

(Csci 5801, Group 22)

**(Voting System)**  
Software Design Document

Name (s): Akar Kaung, Hao Wu, Moyan Zhou, Yiming Yao

Lab Section: 001

Workstation: Team 22

Date: (02/27/2021)

## TABLE OF CONTENTS

<b>1.</b>	<b>INTRODUCTION</b>	<b>3</b>
1.1	Purpose	3
1.2	Scope	3
1.3	Overview	3
1.4	Reference Material	4
1.5	Definitions and Acronyms	4
<b>2.</b>	<b>SYSTEM OVERVIEW</b>	<b>4</b>
<b>3.</b>	<b>SYSTEM ARCHITECTURE</b>	<b>5</b>
3.1	Architectural Design	5
3.2	Decomposition Description	6
3.3	Sequence Diagram	6
<b>4.</b>	<b>DATA DESIGN</b>	<b>7</b>
4.1	Data Description	7
4.2	Data Dictionary	8
<b>5.</b>	<b>COMPONENT DESIGN</b>	<b>9</b>
<b>6.</b>	<b>HUMAN INTERFACE DESIGN</b>	<b>11</b>
6.1	Overview of User Interface	11
6.2	Screen Images	11
6.3	Screen Objects and Actions	12
<b>7.</b>	<b>REQUIREMENTS MATRIX</b>	<b>13</b>

# **1. INTRODUCTION**

## **1.1 Purpose**

This software design document describes the architecture and system design of the voting system. (VTS) This voting system software serves as the reference for faculties and students at the department of computer science and engineering at University of Minnesota -Twin Cities.

## **1.2 Scope**

The document contains a completed description of the design of the voting system (VTS) at Group 22, Csci 5801. The basic architecture of design of VTS is object-oriented programming (OOP). The main programming language of the voting system (VTS) is JAVA (latest version). The feature of the VTS is that users can import voting ballots into the system, and the system will generate the election result based on different voting rules/ methods. This document includes the additional information about voting rules required in the VTS.

## **1.3 Overview**

The remaining chapters and their contents are listed below.

Section 2 is to introduce overview of the VTS, it includes the general description of the functionality, context and design of the VTS. And it also provides the background of voting rules and the instructions of the system.

Section 3 is the architectural design that introduces a modular program structure and the decomposition of the subsystems in the architectural design, the design rationale.

Section 4 concerns data structure design.

Section 5 shows the functional pseudocode description in 3.2, providing a summary of the algorithm in 3.2

Section 6 shows the interaction between users and the system. It includes the user interface, and image/screen shot of design of human interaction.

Section 7 provides a cross reference that traces components and data structures to the requirements in the SRS document.

## 1.4 Reference Material

[IEEE] The applicable IEEE standards are published in “IEEE Standards Collection,” 2001

[Bruade] The principal source of textbook material is “Software Engineering: An ObjectOriented Perspective” by Eric J. Bruade (Wiley 2001).

Reaves, Michael J. “Software Project Management Plan Jacksonville State University Computing and Information Sciences Web Accessible Alumni Database.” Jacksonville State University, 2003.

Reaves, Michael J. “Software Requirement Specifications Jacksonville State University Computing and Information Sciences Web Accessible Alumni Database.” Jacksonville State University, 2003.

## 1.5 Definitions and Acronyms

VTs: Voting System software

OPL: The voting rule/ method of Open Party List

IR: The voting rule/ method of instant runoff.

SRS: Software requirements specification

DFD: Data Flow Diagram

PDL: Procedural Description Language

## 2. SYSTEM OVERVIEW

Functionality:

- Read Voting File/Ballots: Read data from the voting files
- Handle tie: Handle tie when there is a tie between two candidates by flipping a coin
- Generate voting result: generate voting result with given ballots
- Record/Display Result: display and record the voting result
- Generate Audit File: Record the log of how the votes goes to each candidates in election
- Export Audit File: This feature is to export the audit file after election results are completed.

