Software Requirements Specification

for

Party List Voting System

Version 1.0 approved

Prepared by Michael Ung

Zhuoran Bi

Yongfeng Ji

Jing Wu

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Michael Ung Yongfeng Ji Zhuoran Bi Jing Wu	10/10/19	Document creation writing	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the voting system. It will demonstrate the ultimate goal, determining which candidates are the winners in a given election and generating an audit file that corresponds to the aforementioned data. It will also cover features of this system, what functionalities are, what the system will do, how to interface with this system and what the non-functional requirements are. This document is intended for the programmers, testers and election officials who may interact with the system.

1.2 Document Conventions

This document is created based on the IEEE template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

The primary readers would be programmers, testers, and election officials, since they are all actors that will directly interact with the system, for instance, running the program via the command line. The secondary readers would be media reporters or journalists, since they may deliver the election results, through social media or websites, to the public under the agreement of election officials.

Reading suggestions: For users, the next chapter is the Overall Description section, this will give an overview of the functionality of the system. It describes the function of each use case written with understandable sentences for non-technical developers. Chapters 3 and 5 mainly describe the technical requirements which are written for developers. All sections are written for the same software product in its entirety, but are designed for different audiences, thus use different languages.

1.4 Product Scope

The ultimate goal of the software is to determine which candidates are the winners in a given election and generating an audit file that corresponds to the resulting data. Finally, the system will be able to display the result in a decent way.

1.5 Reference

CSCI 5801_Project1_Waterfall_VotingSRS_Fall2019_final.pdf

CSCI5801_Fall2019_SRS_Rubric.pdf

2. Overall Description

2.1 Product Perspective

Two different Voting Systems: OPL(Open Party List) and CPL(Closed Party List).

OPL is developed for voting for the candidate, CPL is developed for voting parties. It is like the calculation program, given the format of voting information, then calculate the final result of voting based on the type of voting system, and it provides two functions: create an audit file that comprises all original information plus the result of voting, and create voting result text file that only include the voting result based on the type of voting system, for example, after executing command line of creating voting result text file in CPL voting system, the winning party would be written in the voting result text file.

2.2 Product Functions

- Specify file name: Type the file path. System will check if the file exists or not. If not, it would return an error. If it exists, systems will be prepared to open it.
- Run Voting System for OPL: System would read the input OPL file, and process all the data in that file and save some data into local database.
- Run Voting System for CPL: System would read the input CPL file, and process all the data in that file and save some data into local database.
- Create Audit file for OPL: After running the OPL system, all the OPL information will be generated in a text file that will be downloaded into a local disk.
- Create Audit file for CPL: After running the CPL system, all the CPL information will be generated in a text file that will be downloaded into a local disk.
- Display voting results: display the voting results of the election onto the user's screen, which include winner, and how many votes they obtained and how many votes the party obtained, and the total amount of votes cast.
- Create voting result text file for OPL: It will create a text file that comprise the result of voting for OPL.
- Create voting result text file for CPL: It will create a text file that comprise the result of voting for CPL

2.3 User Classes and Characteristics

- Typical user is Election Officer, they need to know the result of voting, thus they can decide who is the winner.
- Other users like programmers, testers. They need to test if the voting system works well or not.

2.4 Operating Environment

Program will only be running in the linux operating system in the CSE lab.

2.5 Design and Implementation Constraints

Both voting systems are developed in Java, the first challenge we may encounter is the interface design, a simple command line or an adorned interface. If we choose the interface instead of a simple command line, what java platform should we use. And the second challenge is how to make the system correctly read the input file, since the input file is well formatted. Third challenge is the runtime constraints, the system should be able to run 100,000 ballots in under 5 mins. The last challenge is if the voting results in a tie, how to break the tie.

2.6 User Documentation

how to use the Linux shell with Java in command line：
https://introcs.cs.princeton.edu/java/15inout/linux-cmd.html

Party list description for both OPL and CPL:
https://www.electoral-reform.org.uk/voting-systems/types-of-voting-system/party-list-pr/

System user manual is under design. It will come up soon.

