Updated Product Schedule:

(days to complete in parenthesis next to each task)

Class Due Dates (Week of)	Front End Team	Back End Team	Crowdsourcing Team
1/26 ● Submit Software Design Specification (2/1) ● Submit slides for SDS (2/2)	Familiarity with React and Gulp build system. Learn Material-UI for building React components. Aaron: Fill team in on development stack and architect major modules. (2)	Explore Firebase UML Diagram Nick, Todd: Learned about Parse and Firebase to determine the pros and cons of each(4)	Sequence Diagrams Sonja: Sequence Diagram for user voting (2) Ryan: Sequence Diagram for crowdsourcing. Wrote up docs in Latex and made final changes (2)
2/2 ● Submit zero-feature release (2/8)	Team: A React component for each major feature Aaron: Set up major application scaffolding and build infrastructure to support React front end. Continuous testing bot (3) Geoffrey: Implement “Rate Viewpoints” component screen (1). Implement “Your Candidates” screen (1). Various UI touch-ups (3). Work on a slider for voting (later deprecated) (2). Roee: implement the Political Profile page. Design the cards that will hold the information	Write typescript classes Riley: Preliminary implementation of User class (1) Nick: Preliminary implementation of User and Candidate class (1) Todd: Preliminary implementation of Issue, Category classes (1)	Sonja: Built crowdsourcing content submission form (3) Ryan: Designed and implemented the product website to showcase our work to users and developers. Updated docs to reflect the changes that we made up until this time (3)

	displayed on the page and pick the appropriate Material UI elements for the task (3)		
2/9	Implement interaction for most components. Geoffrey: Document components (1). Mock up voting and getting new issues (2-3). Roee: Adding filtering to the political profile page via constants hard coded into the app (4)	Test the backend thoroughly and ensure smooth integration with frontend Riley: Implement User methods, most significantly the method to get a new issue for a user such that candidates are equally represented (3) Nick: Implement User methods, tweaking the method to get a new issue for a method previously written by Riley and writing half of the ranking method. (3) Todd: Implement Candidate, Category, Issue methods (2)	Sonja: Worked on the crowdsourcing approval component, including a 5 point selector that was used in the rate viewpoints component. (2) Ryan: Make crowdsourcing submission write to Firebase. Wrote code to surface data from subcomponents to the parent component. (2)
2/16 ● Submit beta release (2/19)	Implement Political Profile Add candidate avatars Improve Your Candidates Geoff: Sync Rate Viewpoints with	Improve utility and integration of backend classes Riley: Add features to account for approval of issues (2), create utility methods for categories and candidates (getAll and conversion	Sonja: Normalized styling with rest of app, migrated crowdsourcing to work with Categories and Candidates pulled from Firebase instead of hardcoded data. (2) Ryan: Added a check

	backend, allowing displaying of issues and voting (2). Fixes and improvements for UI issues on Rate Viewpoints (2). Work with backend on issues with User class (3-4). Moving from mock data to real data pulled from database in “Your Candidates” screen (2-3). Work with Aaron on getting webdriver tests set up (1). Aaron: Set up webdriver testing framework, integrate with gulp build system. Setup stand alone selenium runtime. Multiple bug fixes with integration of Model classes with React frontend such as category fetching for issue submission. (2) Roee: adding candidate avatars to Political Profile page. Connecting the political profile page to DB category and candidate data (2)	between ids and names) (2) Nick: Updated tests for User methods and worked on utility methods for categories and candidates. (1 each) Todd: Changed return structure of getRankings to make it not depend on array indices for candidate ids (2). Wrote corresponding tests.	for valid URLs in the crowdsourcing form, updated the UI for required fields to be less intrusive by using red underlines rather than text. (2) Aaron: Categories and candidates pull from firebase instead of being hardcoded constants. (1)
2/23 ● Submit feature complete release (2/26)	Option to skip issues when voting Show how similar a candidate is to the user	Improve ranking algorithm, continue supporting frontend team with additional features Riley: Implement	Sonja: Added ability to add multiple sources and candidates to the crowdsourcing form. Made the news source URLs

	Geoffrey: Implement showing candidate information after voting on an issue (5). Worked with backend team to retrieve issues independent of category (2). Implement skipping issues (and bugfixes for that), with collaboration on backend team (6). Reworking backend to retrieve candidates' avatars (3). Begin work on showing additional candidate ranking data in "Your Candidates" (3). Begin attempting to display categories in sorted order in "Your Candidates" (2-3). Roee: providing Candidate avatars to all app pages. Currently attributing quotes on Political Profile page (2) Aaron: Extensions to webdriver testing framework and authentication infrastructure to allow testuser login to access and vote on production data, while cleaning up temporary data. (2)	improved ranking algorithm (1), improved tests (2), fix errors in User method edge cases (2), normalize ranking results to more grokkable values (1), add ability to get next issue from any category (1) Nick: Continued to support the front-end team in integrating the back-end and front-end code. Also worked on streamlining some back-end code that wasn't necessary for use with Firebase. (5) Todd: Added more utilities (getAllCandidatesSorted, getAllCategoriesSorted) (2). Changed the structure of data returned to resolve front-end issues (1).	clickable in RateViewpoints (2) Ryan: Added module to determine if the user has an internet connection in order to show the correct error in the crowdsourcing form. Updated SRS and SDS docs (2).
3/1 ● Submit release	Webdriver Tests	Fix inconsistencies in data representation	Sonja: Conducted a user study. Fleshed

candidate (3/4)	Geoffrey: Finish work on showing additional candidate ranking data in “Your Candidates” (2). Show profile pictures of candidates in “Your Candidates” (2). Finish work on displaying categories in sorted order in “Your Candidates” (3). Complete code review for Roee (1). Various infrastructure changes (2). Roee: Fix bugs in your Political Profile page that arose as a result of switching to a purely DB-pulling for info and getting rid of the hard-coded issues (3) Aaron: Wrote another webdriver test featuring issue voting. (1)	Riley: Add author field to issues to make handling of direct quotes more convenient (1) Nick: Added back-end functionality to skip issues and filtered content for approving issues so that submitters are unable to approve their own submissions. (3)	out the Group Retrospective document. (1) Ryan: Conducted a user study. Wrote the user study doc and analysis. Added features to improve the crowdsourcing experience. (3) Aaron: Conducted 2 user studies (1)
3/8 ● Submit final release (3/8)	Individual Retrospective	Individual Retrospective	Individual Retrospective