



International Journal of Advanced Research in Education and TechnologY (IJARETY)

Volume 12, Issue 3, May-June 2025

Impact Factor: 8.152



Online Voting System

Pradeepa S, Dr. sheela k

UG student, Department of Computer Science and Information Technology, VISTAS, Chennai, India

Assistant professor, Department of Computer Science and Information Technology, VISTAS, Chennai, India

ABSTARCT: The Online Voting System for College Elections is a secure and efficient digital platform designed exclusively for college students to elect the Secretaries of multiple streams. This system ensures that only verified students can access the voting platform using secure login credentials, preventing unauthorized participation. It replaces traditional paper-based voting with a fast, and automated process, making elections more reliable and accessible.. Additionally, automatic vote counting eliminates manual errors and speeds up result declaration. With a user-friendly interface, students can vote easily from any device, whether on campus or remotely, increasing participation with unique login credentials. Fraud prevention measures ensure that each student can vote only once, maintaining election integrity. Administrators have access to a dashboard to manage candidates, monitor voting progress, and publish results efficiently. By implementing this Online Voting System, the college can ensure a fair, secure, and transparent election process while reducing administrative workload and enhancing student engagement.

I. INTRODUCTION

The Online Voting System is a comprehensive web-based application designed to digitize and streamline the election process within a college environment. The primary purpose of this project is to provide a secure, transparent, and efficient platform that allows students to participate in elections without the need for physical voting infrastructure. Through this system, students can easily register by entering their name, registration number within a specific valid range, and a password. Once registered, users can log in securely, nominate themselves for leadership positions such as Secretary, Assistant Secretary, or Cultural Secretary, and view a dynamic list of all candidates, which includes both default and newly nominated individuals. The system ensures a fair voting process where each user can cast a vote for their preferred candidates and receive an immediate confirmation upon successful submission, after which they are automatically logged out to maintain vote integrity. Additionally, students can view the election results in a user-friendly format once the voting is complete. For administrative purposes, the system includes a separate admin login that grants access to a detailed results page, displaying vote counts for all candidates. The entire project is built using HTML for page structure, CSS for design and layout, and JavaScript for client-side functionality and dynamic behavior. All user data, candidate nominations, and voting records are stored using JSON, ensuring lightweight and easily manageable data handling. The project was developed and tested using Visual Studio Code, making it a practical solution for conducting online elections in an educational setting while promoting digital engagement, transparency, and accuracy.

II. IMPLEMENTATION DETAILS

2.1 HTML Pages Description:

Each HTML page in your project serves a specific function. Here's a breakdown:

index.html (Login Page):

Purpose: Allows users to log in using their registration number and password.

Components:

- Input fields for Reg No and Password.
- Login button.
- Link to the registration page.
- JavaScript checks credentials on login.

register.html (Register Page):

Purpose: New users can register into the system.

Components:

Input fields: Name, Reg No (must be between 22106100 to 22106250), Password.
Validation for Reg No range and uniqueness.
Stores user info into users.json.

nomination.html (Nomination Page):

Purpose: Allows users to nominate themselves for one of the three posts.

Components:

Dropdown or radio buttons to select a position.
Submit button to save nomination.
Checks if the user has already nominated.

candidates.html (Candidate List Page):

Purpose: Shows a list of all candidates (default + nominated).

Components:

Displays candidates grouped by positions: Secretary, Assistant Secretary, Cultural Secretary.
Dynamically loads data from candidates.json.

voting.html (Voting Page):

Purpose: Allows users to vote once for each position.

Components:

Candidate lists with radio buttons.
Submit button to cast vote.
On submission, records vote in votes.json and redirects to success page.

success.html (Vote Successful Page):

Purpose: Displays a confirmation that the vote was successful.

Components:

A simple success message.
Option to return to login or exit.

results.html (Result Page):

Purpose: Displays the public results after voting.

Components:

Shows candidates with their respective vote counts for each position.
Data pulled from votes.json.

2.2 CSS Styling:

The main CSS file is css/style.css, which is linked to all pages.

