Software Requirements Specification Document (SRS)

Project Title: University E-Voting System
Course: CSE 327 - Software Engineering
Date: 19th October 2025

Chapter 01

1. Introduction

1.1 Purpose

This document provides the complete Software Requirements Specification (SRS) for the University E-Voting System. The goal of this project is to create a secure and transparent web application which will be used for conducting university level elections electing student councilors, club members and department representatives. This document specifies the objectives, target users, performance expectations and operational constraints of CAST-1000 system. It will serve as a formal tool for developers, testers, project managers, analysts, learners/educators and stakeholders through the life cycle of the software.

1.2 Intended Audience

This Software Requirement Specification (SRS) is for the audience who have interest directly or indirectly in design, development and use of the University E-Voting System.

- ***Administrators:*** Responsible for managing elections, user accounts, and overseeing system operations.
- ***Developers:*** The technical team in charge of coding, integration, and maintenance of the system.
- ***Project Managers (PMs):*** Oversee project progress, ensure scope alignment, and monitor team activities.
- ***Testers / Quality Assurance Engineers (QA):*** Validate and verify that each requirement is implemented correctly.
- ***University Election Committee:*** Official university body responsible for election supervision and policy compliance.
- ***Students (Voters and Candidates):*** End users who will vote, register as candidates, and access election information.
- ***Stakeholders / Observers:*** Faculty advisors, university authorities, and others interested in election transparency.

This SRS is meant to act as a knowledge tool for all with an interest in the system (from developers through University Election Authorities) so that they understand what role they play and how the system can be effectively designed delivered and operated.

1.3 Intended Use

This document will be used by the audience mentioned above to understand the aims, features and working of University E-Voting system and this section describes how the SRS will be used to enable each group of stakeholders to fulfill their roles effectively.

1.3.1 Administrators

- Use the SRS to understand system requirements, deployment steps, and their administrative responsibilities.
- Manage elections, approve candidate registrations, verify voters, and monitor results.
- Follow security and maintenance guidelines defined in the document to ensure reliable operation**.**

1.3.2 Developers

- Understand all functional, non-functional, and interface requirements.
- Use it as the main reference for implementation, integration, and database design.
- Ensure system features such as login, voting, and result generation match the requirements exactly.

1.3.3 Project Managers (PMs)

- Use the SRS to define project **scope**, **timeline**, and **deliverables**.
- Track team progress and ensure development follows the defined requirements.
- Use it for communication between developers, testers, and the university election committee.

1.3.4 Testers / QA Engineers

- Use this document to develop **test cases** for functional and performance testing.
- Verify that all specified requirements have been met and work as intended.
- Identify defects or inconsistencies and provide feedback to developers.

1.3.5 University Election Committee

- Use the SRS to verify that the system design aligns with **university election policies** and regulations.
- Review how authentication, voting, and result publication ensure fairness and transparency.
- Prepare training materials or manuals for administrative staff based on this document.

1.3.6 Students (Voters & Candidates)

- Understand their interaction flow in the system, like: registration, authentication, and voting.
- Learn about vote privacy and system security to build trust in the digital election process.
- Know the conditions, rules, and responsibilities defined for participants.

1.3.7 Stakeholders / Observers

- Refer to this document to gain insight into the project's scope, functionality, and outcome.
- Ensure that the E-Voting System supports ethical, fair, and transparent elections.
- Use it as a communication tool to align expectations between university authorities and the project team.

1.4 Product Scope

The objective of E-Voting System is to eliminate the traditional paper-based voting system and bring a centralized online based election system. It will help the election cycle from candidate registration to result publication, in a comfortable and secure way.

1.4.1 Purpose

This system aims to computerize all processes involved in the university-wide voting. It offers a quick, safe and easy solution to enable students in casting their votes, and administrators in managing the election effectively. Fair participation, reducing human error and increase access for every student.

1.4.2 Benefits and Objectives

- **Enhancing Election Efficiency:** Reduces time and effort required for managing and counting votes.
- **Ensuring Fairness and Transparency:** Prevents manipulation and guarantees one-person-one-vote.
- **Improving Accessibility:** Allows students to vote securely from anywhere within the university network.
- **Reducing Paper Usage:** Promotes an eco-friendly election process by eliminating paper ballots.
- **Immediate Results:** Automatically generates and displays election outcomes once voting ends.
- **Data Security:** Protects voter information through encryption and secure authentication.

1.4.3 Alignment with Institutional Goals

This system is consistent with the university's aspiration for digital transformation, student engagement and transparent governance. It aligns with the institution's vision of leveraging technology to enhance administrative effectiveness and student involvement. With this online voting system, the university shows agility and it clear it wants to embrace new technologies for decision making.

1.4.4 Relating to Educational Strategies

The university enhances the approach of deploying smart reality solutions towards academic and administration services through deployment of University E-Voting System. The technology encourages fairness, inclusivity and education among students. It is also symbolic of the university's emphasis on contemporary learning experiences and the cultivation of a culture that promotes trust and accountability in a student governance.

