Software Requirement Specification

for

Online College Class Representative (CR) Election System

Version 1.0 approved

Prepared by

Sanga Balanarsimha - 231IT062
G. Durga Sai Pavan - 231IT022
K. Pavan Teja - 231IT030
Department of Information Technology

Table of Contents

Revision History

Date	Reason For Changes	Version	Prepared by
24-08-25	Initial Version	1.0	Sanga Balanarsimha, Durga Sai Pavan, Pavan Teja

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification(SRS) is to formally define the functional and non-functional requirements for the Class Representative Election System (CRES), a secure, web-based platform designed to digitize and streamline the process of nominating, voting, and electing Class Representatives within an educational institution.

The intended audience includes:

- Administrators/Admin– to approve the election process model.
- Project Development Team – to understand and implement the requirements.
- Project Supervisors - to review and validate system functionality.

1.2 Product Scope

The Class Representative Election System (CRES) will be an independent, browser-based application accessible to both Admins and Students(users) via secure login.

Primary Capabilities:

The system will:

- Provide secure login for both students and admin.
- Enable student nomination for elections, subject to eligibility.
- Allow admin management of students, classes, and election schedules.
- Facilitate secure online voting during election periods.
- Store election policies and require student agreement before participation.
- Enforce first-login password change for security.

The system will not:

- Handle offline voting processes.
- Allow students to vote more than once.
- Integrate with national or state-level election bodies.
- Support elections outside the classroom context.
- Include biometric authentication in the prototype phase.

Benefits and Objectives:

- Transparency: Ensures fair and traceable election processes.
- Efficiency: Reduces time and resources compared to manual voting.
- Security: Prevents unauthorized voting and duplicate entries.
- Accessibility: Allows participation from any internet-enabled device

1.3 Glossary

Term	Description
ACID	Atomicity, Consistency, Isolation, Durability
Admin	Staff responsible for managing CRES classes, students, elections, and results
CRES	Class Representative Election System
Class	Specific academic group in which elections are conducted
CSV	Comma Separated Values – file format for storing tabular data
DB	Database
Default Password	Initial login password for students, generated using part of name, special characters, and date of birth (DDMMYYYY)
Election Policy	Rules and regulations that students must accept before nominating or voting
HTTPS	Hypertext Transfer Protocol Secure – secure communication over the internet
IP	Internet Protocol – unique address identifying a device on a network
MFA	Multi-Factor Authentication
OTP	One-Time Password
RBAC	Role-Based Access Control
Roll Number	Unique student identification number used as login credential
RAM	Random Access Memory – computer memory used for temporary storage while running applications
UI	User Interface
Usability	Ease with which the system can be learned and effectively used by students and admins

1.4 References

This SRS is based on the following documents and sources:

1. **Title:** IEEE Std 830-1998 – IEEE Recommended Practice for SRS.
   Link: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=720574
2. **Title:** Electronic voting
   Link: https://en.wikipedia.org/wiki/Electronic_voting
   Date of access: 25 August 2025

1.5 Overview

This document is organized into several sections to clearly define the requirements of the **Class Representative Election System (CRES):**

Section-2: General Description: Provides an overall view of the product, including its perspective, functions, user characteristics, constraints, and assumptions.
Section-3: Specific Requirements: Details all functional requirements (e.g., user authentication, election setup, voting, and result management) and non-functional requirements (e.g., performance, usability, security, maintainability).
Section-4: Goal of Implementation: Outlines general development suggestions for the implementation team to consider, such as scalability, user-friendliness, and adaptability.

Code	Category
C	Design Constraints
F	Functional Requirements
N	Non-Functional Requirements