2.7 Assumptions and Dependencies

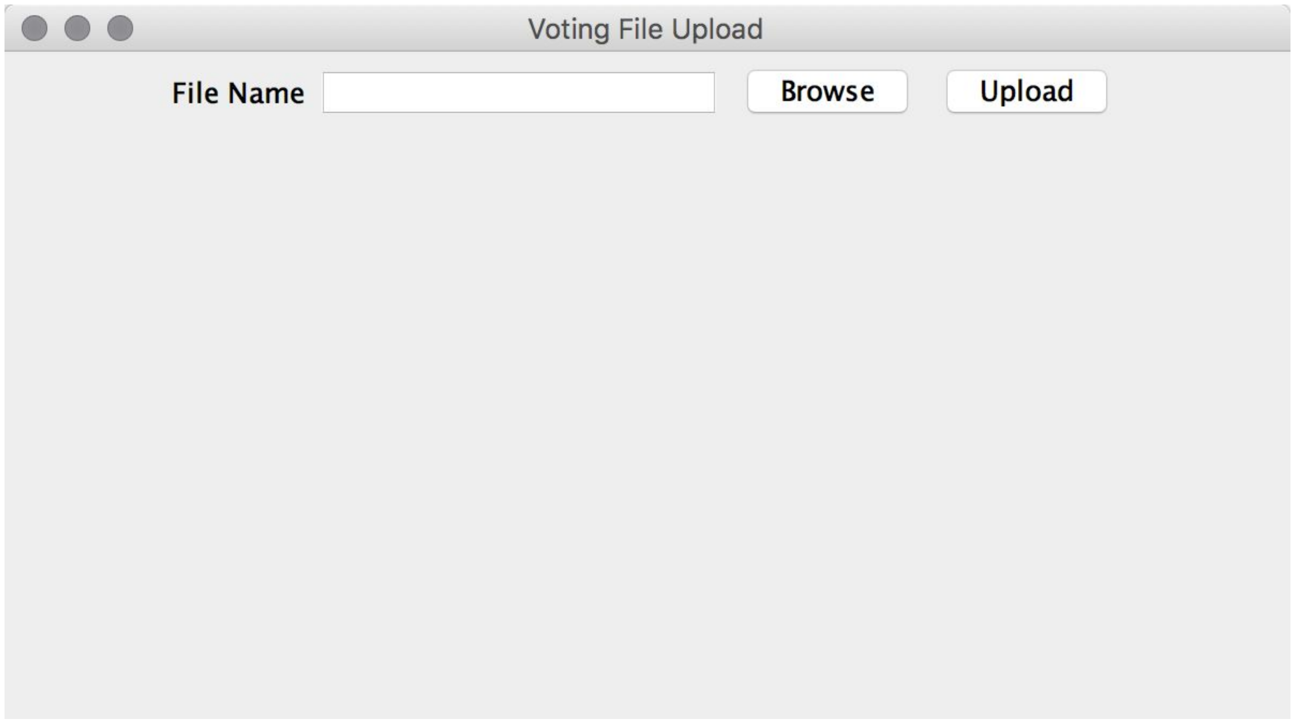
It is assumed that the program will be running under linux system in CSE Lab, no numbering mistake in the file, file will be in CSV format. The java platform should be Processing 3.5 version or newer.

3. External Interface Requirements

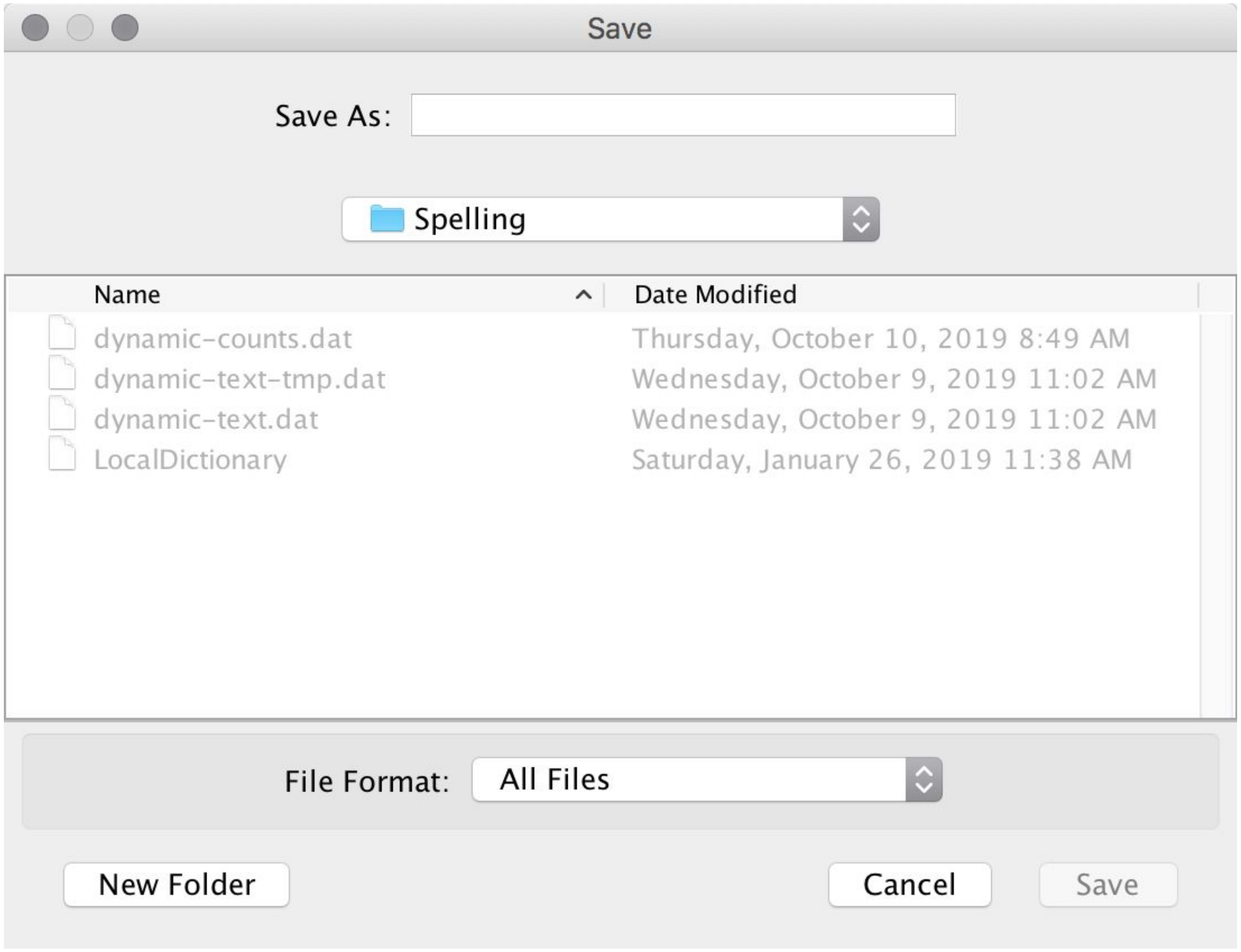
According to the current information, we designed some initial interface, during the implementation and conversation with users, the interface may be revised.

