Software Requirements Specification

for

E-Voting System

**Version 1.0 approved**

**Prepared by:**

**Yash Dodeja …..Bharath R…...Divyam Mehta**

**National Institute of Technology Karnataka**

**17th January 2019**

**Table of Contents**

1. Introduction 3
   1. Purpose 3
   2. Document Conventions 3
   3. Intended Audience and Reading Suggestions 3
   4. Product Scope 3
   5. References 4
2. Overall Description 4
   1. Product Perspective 4
   2. Product Functions 4
   3. Operating Environment 5
   4. Design and Implementation Constraints 5
   5. User Documentation 5
   6. Assumptions and Dependencies 5
3. External Interface Requirements 5
   1. Hardware Interfaces 8
   2. Software Interfaces 8
   3. Communications Interfaces 8
4. System Features 8
   1. System Feature 1 8
   2. System Feature 2 (and so on) 9
5. Other Nonfunctional Requirements 9
   1. Performance Requirements 9
   2. Safety Requirements 9
   3. Security Requirements 10
   4. Software Quality Attributes 10
   5. Business Rules 10
6. Other Requirements 10

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Bharath | 3rd Feb | Updated Use Cases ,Security and Safety features | 2.0 |
|  |  |  |  |

# Introduction

## Purpose

*The purpose of this document is to make the functional and non-functional requirements of the Online National Election Voting System easy to comprehend. It also serves the purpose of making the functionality clear to end users.*

## Document Conventions

## Intended Audience and Reading Suggestions

*The intended audience of this document is the potential end user. The document may also*

*serve as a reference guide to the developers of the system.*

## Product Scope

*This SRS document applies to the initial version (release 1.0) of the “Online National Election System” software package. This document describes the modeling and the requirement analysis of the system. The main aim of the system is to provide a set of protocols that allow voters to cast ballots while a group of authorities collect votes and output final results.*

## References

*<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>*

# Overall Description

## Product Perspective

*The software product is a standalone system and not apart of a larger system. The system*

*will be made up of two parts, one running visible directly to the administrator on the server*

*machine and the other visible to the end users, in this case the voters, through web pages.*

*The two users of the system, namely the voters and the election authority(EA) interact*

*with the system in different ways. The election authority configures the whole system*

*according to its needs on the server where the system is running.*

*The voters cast their votes using the web interface provided. These votes are accepted by*

*the system on the server.*

## Product Functions

*On the EA side, the system can be used to create/update/delete the election details (*

*posts, candidates, electoral rolls etc ). The EA should be able to specify the different*

*attributes it wants for posts/candidates of a particular election instance and voters. For*

*example, one EA may want the candidate’s photograph as an attribute, where as another*

*EA may not find it necessary. Similarly, they may want one set of attributes for voters*

*in one setting and a different one in another. For Example, in a university, the EA might*

*be happy with just the roll numbers of each voter while an election in an association may*

*require voters’ name, phone number, address etc. After the election is set up, passwords*

*must be generated and mailed to voters on request.*

*The system should also be able to run seperate election instances at the same time.*

*From the voters perspective, the system is used to help them cast their votes and after the*

*elections are over, allow them to view the results, which are automatically posted on the*

*same site after the election duration is over.*

**User Classes and Characteristics**

*The users can be divided into two main classes:*

*– The EA :It’s primary objective is to conduct fair and hassle-free elections. The EA has*

*to be a neutral party and should not have any gain/loss from the election results. The*

*EA invites potential candidates to file their nomination for certain posts depending on*

*certain constraints. As explained earlier, the EA decides the classes of voters eligible*

*to vote for a certain post. They should have adequate experience of using a computer*

*to be able to configure the election properly.*

*– The Voters : The voters should have a basic knowledge of how to use a web browser*

*and navigate through web pages. The voters should be aware that they have to keep*

*their user-id and password confidential.*

## Operating Environment

*The server should have Java installed on the machine, along with Java’s cryptographic*

*packages. The election server runs on a http server, that is ”jsp” enabled. The browsers*

*through which the voters access the server should have minimal support for cookies and*

*encrypted transactions.*

## Design and Implementation Constraints

*Even though the system enables voters to poll their vote from any terminal connected*

