Name	Prompting a User for file
ID	UC_01
Description	The system will display a prompt indicating to the user to input the file name. **“Enter the name of the file containing the ballots: ”**
Actors	Election Officials, Programmers, Testers
Organizational Benefits	Removes complexity from making mistakes when processing a file, by clearly indicating directions that can be followed.
Frequency of Use	Every time a file needs to be run at every election.
Triggers	When a voting official wants to run a file containing ballot information.
Preconditions	The user has a file that needs to be processed
Postconditions	The system has saved the name of file
Main Course	1. The voting system is run (started) 2. The system checks if it has command line arguments for file name 3. In this case there are no command line arguments 4. The system then displays the prompt: ***a. “Enter the name of the file containing the ballots: ”*** 5. The user enters a name of a election file. 6. The name gets saved in the program for later use
Alternate Courses	AC1: There are command line arguments 1. The voting system is run (started) 2. The system checks if it has command line arguments for file name 3. In this case there are command line arguments 4. The system does not display a prompt 5. The program saves the name from the command line argument AC2: There are no command line arguments and no input when prompted. 1. The voting system is run (started) 2. The system checks if it has command line arguments for file name 3. In this case there are no command line arguments 4. The system then displays the prompt: ***a. “Enter the name of the file containing the ballots: ”*** 5. The user does not input a file name 6. The program outputs an error or prompts the user again
Exceptions	EX1: The user has no files to run.

Name	Check if the file exists
ID	UC_02
Description	After opening the file will check if the file is NULL (exists)
Actors	Election Officials, Programmers, Testers
Organizational Benefits	Helpful to users to handle their errors when using system
Frequency of Use	Every election
Triggers	After user inputs file
Preconditions	User input a file when prompted
Postconditions	File either exists and the system continues or prints error message to user
Main Course	1. The file name from the user is searched for in the directory. 2. Check if the file is NULL. (AC1) 3. Check if the file is a comma-separated value file. (AC2)
Alternate Courses	AC1 File does not exist 1. If the file is NULL (does not exist) then print an error message to the screen telling the user the file does not exist, then prompt the user to enter a filename again (UC_01). AC2 File is not the correct type 1. If the file is not a CSV file, print an error message to the screen telling the user the file is not the correct format, then prompt the user to enter a filename again (UC_01).
Exceptions	EX1: There is no need to run a file to be processed

Name	Determine if the file is IR or CPL
ID	UC_03
Description	The system will determine if the file is for an IR or CPL election, by looking into the file’s first line.
Actors	Programmer, Tester
Organizational Benefits	Saves time by replacing typing in the type of election each time the program is run for an automated step where the system determines it.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	A file of the correct format has been inputted into the system for processing.
Preconditions	The file is known to exist and is known to be in the correct format.
Postconditions	The type of election is saved to be later used by the program and the audit file.
Main Course	1. Check if there is an error opening the file. (EX1) 2. Open the file for reading. 3. Read the first line of the CSV file. 4. Check if the first line is either ‘IR’ or ‘CPL’. (AC1) 5. Store the type of election for later use.
Alternate Courses	AC1: The first line does not match IR or CPL standards 1. Close the file and stop all processes in the program. Print out an error message to the user and prompt them to enter a filename again (UC_01).
Exceptions	EX1: Error occurred while opening file 1. Stop all processes in the program. Print out an error message to the user and prompt them to enter a filename again (UC_01).

Name	Save IR file Information
ID	UC_04
Description	The system saves into a data structure the file election information such as the number of candidates, candidates separated by commas, and the number of ballots.
Actors	Programmer, Tester
Organizational Benefits	Saves time by automating the process of gathering information about the election, as opposed to typing in each bit of information each time the program is run.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	The system has finished determining which type of election the file is for.
Preconditions	The system has determined the type of election is IR.
Postconditions	The number of candidates, candidate names, and the number of ballots are all saved for later use.
Main Course	1. The program saves the number of candidates 2. The program saves the names of the candidates 3. The program saves the number of ballots
Alternate Courses	AC1: Line does not match election file standards 1. Close the file and stop all processes in the program. Print out an error message to the user and prompt them to enter a filename again (UC_01).
Exceptions	EX1: Error occurred while reading line 1. Stop all processes in the program. Print out an error message to the user and prompt them to enter a filename again (UC_01).