3.1 User Interfaces

1. Prompt users to specify filename Screen:



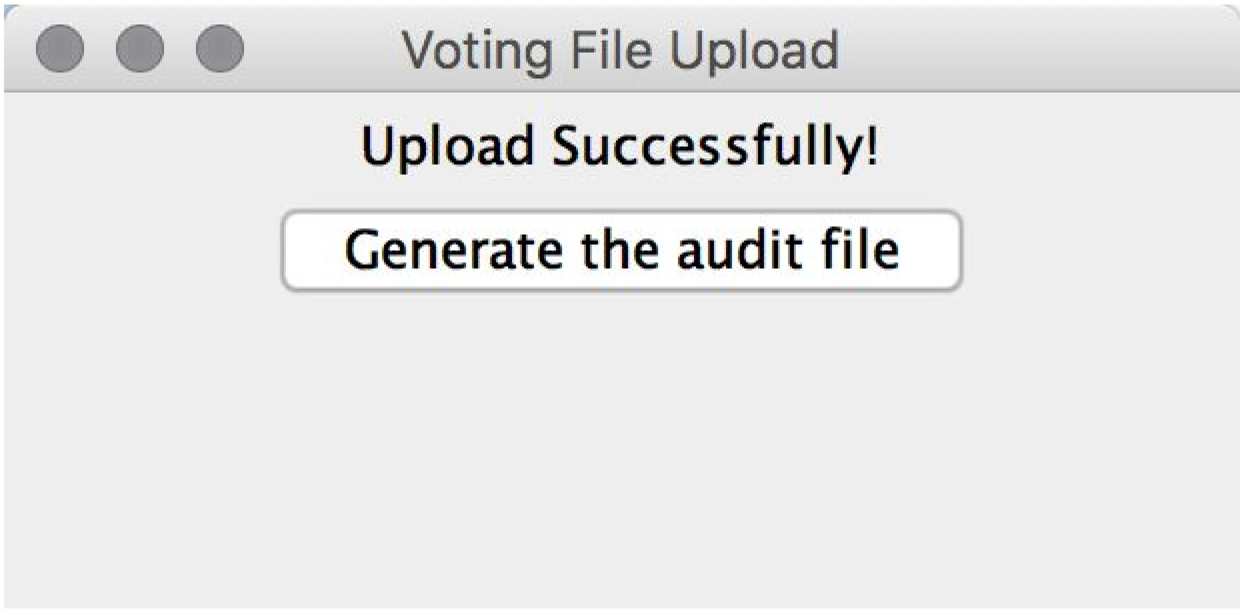
2. Upload CSV format file exported from Excel:



Name	Date Modified
dynamic-counts.dat	Thursday, October 10, 2019 8:49 AM
dynamic-text-tmp.dat	Wednesday, October 9, 2019 11:02 AM
dynamic-text.dat	Wednesday, October 9, 2019 11:02 AM
LocalDictionary	Saturday, January 26, 2019 11:38 AM

