# Software Requirements Specification

**for**

# Voting System

**Version 0.5 approved**

**Prepared by Team 11**

**University of Minnesota Twin Cities**

**02-19-2021**

# Table of Contents



# Revision History

| Name | Date | Reason For Changes | Version |
|---|---|---|---|
| Team 11 | 01-31-2021 | Creation of document. Began filling in core ideas and dividing up tasks among teammates. | 0.1 |

| | | | |
|---|---|---|---|
| Ryan Mower | 02-02-2021 | Added Non-functional requirements, performance requirements, safety requirements, security requirements, and other requirements. | 0.2 |
| Hoai Bui | 02-05-202 | Added Introduction. | 0.3 |
| Eric Palmer, Emma Spindler | 2-08-2021 | Added External Interface Requirements and Overall Description. | 0.4 |
| Team 11 | 02-12-2021 | Added System Features. | 0.5 |

# 1. Introduction

## 1.1 Team Members

1.1.1 Ryan Mower: mower023
1.1.2 Emma Spindler: spind038
1.1.3 Hoai Bui: bui00015
1.1.4 Eric Palmer: palme885

## 1.2 Purpose

This document was created to provide a full explanation of the Voting System. It will describe the Voting System's purpose and features. In addition, it will describe the interfaces, what the system will do, as well as the constraints it must work under. This document is made for users of the system, developers, and testers.

## 1.3 Document Conventions

This document was made using the IEEE template for System Requirement Specification Documents.

## 1.4 Intended Audience and Reading Suggestions

This Systems Requirement Document may be read by anyone who would like to learn more about the details of the Voting System as well as how it will operate. It was primarily written with developers and testers in mind. Developers may have a better understanding of the product and its specifications upon reading this document, so they may carry on with software creation. Testers will also be able to use this document to determine if the system satisfies what is specified.

An outline of the Voting System as well as its features and requirements can be found in this SRS document. At the end, there is an appendix in which the reader may turn to for

clarification. It is recommended that the document be read in the way it was written, linearly, in order to have a full understanding of the product.

## 1.5 Product Scope

The Voting System is a tool intended to be used for both Instant Runoff and Open Party Listing elections. Election officials will organize ballots into a CSV file and the software will perform the appropriate operations in order to determine the winner(s) based on the specified election type. The results along with other important information regarding the election will be returned in an audit and media report to be reviewed and shared.

## 1.6 References

GitHub repository for Voting System:
https://github.umn.edu/umn-csci-5801-S21-001/repo-Team11

