Software Requirements Specification

for

Voting System

Version 1.0 approved

Prepared by Eileen Campbell (camp0870), Hazel Dunn (dunn0441),

Olivia Hansen (hans6055), & Maranda Donaldson (donal163)

Team 6

University of Minnesota CSCI 5801

February 20

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of an online voting system. It is capable of analyzing and performing two types of voting: Instant Runoff Voting and Open Party List Voting. This document will explain the purpose and features of the program, the interface of the software, the constraints it will have, and how the software itself will operate. This document is intended for users of this software such as election officials, potential testers, and developers.

1.2 Document Conventions

This document is based on the IEEE template for System Requirement Specification Documents. It also references use cases, which follow the template provided in section 1.5. Every requirement is to have its own priority throughout this document.

1.3 Intended Audience and Reading Suggestions

This document is intended for general users such as election officials, auditors, and anyone who may need to analyze election results. It will also be useful for programmers who are working to develop this product further or fixing software bugs. This document should be read from top to bottom. Section one and section two provide an overview description of the voting system. Section three describes the interfaces requirements. Section four is a description of each feature of the system. Lastly, section five provides a list of requirements for nonfunctional components of the systems. Every user will find section one and two useful for general information about the system. Users should also look to section three and four for details about specific features available and how to run them. Programmers and testers will find sections three and four especially useful when considering changes to the functionality of the system.

1.4 Product Scope

This software system will be a voting system to be used by election officials. The purpose of this product is to determine the winner of an election. The primary benefit of the use of this product is the quick and automated processing of a large amount of ballots. This system will be designed to run two types of elections, each with a specific voting algorithm. More specifically, during each run of the system, one file will be analyzed and that file will specify which of the two types of voting (Instant Runoff or Open Party List) will be used to determine the election. After running the voting algorithm, the system will produce for the user three different types of output: a quick statistical overview of the election that will be printed to the screen, a more detailed report (in the form of a text file) that can be sent out to media personnel, and an audit file that can be used to track where each individual vote counted during the election.

1.5 References

IEEE Template for System Requirements Specification:

https://docs.google.com/document/d/1zTn-hnF4ooLCzb33D23eqdV8T9QtMBnOH_0YmI4DIDA/edit?usp=sharing

Use Case Template Example (provided by Dr. Shana Watters):
https://docs.google.com/document/d/1yQR7dj6HfFzb0kOlKxdI7gH8QSBhwJ_d0O87N2VfVY0/edit?usp=sharing

Instant Runoff Voting Information:

https://www.fairvote.org/glossary

Part List Voting Information:

https://www.fairvote.org/how_proportional_representation_elections_work

2. Overall Description

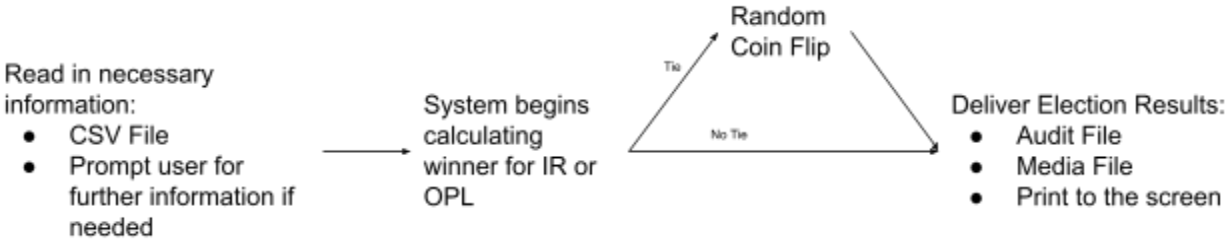
2.1 Product Perspective

This software is being developed for election officials to count ballots and determine the winner of an election. This particular voting software is to be used for two types of voting: Instant Runoff and Open Party listing. It is one component of the voting system, which also involves the casting of the votes. The casting of votes is completely separate from this software. These parts of the voting system interact through a file, which contains the type of election, the number of candidates, the candidates names, the number of seats (if applicable), the number of ballots, and the ballots themselves. For this software, we can assume that the file from the vote casting does not contain any errors.

2.2 Product Functions

Overview:

The product will be able to determine the election results for two types of voting systems, Instant Runoff (IR) and Open Party List Voting (OPL).