Key Styling Elements:

- Responsive Layout: Uses flexbox or grid for clean layout on all screen sizes.
- Form Styling: Input boxes, buttons, and labels are styled for consistency.
- Buttons: Hover effects and disabled states for better UX.
- Colors & Fonts: A consistent color scheme (e.g., blue/white) and readable fonts like Arial or Roboto.
- Cards/Sections: Border-radius, shadows, and padding for modern UI look.
- Error/Success Alerts: Colored messages for feedback.

2.3 JavaScript Functionality:

Located in the js/ folder. Each file handles separate logic:

auth.js:

- **Handles:**

User login validation.

Admin login check.

Registration validation (Reg No range, password strength).

Storing and retrieving users from users.json.

candidates.js:

- **Handles:**

- Display of candidates on candidates.html.

- Adding a new candidate on nomination.

- Preventing duplicate nominations.

- Grouping candidates by role (Secretary, etc.).

voting.js:

- **Handles:**

Voting logic: storing selected candidate per post.

Prevents multiple votes from the same user.

Updates votes.json and redirects to success page.

Disables form post-vote.

results.js:

- **Handles:**

Counting votes per candidate from votes.json.

Sorting and displaying top candidates.

Both user and admin views (admin view may include voter info optionally).

2.4 JSON Data Storage:

JSON files in the data/ folder are used to simulate backend storage.

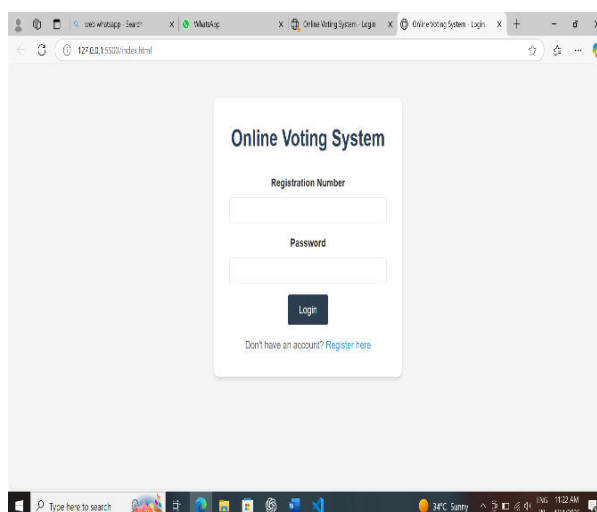
Usage:

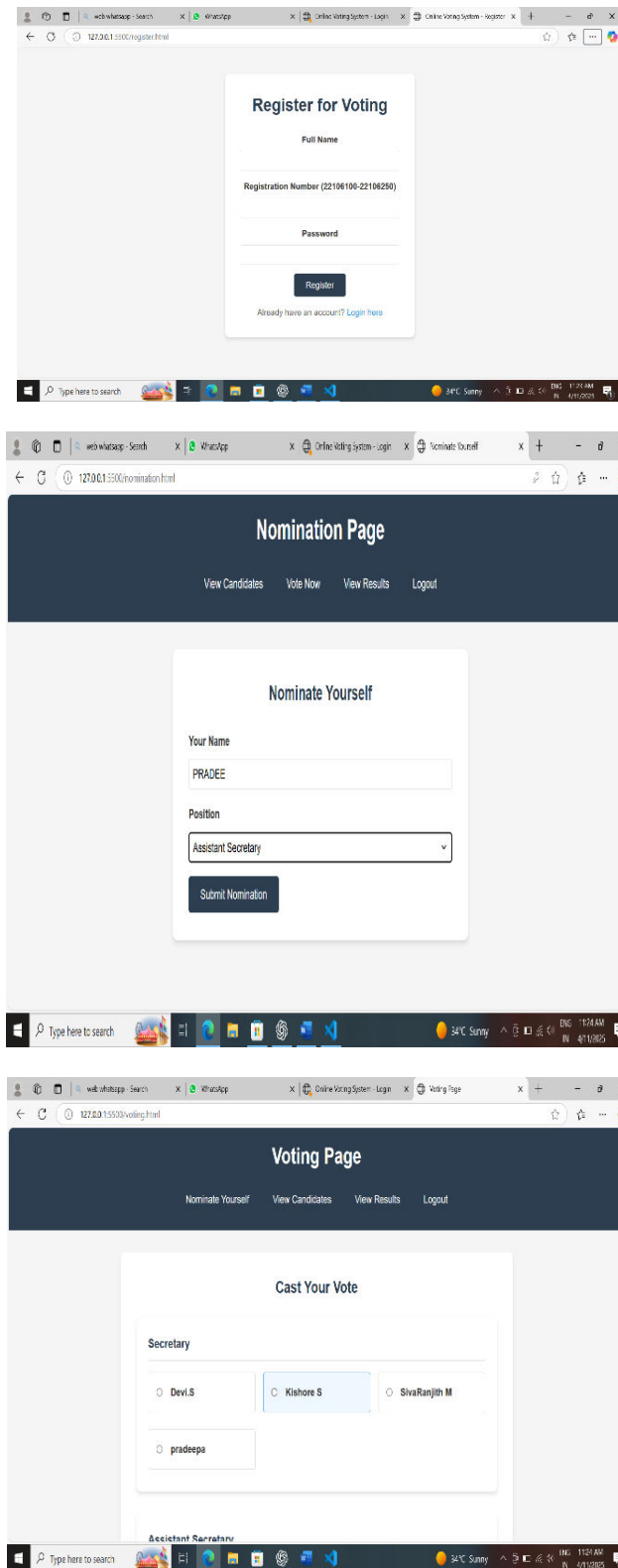
Stores votes per user (1 per position).

