WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

A Modern Web Application for Student Council Elections: Design and Development of an Online Voting System.

[1] Kammili Deepika, [2.] Mr. B. Panna Lal

1P. G. Research Scholar, Dept. of MCA, Aurora Deemed To-Be University, Hyderabad, Telangana, India.
2Assistant Professor, Dept. of CSE, Aurora Deemed To-Be University, Hyderabad, Telangana, India.
Email: [1]deepikakammilid23@gmail.com [2]pannalal@aurora.edu.in

ABSTRACT

The Student Online Voting System is a web app designed to manage student council elections at schools. It's built to be efficient, transparent and secure. On the side it uses React for the frontend Node, with Express for the backend and MongoDB Compass to store data. Essentially there are two types of users. Admins and students. Each with their set of responsibilities that help streamline the election process. When you land on the homepage you'll see a layout with sections, like "Why Vote Online?" which explains the benefits "Candidate Spotlight" that highlights the contenders and a real time feed of "Voting Statistics" to keep everyone updated. Election management becomes a lot easier, with a system. For starters administrators have the ability to set up elections manage the candidates and then take a look at the results in a graphical format. On the side students benefit from being able to register get updates on elections and cast their votes over the internet in a secure manner. What all this adds up to is participation from voters thanks to the system being easy to navigate and providing updates. By switching from paper-based voting, to platforms we can make elections more trustworthy and get students more invested in the process.

Keywords: Online Voting System, Web Application, Student Elections, React.js, Node.js, MongoDB

Introduction

Student elections, in colleges and universities typically involve a lot of paperwork and manual processes. From signing up voters to managing candidates and counting votes it can be a headache. This traditional way of doing things not Soaks up a lot of time. Also leaves room, for mistakes and a lack of openness. As a result, the whole election process can start to look a bit dubious and inefficient. That's where the Student Online Voting System comes in. A web based application that aims to make the entire process simpler and more modern.

The system is designed to tackle these challenges head offering a platform that's both secure and easy to use for everyone involved. This means administrators and students can take part in elections online without a hitch. Built with React.js for the frontend and Node.js along, with Express.js for the backend the application also uses MongoDB Compass to keep all the data in check. Its set up to handle two types of users; those in charge the admins and the students themselves. The admins are, in charge of creating elections deciding who can run and keeping an eye on how things are going. They can even see the results as they come in. From a computer, phone or tablet. Long as it's connected to the internet. Students can sign up log in check out the people running and even cast their votes.

The platform has been built with the user, in mind making sure it's easy to use works and's secure. Some of its features include controls that limit access, to roles the ability to automatically pull up student details using their IDs and real time tracking of votes. The results are also displayed in a simple way using pie charts. All these features come together to make the whole voting process more transparent, accurate and efficient.

The Student Online Voting System is a step, for schools bringing voting into the digital age. By doing it replaces methods with a contemporary and trustworthy system. This change not saves time. Cuts down, on the resources typically needed for elections but it also fosters a greater sense of involvement, trust and equity among students.

Literature Survey

The first system reviewed is an online voting platform built with PHP and MySQL. This technology was one of the first used for web-based elections in small institutions. It allowed administrators to register and manage candidates. Students could log in and vote only once. Results appeared in a simple text format. The main advantage of this system was its simplicity and the use of open-source tools, making it easy to set up and suitable for basic voting needs. However, it had significant limitations. It lacked a modern design, mobile responsiveness, and adequate security. Student identities were not properly verified. The absence of real-time statistics or graphical result visualization hurt both transparency and user experience.

Another notable system is the E-Voting application developed with Java Servlets and JSP. This platform connects to a relational database to manage users and elections. It introduced role-based access control, candidate management, and secure login through session handling. This ensured that each user could cast only one vote. Its strengths included a strong backend, better security compared to PHP-based systems, and reliable server-side logic that increased trust in the election process. Despite these improvements, the system still faced challenges. The user interface was not designed for smartphones, which limited access for mobile users. Additionally, its setup and maintenance were more complex. The results were restricted to text-based displays without any graphical insights. The lack of automated student data verification further limited its scalability.

Methodologies

Existing Methodology

Most online voting systems start with manual registration. Administrators manage registration for both students and candidates. Because this process is manual, there is no automatic way to verify identities, like linking a student ID to a database. This gap can lead to mistakes and misuse. After registering, students log into the platform using a basic system that requires only a username and password. This setup lacks strong security features.

