
Software Requirements Specification

for Voting System Software

Version 1.0 approved

Prepared by:

- 1. Bryan Yen Sheng Lee, lee03627, Team 4**
- 2. Cedric Tan Yee Shuen, tan00205, Team 4**
- 3. Sherryl Ooi Shi Tyng, ooi00004, Team 4**

University of Minnesota, Twin Cities

16th February 2023

Table of Contents

1. Introduction	1
1.1 Purpose	1
1.2 Document Conventions	1
1.3 Intended Audience and Reading Suggestions	1
1.4 Product Scope	2
1.5 References	2
2. Overall Description	3
2.1 Product Perspective	3
2.2 Product Functions	4
2.3 User Classes and Characteristics	4
2.4 Operating Environment	4
2.5 Design and Implementation Constraints	5
2.6 User Documentation	5
2.7 Assumptions and Dependencies	5
3. External Interface Requirements	5
3.1 User Interfaces	5
3.2 Hardware Interfaces	11
3.3 Software Interfaces	11
3.4 Communications Interfaces	12
4. System Features	15
4.1 System Startup	15
4.2 File Identification	15
4.3 File Naming Convention	16
4.4 Open File	16
4.5 Read the File and Perform CSV Processing	17
4.6 Prompt User for Information	17
4.7 Create an Audit File for Instant Runoff	18
4.8 Create an Audit File for Closed Party List	18
4.9 Document Have At Least One Person Ranked	19
4.10 Run Two Types of Elections	19
4.11 Ensures File Structure Does Not Change	20
4.12 Fair Coin Toss	20
4.13 Pool Coin Toss	21
4.14 Check for Majority	21
4.15 Check if Candidates Are Listed In A Particular Order	22

4.16 Disallow Write-In Candidates	22
4.17 Display Election Results	22
5. Use Cases	23
5.1 UC_001 System Startup	23
5.2 UC_002 File Identification	24
5.3 UC_003 File Naming Convention	24
5.4 UC_004 Open File	25
5.5 UC_005 Read the File and Perform CSV Processing	26
5.6 UC_006 Prompt User For Information	27
5.7 UC_007 Create An Audit File For Instant Runoff	28
5.8 UC_008 Create An Audit File For Closed Party List	29
5.9 UC_009 Document Have At Least One Person Ranked	30
5.10 UC_010 Run Two Types of Elections	31
5.11 UC_011 Ensures File Structure Does Not Change	32
5.12 UC_012 Fair Coin Toss	32
5.13 UC_013 Pool Coin Toss	33
5.14 UC_014 Check For Majority	34
5.15 UC_015 Check If Candidates Are Listed In A Particular Order	35
5.16 UC_016 Disallow Write-In Candidates	36
5.17 UC_017 Display Election Results	37
6. Other Nonfunctional Requirements	38
6.1 Performance Requirements	38
6.2 Safety Requirements	39
6.3 Security Requirements	39
6.4 Software Quality Attributes	40
6.5 Business Rules	41
Appendix A: Glossary	43
Appendix B: Analysis Models	44

Revision History

Name	Date	Reason For Changes	Version
Voting System	February 16, 2023		1.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the software product, Voting System Version 1.0 that is capable of performing these two types of voting system: (1) Instant Runoff Voting system (2) Party List Voting using Closed Party List. This document will explain the purpose and features of the voting system, the interfaces of both systems, what both systems will do and the constraints on under which it must operate. This document is intended for election officials, testers, and potential developers of the system.

Moreover, the purpose of this document is to guarantee that all the user requirements will be fulfilled by the software that the developers create. The project requirements that were given to the development team are described and expanded upon in this document. By clearly stating these requirements, it is possible to address any potential misunderstandings at an early stage where the cost of making modifications is still minimal. Once the project is finalized, people in charge of updating or maintaining the system will find this document to be a helpful resource for future usage.

1.2 Document Conventions

This document was created based on the IEEE template for System Requirements Specification Documents. In general, the IEEE uses Times New Roman font size 11 throughout the document for text and italics for comments. Document text is all single-spaced and maintains the 1" margin in this document. The main titles of the body are written in Times New Roman with a font size of 18 and subsection titles are written in Times New Roman with a font size of 14.

1.3 Intended Audience and Reading Suggestions

The types of readers that the document is intended for are as follows:-

- **State and local election officials** - Election authorities from different jurisdictions who are using voting systems in their areas.
- **Software testers** - For the purpose of analysis of software and systems testing in support of the national certification testing process, authorized testing laboratories will use this data to design test plans and procedures as well as perform automated and manual testing of the system.
- **Potential designers and developers** - Voting system designers and developers must ensure their products fulfill the requirements and achieve desired goals as well as minimizes the development cost.

The suggested sequence for reading the documentation will be as follows:-

1. Project functionality which includes functional and non-functional requirements of the system
2. Software, Hardware interfaces, and user interfaces
3. System Accuracy, Security, and performance criteria
4. Goals of implementation and implementation issues (risks), if any

1.4 Product Scope

The scope of Voting System Version 1.0 will cover the capability to perform these two types of voting system: (1) Instant Runoff Voting system (2) Party List Voting using Closed Party List. The system allows election officials to interact with election results that are compiled into a comma separated values (CSV) file. The system will process the comma separated values (CSV) file in a way that is readable and makes it faster to handle. Users can be prompted for any needed information in a short period of time without going through the files individually. One of the organizational benefits is transparency. An audit file provides a clear record of all votes cast, enabling stakeholders to verify the accuracy of the election results. This increases transparency and public trust in the election process. The system has the ability to run two types of elections in a machine. It provides a user-friendly interface as well as lowers the cost of purchasing, installation, management, and maintenance of large amounts of hardware. In addition, the system is designed to handle ties by using a common unbiased random bit which instills trust and integrity in the voting system.

The system will be designed to maximize the election official's productivity by introducing tools to help automate the counting of the ballots and the election process, which would otherwise need to be done manually. The system will satisfy the election official's requirements simply by maximizing the productivity and job efficiency of the election officials while remaining simple to understand and utilize. This product scope and structure of the final set of guidelines will be laid out in this document, which will also facilitate the continued work on the Voting System Version 1.0.

1.5 References

- Editor. (2020, February 11). What is API: Definition, Types, Specifications, Documentation. AltexSoft.
<https://www.altexsoft.com/blog/engineering/what-is-api-definition-types-specifications-documentation/#:~:text=Operating%20systems%20APIs,.API%20and%20kernel%20internal%20API>
- FairVote (2023). Voting System Website.
<https://fairvote.org/>
- Federal Election Commission (2002). Voting Systems Standards Volume 1. United States: Federal Election Commission
- Fowler, M. and Scott, K. (1999, August 18). UML Distilled Second Edition A Brief Guide to the Standard Object Modeling Language.
<https://pja.mykhi.org/6sem/MAS/books/Addison%20Wesley%20-%20UML%20Distilled,%202nd%20Edition.pdf>
- IEEE Template for System Requirements Specification Documents. (n.d.). Retrieved from
<https://goo.gl/nsUFwy>
- Indeed Editorial Team. (2021, May 3). 9 Nonfunctional Requirements Examples. Indeed. Retrieved from
<https://www.indeed.com/career-advice/career-development/non-functional-requirements-examples>
- Michigan State University. (2004, April 15). Software Requirements Specification Version 1.0. Retrieved from
<https://www.cse.msu.edu/~cse435/Handouts/SRSEExample-webapp.doc>
- Mitchell, C. (2022, September 5). What Is File Transfer Protocol (FTP) and What Is It Used For? Investopedia. Retrieved from

[https://www.investopedia.com/terms/f/ftp-file-transfer-protocol.asp#:~:text=File%20transfer%20protocol%20\(FTP\)%20is,computers%20or%20through%20the%20cloud.](https://www.investopedia.com/terms/f/ftp-file-transfer-protocol.asp#:~:text=File%20transfer%20protocol%20(FTP)%20is,computers%20or%20through%20the%20cloud.)

Reagan, R. T., Marie Leary, M. (2017). The Help America Vote Act. CreateSpace Independent Publishing Platform.

University of Houston Clear Lake. (n.d.). Guidelines: Business Rules. Scweb. Retrieved from https://sceweb.uhcl.edu/helm/RationalUnifiedProcess/process/modguide/md_brule.htm

Varvoutas, K. (2017, February). IEEE Software Requirements Specification for Gephi. Retrieved from https://gephi.org/users/gephi_srs_document.pdf

2. Overall Description

2.1 Product Perspective

The product, Voting Systems is a new product that will show the results when using Instant Runoff Voting system and the Party List Voting using Closed Party List in an election. The voting system has 4 active actors and a cooperating system. The actors are the developers (including the software engineers, and product managers), testers, election officials (including other authorizers), and the residents. The developers can directly access and edit the entire system for development and maintenance. The election officials can review the votes in either system, but could not edit the votes. The residents are the reviewer/reader of the entire system so that they could view the results of the election.

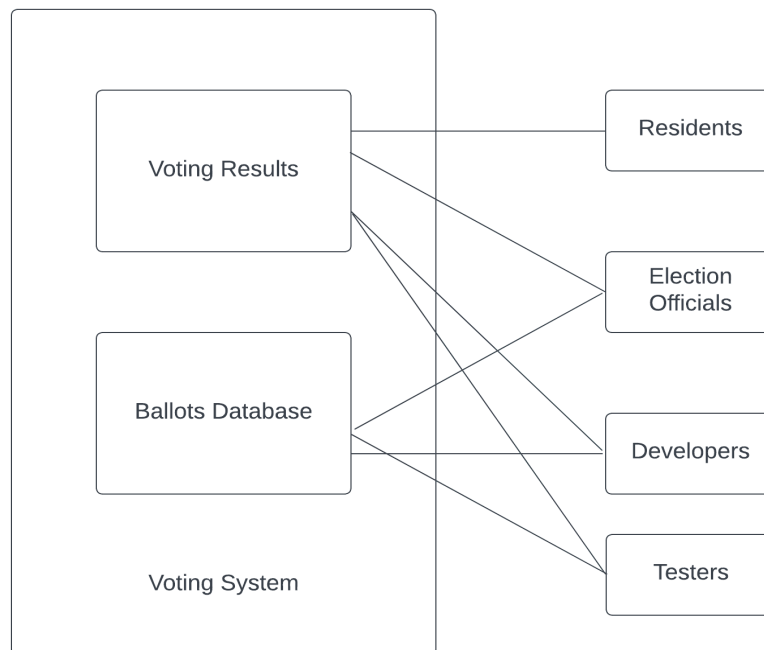


Figure 2.1: This diagram shows which actors have access to the major functions.

2.2 Product Functions

There are a few major functions in the product, including the file identification function, ballot counting function, fair coin tossing function, etc. These functions are all important as these are the base level of our system.