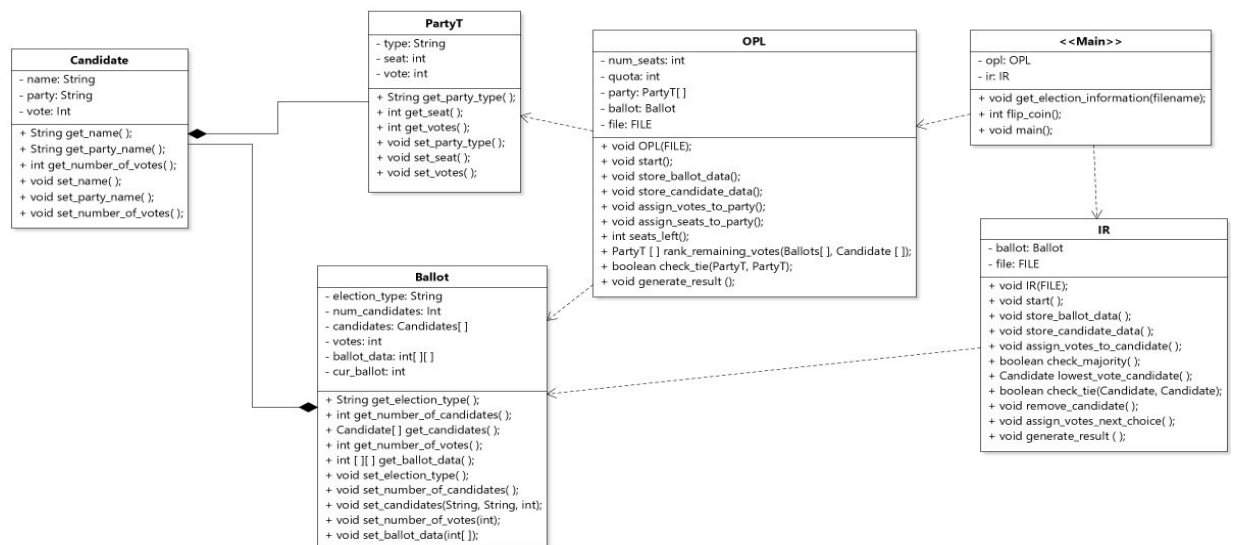
- Close Project: This feature is to exit programs by users after evaluating the election results or they can close immediately.
- Import Ballot File: This feature is to allow users to import ballot files into the system.
- View: This feature is to allow users to review the latest audit report and election result report.
- Instant Runoff Voting: Voters rank the candidates in the order of their preference.
- Open Party List Voting: Voters must cast a vote for an individual candidate and vote counts for the specific candidate as well as for the party.
- Program start: Start the program

### Background:

- OPL: open list describes any variant of party-list proportional representation where voters have at least some influence on the order in which a party's candidates are elected.
- IR: Runoff voting can refer to a two-round system, a voting system used to elect a single winner, whereby only two candidates from the first round continue to the second round, where one candidate will win.

