15-437 / 15-637 Fall 2017 Final Project

The goal of the final course project is to give you a flexible opportunity to demonstrate the web application software engineering techniques you have learned in this course. By now you should have formed a final project team, proposed your final project idea, and you will receive a team GitHub repository and feedback on your project proposal inside that repository soon. You will continue work on your final project by developing a detailed specification of your project and then creating three progressively complete prototypes of your project, culminating in a final project demonstration to the course staff. You will develop your prototypes (including your final product) using an iterative, agile development process loosely based on Scrum; each iteration in Scrum is called a *sprint*.

Specifically, your project deliverables are:

- A detailed project specification, due Thursday, 12 October at 11:59 p.m. to your team repository.
- Sprint 1 backlog (see below) due Thursday, 26 October at 11:59 p.m. to your team repository.
- Sprint 1 presentation, source code and development history, due in class Monday, 06 November at your assigned presentation time.
- Sprint 2 backlog due Tuesday, 07 November at 11:59 p.m. to your team repository.
- Sprint 2 presentation, source code and development history, due in class Monday, 20 November at your assigned presentation time.
- Sprint 3 backlog due Tuesday, 21 November at 11:59 p.m. to your team repository.
- Final project demonstration, source code, and development history, due sometime Tuesday, 28 November - Tuesday, 05 December at an assigned demonstration day and time.
- Self- and peer-evaluations of your team members' contributions, due at a date TBD after your assigned demonstration day and time.

We first discuss Scrum, and then discuss each of these deliverables in turn. We finally describe the methodology by which we will evaluate your project.

Scrum

For this project we require you to use an agile iterative development process that resembles Scrum, but we require less requirements analysis and fewer process-related artifacts than you would produce during a typical execution of Scrum. Each iteration in Scrum is called a *sprint*. The overall set of features yet-to-be-developed for your product is called the *product backlog*. The set of features yet-to-be-developed for each single sprint is called the *sprint backlog*. A Scrum team typically has a *product owner* who is accountable for coordinating the team's efforts and interpreting and prioritizing the project goals for each sprint.

The goal of an iterative development process is to define firm, precise goals for each short sprint, but allow you to evaluate your progress and adapt your goals as needed at well-defined time points at the conclusion of each sprint. The firmness of the goals during each sprint allows your team to work rapidly toward project goals during the sprint, without your individual team members needing to coordinate every implementation detail with each other during the sprint. The flexibility of project goals between the sprints allows your team to evaluate its progress and (re-)define an appropriate set of goals to be achieved for the overall project and in the next sprint.

For each sprint your team must define a single person to be the product owner. For this project we require you to rotate the role of product owner among your team members for each sprint, until all your team members have been the product owner for at least one sprint. (For teams of four or more members you should immediately define who will serve as your three product owners.)

Detailed project specification, due Thursday, 12 Oct. at 11:59 p.m.

The goal of the specification is to clarify and precisely define the functionality of your final project. In the Scrum terminology, your project specification should describe your product backlog in great detail.

Specifically, your project specification must include:

- A copy of the project proposal you are implementing. This should be a copy of your original approved proposal or another team's approved proposal, highlighting any slight modifications you've made to the proposal as you've developed your project specification. If you are implementing another team's proposal, please also clearly describe the technologies (languages, frameworks, and APIs) you will use.
- A complete list of all functionality (i.e., the actions) of your project, and an English description of each action. This is your product backlog. We strongly suggest that you organize these features into groups/modules based on related functionality. For each action (or group/module), clearly specify which team member(s) are primarily responsible for the success of that action (or group/module).
- A complete draft implementation of the data models used by your application. These should be your Django models, SQL to create your database tables, or a corresponding implementation if you use another framework or data service.
- A complete set of wireframes or (preferably) HTML mock-ups for your application, for all non-trivial views within the application.

From your English description of your actions and your wireframes (or mock-ups), it should be clear how a user can navigate your site and what features are available from each page. A typical specification is 2-3 pages plus whatever code is necessary for your models and wireframes (or mock-ups).

You should turn in your project specification to your team's GitHub repository by Thursday, 12 October at 11:59 p.m. Organize files within your repository however you want to organize the files for your overall project, but place any specification-specific documents (such as your original proposal and product backlog) into the `specification` directory of your team's repository.

If necessary, your team may request an extension for the project specification. We will be much more restrictive, however, granting extensions for the project specification than we were for the course homeworks.

Sprint backlogs due 26 Oct., 07 Nov., and 21 Nov.

As described above, the sprint backlog consists mainly of the set of features yet-to-be-developed for that

sprint. A sprint backlog also usually contains additional planning information about the sprint. For each feature, you should break down the feature into a set of tasks needed to accomplish the feature, estimate the cost of (the number of hours needed to implement) each task, and assign each task to a single member of your team.

Your goal for each sprint should be to develop a complete, cohesive, working application, with a subset of the overall features you plan to develop for your site. At the end of each sprint you should have a high quality product, though simpler than your eventual final implementation. When creating your sprint backlog you should choose a subset of features that will allow you to develop a high quality prototype that is a subset of your final implementation, not a broken prototype with many half-working features that you will need to improve later.