- **File Identification function:**

File identification functions here include several functions, including a function that helps the system recognize, open, and read in a comma delimited text file (.csv) that contains all the ballots. It also includes a function that opens, reads, and writes the data from the .csv files.

- **Ballot Counting function:**

This function is used to calculate the ranking of the ballots, to measure which party/candidate has the majority votes.

- **Fair Coin Tossing function:**

This function is used when there is a tie between the majority votes or seats.

2.3 User Classes and Characteristics

2.3.1 Developers

The developers here included the software engineers and the product managers.. They are required to design, develop, test and maintain the product. The software engineers are responsible for developing the product here, including creating all the functions with corresponding functionality. The product manager is responsible for working with the team, from the start till the end, including designing, developing, testing, and maintaining the product. They do not have access to change or edit the ballot while the voting process starts.

2.3.2 Testers

The testers here are responsible for testing the product before it is released, they should create multiple test cases that include all scenarios for testing the product. They will run the system thousands of times before the software is released to ensure that there are no defects during release.

2.3.3 Election officials

The election officials here include both state and local election officials, and anyone who gets authorized to access the systems. The authorizers here are responsible to start the voting process, once it's started, they can view the process, while the ballot should be private and cannot be viewed. They could not edit nor delete the voting as we assume all ballots are perfect. They are also responsible for printing the results of the voting and run the fair coin toss if the majority votes is a tie.

2.3.4 Residents

The residents act as the voters, the reviewers and the readers. They are responsible for voting, and reviewing the results. The votes should be included in the comma separated file (.csv) and they can review the results after being posted by the authorizers. They do not have any access to the voting systems but only to view the results at the end page.

2.4 Operating Environment

The software will be written in Java, and run on CSE-Labs computers. The computer hosting must be capable of running Java bytes. The computer should be able to run 100,000 ballots in under 4 minutes. The software should be able to run in either Linux, MacOS, or Windows that has Java installed and able to run the GUI-based Java computer programs.

2.5 Design and Implementation Constraints

The team should be designing, developing, testing, and maintaining this software within the semester in probably two months' time. They have limitations on time as they are mostly students with other courses work in the university. Due to all these constraints, it is necessary to prioritize the functionality and security over others.

The developing and testing environment is limited to the CSE Labs machines in the U of MN, and the hardware should be able to run the software in under 4 minutes. This means that the developers will not have full access to commercial systems that might be necessary to fully test this system under realistic conditions.

2.6 User Documentation

There will be a quick start guide for first-time users, including how the ballots are counted, what tools are required, and how to start the counting process. There should also be a privacy statement before the user starts the program. There will be tutorial screenshots or videos that act as user manuals, and be delivered together with the software. The users can always reach out to the developers for any assistance or questions.

2.7 Assumptions and Dependencies

- The end user is proficient in English and has basic knowledge of computers.
- Tasks and roles are predetermined.
- The election results will be managed and calculated by the system.
- Assume that there are no numbering mistakes in the file. For example, assume there are no errors in the ballots.
- The actual voting will be done separately from the voting system developed.
- There will never be more than one file given to you per election.
- Assume we will use the most up-to-date CSELabs machines

3. External Interface Requirements

3.1 User Interfaces

The voting system provides a user interface to enable election officials to access election information. It must be designed to meet the needs of the end users and support the overall goals of the software system. The voting system will primarily be used by election officials to obtain audit files, view election-related information, and break ties. Thus, the user interface of the voting system must be intuitive and easy to use. Figure 3.1 below illustrates the relationship between the interfaces:

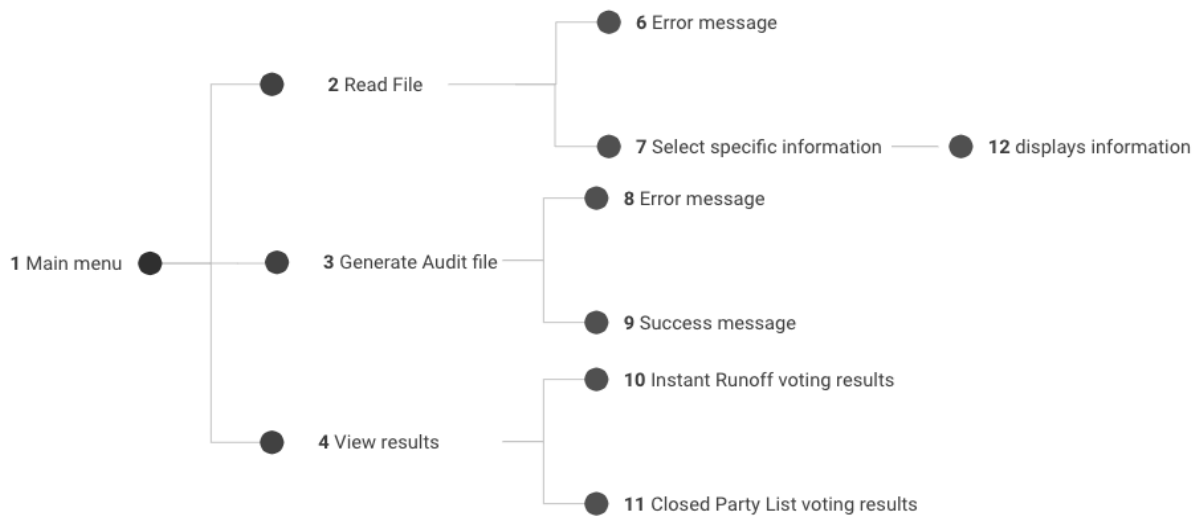


Figure 3.1: User Flow Diagram

A Graphical User Interface (GUI) is utilized in the voting system to enable users to communicate with the voting system. A GUI allows non-programmers such as election officials to interact with the voting system without needing to type commands. All aspects of the voting system will have a simple point-and-click interface using menus, drop-down lists, buttons, and all of the other components of systems with graphical user interfaces. The GUI in Java can be implemented using the javax.swing package.

3.1.1 Main Menu

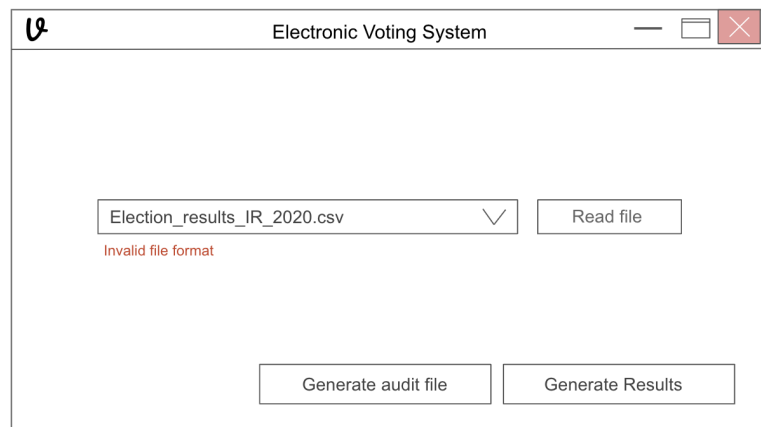


Figure 3.2: Prototype design for the main menu

In the prototype design of the main menu, the screen layout is designed to be simple. Users can interact with the drop-down list to look for files within the directory and click on the read file button when they have selected the file they are looking for. Also, after reading the file, users are given the option to view election results or generate an audit file. If the file does not match the Comma Delimited Text (CSV) format, a warning message, “invalid file format,” appears below the drop-down list. If users persistently click the “read file” button despite the warning, an error message pops up as illustrated below in Figure 3.3:



Figure 3.3: Prototype design for error message

When the file is read successfully, another drop-down element appears below the existing file search drop-down element. Here the users can select specific information from the CSV file and view it. This is illustrated in figure 3.4 below:

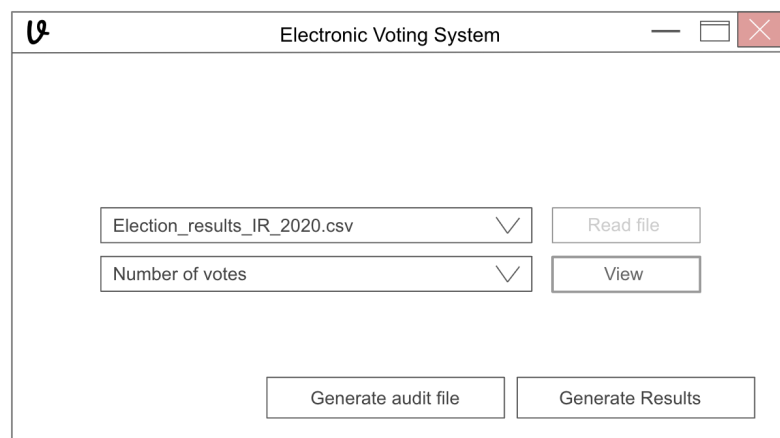


Figure 3.4: Prototype design for the main menu after successful file reading

When specific information is selected, users can select the view button and a dialog box with the selected information appears. This is illustrated in figure 3.5 below:

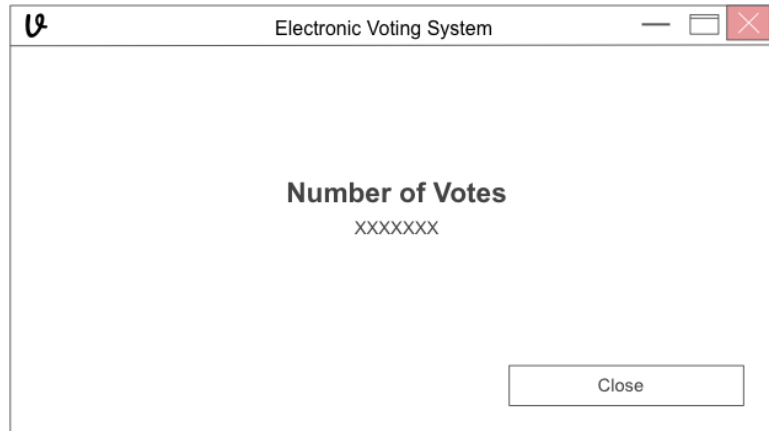


Figure 3.5: Prototype design for selected information dialog box

3.1.2 Tie-breaker

In an event where there is a tie, a tiebreaker dialog box appears; prompting the users to flip a coin and randomly select a winner. This is illustrated in Figures 3.6 and 3.7 below:

