

INSTRUCTIONS

# CS361: Assignment 2: Environment Setup, Course Project Plan, and Sprint 1 Plan (for Milestone #1)

**Overview**

Now that you’ve been introduced to the microservices concept, start planning your course project (except for the microservice you’ll later make for a teammate) and your Main Program (Milestone #1). It’s OK to change your plan later!

This assignment has three parts:

* **Part 1**: Environment Setup. Initialize your GitHub repository and set up GitHub Projects for organizing your project tasks.
* **Part 2**: Course Project Plan. Write all the user stories you would like to be part of your course project (except for the microservice you’ll later make for a teammate). It’s OK if you don’t implement all of them this term, and you will likely need to write additional stories later in the term.
* **Part 3**: Sprint 1 Plan. Select at least three user stories to implement during Sprint 1 (for your Main Program / Milestone #1). Define detailed requirements for each user story.

**Please refer to the template document to complete the assignment as outlined in these instructions.**

**Part 1: Environment Setup**

Set up your development environment. In addition to an IDE or code editor (choose any you prefer), start a GitHub repository and choose a task management system.

1. **GitHub Repository**

Create a GitHub account if you don’t already have one, create a Git repository hosted on GitHub. This first repository will be for your Main Program. Make two **test commits**. The test commits should show up on GitHub.

1. What is your GitHub username?
2. Provide GitHub screenshot(s) of two test commits. Your GitHub username is in the screenshot(s).

Later, it would make sense for you to set up additional repositories, one for each of the microservices you’ll implement.

**2) Task Management System: GitHub Projects**

For your course project, you will use GitHub Projects to keep track of development tasks. Read the instructions to get started with GitHub Projects: <https://docs.github.com/en/issues/planning-and-tracking-with-projects>

**Try** out GitHub Projects:

1. Create a new project. You can choose a template or start from scratch with a Table, Board, or Roadmap. You will be able to modify your template or project type later if you need or want to.
2. Create a task, then update it, assign it, delete it, etc. Provide a **screenshot** of your task. Name the task **"CS361 Test Task"**.

**Part 2: Course Project Plan**

Write an initial set of user stories for your course project (except for the microservice you’ll later make for a teammate). Put the user stories in a **Product Backlog** column/section/category of Github Projects, or a label/tag the user stories “Product Backlog”. You might end up changing these user stories later or adding new ones.

1. **Product Goal and Backlog**

You’ll be using *some* Scrum methods in this course. Unfortunately, the Scrum Master and Product Owner roles don’t work well in this course setting. You will, however, experience Scrum Events and Artifacts.

1. What is your **Product Goal** for your course project? This includes your Main Program, Microservice A that your teammate will later implement for you, and Microservices B, C, and D. It does NOT include the Microservice A you will make for a teammate.

The Scrum Guide (<https://scrumguides.org/scrum-guide.html>) doesn’t give a detailed description of the Product Goal: “**describes a future state**”, “**long-term objective**”.

Example Product Goal: “Develop a desktop app that listens to what people are saying and automatically shows content that might be relevant to their conversation.”

1. **Write user stories** for your **entire** **course project** (except for the microservice you will make for your teammate). Use INVEST to guide you.

**Assignment requirements for Product Backlog user stories**:

* Each has a **name** that briefly describes the functionality (e.g., “Login”)
* Each uses the **“As a… I want to… so that…” format** (explained in textbook)
* Each is about **functionality** and not about the quality of the functionality or a constraint (user stories are functional requirements, not non-functional requirements)
* Total of at least **10** user stories
* As a set, must have **no obvious violations of INVEST**

Enter the user stories into GitHub Projects in a **Product Backlog column/section/category**, or with a “Product Backlog” label. Include a **screenshot** so that the grader can confirm you added the stories.

**2) Quality Attributes**

Quality attributes can help guide the entire development of your project. They can remind you (and other developers) what aspects of your project matter the most and can help you decide which features to implement and in what way.

