Name	Input Election Data CSV File
ID	UC_001
Description	As an Election Official, I want to be able to input an election data CSV file into the system.
Actors	Election Official
Organizational Benefits	Allows for efficient and error-free tabulation of election results with a quick turnaround.
Frequency of Use	This use case is performed whenever an election result is needed.
Triggers	The election official initiates the process of starting the election counter program and inputs a file name.
Preconditions	1. The file strictly follows formatting guidelines for IR or OPL election types and has no errors. 2. The file exists in the same directory as the program. 3. The file is a CSV.
Postconditions	The system successfully opens the file.
Main Course	1. The user starts the Election Counter program and ensures the file they want to process meets the above preconditions. 2. The user follows program directions and enters the filename.
Alternate Courses	None
Exceptions	EX1: Incorrect file name is typed in 1. The system lets the user know that the file does not exist and prompts the user to re-enter a filename. 2. User enters the correct filename for the election they want to process. EX2: User stops program from the CLI 1. The program stops running and will reset and have to be restarted from the CLI.

Name	Read Input File Header to Extract Election Information
ID	UC_002
Description	As a Developer, I want to read the input file header to extract relevant election information, such as the type of election (IR or OPL) and the number of candidates and parties.
Actors	Developer
Organizational Benefits	Facilitates the extraction of critical election information from the input file header, ensuring accurate processing of election data.
Frequency of Use	This use case is performed during the initial setup of an election and whenever a new input file is provided.
Triggers	Developer initiates the process of reading the input file header to extract election information.
Preconditions	1. The input file is available. 2. The developer has access to the input file. 3. The input file follows the specified format (CSV) for either IR or OPL election types.
Postconditions	The system successfully reads and extracts election information from the input file header.
Main Course	1. System opens the input file for reading as mentioned in UC_001. 2. The system reads the first line of the input file, which contains the type of election: IR or OPL. 3. The system extracts the number of candidates and parties from the second line. 4. The extracted election information is stored for further processing.
Alternate Courses	None
Exceptions	EX1: File cannot be opened 1. If the election file cannot be opened or read due to permissions or file integrity issues, the developer logs an error and notifies relevant parties for resolution.

Name	Read Remaining File to Extract Ballot Data
ID	UC_003
Description	As a Developer, I want to read the remaining election file after extracting header information to access and extract the ballot data. This step is essential for processing the election, counting votes, and determining the election results.
Actors	Developer
Organizational Benefits	Enables the software to access and process the ballot data required for vote counting and election result determination.
Frequency of Use	This use case is performed during the election processing phase and is integral to the vote counting.
Triggers	Developer initiates the process of reading the remaining election file after extracting header information.
Preconditions	1. Header information, including the election type (IR/OPL), number of candidates, and candidate names, has already been extracted from the election file. 2. The developer has appropriate permissions and access to the election file. 3. The file is a CSV and follows the correct formatting guidelines for OPL or IR election processing.
Postconditions	The remaining election file is successfully read, and the ballot data is extracted for further processing, including vote counting and result determination.
Main Course	1. Developer, having extracted the header information, accesses the remaining election file for reading. 2. The software opens the election file for reading and positions the reading pointer at the beginning of the ballot data section. 3. Developer iterates through the file, line by line, to extract ballot data. 4. For each line, the software parses the comma-delimited values to identify the ranking or selection made by the voter on the ballot. 5. The extracted ballot data is stored in memory or data structures for subsequent processing, including vote counting. 6. Developer continues reading and extracting ballot data until the

	entire file has been processed.
Alternate Courses	None
Exceptions	EX1: File cannot be opened 1. If the election file cannot be opened or read due to permissions or file integrity issues, the developer logs an error and notifies relevant parties for resolution.

Name	Count Votes for IR Election with Popularity as Tiebreaker
ID	UC_004
Description	As an Election Official, I want to be able to run an Instant Runoff (IR) election, and in the event of no clear majority, declare the winner based on popularity.
Actors	Election Official
Organizational Benefits	Quick calculation of winners for IR voting. Ensuring a transparent and unbiased method for declaring a winner in IR elections when there is not a clear majority, maintaining the integrity of the election process.
Frequency of Use	This use case is performed whenever an IR election is conducted.
Triggers	The election official inputs an IR election file into the program.
Preconditions	1. A file was successfully opened in UC_001 2. The header of the file was successfully read in UC_002. 3. The file has the appropriate header for IR election. 4. The ballot data was correctly parsed by the system in UC_003. 5. Ballots are prepared and have no formatting errors. 6. Ranking numbers are without missing values (e.g., “1, 3, 4” with missing “2” will not occur.).
Postconditions	The IR election is successfully run, and the winner is declared. If there is no clear majority, the winner is based on popularity.

