Software Requirements Specification

for

Online Election System

Version 1.0 approved

Prepared by

CodeCrew

07.02.2025

Table of Contents

Revision History **3**
1. Introduction **4**
1.1 Purpose 4
1.2 Document Conventions **4**
1.3 Intended Audience and Reading Suggestions **4**
1.4 Product Scope **4**
1.5 References **5**
2. Overall Description **5**
2.1 Product Perspective 5
2.2 Product Functions **5**
2.3 User Classes and Characteristics **5**
2.4 Operating Environment **5**
2.5 Design and Implementation Constraints **6**
2.6 User Documentation **6**
2.7 Assumptions and Dependencies **6**
3. External Interface Requirements **7**
3.1 User Interfaces 7
3.2 Hardware Interfaces **7**
3.3 Software Interfaces **7**
3.4 Communications Interfaces **7**
4. System Features **7**
5. Other Nonfunctional Requirements **9**
5.1 Performance Requirements 9
5.2 Safety Requirements **10**
5.3 Security Requirements **10**
5.4 Software Quality Attributes **10**
5.5 Business Rules **10**
6. Other Requirements **10**
7.1 Appendix A: Glossary **11**
7.2 Appendix B: Analysis Models **11**
7.3 Appendix C: To Be Determined List **11**

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The Online Election System is a digital solution designed to streamline the electoral process by ensuring security, efficiency, and transparency. Traditional election methods often suffer from inefficiencies such as manual errors, long processing times, and security vulnerabilities. This system addresses these concerns by incorporating encryption techniques, secure authentication, real-time analytics, and audit logging mechanisms. The primary goal is to provide a seamless, user-friendly platform for voter registration, election management, secure vote casting, and result reporting.

1.2 Document Conventions

This Software Requirements Specification (SRS) follows the IEEE Standard for Software Requirements Documentation. The following conventions are used throughout this document:

- ***Bold text** is used for section headers.*
- *Italic text is used for emphasis.*
- *Numbered headings indicate hierarchical organization.*
- *Requirements are uniquely labeled as REQ-XX to ensure traceability.*
- *Higher-level requirements are assumed to be inherited by detailed requirements unless explicitly stated otherwise.*

1.3 Intended Audience and Reading Suggestions

The intended audience includes software developers, project managers, quality assurance teams, system administrators, government agencies, election commissions, and other stakeholders. The document is structured to provide a high-level understanding in the introductory sections before diving into technical details in later chapters. Readers are encouraged to start with the overall system description and proceed to system features and requirements.

1.4 Product Scope

The Online Election System is designed as a web-based application that enables secure voter registration, real-time vote casting, election configuration, and accurate result reporting. The system enhances electoral transparency and reduces fraud risks while ensuring compliance with legal frameworks. It supports various election types, including first-past-the-post and proportional representation.

1.5 References

- *IEEE Std 830-1998 - Recommended Practice for Software Requirements Specifications*
  - *Available at: IEEE Standards*
- *OWASP Security Guidelines*
  - *Available at: OWASP*
- *ISO/IEC 27001:2013 - Information Security Management*
  - *Available at: ISO Standards*
- *National Election Commission Guidelines*
  - *Available at: Election Commission*
- *Database Encryption Standards (AES, bcrypt)*
  - *AES Details: NIST AES Standard*
  - *bcrypt Details: bcrypt algorithm*

2. Overall Description

2.1 Product Perspective

The Online Election System is an independent, web-based platform designed to modernize the voting process. It consists of multiple microservices handling voter management, election configuration, vote processing, result reporting, and auditing. The system ensures interoperability with government databases and external identity verification systems.

2.2 Product Functions

- *Secure voter registration and authentication*
- *Election setup and candidate management*
- *Vote submission with validation and fraud prevention*
- *Real-time result calculation and visualization*
- *Comprehensive auditing and security logging*

2.3 User Classes and Characteristics

- ***Voters:*** *Individuals eligible to vote, requiring an intuitive and secure interface.*
- ***Election Administrators:*** *Responsible for setting up elections, managing voters, and monitoring results.*
- ***Candidates:*** *Individuals running for election, with access to profile management.*
- ***Auditors:*** *Ensure system integrity by reviewing logs and monitoring security.*

2.4 Operating Environment

The Online Election System is designed to function efficiently across various platforms and devices. The system architecture comprises the following components:

- ***Frontend:*** *Built using React.js and Next.js, ensuring an interactive, responsive, and accessible user interface across desktop and mobile devices.*

