UNIVERSITY OF BARISHAL

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SOFTWARE ENGINEERING PROJECT PROPOSAL

Vote Ballot – An Online Voting Platform

CSE-3104: Software Engineering and Information System Design Lab

Submitted By:

Team No.: 05

Team Name: Steadfast

Team Members:

Md Rayhan (22 CSE 005)

Md Rayhanul Islam Rony (22 CSE 011)

Shaida Khanom Sharna (22 CSE 019)

Ismita Jahan (22 CSE 028)

Sazzad Hossain (22 CSE 031)

Israt Jahan Tamanna (22 CSE 036)

Submitted To:

Md. Samsuddoha

Assistant Professor

Department of Computer Science and Engineering

Submission Date: 27 October 2025

Contents

1 Formation of Team Steadfast

Our project team, **Team Steadfast**, consists of six dedicated members, each contributing specific skills to ensure effective design, development, and implementation of the proposed online voting system.

1.1 Team Composition and Responsibilities

Member 1: Backend & Deployment Lead (Team Leader)

Name: Shaida Khanom Sharna (22 CSE 019)
Responsibilities:

- Coordinate overall project planning and task distribution.
- Lead backend development using Node.js and Express.js.
- Design system architecture and authentication flow.
- Ensure integration between frontend and backend modules.
- Monitor project progress and team collaboration.

Member 2: Requirement Analyst & Database Designer

Name: Md Rayhanul Islam Rony (22 CSE 011)
Responsibilities:

- Analyze system requirements and prepare technical specifications.
- Design and manage database schema using MongoDB.
- Prepare ER diagrams and data flow models.
- Maintain data consistency across system modules.

Member 3: Frontend Lead & UI/UX Designer

Name: Md Rayhan (22 CSE 005)
Responsibilities:

- Develop frontend using React.js and Tailwind CSS.
- Design responsive and user-friendly interfaces.
- Create UI mockups and prototypes.
- Coordinate with backend team for API integration.

Member 4: Frontend Developer & Documentation Officer

Name: Sazzad Hossain (22 CSE 031)
Responsibilities:

- Assist in frontend component development.
- Maintain project documentation and reports.
- Prepare presentation slides and user manuals.
- Support version control documentation.

Member 5: Tester & Backend Support

Name: Israt Jahan Tamanna (22 CSE 036)
Responsibilities:

- Perform functional and integration testing.
- Identify and document system bugs.
- Assist backend testing using Postman.
- Verify fixes after debugging.

Member 6: Maintenance & Backend Support

Name: Ismita Jahan (22 CSE 028)
Responsibilities:

- Handle system maintenance and updates.
- Monitor system performance and stability.
- Support backend optimization tasks.
- Manage backups and basic security updates.

1.2 SWOT Analysis of Team Steadfast

1.2.1 Strengths

- Familiarity with MERN stack technologies.
- Clear role distribution and team coordination.
- Focus on secure authentication and access control.
- Balanced contribution in development and documentation.

1.2.2 Weaknesses

- Limited experience with large-scale deployment.
- Dependence on stable internet connectivity.
- Academic workload may affect development speed.
- Limited budget for premium tools and hosting.

1.2.3 Opportunities

- Practical exposure to complete software development life cycle.
- Future expansion for different institutional elections.
- Potential academic and practical reference project.
- Scope for advanced feature integration in future.

1.2.4 Threats

- Security risks such as unauthorized access or attacks.
- User reluctance toward online voting systems.
- Possible system failure during active elections.
- Administrative or policy constraints.

2 Overview

VoteBallot is a university-centric online voting platform designed to ensure fairness, transparency, and efficiency in campus elections. Traditionally, student union and departmental elections are conducted manually, which often results in delays, logistical challenges, and the risk of vote manipulation. Our system aims to address these issues by providing a secure digital alternative where students can use their student ID credentials to participate either as voters or as candidates.

The platform introduces three major roles: Admin Panel, Election Authorities, and Students. Admins hold the highest level of control, managing users, monitoring activities, and ensuring overall system integrity. Election Authorities, primarily faculty members, act as commissioners to oversee the voting process, approve candidates, post notices, and publish results. Students can register, cast their votes, and stand as candidates for elections.

The system ensures anonymity of votes, strong authentication, and role-based access control to prevent fraud. It also introduces features such as dashboards for authorities, secure result publication, and logs for audit purposes. With this platform, universities can conduct elections such as central student union elections (like BUCSU/DUCSU), departmental associations, or club elections in a modern, secure, and convenient manner.

