Software Requirements Specification

for

Voting System

Version 1.0 approved

Prepared by Team 4

Annabelle Coler - Coler018, Zoe Sepersky - Seper011, Anwesha Samaddar - Samad037, Hilton Nguyen - Nguy4798

CSCI 5801 - Software Engineering I, Spring 2025

University of Minnesota

February 21, 2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This software requirement specification (SRS) document provides a detailed description of the new and standalone voting system. It explains the system's purpose and features, interfaces, what the system will do, how it reacts to external stimuli, the functionalities of Single Transferable Voting (STV) and Plurality voting algorithms, and the constraints in which the system must operate. This document is meant for the stakeholders and the system development team.

1.2 Document Conventions

This document follows the conventions of the IEEE Software Requirement Specifications template.

1.3 Intended Audience and Reading Suggestions

This document is meant for the project stakeholders, developers, testers, and other users. The next section, the overall description of the product, gives an informal description of the requirements and is used as context for the technical requirements in the section following. Thus, it is best to be read by the project managers and stakeholders. Section 4, System Features, is meant for developers and describes the product's requirements in technical detail.

1.4 Product Scope

This software system will be a Vote Counting and Election System, to count ballots and run elections using either Single Transferable Voting (STV) or Plurality. The system will be designed to process and count ballots, then select the winner(s) of an election based on said ballots. Ballots will be counted by using either STV or Plurality, which shall be chosen by the Election Official when the system is run. The system shall facilitate election ballot counting and shall facilitate selecting winners. The system shall take in an input comma-separated value (CSV) file indicated by the election official, process the ballots and count them according to the algorithm chosen, and output the election results via an audit file for the election official.

1.5 References

1. The description of the project is referred to here at the following link: https://canvas.umn.edu/courses/483118/files/50177539?module_item_id=13993087.
2. This document also refers to the IEEE Software Requirements Specification document. "IEEE Guide for Software Requirements Specifications," in IEEE Std 830-1984, vol., no., pp. 1–26, 10 Feb. 1984, doi: 10.1109/IEEESTD.1984.119205.

2. Overall Description

2.1 Product Perspective

There are numerous possible voting methods, with plurality voting being preferred in the United States. However, research on voting theory has shown that ranked-choice voting may be a better method than plurality. Algorithms like the Hare quota or Droop quota are part of the STV family of algorithms. The voting system specified in this SRS will be capable of performing both plurality voting and an STV system using the Droop quota. The program user will specify which algorithm should be used.

2.2 Product Functions

- The user inputs the CSV file name containing the election ballots.
- The user inputs the number of seats up for election and the voting algorithm.
- The system has a simple user interface that can take in file, seat, and algorithm information.
- The system generates an election audit as a text file after the election is complete.
- The user can name the election audit file.
- The system displays the details of the election on the screen once the election is complete.
- The user can run a test file.
- The user can toggle the shuffle option.

2.3 User Classes and Characteristics

The two user classes that will use this product are election officials and testers. Election officials must be U.S. citizens who are both residents of and registered to vote in the state in which they are working as election officials. They must also be 18 years of age or older and have passed the required training program. Testers are individuals who have technical expertise and government authorization to test election systems. The system must satisfy the needs of both user classes.

2.4 Operating Environment

The software will be able to operate in any Linux environment. It will be run through the command prompt.

2.5 Design and Implementation Constraints

- The system must be coded in C++.
- The STV option must be implemented using the Droop quota.
- Only one file can be uploaded per election.
- The program will be run from the command prompt.
- An election should be able to be processed in less than 5 minutes.

2.6 User Documentation

N/A

2.7 Assumptions and Dependencies

- The system will be operated in the U.S. on behalf of the U.S. government.
- There will be no mistakes in the CSV file containing the ballots.
- The file containing the ballots will be stored in the same directory as program files.
- The system does not need to handle security issues.
- No more than 100,000 ballots will be in a file per election.

3. External Interface Requirements

3.1 User Interfaces

The main operational interface of the voting system will be a graphical user interface (GUI). The GUI will be geared towards being user-friendly while also requiring minimal effort to train. It will allow users to interact with the system and respond to system prompts, as well as guide users through the election setup, processing of votes, and displaying of results.

When the program starts, it prompts the user for the number of seats to be filled and an algorithm selection (plurality or STV). It also prompts the user for the CSV filename containing the ballots and the filename in which to store the output election audit. Additionally, users should be able to run a test file with the option to toggle between shuffle-on and shuffle-off using an available command line.

Users will be able to input values in the form field within the GUI, with input validation applied to ensure proper formatting. If a user enters an incorrect input, an error will be displayed asking the user to fix their mistake and re-enter their input.