1.5 Risk Definitions

The Software Requirements Specification identifies potential risks that could affect the success, security, and reliability of the University E-Voting System. These risks may arise from user behavior, technical limitations, or administrative challenges. If not properly managed, such risks could compromise the fairness, accessibility, or transparency of the university's election process.

Risk Type	Description	Mitigation Strategy
Security Risk	Unauthorized access or data manipulation	Use encryption (SSL/TLS), hashed tokens, and RBAC policies
System Failure	Server crash during election period	Daily backups and cloud redundancy
Privacy Risk	Voter identity linked to vote	Separate vote table from user records
Network Risk	Connectivity issues for users	Auto-reconnect and token-based resumption
Data Loss	Database corruption	Versioned backups and transaction logs
Administrative Risk	Untrained staff mismanaging data	Provide training and user manuals

1.5.1 User Inactivity

There's a possibility that students, or candidates may not actively take part in election. The lack of voter turnout in itself contributes to undermining the trustworthiness and full representativeness of both results and is not helping much when it comes to trusting democratic systems.

1.5.2 Administrator Workload

The administrators can have a heavy load when they are checking candidate nominations, maintaining the voter list and handling potential system crashes in the election time. This may result in delays or errors unless it is underpinned by effective tools and automation.

1.5.3 Communication Breakdown

Lack of communication between developers, testers and the university election committee can lead to wrongly identified requirements that need not be met or improper implementation. Frequent meetup and documentation is important to mitigate this risk.

1.5.4 System Failure

And also there may be some unknown server crashes, or network outages which breaks the voting. This is not as good when voters could lose access during pending elections." "There should be redundant systems and failover capability.

1.5.5 Security Vulnerabilities

Hackers can try to tamper election data or access it. Poor validation of authentication or weak encryption result in unauthorized access, double voting or data fraud.

1.5.6 Data Loss

System failures, voter or results data loss may occur as a result of system errors, accidental deletion and database corruption. Backup and restoration schedules need to be established so that this does not happen.

1.5.7 Changing Stakeholder Needs

Active requirements are added, post-development start date (by vote, admin or election-committee members). Regular contact with constituents is needed to handle these shifting expectations.

Chapter 02

2. Overall Description

2.1 User Classes and Characteristics

2.1.1 User Class: Students (Voters and Candidates)

Students are the principal users of the University E-Voting System. They access the system through a secure web interface using their institutional credentials. Students may register as voters or candidates, depending on eligibility. They are expected to possess basic computer and internet-browsing skills. Each student can cast only one vote per election and view results after publication.

2.1.2 User Class: Election Officers / Faculty Administrators

Election Officers (faculty or authorized staff) configure, monitor, and supervise the entire election process. They create election events, verify candidate applications, manage voter eligibility, and ensure compliance with university regulations. Their technical expertise is moderate, and they utilize an administrative web dashboard for control and reporting.

2.1.3 User Class: System Administrators

System Administrators maintain the underlying infrastructure of the platform. They manage the Python-based backend services, databases, and network configurations. Their role requires advanced technical skills, and they possess full access privileges for maintenance, backups, and security enforcement.

2.2 User Needs

2.2.1 Students (Voters and Candidates)

- Need an intuitive and responsive web interface built with HTML and CSS to navigate, authenticate, and cast votes.
- Require secure login and confirmation that each vote is confidential and immutable.
- Expect real-time election information and transparent publication of results.
- Candidates need a structured nomination process and automated notification of approval status.

2.2.2 Election Officers / Faculty Administrators

- Require backend tools for creating elections, defining eligibility criteria, and approving candidates.
- Need automated result generation and graphical summaries to ensure efficiency and accuracy.
- Expect access logs and audit trails for post-election verification.
- Require the ability to export reports and statistics for university archives.

2.2.3 System Administrators

- Need access to server configurations and database management utilities.
- Require detailed system logs for debugging, performance monitoring, and security auditing.
- Expect comprehensive technical documentation for deployment and maintenance procedures.

2.3 Operating Environment

2.3.1 Hardware Platform

- Client: Any device (desktop, laptop, tablet, smartphone) capable of running a modern web browser.
- Server: University-hosted or cloud server with a multi-core processor, 8 GB RAM minimum, 100 GB storage, and redundant backup drives.

2.3.2 Operating System and Versions

- Server OS: Linux (Ubuntu Server 22.04 LTS) or Windows Server 2022.
- Client OS: Windows 10 / 11, macOS, Android, or iOS.

2.3.3 Software Components and Applications

- Frontend: HTML5, CSS3, and JavaScript (for interactivity and responsive design).
- Backend: Python using Django or Flask framework.
- Web Server: Apache or Nginx configured with uWSGI / Gunicorn for Python applications.
- Database: MySQL or PostgreSQL for relational data storage.

2.3.4 Database Compatibility

The database stores encrypted votes, user credentials (hashed), and election metadata. It supports SQL queries, relational integrity, and backup scheduling through Python database connectors.