Once logged in, students see a list of candidates. This list shows each candidate's name and position in a straightforward way. The voting process is simple. Students can cast one vote, which is saved in a backend database, usually MySQL or SQLite. However, once a vote is submitted, voters cannot change their choice.

Election results are typically available only to administrators after voting ends. These results usually appear in plain text or simple tables, which do not offer clear visuals or analysis. The system does not provide real-time updates, so administrators must refresh the page manually to check participation levels.

In terms of usability, many systems have a basic interface that isn't designed for mobile devices. They often lack responsive design. Without modern styling and interactive elements, user engagement may be low. Additionally, there are no visualizations of results, making it hard to quickly assess participation trends.

Security is another major issue with traditional systems. They usually rely only on basic username and password authentication, lacking encryption, one-time passwords, or multi-factor authentication. Administrative tools are limited. Admins can create elections and view results, but they cannot manage different election stages with better dashboards or controls.

Proposed Methodology

The proposed Student Online Voting System addresses the weaknesses of traditional platforms. It offers a more automated, secure, and user-friendly solution. It automates registration using student IDs. Users enter their ID, and their details are fetched automatically from the MongoDB database. This process ensures that only valid students can register, improving authenticity.

The login system is role-based, providing separate access for administrators and students. Each login has unique credentials, which improves security compared to traditional username and password systems. After logging in, users encounter a modern, responsive interface built in React.js. The platform works on desktops, tablets, and mobile devices, making it accessible for all students.

The system includes role-specific dashboards. Administrators can create elections, add candidates, monitor statistics, and view results. Students can see elections organized into ongoing, upcoming, and completed categories, making it easier to track voting opportunities.

Admins can create elections dynamically by specifying the election type (President, Vice President, General Secretary), start date, and end date. Managing candidates is straightforward, as admins can add them directly through the admin dashboard. During the election, students vote through a user-friendly interface. After selecting a candidate, the system confirms the vote with a message like “Vote Cast Successfully” and prevents duplicate voting by checking the student’s voting history.

A key improvement is the addition of real-time statistics. The admin dashboard shows the total registered students, votes cast, and students yet to vote. These statistics update automatically, allowing for clear tracking of election progress. Additionally, results are displayed using pie charts, showing vote distribution clearly.

To improve usability and engagement, the system features a news ticker for live updates, a floating “Vote Now” button, a scroll-to-top option, and a candidate spotlight section that highlights participant details. A simple guide on voting steps and a testimonial section on the homepage make the system more informative and engaging for students.

Through these features, the proposed methodology ensures the voting process is secure, clear, efficient, accessible, and appealing to students, encouraging greater participation in institutional elections.