Buttons and Functionality for GUI:

- Load ballots: browse and select a ballot (a CSV file) within your files
- Start Election: initiate the voting process
- View results: display the election results
- Save report: save the audit file
- Help: display a guide screen on using the system
- Exit/Minimize: exit the application or minimize it

3.2 Hardware Interfaces

It is crucial that the software be able to operate on a University of Minnesota CSE Labs' machine. The software should not be able to interact with any other hardware beyond reading CSV files from the local user's directory. This application does not require any input devices or external hardware devices. The storage requirements for the system are that it should be able to hold 10,000 ballot files and be able to generate audit reports for those election processes.

3.3 Software Interfaces

The CSE Labs' machine will run on the Linux/Ubuntu operating system. The voting system will be developed in C++. To run the environment in C++, the following libraries will be required: <iostream>,

<map>,<random>, <Qt>(a GUI library), <vector>, and <fstream> . A CSV file containing the ballots will be the input and the election audit textfile will be the output. Election results will also be displayed on the screen.

3.4 Communications Interfaces

There will be no need for network communications or email interaction because this system will be running on a local CSE Labs' machine. However, the ballots will be transferred securely through encryption verification before being processed.

4. System Features

4.1 Input CSV File Name - High Priority

Name	Input CSV file name
ID	UC_001
Description	The user is prompted to input the file name that contains the ballots. The name should refer to a CSV file.
Actors	Election official, tester
Organizational Benefits	Adds the functionality to get information from the user for processing ballots.
Frequency of Use	Used every time there is an election and the program is run.
Triggers	An election official or tester starts the program.
Preconditions	No precondition.
Postconditions	The input file will be read in by the file handler.
Main Course	1. System prompts the user to enter the name of the CSV file containing the ballots. 2. User provides the name of the CSV file. 3. File handler reads in the input file.
Alternate Courses	AC_1: User shuts down the voting system program at any time. 1. User restarts the system. 2. Return user to main course step 1. AC_2: User runs the test file 1. Refer to UC_008

Exceptions	EX_1 User enters a non-CSV formatted file. 1. System rejects the input and informs the user they did not enter a CSV file. 2. Prompt the user if they want to try again or exit. 3. The system then prompts the user to enter a filename. If the user makes another mistake, the prompt repeats two more times before automatically exiting. EX_2 User enters an invalid input (multiple filenames, nonexistent files, wrong directory, wrong permissions, etc.). 1. System rejects the input and informs the user they entered invalid input. 2. Prompt the user if they want to try again or exit. 3. The system then prompts the user to enter a filename. If the user makes another mistake, the prompt repeats two more times before automatically exiting.

4.2 Input Number of Seats - High Priority

Name	Input number of seats
ID	UC_002
Description	The user is prompted to input the number of seats that are to be filled.
Actors	Election official, tester
Organizational Benefits	Allows for Droop quota computation
Frequency of Use	Used every time there is an election and the program is run.
Triggers	Election official or tester starts the program.
Preconditions	UC_001 has finished and there is a valid input file.
Postconditions	The number of seats will be known, and the Droop quota can now be calculated.
Main Course	1. The system prompts the user for the number of seats that need to be filled. 2. The system saves that number for Droop quota calculation and for general election information.

Alternate Courses	There are no alternate courses.
Exceptions	EX_1 User enters an invalid input (negatives, 0, decimal points) 1. The system rejects the input and informs the user they have input an invalid number. 2. The system prompts the user if they would like to try again or if they would like to exit. 3. If the user decides to try again, re-prompts the user to enter the number of seats. If the user makes another mistake, the system prompts the user two more times before exiting.

4.3 Select Voting Algorithm - High Priority

Name	User selects voting algorithm
ID	UC_003
Description	The user is prompted to enter the method of election, either STV or Plurality.
Actors	Election official, tester
Organizational Benefits	Allows the user to choose between two voting algorithms.
Frequency of Use	Used every time there is an election and the program is run.
Triggers	Election official or tester starts the program.
Preconditions	UC_001 has finished and there is a valid input file.
Postconditions	The correct voting algorithm for the given election has been chosen.
Main Course	1. The system prompts the user for what voting algorithm the election shall use. 2. The user inputs the desired algorithm. 3. If STV is chosen, refer to UC_004 4. If plurality is chosen, refer to UC_005
Alternate Courses	There are no alternate courses.
Exceptions	EX_1 User enters an invalid algorithm type. 1. The system rejects the input and informs the user they have entered an invalid input. 2. The system prompts the user if they would like to try again or if they would like to exit.

