CS 471: Software Engineering
Spring 2018
Project Assignment (Part 2 of 3) – Sprint 1

Due date: Wednesday, April 4, 2018 (at the end of day)

1. Brief Description

This second part of the group project asks you to **plan** for and **complete** your first sprint. The last part of the group project will require working on the second sprint (see timeline below).

Part 1 of 3	**Part 2 of 3**	**Part 3 of 3**
Product Backlog	Sprint 1	Sprint 2

For the first sprint, the grading rubric emphasizes scrum artifacts, especially the **Sprint Backlog**. Future assignments will emphasize other artifacts and engineering activities. While this semester we aim at two sprints, in CS 481 we aim at completing four sprints.

All resources referenced in this file are found on Piazza under "Projects".

Below are the steps required to complete this project assignment.

2. Read about GitHub and ZenHub conventions

Get familiar with the CS471_GitHubAndZenHubConventions_QuickStartGuide (henceforth referred to as GitHubConventions) document containing the conventions you will need to follow for the group project, until the end of CS481. You should already have access to a private GitHub repository and you should have already installed the ZenHub browser extension (Firefox or Chrome) on your machine.

3. Read about metrics

Read the `CS471_S18_Sprint1_Metrics.xlsx` spreadsheet (found on Piazza) to learn what you need to track in Sprint 1 **before** you begin.

4. Update Scrum Roles

You can keep the same scrum roles, or change the roles in this sprint. The information should be updated in the README.md file (see GitHubConventions).

5. Sponsor meetings

Meet with your sponsor (if you have not done so already) and review the **Product Backlog**.

- Review/revise as necessary the description and status of each **User Story**.
  - Use your project's **Definition of Done** (DoD) to resolve questions about whether a story is complete. If your DoD is not working, raise this issue in your Sprint Retrospective Meeting and change it between sprints.
  - If a story is complete (per DoD) but the sponsor wants something different, create a new story (and Acceptance Criteria) to capture the requested change. Customer feedback (i.e., validation) is an important part of scrum and you should expect to modify your Product Backlog between sprints.
  - Note that you may discover missing stories, or decide that an existing story is no longer required. For example, you can assign the stories the label `abandoned`.
- Review/revise as necessary the **Acceptance Criteria** for each User Story
- Review/revise (use the *Scrum Planning Poker* process) the **estimates** for the User Stories

- Review and revise as necessary the **priority** of each story.
  - Remember: Your sponsor gets to choose the stories and their priorities; The Scrum team is responsible for their estimates.

The meeting with the sponsors should be documented in the GitHub Wiki page (see GitHubConventions).

6. Generate Sprint Backlog

After choosing with the help of the sponsor the stories to be implemented in the first sprint, create 3-10 tasks for each story.

- The tasks should reference in their description the original story (e.g., #123, where 123 is the story number/ID). Establishing traceability between stories and tasks (and vice-versa) is paramount for working in a large group project.
- The tasks should be written in engineering language
- A team member should volunteer for a task by assigning the task to them and providing an estimate
  - NB: the developer who signs up for the task gets to estimate it
- "Research" tasks should be labelled accordingly (see GitHubConventions).
- It is normal at the beginning of the sprint to identify only ~70% of the tasks for the sprint, and as the sprint progresses the team adds more tasks (linked to stories).

7. Working on the Sprint Backlog

To indicate the fact that the task is being currently worked on, drag the task to the `In Progress` pipeline (see GitHubConventions).

Every commit and pull request should be linked to a task (see GitHubConventions). This process ensures a full traceability between requirements (e.g., user stories) and source code (i.e., user stories are referenced by tasks, which are referenced by commit log messages). The full traceability will be essential once the project progresses and the code base increases.

8. Submission

One team member should submit via Blackboard a zip archive titled `<TeamName>_Sprint1.zip` containing the following file:

- `Sprint1Metrics.xlsx`: Use the template file `CS471_S18_Sprint1_Metrics.xlsx` to report your progress.
  - The metrics file requires you to provide a link to a video demonstrating the functionality of every story completed. In other words, it contains the feature "demo" that the sponsor will see at the end of the sprint.

All the remaining artifacts generated by the team (e.g., README.md, DoD, user stories, tasks, commits, etc., mentioned in the GitHubConventions) in Sprint 1 will be considered as the team "submission". And remember:

If it's not on GitHub, it does not exist, and you will not get credit for it.

9. Grading Rubric

The maximum points for this first sprint is 100, and the points are distributed as follows:

Product Backlog (10 points)	**Points**
Sufficient number of detailed, prioritized and estimated user stories (in role-goal-benefit, and written in customer's business language) having adequate	10

acceptance criteria in the Given-When-Then template

Sprint Backlog - Stories (25 points)	**Points**
User stories (in role-goal-benefit) proposed for sprint	10
Proposed Stories Prioritized and Estimated	5
Adequate Acceptance Criteria (in given-when-then template) for user stories	10

Sprint Backlog - Tasks (35 points)	**Points**
Tasks defined and linked to each User Story	20
Tasks have engineers and estimates	5
Evenly distributed workload among team members	10

Other artifacts (30 points)	**Points**
Definition of Done (in the repository Wiki)	5
Complete traceability between stories, tasks, commits and pull-requests (i.e., using #ID)	10
"Constant" work speed throughout the sprint (illustrated in the burndown chart)	5
Metrics and demo videos of your sprint 1 user stories passing the AC (see metrics file)	10

Team Penalties:

- Team did not meet with the sponsor in the first week of the sprint for the Sprint Planning meeting (-50% of grade)
- Team did not meet with the sponsor in the last week of the sprint for the Sprint Review meeting (-50% of grade)
- Team did not use the GitHubConventions or team did not fix all the inconsistencies indicated in the Scrum Linter report (up to -100% of the grade)

Individual Penalties:

- Team member was
  - "absent" from the sprint,
  - did not participate,
  - did not contribute or
  - did not get involved (up to -100% of grade)