Used to calculate results.

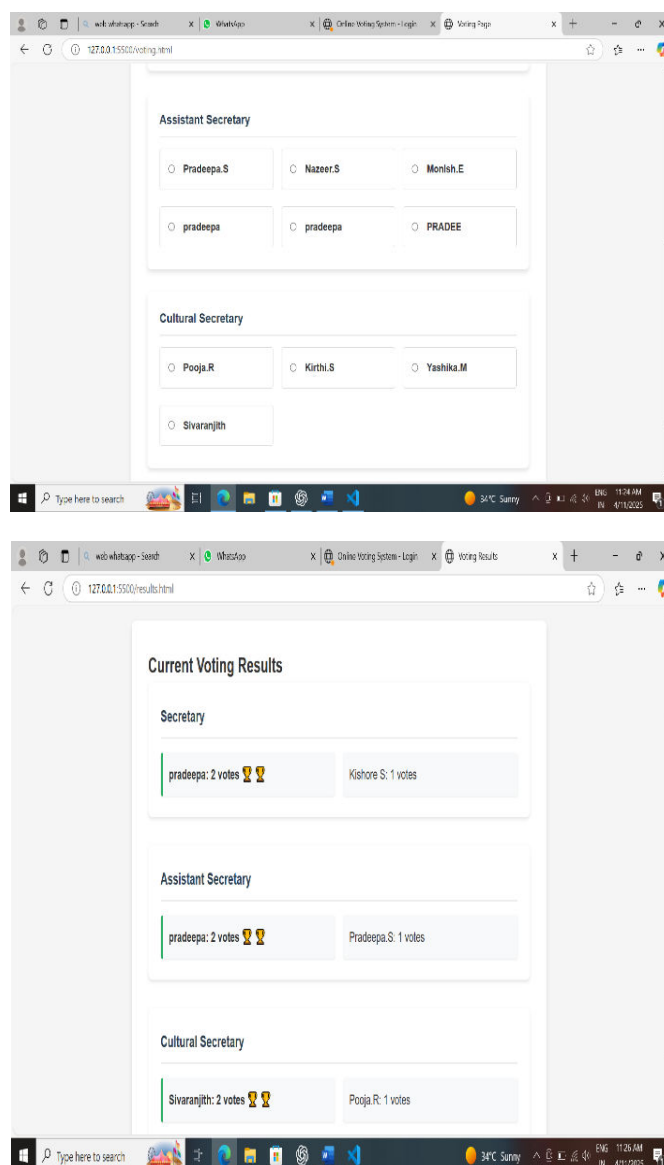
Admin can see voter records; public result page shows only totals.

