Software Requirements Specification

for

Election Voting System

Version 1.4 approved

Prepared by Jacynda Alatoma, Naren Nandyal, Long Nguyen, Ananya Vegesna

University of Minnesota

February 16, 2023

Table of Contents

Table of Contents ii
Revision History iii
1. Introduction 1
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References 1
2. Overall Description 1
2.1 Product Perspective 1
2.2 Product Functions 2
2.3 User Classes and Characteristics 2
2.4 Operating Environment 2
2.5 Design and Implementation Constraints 2
2.6 User Documentation 2
2.7 Assumptions and Dependencies 2
3. External Interface Requirements 3
3.1 User Interfaces 3
3.2 Hardware Interfaces 3
3.3 Software Interfaces 3
3.4 Communications Interfaces 3
4. System Features 3
4.1 Creating Audit File for Election 3
4.2 Processing Header of CSV File 3
4.3 Displaying Winner(s) of an Election 4
4.4 Tiebreaker 4
4.5 Naming the Audit File 4
4.6 Getting Additional Information from the User 4
4.7 Identifying the CSV File 4
4.8 Processing Ballots in the CSV File 4
4.9 Running a Closed List Party Election 4
4.10 Running Plurality Voting/IRV 5
5. Other Nonfunctional Requirements 5
5.1 Performance Requirements 5
5.2 Safety Requirements 5
5.3 Security Requirements 5
5.4 Software Quality Attributes 5
5.5 Business Rules 5
Appendix A: Glossary 5

Revision History

Name	**Date**	**Reason For Changes**	**Version**
Naren Nandyal	02/9/23	Adding to the document	1.0
Jacynda Alatoma	02/12/23	Adding more info. to parts of the document	1.1
Ananya Vegesna	02/14/23	Adding more information to parts of the document	1.2
Long Nguyen	02/14/23	Adding to document	1.3
Everyone	02/15/23	Final touch-ups	1.4

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description of our Voting System software. This is the approved revision of Version 1.4. It will explain the purposes and features of the whole system, what the system will do, and the constraints under which the system operates. This document is intended for users (i.e. election officials) of the system and potential developers.

1.2 Document Conventions

This document was created based on the IEEE template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

The primary users would be election officials, who would use this program to run an election using an Instant Runoff format and/or a Closed Party List format. Another user of this program may be media personnel, who can obtain the election results and share them with the rest of the community. Finally, developers may use this program in order to develop it further and solve any bugs that there may be.

1.4 Product Scope

This system will be able to run instant runoff and closed-party list voting. It will be able to determine a winner and break ties between candidates or parties. This system will save the organization time and money by getting rid of the need to count ballots by hand and determine a winner by hand. Fewer employees will be needed to run this system than if they were to count by hand.

1.5 References

IEEE Template for System Requirement Specification Documents:
IEEE Software Requirements Specification Template

2. Overall Description

2.1 Product Perspective

This system was developed for easier closed-party list voting and instant runoff voting.
It is a new product that will allow for easier running and management of elections. It allows for election votes to be counted unbiased and for it to be far more efficient than votes being hand counted. The system will be set up to handle CSV files and will output an audit file when finished running.

2.2 Product Functions

- Run both Instant Runoff and Party List (using closed party list) Voting systems and determine a winner
- Produce an audit file containing all important information about the election
- Break ties in the elections when they occur
- Display results of election

2.3 User Classes and Characteristics

Election Officials: Most Important users, will use the product to generate results from elections and check for election fraud.
Programmers: Will work on developing projects by debugging or adding new features.
Testers: They will work on test cases and find any bugs.
Media Personnel: Will use the product to announce information to the general public and news sources
Voters: The ballots of each voter will be read by this product in order to properly count all the votes and determine a winner.

2.4 Operating Environment

We will be developing in C++ (Version 9.4.0) and running in a Linux (Ubuntu Version 20.04.5) environment. The software will be run through the University of Minnesota CSE lab machines.

2.5 Design and Implementation Constraints

This Voting System is developed in C++. The program must be able to process 100,000 ballots in under 4 minutes. The product can only read-in comma separated value (CSV) files. Election files must be found in the same directory as the program in order to be executed. The system needs to generate an audit file at the end of each election.

2.6 User Documentation

All user-level interactions and interfaces will be handled by outside parties, this SRS document will serve as the only documentation.

2.7 Assumptions and Dependencies

The program must be given only one CSV file per election with all the votes on it. The ballots will have no errors. The election that is run on this program will have no write-in candidates. The program is developed in C++ and will run from the command prompt.