*to the Internet, the voters should initially contact the election administrator’s office to*

*authenticate themselves and establish their user-ids. This constraint is imposed to ensure*

*that only the genuine person is allowed to vote in the elections.*

*Also, it is assumed that only the EA has access to the server that hosts the election.*

## User Documentation

*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

## Assumptions and Dependencies

*User side assumptions and dependencies*

*– PC (Personal Computer) or workstation with GUI.*

*– A web browser with support for cookies.*

*– Working Internet connection.*

*Server side assumptions and dependencies*

*– A web server with GUI, Java and an http server installed .*

# External Interface Requirements

***3.1 User Interfaces***

*We have given all the use cases that are there for the system to specify the user interface.*

*They are given below:*

***3.1.1 Use case 0 : Welcome screen for the Administrator***

*Main success scenario:*

*1. The EA is asked to login using his administrator password.*

*2. The EA is provided the option of working with voters database or the election in-*

*stances.*

*3. The EA chooses to work with election instances. Refer to use case 4.Extensions*

*1a. The EA enters wrong user-id or password. He is asked to re-enter the password.*

*2a. The EA chooses to work with voters’ database. Refer to Use case 1.*

***3.1.2 Use case 1 : Creating the Voters Database***

*Main success scenario:*

*1. On start-up, the EA is presented with a screen (called the voter start screen hence-*

*forth) where he can choose to either create, modify or delete the voters database.*

*2. The EA chooses ’create’.*

*3. The system takes EA to a new screen and asks the EA to give a name for the new*

*voter class and asks the EA to provide some identification information for the class.*

*4. The system provides a field where the EA can add a roll number and his attributes*

*by clicking add, in addition to a facility to load entries from a file.*

*5. The EA confirms the entries it has made. The system registers the entry and presents*

*the EA with welcome screens of Use case 0.*

***3.1.3 Use case 2 : Modify the voters databases***

*Main success scenario:*

*1. The system presents the EA with the search screen. The EA selects the attributes he*

*wants to search for and specifies their value.*

*2. The system presents the EA with the results. The EA chooses the voter(s) to modify.*

*3. The EA either modifies the details or deletes the voter(s).*

*4. The system asks the EA to confirm the changes. The EA confirms.*

*5. The system presents the EA with the search screen of step 1.*

*Extensions:*

*4a. The EA cancels without confirming. The system does nothing and takes the EA*

*back to the search screen of step 1.*

***3.1.4 Use case 3 : Delete the Voters’ database***

*Main success scenario:*

*1. The system asks the EA to confirm his wish to delete the voters’ database for ever.*

*2. The EA confirms it.*

*3. The database is deleted.*

*Extensions*

*2a. The EA cancels without confirming the deletion. The system does not perform*

*any action.*

***3.1.5 Use case 4 : Creating the Election Instance***

*Main success scenario:*

*1. The EA is presented with a screen (called the start screen henceforth) where he can*

*create, modify or delete an election instance.*

*2. The EA chooses to create an election.*

*3. The system asks the EA whether a candidate can stand for more than one posts.*

*4. The system asks the EA the number of posts. For each of the posts, steps 6-8 are*

*repeated.*

*5. For a post, the system asks the EA the designation and other required attributes of*

*the candidates competing for that post.*

*6. For the post, the system asks the EA for the class of voters that are allowed to vote*

*for the post.*

*7. For the post, the EA specifies the eligible nominated candidates. The system asks the*

*EA to fill in the required attributes.*

*8. The system asks the EA for the start and finish time of the Election. The EA provides*

*the necessary details.*

*9. The system asks the EA to confirm his election instance.*

*10. The system creates an election instance.*

***3.1.6 Use case 5 : Modify the Posts***

*Main success scenario:*

*1. The system presents the user with the update options, namely Posts,Candidates,Voters*

*and time of election.*

*2. The EA chooses to update ’posts’.*

*3. The system presents the EA with the list of all posts and an option to add a post.*

*4. The EA chooses the post it wants to modify.*

*5. The EA either modifies the details or deletes the post.*

*6. The system presents the EA with the screen of step 4.*

***3.1.7 Use case 6 : Modify the Candidates***

*Main success scenario:*

*1. The system presents the user with the update options, namely Posts,Candidates,Voters*

