Phase 2 - Implementation

Due Date: Thursday March 5th, 2015 at 10:00 pm

The main goal of this phase is to implement and present the first working version of your product.

It is up to you to choose what to implement, and how much work to put into this phase. We expect you to try to get as much work done as possible, while being realistic about your busy schedule.

Important: To ensure that you're not planning to do too much or too little, you must get your release plan approved by your TA **by Thursday, Feb 12, at 10 pm**. (The earlier the better, you can do this in the tutorial time or over email).

Another goal of this phase is to get a better understanding of the Scrum model/framework. You will follow a *scrum-like process*, and will be asked to document and evaluate the process.

Team Deliverables

- Your team repo is your main deliverable - It contains your code, .md files, commit history, and issues.

- Live Demo
- `Phase2/Product.md`
- `Phase2/Process.md`

Live Demo

- To be presented to your TA **during tutorial, on March 9**.
- Short - Max 10 minutes (including time for Q&A).
- Presented by 2 team members
  - One doing most of the talking ("the navigator"), and the other runs things on the computer ("the driver").
  - The TA will choose the 2 students on the spot, so you should all be ready to present.
  - The two students get to decide who will be the navigator and who will be the driver.

Phase2/Product.md

- A report describing your **product** (i.e. The *what*, not the *how*).
- Please keep it short (2 min' read, seriously).
- Be informative, but concise.
- You can use your submissions from phase-1 as a starting point.
- Focus on the features that you built during this phase, not on future plans.

Phase2/Process.md

A report describing your scrum-like process. The report should include the following items:

- A summary of your planning and review meetings for all your sprints (4 sprints in total).
  - Includes date and location.
  - Please keep it short, and give us the highlights.

    - Include important decisions you've made, and arguments for/against them.
    - Tip: You can use GitHub to label all the user stories you plan to complete during a sprint, and include a link to an issue search result in your planning meeting summary.
    - For the review/retrospective meeting, we would like to see some numbers and/or interesting statistics.
        - For example: Percentage of planned work that was actually completed, distribution of completed work among team members, percentage of planned work completed categorized by priority, etc.
        - The point is to evaluate the estimations that you've made during the planning meeting.
- A summary of your "daily" scrum meetings.
    - Please include date and location.
    - If you use a chat, you can simply copy-paste the highlight of the chat.
    - Otherwise, please give us a very quick summary.
    - A good format would be a list, specifying the highlights of each member's update (each highlights should be 1-3 sentences long).
- Description of how you used GitHub issue management system (e.g. naming conventions, labels, team's conventions, etc.)
- Description of any other tools (e.g. Project Management/Bug tracking) you have used.
- Description of any other major decisions/conventions you may have made/used during the process, and why.
    - For example: Did you have a "scrum master"? Why? Do you think it was the right decision?

Individual Deliverables

The following files should be submitted to the root of your **personal repo**:

- `proj-phase2-individual-contrib.md`
- `proj-phase2-code-review.md`

- proj-phase2-short-answers.md

proj-phase2-individual-contrib.md

- Describe your most significant individual contribution. 10%
  - Must be a technical contribution (i.e. “I was in charge of documentation” doesn’t fly). For example: A nasty bug that you resolved, a component that you've implemented elegantly, a complicated problem that you had to solve, etc.
  - A good description should include links to specific commit(s) and/or issue(s) on GitHub.

proj-phase2-code-review.md

We want you to review one of your teammate's code, and provide feedback.

This file should be a summary of your code review.

- It’s the team’s responsibility to make sure that everybody’s code gets reviewed by someone.
- If you can offer improvements to your teammate’s code (in terms of correctness and/or code quality), that would be great.
  - You can fork the team repo.
  - Commit your changes to the fork.
  - Send a pull-request with written feedback on your teammate's code.
  - In proj-phase2-code-review.md, you can include a link to the pull request on GitHub.
- If you cannot improve your teammate's code, you should still be able to:
  - Provide them with positive feedback.
  - Describe parts of the code that were well done, and perhaps even taught you some coding techniques/ideas/tricks.

proj-phase2-short-answers.md

- Answer the following questions (2-3 sentences for each answer)
  - What did you like the most about your Scrum-like process, and why?

    - What did you dislike the most about your Scrum-like process, and why?

Marking Scheme

- Team 75%
    - Live Demo 20%
    - `Phase2/Process.md` 15%
    - `Phase2/Product.md` 15%
    - TA's evaluation of your project 25%
        - Delivered everything that was promised in the approved release plan. 10%
        - Overall product quality (e.g. interesting features, creative solutions to problems, attention to details, stable product, etc.) 9%
        - Code quality. 6%
- Individual 25%
    - `proj-phase2-individual-contrib.md` 10%
    - `proj-phase2-code-review.md` 10%
    - `proj-phase2-short-answers.md` 5%

Our Scrum-like Process

Scrum is used by teams where most/all members work full-time, usually in the same physical space, and the project is ongoing. Our situation is different, so we will adapt scrum to fit our realistic constraints.

Here is a description of the scrum-like process we would like you to follow.

The players

- Development team - All members of the team.
- Scrum Master

    - In a real agile team, the scrum master is not a member of the development team.
    - In our case, the “scrum master” will be a team member who volunteers to facilitate the process (e.g. Scheduling, organize note taking, recording the meetings, and/or anything else you might find useful).
    - It is up to you to decide whether or not to have a "scrum master". If nobody volunteers, or the team decides that it is unnecessary, that's totally fine.
- There is no product owner.
    - The closest thing to a product owner is your TA, who has to approve your release plan.

The artifacts

- Use GitHub issue management system to keep track of all user-stories/tasks.
- It is up to you to decide on how to use the system (i.e. Which labels to use, naming conventions, how to assign/pick-up issues, how to prioritize stories, etc.)

The events

- Sprints
    - Four sprints, each sprint 7 days (with the exception for the first sprint).
    - Different teams might have different scheduling constraints, so we’ll leave it up to you to decide on the exact dates.
    - If your schedule permits, we suggest that you review sprint-1 and plan sprint-2 on separate days (to give you some time to reflect, and reach conclusions).
- Sprint planning meeting
    - At the beginning of each sprint.
    - Try your best to get everybody to meet in person. In extreme cases, when scheduling is too problematic, consider using a video chat.
    - At the end of this meeting you should know which user-stories you plan/commit to implement during this sprint.
- Daily Scrum meeting
    - We ask you to hold 2-3 of these short meetings during each sprint.

    - Try to get as many team members as possible.
    - You can run these meetings in person or via video/voice/text chat.
    - Remember, these are supposed to be short meetings (15 min’ max). The goals are to keep everybody in sync, and to exchange relevant information.
- Sprint Review/Retrospective meeting
    - At the end of each sprint.
    - At the end of this meeting you should know how much of the planned work was actually done, and what are the next main challenges.
    - Tip: You’ll need to document this meeting, so please make sure you have enough time between reviewing your last sprint and the due date.