| PBI #4 |
| --- |
| As an election official,<br>I would like to bring in ballots from multiple balloting locations to run the election<br>so that all ballots for an election are processed together from different polling locations. |
| Acceptance Criteria:<br>   1. CompuVote is able to accept multiple files as input.<br>   2. CompuVote is able to parse all the files as one election, only parsing one file's pre-ballot information (i.e. election type, number of candidates, candidates, and number of seats if applicable) |
| Definition of Done:<br>   1. All acceptance criteria are met.<br>   2. All acceptance tests that can be automated are automated.<br>   3. All acceptance tests are met.<br>   4. Test logs have been created for all related tests.<br>   5. Documentation has been created for all related components.<br>   6. The feature is able to be demoed.<br>   7. The changes have been accepted.<br>   8. Duplicate code has been removed through refactoring<br>   9. Messy and poorly-designed code has been cleaned up through refactoring as per our Java style guide |
| Effort: Extra Large |
| PBI Author(s): Aaron Kandikatla, Nikunj Chawla |

- Task 4.1: Converts each file in the command-line arguments to input streams
  - Validates that the files exist in the provided paths in the command-line arguments
- Task 4.2: Passes the input streams to the parse function in some sort of collection (likely an array)
- Task 4.3: The parse function takes in a collection (likely an array) of input streams and the first input stream for the election type, number of candidates, candidates, number of seats (if applicable)
- Task 4.4: The parse function parses all input streams for the number of ballots and the ballots themselves
- Task 4.5: Add a system test that uses multiple files and its respective log file
- Task 4.6: Modify the tests for parse to fit the new signature for parse and modify their respective log files as necessary
- Task 4.7: Create new tests for parse that use multiple files and their respective log files

| PBI #6 |
| --- |
| As an election official, |

| I want the IR election system to invalidate ballots that do not rank at least half of the candidates so that the requirement by state election officials is met. |
|---|
| Acceptance Criteria:<br>    1.  CompuVote removes all ballots that do not rank at least half of the candidates in an instant-runoff voting system from consideration. |
| Definition of Done:<br>    1.  All acceptance criteria are met.<br>    2.  All acceptance tests that can be automated are automated.<br>    3.  All acceptance tests are met.<br>    4.  Test logs have been created for all related tests.<br>    5.  Documentation has been created for all related components.<br>    6.  The feature is able to be demoed.<br>    7.  The changes have been accepted.<br>    8.  Duplicate code has been removed through refactoring<br>    9.  Messy and poorly-designed code has been cleaned up through refactoring as per our Java style guide |
| Effort: Medium |
| PBI Author(s): Aaron Kandikatla |

- Task 6.1: The addBallot function for IR does not add a ballot that ranks less than half of the candidates
- Task 6.2: Add a system test, if it does not already exist, that has ballots that rank less than half of the candidates and its respective log file
- Task 6.3: Modify the IR tests to accommodate the change in the addBallot function and their respective log files
- Task 6.4: Create new tests for addBallot and the runElection functions that use multiple files and create their respective log files

Kanban Board:

| Not Started | In Progress | Completed |
|---|---|---|
| ● Task 4.1<br>● Task 4.2<br>● Task 4.3<br>● Task 4.4<br>● Task 4.5<br>● Task 4.6<br>● Task 4.7<br>● Task 6.1<br>● Task 6.2<br>● Task 6.3 | | |

| | | |
|---|---|---|
| ● Task 6.4 | | |