Software Requirements Specification

for

Online Voter ID Card Application

Version 1.0 approved

Prepared by VISHNUPRIYAN S

31.01.2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

It is to provide an automated and efficient way for eligible voters to apply for and obtain a voter ID card.

The software is designed to facilitate the registration process, collect personal and biometric information, verify eligibility, generate ID cards, and manage voter data securely

1.2 Document Conventions

Document conventions refer to the guidelines or standards for the documentation of the software development project

Document Format: All documents should follow a specific format, such as IEEE or ISO standards.

Naming Conventions: All documents, including software requirement documents, should have a unique and descriptive name.

Version Control: All documents should be under version control, and changes should be documented and tracked.

Terminology: All documents should use consistent terminology and avoid ambiguity to prevent confusion among team members and stakeholders.

References: All documents should reference other relevant documents or sources, such as laws, regulations, or best practices.

Review and Approval: All documents should undergo a review and approval process to ensure accuracy, completeness, and compliance with the requirements and guidelines

1.3 Intended Audience and Reading Suggestions

Intended Audience: The intended audience for this document includes project stakeholders, including business owners, project sponsors, project managers, and development teams.

Reading Suggestions: The following materials may be useful for reading and understanding the software requirements document:

Voter ID card application guidelines and regulations in the relevant jurisdiction.

Best practices and guidelines for software requirements engineering and management.

Technical specifications and requirements for relevant hardware and software components.

User experience and usability guidelines and best practices.

1.4 Product Scope

It defines what features, functions, and capabilities are included in the product and what is not included. Here are some examples of product scope for an online voter ID card application in software engineering project management:

Inclusions: The product scope includes the following features and functions:

Online registration and application process for eligible voters to apply for a voter ID card.

Authentication and verification of voter information and eligibility

.

Capture and storage of personal and biometric information of registered voters.

Generation and printing of secure voter ID cards.

Payment processing for ID card application fees.

User management and administration functions for managing voter data and system performance.

Exclusions:

The product scope excludes the following features and functions:

Offline registration and application process.

Provision for voting or other electoral processes.

Integration with other systems or services beyond those explicitly defined

1.5 References

Election Commission of India (ECI) website: https://eci.gov.in/

Election Commission of India (ECI) guidelines for online voter ID card application.

The Representation of the People Act, 1950: https://indiacode.nic.in/handle/123456789/2314?view_type=browse&sam_handle=123456789/1362

National Voter's Service Portal (NVSP): https://www.nvsp.in/

2. Overall Description

2.1 Product Perspective

It would include details such as the purpose of the application, its features, and its intended users.

The product description would also outline the scope of the project, the constraints and assumptions that the development team should keep in mind, and any other relevant information that would help stakeholders understand the product.

The product description is an essential part of the SRS document, as it sets the foundation for the project's requirements and serves as a guide for the development team throughout the project's life cycle.

2.2 Product Functions

User registration: allowing users to create accounts and set up their profiles

Voter identification: verifying the identity of registered voters

Voter database management: maintaining a secure and up-to-date database of registered voters

Online application form: providing an online form for users to apply for voter ID cards

Application review and processing: reviewing and processing online applications for voter ID cards

Voter ID card printing and delivery: printing and delivering voter ID cards to registered voters

User support: providing help and support for users who encounter issues while using the software product.

2.3 User Classes and Characteristics

Registered voters: Users who have already registered to vote and need a voter ID card to participate in an election. They would need to be able to apply for a voter ID card, track the status of their application, and receive their voter ID card after approval.

Election officials: Users who are responsible for managing and conducting elections. They would need to be able to access the voter database, verify voter identities, and ensure that the election is conducted fairly and securely.

System administrators: Users who are responsible for managing and maintaining the software product. They would need to be able to configure the system, manage user accounts, and troubleshoot any issues that arise.

2.4 Operating Environment

Hardware requirements: specifying the type and configuration of the computers, servers, and other hardware components required to run the software product, such as minimum RAM, processor speed, and storage capacity.

Software requirements: specifying the operating system, web server, database management system, and other software components required to run the software product, such as the minimum version number of each component.