- ***Backend:*** *The backend will be built using Ballerina for its scalability and flexibility. It will manage voter registration, election configuration, vote casting, and result calculation.*
- ***Database:*** *MySQL or PostgreSQL serves as the primary database for storing voter details, election configurations, and voting records. Data is encrypted to ensure privacy and integrity.*
- ***Hosting Environment:*** *The system is deployed on cloud services like AWS or Azure, ensuring scalability, availability, and reliability. Kubernetes and Docker are used for containerization and deployment management.*
- ***Security Protocols:*** *Secure data transmission is enforced through HTTPS and TLS encryption. User authentication follows OAuth 2.0 and JWT mechanisms, preventing unauthorized access.*
- ***Performance Optimization:*** *Load balancing and caching mechanisms (such as Redis) are implemented to optimize performance during high voter turnout periods.*
- ***Cross-Browser Compatibility:*** *The platform is compatible with major browsers including Chrome, Firefox, Edge, and Safari, ensuring a seamless experience across different devices and operating systems.*

2.5 Design and Implementation Constraints

- *The system must comply with regional and international election security regulations.*
- *Sensitive data, such as voter credentials, must be encrypted using AES and bcrypt hashing.*
- *API communications should be secured using HTTPS and JWT authentication to prevent unauthorized data access.*
- *The application must support multi-language capabilities for a diverse voter base.*
- *The system should maintain high availability and handle concurrent user requests efficiently, ensuring smooth performance even during peak election periods.*

2.6 User Documentation

- *A comprehensive user manual for voters, administrators, and auditors.*
- *Detailed API documentation for developers integrating external systems.*
- *System workflow diagrams and technical specifications explaining key system processes.*
- *Frequently Asked Questions (FAQ) section addressing common user concerns.*

2.7 Assumptions and Dependencies

- *Stable and uninterrupted internet connectivity is required for smooth system operation.*
- *Government-provided identity verification services will be available for voter authentication.*
- *The system relies on third-party cloud infrastructure providers (AWS, Azure) for hosting and scalability.*
- *External email and SMS gateways are used for voter notifications and authentication messages.*

3. External Interface Requirements

3.1 User Interfaces

- *Web-based, user-friendly UI with mobile responsiveness.*
- *Accessible interfaces for voters, administrators, and auditors.*
- *Interactive dashboards for election monitoring.*
- *Standardized form elements for easy navigation.*

Figma Prototype: *View UI Design*

3.2 Hardware Interfaces

- *Compatible with desktops, tablets, and smartphones.*
- *Support for biometric verification hardware (if applicable).*
- *Server-side hardware requirements for hosting.*

3.3 Software Interfaces

- *RESTful APIs for frontend-backend communication.*
- *Database integration with MySQL/PostgreSQL.*
- *Integration with external identity verification services.*
- *Secure logging and auditing tools.*

3.4 Communications Interfaces

- *HTTPS for secure web communication.*
- *Email/SMS gateway integration for voter notifications.*
- *WebSocket support for real-time updates.*

4. System Features

4.1 Voter Registration

4.1.1 Description and Priority

Voter registration allows eligible users to create an account, verify their identity, and participate in elections. Priority: ***High***

4.1.2 Stimulus/Response Sequences

- *User enters registration details.*
- *System validates input fields.*
- *User submits the form.*
- *System encrypts and stores voter data.*

- *User receives a confirmation email.*

4.1.3 Functional Requirements

- *REQ-1: Users must provide valid identification details.*
- *REQ-2: The system must verify user identity via OTP/email.*
- *REQ-3: The system should reject duplicate registrations.*

4.2 Election Configuration

4.1.1 ***Description and Priority***

Administrators can set up election parameters, including date, type, and candidate profiles.

Priority: ***High***

4.2.2 Stimulus/Response Sequences

1. *Admin logs in.*
2. *Admin configures election details.*
3. *System validates and saves election settings.*

4.2.3 Functional Requirements

- *REQ-1: The system must allow admins to configure elections.*
- *REQ-2: Elections must prevent duplicate candidate entries.*

4.3 Vote Casting

4.3.1 Description and Priority

Voters can securely submit their votes through the platform. Priority: ***High***

4.3.2 Stimulus/Response Sequences

1. *Voter logs in.*
2. *Voter selects a candidate.*
3. *System verifies voter eligibility.*
4. *Vote is securely submitted and stored.*