	3. If the user decides to try again, the system re-prompts the user to enter the desired voting algorithm. If the user makes another mistake, the system will prompt the user two more times before exiting.

4.4 STV - High Priority

Name	Run Single Transferable Vote (STV) algorithm
ID	UC_004
Description	The user runs a STV algorithm for a given election.
Actors	Election officials, testers
Organizational Benefits	Allows for ranked-choice voting
Frequency of Use	Used every time there is an election and the STV algorithm is chosen to be used for voting.
Triggers	User selects STV as the voting algorithm when prompted by the system.
Preconditions	UC_001 has completed successfully and the system has extracted the number of candidates and names of the candidates from the CSV file. The system has extracted the number of seats to be filled from UC_002. The necessary information to determine the Droop quota has been extracted.
Postconditions	The election votes are counted, and the list of winning and losing candidates have been compiled. The audit report can now be generated.
Main Course	1. The ballots are read in from the input file. 2. The Droop quota is determined from the number of ballots and the number of seats. 3. First choice votes are counted, and any candidates meeting or exceeding the quota are elected, and those ballots are discarded. 4. Surplus votes are redistributed to voter's next preferred candidates. 5. The candidate with the fewest votes is eliminated and their votes redistributed to the voter's next preferred candidate. 6. Repeat steps 4 and 5 until all seats are filled. 7. The two lists of candidates, elected and non-elected, are generated by the system and provided to the user.
Alternate Courses	AC_1: The shuffle functionality is toggled off for testing

	1. Refer to UC_006
Exceptions	EX_1 System was not able to successfully read in the ballots from the input file and was unable to complete the election. 1. System displays a notification that it was unable to complete the election. 2. System gracefully exits the program.

4.5 Plurality - High Priority

Name	Run Plurality voting algorithm
ID	UC_005
Description	The user runs the Plurality voting algorithm for a given election.
Actors	Election officials, tester
Organizational Benefits	Adds the functionality for determining election results using the Plurality voting algorithm.
Frequency of Use	Used every time there is an election, and the user chooses the Plurality voting algorithm.
Triggers	User selects the Plurality voting algorithm when prompted by the system.
Preconditions	UC_001 is complete, and the system has received the following required voting information from the user: 1. Names of candidates 2. Voter's ballot 3. # seats to be filled 4. User selected - Plurality election type
Postconditions	Election is run, and a list of winners and losers is generated. Results can now be displayed.
Main Course	1. Each ballot will be assigned to a candidate. 2. System counts votes for each candidate. 3. The candidate(s) with the greatest number of votes will be declared the winner(s). 4. If there is a tie for a given seat, the winner will be randomly selected by the system.
Alternate Courses	There are no alternate courses.

Exceptions	EX_1 System was not able to read in the ballots from the input file and was unable to complete the election. 3. System displays a notification that it was unable to complete the election. 4. System gracefully exits the program.

4.6 Toggle Shuffle - Medium Priority

Name	Toggle shuffle option for STV
ID	UC_006
Description	Allows the user to turn off ballot shuffle for testing purposes of the STV election algorithm.
Actors	Election officials, tester
Organizational Benefits	Turning off ballot shuffle will enable the system to generate repeatable election results whenever an election is run with the same voting information. This will help ensure the system is calibrated properly and allow for unit testing.
Frequency of Use	Whenever testing or calibration is required for STV.
Triggers	When the user passes in the appropriate command line flag for testing while running STV.
Preconditions	User has completed UC_001 and decides to run the STV election with ballot shuffle - turned off.
Postconditions	Ballots will be processed with shuffling turned off.
Main Course	1. User enters the correct command-line argument to turn off the ballot shuffle. 2. System recognizes the flag and disables shuffling. 3. System runs the STV algorithm and processes ballots in the given order.
Alternate Courses	There are no alternate courses.
Exceptions	EX_1 User enters an invalid argument. 1. System fails to recognize the entered flag and informs the user of available flag options. 2. The system prompts the user if they would like to try again or exit. 3. If the user decides to try again, re-prompts the user to enter a flag. If the user makes another mistake, the system prompts

	the user two more times before exiting.

4.7 Display Election Result - Low Priority

Name	Display election result
ID	UC_007
Description	After an election has been run, the system displays the details of the election results on the screen. This will include the type of election (i.e., STV/plurality), the number of ballots, the number of seats, the number of candidates, the list of winners and losers, the percentage of votes obtained if plurality, and the order of winning and losing for STV.
Actors	Election officials, tester
Organizational Benefits	Allows election officials to view details of election results.
Frequency of Use	Every time an election is run.
Triggers	The election has completed running successfully.
Preconditions	The system has completed running the election.
Postconditions	Election results are displayed.
Main Course	1. The system retrieves the election results once the voting process is complete. 2. The system arranges the results in a standard form for display. 3. The formatted results are displayed on screen.
Alternate Courses	There are no alternate courses.
Exceptions	There are no exceptions.