Main Course	1. The Election Official inputs a file containing IR election data. 2. The system opens the file as stated in UC_001. 3. The system correctly identifies the header for IR election and begins processing the file. 4. The system calculates the results based on the IR Voting algorithm. 5. If there is a clear majority after all votes are tallied, the system declares the candidate with the majority of votes as the winner. 6. If no clear majority is evident: a. The system identifies the candidates with the highest popularity (most votes). b. The candidate with the highest popularity is declared the winner. c. The system updates the audit file and election results to reflect the winner based on popularity. 7. The updated election results are displayed for public viewing and an audit file is generated if the program runs uninterrupted.
Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 1. The program stops running and the gathered file data is discarded. 2. No display is shown to the user and no audit file is generated.

Name	Count votes for OPL Election (Treat Independents as Grouped)
ID	UC_005
Description	As an Election Official, I want to be able to run an Open Party Listing (OPL) election while treating all independent candidates as if they belong to a single party.
Actors	Election Official
Organizational Benefits	Helps tally votes and assign seats for OPL elections. Streamlines the OPL election process by grouping independent candidates, making it easier to manage and calculate results.
Frequency of Use	This use case is performed whenever an OPL election is conducted.

Triggers	The election official inputs an OPL election file into the program. The system correctly recognizes the header of an opened file as “OPL” and has extracted ballot data from the file.
Preconditions	1. A file was successfully opened in UC_001 2. The header of the file was successfully read in UC_002. 3. The file has the appropriate header for the OPL election. 4. The ballot data was correctly parsed by the system in UC_003. 5. Ballots are prepared and have no formatting errors. 6. Each ballot indicates the choice for one candidate within the available parties or independents.
Postconditions	The OPL election is successfully run with independent candidates treated as a single party. Seats are proportionally assigned to each party and the most popular candidates from each party win the seats.
Main Course	1. The Election Official inputs a file containing OPL election data. 2. The system opens the file as stated in UC_001. 3. The system correctly identifies the header for OPL election and begins processing the file. 4. The system calculates the results based on the OPL Voting algorithm. 5. The system identifies all the candidates and their respective parties or independent status. 6. If independent candidates are present: a. The system groups all independent candidates into a single “independent” party. b. This grouping is reflected in the candidates list. 2. The system calculates the election results based on voter choices within parties or the single “independent” party. 3. The election results are displayed for public viewing. An audit file is generated if the program runs uninterrupted.
Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 1. The program stops running and the gathered file data is discarded. 2. No display is shown to the user and no audit file is generated.

Name	Ballots should be fairly shuffled before processing
ID	UC_006
Description	As an election official, I want the ballot order to be shuffled before votes are counted for OPL elections to prevent unfair bias towards a particular candidate.
Actors	Election Official
Organizational Benefits	Maintains the integrity and authenticity of the election results by giving all candidates a fair chance.
Frequency of Use	This use case is performed when the system runs an OPL election.
Triggers	The system recognizes that the input file is for an OPL election and parses the ballots for that election.
Preconditions	The file being processed is for an OPL election, is free from errors and the ballot data is correctly stored in an internal data structure.
Postconditions	The ballots are shuffled randomly before votes are counted.
Main Course	1. Developer opens the software source code. 2. Developer locates the section of code responsible for generating the audit file and election results file. 3. Within the code, the developer sets the permissions for both the audit file and election results file to "read-only." 4. The developer saves the code changes. 5. The audit files and election results are now configured as read-only within the software.
Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 1. The program stops running and the gathered file data is discarded. 2. No display is shown to the user

Name	Test Shuffling of Ballots
ID	UC_007
Description	As a Tester, Developer and Election Official, I want to test the shuffling of ballots to ensure that the process works correctly.
Actors	Tester, Developer, Election Official
Organizational Benefits	Ensures the fairness and effectiveness of the ballot shuffling process in the voting system.
Frequency of Use	This use case is performed during system testing phases.
Triggers	User types in the appropriate testing command with the flag to toggle shuffling of ballots into the CLI.
Preconditions	1. The voting system is set up for testing. 2. Users type in the correct command to access the testing environment for ballot shuffling. 3. The system's shuffling system is set up for testing. 4. The users testing files are free from formatting errors and follow the conventions for OPL or IR election formatting.
Postconditions	The ballot shuffling process has been tested, and results are recorded..
Main Course	1. When prompted by the system, the user enters the command “election-counter test” with the appropriate flags to configure the system to enable ballot shuffling for testing purposes. 2. User prepares a set of sample ballots with known order and enters the filenames when prompted. 3. The system shuffles the ballots randomly while preserving their integrity and anonymity. 4. User verifies the shuffled ballot order against the expected randomization and then records the results of the test, including any discrepancies. 5. In case any discrepancies are found, the user reports the issues to the development team. 6. The user documents the test results, including successful shuffling or any anomalies. 7. The testing environment is reset for further testing or development.

Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 1. The program stops running and the gathered file data is discarded. 2. No display is shown to the user

Name	Test IR Algorithm for Accurate Results Processing
ID	UC_008
Description	As a Tester, Developer and Election Official, I want to test if the software's Instant Runoff (IR) algorithm accurately processes election results according to the specified rules and logic. This testing ensures that the IR algorithm functions correctly and produces accurate results.
Actors	Tester, Developer, Election Official
Organizational Benefits	Ensures the reliability and correctness of the IR algorithm, reducing the risk of errors in election result calculations.
Frequency of Use	This use case is performed during the testing phase of software development and may be repeated as needed to validate the IR algorithm's accuracy.
Triggers	User types in the appropriate testing command with the flag to test IR elections into the CLI.
Preconditions	1. The software's IR algorithm has been implemented and configured. 2. Test data, including sample election files with known outcomes, is available for testing. 3. Test data files are free from errors and follow the formatting guidelines for IR election. 4. The voting system is set up for testing. 5. Users type in the correct command to access the testing environment for IR election
Postconditions	The testing process for the IR algorithm is completed, and the accuracy of election result processing is verified. Test results and any identified issues are documented.
Main Course	1. User prepares the necessary test data, including sample election files and expected results, to evaluate the IR algorithm. 2. User types in the command "election-counter test" with the

	appropriate flags to access IR algorithm testing. 3. The user inputs testing files when prompted by the system. 4. The IR algorithm processes the election data according to the rules and logic defined for Instant Runoff voting. 5. User compares the actual election results generated by the software with the expected results based on the test data. 6. If the actual and expected results match for all test cases, the tester concludes that the IR algorithm is processing results accurately. 7. If discrepancies or inaccuracies are identified, the user reports these issues, providing details and evidence of the problems encountered.
Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 1. The program stops running and the gathered file data is discarded. 2. No display is shown to the user.

Name	Test OPL Algorithm for Accurate Results Processing
ID	UC_009
Description	As a Tester, Developer and Election Official, I want to test if the software's Instant Runoff (OPL) algorithm accurately processes election results according to the specified rules and logic. This testing ensures that the IR algorithm functions correctly and produces accurate results.
Actors	Tester, Developer, Election Official
Organizational Benefits	Ensures the reliability and correctness of the OPL algorithm, reducing the risk of errors in election result calculations.
Frequency of Use	This use case is performed during the testing phase of software development and may be repeated as needed to validate the OPL algorithm's accuracy.
Triggers	User types in the appropriate testing command with the flag to test OPL elections into the CLI.
Preconditions	1. The software's OPL algorithm has been implemented and configured.

	2. Test data, including sample election files with known outcomes, is available for testing. 3. Test data files are free from errors and follow the formatting guidelines for OPL election. 4. The voting system is set up for testing. 5. Users type in the correct command to access the testing environment for OPL election
Postconditions	The testing process for the OPL algorithm is completed, and the accuracy of election result processing is verified. Test results and any identified issues are documented.
Main Course	1. User prepares the necessary test data, including sample election files and expected results, to evaluate the OPL algorithm. 2. User types in the command “election-counter test” with the appropriate flags to access OPL algorithm testing. 3. The user inputs testing files when prompted by the system. 4. The OPL algorithm processes the election data according to the rules and logic defined for Open Party List voting and Largest Remainder approach for seat allocation. 5. User compares the actual election results generated by the software with the expected results based on the test data. 6. If the actual and expected results match for all test cases, the tester concludes that the OPL algorithm is processing results accurately. 7. If discrepancies or inaccuracies are identified, the user reports these issues, providing details and evidence of the problems encountered.
Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 1. The program stops running and the gathered file data is discarded. 2. No display is shown to the user

Name	Resolve Ties by Flipping a Coin
ID	UC_010
Description	As an Election Official, I want the ability to break ties in election results by flipping a coin to ensure a fair and impartial resolution.