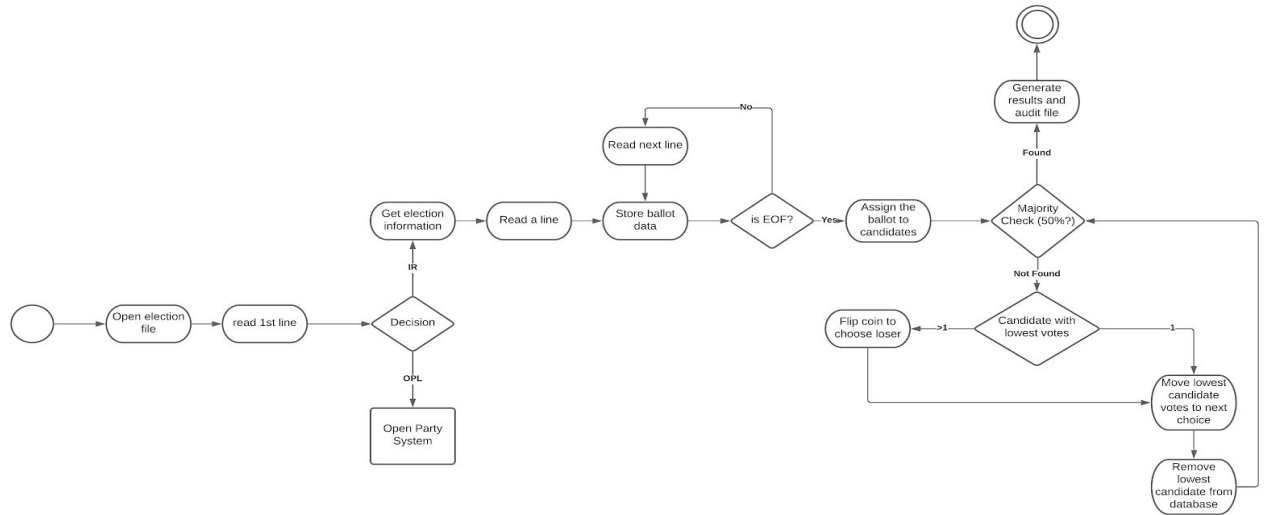
## 3. SYSTEM ARCHITECTURE

### 3.1 Architectural Design

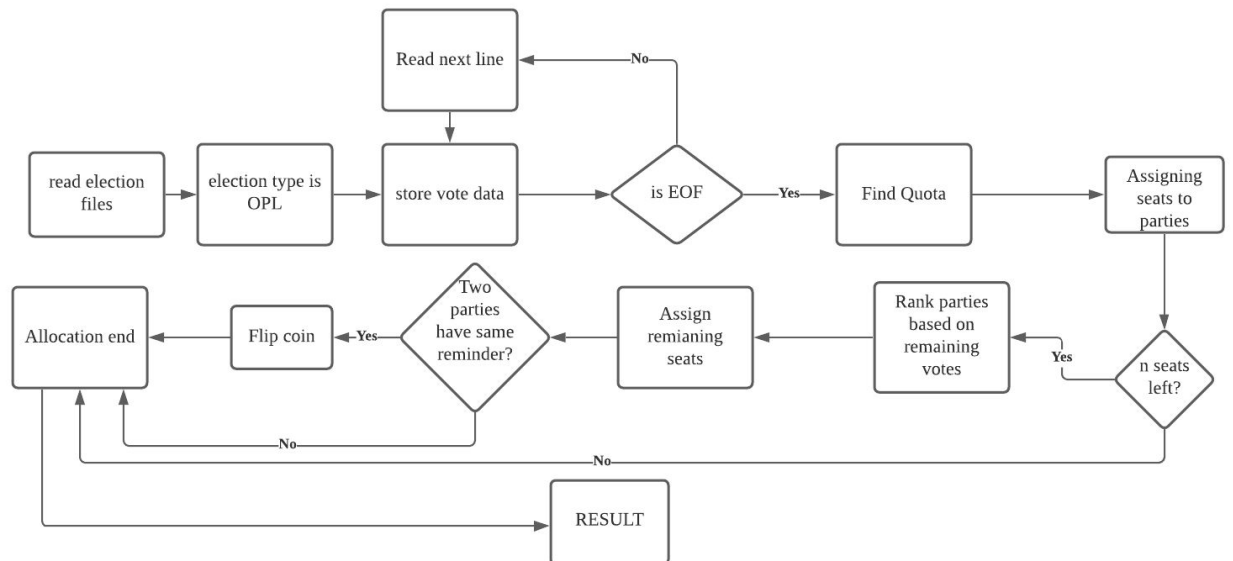


### 3.2 Decomposition Description

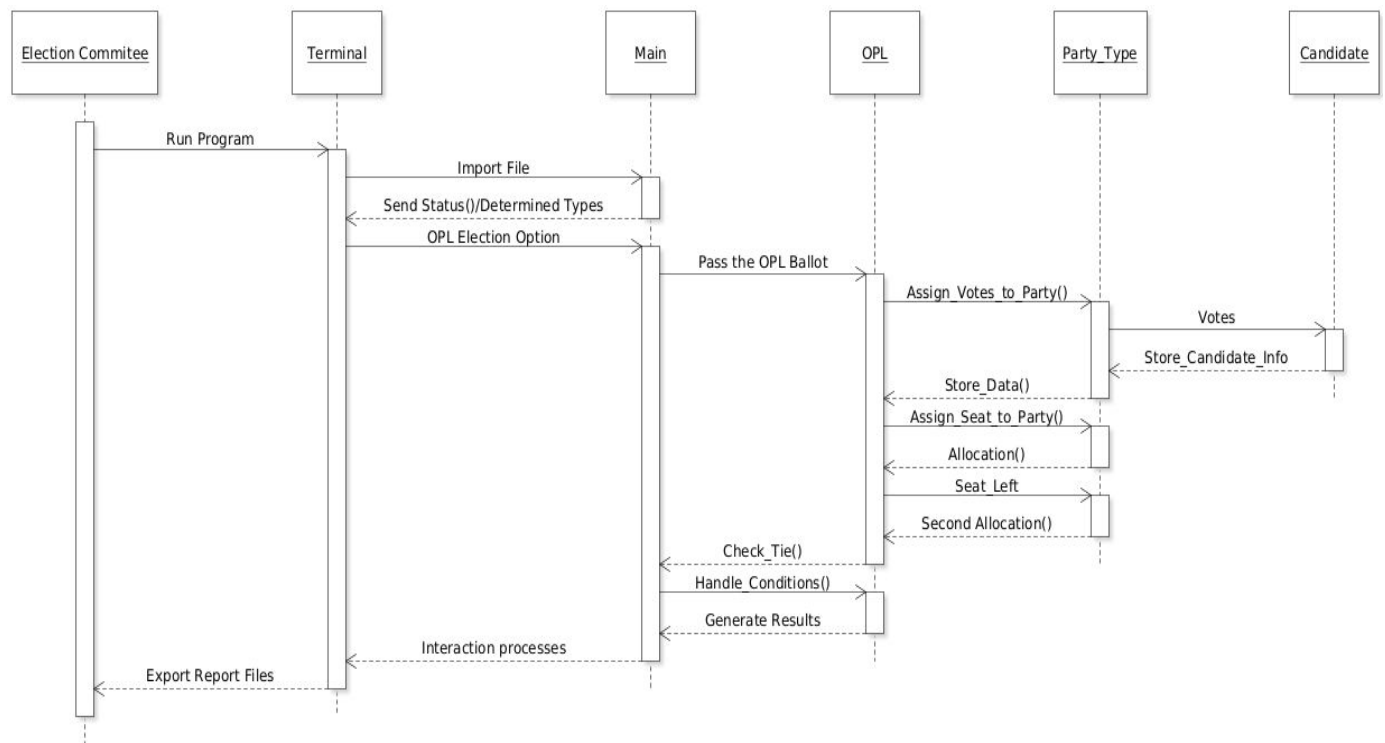
For Instant Runoff Listing.



### For Open Party Listing.



### 3.3 Sequence Diagram



## 4. DATA DESIGN

### 4.1 Data Description

The data of election type, candidates, and ballot will be read from the file the user has provided. Then it will transform and be stored into the database using the structure of the provided file. After the ballot's data are stored, the system checks the type of election data and will proceed accordingly.

<b>Ballot</b> election_type num_candidates candidates votes ballot_data cur_ballot	<b>Candidate</b> name party votes	<b>PartyT</b> type seat vote
<b>OPL</b> num_seats quota party ballot	<b>IR</b> total_votes ballot file	

## 4.2 Data Dictionary

System entities	Class	Type	Description
ballot	IR	Ballot	Storage for data from the file
ballot	OPL	Ballot	Storage for data from the file
ballot_data	Ballot	int[ ][ ]	Ballots data from file
candidates	Ballot	Candidates[ ]	List of candidates belong to election
cur_ballot	Ballot	int	Current ballot number
election_type	Ballot	String	Type of election
file	IR	FILE	File that is provided by user
file	OPL	FILE	File that is provided by user
ir	Main	IR	Access point of instant



			runoff system
name	Candidate	String	Name of the candidate
num_candidates	Ballot	int	Number of candidates
num_seat	OPL	int	Number of available seat
opl	Main	OPL	Access point of Open Party Listing system
party	OPL	PartyT[ ]	List of parties in this election
party	Candidate	String	Type of party the candidate belong
quota	OPL	int	The required number of votes for each seat in the first allocation
seat	PartyT	int	Number of seats the party received
type	PartyT	String	Type of party
vote	PartyT	int	Number of votes the party received
vote	Candidate	int	Number of votes the candidate received
vote	Ballot	int	Total number of votes available

