Software Requirements Specification

for

Voting System

Version 1.0 approved

Prepared by Adam Wall, Christopher Hanson,
Gabriel Jardio, Patrick Perez
Course CSCI 5801

02/21/2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
VotingSystem	02/20/20	Initial Version	0.1
VotingSystem	02/21/20	Added Sections 1, 2 ,4, and 5	0.8
VotingSystem	02/21/20	Added Section 3, and appendices	0.9
VotingSystem	02/21/20	Finalized Initial document draft	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the software Voting System. It will explain the purpose and features of the software, the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for users of the software and also potential developers. This SRS describes only a part of the overall election system, where actual voting is done separate from this software.

1.2 Document Conventions

This document will be written following the IEEE SRS template.

1.3 Intended Audience and Reading Suggestions

This document is intended for software developers in the future who will update this software along with testers of the software. Election officials can also look over this document if they have any concerns about the software that was created and the requirements used in design. It is best to read through this document mostly in order. The images for the user interfaces might not be as important as the section that talks about functionality for software developers.

1.4 Product Scope

Voting System is a software tool that is capable of determining the results of an election. The software will take in ballots and will be able to employ one of two voting algorithms, STV or plurality, to best determine the winners and losers. The software will be able to provide the users with election statistics to analyze about the election after it has been run.

1.5 References

- Project 1 Waterfall Voting SRS Reference Sheet
- Software Requirements Specification Document Template
- Use Case Structured Specification Template Example
- Voting Definitions In glossary came from Wikipedia

2. Overall Description

2.1 Product Perspective

This system will be for election officials to learn the results of an election without doing a physical count. This will decrease the amount of time it takes to count and decide an election, and eliminate some of the potential for human error. This will be a stand-alone software that may be updated in the future with menu options or further implementation alongside other election software tools.

2.2 Product Functions

- Run an election system to determine the winners and losers of an election
    - Two different techniques for determining the winner can be select
        - STV voting
            - A Droop count is calculated and used to find proportional election results
            - Ties are determined by which candidate received their initial vote first
        - Plurality voting
            - Votes are counted off of CSV files and the candidate(s) with the highest number of votes wins
            - Ties will be determined by a random selection between the candidates that tie
    - The system will then give results of the election
        - The type of election
        - Who won and who lost
        - Number of Ballots
        - Number of Seats
        - Candidates
        - Winners and Losers
    - Audit will be created to show how calculations were done

2.3 User Classes and Characteristics

- Voting Official: Most important - and only intended - user of this software. Has access to all functions of the software (collecting ballots, running voting algorithms, etc.) Should be technically experienced in running software.

2.4 Operating Environment

This program must run on a CSElabs machine. This is a linux based operating system with a Dell PC machine being used.

2.5 Design and Implementation Constraints

This software must run on a CSElabs machine so any hardware or interface constraints will come from this machine. No other applications or tools will need to be used in conjunction with running the software. The software will be written using C++ and if any database will be used is still to be determined based on the needs of the project. No security standards need to be used, but the code created must be properly commented and designed so that it can easily be changed in the future.

2.6 User Documentation

All help and information about how to run the software will be given within the software itself (help screen). This screen will provide textual knowledge of how the system is run and how to run the system as an election official. This software will also need documentation of an election (CSV comma-delimited files) to run.

