Oregon State University
Ecampus

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 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, investigate task management systems to organizing your project tasks.

- **Part 2**: Course Project Plan. Write all the user stories you would like to be part of your Course Project. (It's OK if you don't implement all of them this term.)

- **Part 3**: Sprint 1 Plan. Select at least three user stories to implement during Sprint 1 (for Milestone #1). Define detailed requirements for each user story.

Note these minimum requirements for Milestone #1:

- At least three user stories completed
- All features that are part of the milestone must be working. The milestone must not have partially completed features.
- Has a way for 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.*

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.

Complete each item below by replacing the highlighted text (**Usability note**: double-click the text to select it).
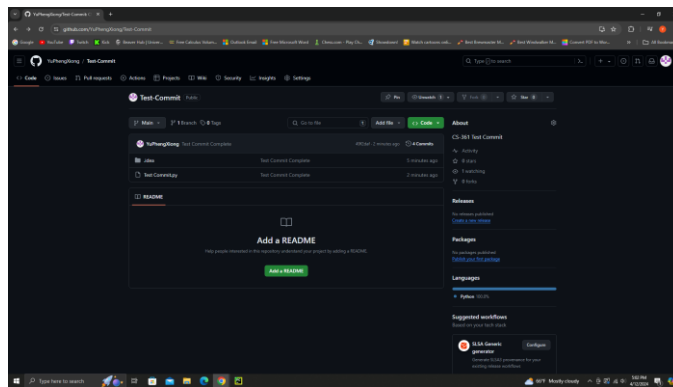
1) **GitHub Repository**

   Create a GitHub account if you don't already have one, create a Git repository hosted on GitHub. Make a **test commit**. The test commit should show up on GitHub.

a) What is your GitHub username?

YuPhengXiong

b) Provide a screenshot of your test commit.

2) **Spike: Task Management Systems**
For your course project, you will be using a task management system to keep track of development tasks. Spike at least **three** task management systems you could use.

A spike is a quick, directed effort focused on getting a question answered. Performing a spike can help you make intelligent decisions. Spikes do take time upfront, but they can also save you from making a bad choice that takes much more time to recover from. This portion of the assignment provides an opportunity to do a spike while making a relatively low-stakes decision (which task management system to use).

Examples of task management systems you could spike: Trello, Jira, Asana.

Requirements for the task management systems:
- Free tier
- Cross-platform (works on Windows, iOS, and Linux)
- Support for collaboration, task definition/deletion/updating, task priorities, task due dates, assigning people to tasks, setting task status, and organizing tasks into different columns.

To do a spike, you need to research the task management systems and also (1) try to **use** them, (2) **evaluate** them based on specific criteria, (3) **compare** them, and (4) **decide** which to use.