*and time of election.*

*2. The EA chooses to update ’candidates’.*

*3. The system presents the EA with the list of all posts.*

*4. The EA chooses the post it wants to modify the candidates of.*

*5. The system presents the EA with the list of all candidates in that post and an option*

*to add a candidate.*

*6. The EA chooses the candidate to modify.*

*7. The EA either modifies the details or deletes the candidate.*

*8. The system presents the EA with the screen of step 4.*

***3.1.8 Use case 7 : Modify the Election Time***

*Main success scenario:*

*1. The system presents the current election time and asks the EA to reset and confirm.*

*2. The system presents the EA with the screen of step 4.*

***3.1.9 Use case 8 : Deletion of the election instance***

*Main success scenario:*

*1. The EA chooses to delete the election instance.*

*2. The system asks the EA to confirm the request.*

*3. The EA confirms the request.*

*4. The system deletes all the informations about the election instance and is now ready*

*for configuration of a new election instance.*

***3.1.10 Use case 9 : The Voting on the Voter’s end***

*Main success scenario:*

*1. The voter is asked to login using the user-id and password provided to him earlier.*

*Steps 2 and 3 are repeated for all the posts that this voter is available to vote for.*

*2. The system presents the voter with successive screens for voting for each of the posts.*

*3. The voter selects one of the candidates and submits his choice to the system.*

*4. The system presents the voter with the final choices of the candidates for each post*

*made by the voter for confirmation.*

*5. The voter confirms the choices.*

*6. The system registers the choices made by the voter.*

*7. The voter logs out..*

## Hardware Interfaces

*There are no hardware interfaces to this software system. The only interfaces are through*

*a computer system.*

## Software Interfaces

*The poll server runs on http server that is enabled to handle server pages (eg. Apache*

*Tomcat for support jsp). It uses a relational database to keep track of the polls, which*

*it connects through standard database connectivity interfaces. In order to run the setup*

*software, the environment needs to have a JVM running on it.*

## Communications Interfaces

This service requires communication through a webform to authenticate users. The form is accessed only once by every user and hence there are no specific guidelines for message formatting. This is just to gather and authenticate user data. The communications standard expected of a polling app is HTTPS, and we intend to stick to that. We have not faced any encryption issues as of yet.

# System Features

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

## System Feature 1

*<Don’t really say “System Feature 1.” State the feature name in just a few words.>*

4.1.1 Description and Priority

*<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>*

4.1.2 Stimulus/Response Sequences

*<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>*

4.1.3 Functional Requirements

*<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

The performance requirements for the system aren’t high for a polling app like these. However, the system must maintain a secure connection throughout and should not be vulnerable to malware attacks. System crashes should be minimised to zero and the response time and feedback should be given promptly. Timing should be minimised, but at the same time, the flow between views should be smooth, and users get an option to verify their choices. The application does not encompass many distinguished functions that might have separate requirements.

## Safety Requirements

*1.In order to prevent data loss in case of system failure, the result of votes that*

*were polled till then have to be saved in the database, for the system to resume*

*the counting process on reboot.*

*2. The EA should set up his system time appropriately for the election process to*

*start at the correct time.*

*3. In case the EA detects any security lapse in the system, he should able to shut*

*down the server and close all connections immediately while preserving the already*

*polled votes.*

*4. The system should be capable of gracefully recovering from earlier crashes and*

*continuing the voting process.*

## Security Requirements

*1.The system should provide basic security features like password authentication*

*and encrypted transactions.*

*2. All the passwords generated and communicated to the users should be stored in*

*the server only in an encrypted form for login management to prevent misuse.*

*3. Serial attacks should be avoided by maintaining a minimum time gap between*

*successive invalid log-in attempts.*

*4. Additional security features like voter anonymity and threshold schemes for mul-*

*tiple EAs might provided later on as an add-on feature to the software.*

## Software Quality Attributes

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

## Business Rules

*<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>*

# Other Requirements

*<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>*

**Appendix A: Glossary**

*<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>*

**Appendix B: Analysis Models**

*<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams*.>

**Appendix C: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*