## 5. COMPONENT DESIGN

**Pseudocode** for Instant Runoff Voting:

```

file = open(election file);
string = read(1st line);
if (string == "IR") {
    for (hasnextline) {
        ballot_data += read(line);
        line++;
    }
    assign_ballots_to_candidates(Candidates, Ballot);

    while {
        if (Candidates.votes > 50%) {
            return Candidate;
        }
        else {
            distribute_votes(Candidates);
            remove_lower(Candidate);
            if (tie) {
                flip_coin();
            }
        }
    }
}
}

```

### Pseudocode for Instant Open Party List Voting:

```

available seats
total votes
calculate quota

for each party:
    assigning seats to parties with formula in the first allocation
    record reminder for each

if there are n (n>0) seats left:
    rank parties based on the reminders
    assign remaining seats

    if two parties have the same reminders
        flip coins
    else:
        return
else:
    return

```

## 6. HUMAN INTERFACE DESIGN

### 6.1 Overview of User Interface

#### 6.1.1 Functionality

The voting system could help users count huge election votes automatically.

#### 6.1.2 Features

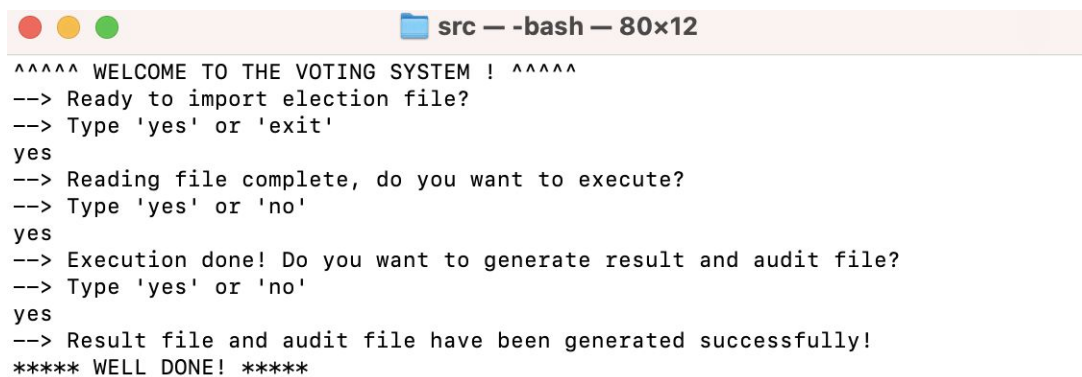
- 1) Users need to put the election file in the same directory of the system.
- 2) The system will generate the result file and audit file automatically.
- 3) The system supports Instant runoff voting and Open Party List Ballot so far.

#### 6.1.3 Feedback Information

- 1) Import File: The system will ask permission of reading in the file, or exit
- 2) Execution: The system will ask permission for execution after completing file reading. Type 'yes' or 'no' for moving to the next step or exit.
- 3) Files Generating: The system will ask permission of generating the result file and the audit file. Type 'yes' or 'no' to end.

### 6.2 Screen Images

a) Example that shows how to import, execut, and file generating.

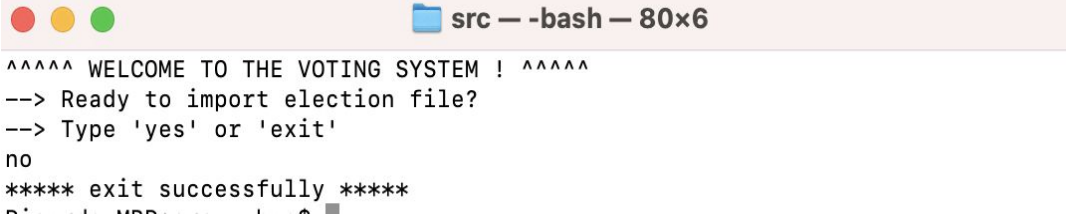


```

src — -bash — 80x12
^^^^^ WELCOME TO THE VOTING SYSTEM ! ^^^^^
--> Ready to import election file?
--> Type 'yes' or 'exit'
yes
--> Reading file complete, do you want to execute?
--> Type 'yes' or 'no'
yes
--> Execution done! Do you want to generate result and audit file?
--> Type 'yes' or 'no'
yes
--> Result file and audit file have been generated successfully!
***** WELL DONE! *****

```

b) Example that exit the system while import.