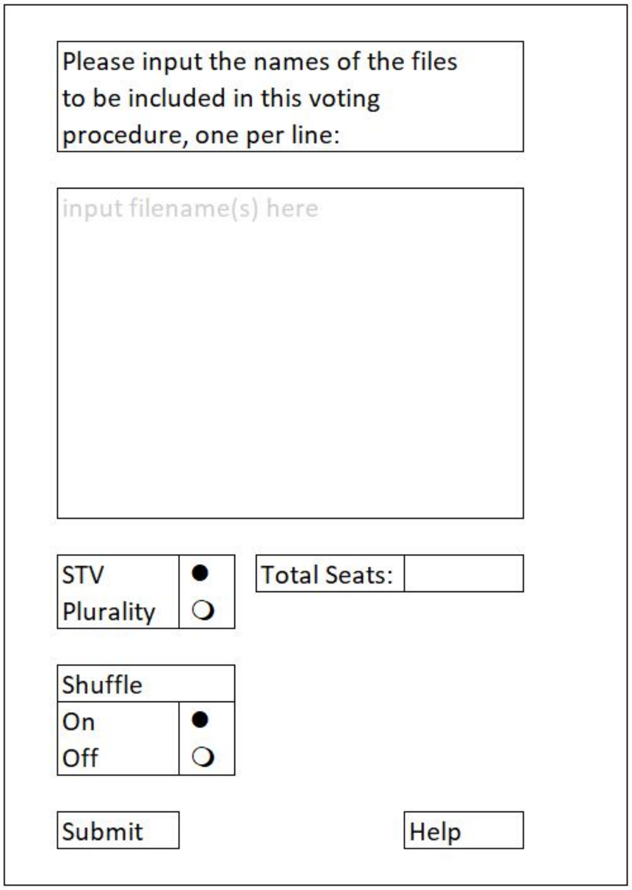
2.7 Assumptions and Dependencies

This software will be run independently with no additional software being needed. No dependencies or assumptions will affect the design of the system.

3. External Interface Requirements

3.1 User Interfaces

Initial Screen:



Please input the names of the files to be included in this voting procedure, one per line:

input filename(s) here

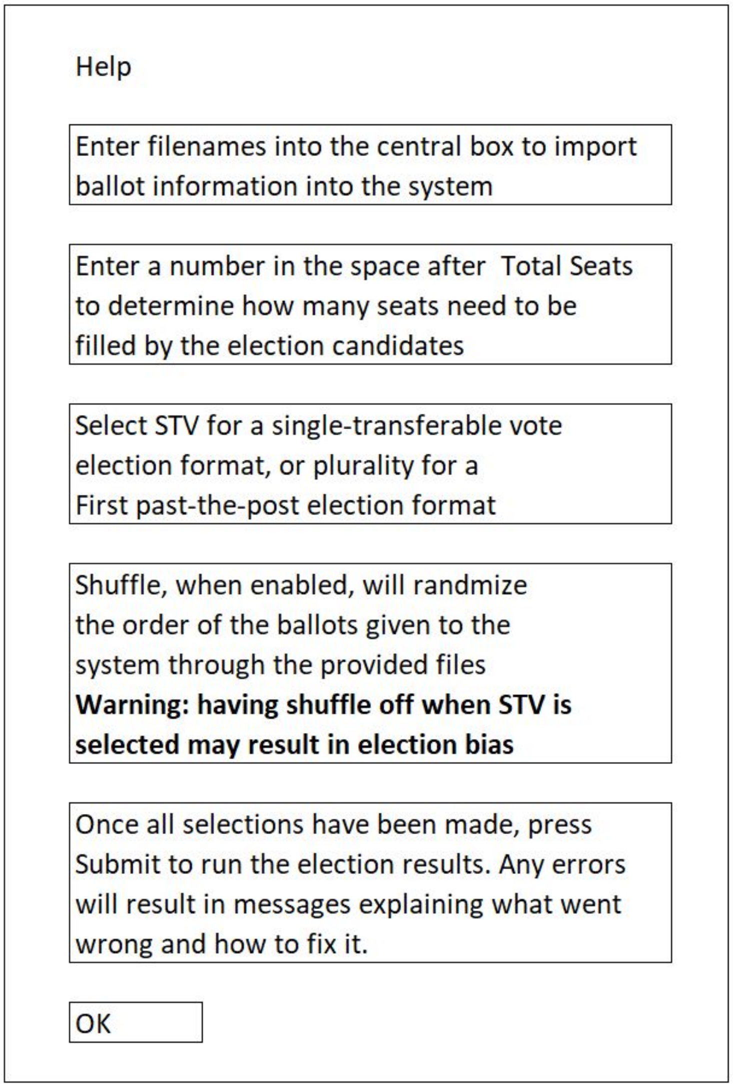
STV ●
Plurality ○

Total Seats:

Shuffle
On ●
Off ○

STV and Shuffle on will be chosen by default, though they can be toggled to plurality and off, respectively.

Help Screen:



Help

Enter filenames into the central box to import ballot information into the system

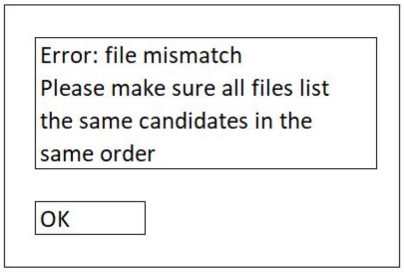
Enter a number in the space after Total Seats to determine how many seats need to be filled by the election candidates

Select STV for a single-transferable vote election format, or plurality for a First past-the-post election format

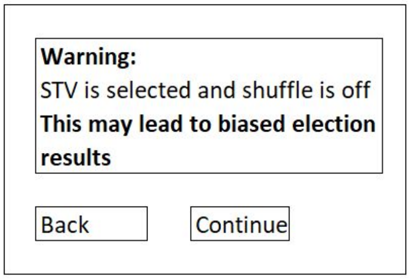
Shuffle, when enabled, will randmize the order of the ballots given to the system through the provided files
Warning: having shuffle off when STV is selected may result in election bias

Once all selections have been made, press Submit to run the election results. Any errors will result in messages explaining what went wrong and how to fix it.

