Software Requirements Specification

for

Voting System

Version 2.0 approved

Prepared by Aahan Tyagi, Alexander Grenier, Dominic Deiman, Jackie Li

Group 2

2/16/2023

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Group 2	2/13/2023	First Draft	1.0

Group 2	2/16/2023	Final Draft	2.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of a voting system. This document will explain the purpose and features of the program, what the program will do, and the constraints under which the program will operate under. This document is intended for both future developers, users of the software, and potential stakeholders.

1.2 Document Conventions

The document is divided into two sections, a header and a body section, where the header is the general idea and the body goes into more detail. The body sections are written in font size 11 and a font of Arial. The headers are in font size 14 and a font of Times. The whole document is following the IEEE template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

The typical user can use this document to learn more about the voting systems they have been using. Future engineers can use this document to have a quicker understanding of how the program operates and what the intended goal is. There are project managers and marketing staff that will want to understand the project before either having it expanded or marketed out to the public.

The sequence for reading this document is to start from the top and move down to the bottom. This is the most natural way to follow the documents flow and to gain the best understanding. However if the goal is to just learn and understand what the system does, skipping past to the 3rd or 4th sections will be best.

1.4 Product Scope

The Voting System software is designed to reduce the amount of time it takes to count and determine the winner of an election. This system will specifically take in a file that accounts for all of the votes and produce a winner and an audit file. The audit file can be kept in storage as a record of the voting process and/or shown to the media. This will be extremely beneficial as this will drastically reduce the amount of time it takes to count votes and determine a winner. The software will be exceedingly useful for counting votes as this drastically increases the speed and will increase the security of voting. This will allow for businesses and corporations to market this software for general voting purposes which will generate revenue.

1.5 References

Use Case

Voting Software Document

IEEE Template

2. Overall Description

2.1 Product Perspective

This product is developed to provide a voting system for users to utilize. It is to ensure that the voting process is simplistic, effective, and unbiased. Built from the ground up, these voting systems can be operated by a user to tally votes through the instant-runoff form of voting and the closed party list method of voting. It is developed to run on Windows, Mac OS, and Linux.

2.2 Product Functions

- Scan CSV file into system
- Determine how to resolve a tie
- Determine an Instant-Runoff popularity winner
- Prompt and store needed information from the user
- Create audit file (.txt)
- Transfer file with other systems
- Calculate IR
- Handle both types of voting
- Read and parse file
- Determine what type of voting is to be done
- For IR voting, audit file shows the order of removal of candidates
- Calculate CPL
- Display results to terminal

2.3 User Classes and Characteristics

Government officials: use the software to process votes and elect a candidate.

2.4 Operating Environment

The environment that this will be running on is the CSE Labs machines. The hardware platform is the machines in the CSE Labs. The operating system is Linux. The program will be written in Java, and the version number is 17.0.5. The application to run the Java program can be either VS code, Eclipse, or IntelliJ. Windows, Mac OS, and Linux.

2.5 Design and Implementation Constraints

This program is designed for anyone, but would have to be federally approved for federal elections.

2.6 User Documentation

A short text file will be included with directions of how to load and operate the program.

2.7 Assumptions and Dependencies

The user is assumed to know that the inputted file should be a CSV file – otherwise the program will not work as intended. The program is also developed in Java, so Java needs to be installed on the user's device.

3. External Interface Requirements

3.1 User Interfaces

The only user interface is text in the standard terminal, there will not be any GUI or on screen menus. Any time input is needed from the user, the terminal will prompt the user, and the user will be able to type directly into the terminal. All error messages will also be printed in text to the terminal.

3.2 Hardware Interfaces

The program is intended to be run on a CSE lab machine, which has 32 GB of RAM, and run on Linux Ubuntu 20.04. Any computer system that is not a CSE labs machine is not guaranteed to work; but must at least be capable of compiling and running Java programs from source files and class files.

The system must also have enough memory and RAM to store and process the input CSV file containing the votes, as well as space to store the generated audit file.

3.3 Software Interfaces

This software requires the latest release of Java to be installed on the system. The intended operating system is the CSE lab machines, see section 3.2 for more information about CSE lab machines.

3.4 Communications Interfaces

The system should have a secure HTTPS internet connection, only to download any updated releases of the software or Java, when released. Running the software itself does not require internet connection.

