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 (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 investigate task management systems to organize 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.
- **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.

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.

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).

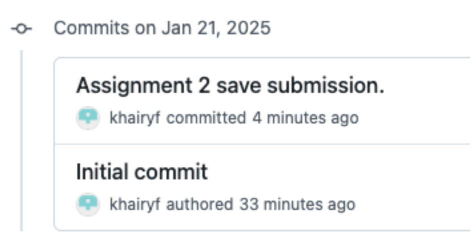
1) GitHub Repository

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

a) What is your GitHub username?

khairyf

b) Provide a screenshot of your test commit.



Commits on Jan 21, 2025

Assignment 2 save submission.
khairyf committed 4 minutes ago

Initial commit
khairyf authored 33 minutes ago

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

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:

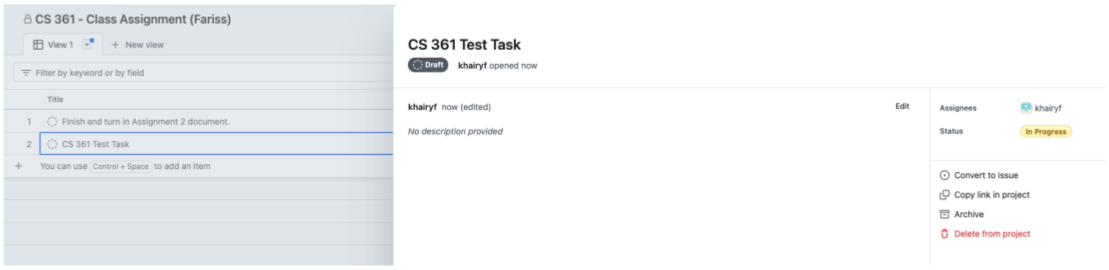
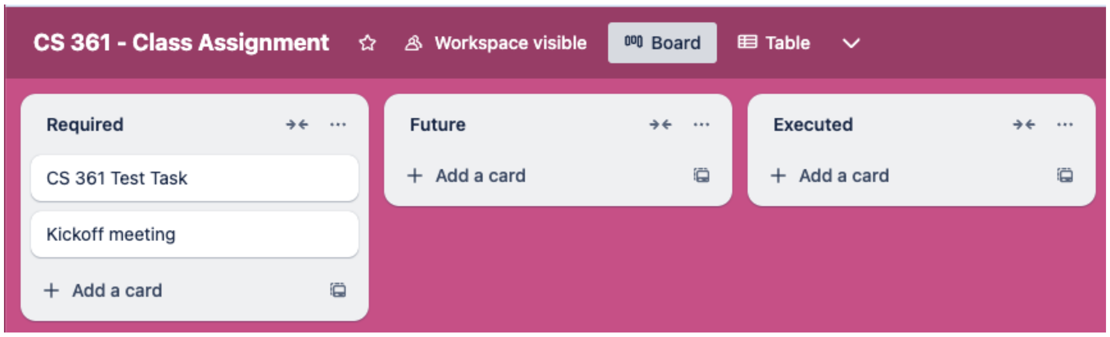
- Software specifically designed for task management
- Support for collaboration, task definition/deletion/updating, task priorities, task due dates, assigning people to tasks, setting task status, and the ability to label/tag tasks or put them in different columns.