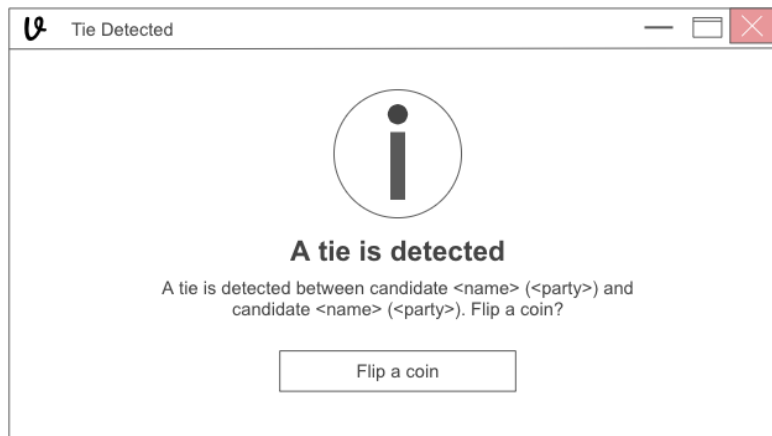


Figure 3.6: Prototype design for tiebreaker dialog box



Figure 3.7: Prototype design for winner-determined dialog box

3.1.3 Generating audit file

An audit file can be generated and downloaded. If the audit file is successfully generated, a message dialog box pops up and users are given the option to download the file or discard the file. If the audit file is not successfully generated, typically due to the incorrect labelings in the CSV file, then an error message pops up. The prototypes for the dialog boxes are illustrated below:

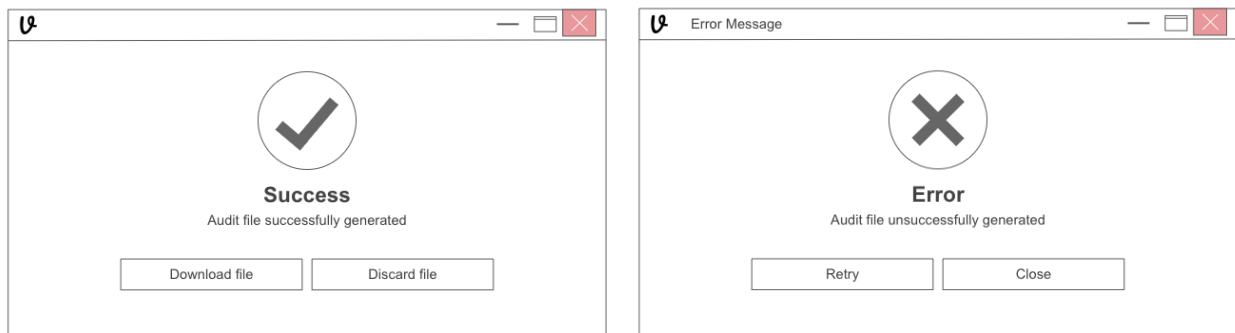
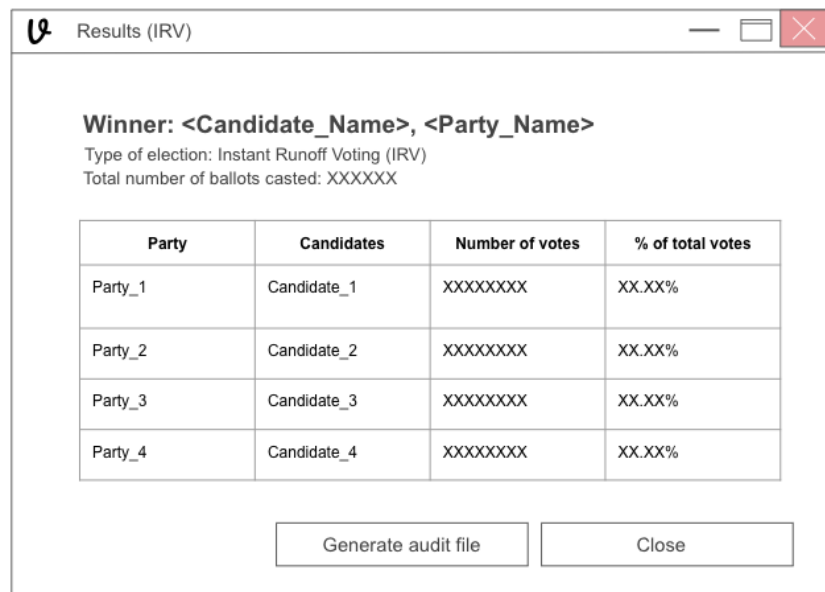


Figure 3.8: Prototype design for error and success message

3.1.3 View Results

When the user selects the view results button, a message dialog box pops up depending on the type of election. If the CSV file contains both Closed Party List Voting and Instant Runoff Voting, then both these message dialog boxes will pop up. The key difference between the Instant Runoff Voting dialog box and the Closed Party List dialog box is that all candidates are listed in the Closed Party List voting box. After displaying the results, users can either close the dialog box or generate an audit file. The diagrams below illustrate the two types of dialog boxes when displaying results:



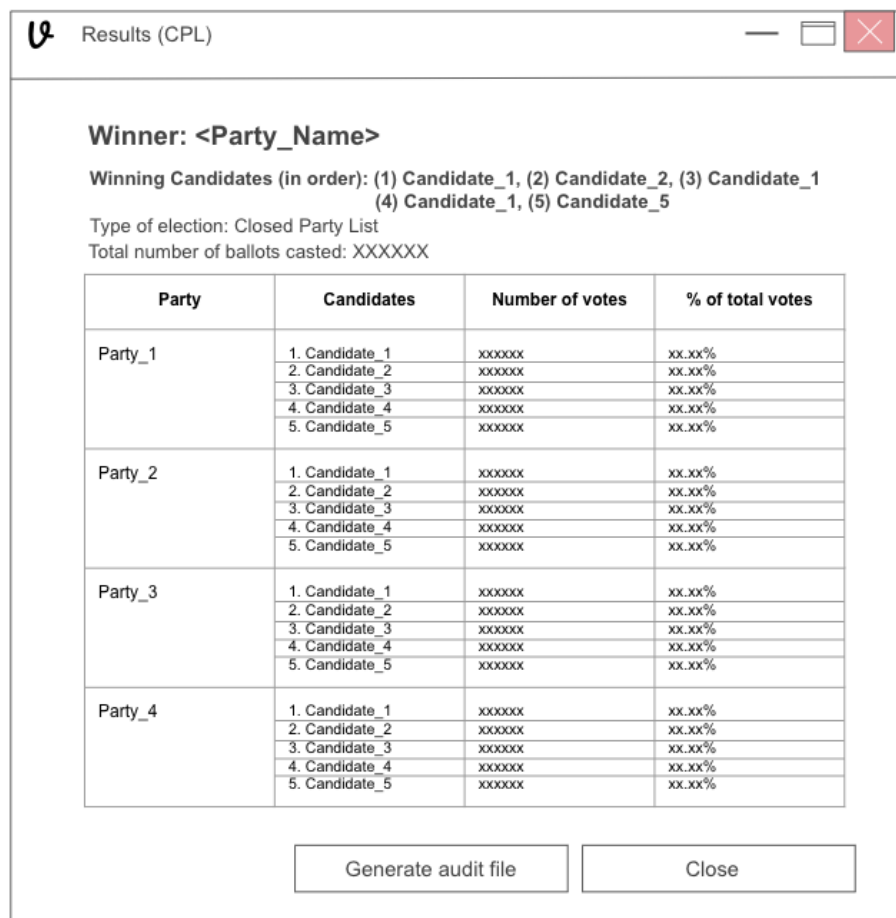
Results (IRV)

Winner: <Candidate_Name>, <Party_Name>
 Type of election: Instant Runoff Voting (IRV)
 Total number of ballots casted: XXXXXX

Party	Candidates	Number of votes	% of total votes
Party_1	Candidate_1	XXXXXXXX	XX.XX%
Party_2	Candidate_2	XXXXXXXX	XX.XX%
Party_3	Candidate_3	XXXXXXXX	XX.XX%
Party_4	Candidate_4	XXXXXXXX	XX.XX%

Generate audit file Close

Figure 3.9: Prototype design for results (Instant Runoff Voting)



Results (CPL)

Winner: <Party_Name>
Winning Candidates (in order): (1) Candidate_1, (2) Candidate_2, (3) Candidate_1 (4) Candidate_1, (5) Candidate_5
 Type of election: Closed Party List
 Total number of ballots casted: XXXXXX

Party	Candidates	Number of votes	% of total votes
Party_1	1. Candidate_1	xxxxxx	xx.xx%
	2. Candidate_2	xxxxxx	xx.xx%
	3. Candidate_3	xxxxxx	xx.xx%
	4. Candidate_4	xxxxxx	xx.xx%
	5. Candidate_5	xxxxxx	xx.xx%
Party_2	1. Candidate_1	xxxxxx	xx.xx%
	2. Candidate_2	xxxxxx	xx.xx%
	3. Candidate_3	xxxxxx	xx.xx%
	4. Candidate_4	xxxxxx	xx.xx%
	5. Candidate_5	xxxxxx	xx.xx%
Party_3	1. Candidate_1	xxxxxx	xx.xx%
	2. Candidate_2	xxxxxx	xx.xx%
	3. Candidate_3	xxxxxx	xx.xx%
	4. Candidate_4	xxxxxx	xx.xx%
	5. Candidate_5	xxxxxx	xx.xx%
Party_4	1. Candidate_1	xxxxxx	xx.xx%
	2. Candidate_2	xxxxxx	xx.xx%
	3. Candidate_3	xxxxxx	xx.xx%
	4. Candidate_4	xxxxxx	xx.xx%
	5. Candidate_5	xxxxxx	xx.xx%

Generate audit file Close

Figure 3.10: Prototype design for results (Closed Party List Voting)

3.2 Hardware Interfaces

The hardware interfaces section describes the characteristics of each interface between the software product and the hardware components of the system. However, since most of the interfaces for the voting system are software-based, the only physical equipment needed for the voting system are computers and CSE lab machines. These are the minimum requirements for the hardware components of computers and CSE lab machines to be able to efficiently run the voting system:

- (i) Disk Space Requirements (Hard Disk / Solid State Drive):** A minimum of 10 Gigabytes (GB) of storage space is required
- (ii) Memory Requirements (Random Access Memory (RAM)):** A minimum of 2 Gigabytes (GB) of RAM
- (iii) Processor Requirements (Central Processing Unit (CPU) processor):** Minimum multi-core processor with a clock speed of 1 GigaHertz (GHz)
- (iv) Other Requirements:**
 - **Graphics Card:** A graphics card or integrated graphics with support for OpenGL or DirectX for displaying user interfaces of the voting system.

3.3 Software Interfaces

The software interfaces section describes the interactions between the voting system and other specific software components. These software components include:

3.3.1 Operating Systems:

The voting system shall be compatible with all versions of Windows, macOS, and Linux. Also, it must be compatible with the most up-to-date CSE Lab machines.

3.3.2 Tools and Libraries:

The voting system shall be compatible with Integrated Development Environment (IDE) which helps developers to develop software code more efficiently. This voting system shall use Visual Studio Code as its IDE and code editor. Also, the voting system will utilize Git and GitHub for a source control repository. IDEs allow developers to write their codes and GitHub acts as a code hosting platform for collaboration between developers. Thus, JAVA files will be shared between Visual Studio Code and GitHub. In JAVA, Java Foundation Classes library (javax.swing package) shall be utilized to create a Graphical User Interface (GUI) for the voting system.