4.8 Input Audit File Name - Low Priority

Name	Input audit file name
ID	UC_008
Description	User inputs the name of the election audit file.
Actors	Election officials, testers

Organizational Benefits	Allows personalizable audit file names to more easily find them and refer to them.
Frequency of Use	Used every time there is an election and the program is run.
Triggers	The user starts the program.
Preconditions	UC_001 has finished and there is a valid input file.
Postconditions	The system saves the name to use for the audit file.
Main Course	1. The system prompts the user for what they would like to name the audit file. 2. The system saves that name to use when the audit file is generated.
Alternate Courses	There are no alternate courses.
Exceptions	There are no exceptions.

4.9 Generate Audit File Report - High Priority

Name	Generate audit file report
ID	UC_009
Description	Show the ballots were assigned to a candidate as the election progressed. Reports include all election data, such as type, seats, candidates, winners, and losers.
Actors	Election officials
Organizational Benefits	Official records for audits and recounts, trusting the voting process integrity
Frequency of Use	Generated at the end of every election
Triggers	The election is completed and the audit file is generated.
Preconditions	The election needs to be complete and the system needs to verify that all ballots have been accounted for.
Postconditions	A text file containing the audit report is generated.
Main Course	1. Outputs an audit file with a user-chosen filename for the report

Alternate Courses	AC_1: Duplicate File Name 1. The system detects that the specified file name already exists. 2. The system prompts the user to enter a new file name. 3. The user provides a new file name, and the system returns to main course step 1.
Exceptions	EX_1 System Fails to Generate Audit File 1. The system notifies the user that generating the file failed due to an internal error. 2. The user is given an option to proceed with the rest of the election cycle or exit the program.

4.10 Run Test File - Medium Priority

Name	Run test file
ID	UC_010
Description	The user has decided to run a test file to test the election system.
Actors	Tester
Organizational Benefits	Allows for unit testing and proper calibration.
Frequency of Use	Used when the user decides to test the program.
Triggers	The tester starts the program.
Preconditions	No precondition.
Postconditions	The input file will be read in by the file handler.
Main Course	1. The user is prompted by the system to enter a valid file. 2. User provides the name of the testing file. 3. The file input is read in by the file handler.
Alternate Courses	AC_1: User turns the shuffle off 1. Refer to UC_006 AC_2: User shuts down the voting system program at any time. 1. User restarts the system. 2. Return user to main course step 1.
Exceptions	EX_1 User enters a non-CSV formatted file. 1. System rejects the input and informs the user they did not enter a CSV file.

	2. Prompt the user if they want to try again or exit. 3. The system then prompts the user to enter a filename. If the user makes another mistake, the prompt repeats two more times before automatically exiting. EX_2 User enters an invalid input (multiple filenames, nonexistent files, wrong directory, wrong permissions, etc.). 1. System rejects the input and informs the user they entered invalid input. 2. Prompt the user if they want to try again or exit. 3. The system then prompts the user to enter a filename. If the user makes another mistake, the prompt repeats two more times before automatically exiting.

4.11 User Interface - Low Priority

Name	User interface
ID	UC_011
Description	There is a user interface where the user can input file, seat, and election choice information.
Actors	Election official, tester
Organizational Benefits	Added the functionality to get information from the user in a convenient manner.
Frequency of Use	Used every time there is an election and the program is run.
Triggers	Election official or tester starts the program.
Preconditions	No precondition.
Postconditions	No postcondition.
Main Course	1. User starts the program. 2. The user is presented with a user interface where they can enter information.
Alternate Courses	AC_1: User shuts down the voting system program at any time. 1. User restarts the system. 2. Return user to main course step 1.
Exceptions	No exceptions.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

1. Training for the user interface should take less than 15 minutes.
2. The system should be able to process an election in under 5 minutes.
3. The system should be able to process up to 100,000 ballots present in a file.
4. After an election has been run, the election results should be displayed to the user within a reasonable time period.

5.2 Safety Requirements

We assume there are no safety requirements to be handled at this point for the voting system.

5.3 Security Requirements

The voting system shall be used by election officials and testers. There are no security issues to be handled. The security of the voting will be handled at the voting precinct itself. Files received by the system will have been transmitted via a secure means outside of the system's scope.