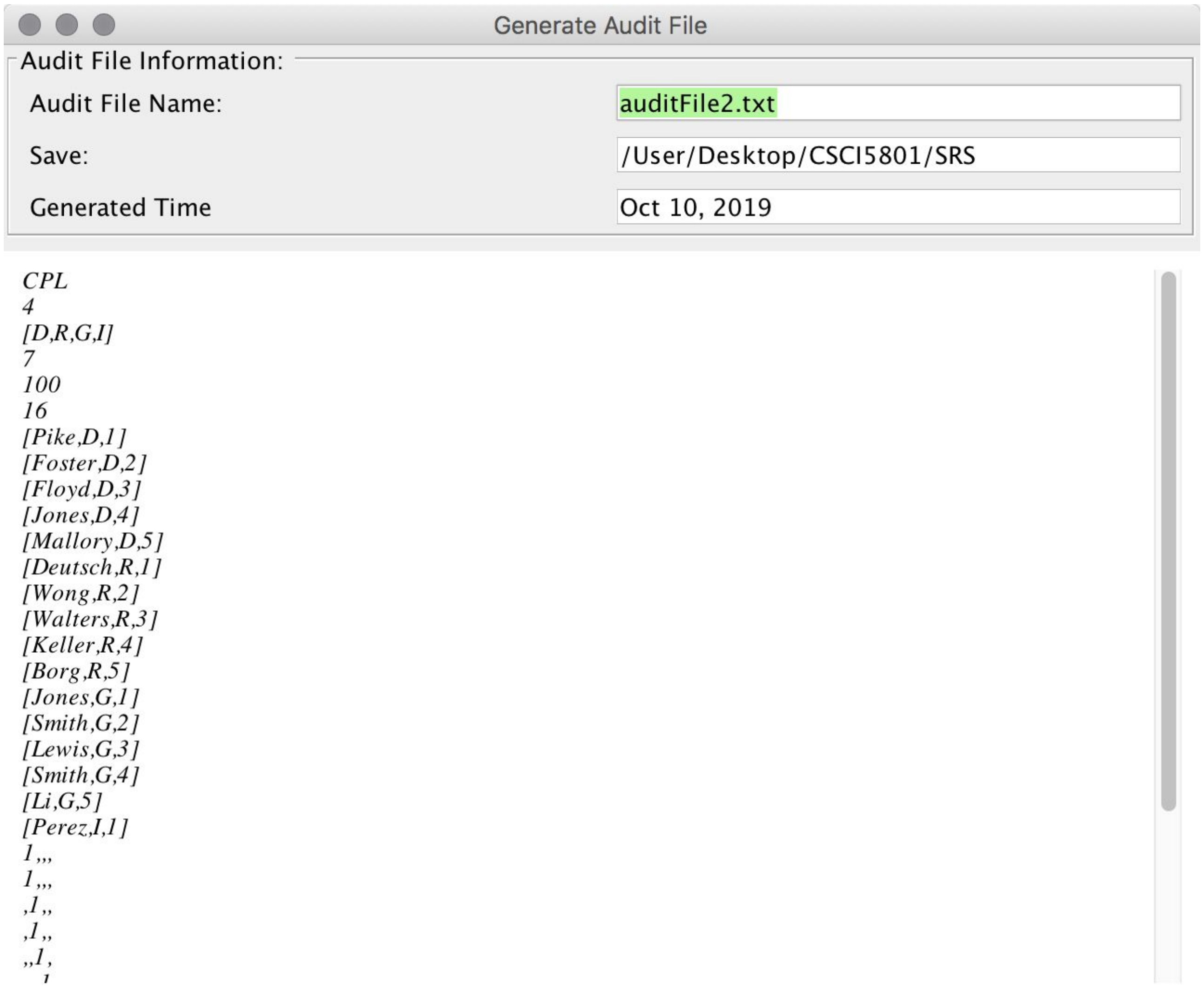
File Format: All Files

3. Prompt users "Upload Successfully":



Upload Successfully!

4. Generate the audit file:
   (1) Generate CPL audit file



Audit File Information:

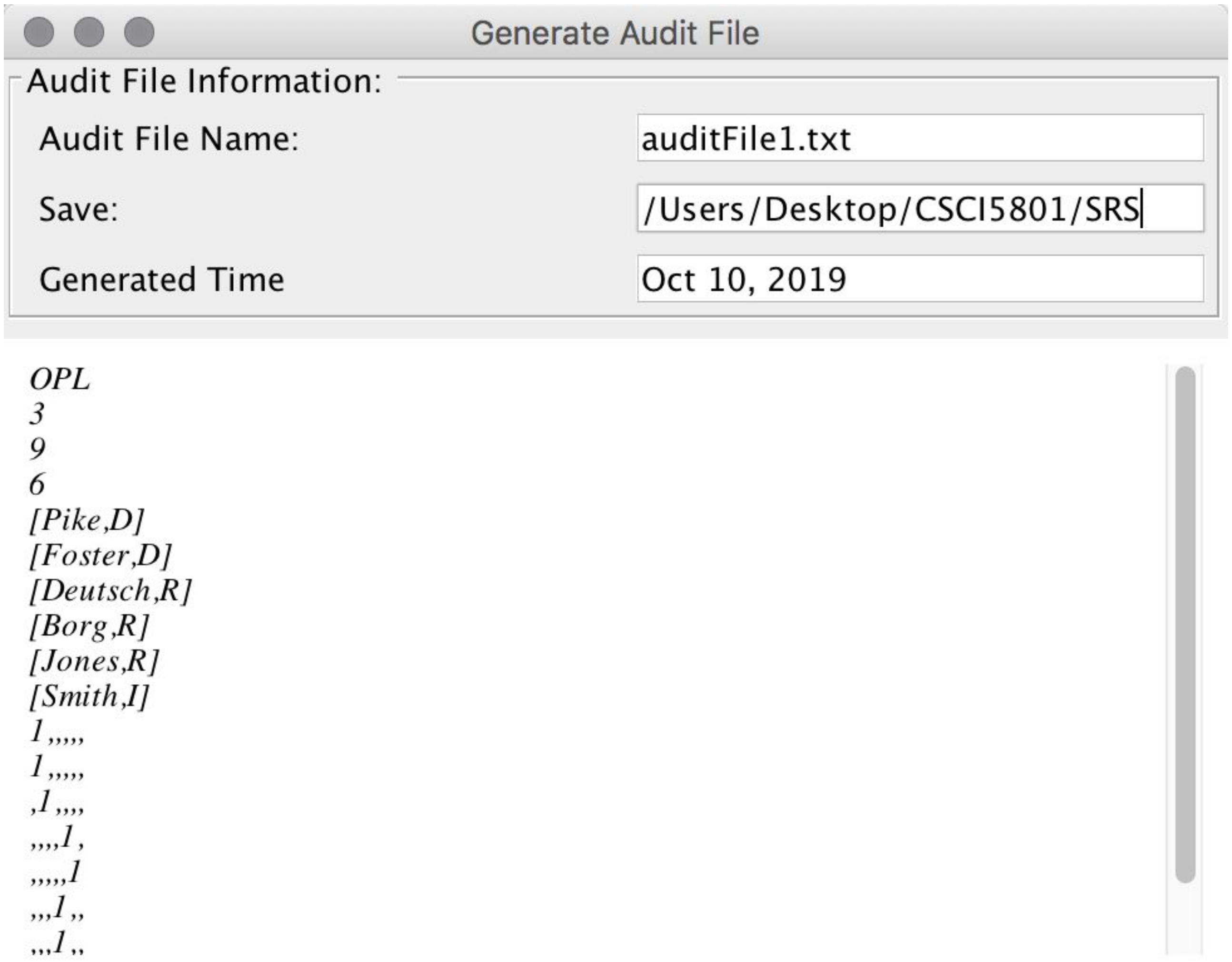
Audit File Name: auditFile2.txt

Save: /User/Desktop/CSCI5801/SRS

Generated Time Oct 10, 2019

```
CPL
4
[D,R,G,I]
7
100
16
[Pike,D,1]
[Foster,D,2]
[Floyd,D,3]
[Jones,D,4]
[Mallory,D,5]
[Deutsch,R,1]
[Wong,R,2]
[Walters,R,3]
[Keller,R,4]
[Borg,R,5]
[Jones,G,1]
[Smith,G,2]
[Lewis,G,3]
[Smith,G,4]
[Li,G,5]
[Perez,I,1]
1,,,
1,,,
,1,,
,1,,
,,1,
```