In short, VoteBallot aims to digitalize the current manual process into a transparent and tamper-proof solution, saving time, reducing costs, and building trust among students and faculty.

3 Problem Statement

University elections are critical in shaping student leadership and representation. However, the current manual voting system used in most universities suffers from several limitations. Firstly, manual elections are resource-intensive, requiring physical ballot papers, booths, and staff for management. This often leads to inefficiency, delays in vote counting, and human errors. Secondly, manual elections are vulnerable to manipulation and vote rigging, which raises concerns about fairness and transparency. Such issues undermine students' trust in the electoral process.

Another challenge is limited accessibility. Many students cannot participate actively if they are away during the election period. Similarly, authorities face difficulties in monitoring, organizing, and ensuring smooth operations of large-scale elections. The absence of centralized digital records also creates problems in verifying results and maintaining accountability.

Given the size of modern universities with thousands of students across multiple departments, managing elections manually has become increasingly impractical. There is also no real-time monitoring or automated way of publishing results, making the process time-consuming and vulnerable to disputes.

Therefore, there is a significant need for a secure online voting platform that can replicate the existing system digitally while eliminating the possibility of fraud, improving accessibility, and reducing operational overhead. VoteBallot directly addresses these problems by offering a transparent, secure, and scalable election management system tailored for universities.

4 Motivation

The motivation behind VoteBallot comes from the challenges and controversies surrounding university elections. Elections such as BUCSU/DUCSU and departmental associations are vital for student governance, but the traditional manual methods often result in disputes, low trust, and inefficiency. As students of computer science, we recognized this gap and wanted to apply our technical skills to design a solution that modernizes the electoral process while maintaining fairness.

We were inspired by how digital platforms have improved systems in banking, education, and government services. If online systems can handle financial transactions securely, then university-level elections can also be digitized in a trustworthy way. Furthermore, the COVID-19 pandemic highlighted the importance of remote and online participation, motivating us to create a system that ensures inclusivity even when students are not physically present on campus.

Our project not only addresses a real-world problem but also gives us the opportunity to learn about system security, authentication, data privacy, and large-scale application design. It carries significant social impact, as it promotes transparency and strengthens democratic practices within the university.

In summary, the motivation is twofold:

1. To solve the real problem of unfair, inefficient manual voting.
2. To gain valuable technical and professional learning by designing a secure, scalable, and impactful software system.

5 User & Admin Requirements

5.1 Students

Students are the primary users of the system and will interact with the platform in two ways: as voters and as candidates.

5.1.1 Voter Registration

- Students register using student ID, department, session, district, email, and phone number.
- The system performs backend validation to prevent duplicate accounts.
- Email is verified via OTP before account activation.

5.1.2 Student Profile

- Display student information: student ID, department, session, district, email, phone number and password.
- All fields are read-only except for profile picture, which the student can upload or update.
- Show current and past election participation.

5.1.3 Voting Process

- Secure login using student credentials.
- View ongoing elections (department-level or central).
- Select candidates from approved participant list.

- System enforces one vote per student.
- Votes stored as candidate name and ballot only (maintaining anonymity).

5.1.4 Notices and Results

- Students can view official election notices, schedules, and updates.
- Once elections conclude, students can view results published by the authorities.

5.2 Election Authorities

Election authorities are responsible for conducting elections fairly and efficiently.

5.2.1 Registration & Access

This is completely controlled by the admin. The authorities only log in to the website and view their dashboard.

5.2.2 Candidate Management

- Approved candidates are visible in the public candidate list.
- Authorities cannot access voting results until published; the "Show Result" button becomes active after a defined time.
- Insert candidate field in the student database.

5.2.3 Election Operations

- Schedule elections for specific departments, clubs, or central student bodies.
- Start and end voting sessions at designated times.
- Preview election results privately before official publication.

5.2.4 Notices and Communication

- Use internal dashboard for authority communication and handling complaints.
- Post important election-related announcements for students.
- Create polls to collect student's opinions.

5.3 Admin Panel

Admin Panel is the highest authority in the system, typically consisting of the Vise-Chancellor and designated team members.

5.3.1 Responsibilities

- Create a new admin.
- **User Management:** Add, delete, or update students and authority records.
- **Authority Approval:** Approve the registration of election authorities only.
- **Election Control:** Start, pause, or cancel elections in case of emergency.
- **Notice Management:** Publish or remove official announcements.
- **Activity Monitoring:** Track system activities and generate audit logs.
- **Emergency Handling:** Stop voting sessions immediately in case of technical or security issues.