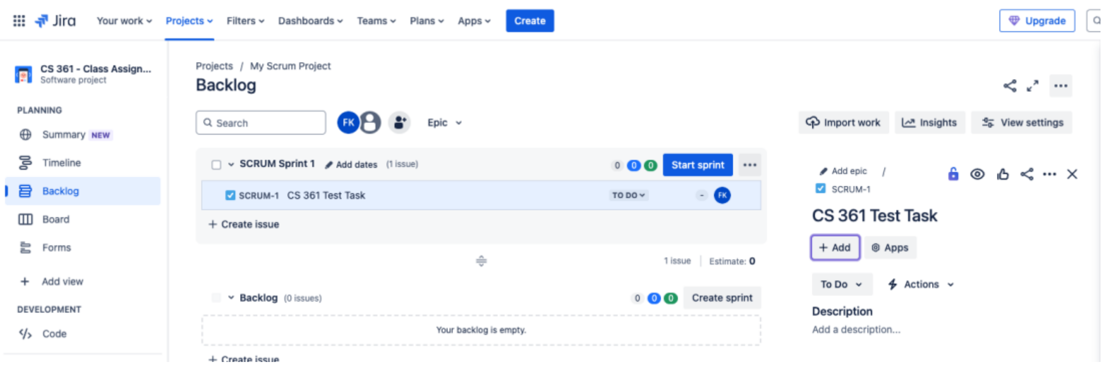
To do a spike, you need to research the task management systems and also (1) try to **use** them, (2) **evaluate** them based on specific criteria, (3) **compare** them, and (4) **decide** which to use.

a) Which task management systems did you spike?

GitHub - Projects
Trello
Jira

b) **Try** each system. Create a task then update it, assign it, delete it, etc. **Screenshot** your task in each system and paste it below. Name the tasks **"CS361 Test Task"**.

 CS 361 - Class Assignment (Fariss) View 1 · New view Filter by keyword or by field Title 1 Finish and turn in Assignment 2 document. 2 CS 361 Test Task You can use Control + Space to add an item CS 361 Test Task Draft khairyf opened now khairyf now (edited) Edit No description provided Assignees khairyf Status In Progress Convert to issue Copy link in project Archive Delete from project GitHub - Projects
 CS 361 - Class Assignment · Workspace visible · Board · Table Required: CS 361 Test Task; Kickoff meeting; + Add a card Future: + Add a card Executed: + Add a card Trello



Jira

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: *GitHub -Projects*

Evaluation of system 1's ease of use (2+ sentences):

Simple to use, with easy navigation between types of layouts to use to manage tasks. Layouts include table, columns, and timeline. Fully integrated with GitHub repository.

Name of system 2: *Trello*

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

Software's ease of use is very simple - all functionality is easy to find. Required buttons for quick access are all clear and simple to navigate around.

Name of system 3: *Jira*

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

Simplistic for a dense application. Although there are many moving parts to the software, the start guide made it easy to set up the system.

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: *GitHub - Projects*

Evaluation of system 1's speed/responsiveness (2+ sentences):

Speed and responsiveness is instantaneous. Components of website load on a click with no lag.

Name of system 2: *Trello*

Evaluation of system 2's speed/responsiveness (2+ sentences):

Software's speed is instantaneous. Responsiveness is click-immediate.

Name of system 3: *Jira*

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

Instantaneous responsiveness. I did not notice any lag.

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

Name of system 1: *GitHub - Projects*

Evaluation of system 1 (2+ sentences):

Full integration with repository on GitHub. Makes for easy integration with assignment files and commits.

Name of system 2: *Trello*

Evaluation of system 2 (2+ sentences):

Trello has a automated weekly summary of tasks and information for the selected project. I would use this to update my teammates automatically.

Name of system 3: *Jira*

Evaluation of system 3 (2+ sentences):

Jira has full integration of Agile Scrum. Many of the software components were identifiable after reading the Scrum assignment, so I would want to use this for its Scrum integration.

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

Name of system 1: *GitHub- Projects*

Evaluation of system 1's relevance popularity (2+ sentences):

In terms of relevance, I will see this software again. It is highly integrated with GitHub's repository software, so I will use it if I need to in the future.

Name of system 2: *Trello*

Evaluation of system 2's relevance popularity (2+ sentences):

Because you need to pay to use this software, I will not use it again. I can use other options that don't require payment.

Name of system 3: *Jira*

Evaluation of system 3's relevance popularity (2+ sentences):

No, you have to pay for full features. There are other options I can use without payment.