   (2) Generate OPL audit file

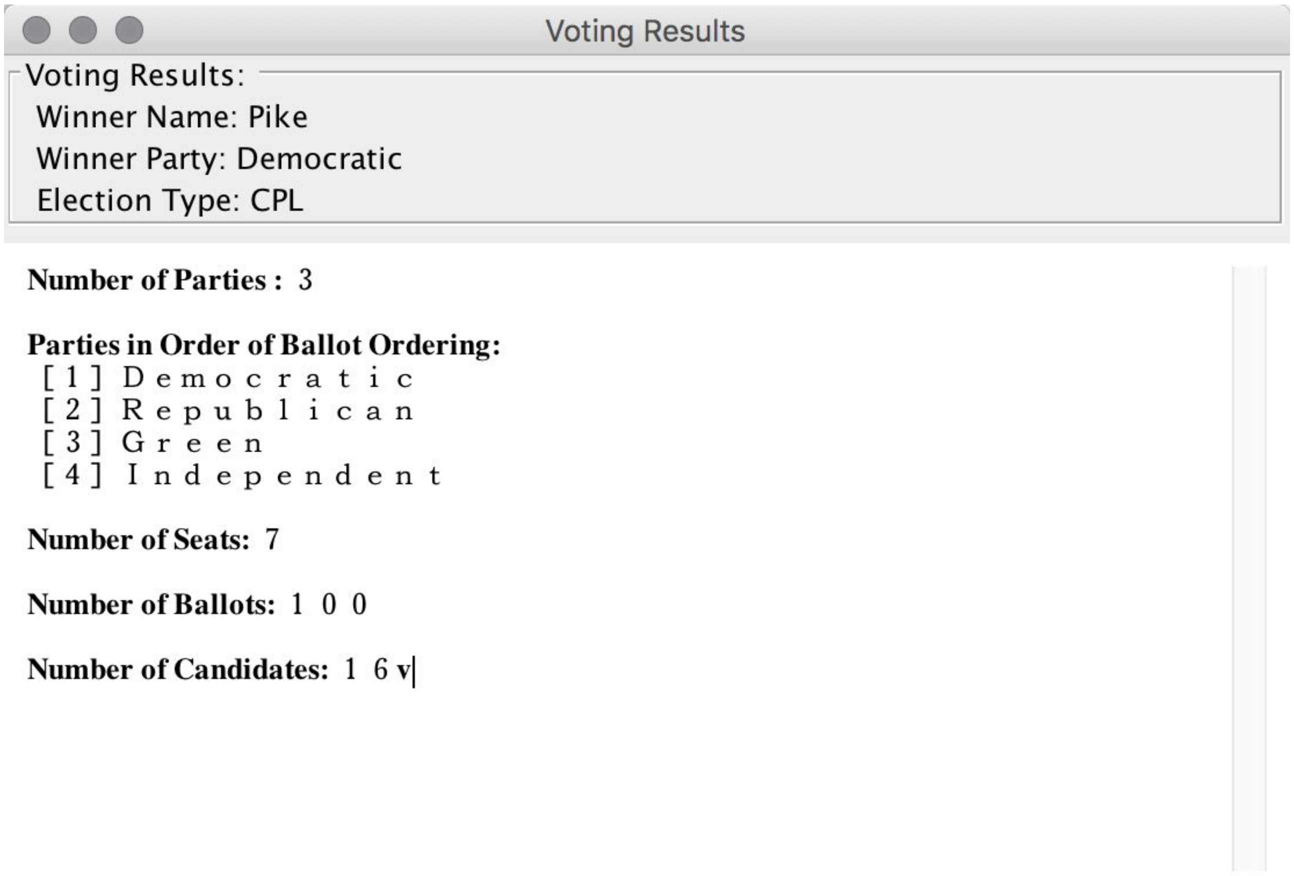


Audit File Information:

Audit File Name:	auditFile1.txt
Save:	/Users/Desktop/CSCI5801/SRS
Generated Time	Oct 10, 2019

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

5. Display results to screen:
   (1) Display results for CPL



Voting Results

Voting Results:
Winner Name: Pike
Winner Party: Democratic
Election Type: CPL

Number of Parties : 3

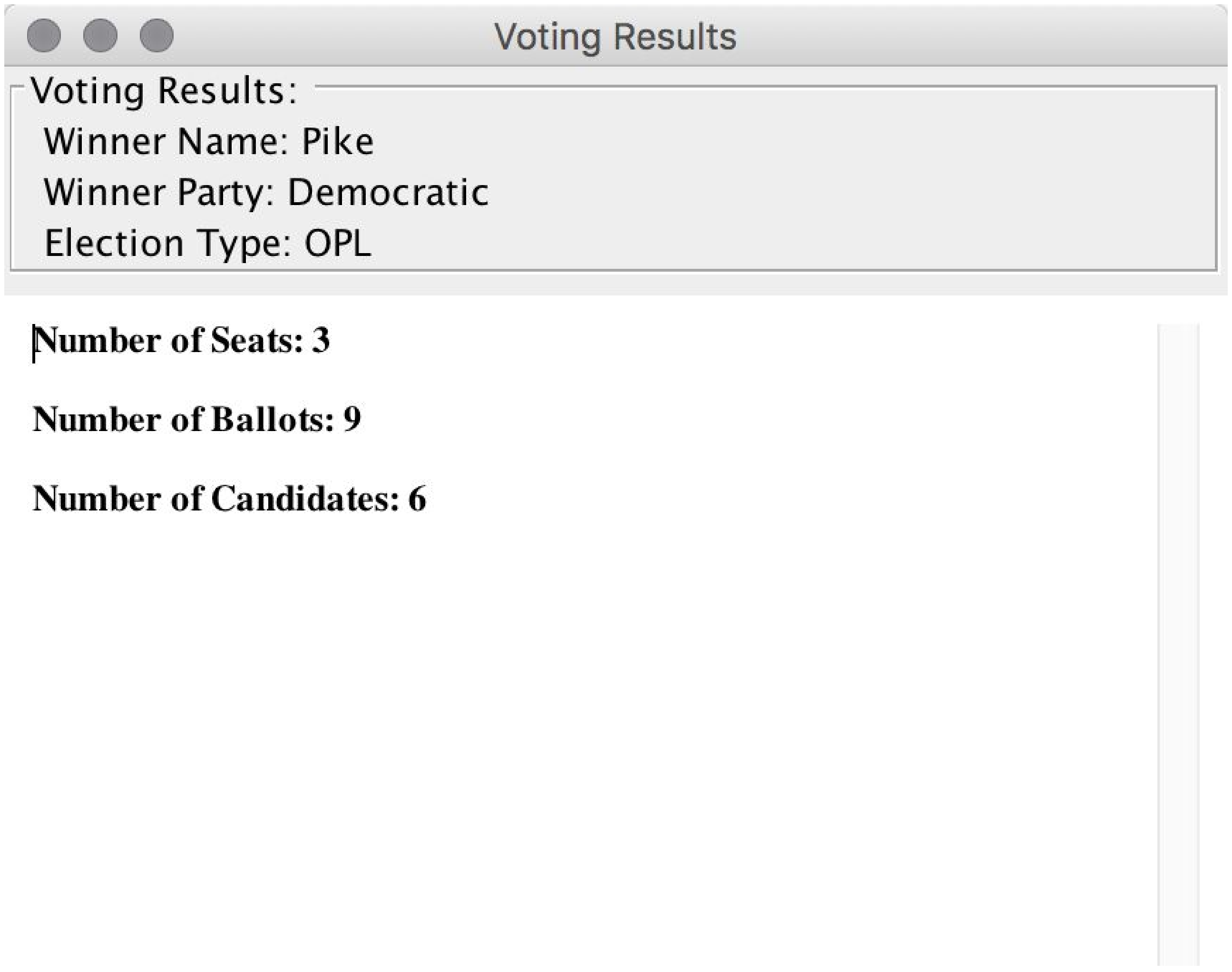
Parties in Order of Ballot Ordering:
[1] Democratic
[2] Republican
[3] Green
[4] Independent