Network requirements: specifying the network infrastructure required to support the software product, such as the type of network connection, bandwidth, and firewall configurations.

Security requirements: specifying the security measures required to protect the software product and its data, such as encryption, access control, and secure communication protocols.

2.5 Design and Implementation Constraints

Compatibility constraints: specifying the hardware and software compatibility requirements for the software product, such as specific web browsers or database management systems that must be supported.

Technical constraints: specifying the technical limitations that might impact the design or implementation of the software product, such as limitations in processing power or storage capacity.

Legal and regulatory constraints: specifying any legal or regulatory requirements that the software product must comply with, such as data protection laws or accessibility regulations.

Budget and time constraints: specifying any budget or time constraints that might affect the development team's ability to implement certain features or complete the project within the specified timeframe.

2.6 User Documentation

User manuals: providing step-by-step instructions for using the software product, including screenshots and explanations of different features and functionalities.

Help files: providing context-sensitive help within the software product to assist users with specific tasks or questions.

Online documentation: providing an online knowledge base or frequently asked questions (FAQ) section for users to access and search for information.

Video tutorials: providing video-based training materials to help users better understand how to use the software product.

2.7 Assumptions and Dependencies

Third-party systems: assuming that third-party systems or services, such as database management systems, web servers, or email services, will be available and compatible with the software product.

User behavior: assuming that end-users will use the software product in a particular way or that they will have a certain level of technical expertise.

Security requirements: assuming that certain security measures, such as firewalls or encryption, are in place to protect the software product and its data.

Legal or regulatory requirements: assuming that the software product complies with specific legal or regulatory requirements, such as data protection laws or accessibility regulations.

The SRS document would describe each assumption and dependency in detail, including the potential risks or impacts on the software product's performance or functionality if the assumptions or dependencies are not met.

This information would help ensure that the development team and other stakeholders are aware of any external factors that may impact the project's success and can make informed decisions to mitigate any potential risks or issues.

3. External Interface Requirements

3.1 User Interfaces

It includes all the graphical elements, controls, and menus that enable users to input information, view output, and perform tasks within the software.

In External Interface Requirements in software specifications requirements in software engineering project management, user interface requirements may specify the

appearance, functionality, and usability of the UI, as well as how it interacts with other external systems or interfaces. These requirements are essential to ensure that the software meets the needs of its intended users and can be integrated into other systems as required.

3.2 Hardware Interfaces

A hardware interface refers to the physical connections and protocols used to communicate with external hardware devices, such as sensors, cameras, or printers.

In External Interface Requirements in software specifications requirements in software engineering project management, hardware interface requirements may specify the types of hardware devices that the software needs to communicate with, the protocols and data formats to be used for communication, and the specific actions that the software needs to take in response to hardware events or inputs.

These requirements are important for ensuring that the software can interact with external hardware devices in a reliable and predictable manner, and for providing a seamless user experience

3.3 Software Interfaces

A software interface refers to the point of interaction between different software components, subsystems, or modules software interface requirements may specify the protocols, data formats, and methods to be used for communication between different software components, as well as the specific inputs and outputs that each component should expect or produce.

These requirements are important for ensuring that different software components can work together effectively, regardless of their internal implementation details, and for promoting reusability, modularity, and maintainability of the software system as a whole

3.4 Communications Interfaces

A communications interface refers to the means by which two or more software systems, devices, or subsystems exchange information or data.

In External Interface Requirements in software specifications requirements in software engineering project management, communication interface requirements may specify the protocols, data formats, and methods to be used for data exchange between different systems, as well as the specific actions that each system should take in response to different events or inputs.

These requirements are essential for ensuring that different systems can exchange data effectively, securely, and efficiently, regardless of their underlying technologies, architectures, or implementations.

Communication interfaces are particularly important for distributed systems or systems that need to interact with external entities such as databases, web services, or other networked devices

4. System Features

System features refer to the specific functionalities, capabilities, or behaviors that a software system is expected to exhibit in order to satisfy the needs and requirements of its stakeholders.