2. General Description

2.1 Product Perspective

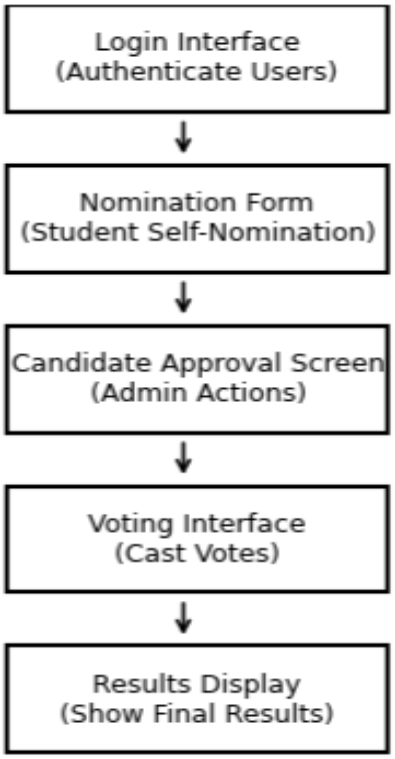
The Class Representative Election System (CRES) is a standalone, web-based application designed to automate and secure the process of conducting Class Representative elections in an educational institution. It operates independently but can optionally integrate with the institution's student information system to import student data.

The system consists of two primary modules:

- **Admin Module:** Manages student records, classes, nominations, and elections.
- **Student Module:** Allows students to log in, nominate themselves, vote, and view results.

The CRES system provides role-based access control ensuring that students and admins only access functionalities permitted for their roles.

Block Diagram - Class Representative Election System



2.1.1 System interfaces

- Secure admin and student authentication. Optional integration with institutional database for student details.
- Web browsers as the access interface for both user and admin roles.

2.1.2 User interfaces

- Separate web dashboards customized for Admin and user(students).
- Responsive design supporting desktop browsers.
- Clear form layout for nominations, voting, and administration tasks.

2.1.3 Hardware interfaces

- Server-side hosted on standard web server infrastructure.
- Client-side accessible from any modern desktop with an up-to-date web browser.

2.1.4 Software interfaces

- Backend stack utilizing Node.js and Express framework.
- Frontend built using React, HTML, CSS, and Bootstrap.
- Operating System: Windows and Linux.
- MySQL database for data persistence and Secure HTTPS communication.

2.1.5 Communications Interfaces

- All client-server communication is secured using HTTPS.

2.1.6 Memory Constraints

- Server deployment to have at least 4GB RAM and scalable storage depending on the institution's size and user base.

2.1.7 Operations

- Users log in to access the system and perform their tasks.
- The voting phase is interactive (students vote online), while result storage and counting are automatic.
- The system provides support like user authentication, vote encryption, and data validation.
- Regular backups of votes and results are created to prevent data loss.

2.1.8 Site Adaptation Requirements

- The system needs initial setup of class details (student list, candidate nominations).
- Election-specific parameters like voting dates and deadlines must be configured.
- The site may require custom rules (e.g., max number of candidates, eligibility criteria).
- Local server or cloud setup required to store voting and result data securely.
- Safety limits like restricting multiple votes per student must be enforced.

2.2 Product Functions

2.2.1 Administration Functions:

These functions empower administrators to manage all aspects of the election lifecycle, from setup to final results.

- **User and Class Management:** Add, edit, or delete student and class records.
- **Nomination Management:** Approve or reject student nominations for a position.
- **Election Scheduling:** Define and schedule specific election periods (e.g., start and end dates).
- **Results Publication:** Publish official election results for all to view.

2.2.2 Student Functions:

These features allow eligible students to participate in the election process, from self-nomination to casting a vote.

- **Secure Authentication:** Log in securely using a personal account, including the ability to change passwords.
- **Candidacy Management:** Nominate themselves for an election position, provided they meet eligibility criteria.
- **Candidate Viewing:** View a list of all approved candidates for each position.
- **Voting:** Cast a secure and verifiable vote during an active election period.

2.2.3 System Functions:

These are automated, backend functions that ensure the integrity and security of the entire election system.

- **Password Enforcement:** Mandate a password change upon a user's first login to enhance security.
- **Vote Integrity:** Prevent users from casting multiple votes or tampering with the voting process.
- **Audit Logging:** Store and maintain logs of user actions, such as the acceptance of election policies, for auditing purposes.

2.2.4 Security and Session Management:

These functions are critical for maintaining the security and integrity of user accounts and sessions.