4. System Features

Name	Scan file into system
ID	UC_001

Description	Program receives file and processes it
Actors	System, user
Organizational Benefits	Helps the organization make more money because the process is easier for the users -> happy customers.
Frequency of Use	Multiple times during this year for election officials more often for programmers, will be run each time that the user does not input the correct file name
Triggers	The user will prompt the program by inputting a file name.
Preconditions	The user is able to input the file name, and the file that is being scanned in exists with no bugs. No checks to validate access or credentials of anysort. User guarantee that the file is in the same location as the program, If open by someone else, two people can't open it Must have read ability but not write.
Postconditions	File has been opened, ready to be processed. Return success or failure signals.
Main Course	1: System prompts user to input a file to scan 2: User can enter a file name 3: System will open the entered file name (see EX1)
Alternate Course	
Exceptions	EX1 The entered file name does not exist 1: Alert user that file name is invalid 2: Return to main course step 1

Name	Determine how to resolve a tie
ID	UC_002
Description	The program must declare one winner, so if there is a tie, that should be resolved with a fair coin toss.

Actors	System engineer, test engineer, election official
Organizational Benefits	Helps us keep the election fair and unbiased, to maintain the integrity of the election.
Frequency of Use	Every time there is a tie during an election results calculation.
Triggers	During results calculation, when two or more candidates have the same number of votes.
Preconditions	Two have the same number of votes. We have a list containing names of the tied candidates.
Postconditions	A winner has been decided.
Main Course	1: System has a list of candidates who are tied 2: Do a one fair coin toss decides winner 3: System returns the results of the tie-breaker
Alternate Course	
Exceptions	

Name	IR Popularity winner
ID	UC_003
Description	If there is not a clear majority (one person with *more than* 50.0%) in IR voting, then popularity wins after all votes have been handed out.
Actors	System
Organizational Benefits	Following the rules set by the customer -> satisfied customer gets our organization more money.
Frequency of Use	Every time there is not a clear majority during IR voting.
Triggers	During IR voting calculation, there is never one candidate with more than 50% (ie, at least 50.1%).
Preconditions	Voting system is IR No candidate has a clear majority We have all the data of the votes for each candidate.
Postconditions	A winner has been decided.
Main Course	1: Determine which candidate has the most votes (See AC1) 2: System returns the results
Alternate Course	AC1: There is a tie for most votes 1: Create a list of tied candidates 2: Use UC_002 to resolve the tie 3: Return to Main Course step 2
Exceptions	

Name	Get needed information, and maybe prompt the user for extra information.
ID	UC_004
Description	If info can be extracted from the head of the input file, then get the information from there. Otherwise, if more info is needed, prompt the user for more info. The needed information is type of voting, number of parties, the parties names, number of seats, number of ballots.
Actors	System, User
Organizational Benefits	Make the system easier to use, and it helps maintain accuracy.
Frequency of Use	Every time a file is being processed
Triggers	A file is being processed, and we are getting additional information from it, before counting the ballots.
Preconditions	We have a valid file that is able to be processed. We know what information will be needed (type of voting, number of parties, the parties names, number of seats, number of ballots).
Postconditions	We have all the needed extra information, stored in variables and/or lists.
Main Course	1: Scan the first line of the file to determine type of voting (see AC1) 2: Scan second line of the file to determine number of parties (see AC2) 3: Scan third line of the file to get the list of party names (see AC3) 4: For security, ensure the number of party names (from step 3) equals the expected number of parties (from step 2) (See EX6) 5: Scan fourth line of the file to get the number of seats (see AC4) 6: Scan fifth line to get the number of ballots (see AC5) 7: Gathered information is stored in variables/lists.
Alternate Course	AC1: File does not provide a valid type of voting 1: Prompt user to input type of voting (IR or CPL) 2: User can input a voting type (see EX1) 3: Continue to Main Course step 2 AC2: File does not provide a valid number of parties 1: Prompt user to input the number of parties 2: User can input a number (see EX2) 3: Continue to Main Course step 3 AC3: File does not provide a valid list of party names

	1: Prompt user to input the list of party names, separated by commas. 2: User can input list of candidates (see EX3) 3: Continue to Main Course step 4 AC4: File does not provide a valid number of seats 1: Prompt user to input the number of seats 2: User can input a number (see EX4) 3: Continue to Main Course step 6 AC5: File does not provide a valid number of ballots 1: Prompt user to input the number of ballots 2: User can input a number (see EX5) 3: Continue to Main Course step 7
Exceptions	EX1: User does not enter a valid voting type 1: Alert user that entered voting type is invalid 2: Return to AC1 step 1 EX2: User does not enter a valid number 1: Alert user that entered number is invalid 2: Return to AC2 step 1 EX3: User does not enter a valid list of candidates 1: Alert user that entered list is invalid 2: Return to AC3 step 1 EX4: User does not enter a valid number 1: Alert user that entered number is invalid 2: Return to AC4 step 1 EX5: User does not enter a valid number 1: Alert user that entered number is invalid 2: Return to AC5 step 1 EX6: Number of entered party names does not match expected number of parties 1: Alert user that an error has occurred. 2: Print to terminal the expected number of parties, and their entered list of parties 3: Terminate program, it is unsafe to continue running with conflicting information.