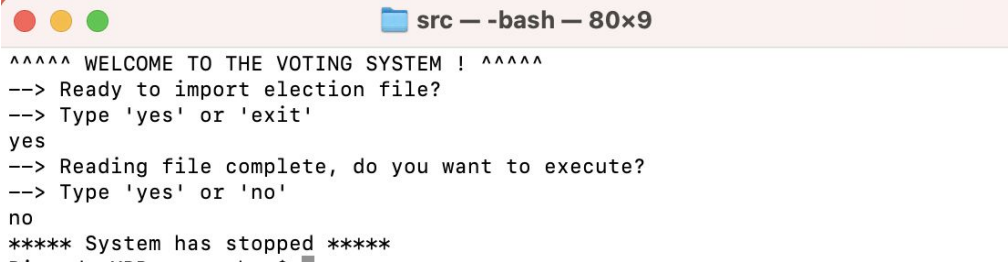


```

src - -bash - 80x6
^^^^^ WELCOME TO THE VOTING SYSTEM ! ^^^^^
--> Ready to import election file?
--> Type 'yes' or 'exit'
no
***** exit successfully *****

```

*c) Examples that stop execution.*

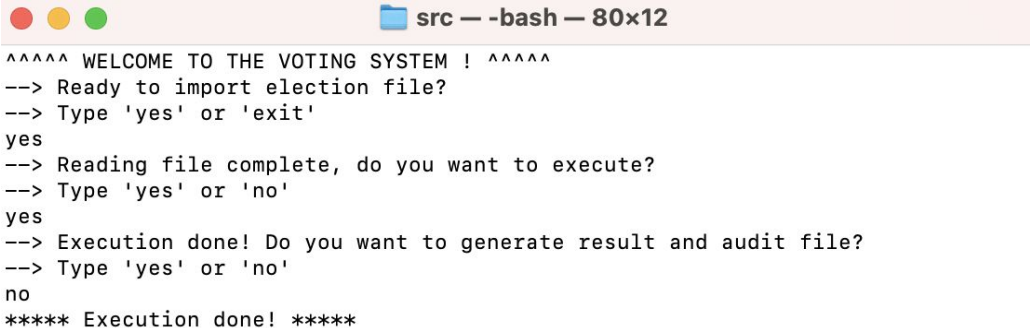


```

src - -bash - 80x9
^^^^^ WELCOME TO THE VOTING SYSTEM ! ^^^^^
--> Ready to import election file?
--> Type 'yes' or 'exit'
yes
--> Reading file complete, do you want to execute?
--> Type 'yes' or 'no'
no
***** System has stopped *****

```

*d) Example that stops file generating.*



```

src - -bash - 80x12
^^^^^ WELCOME TO THE VOTING SYSTEM ! ^^^^^
--> Ready to import election file?
--> Type 'yes' or 'exit'
yes
--> Reading file complete, do you want to execute?
--> Type 'yes' or 'no'
yes
--> Execution done! Do you want to generate result and audit file?
--> Type 'yes' or 'no'
no
***** Execution done! *****

```

## 6.3 Screen Objects and Actions

All the interaction will be accomplished in the terminal. The system will show instructions and hints once the user starts the system. Users need to type commands to move to the next step or exit. (for more details, please move to 6.1 and 6.2)

## 7. REQUIREMENTS MATRIX

Name	ID in SRS	Functionality/ Description	Reference in SDD	Functionality/ Description	Additional Note
Reading File	VTS_01	Read data from the voting files	6.1 6.2	Interface Description	
Handle Tie	VTS_02	Handle tie	4.2	Data structure	
Generate voting result	VTS_03	generate voting result	6.1 6.2	The way to generate file	
Record/Display Result	VTS_04	Display Result	6.2	Show the election result	
Generate Audit File	VTS_05	Record the log	6.1 6.2	Generate audit file	
Export Audit File	VTS_06	Export the audit file	6.1 6.2	Export the audit file	
Close Project	VTS_07	Exit the system	6.1 6.2	Exit	
Import Ballot File	VTS_08	Import election files	6.1 6.2	Import file into system	
Instant Runoff Voting	VTS_10	IR election type algorithm	2/ 3.1 4.1/ 4.2	Data structure and diagram	
Open Party List Voting	VTS_11	OPL election type algorithm	2/ 3.1 4.1/ 4.2	Data structure and diagram	