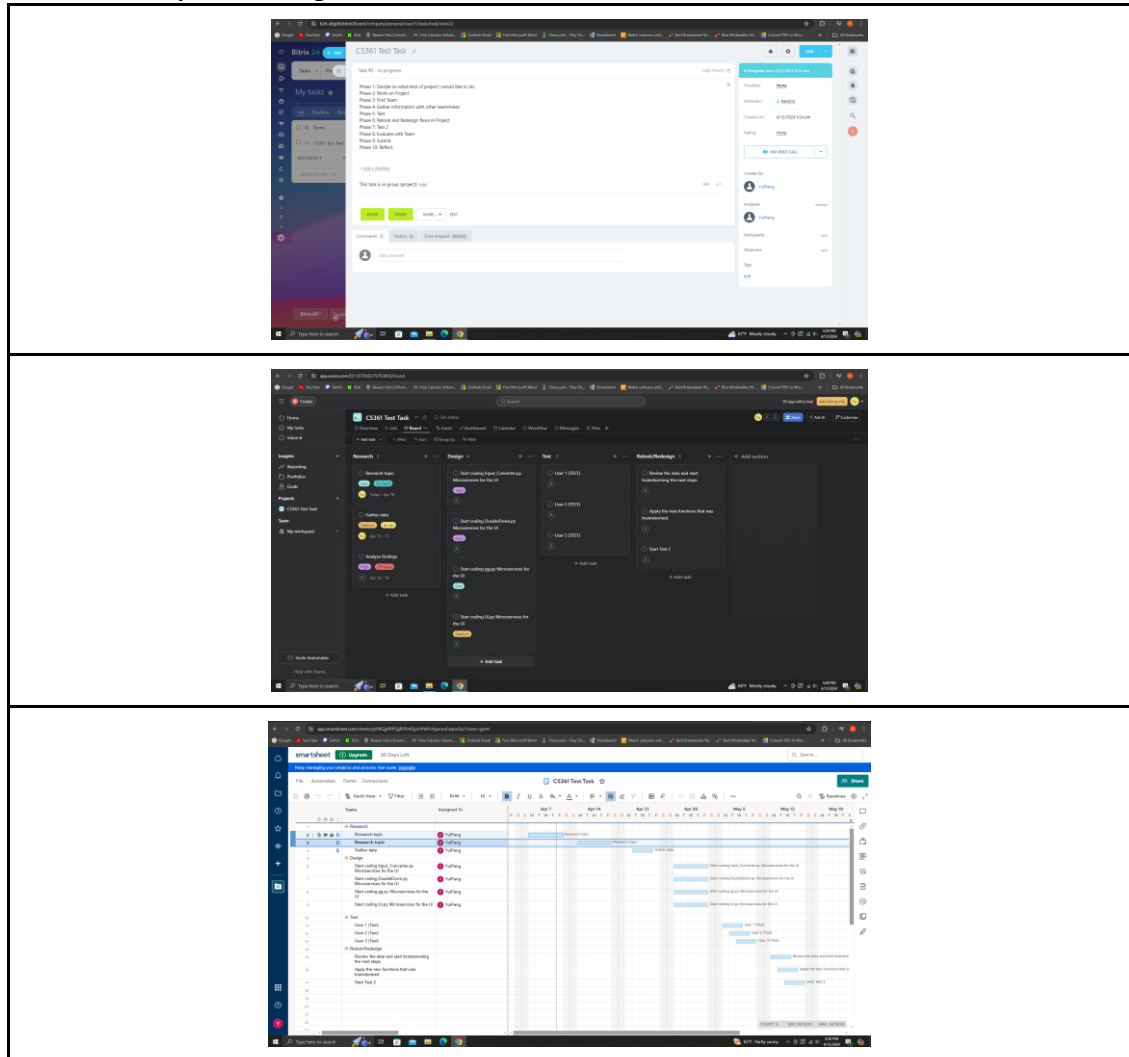a) Which task management systems did you spike?

BitRix 24
Asana
SmartSheet

b) **Try** each system. Create a task then update it, assign it, delete it, etc. **Screenshot** your

task in each system and paste below. Name the tasks **"CS361 Test Task"**.

c) For each, **evaluate** against at least these criteria:

Ease of use. Ex: Is it intuitive to learn? Easy to remember how to use it? Do you find yourself making lots of errors trying to use it? Are there tutorials and documentation?

Name of system 1: BitRix 24

Evaluation of system 1's ease of use (2+ sentences): I would say that at first it was confusing to use and there was no instruction of what to do. Which made it harder for me, because I couldn't understand what I was doing and setting it up.

Name of system 2: Asana

Evaluation of system 2's ease of use (2+ sentences): The ease of use of this software is incredible, it had detailed instructions of what to set up and step by step process. What looks helpful is that it is short and simple in design to get the point out to individuals.

Name of system 3: SmartSheet

Evaluation of system 3's ease of use (2+ sentences): The ease of use of this

software is simple and detailed. It is very clean and clear to showing details to the point and makes information collection process easier.

Speed / responsiveness. Ex: Does it take an annoyingly long time to log in / load / create new projects / etc. or is it peppy?

Name of system 1: BitRix 24
Evaluation of system 1's speed / responsiveness (2+ sentences): It was relatively simple, and it doesn't ask for too much information. It only needed my email to connect.

Name of system 2: Asana
Evaluation of system 2's speed / responsiveness (2+ sentences): The login process was longer than the first system. Asking for more information and more than just an email.

Name of system 3: SmartSheet
Evaluation of system 3's speed / responsiveness (2+ sentences): This

Feature set. Ex: Besides the required features, does the system have other features you are likely to need?

Name of system 1: BitRix 24
Evaluation of system 1 (2+ sentences): It had an email kind of style, where you can individually send information to individuals. Which could be helpful to give the individual more details of what you might want from their part.

Name of system 2: Asana
Evaluation of system 2 (2+ sentences): Has the ability to set goals for what we want from ourselves, where no one else could see. But overall, I don't see much else that would be useful.

Name of system 3: SmartSheet
Evaluation of system 3 (2+ sentences): There is pretty much nothing but a spreadsheet. So, I would say that this wouldn't have a lot of different features.

Relevance / popularity. Ex: Is it likely you will ever see the task manage- ment system again after the course?

Name of system 1: BitRix 24
Evaluation of system 1's relevance popularity (2+ sentences): Most likely not, because there are much better sources, that is more in-depth and have more details and features. The time it takes to set up this and learn what to do, you could have done it faster on the other softwares.