3.3.3 Application Programming interfaces (API):

The voting system shall follow established API protocols to ensure compatibility and ease of integration with other software components. APIs are mechanisms that allow the voting system to make requests for services with other software components (Editor, 2020). Here are the types of APIs which will be utilized in the voting system:

- **Operating Systems APIs:**
 - **Windows:** Windows API will be utilized in Windows to communicate with the voting system.
 - **macOS:** Cocoa (API) will be utilized in mac OS to communicate with the voting system
 - **Linux:** Linux API (kernel user-space API and kernel internal API) will be utilized in Linux to communicate with the voting system.
- **Remote APIs:** For CSE Lab machines with Unix/Linux environments, voting systems can be edited remotely using Visual Studio Code (VS Code) by using a Secure Shell (SSH) client extension.

3.4 Communications Interfaces

All communications functions required by the voting system must be designed to ensure secure and reliable communication between the voting system and its external entities; the election officials, the software developers, and the testers. These functions include message formatting, communication standards, communication security, encryption issues, data transfer rates, and synchronization mechanisms.

3.4.1 Message formatting for Voting System:

The format of messages exchanged between the voting system and its external entities must be defined and standardized. The voting system will read Comma Separated Values (CSV) files where each row is separated by a newline and the format in which the voting information is provided in the CSV file will adhere to a given format. Data sent to the voting system will conform to a pre-defined format to enable the voting system to detect any tampering with data in transit; if detected, an error pops up and the data will be discarded. The user will then have to re-open the Comma Delimited Text (CSV) file again. The message formatting is as illustrated below in 3.4.1.1 and 3.4.1.2:

3.4.1.1 Message formatting for Instant Runoff 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
```

Figure 3.10: Format for Instant Run-off (IR) voting

1st Line: IR which stands for Instant Run-off

2nd Line: The number of candidates

3rd Line: The candidates separated by commas

4th Line: The number of ballots in the file

3.4.1.2 Message formatting for Closed Party List CSV file:

```
CPL
6
Democratic, Republican, New Wave, Reform, Green,
Independent
Foster, Volz, Pike
Green, Xu, Wang
Jacks, Rosen
McClure, Berg
Zheng, Melvin
Peters
3
9
1,,,,
1,,,,
,1,,,,
,,,1,
,,,1
,,,1,,
,,,1,,
1,,,,
,1,,,,
```

Figure 3.11: Format for Close Party List (CPL) voting.

1st Line: CPL for closed party listing

2nd Line: The number of parties

3rd Line: The parties separated by commas

Candidates lines: Candidates from the order of each party are ranked in order

After candidates lines: The number of seats

After number of seats line: The number of ballots.

3.4.2 Communication Standards

In terms of communication standards, the voting system shall adhere to industry-standard communication protocols — Transmission Control Protocol / Internet Protocol (TCP/IP), Secure Socket Layer (SSL), and File Transfer Protocol (FTP) — to ensure compatibility with other systems and secure communication. These protocols are vital because they safely permit the uploading and downloading of files on the computer over the internet.

3.4.2.1 Transmission Control Protocol / Internet Protocol (TCP/IP)

The TCP/IP is a communications standard that enables the voting system and servers to exchange files over a network. The voting system will be sending audit files and receiving CSV files from servers.

3.4.2.2 Secure Socket Layer (SSL)

The SSL is an encryption protocol that makes an internet connection secure and safeguards voting data being transferred between the voting systems and other systems. All communication between components in the voting system will be achieved by sending raw byte data over SSL connections. This is to prevent any fraudulent changes from being made to voting data during transit.

3.4.2.3 File Transfer Protocol (FTP)

The FTP provides a way for users to download, upload, and transfer CSV files on the internet or between computer systems. This is crucial for the users of the voting system to securely download CSV files either from the host website or through accessing FTP servers (Mitchell, 2022).

4. System Features

4.1 System Startup

4.1.1 Description and Priority).

The use case [UC_001 System Startup](#) describes the actions performed by the system during start-up and initialization phase. It is of high priority as if the system fails to startup, it will cause an urgent problem that blocks the system use until the issue is resolved.

4.1.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. Once the switch on the panel is turned to the “on” position by the operators, the system is activated. The system sends a message to all external physical units to check if they are ready to start. The physical units send a message back to the system indicating that they are ready to start operating. The welcome screen is displayed. The system is deactivated when the operator turns the switch to the “off” position. If the system detects an error and a transmission failure occurs, the system will be put into the emergency stop mode and will exit automatically.

4.1.3 Functional Requirements

Other system features include:

[UC_002: File Identification](#)

[UC_003: File Naming Convention](#)

[UC_004: Open File](#)

[UC_005: Read the File and Perform CSV Processing](#)

[UC_006: Prompt User for Information](#)

[UC_007: Create an Audit File for Instant Runoff](#)

[UC_008: Create an Audit File for Closed Party List](#)

[UC_009: Document Have At Least One Person Ranked](#)

[UC_010: Run Two Types of Elections](#)

[UC_011: Ensures File Structure Does Not Change](#)

[UC_012: Fair Coin Toss](#)

[UC_013: Pool Coin Toss](#)

[UC_014: Check for Majority](#)

[UC_015: Check If Candidates Are Listed In A Particular Order](#)

[UC_016: Disallow Write-In Candidates](#)

[UC_017: Display Election Results](#)

4.2 File Identification

4.2.1 Description and Priority

The use case [UC_002 File Identification](#) describes how the system recognizes the type of the file. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.2.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the actor runs the program to process the file from a directory. The system will read in the first line of the file that contains the type of election. The type of the file is identified by the system and the file is ready to be named. If the system fails to read in the first line of the file that contains the type of election, the file cannot be identified and an error message is displayed on the screen.

4.3 File Naming Convention

4.3.1 Description and Priority

The use case [UC_003 File Naming Convention](#) describes how the name of the file is being named by the system. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.3.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the actor runs the program to process the file from a directory.

The name of the file contains the following information:

- scope of the election (national, state, local)
- type of the voting system (Instant Runoff or Closed Party List)
- the date of the election being held (DD-MM-YEAR)

The name of the file is passed into the program as a command line argument or the system will ask for the name of the file within the program itself. If the election is conducted multiple times on the same day, the system will overwrite the ballots in the same file which have the same name.

4.4 Open File

4.4.1 Description and Priority

The use case [UC_004 Open File](#) describes how the system opens a file which contains ballots that were cast online. It is of high priority as if the file fails to be open, it will cause an urgent problem that blocks the system use until the issue is resolved as the system will not be able to identify the file type and determine the type of election to be run.

4.4.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the actor runs the program to process the file from a directory. The file is successfully being brought into the program and the system successfully opens the file. If the file fails to be brought into the system, an error message will be sent to the user. If the system fails to open the file, an error message will be displayed on the screen. If the exit option is chosen by the user, the task ends and the user is returned to the system menu.

4.5 Read the File and Perform CSV Processing

4.5.1 Description and Priority

The use case [UC_005 Read the File and Perform CSV Processing](#) describes the actions performed by the system by reading in a file and performing CSV processing. It is of high priority as if the system fails to read in the file and perform CSV processing, it will cause an urgent problem that blocks the system use until the issue is resolved. It will cause a failure to run a complete election and display the election results.

4.5.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the CSV file is being opened by the system.

Process CSV file for Instant Runoff Voting

1. A valid comma separated values (CSV) file is imported and successfully opened by the system.
2. The UC_002 File Identification use case identifies that this is an Instant Runoff Voting election by reading in the first line of the CSV file that contains the type of the elections.
3. The system will perform CSV processing by reading in all the information from the file generated for Instant Runoff Voting.
4. The file generated for Instant Runoff Voting will be used for message formatting which follows the format as stated in 3.4.1.1 Message Formatting for Instant Runoff Voting.

Process CSV file for Closed Party List Voting

1. A valid comma separated values (CSV) file is imported and successfully opened by the system.
2. The UC_002 File Identification use case identifies that this is a Closed Party List Voting election by reading in the first line of the CSV file that contains the type of the elections.
3. The system will perform CSV processing by reading in all the information from the file generated for Closed Party List Voting.
4. The file generated for Closed Party List Voting will be used for message formatting which follows the format as stated in 3.4.1.2 Message Formatting for Closed Party List Voting.

Alternative Courses:

If the system fails to read in the file, an error message will be sent to the user and displayed on the screen. If the system fails to perform CSV processing, an error message will be displayed on the screen.

4.6 Prompt User for Information

4.6.1 Description and Priority

The use case [UC_006 Prompt User for Information](#) illustrates that the user can be prompted for any needed information that cannot be extracted from the file when the program is executed. It is of low priority as it is not one of the core functionality that the voting system is explicitly supposed to perform and is a compromise. The problem should be fixed if time permits but can be postponed.

4.6.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the user is prompted for information. A GUI pops up with a textbox to type in the information they require from the file. Election officials use it to obtain the required information from the file. If the information type into the textbox is not recognized in the file, a message indicating that it is not found will be sent to the user.

4.7 Create an Audit File for Instant Runoff

4.7.1 Description and Priority

The use case [UC_007 Create an Audit File for Instant Runoff](#) describes how an audit file is produced with the election information at the time for Instant Runoff Voting System. The audit file contains election information such as type of voting (IR for Instant Runoff), number of candidates, candidates, and number of ballots. The audit file should be able to show calculations, how many votes a candidate had, the winner(s), and show how the election progressed so that the audit could replicate the election itself. An auditing should be able to follow the order of the ballots being assigned to the candidates. It should also show the order of candidates in Instant Runoff and what ballots were redistributed. The audit file should show all of the steps. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.7.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the user prompts the system to produce an audit file for the Instant Runoff Voting System. The user prompts the system to produce an audit file. The system gathers all of the necessary data for the audit file. The collected data is collected and processed. The processed data is then stored securely in an audit file. The audit file is then analyzed to ensure that it accurately reflects the results of the election. Once the data has been analyzed, a report is generated that summarizes the results of the election. The audit file is archived for future reference.

4.8 Create an Audit File for Closed Party List

4.8.1 Description and Priority

The use case [UC_008 Create an Audit File for Closed Party List](#) describes how an audit file is produced with the election information at the time for Closed Party List Voting System. The audit file contains election information such as type of voting (CPL for Closed Party List), number of parties, parties following by candidates of each party in ranked order, number of seats, number of ballots, calculations, how many votes a party had, the winner(s), and show how the election progressed so that the audit could replicate the election itself. The audit file should show all of the steps. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.8.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the user prompts the system to produce an audit file for the Closed Party Voting System. The user prompts the system to produce an audit file. The system gathers all of the necessary data for the audit file. The collected data is collected and processed. The processed data is then stored securely in an audit file. The audit file is then analyzed to ensure that it accurately reflects the results of the election. Once the data has been analyzed, a report is generated that summarizes the results of the election. The audit file is archived for future reference.

4.9 Document Have At Least One Person Ranked

4.9.1 Description and Priority

The use case [UC 009 Document Have At Least One Person Ranked](#) ensures that the ballot has at least one of the candidates ranked as their top choice. For this iteration, assume that if there is a ballot in the file, it will have at least one of the candidates ranked. As you will see, the more rankings you complete as the voter, the more likely your vote will actually be counted towards electing the candidate(s) that you have as your top rankings. The long-term goal is for this system to be part of an integrated online voting system and all preprocessing will be done via the voting system itself. There will be no need to do any preprocessing of the file. Instead of voting for one candidate, voters rank their candidates in order of preference. A candidate can be given only one ranking. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.9.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the user runs the system for Instant Runoff, it will check if the document, The Instant Runoff Voting election, is being called. The system will check if the voters rank their candidates in order of preference where one candidate can be given only one ranking in the document. If the Instant Runoff Voting election is not being called, the system will not execute this use case. If the system checks that there is more than one ranking for a candidate in the document, the ballot will be considered invalid.

4.10 Run Two Types of Elections

4.10.1 Description and Priority

The use case [UC 010 Run Two Types of Elections](#) ensures that the system is capable of handling both types of elections which are Instant Runoff Voting (Plurality/Majority Preferential Voting) and Closed Party List Voting (Proportional representation). It is of high priority as if the system fails to run one of the two types of elections, it will cause an urgent problem that blocks the system use until the issue is resolved.

4.10.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the CSV file is processed for both Instant Runoff Voting and Closed Party List Voting.

Instant Runoff Voting is being called

1. The system reads in the CSV file and identifies that the Instant Runoff Voting election is being selected.
2. The system will run all the tasks based on the type of election being called. In this case, it is an Instant Runoff Voting election.

Closed Party List Voting is being called

1. The system reads in the CSV file and identifies that the Closed Party List Voting election is being selected.
2. The system will run all the tasks based on the type of election being called. In this case, it is a Closed Party List election.

Alternative Courses:

If the system fails to read in the CSV file and determine which type of election to be run, an error message will be sent to the user and displayed on the screen.

4.11 Ensures File Structure Does Not Change

4.11.1 Description and Priority

The use case [UC_011 Ensures File Structure Does Not Change](#) ensures that there will be no changes in the file structure outside of the program since the election files will come in the predetermined format. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.11.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the CSV file is processed for both Instant Runoff Voting and Closed Party List Voting. The file is being brought into the system. The system checks whether there are changes in the file structure outside of the program. If the system fails to bring the file into the system, an error message is displayed on the screen. If there are changes detected in the file structure, an error message is displayed on the screen.

4.12 Fair Coin Toss

4.12.1 Description and Priority

The use case [UC_012 Fair Coin Toss](#) describes how the system generates a common unbiased random bit to determine a winner when there is a two-way tie. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.12.2 Stimulus/Response Sequences