Deliver Election Results:

- Audit File: contains the distribution of ballots, path of each individual ballot, election results, and basic overview election statistics.
- Media File: contains a written statement of the election results and percentage each candidate and/or each party received.

- Print the election results to the terminal screen for immediate viewing by the user.

Read in Necessary Information:

- CSV File: containing basic election statistics and completed ballot information. This will be used to determine the winner.
- Terminal input from user: the user will be prompted for important election information not found in the CSV file.

Special Cases

Tie: if this occurs a random coin toss will determine the winner and loser.

- No Majority in IR: if this occurs, the popularity vote will be used to determine the winner.

2.3 User Classes and Characteristics

The users of this system (those who must directly interact with it) will fall into one of the following categories:

- Election Officials: These users will provide the main interaction with the system. It is expected that these users will be able to run the system from the command line and that they have access to the correct folders in order to place the input file in the correct spot. These users will mainly benefit from the output to the screen and the media report that is generated. It is assumed that these officials will be the ones communicating results to the media outlets.
- Testers: It is expected that these users will be able to run the system from the command line and that they have access to the correct folders in order to place the input file in the correct spot. These users will be the primary beneficiaries of the audit file. Using this audit file, the testers should be able to guarantee that the programmed algorithm will consistently produce the same results, and that votes are treated in the same manner.
- Programmers: The primary interaction with the system that these users will have is any further development of the system or possible bug fixes that need to be implemented.

2.4 Operating Environment

- CSV File comes from any operating system
- CSE lab machines
- possible machine recommendations: CSE Lab Machines, but you can run on your personal machine with minimum ram and processor
- Unix Lab Machines -> Ubuntu, Ubuntu 16.04, Ubuntu 20.04, Windows 10, Mac OS X
- Minimum Processing Requirements:
    - Space: 8 GB (Servers), 16 GB RAM, 32 GB RAM, 64 GB RAM, 192 (Server maximus)
    - Processors: i5, i7, i7-4770, i7-6700, i7-4790, i7-7700, Xeon, Xeon e5-2460, AMD Opteron
    - Monitors: One 34” LCD Monitor, one 30" LCD Monitor per computer

2.5 Design and Implementation Constraints

The voting software is to be designed in either C++ or Java. It can either provide a prompt to the user or a GUI as well. The program will be built to run on University of Minnesota CSE Lab machines. The program will have to handle one CSV file per election, and the way in which that

file is read in is up to the programmer. The program should be able to run an election with 100,000 ballots within 8 minutes. It can be assumed that the election file is in the same directory as the program. As the voting is separate from the program, there are no specific security requirements.

2.6 User Documentation

A basic terminal command line tutorial:
https://tutorial.djangogirls.org/en/intro_to_command_line/

2.7 Assumptions and Dependencies

This system is developed in java. Java 11 or later must be installed on the machine the software is to be run on. It is assumed that if power is cut to the machine that the system is running on, the system will have to be restarted and any files created should be deleted to avoid errors.
The following are assumptions about the input that will be used during the development process:

- There are no numbering mistakes in the input file.
    - For any Open Party List files, this means each ballot will have exactly one candidate marked with a 1.
    - For any instant runoff files, this means that each ballot will have at least one ranking and that there will be no ranking number issues.
- The input CSV file will be preprocessed before it reaches the system. It is expected that the first line of the file will provide the type of voting and the next 3-4 lines will indicate the number of votes, candidates/parties, etc (specified format in section 6.1).
- Only one file can be processed by the system during a run.
- To input the file, it will be placed in the same directory as the program.

In addition to these assumptions, we will assume the following about the elections when making the election algorithms:

- For both the Open Party List and the Instant Runoff algorithms, there will be no write in candidates
- In the Open Party Listing algorithm, all candidates that are independents will be grouped into one Independent party for analysis.
- In the Instant Runoff algorithm, it will be assumed that every ballot in the file will have at least one of the candidates ranked.