During each sprint you should work fervently toward completing all the features in the sprint backlog. You may update your sprint backlog document during a sprint to track your progress on each task, but you should not add or remove features from your backlog during the sprint. The goal of the sprint backlog is to provide a fixed reference document that allows you to evaluate your progress (or lack thereof) during the sprint, and allows your team members to work independently toward your project goals without coordinating every activity. At the conclusion of the sprint you should evaluate your progress, examine unforeseen problems you have encountered, and plan (producing a new sprint backlog) for the next sprint.

Your sprint backlog document should also name your team's product owner for the upcoming sprint.

If necessary, your team may request an extension for the sprint backlogs. We will be much more restrictive, however, granting extensions for the sprint backlogs than we were for the course homeworks.

Sprint presentations due Monday, 06 Nov. and Monday, 20 Nov.

You will demonstrate your progress to the course staff and other students in the course in class on Monday, 06 November and Monday, 20 November at a pre-assigned day and time. These dates are the conclusion of Sprint 1 and Sprint 2.

At the end of each sprint you should prepare a short (8-10 minutes, including questions) presentation of your progress. In this presentation you should explicitly demonstrate your site, showcasing the features you have completed so far. The prototype you demonstrate should be a complete, high-quality and cohesive implementation of some subset of your overall product features.

Your presentation should also include:

- A very short background of what your project is and who is working on it.
- Your original goals for the just-completed sprint.
- What you have completed during the sprint.
- A discussion of your progress and the problems you encountered.
- A brief description of your goals for the next (upcoming) sprint.

Before you presentation you also must commit to your team repository:

- Current source code for your project.

- All presentation materials (slideshow, etc.).

You may not receive an extension for your sprint presentation unless the Dean of Student Affairs requests an extension on your behalf. Do not request an extension for yourself under any circumstances. Please talk to the course instructor if you feel that your circumstances might warrant the involvement of the Dean of Student Affairs.

Final project demonstration, due date TBD 28 November - 05 December

Final project demonstrations will occur throughout the week of 28 November - 05 December. You will schedule your project demo time with the course staff soon; an announcement will be made on the course site to schedule your project demos.

Your final project demonstration is not a presentation; it is a mere demonstration of your project's features, an evaluation by the course staff of your features, and a code review of your final product. You should not prepare a presentation for your final demo; you will merely guide us as we examine the key features of your site and evaluate your implementation. A typical demonstration and code review requires 35-50 minutes, depending on the size of your project.

For your project demo, you must deploy your project on a server accessible by the course staff from their own laptops. Your deployment may be local to your own computer using Apache or Nginx or another server, but you may not use Django's development server. You also may use cloud services such as Google App Engine, AWS, Heroku, or DigitalOcean. In any case you must ensure that your site is accessible to remote clients (i.e., not just your laptop or server) before the project demo so that course staff do not encounter problems accessing your site during the demo.

You may not receive an extension for your project demo unless the Dean of Student Affairs requests an extension on your behalf. Do not request an extension for yourself under any circumstances. Please talk to the course instructor if you feel that your circumstances might warrant the involvement of the Dean of Student Affairs.

Evaluation

Even though this is a team project, you will receive individual grades for the project. Historically, for most teams all members of the team receive the same project grade unless there are clear differences in the team members' contributions to the project.

Your final project grade will be based both on your final product and also on the artifacts you produce during your development process, including your proposal, specification, and non-final sprints. Approximately 50% of your project grade will be based on your final product, and approximately 50% of your project grade will be based on non-final artifacts. We reserve the right to adjust this calculation as necessary to ensure fairness in grade assignments among team members and among project teams.

To help our evaluation, all team members must use Git as a version control system and actively contribute to your team's GitHub repository. Each commit message should describe your contribution to the project as well as (if necessary) the context for the work you just committed. If your team uses techniques such as pair programming, your commit messages must explicitly describe the active contributors and you must rotate the active member of the pair so that all of your team members contribute fairly (and visibly) to the project.

Your project grade will be based on a broad range of criteria, including at least:

- Functional correctness and functional design, including correct validation of all user input, positively demonstrating an understanding of concurrency, management of web application state, usability, and avoiding basic security flaws such as XSS, CSRF, and SQL injection attacks.
- Your development process, including your use of the Git version control system, substantial active contributions from all team members, evidence of quality control and unit testing, evidence of an internal collaborative process with substantial internal deadlines, explicit planning for integration, and evidence of reflection on the development process.
- Use and positive demonstration of understanding web framework tools, including models and ORM interactions, forms and model-based forms, templates and the use of view-based template languages, sessions, and possibly cookies.
- Software design principles, including cohesion, modularity, information hiding, separation of concerns, and demonstrating an understanding of MVC frameworks.
- Overall complexity of your project, including the overall size and sophistication of your project idea, diversity of implementation tools and technologies, diversity of external libraries and data sources, and deployment-related issues.