Voting Application

Advanced Web Development

INT-219

Software Requirements Specification

12-April-2024

Prepared for
Continuous Assessment 3
Spring 2024

Submitted by: - Submitted to: -
Anuj Dadwal Dr. Brijesh
Pandey
Reg. no: - 12216794 UID: - 31141
Section: - K22LQ

Table of Contents

Introduction

The Software Requirements Specification (SRS) document provides a comprehensive overview of the requirements for the development of the Voting Application system. This document serves as a guide for software engineers involved in the design and implementation of the software product described by the requirements listed herein.

The Voting Application system aims to provide users with the ability to participate in the voting process by casting their votes for their respective candidates. Additionally, users will have access to features such as viewing total votes cast, accessing the list of candidates, and engaging in the voting process seamlessly.

This document contains all the necessary information needed by software engineers to adequately design and implement the Voting Application system. It outlines the functional and non-functional requirements, external interface requirements, design constraints, and other relevant details crucial for the successful development of the system.

Throughout this document, careful attention has been paid to ensure clarity, completeness, and consistency of the requirements stated. By adhering to the guidelines provided herein, the software engineering team can effectively translate these requirements into a functional and reliable software product that meets the needs of its users.

With the Voting Application system, users will experience a seamless and intuitive voting process, while administrators will have the tools necessary to manage candidates and monitor the overall voting activity. This SRS document serves as a roadmap for the development of a robust and efficient voting system that upholds the principles of transparency, accessibility, and integrity.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to outline the requirements for the development of the Voting Application system. It serves as a comprehensive guide for software engineers tasked with designing and implementing the software product. The primary objectives of this SRS include:

- Clearly defining the functional and non-functional requirements of the Voting Application system.
- Providing a detailed understanding of the system's features, functionalities, and constraints.
- Serving as a communication tool between stakeholders, including developers, project managers, and clients, to ensure a common understanding of project goals and deliverables.
- Facilitating the design, development, and testing phases of the project by providing a reference document that outlines the project's scope and requirements.

- Serving as a basis for project planning, estimation, and resource allocation.

Intended Audience:

The intended audience for this SRS includes:

- Software Engineers: Responsible for designing, developing, and testing the Voting Application system.
- Project Managers: Responsible for overseeing the project's progress, resource allocation, and timeline management.
- Quality Assurance (QA) Engineers: Responsible for testing the system to ensure that it meets the specified requirements and quality standards.
- Stakeholders: Including clients, end-users, and any other individuals or entities with a vested interest in the successful development and deployment of the Voting Application system.

1.2 Scope

Identified Software Product:

The software product to be produced is the Voting Application system.

Explanation of Software Product:

The Voting Application system will enable users to participate in voting processes by casting their votes for respective candidates. It will facilitate functionalities such as user authentication, candidate management, voting processes, and administrative tasks.

Description of Application:

The Voting Application system aims to achieve the following objectives:

- Provide users with a seamless and intuitive platform to cast their votes securely.
- Allow users to view the list of candidates and their respective profiles.
- Enable administrators to manage candidates, including adding, updating, and deleting candidate information.
- Ensure data integrity and security throughout the voting process.
- Facilitate transparent and fair elections by recording and tallying votes accurately.
- Enhance accessibility by providing user-friendly interfaces and support for diverse user demographics.

Scope of Software Product:

The scope of the Voting Application system includes:

User authentication: Users will be able to sign up, log in securely, and manage their profiles.
Candidate management: Administrators will have the ability to add, update, and delete candidate information.
Voting processes: Users will be able to cast their votes for respective candidates, with measures in place to prevent duplicate voting and ensure fairness.

Reporting and analytics: The system may include features for generating reports and analysing voting data, depending on the project requirements and specifications.

Exclusions:

The Voting Application system will not include:

- *Frontend development:* This SRS focuses solely on the backend development of the Voting Application system. Frontend design and implementation are outside the scope of this document.
- *Hardware integration:* The system will not involve integration with specific hardware devices or sensors.
- *Real-time monitoring:* While the system may include features for reporting and analytics, real-time monitoring of voting activities is not within the scope of this document.
- Consistency with Higher-Level Specifications:

This SRS will be consistent with any higher-level specifications, such as the System Requirements Specification (SRS), if they exist. It will ensure alignment with overarching project goals, objectives, and constraints outlined in those documents.