- **Multi-Factor Authentication (MFA):** The system will require multi-factor authentication (e.g., OTP) for user identity verification, in addition to a password.
- **Password Reset via OTP:** OTP will be sent to the user's registered email for resetting passwords.
- **Last Login Notification:** Upon a successful login, the system shall display the date and time of the user's last login for security awareness.
- **Session Timeout:** User will be logged out automatically after inactivity.
- **Single-Device Login:** Only one active login per user at a time.

2.3 User Characteristics

- **Admins:** Institutional staff with moderate computer literacy responsible for election management and user administration.
- **Students:** Enrolled students familiar with using web-based platforms for forms and information access.

2.4 Constraints

- Requires active internet connection.
- Must comply with institutional election policies.

- Passwords must be securely stored using hashing algorithms.
- Supports only modern browsers.
- No offline or paper-based voting support.

2.5 Assumptions and Dependencies

- The institution provides accurate student data before system setup.
- Students have unique roll numbers and valid DOB records.
- Reliable internet connectivity during election periods.
- The hosting environment meets the software's minimum requirements.

2.6 Apportioning of Requirements

The following requirements may be deferred to future releases:

- Biometric authentication for voting.
- AI-based fraud detection.
- Multi-language support.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 Authentication and Access Control

Description: The system shall securely authenticate users and enforce access control policies.

F-001: Login

- Input: Roll number, password, OTP (multi-factor authentication)
- Processing: Validate credentials against database, enforce single-device login
- Output: Redirect to Student/Admin dashboard; display error message on failure

F-002: First-time Password Change

- Input: New password entered after default login
- Processing: Update password hash in database, enforce password complexity rules.
- Output: Confirmation message displayed

F-003: Email-based Password Reset

- Input: Registered email address
- Processing: Send reset link/OTP, allow user to set new password

- Output: Password reset confirmation message

F-004: Display Last Login

- Input: Login request
- Processing: Fetch last login timestamp from database
- Output: Last login time displayed on dashboard

F-005: Session Timeout

- Input: Inactivity period > 20 minutes
- Processing: Automatically log out user
- Output: Redirect to login page

F-006: Prevent Multiple Logins

- Input: Attempted login from a second device
- Processing: Deny login on second device
- Output: Error message displayed

3.1.2 Student Records and Class Management (Admin)

Description: Admins manage student profiles and class assignments.

F-007: Add Student

- Input: Student details (roll number, name, date of birth, class)
- Processing: Verify uniqueness of roll number, insert student into database
- Output: Confirmation message displayed

F-008: Manage Classes

- Input: Class details (create/edit/delete)
- Processing: Assign students to class, enforce one-class-per-student rule
- Output: Confirmation message displayed

3.1.3 Election and Nomination Management

Description: Admins schedule elections and approve nominations; students can self-nominate.

F-009: Schedule Election

- Input: Election nomination start/end dates, voting period
- Processing: Store schedule in database
- Output: Schedule confirmation message displayed

F-010: Self-Nomination (Student)

- Input: Candidate details (manifesto, photo ≤2MB)

- Processing: Validate input, store record for admin approval
- Output: Nomination submitted message displayed

F-011: Approve/Reject Nomination (Admin)

- Input: Admin approval/rejection
- Processing: Update candidate status in database
- Output: Candidate list updated with approval status

3.1.4 Voting Process

Description: Students cast votes securely and the system prevents double voting.

F-012: Cast Vote

- Input: Selected candidate from approved list
- Processing: Verify student eligibility, check prior voting, store vote securely
- Output: Vote confirmation message displayed

F-013: Prevent Double Voting

- Input: Second attempt to vote
- Processing: Check voting history
- Output: Error message displayed

3.1.5 Election Policy Acceptance

Description: Users must read and accept policy before nominating or voting.

F-014: Display Policy

- Input: User requests to nominate or vote
- Processing: Retrieve policy from database
- Output: Policy text displayed on screen

F-015: Accept Policy

- Input: User clicks “Accept”
- Processing: Store timestamp and user acknowledgment in database
- Output: Access granted to nomination or voting

3.1.6 Results Management

Description: System tallies votes automatically and publishes results.

F-016: Auto Tally Votes

- Input: End of voting period
- Processing: Count all votes securely
- Output: Election results stored in database

F-017: Display Results

- Input: Request to view results
- Processing: Retrieve final vote counts
- Output: Results displayed in tabular format

3.1.7 Audit Logging

Description: Logs sensitive actions for transparency and security.

