As an election official,
I want to be able to name and save files in a user friendly way. No command line navigation!

Acceptance:

- A file contains valid election audit information.
- Audit file is named by user.
- User chooses in which directory to store the file.
- File is verified to be named correctly and stored in the correct directory.

Done:

- Passes all regression tests
- Passes all test per acceptance criteria

Effort: Medium

As an election official,
I want to be able to print a short report with only the most important information about an election on it.

Acceptance:

- A short report is created by the system.
- A selection is made to print the report; it does not happen automatically.
- All necessary tests are run.

Done:

- Passes all regression tests
- Passes all test per acceptance criteria

Effort: Small

As an election officer,
I want to be able to select a file instead of manually type the whole file path to process the file so that inputting process is not that tedious and time consuming.

Acceptance:

- Upon running, the system shows the file select window and let the user to select a file and load it
- The file doesn't have to be in the same directory with the program
- The system can run without errors
- The system successfully loads the file selected and produces the correct election result

Done:

- Passes all regression tests.
- All tests pass according to acceptance criteria

Effort: Medium

As an election officer,
I want to be able to stop collecting additional ballots and run the program after I finish inputting in ballot information, so that after I pressed the button, I know the program begins to run and the csv file is generated.

Acceptance:

- The system allows the user to use a graphic user interface to input ballots
- The ballots' information the user created are correctly stored and loaded to the program
- The ballots' information is correctly saved as a CSV file
- There is a button on the user interface that lets the user run the program and also produces the completed CSV file

Done:

- Passes all regression tests.
- All tests pass according to acceptance criteria

Effort: Small

As an election official,
I want to be able to run a 500k ballot election in a timely fashion to limit wasted time.

Acceptance:

- Completes upwards of 500k ballot election in timely and accurate fashion.
- Minimal time scaling as ballot number increases.

Done:

- Passes all regression tests.
- All tests pass according to acceptance criteria.

Effort: Small

As an election official,
I want to be able to manually create ballot files from a user gui so that I don't need to spend extra time creating ballot elsewhere and having to open inside voting system.

Acceptance:

- Accurately creates ballot file based on user input.
- Voting system opens ballot file and correctly reads information.
- Input stored to a new csv file of user specified name.

Done:

- Passes all regression tests.
- All tests pass according to acceptance criteria.

Effort: Medium

As a user with limited RAM,
I want the program to free all memory as soon as possible, so that I can run the program faster.

Acceptance:
- Valgrind runs with 0 leaks.

Done:
- Passes all regression tests.
- All tests pass according to acceptance criteria.

Effort: Medium

As a voter,
I would like to vote for a candidate using a GUI.
So that I can vote hassle free.

Acceptance:
- Ballot correctly displays all candidates.
- Voter can vote for any candidate (once).
- Vote is counted toward the correct candidate and party.

Done:
- Passes all regression tests.
- All tests pass according to acceptance criteria.

Effort: Medium

As a voter,
I would like to use a CSV file to obtain the information about seats, parties, candidates, and ballots so that I can quickly get an idea about the election.

Acceptance:
- CSV file is successfully loaded to the program
- After loading, the gui correctly shows seats, parties, candidates and ballots based on the loaded CSV file

Done:
- Passes all regression tests.
- All tests pass according to acceptance criteria.

Effort: Medium