Name of system 2: Asana
Evaluation of system 2's relevance popularity (2+ sentences): I think this is

also a good system that I would see myself using in the feature, just because it is so simple to use and is very straightforward. It makes me feel more comfortable working with multiple tasks, because it is very easy to set and get done.

Name of system 3: SmartSheet
Evaluation of system 2's relevance popularity (2+ sentences): I think I would see more of this kind of style, because it is a spreadsheet, which is plain and simple to make sure everyone knows what is assigned.

d) **Compare** the systems by **ranking** them based on the criteria above. Best to worst for each criterion. **List or table format.**

> **System 1 name:** BitRix 24
> **System 1 ease of use:** Worst
> **System 1 speed/responsiveness:** Best
> **System 1 feature set:** Worst
> **System 1 relevance/popularity:** Worst
>
> **System 2 name:** Asana
> **System 2 ease of use:** Best
> **System 2 speed/responsiveness:** Best
> **System 2 feature set:** Best
> **System 2 relevance/popularity:** Best
>
> **System 3 name:** SmartSheet
> **System 3 ease of use:** Best
> **System 3 speed/responsiveness:** Best
> **System 3 feature set:** Worst
> **System 3 relevance/popularity:** Best

e) Which system is **highest ranked?**

Asana

Decide which task management system you're going to use and use it to complete Parts 2 and 3.

Part 2: Course Project Plan

Write the entire set of user stories for your course project. Put the user stories in a **Product Backlog** column/section/category of your task management system. **You probably won't have to finish implementing the entire Product Backlog this term**.

Complete each item below by replacing the highlighted text (**Usability note**: double-click the text to select it).

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.

a) What is your **Product Goal** for your individual project?

> The product goal for this project is to take a user's input and produce different mathematic equations.

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."

b) In a **Product Backlog** column, section, or category of your task management system create **user stories** for your entire **individual project**. 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 (you will not have to implement all of these)
- As a set, must have **no obvious violations of INVEST**

- **User story 1 name:** User input
- **User story 1 "As a…" format:** As a user, I want to be able to give an input, so that I can see the result of what I give.

- **User story 2 name:** Clarification
- **User story 2 "As a…" format:** As a developer, I want to relook over my code, so that it can become more simplified.

- **User story 3 name:** Bug fixes
- **User story 3 "As a…" format:** As a developer, I want to be able to take test results and fix any issues that are present.

- **User story 4 name:** Usability of software
- **User story 4 "As a…" format:** As a user, I want to be able to use the software when I'm at a computer, so that I can get results.

- **User story 5 name:** Performance of software
- **User story 5 "As a…" format:** As a software, I want to be able to perform at the best possible case, so that I won't run into any issues.

- **User story 6 name:** Reliability of software
- **User story 6 "As a…" format:** As a software, I want to be able to run without any issues, so that I can never have failures.

- **User story 7 name:** Precision of increments
- **User story 7 "As a…" format:** As a user, I want to be able to read each increment of measurement, so that I understand what has been done with my input.

- **User story 8 name:** Design Changes
- **User story 8 "As a…" format:** As a developer, I want to incorporate design changes, so that it can be up to date with technology.

- **User story 9 name:** Reflection of Testers
- **User story 9 "As a…" format:** As a user, I want my results of testing to be recorded, so that it will have an impact when the software is updated.

- **User story 10 name:** Reflection of Development
- **User story 10 "As a…" format:** As a developer, I want to reflect on the test results of my software, so that I can further enhance the software in the future.

Enter the user stories into your task management system in a **Product Backlog column/section/category**. Paste a **screenshot** below 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.

a) **Which three quality attributes did you choose? Name** and **define** each.

- **Quality attribute 1:** Reliability
- **Quality attribute 1 definition:** Is the ability of an object/software function without failure.

- **Quality attribute 2:** Precision
- **Quality attribute 2 definition:** The exact quantity of a certain measurement being expressed.

- **Quality attribute 3:** Simplicity
- **Quality attribute 3 definition:** How simple the design is and how easy it is to understand.

b) **Why did you choose these quality attributes?** Explain how each quality attribute is particularly relevant to your particular project (1+ sentence per quality attribute)

- **Why quality attribute 1 is relevant to your project:** It is relevant, because a reliable project will always work, so if it was reliable, it would not have any issues when being used.
- **Why quality attribute 2 is relevant to your project:** It is relevant, because with precision, dealing with math, units are very important, so being able to translate the unit precisley would allow better understanding.
- **Why quality attribute 3 is relevant to your project:** The simpler it is the easier it is to use, and easier to understand what the project is doing.