Number of Seats: 7

Number of Ballots: 1 0 0

Number of Candidates: 1 6 v

(2) Display results for OPL



Voting Results:
Winner Name: Pike
Winner Party: Democratic
Election Type: OPL

Number of Seats: 3

Number of Ballots: 9

Number of Candidates: 6

3.2 Hardware Interfaces

The minimum hardware requirements of our program are Unix based operating systems with at least 500MB RAM. For bigger networks, additional memory is required.

3.3 Software Interfaces

Our system requires Java to be installed on the system, any versions may be fine.

It can offer the GUI, which allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation.

3.4 Communications Interfaces

Our system does not require network connection. However, we do offer network connections for users to operate online.

4. System Features

This section breaks down the core areas of the system and the interactions between the user and the system and a few aspects of what the system will do in response to user interaction. Please see the "List of Possible Use Cases" for a more detailed breakdown.

4.1 Analyzing Voting Files for OPL

4.1.1 Description and Priority

This feature handles the general course of action for the OPL voting method in response to a user file input. The feature encompasses the reading and processing of the input data in order to determine the seat winners for the election. The feature is given a high priority due to the system requiring an analysis of votes for an election using the OPL voting method, in addition to the need for analysis to be done prior to implementing features that require use of the results generated by this feature. A benefit of completing this feature will allow for easier implementation of this feature's counterpart, "Analyzing Voting Files for CPL", which can be seen in section 4.2. There are no negative repercussions if this feature is implemented first, other than having to work with what is likely the most complex part of the system.

4.1.2 Stimulus/Response Sequences

1. The user should have specified the name of a valid input file to analyze.
2. The system will read the first line of the file and determine that the voting method used is OPL.
3. The system then reads through the remaining data and processes the data in accordance with how the voting method functions.
4. In the case of a tie, the system will assign a value to each of the members involved and execute a fair coin toss to determine the winner.
5. The system then completes the process and proceeds to display the results.

4.1.3 Functional Requirements

REQ-1:	System will be able to access a valid file specified by the user.
REQ-2:	System will be able to determine which voting method the file specifies.
REQ-3:	System will be able to allocate votes to a specific candidate.
REQ-4:	System will be able to resolve any ties.
REQ-5:	System will be able to determine which candidates receive which seats.

REQ-6: System will be able to handle the exception where an error occurs during processing voting information.

4.2 Analyzing Voting Files for CPL

4.2.1 Description and Priority

This feature handles the general course of action for the CPL voting method in response to a user file input. This feature covers similar actions to the "Analyzing Voting Files for OPL", that include reading and processing data. Alternatively, this feature has a different way for both allocating votes and seats. This feature should be given a high priority, very similar to the related OPL voting feature, except slightly less. The reason being that this feature will likely build off of the OPL voting feature, except it requires more actions in order to take into account that parties are being voted for instead of candidates. It is beneficial to work on this in the early stages of the project due to features that are dependent on the results of this feature that will need to be implemented at a later stage. Similar to this feature's counterpart, this is likely the most complex part of the system and will take longer to complete.

4.2.2 Stimulus/Response Sequences

1. The user should have specified the name of a valid input file to analyze.
2. The system will read the first line of the file and determine that the voting method used is CPL.
3. The system then reads through the remaining data and processes the data in accordance with how the voting method functions.
4. In the case of a tie, the system will assign a value to each of the parties involved and execute a fair coin toss to determine the winner. In this case the winner receives all of the remaining seats.
5. The system then completes the process and proceeds to display the results.

4.2.3 Functional Requirements

REQ-1: System will be able to access a valid file specified by the user.
REQ-2: System will be able to determine which voting method the file specifies.
REQ-3: System will be able to allocate votes to a specific party.
REQ-4: System will be able to resolve any ties.
REQ-5: System will be able to determine which candidates receive which seats based on the number of seats and the threshold to obtain a seat.
REQ-6: System will be able to handle the exception where an error occurs during processing voting information.

4.3 Displaying Voting Results

4.3.1 Description and Priority

The feature handles the displaying of the results that are created by either the OPL or CPL voting analysis. This feature takes the results of the previous two features and displays them in a coherent and easily read manner. This feature has a medium priority due to it not being a dependency for any other feature. It is not a low priority feature due to the need to display the results of the previously mentioned features being present, but there are no major benefits of completing this feature aside from being able to verify that the previous algorithms worked properly.

4.3.2 Stimulus/Response Sequences

1. The system has completed running its analysis and has the results accessible by the display feature.
2. The system will display the winners in list in addition to other statistics such as: total votes, votes received by each candidate/votes received by each party, number of seats
available, and number of candidates/parties running.
3. The system will be able to display additional statistics about each candidate or party by clicking on the name.