Name	Create audit file

ID	UC_005
Description	If the program runs correctly, it produces an audit file that contains the correct information, is a txt, and has a title that is election type + date.
Actors	System
Organizational Benefits	Helps organization produce easily readable audit file done automatically
Frequency of Use	Everytime a file is processed without error
Triggers	After a file is processed without error
Preconditions	A file is being processed without error. The file is a valid file.
Postconditions	A file has been returned with the prerequisite title and a .txt
Main Course	1: Program builds audit file (EX1) 2: Audit file is returned (EX2)
Alternate Course	AC1: Ask for file again 1: Prompts user to input file name again
Exceptions	EX1: Building audit file fails 1: alert user that build fails(AC1) EX2: File name is incorrect 1: alert user that fail name is incorrect (AC1)

Name	Results of the election can be shared with media personnel(transfer file)
ID	UC_006
Description	Ability to transfer the file to the media.
Actors	User
Organizational Benefits	Allows User to easily transfer files to media
Frequency of Use	Whenever media needs to see results, potentially everytime program is run
Triggers	User triggers to send results to media
Preconditions	File has been processed and an audit file has been created
Postconditions	Media has received the file with no errors
Main Course	1: Obtained an audit file from the program (EX1) 2: User is able to send file to media (AC1, EX2)
Alternate Course	AC1: User can send to many different media 1: User will decide who to send audit file to
Exceptions	EX1: File is not obtained 1: Reprompt program and rerun EX2: File is not sent properly 1: System notifies user that an error has occurred 2: Reprompt program and rerun

Name	Calculate Instant-Runoff
ID	UC_007
Description	Program eliminates the candidate appearing as the first preference on the fewest ballots. If only one candidate remains, elect the candidate and stop; otherwise, go redo step one.
Actors	System
Organizational Benefits	Gives results
Frequency of Use	Many times in 1 session
Triggers	Once voting stops
Preconditions	File is processed Voting has stopped
Postconditions	Returns the winner
Main Course	1. Program eliminates candidate appearing as the first preference on the fewest ballots 2. If one remains, elect them 3. If more than one remains, repeat step 1
Alternate Course	AC1: Return error
Exceptions	EX1: Calculating fails 1. Throw “failure.” EX2: File name incorrect 1. Throw “name incorrect.”

Name	One Program to Handle Both Types of Voting
ID	UC_008
Description	Program can use either IR (instant-runoff) or CPL (Closed Party List) to determine the outcome based upon user input.
Actors	System, User
Organizational Benefits	Offers multiple procedures
Frequency of Use	Once
Triggers	User selects which style
Preconditions	User can operate machine Machine can run both styles of voting
Postconditions	System shows the results
Main Course	1. System prompts user with which system they would like to use 2. User picks 3. Runs desired method
Alternate Course	AC1: User does not pick 1. Machine waits for choice 2. Times out after none given
Exceptions	EX1: File name incorrect 1. Throw “name incorrect.”

Name	Read and Parse File
ID	UC_009
Description	System can take in and parse the file given to it by the user.
Actors	System, User
Organizational Benefits	Starts program
Frequency of Use	Once
Triggers	User input
Preconditions	User can input a file of the correct extension
Postconditions	File passes without error
Main Course	1. User inputs file 2. File is correct extension 3. File gets read in by system 4. System parses together information from its findings
Alternate Course	AC1: User does nothing 1. Terminate
Exceptions	EX1: User enters wrong file type 1. Reject and ask for the right type

Name	Determine what type of voting is going to be done
ID	UC_010
Description	Read in the file, the type of election will be stored on the first line of the file that contains the ballots.
Actors	System
Organizational Benefits	By using a system to determine the type of voting that needs to be done, organizations can make informed decisions that are based on a transparent and fair process. This system can automate the process of determining the type of voting that is required, reducing the time and effort required to manually assess each situation.
Frequency of Use	Multiple times during the year at normal election times and special elections.
Triggers	User enters the filename to be parsed.
Preconditions	User enters the correct filename to be parsed.
Postconditions	The type of voting has been determined and system works on the chosen voting system.
Main Course	1. User inputs a file 2. System verifies that the file has correct extension 3. File gets read by the system 4. System parses the first line of the file
Alternate Course	AC1 User does nothing 1. Program terminates
Exceptions	EX1 User enters incorrect filename 1. Reprompt the user for the correct filename