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

Next, move some user stories from your Product Backlog to your Sprint Backlog—these will be the user stories you WILL complete during Sprint 1 (for Milestone #1) and comprise your Sprint Plan.

1) What is your **Sprint Goal**?

> To create codes that will take in user input and increment in different measurements. That way we can see other types of measurements used in the world.

2) Next, you will need to select **at least three** user stories from your Product Backlog and move them to your 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 go on the front of the card and the acceptance criteria can go on the back. **Example**:

> (Front of index card)
> **Automatic IMDB**
>
> 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 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" 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 requirement must be testable.

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

First user story

(Front of index card)
Usability of software As a user, I want to be able to use the software when I'm at a computer, so that I can get results.

| (Back of index card) |
|---|
| **Acceptance criteria**

Functional requirements
    • When using the software, you should be able to input a number, and after the software will display different measurements based on the number that is given.

Quality attributes & Non-functional requirements
    • Once the input is given to the software, the software will calculate the other measurements and display it after a couple of seconds. |

Second user story

(Front of index card)
User input
As a user, I want to be able to give input, so that I can see the result of what I give.

(Back of index card)
Acceptance criteria
Functional requirements
• The software will take in the input from a user and display that to the user.
Quality attributes & Non-functional requirements
• Once given the input, the software will start running and incrementing different measurements.

Third user story

(Front of index card)
Clarification
As a developer, I want to incorporate small instructions, so that it can become more simplified and clearer of how to use the software.

(Back of index card)
Acceptance criteria
Functional requirements
• The software will be clear and simple so that the user will be able to understand and test the software.
Quality attributes & Non-functional requirements
• The instructions given will allow clarification of what to do when stuck or confused.

3) Take a **screenshot** that shows you've moved these user stories into a Sprint Backlog in your task management system.
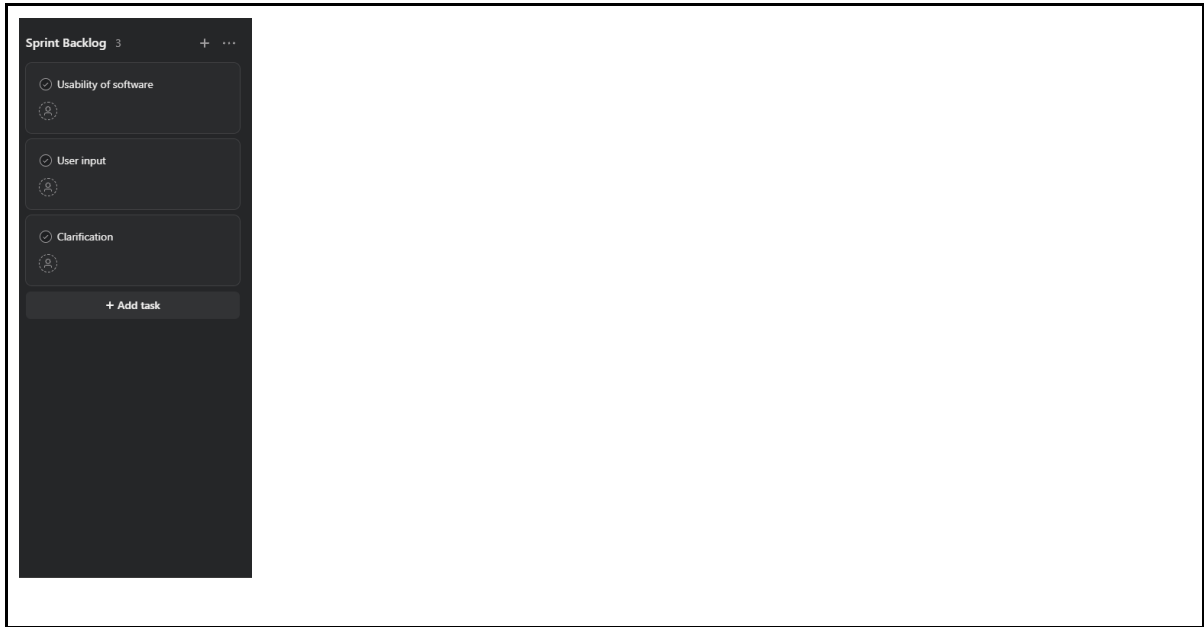
Your **Definition of Done** for the Sprint would typically include, "The acceptance criteria are satisfied for all Sprint Backlog user stories." You aren't required to write your DoD or put it in your task management system.

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.

You must follow instructions at Modules > 'HOW TO: Attach a Document to "Text Entry" Field'.

Grading

You are responsible for satisfying all criteria listed in the Canvas rubric for this assignment. You will be able to revise this assignment if you miss points.

Questions?

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