5.3.2 Technical Issue Handling

- Students and authorities can report system issues through a visible "Report Issue" button.
- Admin panel receives, tracks, and resolves reported issues.

5.3.3 Security & Privacy

- Admin has access to all data (students, authorities, candidates), but cannot see individual vote choices.
- Audit logs ensure transparency without compromising voter privacy.

6 Development Approach

6.1 SDLC Approach (Agile Methodology)

The project will follow the **Agile Development Model**, ensuring flexibility, iterative progress, and frequent evaluation.

- The project will be divided into small, manageable sprints.
- Each sprint will focus on specific feature development and testing.
- Regular feedback sessions will allow continuous refinement and faster delivery.

Advantages:

- Faster and incremental delivery.
- Clear division of responsibilities among team members.
- Easy adaptation to new or changing requirements.

6.2 Tools & Technology

6.2.1 Frontend (Client Application)

- **React.js** — For building a dynamic and responsive user interface (Students, Authorities, and Admin dashboards).
- **Redux / Context API** — For efficient state management (user login, voting session, etc.).
- **Material UI / Tailwind CSS** — For modern, consistent, and responsive UI design.

6.2.2 Backend (Server Application)

- **Node.js & Express.js** — RESTful API development and core business logic.
- **JWT Authentication** — Secure login with token-based access.
- **Bcrypt** — For password hashing and secure user data.
- **Nodemailer / Twilio** — Email and SMS verification for account and OTP validation.

6.2.3 Database

- **MongoDB (NoSQL)** — For storing user data, votes, results, and activity logs.
  - Flexible schema: Separate collections for Students, Authorities, and Admins.
  - High scalability: Supports concurrent voting from thousands of users.

6.2.4 Security Measures

- HTTPS protocol and AES/RSA encryption for all data transfer.
- Role-based access control (RBAC) to isolate permissions for Students, Authorities, and Administrators.

6.2.5 Deployment & DevOps

- **Docker / Vercel / Netlify** — For frontend and backend deployment (demo stage).
- **MongoDB Atlas** — For secure and scalable cloud database hosting.
- **GitHub / GitLab** — For version control and collaborative development.

6.3 Project Timeline (10-12 Weeks Plan)

Week	Deliverables
1–2	Requirement analysis, documentation & project proposal preparation.
3–4	Database design (MongoDB schema) and UI mockup (Figma).
5–6	Student registration & authentication module.
7–8	Voting mechanism and candidate registration module.
9	Election Authority dashboard & result preview.
10	Admin panel, activity monitoring & notices.
11	Unit, integration, and security testing.
12	Final deployment, documentation, and presentation.

6.4 Software Development Life Cycle

1. **Requirement Analysis:** Gathering and refining all system requirements.
2. **System Design:** Creating ER diagrams, data flow diagrams, and architecture models.
3. **Implementation:** Developing modules using React, Node/Express, and MongoDB (database).
4. **Testing:** Conducting unit, integration, and security tests using Postman and manual QA.
5. **Deployment:** Hosting the system on Heroku/Vercel with MongoDB Atlas.
6. **Maintenance:** Continuous improvement, bug fixing, and version updates.

7 Budget and Estimation

Category	**Description**	**Estimated Cost (BDT)**
Development & Team Compensation	Salaries and allowances for developers, designers, testers, and coordinators during the 10-week cycle	600,000
Tools & Software Subscriptions	Figma Pro, Jira/Trello Premium, GitHub Pro, API testing, and design tools	45,000
Internet & Maintenance	Internet, power backup, and local maintenance during development	30,000
Consulting & Expert Review	Occasional expert consultations for UI/UX, security audit, and technical review	80,000
Hosting & Domain Services	Domain purchase, web hosting, SSL certificate, and server setup for deployment	45,000
Marketing & Promotion	Design of posters, social media campaigns, and awareness for user adoption	70,000
Documentation & Printing	Project proposal, reports, presentations, and printing materials	8,000
Contingency & Future Support	Backup funds for bug fixes, updates, and post-deployment support	100,000
Total Estimated Budget		**9,78,000**

8 Conclusion

Vote Ballot aims to modernize and secure the university election process through a reliable online platform. Using the Agile methodology and MERN stack technologies, the system will be developed incrementally to ensure flexibility, efficiency, and continuous improvement. It provides secure authentication, encrypted vote storage, and role-based access control to maintain transparency and fairness throughout the election. The platform minimizes manual errors, reduces administrative workload, and delivers real-time results with full accuracy. Overall, Vote Ballot represents a step toward digital transformation in academic governance ensuring a fair, efficient, and trustworthy election experience for all stakeholders.