IEEE SRS document template used:
Download ieeeSoftwareRequirements.docx

# 2. Overall Description

## 2.1 Product Perspective

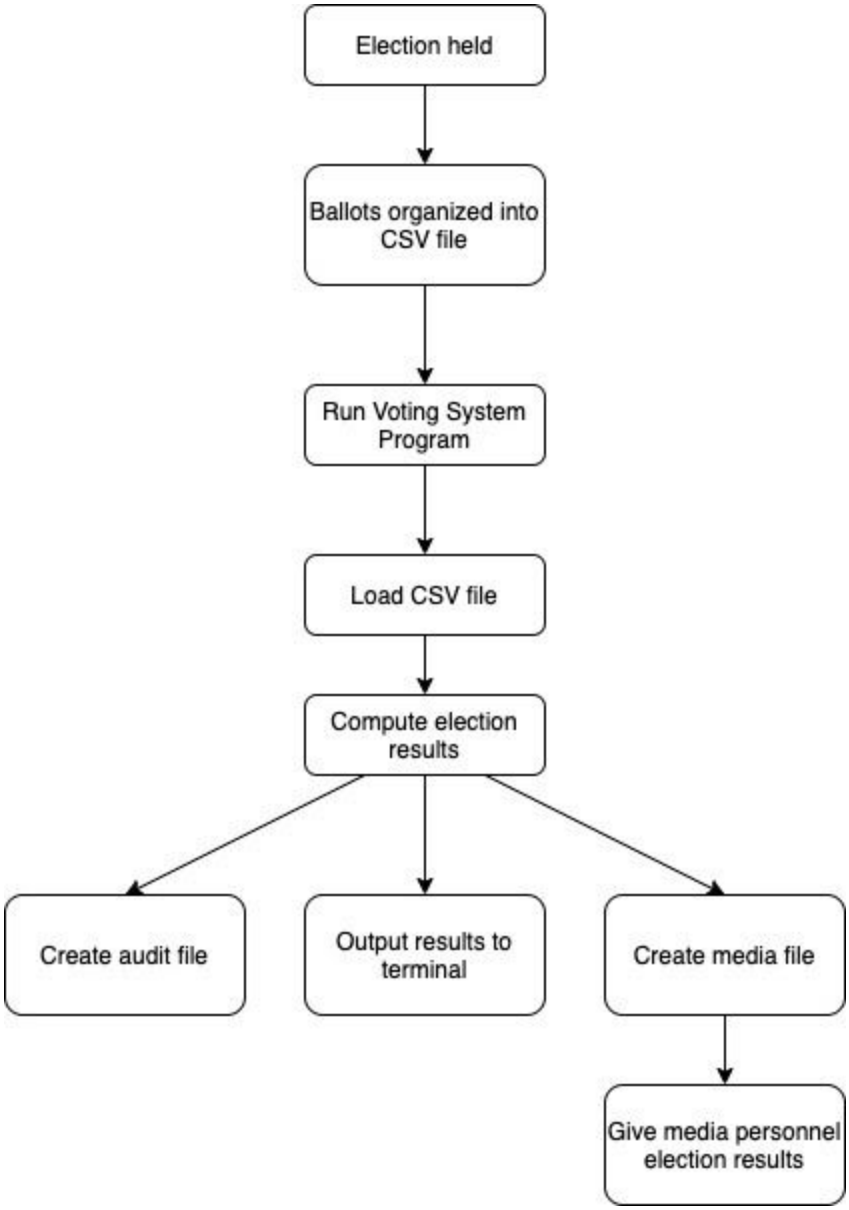
The Voting System is designed to replace the manual task of computing election results by hand. Manual counting is tedious and prone to human error. Replacing the evaluation of election results with a deterministic voting system program decreases the required time for obtaining results and minimizes error. The Voting System will not handle the actual voting, but rather the ballots once they are casted. Voting System termination produces media and audit reports displaying election results computed from the passed in CSV file.

## 2.2 Product Functions

Voting System major functions:

- Load csv file
- Compute election results based on election type
- Output results to the screen
- Create audit file that displays election progression and election results
- Create media file with election results

Voting System Flowchart:
The Voting System Flowchart displays the dynamic progression of steps from hosting an election to the generation of the election results. This includes hosting the election to the creation of the audit and media report.




## 2.3 User Classes and Characteristics

- Programmers who are interested in further development of the project. To continue development, they will need a maximum access level to modify the source code of the Voting System.
- Testers will have access to run the program and provide feedback to programmers. This is so they can report any bugs or other issues within the Voting System.
- Election Officials can run and view election results as they execute the program. Election Officials should only have the executable of the Voting System.

## 2.4 Operating Environment

The Voting System program is being developed on several different operating systems including Windows 10, Ubuntu 20.04 LTS, and Macintosh Catalina. However, the target machine should be running Linux Ubuntu 16.04 LTS or a more recent version. Additional requirements for the target machine are that it has at least 16 GB of DDR4 1800 MHz of random access memory and 1 GB of disk space for the audit and media report file. The target machine must have at least 4 cores and an Intel 64 bit, i7-4790, 3.60GHz processor or better.

## 2.5 Design and Implementation Constraints

The Voting System will be developed in C++. It must be able to be run on the most up-to-date CSE lab machine and be able to evaluate 100,000 ballots within 8 minutes. Voting System should be able to evaluate Instant Runoff and Open Party Listing. Voting System will be able to be used on Linux Ubuntu 16.04 LTS or a more recent version. The target machine should have at least 16 GB of RAM and 1 GB of disk space for the generated reports. Programmers will be responsible for future maintenance. No graphical user interfaces will be necessary because the program will be executed via command line.

## 2.6 User Documentation

C++ user manual available on website:
http://www.cplusplus.com/doc/oldtutorial/introduction/

Programmers will use the Makefile to compile the Voting System and then use terminal commands to execute it.

Developers will be prompted by the terminal for additional information if needed.

