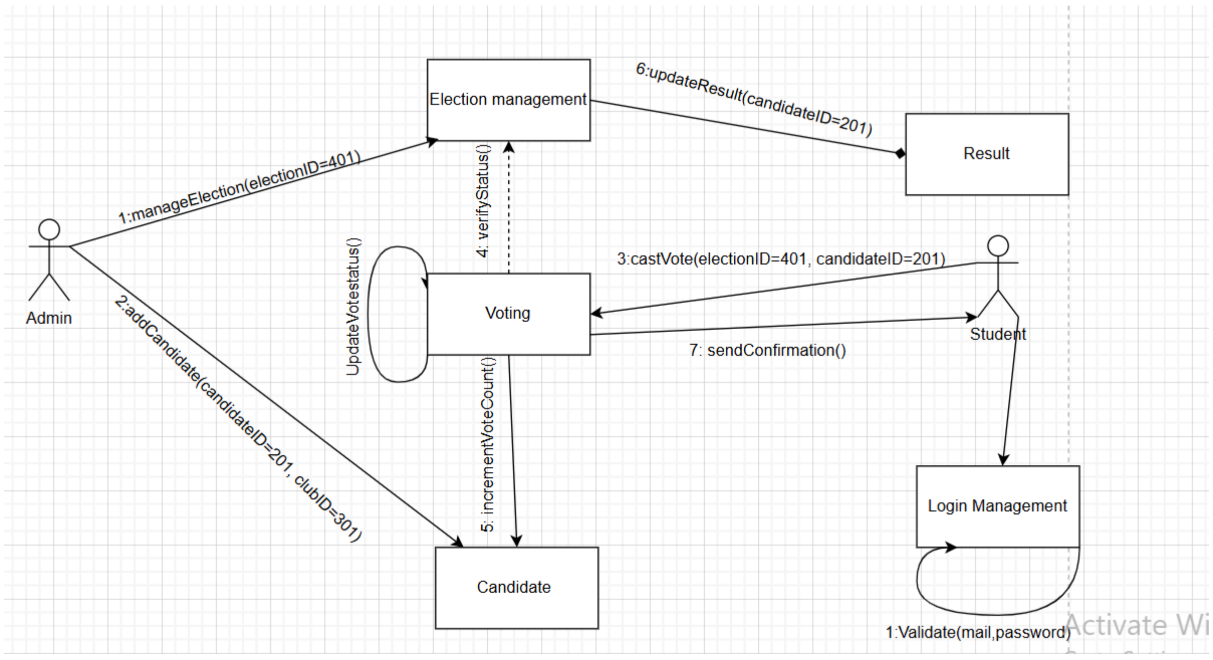
8. Sequence Diagram

A **Sequence Diagram** shows how objects interact in a particular sequence, detailing the order of method calls, messages, and responses. It emphasizes the time aspect of object interactions, representing the flow of messages between objects over time.



9. Collaboration Diagram

A **Collaboration Diagram** (also known as a communication diagram) shows the interactions between objects in a system, with an emphasis on the relationships between them. It is similar to a sequence diagram but focuses more on the structural aspect (which objects communicate with each other) rather than the time sequence.


Election management
6:updateResult(candidateID=201)
Result
1:manageElection(electionID=401)
4: verifyStatus()
UpdateVotestatus()
3:castVote(electionID=401, candidateID=201)
Admin
Voting
Student
2:addCandidate(candidateID=201, clubID=301)
7: sendConfirmation()
5: incrementVoteCount()
Login Management
Candidate
1:Validate(mail,password)
Activate Wi

1. Introduction

1.1 College Voting System

1.1.1 Importance of Voting Systems in Educational Institutions

Voting plays a crucial role in promoting democratic values within colleges and universities. It empowers students to choose their representatives and fosters leadership qualities. A structured voting system ensures transparency, fairness, and encourages maximum participation among the student community.

1.1.2 Traditional vs Digital Voting Systems

Traditionally, college elections have relied on paper ballots and manual counting, often leading to errors, mismanagement, and delayed results. A digital voting system, by contrast, automates the election process, eliminates manual errors, speeds up result processing, and ensures enhanced security and trust among voters.