2.3.5 Interoperability

The system integrates with the university's Single Sign-On (SSO) or LDAP authentication server. It is interoperable with institutional email systems for automatic notifications and supports all major browsers (Chrome, Firefox, Edge, and Safari).

2.3.6 Network Requirements

A reliable LAN / internet connection is required throughout the election period. The backend server must handle concurrent HTTP requests efficiently using asynchronous Python processing and load balancing if hosted on cloud infrastructure.

2.3.7 Security Considerations

- All data transmissions occur via HTTPS (SSL/TLS).
- Passwords are hashed (Bcrypt/SHA-256) and votes are encrypted before storage.
- Role-based access control (RBAC) prevents unauthorized operations.
- Audit logs and anomaly detection are enforced through Python middleware scripts.

2.4 Constraints

2.4.1 Technical Constraints

The system will function entirely as a browser-based web application using HTML/CSS for the interface and Python for the backend. No mobile-native or offline version is required. It must handle at least 1,000 simultaneous logins.

2.4.2 Time Constraints

Voting sessions and registration periods are predefined by administrators. Access is automatically locked after the configured election deadline.

2.4.3 Budget Constraints

The project prioritizes open-source tools (Python, Django, and MySQL) to minimize software costs. Hosting will utilize existing university servers or a limited-cost cloud instance.

2.4.4 Regulatory and Compliance Constraints

All operations must conform to the university's election guidelines, data-protection acts, and ethical standards governing electronic voting systems.

2.4.5 Resource Constraints

System efficiency depends on server bandwidth, university network quality, and IT-staff availability during active elections.

2.5 Assumptions

2.5.1 User Participation

All eligible students will actively participate in elections and adhere to voting regulations.

2.5.2 User Proficiency

Users possess basic familiarity with web browsers and online authentication procedures.

2.5.3 Administrator Authority

Election Officers and System Administrators have official authorization to configure elections and release results.

2.5.4 Document Accessibility

User manuals, administrator guides, and technical setup instructions will be available in PDF and HTML formats.

2.5.5 Document Relevance

This SRS remains valid until major technological or policy updates require revision.

2.5.6 Aligned Objectives

All stakeholders share the common goal of ensuring secure, efficient, and transparent university elections.

2.5.7 Stakeholder Collaboration

Effective coordination among developers, testers, administrators, and the Election Committee is assumed throughout the project.

2.5.8 Consistent Connectivity

Stable internet connectivity and uninterrupted server uptime are assumed during the entire election cycle.

Chapter 03

3. Requirements

3.1 Functional Requirements

3.1.1 Register and ID verification

As a user (voter or candidate), I want to access the system and register on the platform with my ID card or candidate card to verify the access in the system. I want to fill the registration form with my username, password, email, address, phone number, (photo optional) and submit to have a verified account.

Success:

- After successful submission, there will appear a Terms and Condition.
- If users agree to those conditions the form will proceed to supervisors.
- Users will receive confirmation Via Email.

Failure

- If the supervisor disproves also the applicant will be notified Via Email.
- Also if the applicants (both voters and candidates) the system will lead them to the registration page.

3.1.2 User Authentication (Login)

As a user (as voter or candidate), they should be able to securely log in the system and access all the facilities from the systems.

Success:

- If the credentials are valid from the users the system will take them to their dedicated homepage.
- Different users have different homepage and their accessibility will be different. So voters will be taken to different homepage and candidate will be taken to different homepage.

Failure:

- System shows Invalid user id or password.
- Also it will return to the login page.

3.1.3 Create elections

The supervisor can create an election like a module where voters and candidates can participate.

Success

- The platform has voters and candidates which will take all the information of the voters and candidates and system will show all the data.

3.1.4 Manage voter eligibility list

The supervisor can manage whether a voter is eligible or not.

Success

- When a voter is eligible it will notify that they are eligible via Email.

Failure

- If they are not eligible also notified with reason via Email.

3.1.5 Approve candidates

The candidates have to apply for their nomination. It will be approved by the supervisor. They will have some requirements. Those will be in form which they will fill and submit.

Success:
If approved they will be confirmed by Email.

Failure:
If they are not approved they will also be notified via Email.

3.1.6 General ballot tokens / send invites

Supervisor can create tokens for eligible candidates or send invitations

Success

Failure

3.1.7 Cast Vote (Once)

This is applicable for both candidates and voters. Even candidates can vote themselves but just once.

Success

Failure

3.1.8 View personal ballot status (cast/not cast, never choices)

Regardless of voters and candidates, they can see the status of their casting vote and also see the other candidates.

Success

Failure

3.1.9 View live/final result (Aggregate only)

Regardless of voters and candidates, they can see the ultimate results of who is the winner and the other candidates' positions.

Success

Failure

3.1.10 Audit logs

This is accessible by supervisor and public can only read the summary.

Success

Failure

3.2 Nonfunctional requirements

Not specified yet.