Name	Save IR Ballots
ID	UC_05
Description	Make a data structure to save the IR ballots information
Actors	Programmer, Tester
Organizational Benefits	Saves ballots to the program to be used/seen later for election algorithm processing or any other needed information
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	User inputs existing files that are opened by the system to be read
Preconditions	Existing file is opened and checked to be IR voting
Postconditions	Every ballot will be saved to a data structure
Main Course	1. Create a data structure to store ballot information 2. Read every ballot information in opened CSV file starting from line where ballots are located. 3. Save information to program
Alternate Courses	AC1: Unable to save information 1. Create a data structure to store ballot information 2. Read every ballot information in opened CSV file starting from line where ballots are located. 3. Failed to save information 4. Indicate an error occurred and attemp to run again.
Exceptions	EX1: Ballot is incomplete/missing information

Name	Run IR Algorithm
ID	UC_06
Description	The system will process IR ballots according to the IR voting algorithm.
Actors	Election Officials, Programmers, Testers
Organizational Benefits	This facilitates the process of determining a winner when using Instant runoff voting by assuring the process is automated and not by hand.
Frequency of Use	Each time an IR election file needs to be run to test the system, as well as at every election time.
Triggers	The file has been identified as an IR and it needs to be processed.
Preconditions	- The file is an IR election file - IR CSV has been saved to program
Postconditions	- Results of running IR Algorithm are determined
Main Course	Part 1 of algorithm: Processing Ballots 1. Look at each individual ballot 2. Process the ranking of each ballot. 3. Once all ballots have been processed the current state is saved Part 2 of algorithm: Ranking candidates 4. Count the number of ballots each candidate has received 5. Set a ranking for each candidate Part 3 of algorithm: Determine winner 6. Look at each candidate ranking 7. Candidates who receive over 50% of the first-choice votes are declared the winner.
Alternate Courses	AC1: There is no candidate with a clear majority Run part 1 of algorithm; Run part 2 of algorithm; Part 3 of algorithm: Determine winner 1. Look at each candidate ranking 2. There is no candidate with a clear majority Part 4 of algorithm: Drop candidate 3. Select the candidate ranked last 4. Now run the IR algorithm for the ballots that were given to the selected candidate. 5. This candidate is now dropped. AC2: For main course and AC1 there is a tie when ranking candidates 1. When getting to ranking part of the algorithm there is a tie between 1 or more candidates 2. In this case refer to UC_14
Exceptions	EX1: Ballot is incomplete/missing information EX2: Candidates information is incomplete/missing.

Name	Name IR Audit File
ID	UC_07
Description	The System will create an audit file for IR, this file will be of type .txt
Actors	Election Officials, Programmers, and Testers
Organizational Benefits	Will automate file naming therefore it is clear and concise what kind of audit file is being produced.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	The program has finished running the IR algorithm.
Preconditions	An audit file for IR has been created
Postconditions	An audit file is constructed and saved to the proper directory.
Main Course	1. Create a file called IR_Audit.txt 2. Make sure to check that the audit file has been created 3. In this case it has proceed to UC_08
Alternate Courses	AC1: Failed to make an IR audit file 1. Create a file called IR_Audit.txt 2. Make sure to check that the audit file has been created 3. In this case it failed and it alerts the user of error 4. Retry to create a file
Exceptions	EX1: The is no need to create an IR audit file

Name	Produce IR Audit File
ID	UC_08
Description	The System will save into an IR audit file type .txt the following: Number of candidates, number of ballots, names of each candidate, 1st count: original first choices, 2nd count: transfer of disqualified candidate and new total, and so on for each round, and the winner.
Actors	Election Officials, Programmers, and Testers
Organizational Benefits	Will automate saving results of elections which will save time from having to gather data manually into a file.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	The program has finished determining the winner of the election.
Preconditions	All election information and winners have been stored by the program to be accessed by the audit file.
Postconditions	An audit file is constructed and saved to the proper directory.
Main Course	1. Write the number of candidates who were in the election 2. Write the number of ballots cast for election 3. Write the names of each of the candidates in the election 4. Write next to the names of the first count results from the election 5. Write the winner at the end of the file
Alternate Courses	AC1: There is more than 1 round of calculations 1. Write the number of candidates who were in the election 2. Write the number of ballots cast for election 3. Write the names of each of the candidates in the election 4. Write next to the names of the first count results from the election 5. There is a second round as there was no clear winner in first round 6. Write next to the first count the results of the second count from the election. 7. Output the winner at the end of the file AC2: There are multiple rounds of calculations These will be the same steps as AC_01 but keep repeating for each round that had been run by the IR algorithm.
Exceptions	EX1: There was no clear winner at the end.