1.1.3 Challenges in Manual Voting

Manual voting systems face issues like logistical challenges, ballot tampering, lack of anonymity, and time-consuming result calculation. Human errors and biased practices can affect the credibility of the elections. Moreover, physical arrangements for voting booths and ballot storage add to operational costs.

1.1.4 Benefits of Digital Voting Systems

Online voting systems bring in automation, transparency, and accuracy. Students can vote securely from their devices, results can be declared instantly, and administrative overhead is significantly reduced. The system also ensures a tamper-proof voting experience with traceable yet confidential records.

1.2 Web Technology in Voting Systems

1.2.1 Role of Web Applications in Elections

The rise of web technologies has enabled institutions to develop platforms that allow remote, real-time participation in elections. Web applications offer accessibility, ease of use, cross-platform support, and better scalability, making them ideal for handling college elections seamlessly.

1.2.2 MERN Stack Overview

MERN stands for MongoDB, Express.js, React.js, and Node.js.

- **MongoDB**: NoSQL database to store user details, candidate information, voting records, and results.

- **Express.js**: A backend web application framework for managing server-side logic and API endpoints.

- **React.js**: A frontend JavaScript library that creates dynamic and responsive user interfaces for students and administrators.

- **Node.js**: A runtime environment that runs JavaScript on the server and handles client-server communication effectively.

1.2.3 Advantages of Using MERN Stack for Voting Systems

- Seamless integration between frontend and backend.

- High scalability to handle a large number of users.

- Real-time updates and instant result display.

- Secure handling of voting and authentication processes.

- Open-source technologies, reducing cost.

1.3 System Modules and Features

1.3.1 Student Authentication and Voting

Students must register/login into the system. After successful authentication, they can view available elections for clubs and cast their vote securely. Voting is restricted to one vote per election per student to maintain integrity.

1.3.2 Candidate Registration and Management

Authorized users (admin) can add candidate details, assign them to respective clubs, sub-clubs, and election positions. Candidate information like name, photo, description, and election manifesto can be uploaded and displayed.

1.3.3 Election Management by Admin

Admins can create new elections, open/close voting periods, manage candidate registrations, and oversee the entire election lifecycle. Admins also have access to view and declare results after the election ends.

1.3.4 Result Display

Results are calculated automatically based on the votes cast. The system displays live or final results in a graphical and tabular format, ensuring transparency in the election process.

1.4 Project Objective and Scope

1.4.1 Objective of the System

The main goal is to design and develop a secure, web-based college voting system that ensures transparency, quick processing, and easy access for both students and administrators.

1.4.2 Scope and Target Users

The system is intended for use within educational institutions such as colleges and universities.
Target users include:

- Students (as voters)
- Candidates (contestants for various positions)
- Admins (election organizers and system managers)

1.5 Limitations of the Current System

1.5.1 Need for Internet Connectivity

Since it is a web-based application, a stable internet connection is necessary for students and admins to access and use the platform.

1.5.2 Limited to College-Level Elections

The system is tailored for small to medium-scale college elections. Large-scale public elections would require additional security measures and load management optimizations.

2.Methodology

The development of the College Voting System followed a structured and modular approach, involving both frontend and backend technologies.

2.1 System Overview

The system is designed as a full-stack web application:

- **Frontend**: Built using **React.js**, providing a responsive and interactive user interface for students and admins.
- **Backend**: Developed using **Node.js** and **Express.js**, handling API requests, authentication, and database operations.
- **Database**: **MongoDB** is used for storing user details, candidate information, election data, voting records, and results securely.

2.2 User Roles

- **Admin**:
  - Manage elections (create, edit, delete)
  - Register candidates
  - View and download results
- **Student**:
  - Register/Login
  - View available elections, candidates, and their profiles
  - Cast their votes (one vote per position)

2.3 Frontend Development

- **React.js** is used to build a dynamic and responsive user interface.