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
User Interface (UI)	The graphical or textual interface through which users interact with the system.
Node.js	An open-source, cross-platform JavaScript runtime environment that executes JavaScript code outside a web browser.
Authentication Controller	Process responsible for handling user authentication requests and responses.
Express.js	A web application framework for Node.js used to build web applications and APIs.
MongoDB	A NoSQL database program that uses a document-oriented data model for storing data.

JWT	JSON Web Token, a compact, URL-safe means of representing claims to be transferred between parties securely.
Middleware	Software that acts as a bridge between an operating system or database and applications. It handles communication and data management.
HTTP	Hypertext Transfer Protocol, the foundation of data communication on the World Wide Web.
API	Application Programming Interface, a set of rules that allow different software applications to communicate with each other.
Endpoint	A URL (Uniform Resource Locator) that is used to send requests to the server for specific actions or resources.
Payload	The actual data being transmitted in a network packet or secured message.
Role	Defines the level of access or permissions granted to a user within the system.
Password Hashing	A cryptographic technique used to convert a password into a scrambled representation, making it more secure for storage.
Admin	Short for Administrator, a user role with elevated privileges, often responsible for managing the system.
Status Code	A three-digit number in an HTTP response that indicates the outcome of the server's attempt to fulfill the client's request.
Error Handling	The process of gracefully managing and responding to unexpected or erroneous situations that occur during program execution.
Session	A period of interaction between a user and a system, starting from the user's login until logout.
Token	A piece of data used in authentication and authorization processes to prove the identity of a user.

1.4 References

Node.js Documentation

Official documentation for Node.js, providing information on installation, usage, and APIs.
Available at: https://nodejs.org/docs/latest/api/

Express.js Documentation

Official documentation for Express.js, offering guidance on building web applications and APIs.
Available at: https://expressjs.com/en/guide/routing.html

MongoDB Documentation

Official documentation for MongoDB, including information on installation, data modeling, and query operations.
Available at: https://www.mongodb.com/docs/

JWT (JSON Web Tokens) Documentation

Documentation for JWT, explaining the usage and implementation of JSON web tokens for authentication.
Available at: https://jwt.io/introduction

Postman Documentation

Documentation for Postman, providing guidance on using the tool for API development and testing.
Available at: https://learning.postman.com/docs/introduction/overview/

Other relevant textbooks, research papers, online tutorials, or articles that were consulted during the development of the SRS.

1.5 Overview

Description of Contents:
The rest of the Software Requirements Specification (SRS) document contains detailed information regarding the requirements, functionalities, and design considerations for the Voting Application system. It is structured to provide a comprehensive understanding of the system's specifications and serve as a guide for the software engineering team throughout the development process.

Organization of the SRS:
The SRS is organized into several sections, each focusing on different aspects of the Voting Application system. The document follows a logical flow, starting with introductory sections and gradually delving into more specific details. The organization of the SRS is as follows:

Introduction: Provides an overview of the SRS document, including its purpose, scope, and intended audience.

General Description: Describes the general factors that affect the Voting Application system and its requirements, such as product perspective, functions, user characteristics, constraints, assumptions, and dependencies.

Specific Requirements: This section, the largest and most important part of the SRS, outlines the detailed requirements for the Voting Application system. It includes subsections for external interface requirements, functional requirements, non-functional requirements, design constraints, and other miscellaneous requirements.

Analysis Models: Lists all analysis models used in developing specific requirements previously given in the SRS. This section includes data flow diagrams (DFDs) and other relevant models.

Appendices: Provides additional information that may be helpful for understanding the Voting Application system. This could include conceptual documents, marketing materials, meeting minutes, or any other relevant documents.

By following this organized structure, the SRS ensures that all stakeholders, including developers, project managers, and clients, have access to comprehensive and well-structured information about the requirements and specifications of the Voting Application system. This facilitates effective communication, planning, and implementation of the project.

2. General Description

The General Description section of the Software Requirements Specification (SRS) provides an overarching view of the Voting Application system, outlining factors that influence its development and functionality without specifying detailed requirements.

2.1 Product Perspective:

The Voting Application system exists as an independent backend software solution designed to facilitate voting processes for various purposes, such as elections, surveys, or decision-making procedures. It operates as a standalone system, interacting with users through APIs and interfaces, without direct integration with other systems. However, it may integrate with frontend applications or data analysis tools for comprehensive functionality.

2.2 Product Functions:

The Voting Application system performs several key functions to enable seamless voting experiences for users. These functions include:

- *User authentication:* Users can sign up and log in using their *Aadhaar Card* Number and password.
- *Candidate management:* Admin users can add, update, or delete candidate information.
- *Voting:* Authenticated users can vote for candidates, with restrictions to prevent multiple votes.
- *Data retrieval:* Users can view the list of candidates and check the total votes received by each candidate.