**Select the top three quality attributes you care about** for your course project. See <https://en.wikipedia.org/wiki/List_of_system_quality_attributes> **for ideas.**

1. **Which three quality attributes did you choose? Name** and **define** each**.**
2. **Why did you choose these quality attributes?** Explain how each quality attribute is particularly relevant to your project (1+ sentence per quality attribute)

**Part 3: Sprint 1 Plan (for Milestone #1)**

Next, move some user stories from your Product Backlog to a Sprint Backlog column/section/category, or change the label/tag to “Sprint Backlog”. These will be the user stories you WILL implement during Sprint 1 (for Milestone #1 / your Main Program) and comprise your Sprint Plan. **Your Milestone #1 Main Program implementation must offer value to users.**

**Note these minimum requirements for Milestone #1:**

* Your Main Program implementation must offer value to users
* At least three user stories are completed
* All features that are part of the milestone must be working. The milestone must not have partially completed features.
* It allows users to interact (e.g., provide input, push buttons, etc.)
* Reflects each of the Inclusivity Heuristics
* Reflects three quality attributes of your choice (i.e., satisfies the non-functional requirements you write for each quality attribute)

*Hint: If you choose “usability” or “inclusivity” as a quality attribute, your corresponding non-functional requirement can involve the Inclusivity Heuristics.*

1. What is your **Sprint Goal**? The Sprint Goal must clearly communicate what you plan to work on (e.g. what pages, what functionality, etc)
2. Select **at least three** user stories from your Product Backlog and move them to your Sprint Backlog, or re-label/tag them “Sprint Backlog”. Because you will be implementing these user stories during the Sprint, you need to write more specific requirements in the form of **acceptance criteria**.

Acceptance criteria can cover both functional and non-functional requirements. The non-functional requirements can serve to carry through your intention to reflect quality attributes.

Some developers write their user stories on 3” by 5” index cards: The user story name and “As a” format goes on the front of the card and the acceptance criteria can go on the back. **Example**:

|  |
| --- |
| (Front of index card)  **Automatic IMDB** (“As a … I want to … so that …”)  **As a user** speaking during a conversation, **I want to** automatically see the IMDB.com webpage for the movie I’m talking about **so that** I can continue with my conversation and examine the webpage as needed. |
| (Back of index card)  **Acceptance criteria**  Functional requirements (“Given… when… then…”)   * **Given** a person is speaking in English at 60 dB or louder **when** the software is at least 80% sure it knows what movie the person is talking about, **then** it will open and focus the default web browser and navigate to the movie’s IMDB.com webpage.   Quality attributes & Non-functional requirements   * Responsiveness: Once the software is 80% sure about what movie is being spoken about, it will display the movie’s IMDB.com webpage within 3 seconds. |

Use this format to fill out each of your Sprint Backlog user stories.

**Assignment requirements for Sprint Backlog user stories:**

* For each of the three (or more) user stories…
  + The front of the card must contain the user story’s name and “As a… I want to… so that…” format
  + The back of the card must contain at least one functional requirement and each functional requirement must use the “Given… when… then…” format.
* Each of your three quality attributes must appear at least once on a user story’s “back of index card” and must be converted to a non-functional requirement.
* All of the functional and non-functional requirements must be testable.

Later, you will be asked to show that your functional and non-functional requirements are met.

1. Take a **screenshot** that shows you’ve moved these user stories into a Sprint Backlog in GitHub Projects.

Your **Definition of Done** for Sprint would typically include, “The acceptance criteria are satisfied for all Sprint Backlog user stories.” You aren’t required to write your DoD or put it in GitHub Projects.

This would also be **a good time to break each of your user stories into a list of specific tasks** you need to complete. Task management systems are, as you might imagine, a great place to do that!

**Submission**

PDF or Word format via Canvas.

**Grading**

You are responsible for satisfying all criteria listed in the Canvas rubric for this assignment.

**Questions?**

Please ask via Ed so that others can benefit from the answer.