- Key pages include:
    - Login/Register Page
    - Candidate Registration Page
    - Voting Page (positions categorized under clubs and sub-clubs)
    - Results Page
- **State management** is handled using React Hooks and Context API.

2.4 Backend Development

- **Node.js** with **Express.js** provides RESTful API services for:
    - User Authentication (JWT-based tokens)
    - Candidate registration and listing
    - Voting functionality with vote validations
    - Result calculation and retrieval
- **APIs** ensure communication between frontend and backend in a secure and efficient manner.

2.5 Database Structure

The **MongoDB** database consists of several collections:

- **Users**: Student registration/login information
- **Candidates**: Candidate details including club, sub-club, position, and manifesto
- **Votes**: Records each student's vote ensuring one vote per position
- **Results**: Stores and calculates voting results dynamically

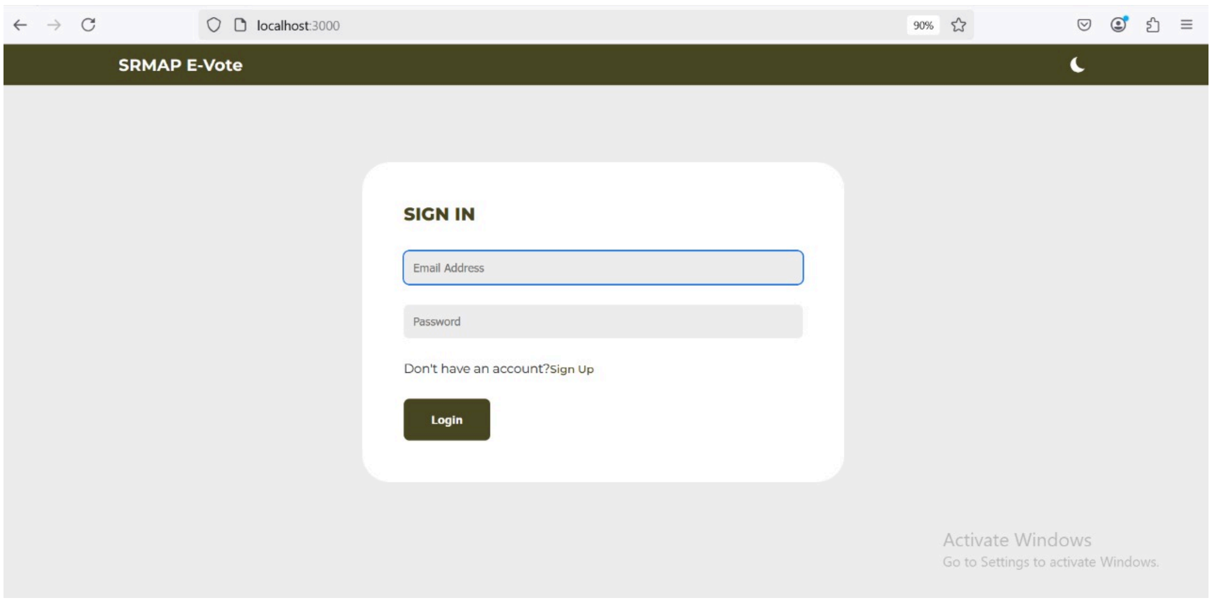
2.6 Security Measures

- **Authentication**: JWT (JSON Web Token) is used for secure login sessions.
- **Authorization**: Admin-specific routes are protected to prevent unauthorized access.
- **Vote Integrity**: Each student can vote only once for each position; multiple voting is prevented at the database level.
- **Data Validation**: Input validation is performed on both frontend and backend.