Results

Fig 1 Welcome page

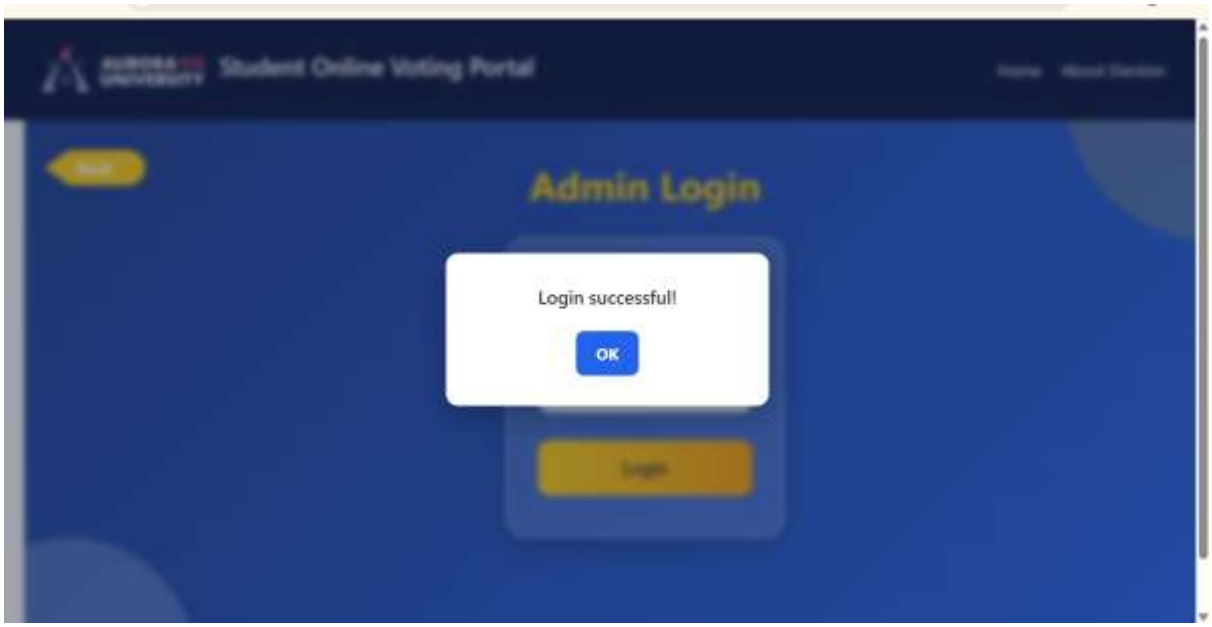

Fig 2 Admin Login

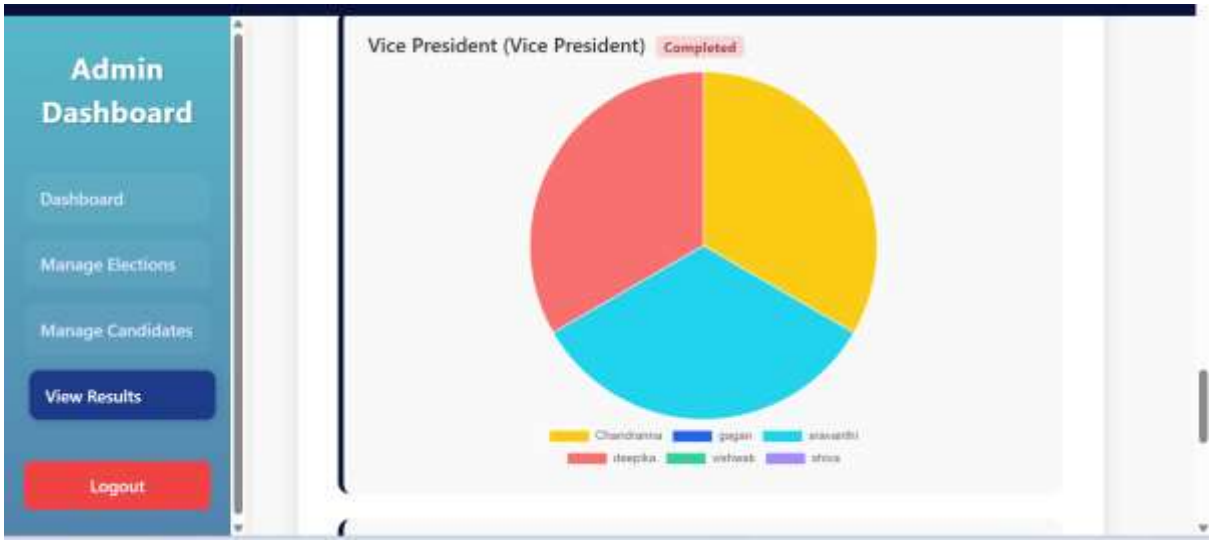

Fig 3: Results Page

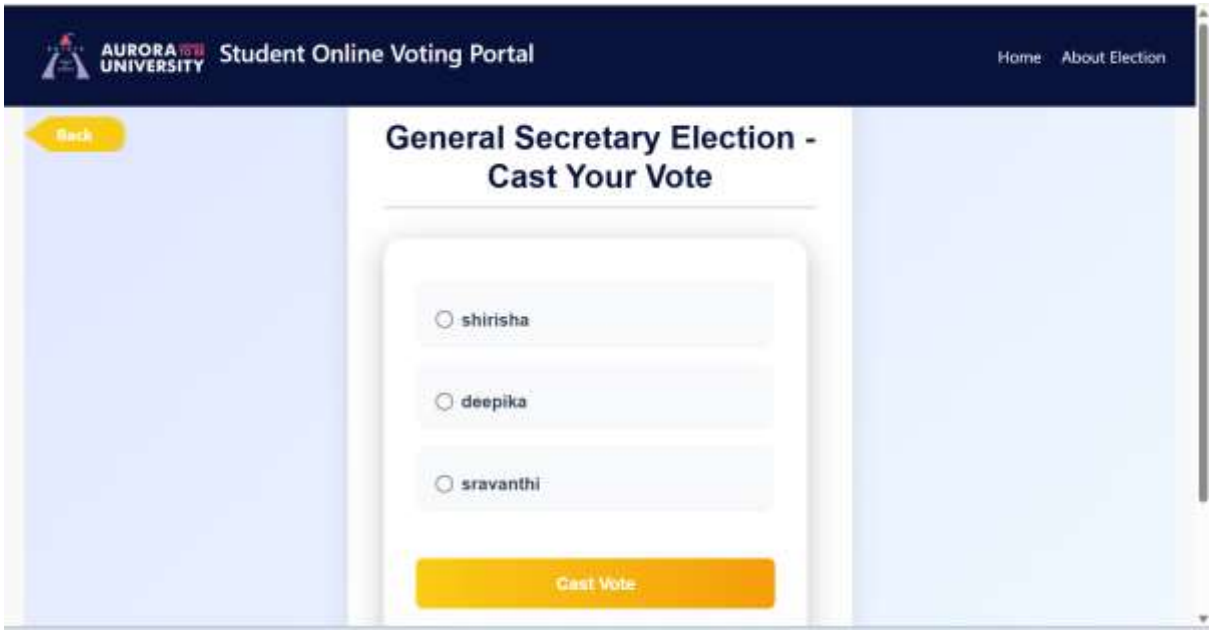

Fig 4: Voting Page

Discussion

The development of the Student Online Voting System was aimed at creating a secure, efficient, and user-friendly platform that streamlines the election process in educational institutions. Traditional voting methods, which involve paper ballots and manual counting, are not only time-consuming but also prone to errors and manipulation. With the increasing availability of technology and internet access, the idea of conducting student elections online has become both practical and essential.

This project addresses several real-world problems faced by colleges during elections such as managing voter data, ensuring each student votes only once, keeping records secure, and delivering accurate results on time. By using React.js for the frontend, the application offers a modern and responsive user interface that enhances the user experience across different devices. The use of Node.js and Express.js on the backend ensures smooth handling of requests, session management, and secure login functionalities. For data storage, MongoDB Compass provides a flexible and scalable NoSQL solution where all voter, candidate, and election information is stored safely.

One of the key highlights of the system is the automation of the registration process. When a student enters their Student ID during signup, the system automatically fetches their details from the database, reducing errors and preventing fake registrations. The system also restricts students to vote only once, and displays clear messages to prevent repeat voting attempts. Another significant feature is the admin dashboard, which shows real-time statistics like total voters, how many have voted, and how many are yet to vote. This data helps election organizers track participation and ensure fairness.

In addition, the system supports the creation of different election posts such as President, Vice President, and General Secretary. Admins can set election dates, add candidates for each post, and manage the overall election process efficiently. The final results are displayed in a pie chart format, offering a visual summary of votes received by each candidate, making it easy for everyone to understand the outcome.