Error Message Examples:

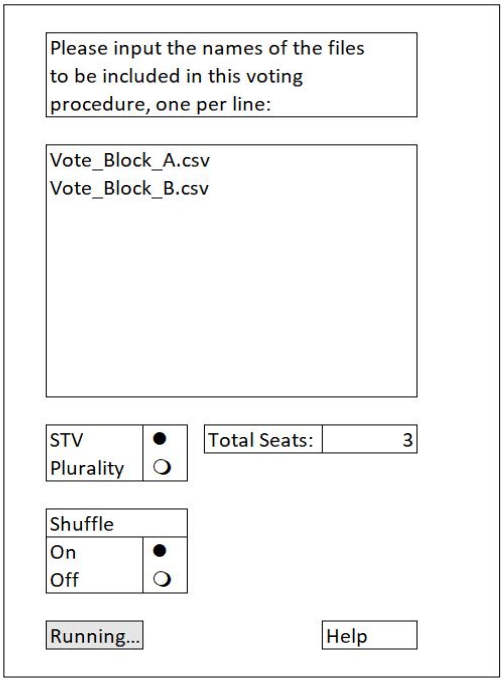


Error: file mismatch
Please make sure all files list the same candidates in the same order



Warning:
STV is selected and shuffle is off
This may lead to biased election results

Running Screen:



Please input the names of the files to be included in this voting procedure, one per line:

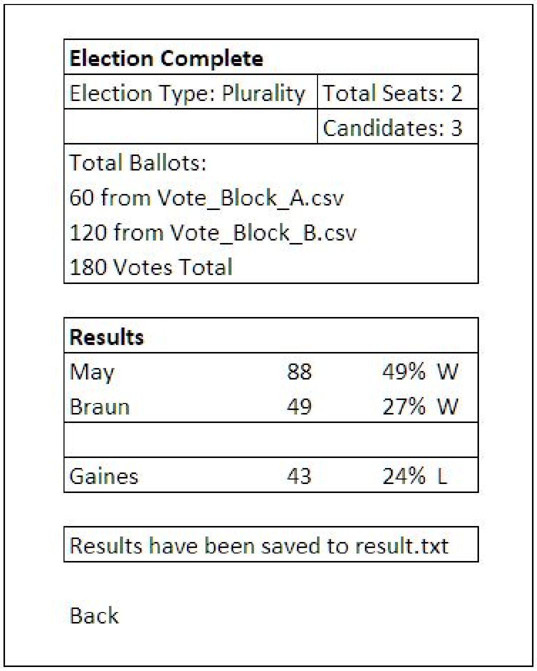
Vote_Block_A.csv
Vote_Block_B.csv

STV	●
Plurality	○

Total Seats: 3

Shuffle	
On	●
Off	○

Audit Screen:



Election Complete	
Election Type: Plurality	Total Seats: 2
	Candidates: 3
Total Ballots: 60 from Vote_Block_A.csv 120 from Vote_Block_B.csv 180 Votes Total	

Results			
May	88	49%	W
Braun	49	27%	W
Gaines	43	24%	L

Results have been saved to result.txt

3.2 Hardware Interfaces

The hardware interface for this system is out of the scope of the project. Software must work on CSElabs.

3.3 Software Interfaces

Once the program has been started, it will only be interacting with the user and the data. Data coming into the software includes csv files of votes for a candidate with a line for each voter and commas separating each vote.

3.4 Communications Interfaces

No communication with the internet is done in this software.

4. System Features

4.1 File Selection

4.1.1 Description and Priority

This is an input from the user to select which file they would like to perform the vote counting algorithm on. This feature has a high importance as it is needed for being able to identify what file is being used for counting. This approach can be done with many different methods but it must be easy for the user to understand how to select a file. In the case of this software, a file is selected by an input of the file address by the user.

4.1.2 Stimulus/Response Sequences

This process is done after the voting type has been selected. The software will ask the user for the file address of the .csv file. If the file is not found, it will prompt the user to ask again for an address or to return to the main screen.

