Rank	Product Backlog Item Description	Acceptance Criteria	Effort	Not Started	In Progress	Finished
1	As the product owner, I would need the OPL and IR algorithms to work fully as desribed in project 1 so that correct results are being generated.	1. Audit and Summary have no conflicts 2. Results of both elections are displayed to the screen and are correct. 3. Process of election is written to an audit file and is correct.	Small			1. Check tests (Shi Chen) 2. See response of graders (Shi Chen)
2	As an election official, I need all ballots in an IR election that do not have at least half the candidates ranked to be invalidated and written to file named invalidated_dateofelection_xxx in order to audit the election and because ballots need to have at least half the candidates be ranked to be valid and impact the election.	1. File is visually neat 2. File invalidated_dateofelection_xxx is written to the directory. 3. File is readable and shows the ID of the ballot	Medium	1. Code for checking validation of ballots 2. Code for writing the file 3. Code for testing 4. Verify correctness of election		
3	As an election official, I would like to see a table of the election showing each round of the IRV and the number of votes that the candidate added/subtracted for that round. I would like the table should be displayed to the screen so that I can inspect the correctness of the runoffs and understand it easily.	1. Table displays each round and each candidate. 2. Table is formatted properly and displayed to the official after the election is done. 3. The number of votes distributed for each candidate after each round.	Large		5. Code for testing	1. Verify current IRV is correct (Qing Hong/Song Liu, paired) 2. Code for tracking the addition and subtraction (Qing Hong/Song Liu, paired) 3. Code for laying out information in the system (Qing Hong/Song Liu, paired) 4. Code for displaying the table (Qing Hong/Song Liu, paired)

Rank	Product Backlog Item Description	Acceptance Criteria	Effort	Not Started	In Progress	Finished
4	I, as the user, would like two ways to run the election file. One would to use a command line argument and the other, a prompt for the name of the file. This is the only input that I would like to be prompted for and only if I do not run a command line argument.	1. Command line argument is accepted and user is not prompted 2. User can be prompted. 3. Only one of the options is working at one time.	Small			1. Code for main function in VoteCounter to read Command Line Arguments (Shi Chen) 2. Code for passing in file name (Dong Liang) 3. Unit test for main function VoteCounter to read Command Line Arguments (Shi Chen)
5	I as the user, would like the filename to be the only information requested from me because inputting other information is tedious and prone to error.	1. Filename can be read. 2. Prompt only asks user for filename. 3. Election can be done by reading information from the file itself.	Small			1. Code for requesting file name (Shi Chen) 2. Write test code (Shi Chen)
6	I, as the user, would like to be prompted for the filename via a GUI instead of a text prompt and I would like a window to appear and be able to type in a file name or look for a file on disk so that I am not working in the terminal as the official.	1. Text terminal prompt does appear. 2. GUI is shown to the user. 3. The correct file is read and the correct results are generated.	Large	2. Code for prompting file name 3. Code for accepting and reading in file name from the GUI 4. Code to allow file searching on disk 5. Code for system testing and unit testing	1. Code for GUI window formatting (Dong Liang)	