4.1 System Feature 1

4.1.1 Description and Priority

The "priority" of a system feature refers to its relative importance or urgency compared to other features in the software system.

Priority can be classified as high, medium, or low and is typically used to guide the development process and ensure that critical features are implemented first, while less important features can be deferred to later stages.

4.1.2 Stimulus/Response Sequences

The stimulus can be any kind of input that the software system is designed to handle, such as user actions, external messages, sensor readings, or system events.

Stimulus-response specifications are important for defining the expected behavior of the software system and for guiding the design, development, and testing activities to ensure that the system responds correctly to various inputs and events

4.1.3 Functional Requirements

Functional requirements typically describe what the software system should do in terms of inputs, outputs, and processing or computation steps.

They may include specific functionalities such as data entry, data processing, reporting, and workflow management.

Functional requirements may also specify the expected behavior of the software system under various conditions, such as error handling, user interactions, and system events.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Performance requirements typically specify the response time, throughput, scalability, and resource usage constraints that the software system must satisfy under various conditions, such as user loads, data volumes, or system configurations.

They may also include other factors that affect system performance, such as network latency, hardware limitations, and third-party dependencies

5.2 Safety Requirements

Safety requirements typically specify the measures and controls that the software system must implement to prevent or mitigate potential safety hazards or risks that may arise from its use.

They may include requirements for system reliability, availability, fault tolerance, and error handling, as well as requirements for user safety training, warning messages, and emergency procedures

5.3 Security Requirements

Security requirements typically specify the measures and controls that the software system must implement to protect its assets, data, and users from unauthorized access, use, or disclosure.

They may include requirements for authentication, access control, encryption, data integrity, audit trails, and incident response, as well as requirements for compliance with security standards and regulations

5.4 Software Quality Attributes

Software quality attributes describe the non-functional aspects of the software system, such as its reliability, maintainability, scalability, usability, performance, security, and compatibility.

These attributes are often expressed in terms of specific metrics or measures, such as availability, response time, fault tolerance, and user satisfaction, that the software system must satisfy

5.5 Business Rules

Business rules are a type of non-functional requirement in software requirements specifications in software engineering project management that describe the specific rules, constraints, policies, and procedures that the software system must follow to meet the business needs and objectives of its stakeholders.

Business rules typically specify the logic, calculations, validation, and decision-making processes that the software system must implement to ensure compliance with the business rules and regulations of the organization.

They may include requirements for data validation, data integrity, workflow management, exception handling, and reporting, as well as requirements for compliance with industry standards and best practices.

Business rules are important for ensuring that the software system supports the business processes and goals of the organization and meets the expectations of its stakeholders. They are typically documented in the business requirements specification and used as a basis for system design, development, and testing.

6. Other Requirements

Other types of non-functional requirements in software requirements specifications in software engineering project management may include:

Usability requirements: These specify the ease of use and user experience of the software system, such as its navigation, responsiveness, learnability, and accessibility.

Maintainability requirements: These specify the ease and cost-effectiveness of maintaining and modifying the software system, such as its code maintainability, testability, and reusability.

Compatibility requirements: These specify the ability of the software system to integrate with other systems and platforms, such as its interoperability, data exchange, and migration capabilities.

Legal and regulatory requirements: These specify the legal and regulatory compliance requirements that the software system must meet, such as data privacy, security, and intellectual property laws.

Environmental requirements: These specify the environmental constraints and considerations that the software system must account for, such as power consumption, noise, and temperature.

Appendix A: Glossary

Appendix A is typically a section in the document that contains a glossary of terms and definitions used in the document.

The glossary provides a common vocabulary and understanding of the terms and concepts used throughout the document, which can help to avoid misunderstandings and confusion among stakeholders.

The glossary in Appendix A may include definitions of technical terms, acronyms, and abbreviations used in the document, as well as business terms, industry jargon, and other domain-specific terminology.

The glossary may also include references to external standards, regulations, or best practices that are relevant to the software system or the project.

Including a glossary in Appendix A can improve the clarity and consistency of the document and facilitate communication and collaboration among the project team and stakeholders.

Appendix B: Analysis Models

Appendix C: To Be Determined List