This use case will run only when there is a tie detected between two candidates/parties in the system. The behavior of this feature is triggered when there is a tie in the number of votes between two candidates or two parties or when there are additional seats that need to be redistributed to two parties. A tie is detected between two candidates/parties. The system will generate a fair coin toss

using a common unbiased random bit. The coin is flipped via the computer. The winner is determined based on the results displayed on the screen in an unbiased manner. If no tie occurred between two candidates/parties, the use case will not be executed. If there are no additional seats to be redistributed to two candidates/parties in the election, the use case will not be executed.

4.13 Pool Coin Toss

4.13.1 Description and Priority

The use case [UC_013 Pool Coin Toss](#) describes how the system generates a common unbiased random bit to determine a winner when there is a tie occurring among three or more candidates/parties. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.13.2 Stimulus/Response Sequences

This use case will run only when there is a tie detected among three or more candidates/parties in the system. The behavior of this feature is triggered when there is a tie in the number of votes among three or more candidates/parties or when there are additional seats that need to be redistributed to three or more parties. A tie is detected among three or more candidates/parties. The system will generate a pool coin toss using a common unbiased random bit. The coin is flipped via the computer. The winner is determined based on the results displayed on the screen in an unbiased manner. If no tie occurred among three or more candidates/parties, the use case will not be executed. If there are no additional seats to be redistributed to three or more candidates/parties in the election, the use case will not be executed.

4.14 Check for Majority

4.14.1 Description and Priority

The use case [UC_014 Check for Majority](#) checks if there is not a clear majority in Instant Runoff Voting, then the popularity wins after all votes have been handed out. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.14.2 Stimulus/Response Sequences

This use case will run when there is not a clear majority in Instant Runoff Voting election. The behavior of this feature is triggered when none of the candidates have more than 50% of the votes in an Instant Runoff Voting election. The system checks whether a candidate receives a clear majority in an Instant Runoff Voting. If no candidate receives a majority, then the candidate with the fewest votes is eliminated. The ballots of supporters of this defeated candidate are then transferred to whichever of the remaining candidates they marked as their number two choice. The system essentially operates as a series of runoff elections, with progressively fewer candidates each time, until one candidate gets a majority of the vote. After this, the votes are then recounted to see if any candidate now receives a majority of the vote. The process of eliminating the lowest candidate and transferring their votes continues until one candidate receives a majority of the continuing votes and wins the election. If a candidate receives over 50% of the first choice votes in an Instant Runoff Voting election, he or she is declared elected.

4.15 Check if Candidates Are Listed In A Particular Order

4.15.1 Description and Priority

The use case [UC_015 Check if Candidates Are Listed In A Particular Order](#) checks if the candidates are listed in a particular order. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.15.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the Closed Party List ballots are processed by the voting system. The system that runs the Closed Party List Voting election is being called. The system checks if the candidates are listed in a particular order in the Closed Party List Voting. The system calculates the winner(s) based on the number of seats and the order of the candidates. If the system fails to run the Closed Party Voting election, an error message will be sent to the user and displayed on the screen. If the system detects that the candidates are not listed in the particular order, the ballot will be considered invalid and an error message will be displayed on the screen.

4.16 Disallow Write-In Candidates

4.16.1 Description and Priority

The use case [UC_016 Disallow Write-In Candidates](#) ensures that write-in candidates are not allowed in the system for both types of elections. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.16.2 Stimulus/Response Sequences

This use case will run once per election by the election officials and thousands of times by the tester. The behavior of this feature is triggered when the list of candidates section of the CSV file is amended. The CSV file is being brought into the system. The systems that run both Instant Runoff Voting and the Closed Party List Voting election are being called. The system checks if there are any write-in candidates in the CSV file. If the system fails to bring in the file, an error message is displayed on the screen. If the system detects any write-in candidates in the system, the ballot will be considered invalid and an error message is displayed on the screen.

4.17 Display Election Results

4.17.1 Description and Priority

The use case [UC_017 Display Election Results](#) illustrates the actions performed by the system by showing the outcomes which include the winner(s) and information about a specified election on the screen. It is of medium priority as it is one of the core functionality that the voting system is explicitly supposed to perform and is a compromise.

4.17.2 Stimulus/Response Sequences

The use case will run once per election by the election officials and thousands of times by the tester. When the election officials choose the display election results option in the system menu, the system will display the results of the specified date and scope of the election on the screen. Only elections that are completed will be selectable. If the system fails to display the results of the specified election, an error message is displayed on the screen. If the user chooses the exit option, the task ends and the user is returned to the system menu.

5. Use Cases

5.1 UC_001 System Startup

Name	System Startup
ID	UC_001
Description	This use case describes the actions performed by the system during start-up and initialization phase.
Actors	Election officials, testers, and developers
Organizational Benefits	It brings up the interface.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When the user turns the switch on the panel to the “on” position.
Preconditions	None
Postconditions	The system is started in the normal mode.
Main Course	<ol style="list-style-type: none"> 1. The user turns on the start button and the system sends a message to all external physical units to check if they are ready to start. 2. The physical units send a message back to the system indicating that they are ready to start operating. 3. The welcome screen is displayed.
Alternate Courses	AC1: The system is deactivated when the operator turns the switch to the “off” position. AC2: If the system detects an error and a transmission failure occurs, the system will be put into the emergency stop mode and exits automatically.
Exceptions	The system fails to start-up and an error message is displayed on the screen. The system is powered off in an abnormal manner.

5.2 UC_002 File Identification

Name	File Identification
ID	UC_002
Description	The use case describes how the system recognizes the type of the file.
Actors	Election officials, testers, developers
Organizational Benefits	Each file can be distinguished quickly so election officials, testers, and developers can obtain the file easily.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	The actor runs the program to process the file from a directory
Preconditions	PC1: The file has to be in the same directory as the program. PC2: The file is successfully opened by the system.
Postconditions	The file is identified and the type of election is determined by the system.
Main Course	1. The system will read in the first line of the file that contains the type of election. 2. The type of the file is identified by the system and the file is ready to be named.
Alternate Courses	AC1: If the system fails to read in the first line of the file that contains the type of election, the file cannot be identified and an error message is displayed on the screen.
Exceptions	The file cannot be opened and identified.

5.3 UC_003 File Naming Convention

Name	File Naming Convention
ID	UC_003
Description	The use case describes how the name of the file is being named by the system.
Actors	Election officials, testers, developers
Organizational Benefits	Each file is categorized and can be found and accessed easily by the users.
Frequency of Use	This will run once per election by the election officials and thousands of times

	by the tester.
Triggers	The actor runs the program to process the file from a directory
Preconditions	The file is successfully opened and identified by the system.
Postconditions	The file is successfully named.
Main Course	<p>The name of the file contains the following information:-</p> <ul style="list-style-type: none"> - scope of the election (national, state, local) - type of the voting system (Instant Runoff or Closed Party List) - the date of the election being held (DD-MM-YEAR) <p><u>Name the File Through Command Line</u></p> <ol style="list-style-type: none"> 1. The type of election as well as the file type are identified. 2. The name of the file is passed into the program as a command line argument. <p><u>Ask Within Program Itself</u></p> <ol style="list-style-type: none"> 1. The file is successfully opened and identified by the system. 2. The system will ask for the name of the file within the program itself.
Alternate Courses	AC1: If the election is conducted multiple times on the same day, the system will overwrite the ballots in the same file which have the same name.
Exceptions	The system fails to open and identify the file.

5.4 UC_004 Open File

Name	Open File
ID	UC_004
Description	The use case describes how the system opens a file which contains ballots that were cast online.
Actors	Election officials, testers, and developers
Organizational Benefits	Each file can be distinguished quickly so election officials, testers, and developers can obtain the file easily.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	The actor runs the program to process the file from a directory
Preconditions	The file is successfully being brought into the system and is in the same directory as the program.

Postconditions	The file is successfully being opened and ready to be read and processed by the system.
Main Course	<ol style="list-style-type: none"> 1. The file is successfully being brought into the program. 2. The system successfully opens the file.
Alternate Courses	AC1: If the file fails to be brought into the system, an error message will be sent to the user. AC2: If the system fails to open the file, an error message will be displayed on the screen. AC3: If the exit option is chosen by the user, the task ends and the user is returned to the system menu.
Exceptions	EX1: The file fails to be brought into the system. EX2: The system fails to open the file.

5.5 UC_005 Read the File and Perform CSV Processing

Name	Read the file and perform CSV processing
ID	UC_005
Description	This use case describes the actions performed by the system by reading in a file and performing CSV processing.
Actors	Election officials, testers, and developers
Organizational Benefits	The system will process the comma separated values (CSV) file in a way that is human readable and makes it faster to handle.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When the CSV file is being opened by the system.
Preconditions	A valid comma separated values (CSV) file is imported. The CSV file can be successfully opened and read by the system.
Postconditions	The type of election is determined and run by the system. The results of the elections that are run completely are ready to be displayed.
Main Course	<u>Process CSV file for Instant Runoff Voting</u> <ol style="list-style-type: none"> 1. A valid comma separated values (CSV) file is imported and successfully opened by the system. 2. The UC_002 File Identification use case identifies that this is an Instant Runoff Voting election by reading in the first line of the CSV file that contains the type of the elections.

	<ol style="list-style-type: none"> 3. The system will perform CSV processing by reading in all the information from the file generated for Instant Runoff Voting. 4. The file generated for Instant Runoff Voting will be used for message formatting which follows the format as stated in 3.4.1.1 Message Formatting for Instant Runoff Voting. <p><u>Process CSV file for Closed Party List Voting</u></p> <ol style="list-style-type: none"> 1. A valid comma separated values (CSV) file is imported and successfully opened by the system. 2. The UC_002 File Identification use case identifies that this is a Closed Party List Voting election by reading in the first line of the CSV file that contains the type of the elections. 3. The system will perform CSV processing by reading in all the information from the file generated for Closed Party List Voting. 4. The file generated for Closed Party List Voting will be used for message formatting which follows the format as stated in 3.4.1.2 Message Formatting for Closed Party List Voting.
Alternate Courses	<p>AC1: If the system fails to read in the file, an error message will be sent to the user and displayed on the screen.</p> <p>AC2: If the system fails to perform CSV processing, an error message will be displayed on the screen.</p>
Exceptions	The system fails to read in the file and perform CSV processing.

5.6 UC_006 Prompt User For Information

Name	Prompt user for information
ID	UC_006
Description	When the program is run, the user can be prompted for any needed information that cannot be extracted from the file.
Actors	Election officials, testers, and developers
Organizational Benefits	Users can be prompted for any needed information in a short period of time without going through the files individually.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When prompted, a GUI pops up with a textbox to type in the information they require from the file.
Preconditions	The file has to be in the same directory as the program.
Postconditions	Once the required information is keyed into the textbox, the information is

	extracted from the file.
Main Course	Election officials use it to obtain required information from the file.
Alternate Courses	AC1: If the information type into the textbox is not recognized in the file, a message indicating that it is not found will be sent to the user.
Exceptions	The information typed into the textbox is not recognized in the file.

5.7 UC_007 Create An Audit File For Instant Runoff

Name	Create an audit file for Instant Runoff
ID	UC_007
Description	<p>The use case describes how an audit file is produced for Instant Runoff Voting System. The audit file contains election information at the time. The information includes</p> <ul style="list-style-type: none"> - Type of Voting (IR for Instant Runoff) - Number of Candidates - Candidates - Number of Ballots - Calculations - Number of votes per candidate - The winner(s) - Election progression
Actors	Election officials, testers, and developers
Organizational Benefits	Transparency: An audit file provides a clear record of all votes cast, enabling stakeholders to verify the accuracy of the election results. This increases transparency and public trust in the election process.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	User prompting the system to produce an audit file