4.3.3 Functional Requirements

REQ-1:	System will be able to access the information created by the analysis.
REQ-2:	System will be able to determine which style to display the results in, based on the voting method used.
REQ-3:	System will be able to display the results of the analysis.
REQ-4:	System will be able to display options that are possible after displaying the results.
REQ-5:	System will be able to handle the exception where an error occurs during displaying the voting results.

4.4 Exporting Results to a File

4.4.1 Description and Priority

This feature handles the conversion of the results of the analysis to different file types. This feature encompasses two different options: exporting basic statistics to a .txt file and exporting all data to an audit file. This feature takes on a low priority due to not having other features depending on it. These features also cover a very small portion of the system and are not as important as the larger features. The only benefit that the completion of this feature presents is that a requirement is completed. Attempting to complete this feature early may result in the programmer needing more time to implement this feature, due to the prerequisite features being incomplete.

4.4.2 Stimulus/Response Sequences

1. The system has completed running its analysis and is displaying the results.
2. The user clicks a the button indicating that the user wants to create a .txt file of the results (see #2 of the next sequence).
3. The system compiles the displayed data into a .txt file[1] and downloads the file onto the user's filesystem.

2. Alternatively, the user may click on the button indicating that the user wants to create
an audit file of the voting process.
3. The system parses through the data and compiles the results into the audit file[2] and downloads it onto the user's filesystem.

4.4.3 Functional Requirements

REQ-1:	System will be able to create a file and write the resulting data to it.
REQ-2:	System will be able to know which voting file was input.
REQ-3:	System will be able to handle errors when creating the file to write to.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Since our voting system requires users to type the file name through command lines, we don't require any device performance requirements for each individual user. However, we do require users to use operating systems which support command line typing, which means our voting system should at least run on PC, rather than a mobile device. In addition, the system needs to display the voting result, which requires the users using devices having a good visualization display.

5.2 Safety Requirements

The voting system does not have any special safety requirement, because this system only receives external input file as data source to process. Therefore, the system itself does not store too much data in local memory, then there are no possibilities of losing data while the system is running. The only concern may be to make sure the local storage has enough room to store the output audit files.

5.3 Security Requirements

The system itself does not cause security issues since it just processes the given data and generates the corresponding output files. However, we do encourage all the users to keep voting data safe and make sure nobody could falsify the original voting input files in order to ensure fairness of the whole election. As for the voting result, only election officials have the right to determine if the result can be exposed to the public, testers and programmers must keep the results private until election officials agree with the transparency of the voting results.

"Security such as ensuring one vote for one person is handled at the voting centers."

5.4 Software Quality Attributes

Correctness: Correctness is the basic and essential requirement of our voting system since it is the most important to pick the winner based on the number of ballots he or she got. Or this system can not be treated as successful. Thus, we do ensure correctness.

Robustness: We do require all users to provide a valid input file name including the correct syntax and format of data. In that way, the system can ensure robustness for any input files.

Testability & Reusability: For our voting system, every new test means a new use. Therefore, once our system completes, and the input file format does not change, our system can be used forever, no matter if it is a single test or a real election event.

5.5 Business Rules

All users have to provide a valid input file name, and the file will be exported from Excel into CSV format. The content of this input file must follow the specific format mentioned by election officials.

6. Other Requirements

1. An election should be able to run 10,000 ballots in under 5 minutes.
2. There will never be more than one file given to you per election.
3. The election file will be located in the same directory as the program.

Appendix A: Glossary

<Party List Voting>
systems are by far the most common form of proportional representation. Over 80% of the PR systems used worldwide are some form of party list voting. It remains the system used in most European democracies and in many newly democratized countries, including South Africa.

<Closed Party List (CPL)>
In a closed list system--the original form of party list voting--the party fixes the order in which the candidates are listed and elected, and the voter simply casts a vote for the party as a whole. This is shown in the first ballot below, which illustrates an election for the House of Representatives in a five-seat district. Voters are not able to indicate their preference for any candidates on the list, but must accept the list in the order presented by the party. Winning candidates are selected in the exact order they appear on the original list.

<Open Party List (OPL)>
This approach allows voters to express a preference for particular candidates, not just parties. It is designed to give voters some say over the order of the list and thus which candidates get elected. One version of this is illustrated in the ballot below. Voters are presented with unordered or random lists of candidates chosen in party primaries. Voters cannot vote for a party directly, but must cast a vote for an individual candidate. This vote counts for the specific candidate as well as for the party. So the order of the final list completely depends on the number of votes won by each candidate on the list. The most popular candidates rise to the top of the list and have a better chance of being elected. In our example, if the Democrats won 2 seats, and Volz and Gentzler received the highest and next highest number of individual votes, they would rise to the top of the list and be elected.

Appendix B: Analysis Models

We may use diagrams to display the relation between each use case.

Appendix C: To Be Determined List

1. The exact contents of the .txt file generated for the results of the analyzing process.
2. The exact contents of the audit file generated.
3. Additional statistics to show when clicking on a candidate/party.