3. External Interface Requirements

3.1 User Interfaces

1. Welcome message displayed to user, then ask for file name - before program is run

```
Welcome!
Please enter the name of your file:
voting1.csv
You have entered your file name as voting1.csv
Is this correct? (Press Y/y for yes and N/n for no
N
Because you said the file name was incorrect, we will prompt you again.
Please enter the name of your file:
voting2.csv
You have entered your file name as voting2.csv
Is this correct? (Press Y/y for yes and N/n for no
Y
Great thanks! We will now continue.
hazeldunn@Hazels-MacBook-Pro CSCI5801 %
```

2. Ask user what to name media file, and what to name the audit file, have a default as well - before program is run

```
Great thanks! We will now continue.
We will now ask you a couple more questions before we begin the program.
What do you want the name of the media file to be?
If no name is entered, the default of media.txt will be used
med.txt
You have entered your desired media file name as med.txt
Is this correct? (Press Y/y for yes and N/n for no
N
Because you said the file name was incorrect, we will prompt you again.
What do you want the name of the media file to be?
If no name is entered, the default of media.txt will be used
mediaFile.txt
You have entered your desired media file name as mediaFile.txt
Is this correct? (Press Y/y for yes and N/n for no
Y
Great thanks! We will now continue.
What do you want the name of the audit file to be?
If no name is entered, the default of audit.txt will be used
aud.txt
You have entered your desired audit file name as aud.txt
Is this correct? (Press Y/y for yes and N/n for no
Y
Great thanks! We will now continue.
hazeldunn@ebm-er-10-131-75-3 CSCI5801 %
```

3. After the program is run: Display election results and information to the user

```
Great thanks! We will now continue.
We will now run the program. The election results will be displayed to the screen when the process has been completed.
Running program...
The Waterfall program has completed! Here are the election results:
Type of election: OPL
Number of candidates: 4
Cadidate names: Barack Obama(D), Donald Trump(R), Joe Biden(D), Kamala Harris(D)
Number of seats: 2
Number of ballots cast: 20,000
Winners: Joe Biden, Kamala Harris
Joe Biden: number and percent of votes received: 10,000 votes = 50%
Kamala Harris: number and percent of votes received: 10,000 votes = 50%
More detailed election information and individual ballot results will be store in the audit file.
Media results will be stored in the media file. Both are in your current directory.
The program will be closing now. Thank you.
hazeldunn@ebm-er-10-131-75-3 CSCI5801 %
```

The program is running completely in the terminal, so the screen layout constraints are that the screen size is as each terminal allows for the type of monitor.

Standard buttons and functions that will appear on every screen:

- Quit button
- Error message display standards - ERROR: some small description

```
Welcome!
Please enter the name of your file:
file
You have entered your file name as file
Is this correct? (Press Y/y for yes and N/n for no)
L
ERROR: Value entered is not Y/y yes, or N/n no. You will now be reprompted.
Please enter the name of your file:
file
You have entered your file name as file
Is this correct? (Press Y/y for yes and N/n for no)
Y
```

- Great thanks! We will now continue.

The software components for which a user interface is needed are that the machine being used must have access to the terminal command line. The user of the program must also have administrative privileges on the machine, have java, and be able to run it from the terminal.

3.2 Hardware Interfaces

This software is being developed to run on the University of Minnesota CSELab machines. See section 2.4 for hardware specification for these machines. The data will be input as a CSV file, with the majority of the file being each individual ballot. It will be assumed that there will be at least 8 GB RAM on each computer that the system is run on.

3.3 Software Interfaces

The voting system requires Java to run. Because the CSE Lab machines run Java 11, this program will be created with Java 11 and is the recommended software to use. It can be run on any computer which can compile a .java file in the terminal. It will make use of the built in java Math, FileReader, IOException, FileNotFoundException, Scanner, and FileWriter libraries. It does not require a database, a wifi connection, or any other device connection to share information. It is up to the user how they want to get the input CSV file into the same directory to run the program.

3.4 Communications Interfaces

The report and audit files produced by this system will be .txt files, so that they can be opened on any computer. It will be up to the user to send these files to any additional personnel. The system will not require an internet connection to run, as the program can be stored locally. Any updates that are made can be accessed either over the internet or can be transferred to the system with a drive. Since there are currently no security restrictions, there is no need for the system to communicate with any security services or encryption services.

4. System Features

4.1 Produce an Audit File

4.1.1 Description and Priority

The system is to produce an audit file for each election. The audit file should contain all necessary information to step through the election to ensure all ballots are accurately counted. This should include: The type of election, the number of candidates, the candidates names, the candidates parties, the percentage of votes awarded to each candidate, and the path of each ballot. This is of high priority, as being able to confirm the results of an election is crucial for public acceptance.