Preconditions	User is searching for audit file
Postconditions	The audit file is saved and converted into a pdf and can be downloaded
Main Course	<ol style="list-style-type: none"> 1. User prompts the system to produce an audit file 2. The system gathers all of the necessary data for the audit file 3. The collected data is collected and processed. 4. The processed data is then stored securely in an audit file. 5. The audit file is then analyzed to ensure that it accurately reflects the results of the election. 6. Once the data has been analyzed, a report is generated that summarizes

	<p>the results of the election</p> <p>7. The audit file is archived for future reference</p>
Alternate Courses	<p>AC1: Update election results</p> <p>AC2: Amend candidate information</p> <p>AC3: Software updates and testing</p>
Exceptions	<p>EX1: User decides to discard report</p> <p>EX2: System fails to compile the report and notifies the user that an error has occurred.</p>

5.8 UC_008 Create An Audit File For Closed Party List

Name	Create an audit file for Closed Party List
ID	UC_008
Description	<p>The use case describes how an audit file is produced for Closed Party List Voting System. The audit file contains election information at the time. The information includes</p> <ul style="list-style-type: none"> - Type of Voting (CPL for Closed Party List) - Number of Parties - Parties - Candidates of Each Party in Ranked Order - Number of Seats - Number of Ballots - Calculations - Number of votes per party - The winner(s) - Election progression
Actors	Election officials, testers, and developers
Organizational Benefits	Transparency: An audit file provides a clear record of all votes cast, enabling stakeholders to verify the accuracy of the election results. This increases transparency and public trust in the election process.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	User prompting the system to produce an audit file
Preconditions	User is searching for audit file
Postconditions	The audit file is saved and converted into a pdf and can be downloaded
Main Course	<ol style="list-style-type: none"> 1. User prompts the system to produce an audit file 2. The system gathers all of the necessary data for the audit file

	<ol style="list-style-type: none"> 3. The collected data is collected and processed. 4. The processed data is then stored securely in an audit file. 5. The audit file is then analyzed to ensure that it accurately reflects the results of the election. 6. Once the data has been analyzed, a report is generated that summarizes the results of the election 7. The audit file is archived for future reference
Alternate Courses	AC1: Update election results AC2: Amend candidate information AC3: Software updates and testing
Exceptions	EX1: User decides to discard report EX2: System fails to compile the report and notifies the user that an error has occurred.

5.9 UC_009 Document Have At Least One Person Ranked

Name	Document have at least one person ranked
ID	UC_009
Description	This use case ensures that the ballot has at least one of the candidates ranked as their top choice.
Actors	Election officials, testers, and developers
Organizational Benefits	To reduce the risk of collecting faulty ballots.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	While the user runs the system for Instant Runoff, it will check if the document has at least one person ranked.
Preconditions	When Instant Runoff Voting election is selected
Postconditions	The document is ready to be calculated as a valid ballot.
Main Course	<ol style="list-style-type: none"> 1. The Instant Runoff Voting election is being called. 2. The system will check if the voters rank their candidates in order of preference where one candidate can be given only one ranking in the document.
Alternate Courses	AC1: If the Instant Runoff Voting election is not being called, the system will not execute this use case. AC2: If the system checks that there is more than one ranking for a candidate in the document, the ballot will be considered invalid.

Exceptions	EX1: When the Instant Runoff Voting election is not selected. EX2: When there is more than one ranking for a candidate in the document.
-------------------	--

5.10 UC_010 Run Two Types of Elections

Name	Run two types of elections
ID	UC_010
Description	The use case ensures that the system is capable of handling both types of elections which are Instant Runoff Voting (Plurality/Majority Preferential Voting) and Closed Party List Voting (Proportional representation).
Actors	Election officials, testers, and developers
Organizational Benefits	It provides a user-friendly interface as well as lowers the cost of purchasing, installation, management and maintenance of large amounts of hardware.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When the CSV file is processed for both Instant Runoff Voting and Closed Party List Voting.
Preconditions	The file reads in and identifies the type of election that is stored on the first line of the CSV file that contains the ballots.
Postconditions	The results for both types of elections are completed and ready to be displayed.
Main Course	<p><u>Instant Runoff Voting is being called</u></p> <ol style="list-style-type: none"> 1. The system reads in the CSV file and identifies that the Instant Runoff Voting election is being selected. 2. The system will run all the tasks based on the type of election being called. In this case, it is an Instant Runoff Voting election. <p><u>Closed Party List Voting is being called</u></p> <ol style="list-style-type: none"> 1. The system reads in the CSV file and identifies that the Closed Party List Voting election is being selected. 2. The system will run all the tasks based on the type of election being called. In this case, it is a Closed Party List election.
Alternate Courses	AC1: If the system fails to read in the CSV file and determine which type of election to be run, an error message will be sent to the user and displayed on the screen.
Exceptions	EX1: The system fails to read in the CSV file and determine which types of election to be run.

5.11 UC_011 Ensures File Structure Does Not Change

Name	Ensures file structure does not change
ID	UC_011
Description	This use case ensures that there will be no changes in the file structure outside of the program since the election files will come in the predetermined format.
Actors	Election officials, testers, and developers
Organizational Benefits	Mitigates the risk of encountering system errors when processing CSV files.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When a CSV file is processed for both Instant Runoff and Closed Party List.
Preconditions	A valid CSV file is successfully being brought into the system. The system successfully opens and reads the CSV file.
Postconditions	The document is valid and is safe to perform CSV processing.
Main Course	<ol style="list-style-type: none"> 1. The CSV file is being brought into the system. 2. The system checks whether there are changes in the file structure outside of the program.
Alternate Courses	AC1: If the system fails to bring the file into the system, an error message is displayed on the screen. AC2: If the system detects any changes in the file structure, an error message is displayed on the screen.
Exceptions	EX1: The system fails to bring the file into the system EX2: There are changes detected in the file structure

5.12 UC_012 Fair Coin Toss

Name	Fair Coin Toss
ID	UC_012
Description	Upon a two-way tie in voting the winner shall be determined by a fair coin flip.
Actors	Election officials, testers, and developers
Organizational Benefits	It is an unbiased way to determine a tie which instills trust and integrity for the voting system

Frequency of Use	It is used only when there is a tie between two candidates or parties.
Triggers	In an event where there are two candidates or two parties with the same number of votes. When two parties are eligible for additional seats.
Preconditions	The input must be valid. Eligibility: <ul style="list-style-type: none"> - There has to be a tie in the number of votes between two candidates or two parties. - When there are additional seats and need to be redistributed to two parties.
Postconditions	Winner is declared in an unbiased manner
Main Course	<ol style="list-style-type: none"> 1. A tie is being detected between two candidates/parties. 2. The system will generate a fair coin toss using a common unbiased random bit. 3. The coin is flipped via the computer. 4. The winner is determined based on the results displayed on the screen in an unbiased manner.
Alternate Courses	AC1: If no tie occurred between two candidates/parties, the use case will not be executed. AC2: If there are no additional seats to be redistributed to two candidates/parties in the election, the use case will not be executed.
Exceptions	EX1: There is no tie detected between two candidates/parties in the system. EX2: There are no additional seats to be redistributed to two candidates/parties in the election.

5.13 UC_013 Pool Coin Toss

Name	Pool Coin Toss
ID	UC_013
Description	The use case describes how the system generates a common unbiased random bit to determine the winner.
Actors	Election officials, testers, and developers
Organizational Benefits	It is an unbiased way to determine a tie which instills trust and integrity for the voting system
Frequency of Use	It is used only when there is a tie between three or more candidates or parties.
Triggers	In an event where there are three or more candidates/parties with the same

	number of votes. When three or more parties are eligible for additional seats.
Preconditions	The input must be valid. Eligibility: <ul style="list-style-type: none"> - There has to be a tie in the number of votes among three or more candidates/parties. - When there are additional seats and need to be redistributed to three or more parties.
Postconditions	Winner is declared in an unbiased manner
Main Course	<ol style="list-style-type: none"> 1. A tie is being detected among three or more candidates/parties. 2. The system will generate a pool coin toss using a common unbiased random bit. 3. The coin is flipped via the computer. 4. The winner is determined based on the results displayed on the screen in an unbiased manner.
Alternate Courses	AC1: If no tie occurred among three or more candidates/parties, the use case will not be executed. AC2: If there are no additional seats to be redistributed to three or more candidates/parties in the election, the use case will not be executed.
Exceptions	EX1: There is no tie detected among three or more candidates/parties in the system. EX2: There are no additional seats to be redistributed to three or more candidates/parties in the election.

5.14 UC_014 Check For Majority

Name	Check for Majority
ID	UC_014
Description	This use case checks if there is not a clear majority in an Instant Runoff Voting, then the popularity wins after all votes have been handed out.
Actors	Election officials, testers, and developers
Organizational Benefits	It is an unbiased way to determine the winner which instills trust and integrity for the voting system
Frequency of Use	This will run when there is not a clear majority in Instant Runoff Voting election.
Triggers	When none of the candidates have more than 50% of the votes

Preconditions	In an Instant Runoff Voting election, none of the candidates have more than 50% of the votes.
Postconditions	A winner is determined.
Main Course	<ol style="list-style-type: none"> 1. All the number one preferences of the voters are counted. 2. The system checks whether a candidate receives a clear majority in an Instant Runoff Voting. 3. If no candidate receives a majority, then the candidate with the fewest votes is eliminated. The ballots of supporters of this defeated candidate are then transferred to whichever of the remaining candidates they marked as their number two choice. 4. The system essentially operates as a series of runoff elections, with progressively fewer candidates each time, until one candidate gets a majority of the vote. 5. After this the votes are then recounted to see if any candidate now receives a majority of the vote. 6. The process of eliminating the lowest candidate and transferring their votes continues until one candidate receives a majority of the continuing votes and wins the election.
Alternate Courses	AC1: If a candidate receives over 50% of the first choice votes in an Instant Runoff Voting election, he or she is declared elected.
Exceptions	EX1: There is a candidate that receives over 50% of the first choice votes in an Instant Runoff Voting election.

5.15 UC_015 Check If Candidates Are Listed In A Particular Order

Name	Check if candidates are listed in a particular order
ID	UC_015
Description	This use case checks if the candidates are listed in a particular order in the Closed Party List Voting.
Actors	Election officials, testers, and developers
Organizational Benefits	To reduce the risk of collecting faulty ballots.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When Closed Party Listing ballots are processed by the voting system
Preconditions	When the system runs the Closed Party List Voting.

Postconditions	The winner(s) are being calculated based on the number of seats and the order of the candidates.
Main Course	<ol style="list-style-type: none"> 1. The system that runs the Closed Party List Voting election is being called. 2. The system checks if the candidates are listed in a particular order in the Closed Party List Voting. 3. The system calculates the winner(s) based on the number of seats and the order of the candidates.
Alternate Courses	AC1: If the system fails to run the Closed Party Voting election, an error message will be sent to the user and displayed on the screen. AC2: If the system detects that the candidates are not listed in the particular order, the ballot will be considered invalid and an error message will be displayed on the screen.
Exceptions	EX1: The system fails to run the Closed Party Voting election. EX2: The system detects that the candidates are not listed in the particular order.

5.16 UC_016 Disallow Write-In Candidates

Name	Disallow write-in candidates
ID	UC_016
Description	This use case ensures that write-in candidates are not allowed in the system for both types of election.
Actors	Election officials, testers, and developers
Organizational Benefits	Mitigates the risk of interrupting the voting process.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When the list of candidates section of the CSV file is amended.
Preconditions	When the system runs both Instant Runoff Voting and Closed Party List Voting. When a CSV file is processed for both Instant Runoff and Closed Party List.
Postconditions	The accurate results of the election are ready to be displayed.
Main Course	<ol style="list-style-type: none"> 1. The CSV file is being brought into the system. 2. The systems that run both Instant Runoff Voting and the Closed Party List Voting election are being called. 3. The system checks if there are any write-in candidates in the CSV file.

Alternate Courses	AC1: If the system fails to bring in the file, an error message is displayed on the screen. AC2: If the system detects any write-in candidates in the system, the ballot will be considered invalid and an error message is displayed on the screen.
Exceptions	EX1: The system fails to bring in the file. EX2: The system detects write-in candidates in the ballots.

5.17 UC_017 Display Election Results

Name	Display Election Results
ID	UC_017
Description	This use case illustrates the actions performed by the system by showing the outcomes which include the winner(s) and information about a specified election on the screen.
Actors	Election officials, testers, developers, and media personnel
Organizational Benefits	Election officials get to access the results of a specified election that is completed in a short period of time without going through the files individually.