5.4 Software Quality Attributes

1. Usability: The voting system interface would be easy to use. It should require less than 15 minutes of training for election officials.
2. Reliability: The system would produce accurate results for every election, even while processing up to 100,000 ballots
3. Robustness: The system would produce election results within 5 minutes.
4. Testability: The system shall support unit testing and calibration. Users would be able to run a test file and toggle the shuffle option via a command-line argument.

5.5 Business Rules

Election officials can perform the following roles:

1. Input ballot file location
2. Input the number of seats to fill
3. Run the STV type election
4. Run the Plurality type election
5. Name the audit file
6. View election results
7. Turn off the ballot shuffle option for unit testing or system calibration

Testers can perform the following roles:

1. Run test file to facilitate unit testing or system calibration
2. Have access to all the functionalities described for election officials

Input file rules:

1. Input file must be in comma-separated values (CSV) format
2. The first line of the file should contain the names of the candidates
3. Each subsequent line should be a voter's ballot

Ballot rules:

1. STV
    a. Each ballot must rank at least half of the candidates.
    b. Each ballot will have numbers listed for candidates in order of voting preference.
    c. An example of an STV ballot file is given below. The ballot has at least half of the candidates ranked.

    Bill Jones, Alice Mix, Sally Ride, Ahmed Mohamed, Siyang Xiong, Preeti Banerjee
    1,,2,,3,
    3,2,1,4,6,5
    1,2,,,3,4
    4,1,,2,,3

2. Plurality
    a. Each ballot will only have one candidate indicated as their choice for the vote.
    b. The ballot will only have a single 1 in the candidate's position that was voted for by the voter.
    c. An example of a plurality ballot file is given below.

    A, B, C, D, E, F
    1,,,,,
    ,,1,,,
    ,,,,,1

STV Algorithm rules:

1. For the STV election, the Droop Quota method would be followed, as shown below:
    a. Shuffle the ballots.
    b. Calculate the Droop Quota:

$$\left\lfloor \left(\frac{numberOfBallots}{numberOfSeats+1} \right) \right\rfloor + 1$$

    c. Distribute the ballots
        i. One at a time into piles for the candidates. The piles are ordered by the first vote received.
        ii. Any candidate who reaches the quota is immediately declared elected, and those ballots are removed permanently from the process. The elected candidate's name is recorded in the order of election.

iii. Subsequently, any ballot with a vote for the winning candidate will go to the next name on the ballot list.

d. Redistribution of ballots from losing candidates
   i. After the first round of ballot distribution, the candidate with the lowest number of votes is permanently eliminated, and their ballots are redistributed.
   ii. In case of a tie, the candidate who was last to receive their first ballot is eliminated.
   iii. Upon redistribution, if any candidate reaches the quota, they are immediately declared elected and added to the winner's list.
e. Continue until all ballots have been processed
   i. On completion of the process, there would be two lists, for the elected and the non-elected.
   ii. The lists will be ordered based on when they were elected and when they were placed on the non-elected list.
   iii. The candidates placed on the non-elected list last will be higher up on the election list itself.
   iv. Both lists would be provided to the user.
f. Provide election audit report to the user
   i. The report will show the ballots that were assigned to a candidate as the election progressed.
   ii. The report should not print to the screen but should be in a text file.
   iii. All relevant election information, including type of election, number of seats, number of candidates, winners, and losers, must be included at the top of the report.
   iv. Ballots will be numbered as they come in, and unique numbers will be used to show the progression of ballots being handed out to each candidate.
g. The user should be able to name the audit file.

Plurality Algorithm Rules:

1. Each ballot will be assigned to a candidate.
2. A count of votes will be maintained for each candidate.
3. The candidate(s) with the greatest number of votes will be declared the winner(s).
4. If there is a tie for a given seat, the winner will be randomly selected for that seat.
5. An audit report will be generated.

6. Other Requirements

This voting system should be able to be reused in future elections.

Appendix A: Glossary

- Ballot: A representation of voter preferences for Plurality and STV elections. "1" denotes the top choice in candidate selection.
- Candidate: Person applying for a position through election
- Election official: Person responsible for entering ballots into the voting system and running elections

- Plurality Voting: Each voter is allowed to vote for only one candidate, and the candidate who polls the most votes is elected
- CSV File: Comma-separated file
- STV: Single transferable voting. Voters rank candidates according to their preferences. Their vote can be transferred according to alternate preferences if their preferred candidate is eliminated
- GUI: Graphical user interface
- Linux: An open-source operating system in which this project will be run
- Droop Quota: The minimum number of votes a party or candidate needs to earn to win at least one seat in a district.

Appendix B: Analysis Models

N/A

Appendix C: To Be Determined List

N/A