III. RESULTS





The image displays three sequential screenshots of a web application interface, likely for a voting system, shown within a browser window. The browser's address bar indicates the URL is 127.0.0.1:5500/register.html for the first screenshot, 127.0.0.1:5500/nomination.html for the second, and 127.0.0.1:5500/voting.html for the third. The first screenshot, titled "Register for Voting", shows a registration form with fields for "Full Name", "Registration Number (22106106-22106250)", and "Password", along with a "Register" button and a link to "Login here". The second screenshot, titled "Nomination Page", features a "Nominate Yourself" form with fields for "Your Name" (containing "PRADEE") and "Position" (a dropdown menu set to "Assistant Secretary"), and a "Submit Nomination" button. The third screenshot, titled "Voting Page", shows a "Cast Your Vote" form with a "Secretary" section containing three radio button options: "Devi S", "Kishore S" (which is selected), and "SivaRanjith M", and a "pradeepa" option below. The browser's taskbar at the bottom of each screenshot shows the system clock as 11:24 AM on 4/11/2025.



IV. CONCLUSION AND FUTURE ENHANCEMENTS

The Online Voting System project effectively simulates a complete election process using only front-end technologies like HTML, CSS, and JavaScript, with JSON files serving as the data storage layer. It includes key features such as user registration with validation, secure login, candidate nomination for three positions, dynamic candidate display, one-time voting functionality, and real-time result updates. An admin-only results page ensures restricted access for vote monitoring. The user interface is designed to be simple and intuitive, making the system easy to navigate. While the project succeeds in replicating the core elements of digital voting, it operates entirely on the client side, making it vulnerable to data manipulation and limiting its scalability. The absence of a backend also means data is not persistently or securely stored. Despite these limitations, the system is ideal for classroom demonstrations or small-scale elections. It provides a solid foundation for understanding the voting process and could be further enhanced with server-side integration. Overall, it's a well-rounded educational project with practical relevance. The current Online Voting System effectively meets its basic goals using a frontend-only setup, but several enhancements could significantly elevate its functionality, security, and scalability. Integrating a backend with technologies like Node.js and databases such as MongoDB or Firebase would allow secure, persistent data handling across devices. Implementing features like encrypted password storage and authentication systems would strengthen user security. A more advanced admin panel with live vote tracking and activity logs could offer better control and transparency. To improve accessibility, adopting responsive design through frameworks like Bootstrap or Tailwind CSS would ensure a seamless experience on all

devices. Adding email or SMS verification during registration can prevent duplicate or fake entries. Enhancing privacy through encrypted vote submissions would uphold voter anonymity. Hosting the system online via platforms like Netlify or AWS would enable real-world deployment and accessibility. These upgrades would transition the project from a demonstration tool to a reliable digital voting solution.

REFERNCES:

1. Mozilla Developer Network (MDN Web Docs) – HTML, CSS, and JavaScript Documentation <https://developer.mozilla.org/>
2. W3Schools – Web Development Tutorials and Examples <https://www.w3schools.com/>
3. GeeksforGeeks – JavaScript and Web Development Articles <https://www.geeksforgeeks.org/>
4. StackOverflow – Community Discussions and Troubleshooting <https://stackoverflow.com/>
5. FreeCodeCamp – JavaScript Projects and Frontend Development Learning <https://www.freecodecamp.org/>
6. JSON.org – JavaScript Object Notation (JSON) Syntax and Usage Guide <https://www.json.org/>
7. CSS-Tricks – CSS Layouts, Styling Tips, and Best Practices <https://css-tricks.com/>
8. Visual Studio Code Documentation – Editor Setup and Usage <https://code.visualstudio.com/docs>
9. GitHub – Sample Projects and Code Snippets <https://github.com/>
10. YouTube Tutorials – Online Voting System in JavaScript / Frontend Projects
e.g., "Build an Online Voting System using HTML, CSS, and JS"

International Journal of Advanced Research in Education and Technology

ISSN: 2394-2975

Impact Factor: 8.152