Name	Save CPL file Information
ID	UC_09
Description	The system saves into a data structure the file election information such as the number of parties, the parties separated by commas, the number of seats, and the number of ballots.
Actors	Programmer, Tester
Organizational Benefits	Saves time by automating the process of gathering information about the election, as opposed to typing in each bit of information each time the program is run.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	The system has finished determining which type of election the file is for.
Preconditions	The system has determined the type of election is CPL.
Postconditions	The number of parties, party names, number of seats, and number of ballots are all saved for later use.
Main Course	1. The program saves the number of parties 2. The program saves the names of the parties 3. The program saves the number of seats 4. The program saves the number of ballots
Alternate Courses	AC: file does not match election file standards 1. Close the file and stop all processes in the program. Print out an error message to the user and prompt them to enter a filename again (UC_01).
Exceptions	EX1: Error occurred while reading line 1. Stop all processes in the program. Print out an error message to the user and prompt them to enter a filename again (UC_01).

Name	Save CPL Ballots
ID	UC_10
Description	Make a data structure to save the CPL ballots information
Actors	Programmer, Tester
Organizational Benefits	Saves ballots to the data structure to be used/seen later for election processing or any other needed information
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	User inputs existing files that are opened by the system to be read
Preconditions	Existing file is opened and checked to be CPL voting
Postconditions	Every ballot will be saved to a data structure
Main Course	1. Create a data structure to store ballot information 2. Read ballot information in opened CSV file starting from line 6 3. Save information to data structure 4. Parse to the next ballot in file 5. Continue for all ballots
Alternate Courses	AC1: Unable to save information 1. Create a data structure to store ballot information 2. Read every ballot information in opened CSV file starting from line where ballots are located. 3. Failed to save information 4. Indicate an error occurred and attemp to run again.
Exceptions	EX1: Ballot is incomplete/missing information

Name	Run CPL Algorithm
ID	UC_11
Description	The system will process CPL ballots using the CPL voting algorithm.
Actors	Programmer, Tester
Organizational Benefits	This facilitates the process of determining a winner when using CPL voting by assuring the process is automated and not by hand.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	CPL data structure needs to be processed.
Preconditions	- Existing file is opened and checked to be CPL voting - CPL Ballots have been saved to data structure
Postconditions	- Results of running CPL ballots are determined
Main Course	Part 1 of algorithm: Process CPL Ballots 1. Look at each individual ballot 2. Process the ranking of each ballot. 3. Once all ballots have been processed the current state is saved Part 2 of algorithm: Determine Seats for each party 4. Count the number of ballots each Party has received 5. Party receive seats depending on the percentage of votes determined by the CPL algorithm equation. Part 3 of algorithm: Allocate Seats 6. Allocate seats to each party candidate according to how many seats each party has received.
Alternate Courses	AC1: There is a Tie in determining how many seats are allocated to each party Part 1 of algorithm: Process CPL Ballots 1. Look at each individual ballot 2. Process the ranking of each ballot. 3. Once all ballots have been processed the current state is saved Part 2 of algorithm: Determine Seats for each party 4. Count the number of ballots each Party has received 5. Parties receive seats depending on the percentage of votes determined by the CPL algorithm equation. 6. There is a tie between 2 or more parties when dividing seats see UC_14
Exceptions	EX1: Ballot is incomplete/missing information

Name	Name CPL Audit File
ID	UC_12
Description	The System will create an audit file for CPL, this file will be of type .txt
Actors	Election Officials, Programmers, and Testers
Organizational Benefits	Will automate file naming therefore it is clear and concise what kind of audit file is being produced.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	The program has finished running the IR algorithm.
Preconditions	An audit file for IR has been created
Postconditions	An audit file is constructed and saved to the proper directory.
Main Course	1. Create a file called CPL_Audit.txt 2. Make sure to check that the audit file has been created 3. In this case it has proceed to UC_12
Alternate Courses	AC1: Failed to make an IR audit file 1. Create a file called CPL_Audit.txt 2. Make sure to check that the audit file has been created 5. In this case it failed and it alerts the user of error 6. Retry to create a file
Exceptions	EX1: The is no need to create an CPL audit file