4.1.3 Functional Requirements

REQ-1: User inputs a file address for the software to find and begin counting votes of the file

REQ-2: The software will notify the user if a wrong file has been selected (such as not a CSV file or a CSV file without the proper

REQ-3: The software allows the user to return to the main screen if the file entered was incorrect

4.2 Plurality Voting System

4.2.1 Description and Priority

Plurality voting is a specific voting type where each vote is counted as one vote and the candidate(s) with the highest number of votes wins the election. This process is selected before a file is picked where the user is asked if CTV voting or plurality voting will be chosen. After all other required items have been determined, the plurality voting algorithm will determine the winners. This process isn't as high priority as STV voting but is still required to fulfill the requirements for this system.

4.2.2 Stimulus/Response Sequences

This process is done right at the start after the software has begun. The software will prompt the user with a question on which voting system they want to use. Once Plurality voting has been chosen, the voting system of plurality will be completed.

4.2.3 Functional Requirements

REQ-1: Users must be able to select which voting system they would like - plurality voting being one option

REQ-2: The software must allow the user to run plurality voting which is done with an algorithm that counts every vote and has the winners with the most votes displayed to the user

4.3 STV Voting

4.3.1 Description and Priority

STV (Droop) voting is a specific voting type where each vote is counted until a "droop" number has been reached, once this number has been reached a winner and a loser is selected and the process starts over until all candidates have been placed in the winner or loser category. This process is selected before a file is picked where the user is asked if CTV voting or plurality voting will be chosen. After all other required items have been determined, the STV voting algorithm will determine the winners. This process is of high priority.

4.3.2 Stimulus/Response Sequences

This process is done right at the start after the software has begun. The software will prompt the user with a question on which voting system they want to use. Once STV voting has been chosen, the voting system will begin. Once completed, a list of winners and losers will be provided along with a method on how it was determined.

4.3.3 Functional Requirements

REQ-1: Users must be able to select which voting system they would like - STV voting being one option

REQ-2: The software must allow the user to run an STV voting system. This process will show winners and losers of the election based on the STV algorithm

REQ-3: This process is documented so that users understand how exactly a winner was determined

4.4 Help Window

4.4.1 Description and Priority

The help window is designed and included in software in case the user has questions or needs help running the software. Once accessed it should display all the information the user needs in order to access every function in the software. The priority for this function is not high since the voting official should be well versed or trained in running the software. This function is more targeted for newer users/voting officials who have less experience in running the software.

4.4.2 Stimulus/Response Sequences

This function can be accessed right after the software begins running. The user can select the help button located on the bottom right of the softwares initial screen. Then the software responds by creating a help window that displays help information in running the software.

4.4.3 Functional Requirements

REQ-1: The software must be running and be in the initial screen

REQ-2: The software must include a help button in order to access the help window

REQ-3: The software must be able to display a help window after the help button is pushed.

4.5 Audit

4.5.1 Description and Priority

The audit feature acts as a report for the election which will show relevant stat information. The report will print to a text file and will print information such as ballots that were assigned to each candidate, type of election, number of seats, number of candidates, winners, losers, etc. The priority of this feature is moderate as it can be used for election officials to analyze the validity of the election.

4.5.2 Stimulus/Response Sequences

This function begins after the user runs the voting algorithm in the software. As the software is running the algorithm, it will also be printing information to a text file at relevant steps until the software finishes the algorithm.

4.5.3 Functional Requirements

REQ-1: The software must be able to have access to a text file and print to it.

REQ-2: The software must be able to run the two voting algorithms and be able to print to text file at the same time.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The calculation of winners and losers must be done in under 5 minutes.

5.2 Safety Requirements

All data used in testing must not be deleted. All data will be properly counted so that the correct winners and losers of an election can be chosen.

5.2 Security Requirements

This Software will not require any security requirements. All privacy and log in possibilities are out of the scope of this project.

5.4 Software Quality Attributes

- The software must have an intuitive UI to increase usability of the software to the user.

6. Other Requirements

No other requirements currently need considered

Appendix A: Glossary

STV (Single Transferable Vote): *a proportional voting system designed to achieve or closely approach proportional representation through voters ranking candidates in multi-seat organizations or constituencies (voting districts).*
Droop Count (or Droop Quota): *The minimum number of votes a candidate must receive in order to be elected*

Appendix B: Analysis Models

None at this time

Appendix C: To Be Determined List

The following assumptions apply to this version of the Voting System, and may be revisited and addressed in later project iterations:

- Provided files are assumed to be correctly formatted
- Provided files are stored in the same directory as the program
- Provided STV ballots all have given at least half of all candidates an appropriate ranking