F-018: Record Events

- Input: System actions (login, vote, approval)
- Processing: Write events to log with timestamp, user ID, and action details
- Output: Audit records stored

F-019: Access Logs

- Input: Admin request to view logs
- Processing: Check authorization, retrieve log data
- Output: Logs displayed (CSV or on-screen)

3.2 Non-Functional Requirements

3.2.1 Design and Implementation Constraints

C-001: The system shall follow university election policies and regulatory guidelines.

C-002: The system shall be limited to campus-provided hardware and network infrastructure.

C-003: The system shall integrate with the university email system for multi-factor authentication (MFA) and password resets.

C-004: The system shall use MySQL as the database management system.

C-005: The system shall adhere to common communication protocols such as HTTPS and SMTP.

C-006: The system shall use JavaScript as the programming language, following coding conventions and standards.

3.2.2 External Interface Requirements

3.2.2.1 User Interface

N-001: The system shall provide a responsive and intuitive interface accessible on both desktop and mobile devices.

N-002: The interface shall follow consistent design standards for clarity and ease of use.

N-003: All user-facing messages and error prompts shall be clear and easily understandable to novice users.

3.2.2.2 Maintainability

N-004: The system codebase shall follow consistent naming conventions and modular design for easy maintenance.

N-005: Detailed audit logs shall be maintained to assist in troubleshooting and updates.

3.2.2.3 Portability

N-006: The system shall run on both Linux and Windows Server environments without modifications.

N-007: The system shall support major modern browsers.

3.2.2.4 Usability

N-008: Users shall be able to complete login, nomination, and voting processes in a maximum of 3 steps.

N-009: The interface shall maintain consistent navigation and layout across different devices and screen sizes.

N-010: The system shall provide tooltips and guidance for first-time users where necessary.

3.2.3 Other Non-Functional Requirements

3.2.3.1 Performance Requirements

N-011: The system shall support at least 50 concurrent users during peak voting periods.

N-012: 95% of user requests (login, vote, nomination) shall be processed within 2 seconds.

N-013: Database queries shall respond within 500 ms under normal load.

N-014: The system shall achieve 99.9% uptime during scheduled election periods.

3.2.3.2 Reliability Requirements

N-015: All transactions shall be ACID-compliant to prevent data corruption.

N-016: The system shall ensure that no duplicate or lost votes occur under any condition.

N-017: The system shall recover automatically from unexpected server restarts within 5 minutes.

3.2.3.4 Availability Requirements

N-018: The system shall be accessible 24/7 during election periods, excluding scheduled maintenance.

N-019: Scheduled maintenance windows shall not exceed 1 hour per month.

3.2.3.5 Security Requirements

N-020: All communications shall be encrypted using HTTPS (TLS 1.2 or above).

N-021: User passwords shall be stored only as secure salted hashes.

N-022: The system shall implement role-based access control (RBAC) to restrict admin vs student access.

N-023: Every sensitive action (nomination, approval, voting, results publication) shall be logged with timestamp, user ID, and IP address.

3.2.3.6 Maintainability Requirements

N-024: Audit logs shall assist in troubleshooting and tracking system changes.

N-025: Bug fixes and feature updates shall be deployable with minimal downtime (<15 minutes).

3.2.3.7 Usability Requirements

N-026: The system shall provide clear error messages and intuitive guidance for all users.