3. External Interface Requirements

3.1 User Interfaces

The primary interaction with the user interface is if extra information is needed by the user. If this occurs, a text input will appear for the user to enter that information. In addition to this part of the user interface, when a winner is selected, they are displayed to the user on the screen after the election has occurred and all votes have been counted.

3.2 Hardware Interfaces

The minimum hardware requirements of this program will be an Intel CPU with a Quad-core 2GHZ or higher and an AMD CPU with a Quad-core 3GHZ or higher. The recommended amount of RAM is 8 GB. The CSE lab machines that this program will be expected to run on have an Intel Core i7 at 2.5 GHz and 32 GB of RAM.

3.3 Software Interfaces

This program requires C++ (Version 9.4.0) to be installed on the system. This program also requires the handling of CSV files.

3.4 Communications Interfaces

This program requires a CSV file prior to the program being run. There is a possibility that compatibility with Excel files is needed.

4. System Features

This section demonstrates this program's features and functionality. More detailed information on all the use cases/system features can be found in the additional file named: UseCases_Team3.pdf.

4.1 Creating Audit File for Election

4.1.1 This will be one of the most important features. This audit file will include the winner, what type of election it was, the number of candidates, the names of candidates, calculations, the number of ballots, and the number of votes each candidate received. The benefits of creating this audit file are that it saves searching time and documents the order of steps if there are any discrepancies in vote counting. Election officials will use this file.

4.2 Processing Header of CSV File

4.2.1 The program reads in the header of the CSV file with party names and type of election and runs the correct vote count.

4.3 Displaying Winner(s) of an Election

4.3.1 At the end of an election this program will display the winners of an election and include information such as the winner(s), type of election, number of ballots cast, other candidates, etc. The benefits of this feature are that there is easy communication of who won an election as well as transparency in vote counts.

4.4 Tiebreaker

4.4.1 When there is a tie between multiple candidates and parties in the election, a winner will be randomly selected. This is more efficient than being done by hand. This function will only be used when there is a tie. The tiebreaker will consist of a fair coin toss.

4.5 Naming the Audit File

4.5.1 After the program is run and the audit file is created it will be named using the type of election and the date and saved as a .txt file into the same directory as the original CSV file. The benefits of this function are that the votes are accurately counted and an electronic copy of the results is saved in an easy-to-find location.

4.6 Getting Additional Information from the User

4.6.1 When the program is run, there may be additional information that is needed from the user that is not in the input file. In this case, the program will prompt the user for this information. This saves time by having to rely less on total file input, which can be very time-consuming.

4.7 Identifying the CSV File (File Input)

4.7.1 When the program is run the system will prompt for a CSV file with the ballots either using a command line argument or asking the user for the name.

4.8 Processing Ballots in the CSV File

4.8.1 The program reads the number of votes for each candidate that is in the CSV file and determines a winner based on the counts.

4.9 Running a Closed List Party Election

4.9.1 The program will run an election using a party list system. This means that political seats will be split based on votes for each party. For instance, if 40% of votes are for democrats, 40% of the seats would go toward democrats.

4.10 Running Plurality Voting/IRV

4.10.1 Instant runoff voting is when voters vote for candidates in the order of their preference. This means voters can rank the candidates (however, they do not have to rank everyone if they do not want to). Votes are counted in the order of the voter's preferences. Unless a candidate were to receive a clear majority (over 50%), the candidate with the lowest number of votes gets dropped and the voters who voted for that candidate would get their 2nd preference counted and so on.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

This program requires a system with at least a 2GHZ Intel CPU or a 3 GHZ AMD CPU with a minimum of 8 GB of RAM.

5.2 Safety Requirements

The CSV file that is input into the program has to be read-only to prevent any tampering with the file before it is processed. The audit file that is created will be a text file and also be read-only to prevent any fraud.

5.3 Security Requirements

Only election officials can look at the CSV files/ballots when running the system. No outside people should be able to access the program and be able to edit any of the information in the program and/or the files. Testers and programmers can access the program for debugging purposes.

5.4 Software Quality Attributes

This program provides users with simple features and can be used by both experts and typical users. It has an easy-to-use interface and no prior knowledge is required before use.

5.5 Business Rules

The program will be used by election officials for easy and efficient vote counting. The election officials will input the CSV file and the program will output the results of the election.

Appendix A: Glossary

Instant Runoff Voting (IRV) - This is a type of ranked preferential voting. Voters vote for candidates in the order of their preference. Votes are counted in the order of the voter's preferences. Unless a candidate were to receive a clear majority (over 50%), the candidate with the lowest number of votes gets dropped and the voters who voted for that candidate would get their 2nd preference counted, and so on.

Closed Party Line voting (CPL) - Political seats will be split on votes towards each party. For instance, if 40% of votes are for democrats, 40% of the seats would go toward democrats.