4.3.3 Functional Requirements

- *REQ-1: Voters must be authenticated before voting.*
- *REQ-2: System must prevent duplicate votes.*

4.4 Result Calculation and Reporting

4.4.1 Description and Priority

The system automatically calculates and reports election results. Priority: ***High***

4.4.2 Stimulus/Response Sequences

1. *Election ends.*
2. *System aggregates and calculates votes.*
3. *Results are displayed to authorized users.*

4.4.3 Functional Requirements

- *REQ-1: System must accurately calculate election results.*
- *REQ-2: Results must be securely stored.*

4.5 Auditing and Security

4.5.1 Description and Priority

Ensures transparency through secure audit logs and security measures. Priority: ***High***

4.5.2 Stimulus/Response Sequences

1. *System records all administrative actions.*
2. *Security mechanisms detect unauthorized access.*

4.5.3 Functional Requirements

- *REQ-1: The system must log all critical actions.*
- *REQ-2: Multi-layer security must prevent unauthorized access.*

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- *The system must handle up to 100,000 concurrent users without performance degradation.*
- *Voter authentication and vote submission should be processed within 2 seconds.*
- *Database queries should be optimized to retrieve results within 1 second for election results and voter verification.*
- *The system should support load balancing and auto-scaling to accommodate peak election periods.*

5.2 Safety Requirements

- *The system must ensure data redundancy and regular backups to prevent data loss.*
- *A failover mechanism should be in place to guarantee continuous operation during unexpected downtimes.*
- *System integrity checks should be performed periodically to identify data corruption or inconsistencies.*
- *User session timeouts should be implemented to prevent unauthorized access to voter data.*

5.3 Security Requirements

- *All data transmissions must be encrypted using TLS 1.2 or higher.*
- *Voter credentials must be securely stored using bcrypt hashing.*
- *Role-Based Access Control (RBAC) must be enforced to restrict user privileges.*
- *Multi-Factor Authentication (MFA) should be implemented for administrative accounts.*
- *Anomaly detection and intrusion prevention systems should be integrated to monitor for suspicious activities.*

5.4 Software Quality Attributes

- ***Reliability:*** *The system should maintain at least 99.99% uptime.*
- ***Usability:*** *The UI should be intuitive and accessible, complying with WCAG 2.1 guidelines.*
- ***Maintainability:*** *The system should follow modular coding practices to simplify updates and patches.*
- ***Scalability:*** *Cloud-based infrastructure should allow horizontal scaling to accommodate varying loads.*
- ***Interoperability:*** *The system should support integration with external identity verification services.*

5.5 Business Rules

- *Only verified users are allowed to register as voters.*
- *Election administrators must finalize configurations before the election begins.*
- *A voter may only cast one vote per election.*
- *Votes cannot be altered or removed once submitted.*
- *Election results should be verified by auditors before being published.*

6. Other Requirements

- *The system must comply with local and international election laws.*
- *Multi-language support should be provided to accommodate diverse voter demographics.*
- *The application should be deployable in both on-premise and cloud environments.*
- *A comprehensive logging system should be in place for auditing and compliance tracking.*

- *Regular security audits and penetration tests should be conducted to ensure the system remains secure..*

7.1 Appendix A: Glossary

- ***JWT (JSON Web Token):*** *A secure method for transmitting information.*
- ***RBAC (Role-Based Access Control):*** *Permission management based on user roles.*
- ***HTTPS (Hypertext Transfer Protocol Secure):*** *A protocol for secure communication.*
- ***MFA (Multi-Factor Authentication):*** *A security mechanism requiring multiple forms of verification.*
- ***Ballerina:*** *A cloud-native programming language optimized for API integrations.*

7.2 Appendix B: Analysis Models

- **Entity-Relationship (ER) Diagram:** Illustrates the relationships between system entities. (View on GitHub)
- **Sequence Diagrams:** Show the interactions between users and the system during voter registration and vote casting. (View on GitHub)
- **Use Case Diagrams:** Define the interactions of different system users (voters, administrators, auditors). (View on GitHub)
- **Class Diagrams:** Represent the structure of the system with classes, attributes, and relationships. (View on GitHub)
- **Activity Diagrams:** Model the flow of activities within the election process, from registration to result calculation. (View on GitHub)

7.3 Appendix C: To Be Determined List

- *Final hosting and deployment infrastructure decisions.*
- *Additional security measures for advanced authentication.*
- *Integration with biometric authentication (if applicable).*
- *Finalization of third-party API integrations.*