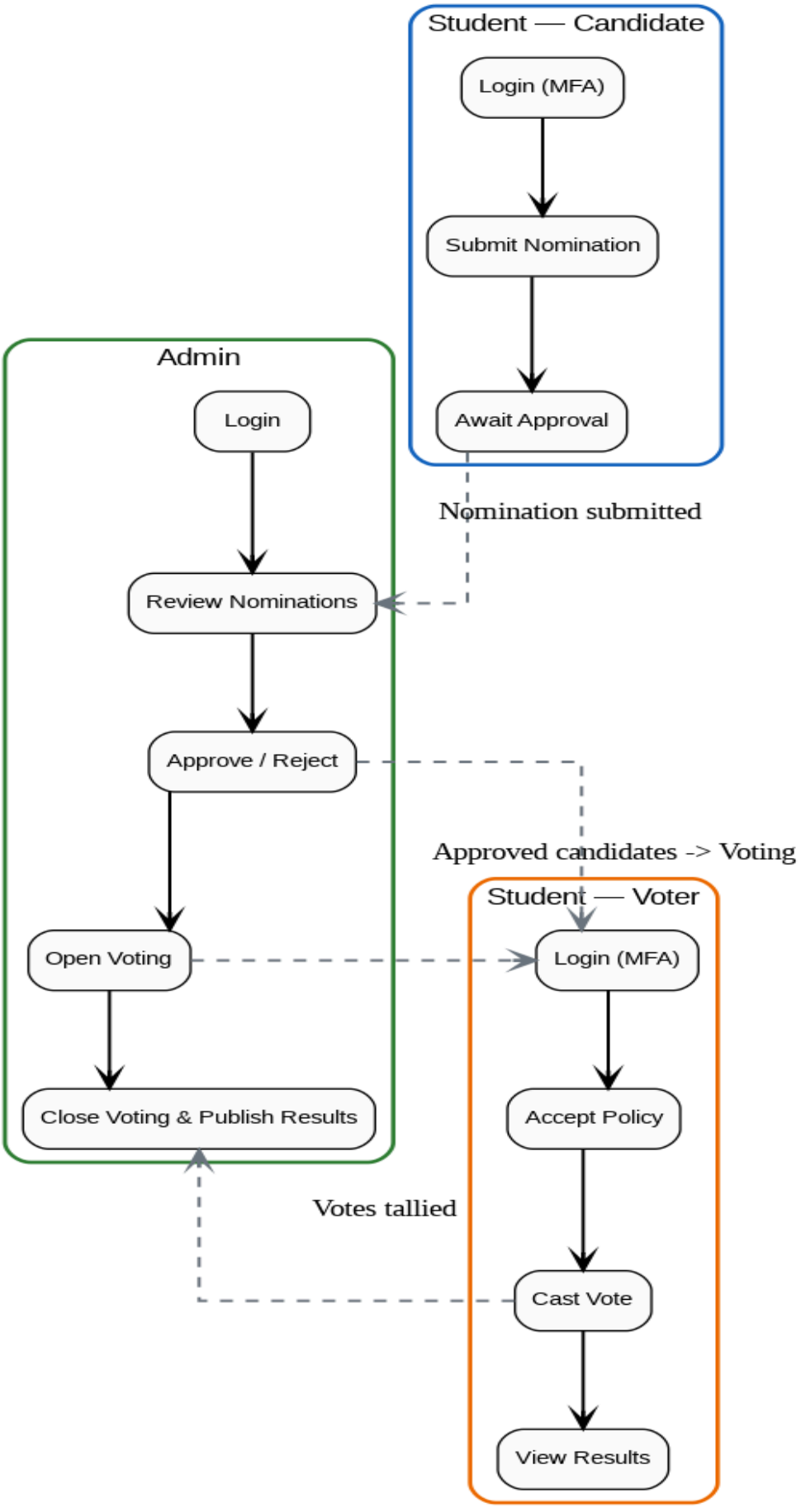
3.3 Goal of Implementation

- Use modular, reusable code for maintainability.
- Keep the user interface simple and consistent.
- Design with scalability in mind for future expansion.
- Add logging/monitoring for audit and troubleshooting.

Appendix A

Process Flow Diagram

Description: This diagram outlines a student election process with three main roles: Student Candidate, Admin, and Student Voter. A Candidate first submits a nomination for Admin approval. Once approved, the Admin opens voting. A Voter can then log in to cast their vote. After voting closes, the Admin tallies the votes and publishes the final results for everyone to view.


Student — Candidate
Login (MFA)
Submit Nomination
Await Approval
Nomination submitted
Admin
Login
Review Nominations
Approve / Reject
Approved candidates -> Voting
Student — Voter
Open Voting
Login (MFA)
Close Voting & Publish Results
Accept Policy
Votes tallied
Cast Vote
View Results