From the user's side, the experience is simple and intuitive. After logging in, students can view a categorized list of elections Ongoing, Upcoming, and Previous. They can read candidate details, cast their vote (Fig:4) with a single click, and immediately see a confirmation message. All pages are designed with clarity and consistency, including helpful features like a News Ticker, How to Vote guide, and Candidate Spotlight section, which make the platform not only functional but also engaging.

Throughout development, key challenges included ensuring data security, preventing duplicate voting, and maintaining session control for different types of users. These were resolved through proper backend validation, use of unique identifiers, and structured role-based routing.

Conclusion

The Student Online Voting System shows how technology can simplify, secure, and modernize the election process in schools. By switching from traditional paper voting to a digital solution, this project makes the process more efficient, clear, and user-friendly for both students and administrators.

Using React.js for the frontend creates a smooth and responsive experience. Node.js and Express.js on the backend ensure secure and reliable communication between users and the database. Integrating with MongoDB allows for easy data management and real-time updates.

This system offers several key benefits. It includes automatic student data fetching, role-based dashboards, categorized election views, live voting statistics, and graphical result presentations. These features improve overall functionality and encourage more students to participate in campus elections by making the process straightforward and accessible.

The project also supports fairness and integrity. It ensures that each student can vote only once and that votes are stored and counted correctly. The admin panel makes it easy to manage elections, monitor participation, and view results with just a few clicks.

References

1. https://github.com/topics/online-voting-system?utm_source
2. https://github.com/HariKishorePec/Student-Election-System?utm_source
3. https://github.com/Jaweki/online-students-voting-system?utm_source