GitHub page for Voting System:
https://github.umn.edu/umn-csci-5801-S21-001/repo-Team11

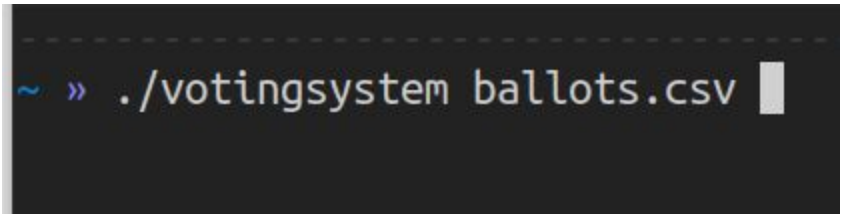
## 2.7 Assumptions and Dependencies

- The user's device must be running Linux Ubuntu 16.04 LTS or a more recent version.
- The CSV file is formatted correctly.
- The election type being used is either Instant runoff or Open party listing.
- Voting System is run once for each election.
- There is at least one ballot entered in the CSV file.

# 3. External Interface Requirements

## 3.1 User Interfaces

The user interface will use the bash terminal. The user will be able to run the program using textual input seen below.




A lot of the user's interaction with the Voting System will be outside the scope of the program. They will create and modify the CSV input file using a tool of their choice, whether that's Excel, Google Sheets, or some command line tool. Various text editors will be able to read and modify the generated audit and report file.

## 3.2 Hardware Interfaces

The interactions between the software and the hardware components of this system primarily consists of file input and output. A CSV file will be read by the program from the hard drive, and an audit file will be written onto the hard drive by the program. Because the program will be compiled in C++, it should be executable on most 64 bit Unix systems.

## 3.3 Software Interfaces

The data coming into the program will be in the form of a CSV file. The user will type the name of a file as a command line argument to the program. This file's first line will contain the type of voting, either Instant Runoff or Open Party, and the second line will be the number of candidates. From there, the CSV's format will diverge. In the case of Instant Runoff, the third line is the list of candidates separated by commas. Instant Runoff's fourth line is the number of ballots in the file. In the case of Open Party voting, the third line will consist of sets of [Candidate, (D, R, or I) representing party] separated by commas. Open Parties' fourth line is the total number of seats. The fifth line in Open Party Listing is the total number of ballots.

The data will be output in the form of an audit and report file. This file contains the type of voting, number of candidates, names, number of ballots, calculations (rankings of candidates, overall election winner, determination of how many seats each party got), how many votes each candidate received, and, if relevant, candidate's party. In addition to this final information, there will also be a progression history stored in order to replicate every step of the election along the way.

## 3.4 Communications Interfaces

This product has no internet-based communications. Communication will instead be done through an input CSV ballot file. Additionally, the election results communicated through an audit and media file. The media file contains general election information, while the audit file contains general election information as well as a progression history to allow for election replication.

# 4. System Features

## 4.1 Accept and Load a CSV Ballot File

### 4.1.1 Description and Priority

Accepting and loading a CSV ballot file includes obtaining a properly formatted CSV ballot file from an election and loading it into the Voting System program. This is highly important, rating of 9 for priority, because it is crucial to have the election information to compute the election results.

### 4.1.2 Stimulus/Response Sequences

Election officials will interact with the CSV ballot file and the Voting System by executing the Voting System executable followed by the name of the ballot CSV file. The response of this action will be the execution of the Voting System which will compute and display the election statistics.

### 4.1.3 Functional Requirements

See document Use-Cases_Team11 for a complete description of the use-cases. ALBF stands for accepting and loading the ballot file.

Accepting and loading a CSV ballot file includes use-cases:

1. ALBF_001
2. ALBF_002

## 4.2 Compute Election Results

### 4.2.1 Description and Priority

Runs the contents of the CSV file through a series of steps based on the specified election type in order to determine the winner(s). This feature is of high priority because it is the main purpose of the election system. Without this feature, there is nothing to handle the election, thus it is of level 9 priority.

### 4.2.2 Stimulus/Response Sequences

Election results will be computed as soon as the CSV ballot file is loaded. The suitable operations will then be performed depending on if the election is specified to be Instant Runoff or Open Party Listing. Ties will be settled with a coin flip. The results will be returned in media and audit reports.

### 4.2.3 Functional Requirements

See document Use-Cases_Team11 for a complete description of the use-cases. CER stands for compute election results.