Actors	Election Official, Developer
Organizational Benefits	Ensures a transparent and unbiased method for resolving ties in election results.
Frequency of Use	This use case is performed as needed when a tie occurs in the election.
Triggers	The election results indicate a tie between two candidates or options.
Preconditions	1. An election file is successfully processed by the system 2. The data indicates that there is a tie between two candidates for IR or OPL.
Postconditions	The tie is successfully resolved, and the election results are updated to reflect the outcome of the coin flip.
Main Course	1. The Election Official initiates the calculation of election results. 2. The software calculates the election results. 3. During the calculation, the software identifies a tie where two or more candidates have an equal number of votes. 4. The software automatically initiates a fair and unbiased coin flip to resolve the tie. 5. The result of the coin flip is recorded by the software. 6. The software updates the audit file with information about the tie and the coin flip result. 7. The software updates the election results to reflect the outcome of the coin flip, declaring the winner among the tied candidates. 8. The updated election results are sent out to be printed on the screen for public viewing.
Alternate Courses	None
Exceptions	EX1: User stops program from the CLI 3. The program stops running and the gathered file data is discarded. 4. No display is shown to the user

Name	Verify Election Results Accuracy using a Read-Only Audit file
ID	UC_011
Description	As an Election Official, I want to be able to recreate and validate the process used by the system to determine a winner for the election using an audit file.
Actors	Election Official
Organizational Benefits	Maintains the integrity and authenticity of the election results and audit records.
Frequency of Use	When an election is called and the system has processed election results.
Triggers	The system lets the user know the winner of the election and the location of the audit file.
Preconditions	1. The user inputs an appropriate election file to be processed and UC_001, UC_002, UC_003 and one of UC_004 or UC_005 have been executed appropriately. 2. The system was successfully able to generate an audit file in UC_014. 3. The vote counting system is not interrupted from the CLI and an audit file was generated.
Postconditions	The audit file is read-only and comprehensively describes all actions taken by the system to determine the election winner.
Main Course	1. The algorithm successfully parses the election file data and returns the election winner, other election data and the location of the audit file as mentioned in UC_013. 2. The election officials open the audit file and check to ensure that it cannot be modified externally.
Alternate Courses	None
Exceptions	None

Name	Removal of Candidate for IR
ID	UC_012
Description	As an Election Official, I want the audit file to show the removal of a candidate during the Instant Runoff (IR) voting process for transparency and record-keeping.
Actors	Election Official, Developer, Tester
Organizational Benefits	Provides transparency in the IR voting process, allowing for verification and accountability.
Frequency of Use	This use case is performed during IR elections.
Triggers	Removal of a candidate occurs during the IR voting process.
Preconditions	1. An IR election is in progress. 2. The voting system is configured to create an audit file. 3. The system has identified the need to remove a candidate. 4. The vote counting system ran uninterrupted.
Postconditions	The audit file contains a record of the candidate removal and associated details.
Main Course	1. During the IR process, the voting system identifies the need to remove a candidate due to having the fewest votes. 2. The system records the candidate's name and the reason for removal in the audit file. 3. The audit file includes a timestamp to indicate when the removal occurred. 4. The audit file may also include the number of votes received by the removed candidate. 5. The election process continues with the candidate removed from further consideration. 6. The audit file is periodically updated to reflect the progress of the IR voting process, including additional removals if needed.
Alternate Courses	None
Exceptions	None

Name	View Election Results and Winners
ID	UC_013
Description	As a user, I want to be able to view winners and other election information on my screen after the system has processed the results.
Actors	Election Official, Tester, Developer
Organizational Benefits	Enhances user experience and transparency by providing election outcome information.
Frequency of Use	This use case is performed every time an election is called.
Triggers	The user initiates the process to calculate election results by inputting a ballot data file.
Preconditions	1. The user inputs an appropriate election file to be processed and UC_001, UC_002, UC_003 and one of UC_004 or UC_005 have been executed appropriately. 2. The vote counting system is not interrupted from the CLI.
Postconditions	The user can view winners and election results on their screen.
Main Course	1. User launches the vote counting system. 2. User inputs a ballot data file that follows formatting requirements and UC_001, UC_002, UC_003 are executed. 3. System calculates the election results based on the appropriately identified voting algorithm. 4. System displays election winners and relevant information on the user's screen, including the type of election, number of seats, and candidate details in an easy to understand format. 5. System displays the name and location of the generated audit file.
Alternate Courses	None
Exceptions	None

Name	Generate Audit File for Election

ID	UC_014
Description	As a developer, I want to have an audit file that consists of all the details of the election as soon as a file is inputted into the system.
Actors	Developer
Organizational Benefits	Provides transparency in the election process and records all relevant details for auditing.
Frequency of Use	This use case is executed every time a file is opened by the system.
Triggers	The user initiates the process to calculate election results by inputting a ballot data file.
Preconditions	1. A valid ballot data file is entered into the system for election results calculation as mentioned in UC_001. 2. The file is opened successfully in UC_001.
Postconditions	An audit file is generated and contains comprehensive details of the election.
Main Course	1. User launches the vote counting system. 2. User inputs a ballot data CSV file. 3. System reads the contents of the input file. 4. System generates a blank audit file and saves it . 5. System writes the information from the input file to the audit file. 6. System calculates the election results. 7. System updates the audit file with comprehensive details of the election results and calculation process.
Alternate Courses	None
Exceptions	None