2.3 User Characteristics:

The eventual users of the Voting Application system include both general users and administrators. User characteristics may vary but typically include:

- *General users:* Individuals eligible to vote in the context of the application, possessing *Aadhaar Card* Numbers and passwords for authentication.
- *Administrators:* Authorized users responsible for managing candidate information and overseeing the voting process.

2.4 General Constraints:

Several general constraints may influence the design and development of the Voting Application system, including:

- *Compliance with data protection regulations:* The system must adhere to relevant data protection laws and regulations to ensure the privacy and security of user data.
- *Scalability:* The system should be designed to accommodate potential increases in user traffic and candidate data without sacrificing performance.
- *Resource limitations:* Constraints on hardware resources, such as server capacity and memory, may impact system scalability and performance.

2.5 Assumptions and Dependencies:

The development and operation of the Voting Application system rely on certain assumptions and dependencies, including:

- *Availability of required technologies:* The system assumes the availability of Node.js, Express.js, MongoDB, Mongoose, and JSON Web Tokens (JWT) for authentication.

- *Compliance with Aadhaar Card Number format:* Users are expected to provide valid *Aadhaar Card* Numbers conforming to the prescribed format.
- *Stable internet connectivity:* The system assumes users have stable internet connectivity to access the application and submit votes.

These factors provide context for understanding the requirements and functionalities outlined in subsequent sections of the SRS, guiding the development and implementation of the Voting Application system.

3. Specific Requirements

This section of the Software Requirements Specification (SRS) outlines the detailed requirements for the Voting Application system. Each requirement is defined to guide the project's software design, implementation, and testing. The requirements are structured to ensure correctness, traceability, clarity, verifiability, prioritization, completeness, consistency, and uniqueness.

3.1 External Interface Requirements

3.1.1 User Interfaces

- The system shall provide a user interface for user registration and authentication.
- The system shall display a list of candidates with their respective details, including name and party affiliation.
- The system shall allow users to vote for a candidate using the user interface.
- The system shall provide feedback to users upon successful voting, indicating the completion of the voting process.

3.1.2 Hardware Interfaces

N/A (As this is a backend application, it does not have direct hardware interfaces.)

3.1.3 Software Interfaces

- The system shall interact with MongoDB database for storing candidate and user information.
- The system shall utilize JSON Web Tokens (JWT) for user authentication and authorization.
- The system shall expose API endpoints for user authentication, candidate management, and voting functionalities.

3.2 Functional Requirements

3.2.1 User Authentication

- The system shall allow users to sign up by providing their Aadhaar Card Number and password.
- The system shall verify the uniqueness of Aadhaar Card Numbers during the signup process.
- The system shall allow users to log in using their Aadhaar Card Number and password.

3.2.2 Candidate Management

- The system shall allow admin users to add new candidates to the system.
- The system shall allow admin users to update existing candidate information, including name and party affiliation.
- The system shall allow admin users to delete candidates from the system.

3.2.3 Voting

- The system shall prevent users from voting multiple times for the same candidate.
- The system shall record each vote cast by a user and update the corresponding candidate's vote count.
- The system shall prevent admin users from voting.

3.5 Non-Functional Requirements

3.5.1 Performance

- The system shall process user requests within a reasonable response time, aiming for a maximum response time of 500 milliseconds.
- The system shall handle concurrent user requests efficiently, supporting a minimum of 1000 simultaneous connections.

3.5.2 Reliability

- The system shall maintain data integrity and consistency, ensuring that all user and candidate information is accurately stored and retrieved.

3.5.3 Availability

- The system shall strive for high availability, aiming for a minimum uptime of 99.9% over a given period.

3.5.4 Security

- The system shall implement proper encryption mechanisms to secure sensitive user data, such as passwords and Aadhaar Card Numbers.
- The system shall enforce user authentication and authorization to prevent unauthorized access to system functionalities.

3.5.5 Maintainability

- The system shall be designed with modularity and encapsulation, facilitating ease of maintenance and future enhancements.
- The system shall adhere to coding standards and documentation practices to support maintainability.

3.5.6 Portability

- The system shall be platform-independent, allowing deployment on various operating systems and environments supporting Node.js and MongoDB.

These specific requirements provide a comprehensive outline of the functionality, interfaces, and non-functional characteristics of the Voting Application system. Each requirement is structured to ensure clarity, completeness, and suitability for guiding the software development process effectively.