Compute election results includes use-cases:

1. CER_001
2. CER_002
3. CER_003

## 4.3 Produce and Distribute Reports Displaying Election Statistics

### 4.3.1 Description and Priority

Producing reports for media officials is necessary to spread the election results to the public. The additional audit file allows for election replication to ensure an accurate and fair election. Additionally, the election results will be outputted to the terminal upon program completion to quickly announce the winning candidates to the election officials. Producing these reports is the main purpose of the Voting System, so they are considered a 9 priority.

### 4.3.2 Stimulus/Response Sequences

Media and audit reports will be generated right before program termination. These media and audit reports can be distributed to the public and media officials to display the winners of the election.

### 4.3.3 Functional Requirements

See document Use-Cases_Team11 for a complete description of the use-cases. PER stands for producing election reports.

Accepting and loading a CSV ballot file includes use-cases:

1. PER_001
2. PER_002
3. PER_003
4. PER_004

# 5. Other Nonfunctional Requirements

## 5.1 Performance Requirements

The Voting System is tasked with collecting votes for many users, ranging up to 100,000. Being such a large number, the Voting System must be able to process and successfully terminate in 8 minutes or less. Another requirement is that the election CSV file is located within the same directory as the executable for the Voting System's algorithm. The executable is required to be one file that contains the entire program. In addition, the target machine must have at least 16 GB of RAM and 1 GB of disk space for the outputted reports. It is also important to note that the Voting System does not need a network connection for successful execution.

## 5.2 Safety Requirements

There are currently no special safety requirements for the Voting System.

## 5.3 Security Requirements

There are currently no special security requirements for the Voting System. Security such as ensuring one vote for one person is handled at the voting centers.

## 5.4 Software Quality Attributes

A completely deterministic voting algorithm is required to ensure the correct candidates are elected every execution, unless there is a tie. In the case of a tie, a random coin flip will then select a winner out of the candidates competing in a runoff. Additionally, the program should be able to handle multiple executions with unique voters and candidates to improve reusability. This is because the Voting System is designed to be run several times throughout the year at normal election times and special elections. Teaching new users how to start the program on a desired ballot file should take less than one minute to ensure simplicity. Results outputted by the program should be clear and easy to read. In the case of the terminal output, users should be able to see who won the election for each type of election within 10 seconds. Users should be able to interpret the audit file in less than 1 minute.

## 5.5 Business Rules

Election periods are when the Voting System will be utilized. To use the system, election officials will be the primary actors. They should receive a ballot comma separated file which contains all of the results from the election. From here, they can execute the Voting System on this CSV file.

# 6. Other Requirements

Voting progression must be recorded throughout program execution. To accomplish this, an audit file must be created and updated. The audit file should display constant progression towards the generated results. Due to constant progression tracking, replaying an audit file should allow the election to replicate itself. The audit file should include the type of Voting System, the number of candidates, candidates, number of ballots, the order in which the ballots were received, all the calculations and how many votes each candidate had, and the winners of the election. In addition to the audit file is to output the results to the screen upon successful termination. This includes the winners of the election, number of candidates, number of ballots, the type of election and other information displaying how the election progressed. An additional requirement is that the Voting System program is one executable file to provide an easy form of execution.

# Appendix A: Glossary

- **AC** - Alternate course
- **ALBF** - Accept and load a CSV ballot file.
- **Ballot file** - The ballot file is the comma separated file that contains user's votes. This file will then be passed to the Voting System algorithm to calculate the results of the election.
- **CER** - Compute election results

- **CSV** - Comma separated file. This is a file that contains data that is delimited by a comma.
- **Election information** - A vague term that represents election statistics. It can be broken down into three-sub categories denoted by the context in which the phrase is used.
    - Audit file election information - How the election progressed (which vote went to which candidate), election type, number of seats and ballots, candidates and their party, winners and losers, and which candidates were eliminated at specific rounds.
    - Terminal/Media report election information - The results and general information about the election.
    - Voting information - Who the people voted for, candidates and their party, election type, number of ballots, and the number of seats. This information will compose the ballot CSV file, so everything necessary to compute election results must be inside this file.
- **EX** - Exception
- **MISC** - Miscellaneous
- **PER** - Produce and distribute reports displaying election statistics.
- **Target Machine** - The computer in which the Voting System will be run on. Our target machines are the college of science and engineering computers. These computers are running Ubuntu 16.04 LTS or earlier.

# Appendix B: Analysis Models

There are currently no additional diagrams or models at this time.

# Appendix C: To Be Determined List

1. How will the audit file be used?
2. How to show election progression in an audit file?