Frequency of Use	This will run once per election by the election officials and thousands of times by the tester.
Triggers	When the election officials choose the display election report results option from the system menu.
Preconditions	Elections that are completed will be selectable.
Postconditions	The system reports the results for the specified election.
Main Course	<ol style="list-style-type: none"> 1. The election officials will select the date and type of the desired election that are completed to report the results for. 2. The system will display the specified election results on the screen.
Alternate Courses	AC1: If the system fails to display the report, an error message is displayed on the screen. AC2: If the exit option is chosen by the user, the task ends and the user is returned to the system menu.
Exceptions	The results of the specified election failed to be displayed on the screen.

6. Other Nonfunctional Requirements

6.1 Performance Requirements

The performance requirements section of the SRS document provides a clear understanding of the expected performance characteristics of the voting system as well as the measures taken to ensure that the system meets the performance requirements of its intended users and environment. This information will assist the developers in understanding the intention and making suitable design choices to meet the performance needs of the system. The performance requirements include:

6.1.1 Runtime constraints

Runtime constraints specify the maximum amount of time the voting system requires to complete tasks. The general runtime constraints include:

- (i) The voting system shall run 100,000 ballots in under 4 minutes.
- (ii) The voting system shall respond to user requests in under 5 seconds.
- (iii) The voting system shall process election files in under 10 seconds.

6.1.2 Timing relationships for real-time systems

Response Time for each functional requirement:

- (i) **(UC_001) System Startup:** The system shall start up in under 2 seconds
- (ii) **(UC_002) File Identification:** The system shall identify the file in under 5 seconds
- (iii) **(UC_003) File Naming Convention:** The system shall name the file in under 5 seconds
- (iv) **(UC_004) Open File:** The system shall open the file in under 5 seconds
- (v) **(UC_005) Read File and Perform CSV processing:** The system shall read the file and do CSV processing in under 5 seconds.
- (vi) **(UC_006) Prompt User for information:** The system shall prompt the user for information in under 5 seconds.
- (vii) **(UC_007) Create Audit file for Instant Runoff:** The system shall create an audit file for Instant Runoff in under 5 seconds.
- (viii) **(UC_008) Create an Audit file for Close Party List:** The system shall create an audit file for Close Party List Voting in under 5 seconds.
- (ix) **(UC_009) Document At Least One Person Ranked:** The system shall check the CSV file to see if at least one person is ranked in under 10 seconds.
- (x) **(UC_010) Run Two Types of Elections:** The system shall run two types of elections in under 5 seconds.
- (xi) **(UC_011) Ensure File Structure:** The system shall ensure file structure is preserved in under 5 seconds.
- (xii) **(UC_012) Fair Coin Toss:** The system shall flip a coin in under 2 seconds
- (xiii) **(UC_013) Pool Coin Toss:** The system shall flip a coin in under 2 seconds
- (xiv) **(UC_014) Check for Majority:** The system shall check for majority in IR in under 5 seconds
- (xv) **(UC_015) Check If Candidates Are Listed In A Particular Order:** The system shall check if candidates are listed in a particular order in under 5 seconds
- (xvi) **(UC_016) Disallow Write-In Candidates:** The system shall check for write-in candidates in under 5 seconds.
- (xvii) **(UC_017) Display Election Results:** The system shall display election results in under 5 seconds.

6.1.3 Scalability

Scalability measures the voting system's ability to increase performance in response to changes in processing demands.

- (i) The voting system shall be scalable to support 10,000 visits at the same time while maintaining optimal performance.
- (ii) The voting system shall be able to support 100 political candidates while maintaining optimal performance.
- (iii) The voting system shall be able to start up in under 2 seconds during high traffic volume.

6.2 Safety Requirements

The safety requirements section identifies requirements that safeguard the voting system; to protect voting data, to ensure the privacy of voters, and to maintain control such that inadvertent errors are minimized (Federal Election Commission, 2002). The safety requirements section is made up of two components — actions to be taken and actions that must be prevented.

6.2.1 Actions to be taken for safeguarding

- (i) Testers shall conduct thorough testing of the voting system 1 week prior to the election date.
- (ii) The Voting system shall be audited twice: 7 days before the election to test the system and another 7 days after the election when the results have been certified.
- (iii) Testers shall conduct a risk assessment every 6 months to analyze potential risks within the voting system through 3 steps: risk identification, risk analysis, and risk prioritization.

6.2.2 Actions to be prevented

- (i) Voting system shall not achieve a target error rate exceeding one in 10,000,000 ballot positions
- (ii) Users shall not disclose voting data to organizations other than the Department of Justice (DoJ), Help America Vote Act (HAVA) of 2002, Election Assistance Commission (EAC), and state governments.
- (iii) Voting system shall not exchange CSV and audit files between host servers if it does not comply with Transmission Control Protocol / Internet Protocol (TCP/IP).

6.3 Security Requirements

The security requirements describe the essential security capabilities for the voting system; to protect voting data, to ensure the privacy of voters, and to maintain control such that errors are minimized. The security standards include access control policies, data integrity, and compliance with regulations (Federal Election Commission, 2002). The objectives of the security requirement for the voting system include:

6.3.1 Access Control Policies

Access controls are procedures and system capabilities that detect and limit access to system components in order to guard against loss of system integrity, confidentiality, availability, and accountability (United States: Federal Election Commission, 2002). For individual access privileges, the voting system vendor shall:

- (i) Only a maximum of 40 people can be authorized to access the voting system — 25 representatives from political parties, 5 testers, and 10 software developers.
- (ii) For each person to whom access is granted, a government-issued ID and digital signature are required for user authentication.
- (iii) The maximum time interval of an individual's authorization to the voting system is 20 hours per week. The time interval increases to 40 hours per week from the week before until the week after the election.

6.3.2 Data Integrity and Protection

The voting system shall ensure that voting data is protected against unauthorized access. This includes:

- (i) Data can only be disclosed to these organizations: Department of Justice (DoJ), Help America Vote Act (HAVA) of 2002, Election Assistance Commission (EAC), and state governments.
- (ii) Comply with guidelines and procedures on voting data established by the Help America Vote Act (HAVA) of 2002. HAVA requires that voting systems be certified by an Independent Testing Authority (ITA) (Reagan, 2017).
- (iii) Voting system shall be audited twice: once a week before the election to test the system and another a week after the election when the results have been certified.
- (iv) Voting system shall not achieve a target error rate exceeding one in 10,000,000 ballot positions
- (v) Voting system shall not achieve a target error rate exceeding one in 500,000 ballot positions during the testing process.

6.3.3 Compliance with Regulations

Voting systems shall comply with voting requirements in the United States which are:

- (i) Comply with guidelines and procedures established by the Help America Vote Act (HAVA) of 2002 on security standards, auditing, and testing.
- (ii) Comply with integrity requirements of voting data established by the Election Assistance Commission (EAC) Voluntary Voting System.
- (iii) Comply with security requirements as per ISO 27001 Information Security Standards.

6.4 Software Quality Attributes

The software quality attributes section provides additional quality characteristics which are important to election officials, developers, and testers. This information will help the developers understand the needs and priorities of the system, and make suitable design choices to meet these needs (Indeed Editorial Team, 2021).

6.4.1 Usability

Usability is focused on ensuring that the voting system is user-friendly, efficient, and accessible to its users. The non-functional requirements are:

- (i) Election officials shall be able to use the voting system after one hour of training.
- (ii) Election officials shall be able to navigate through the Graphical User Interface (GUI) and generate an audit file within 10 minutes.
- (iii) Election officials shall be able to navigate through the Graphical User Interface (GUI) and display election results within 10 minutes.
- (iv) The average number of errors made by experienced users shall not exceed two per election.

6.4.2 Reliability

Reliability is focused on ensuring that the voting system is designed to minimize downtime and ensure that votes are counted accurately. The non-functional requirements are:

- (i) The Mean Time to Failure (MTTF) of the voting system shall be 10 years.
- (ii) The rate of failure occurrence of the voting system shall not exceed 1 per 10 elections.
- (iii) The number of critical failures during testing shall not exceed 1 per 10 tests.
- (iv) Testing shall be conducted on the voting system once every two weeks; therefore, the time between critical failures shall not exceed 20 weeks.

6.4.3 Robustness

Robustness is focused on ensuring that the voting system is designed to handle heavy user traffic and large amounts of data. The non-functional requirements are:

- (i) Time needed for the voting system to restart after failure should not exceed 10 seconds.
- (ii) The probability of data corruption on failure when processing CSV files on the voting system should not exceed 1%.
- (iii) The probability of data corruption on failure when creating audit files on the voting system should not exceed 1%.
- (iv) The probability of displaying incorrect election results on the voting system should not exceed 1%.

6.5 Business Rules

A business rule is a rule that defines a specific constraint within the context of a business. Business rules lay the foundation for automated systems by obtaining information and translating it into conditional statements (If, only if, when, then, else, it must always hold that, is correctly completed). There are two types of business rules — constraint rules and derivation rules (University of Houston Clear Lake, n.d.).

6.5.1 Constraint Rules

Constraint rules define conditions that place restrictions on object structures. A conditional path or an alternative path through the workflow is shown in constraint rules. The constraint rules for the voting system include:

- (i) It must always hold that the election file follows the naming conventions and the given format.
- (ii) It must always hold that the election file is located in the same directory as the program.
- (iii) It must always hold that the file structure of the election file cannot be changed.
- (iv) It must always hold that there will be no write-in candidates for this system.
- (v) It must always hold that only one election file will be given per election.
- (vi) Read the file and perform CSV processing only if the file is opened.
- (vii) Prompt the user for information only if the file is opened.
- (viii) Popularity wins after the votes have been handed out only if there is not a clear majority in Instant Runoff Voting.
- (viii) Display election results only if the user is prompted for information.
- (ix) If there is a tie between two political parties or two political candidates, then toss a fair coin. A winner is then determined based on the coin toss.
- (x) If there is a tie between three or more political parties or candidates, then do a pool coin toss. A winner is then determined based on the coin toss
- (xi) If there is no clear majority in Instant Runoff (IR), then popularity wins after all votes have been handed out.

6.5.2 Derivation Rules

Derivation rules define conditions under which facts can be inferred from other information. This may include a few more steps that need to be thought through to arrive at the conclusion. The derivation rules for the voting system include:

- (i) An election file can be opened if and only if the naming convention of the election file matches the predetermined format and if the file can be identified.
- (ii) Users can only be prompted for information if and only if the file is identified, read, and successfully processed.
- (iii) The election results for Close Party List Voting can be displayed if and only if the election file is successfully processed and there is no tie between the political parties.

(iv) The election results for Instant Runoff Voting can be displayed if and only if the file is successfully processed, at least one person is ranked, there is no tie between political candidates or parties, there are no write-in candidates, and the candidates are listed in a particular order.

(v) An audit file for Close Party List Voting can be generated if and only if the election file is successfully processed and there is no tie between the political parties.

(vi) An audit file for Instant Runoff Voting can be generated if and only if the file is successfully processed, at least one person is ranked, there is no tie between political candidates or parties, there are no write-in candidates, and the candidates are listed in a particular order.

Appendix A: Glossary

Ballot: a means of registering a vote, or a device to cast votes in an election.

CPL: Closed Party List ballot. It describes the variant of party-list systems where the voters can effectively only vote for political parties as a whole.

CSE Labs: Lab machines at the College of Science and Engineering, University of Minnesota. Details can be found here <https://cse.umn.edu/cseit/classrooms-labs>

CSV: Comma Separated files, a text file that has a specific format that allows data to be saved in a table structured format.

Developers: A team of developers with a software engineering background, developing the software.

Dialog box: Graphical components that are usually used to display errors give information to the user.

Election officials: Authorizers who run an election, did not belong to any parties.

Git/Github: Open source distributed version control system designed to handle everything, including any set of computer files.

GUI: A graphical user interface that incorporates icons, buttons, pull-down menus, and a mouse.

IDE: An integrated development environment that provides comprehensive facilities to programmers for software development.

IR: Instant Runoff. A type of ranked preferential voting method. Used the majority voting rule in a single-winner election.

Java: A high-level, class-based, and object-oriented programming language.

Linux: An operating system, version of Unix

macOS: An operating system, version of Unix

Residents: The voter who is on the enumeration list and who has been given the means to vote.

Testers: A team of developers with a software engineering background; who create test cases to test every scenario.

U of MN: The University of Minnesota.

Windows: An operating system

Appendix B: Analysis Models

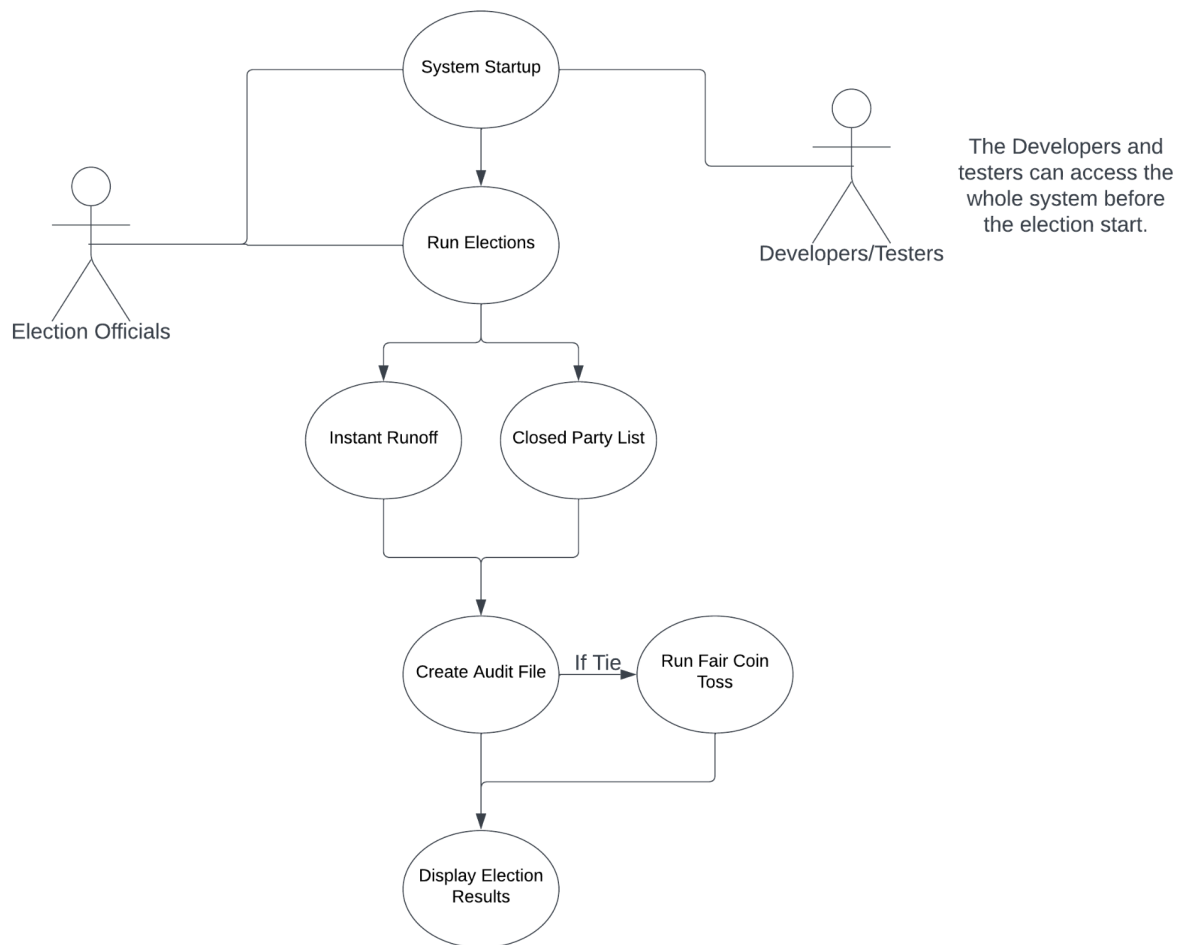


Figure: Higher level view of relations between use cases and actors.