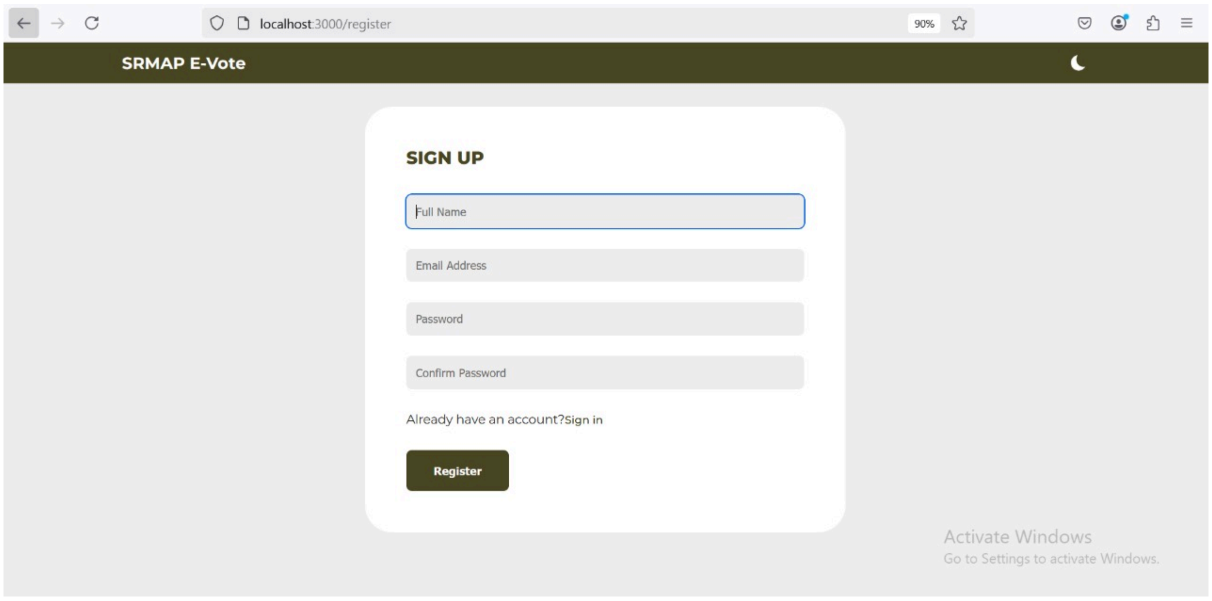
2.7 Deployment

- The application is designed to be hosted on platforms like **Render**, **Vercel**, or **AWS**, ensuring easy accessibility for students and admins through any device.