4. Analysis Models

In the development of the Voting Application system, several analysis models have been utilized to refine and define specific requirements outlined in the Software Requirements Specification (SRS). These models provide a structured approach to understanding the system's functionality, interfaces, and user interactions. Below are the analysis models employed in the development process:

4.1 Data Flow Diagrams (DFD)

Introduction:

Data Flow Diagrams (DFDs) are graphical representations that depict the flow of data within a system. They illustrate how data moves from external sources through processes to various destinations. DFDs are instrumental in understanding the system's data flow and identifying interactions between different components.

Narrative Description:

In the context of the Voting Application system, DFDs are used to visualize the flow of data related to user authentication, candidate management, and voting processes. The DFDs outline the interactions between users, the system, and the database, illustrating how data moves between these entities.

User Authentication DFD:

This DFD illustrates the flow of data during the user authentication process. It shows how user credentials are inputted by the user, verified by the system, and authenticated against the database. Additionally, it depicts the generation and validation of JSON Web Tokens (JWT) for secure user sessions.

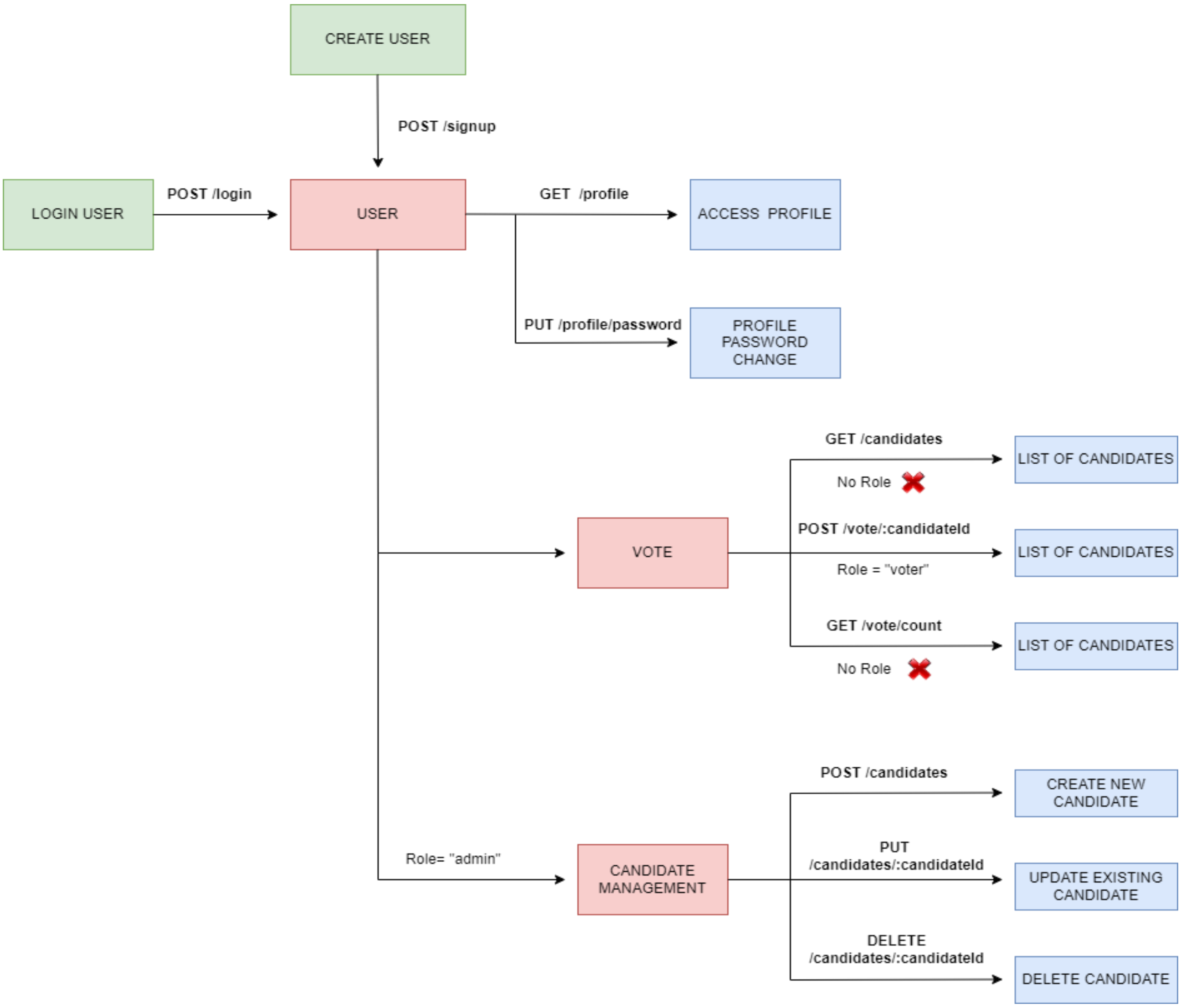
Candidate Management DFD:

This DFD portrays the data flow involved in candidate management functionalities. It outlines how admin users interact with the system to add, update, or delete candidate information. The diagram delineates the processes involved in handling candidate data and updating the database accordingly.

Voting Process DFD:

The Voting Process DFD elucidates the flow of data during the voting process. It illustrates how users cast their votes for specific candidates, how the system verifies user eligibility and prevents multiple voting, and how vote data is recorded and updated in the database.

Traceability to SRS Requirements:
Each DFD is directly traceable to the specific requirements outlined in the SRS. For instance, the User Authentication DFD corresponds to the functional requirements related to user authentication (3.2.1 User Authentication). Similarly, the Candidate Management DFD aligns with requirements concerning candidate management functionalities (3.2.2 Candidate Management). By referencing these DFDs, developers can ensure that the system's design and implementation align with the specified requirements in the SRS.



5. GitHub Link: -

- Available at: https://github.com/ANUJ3DADWAL/Voting-Application-Backend-Only

A.1 Appendix 1: Sample User Interface Mock-ups

Introduction:

Appendix 1 contains sample user interface mock-ups for the Voting Application system. These mock-ups provide visual representations of the system's user interfaces, including login screens, candidate lists, voting interfaces, and admin panels. They serve as a reference for designers and developers to understand the layout, functionality, and user interactions within the application.

Narrative Description:

The appendix includes a series of user interface mock-ups created using design tools such as Adobe XD or Sketch. Each mock-ups is labelled and annotated to explain its purpose and functionality. The mock-ups cover various aspects of the application, including:

Login Screen Mock-up:
This mock-up showcases the login interface where users can enter their Aadhaar Card Number and password to access the system. It includes fields for user input, buttons for login and signup, and error messages for invalid credentials.

Candidate List Mock-up:
The candidate list mockup displays the interface for users to view the list of candidates participating in the election. It presents candidate names, parties, and other relevant details in a structured format for easy navigation.

Voting Interface Mock-up:
This mock-up demonstrates the voting interface where users can select their preferred candidate and cast their votes. It includes options to choose candidates, submit votes, and displays confirmation messages upon successful voting.

Admin Panel Mock-up:
The admin panel mock-up exhibits the interface accessible to administrators for managing candidates, user accounts, and system settings. It provides functionalities such as adding, updating, or deleting candidates, as well as user management capabilities.

Purpose:
The purpose of including these *mock-ups* in the SRS is to provide stakeholders, designers, and developers with a visual representation of the system's user interfaces. These mockups aid in eliciting feedback, validating design decisions, and ensuring alignment with user requirements and expectations.

A.2 Appendix 2: Database Schema

Introduction:
Appendix 2 contains the database schema for the Voting Application system. The schema outlines the structure of the database tables, including entities, attributes, relationships, and constraints. It serves as a reference for database administrators and developers to understand the data model and design effective database queries and optimizations.

Narrative Description:
The appendix presents the database schema in a structured format, typically using SQL Data Definition Language (DDL) statements or visual diagrams. It includes detailed descriptions of each table, including primary keys, foreign keys, data types, and any additional constraints or indexes.

User Table Schema:
This section outlines the schema for the User table, which stores user information such as Aadhaar Card Number, password hashes, roles, and voting status. It defines the primary key, unique constraints, and relationships with other tables.

Candidate Table Schema:
The Candidate table schema describes the structure for storing candidate data, including names, parties, and vote counts. It specifies primary and foreign key relationships to ensure data integrity and consistency.

Vote Table Schema:
This section delineates the schema for the Vote table, which records voting transactions, including user IDs, candidate IDs, and timestamps. It establishes relationships between users and candidates to track voting activities.

Purpose:
The inclusion of the database schema in the SRS aims to provide stakeholders and developers with insights into the system's data architecture and organization. By documenting the database structure, developers can implement efficient data storage solutions and ensure seamless integration with the application's functionalities. Additionally, the schema serves as a foundation for database management and maintenance throughout the system's lifecycle.