Name	For IR voting, audit file shows the order of removal of candidates
ID	UC_011
Description	Program eliminates the candidate appearing as the first preference on the fewest ballots. The candidates with the fewest ballots are added to the audit file.
Actors	Programmer, Tester, System
Organizational Benefits	The order of removal helps to establish accountability among the election officials and provides a clear trail of the events that took place during the voting.
Frequency of Use	Multiple times during the year at normal election times and special elections.
Triggers	After a file is processed without error and the voting ends.
Preconditions	A file is being processed without error. The file is a valid file.
Postconditions	Returns the order of removal of candidates in the audit file.
Main Course	1. Program eliminates candidate appearing as the first preference on the fewest ballots 2. The candidates with the fewest ballots are added to the audit file. 3. If more than one remains repeat step one
Alternate Course	AC1: Ask for file again 1: Prompts user to input file name again
Exceptions	EX1: Building audit file fails 1: alert user that build fails(AC1) EX2: File name is incorrect 1: alert user that fail name is incorrect (AC1)

Name	Calculate CPL
ID	UC_012
Description	The candidates are selected based on their position on the party's list and the number of seats won by the party in the election.The winning candidates are then selected from the party's list based on their position on the list.
Actors	System
Organizational Benefits	The use of a voting system can increase transparency in decision-making, providing a clear and auditable record of how decisions were made.
Frequency of Use	Multiple times during the year at normal election times and special elections.
Triggers	CPL is calculated whenever the voting ends.
Preconditions	File has been read in and the voting has ended
Postconditions	Returns the list of candidates elected from each party
Main Course	1. Program calculates the quota by taking the total number of valid votes and dividing it by the number of seats. 2. The quota is then divided into the vote that each party receives and the party wins one seat for each whole number produced. 3. After this first allocation of seats is complete than the remainder numbers for the parties are compared and the parties with the largest remainders are allocated the remaining seats.
Alternate Course	AC1: Return error
Exceptions	EX1: Calculating fails 2. Throw “failure.” EX2: File name incorrect 2. Throw “name incorrect.”

Name	Display results to terminal
ID	UC_013
Description	Display the winner and potentially other information to the terminal
Actors	System, user, program
Organizational Benefits	No audit need to check audit file, able to check winner quickly
Frequency of Use	Anytime a file is processed
Triggers	After a file has been processed
Preconditions	A file has been processed with no errors
Postconditions	Results(varying level of information) will be displayed in the terminal
Main Course	1: File is processed with no errors (EX1) 2: Results appear in terminal (AC1, EX2)
Alternate Course	AC1: Can change kinds of information showing up 1: User can select what kind of information appears in terminal
Exceptions	EX1: File is errored out 1: Reprompt program and rerun EX2: Information appearing in terminal is not correct 1: System notifies user that an error has occurred 2: Reprompt program and rerun

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The voting system should have a fast and efficient ballot processing mechanism that allows it to process large numbers of ballots in a short amount of time. The system should be designed in such a way that it can handle high volumes (~100,000) of ballots simultaneously and still be able to complete the processing in under 4 minutes. This requirement is crucial as it ensures that the results of the election are obtained quickly, reducing the wait time for the stakeholders.

5.2 Safety Requirements

No safety requirements as mentioned in the project document.

5.3 Security Requirements

Voting System does not have any security requirements and thus any type of user can use it without any additional privileges.

5.4 Software Quality Attributes

1. The system should be reliable and available for use at all times, with a minimal amount of downtime. This will help to ensure that voters are able to cast their votes whenever they need to.
2. Usability: The system should be easy to use and understand for voters, with clear instructions and user-friendly interfaces. This will help to ensure that voters are able to cast their votes quickly and efficiently.
3. Accuracy: The system should accurately tally votes and produce correct results. This will help to ensure that the election outcome is fair and accurate.
4. Scalability: The system should be able to handle a large number of voters and votes, and be able to accommodate growth and changing demand over time.

5.5 Business Rules

1. Vote counting accuracy: The system should accurately count the votes and provide a reliable tally of the results.
2. Auditability: The system should provide a transparent and auditable voting process.
3. Tamper-proof: The system should be tamper-proof and prevent any attempts to manipulate the results.

6. Other Requirements

Requirements not defined in the SRS are database requirements, legal requirements, and potentially expanding this system. The database to store each one of these voting files and audit files will have to be both large enough to hold them all and secure enough to be shielded against people. The legal implications of creating a software that will count votes is always difficult to gauge. The software will have to go through tons of scrutiny and tests to be verified as ready to be used in real world applications of votes. There is a potential to expand this software into including other government tasks. Since the system will be theoretically secure already, it could be expanded to include other government tasks that need a large database and secure servers.

Appendix A: Glossary

IR: Instant Runoff voting

CPL: Closed Party Listing voting

CSV: Comma Separated Values file, used to import all the ballots and additional information about the election

Audit File: A text file used to archive information about the election, including which party won, and what order the other parties were eliminated in.

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

1: Potentially more references to more system features
2: UML diagrams can be added