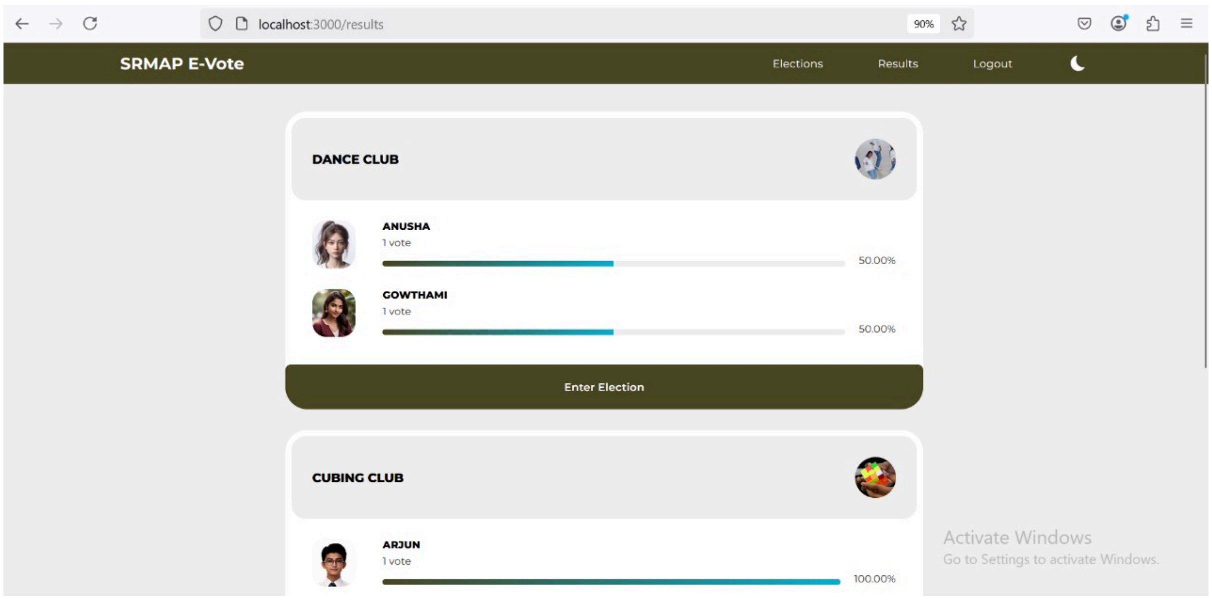
3.Results



localhost:3000
90%
SRMAP E-Vote
SIGN IN
Email Address
Password
Don't have an account?Sign Up
Login
Activate Windows
Go to Settings to activate Windows.



localhost:3000/register
90%
SRMAP E-Vote
SIGN UP
Full Name
Email Address
Password
Confirm Password
Already have an account?Sign in
Register
Activate Windows
Go to Settings to activate Windows.



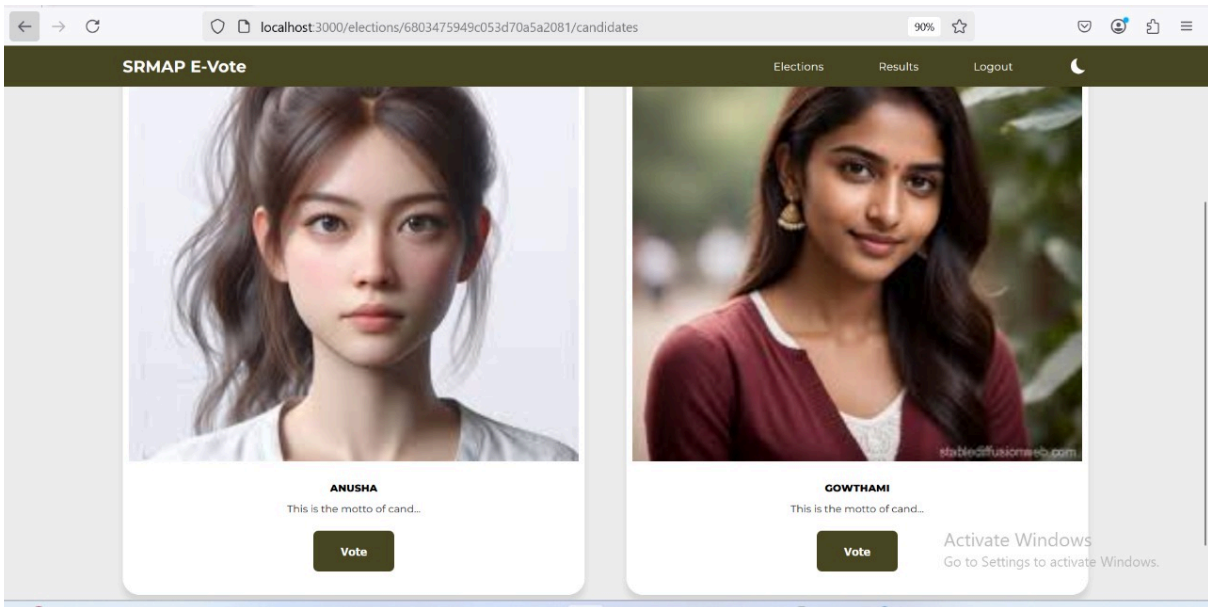
DANCE CLUB

ANUSHA
1 vote
50.00%

GOWTHAMI
1 vote
50.00%

CUBING CLUB

ARJUN
1 vote
100.00%



localhost:3000/elections/6803475949c053d70a5a2081/candidates
90%
SRMAP E-Vote
Elections
Results
Logout
Vote
Vote
Activate Windows
Go to Settings to activate Windows.

ANUSHA
This is the motto of cand...

GOWTHAMI
This is the motto of cand...

3.Concluding Remarks

3.1.1 Summary of the Project

The College Voting System developed using the MERN stack provides a modern, secure, and efficient way to conduct elections in educational institutions. It successfully overcomes the challenges of manual voting by ensuring accuracy, transparency, and ease of participation for students and administrators alike.