4.1.2 Stimulus/Response Sequences

The system will prompt the user for an audit file name (UC_003). After this, the system will create the audit file with the same name input from the user (UC_009) and write the type of election, the number of candidates, and the number of ballots to the file. From there, the system will run the algorithm, keeping track of the path of each ballot. It will input this information into the audit file as the algorithm is run. After the algorithm has completed, the system will then write the winner of the election and the percentage of votes awarded to each candidate.

4.1.3 Functional Requirements

REQ-1: The system must prompt the user for the audit filename. If no name is provided a default name will be used.

REQ-2: The system must keep track of and store each ballot's path throughout the algorithm.

REQ-3: The system must create an audit file with the name provided by the user or the default name.

REQ-4: The system must write all of the election information to the audit file: The type of election, the number of candidates, the candidates' names, the candidates' parties, the percentage of votes awarded to each candidate, and the path of each ballot.

4.2 Produce a Media File

4.2.1 Description and Priority

This system is to produce a media file for each election. The file should contain all information pertaining to the election that can be released to the public. This includes the type of voting system, the number of candidates, the name of each candidate and their party, the percentage of votes awarded to each candidate, and the winner of the election. This is of medium importance as it makes the data more presentable but is not necessary to run our system.

4.2.2 Stimulus/Response Sequences

The system will prompt the user for a media file name (UC_004). After this, the system will create the media file with the same name input from the user (UC_010) and write the type of election, the number of candidates, and the name of each candidate and their party. From there, the system will run the algorithm. After the algorithm has completed, the system will then write the winner of the election and the percentage of votes awarded to each candidate.

4.2.3 Functional Requirements

REQ-1: The system must prompt the user for the media file name. If no name is provided a default name will be used.

REQ-2: The system must create a media file with the name provided by the user or the default name.

REQ-3: The system must write the election information that can be released to the public to the media file: The type of election, the number of candidates, the candidates names, the candidates parties, the percentage of votes awarded to each candidate, and the winner(s) of the election.

4.3 Runs Two Election Systems

4.3.1 Description and Priority

This feature is of high priority. It is one of the main aspects that our system has to have in order to effectively run and produce election results for the two voting systems. Our main system must have the ability to analyze results from Instant Runoff Voting and Party List Voting elections. There will be a separate system in our program for each type of voting, as they each have their own unique features and ballot layouts. It is important that the system can accurately analyze results from both types of ballots and produce appropriate media and audit files.

4.3.2 Stimulus/Response Sequences

The use cases of Provide File Name Prompt(UC_001), Read in the File(UC_002), Prompt User for Audit File Name(003), Prompt User for Media File Name(004) and Determine Voting Type(005) have to be satisfied before this system feature is able to be stimulated. By reading in the first line of the CSV file and analyzing the information, the system will be able to determine the voting type. If the type is Instant Runoff Voting, the program will run a system that calculates the winners based on each ballot's ranking of each candidate. If the election is Open Party Listing, the program will run a system that calculates the winner based on what each ballot has as the chosen candidate, and the vote will go to the respective party (details of both elections are in Appendix A).

4.3.3 Functional Requirements

REQ-1: System must read the first line of the CSV file to determine the type of election, either IR or OPL, then will use it to run the corresponding system within the program.
REQ-2: The Instant Runoff System will determine the winner according to each ballot's ranking of each candidate. It is important that the system keep track of the path of each ballot. The system will analyze each ballot, eliminate the candidates with the lowest rankings, and redistribute votes until there are definite winner(s). Implementation details are TBD.
REQ-3: The Open Party List System will determine the winner according to the candidate that each ballot votes for. The votes will go to the corresponding party which will then determine how many seats each gets. Implementation details are TBD.
REQ-4: The system will store the winner and calculated information for later use in the program.

4.4 Read in a CSV File

4.4.1 Description and Priority

The system will take in a CSV file containing the ballots and other important election information. This is of high priority. The CSV file is how the program gets all the information it needs to analyze the election. Without a CSV file, the program will be useless. The software depends on an accurate and uncorrupted CSV file. The file must be in the same directory as the program.

4.4.2 Stimulus/Response Sequences

This is done as listed in use case (UC_001), Provide File Name Prompt, and use case (UC_002), Read in a CSV File. The program will prompt the user for a file name. The user will type in the name and confirm it is correct. If there exists a file matching the input, the program will begin to analyze the file. If no file exists, the program will continue to prompt the user until a matching CSV file is found. The program will read in and store all information which will be used to calculate the winner(s) of the election.

