

PBI	Task	Hours	Who	Status
<p>As an election official, I want ties to be handled fairly so the decision is unbiased when running an OPL election. AC: Ties in OPL are decided randomly. Done: Tested, documentation done, source code updated to reflect this change, reviewed and refactored. Effort: Small (30 minutes)</p>	Code OPL ties	1	Bek	Not Started
	Design OPL tie test cases	1	Bek	Not Started
	Execute OPL tie test cases	0.5	Bek	Not Started
	Refactor OPL (if necessary)			Not Started
<p>As an election official, I want the program to run smoothly because I don't know how to handle program crashes. AC: On any input the program exits gracefully, notifying the user of any error in layman's terms. Done: Tested, documents, and source code updated to reflect this change, reviewed and refactored. Effort: Small</p>	Fix divide by zero bug in OPL	1	Logan	Not Started
	Execute no ballot test case	0.25	Logan	Not Started
<p>As an election official, I want to choose the winner(s) based upon who has received the most votes because the Secretary of State has decided this is a fair way to choose winners. AC: The winner(s) is/are chosen based upon who received the most votes in one round. The program can be run on CSE lab machines. Done: Implemented, tested, documented, reviewed and refactored. Effort: M</p>	Design MPO voting system (UML etc)	1	Perrie	Not Started
	Implement MPOVotingSystem class	2	Perrie	Not Started
	Design unit test cases for MPOVotingSystem class	1	Matthew	Not Started
	Execute unit test cases for MPOVotingSystem class	0.5	Matthew	Not Started
	Document MPOVotingSystem class	0.5	Matthew	Not Started
<p>As an election official, I need ties between candidates to be broken by a coin toss so that the election is fair and accurate. AC: The coin toss leads to each candidate winning the given seat an even amount of time. Done: Implemented, tested, documented, reviewed and refactored Effort: S</p>	Implement MPO tie breaking functionality	1	Logan	Not Started
	Design unit test cases for tie breaking functionality	1	Logan	Not Started
	Execute unit test cases for tie breaking functionality	0.5	Logan	Not Started
	Document tie breaking functionality	0.5	Logan	Not Started
<p>As an election official, I need the program to be able to handle files of the MPO type so that the system can process the results automatically. AC: File is read in and processes all election information and assigns ballots to candidates. Done: Implemented, tested, documented, reviewed and refactored. Effort: S</p>	Update Election class to handle MPO	1	Perrie	Not Started
	Update HeaderProcessor class to handle MPO	1	Matthew	Not Started
	Update FileProcessor to work with MPO voting system	1	Matthew	Not Started
	Manually test MPO file parsing (test files)	1	Bek	Not Started
	Update Election class documentation	0.5	Perrie	Not Started
	Update HeaderProcessor class documentation	0.5	Matthew	Not Started
	Update FileProcessor class documentation	0.5	Matthew	Not Started
	Design unit tests for Election class updates	1	Perrie	Not Started
	Execute unit tests for Election class updates	0.5		Not Started
	Design unit tests for HeaderProcessor class updates	1	Bek	Not Started
	Execute unit tests for HeaderProcessor class updates	0.5	Bek	Not Started
	Design unit tests for FileProcessor class updates	1	Bek	Not Started
	Execute unit tests for FileProcessor class updates	0.5	Bek	Not Started
	System integration testing for MPO voting system file handling	2		Not Started
<p>As an election official, I need the stats from the MPO election to be outputted to the screen once the election is decided. AC: Each candidate has their statistics displayed to the screen upon completion of the election. Can run on CSELabs machines. Done: Implemented, tested, and documented. Effort: Small</p>	Update MPOVotingSystem to output table with stats showing percentages when the election is complete.	2		Not Started
	Unit test MPOVotingSystem output to table.	1		Not Started
	Update MPOVotingSystem documentation to reflect added ability to output stats.	1		Not Started
<p>As an election official, I want ballots with less than half of the candidates ranked in an IRV election to be discounted and noted in a file. AC: Program invalidates ballots with less than half of the candidates ranked and reports which ballots were invalidated to the audit file. Done: Implemented, tested, documented, reviewed and refactored Effort: Medium</p>	Update FileProcessor to disqualify IRV Ballots that do not have >50% of candidates ranked.	1		Not Started
	Update FileProcessor documentation	0.5		Not Started
	Unit test FileProcessor on an IRV election with ballots that do not have >50% of candidates ranked.	1		Not Started