3.1.2 Achievement of Objectives

The project fulfilled all its initial objectives:

- A fully functional online platform for voting was developed.
- User authentication, candidate management, and result computation were successfully implemented.
- The system ensures that each user can vote only once per election, preserving the integrity of the voting process.

3.1.3 Benefits and Impact

The digital voting system streamlines election processes, significantly reduces administrative overhead, and increases student engagement by offering an accessible and user-friendly platform. It promotes democratic practices within the institution and builds trust among participants.

3.2 Challenges Faced

3.2.1 Technical Challenges

- Managing real-time data updates and ensuring synchronization between frontend and backend services.
- Implementing a secure authentication system to prevent unauthorized access and voting fraud.

- Designing a responsive frontend to cater to devices with different screen sizes and operating systems.

3.2.2 Project Management Challenges

- Coordinating different modules like login, registration, candidate management, and result display within tight timelines.
- Testing the application thoroughly to ensure no bugs, security loopholes, or voting irregularities exist.

3.3 Future Enhancements

3.3.1 Enhancing Security Measures

- Implementing Two-Factor Authentication (2FA) for users during login.
- Encrypting votes using advanced cryptographic techniques to further ensure privacy and data protection.

3.3.2 Introducing Notifications and Reminders

- Adding SMS or Email notifications to remind students about election dates and results announcement.

3.3.3 Expanding to Mobile Platforms

- Developing a dedicated mobile application to allow voting through smartphones, thus increasing accessibility and participation.

3.3.4 Analytics and Reporting Features

- Providing detailed analytics reports on voter turnout, candidate votes, and election trends for administrative review.

4.Future Work

4.1 Scope for Future Enhancements

As the current version of the College Voting System successfully delivers the core functionalities, there is significant potential for further expansion and improvement. Future development can focus on introducing advanced features to make the system even more secure, scalable, and user-friendly.

4.2 Proposed Future Improvements

4.2.1 Biometric Authentication

Integrating biometric authentication such as fingerprint or facial recognition can add an additional security layer to prevent identity fraud during the voting process.

4.2.2 Blockchain-based Voting

Implementing blockchain technology would ensure a tamper-proof and transparent voting process. Each vote can be recorded securely on a decentralized ledger, making the election process immutable and trustworthy.

4.2.3 Real-Time Voting Analytics Dashboard

Developing an admin dashboard that displays real-time voting statistics like voter turnout, active users, and live vote counts (without revealing results) can help administrators monitor the election efficiently.

4.2.4 Offline Voting Mode

For colleges in areas with limited internet access, an offline voting option (with later synchronization) could be explored to ensure wider participation.

4.2.5 Role-Based Access Control (RBAC)

Introducing RBAC would allow differentiated access rights for various users such as Admins, Club Coordinators, Sub-Club Heads, and Students, improving management and security.

4.2.6 Result Certification and Archiving

Automated generation of election certificates for candidates and a digital archive of past elections can be added for maintaining institutional records.

4.3 Research Possibilities

Future research can focus on:

- Studying voter behavior and participation trends through AI-based data analysis.

- Exploring machine learning models to predict voter turnout and enhance election planning.

- Investigating user experience improvements using surveys and feedback mechanisms.

References

1. **MongoDB Official Documentation**
   https://www.mongodb.com/docs/

2. **Express.js Official Website**
   https://expressjs.com/

3. **React.js Official Documentation**
   https://react.dev/

4. **Node.js Official Documentation**
   https://nodejs.org/en/docs

5. **JWT (JSON Web Tokens) Authentication**
   https://jwt.io/introduction/

6. **Mongoose ODM Documentation**
   https://mongoosejs.com/docs/

7. **Bcrypt.js for Password Hashing**
   https://www.npmjs.com/package/bcrypt

8. **Material-UI (MUI) for Frontend Styling**
   https://mui.com/

9. **CORS Middleware for Node.js**
   https://www.npmjs.com/package/cors

10. **Axios - Promise based HTTP client for the browser and node.js**
    https://axios-http.com/