4.4.3 Functional Requirements

REQ-1: The program will prompt the user for the file name.
REQ-2: The program will open and read the file to analyze the election.
REQ-3: The program will store the data from the file to be referenced later.
REQ-4: The CSV file is in the format specified in section 6.1, CSV File Requirements.

4.5 Output Election Results to the Screen

4.5.1 Description and Priority

Each time a file is successfully processed, the system should display to the screen a small summary of the election results. This summary will include statistics such

as the type of election, the number of ballots cast, the winner of the election, and the percentage of votes awarded to each candidate. This is of medium importance, as it is not the primary form of releasing statistics to the public.

4.5.2 Stimulus/Response Sequences

For any results to be output to the screen, the system must first successfully process an input CSV file. During the run of the algorithm, statistics will be collected and stored in a data structure that will be used to print results to the screen. Once the algorithm has been run and the data collected, this feature will display a small summary of the election results. This will include the type of election, the number of candidates, the candidates names and party affiliations, the number of seats, the number of ballots cast, the winner(s) of the election, and the percentage of votes that was cast for each candidate (UC_011).

4.5.3 Functional Requirements

REQ-1: The program will display the type of election.
REQ-2: The program will display candidate information, including name, party affiliation, and percentage of votes received.
REQ-3: The program will display the number of candidates involved in the election and the number of seats up for election (if applicable to the type of election).
REQ-4: The program will display the number of ballots cast.
REQ-5: The program will display the determined winner(s) of the election.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The program must be able to process 100,000 ballots in under 8 minutes.

5.2 Safety Requirements

There are no safety requirements.

5.3 Security Requirements

There are no security requirements. Anyone with a copy of the software and a set of voter data can use the program to obtain election results.

5.4 Software Quality Attributes

This program is available for regular and special elections throughout the entire year. It relies on the user having basic computer terminal knowledge, such as compiling java files and moving throughout directories. Though this may not be common knowledge, it requires minimal steps that are easily learned. Refer to section 2.6 for a basic terminal command line tutorial. This software can be reused to find reliable results unless standard voting system requirements are changed.

5.5 Business Rules

There are no business rules that will impact the way the program is developed and run.

6. Other Requirements

6.1 CSV File Requirements

The CSV File passed into the program will follow the format below depending on the voting system of that election.

6.1.1 **Instant Runoff Voting**

- Each ballot must have at least one candidate ranked as their top choice.
- At least one candidate must be ranked per ballot.
- A candidate can only be given one ranking.

1st Line: IR if instant runoff
2nd Line: Number of Candidates
3rd Line: The candidates separated by commas
4th Line: Number of ballots in the file
5th Line and on: One ballot per line where their ranked votes are represented with 1 through 4 (1 being the favorite and 4 being the least preferred).

Example CSV File:

```
IR
4
Rosen (D), Kleinberg (R), Chou (I), Royce (L)
6
1,3,4,2
1,,2,
1,2,3,
3,2,1,4
,,1,2
,,,1
```

6.1.2 **Open Party List Voting**

- Each ballot can only have one candidate indicated as a choice for the vote.

1st Line: OPL for open party listing
2nd Line: Number of Candidates
3rd Line: The candidates and their party in this form: [candidate, party].
Notice the name and party are separated by commas
4th Line: Number of Seats
5th Line: Number of Ballots
6th Line and on: One ballot per line where a 1 indicates a vote

Example CSV File:

```
OPL
6
[Pike,D], [Foster,D],[Deutsch,R], [Borg,R], [Jones,R],[Smith,I]
3
9
1,,,,,
1,,,,,
,1,,,,
,,,,1,
,,,,,1
,,,1,,
,,,1,,
1,,,,,
,1,,,,
```

Appendix A: Glossary

- Instant Runoff Voting (IR): All candidates are listed on the ballot and voters rank candidates in order of their preference.
- Open Party List Voting (OPL): Voters are presented an unordered list of candidates chosen in party primaries. Voters cast a vote for individual candidates. This vote counts for the specific candidate as well as for the party. Number of seats are delegated by which party got the most votes and which candidates were the most popular in each party.
- CSV File: Comma Separated Values. A plain text file that contains a list of data. Every value separated by a comma.

Appendix B: To Be Determined List

- Details of how the system for Open Party Listing will operate (Section 4.3).
- Details of how the system for Instant Runoff Voting will operate (Section 4.3).