Name	Produce CPL Audit File
ID	UC_13
Description	The System will save into a CPL audit file the following: Number of parties, number of ballots, names of each party, number of ballots for each party, 1st allocation of seats, Remaining votes, second allocation of seats, etc.., final seat total, % of the vote to % of seats, and winner(s).
Actors	Election Officials, Programmers, and Testers
Organizational Benefits	Will automate saving results of elections which will save time from having to gather data manually into a file.
Frequency of Use	Each time a file needs to be run to test the system, as well as at every election time.
Triggers	The program has finished determining the winner of the election.
Preconditions	All election information and winners have been stored by the program to be accessed by the audit file.
Postconditions	An audit file is constructed and saved to the proper directory.
Main Course	1. Write the number of parties who were in the election 2. Write the number of ballots cast for election 3. Write the names of each party in the election 4. Write the number of ballots received for each party 5. The program saves the names of candidates in parties who got seats 6. Write 1st round of seat allocation as the CPL algorithm progressed 7. Write final seat total, % of the vote to % of seats 8. Write the winner(s) at the end of the file
Alternate Courses	AC1: There is more than 1 round of calculations 1. Write the number of parties who were in the election 2. Write the number of ballots cast for election 3. Write the names of each party in the election 4. Write the number of ballots recieved for each party 5. The program saves the names of candidates in parties who got seats 6. Write multiple rounds of seat allocation as the CPL algorithm progressed 7. Write final seat total, % of the vote to % of seats 8. Write the winner(s) at the end of the file
Exceptions	EX1: There is no clear winner at the end.

Name	Coin Flip
ID	UC_14
Description	In the case of a tie, the program will flip a coin to determine the winner of an election, or in the case of more than two candidates, randomly decide who the winner is.
Actors	Election Officials, Developers, and Testers
Organizational Benefits	Will make deciding who the winner is faster than redoing and recounting the election.
Frequency of Use	Whenever a tie occurs between candidates at the end of an election.
Triggers	If no candidate has a majority in IR voting and some candidates have the same amount of popular votes popularity-wise. If seats remain in CPL and the remaining parties have the same number of remaining votes.
Preconditions	Any number of candidates are tied in some way at the end of counting an election.
Postconditions	One candidate is fairly selected as the winner of the election, regardless of type.
Main Course	1. If more than two candidates are tied, use the alternative method. (AC1) 2. The system uses an algorithm that will choose one of two choices with an even 50-50 chance. 3. Assign the two candidates to the algorithm, and use it to choose one winner. 4. Save this winner in the election audit file.
Alternate Courses	AC1: More than two tied candidates 1. Assign each candidate a number to represent them, ranging from one to the number of tied candidates. 2. Use a random number generator with a range of one to the number of tied candidates to obtain a number. 3. Match this number with the candidate assigned to it to determine the winner of that particular election. 4. Save this winner in the election audit file.
Exceptions	

Name	Display Winner(s)
ID	UC_15
Description	After the file has been read and the votes have been tallied, the winner of the election is displayed to the user.
Actors	Election Officials, Developers, Testers
Organizational Benefits	By visually displaying the winner to the user, it will make it easier for the user to understand who won the election.
Frequency of Use	Run once per Election by election by the officials and possibly 1000 times by tester.
Triggers	Once the program has determined the winners of the election.
Preconditions	The file has been read and the votes have been tallied. The program has determined the winner of the election, either by votes or coin flip. The file has no errors.
Postconditions	The winner(s) of the election are displayed on the screen for the user to see.
Main Course	1. The program confirms that the winner(s) have been determined. 2. If the election was an Instant Runoff election, only one candidate will be displayed to the screen, that being the winner. 3. If the election was a Closed Party List election, all candidates who filled the open seats will be displayed on the screen. 4. Any additional information about the election will also be displayed as well.
Alternate Courses	AC1: Failed to display 1. The program confirms that the winner(s) have been determined. 2. It will display the winning candidate’s name on the screen via text. 3. Error occurs when displaying information, print an error message to user.
Exceptions	EX1: There is not information to display

Name	Help feature
ID	UC_16
Description	After the user has started the system, if they do not know what to do they can type in “help” to the input to be given a more detailed description of what to do.
Actors	Election Officials, Developers, Testers
Organizational Benefits	Will provide users with helpful information on how to use the system properly.
Frequency of Use	Can be used anytime the user inputs “help”
Triggers	User types “help” into input
Preconditions	The program has been started and prompted the user for input.
Postconditions	The help message is printed to the screen.
Main Course	1. The user inputs “help” 2. The help message is printed to the screen.
Alternate Courses	1. User misspells or types in invalid input an error message will be printed telling the user to type “help” if they need help.
Exceptions	Error must be printed to the user if they type in false input.