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: *GitHub - Projects*

System 1 ease of use: *Best*

System 1 speed/responsiveness: *Best*

System 1 feature set: *Middle*

System 1 relevance/popularity: *Best*

System 2 name: *Trello*

System 2 ease of use: *Middle*

System 2 speed/responsiveness: *Middle*

System 2 feature set: *Worst*

System 2 relevance/popularity: *Worst*

System 3 name: *Jira*

System 3 ease of use: *Worst*

System 3 speed/responsiveness: *Worst*

System 3 feature set: *Best*

System 3 relevance/popularity: *Middle*

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

GitHub - Projects

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 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 your task management system, or a label/tag the user stories "Product Backlog". You might end up changing these user stories later or adding new ones.

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 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.

[SOFTWARE NAME] = iilm
Build a software that shows users what they want to see and read. Things to be seen or read will be displayed on an online page.
Software will ask users to type something - either an image or words will be produced.

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) **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**

- **User story 1:** *Show image*
- **User story 1:** "As a user of the software, when I describe an image, I want to see it so that I can see what I want."

- **User story 2:** *Read text*
- **User story 2:** "As a user of the software, when I describe something in text, I want to read about it so that I can read/understand what I want about something."

- **User story 3:** *See past searches*
- **User story 3:** "As a frequent user of the software, I want to see what I have searched in the past so that I can navigate back to something I was thinking about in the past."

- **User story 4:** *Security of software use*
- **User story 4:** "As a previous user of the software, I want my data to only reflect in my account so that other people won't have my data/past searches."

- **User story 5:** *Greeting from software*
- **User story 5:** "As a frequent user, I want to be greeted with the time, date, and weather when I log in so that I feel welcomed and feel like I am communicating with someone when using the software, rather than just typing to a blank screen."

- **User story 6:** *Farewell from software*
- **User story 6:** "As a frequent user of the software, I want to be said my farewells (good-bye message) when exiting the software so that I can leave the software with good feelings and a time well spent."

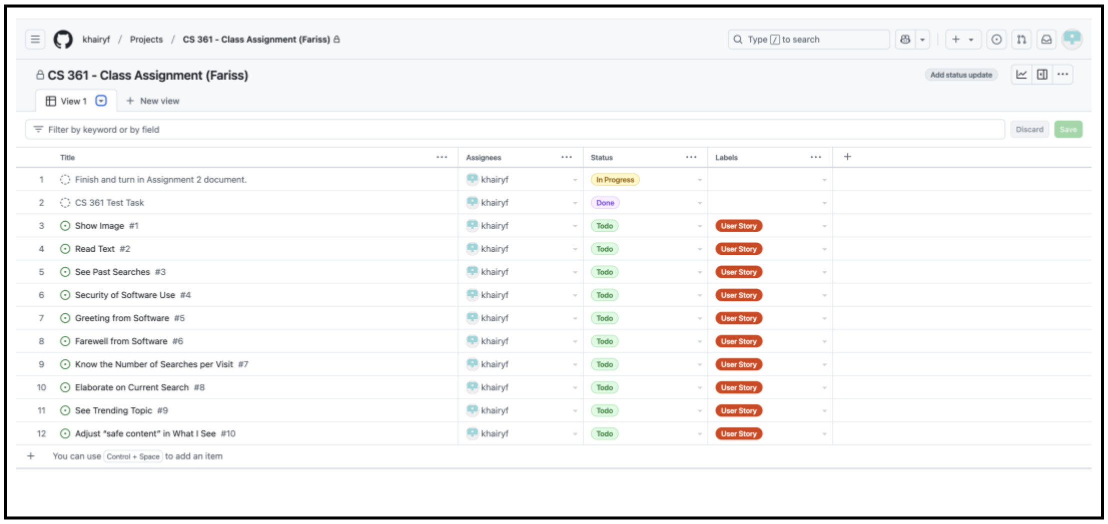
- **User story 7:** *Know the number of searches per visit.*
- **User story 7:** "As a user, I want to know how many searches have occurred so I can keep track of my time and efforts when using the software."

- **User story 8:** *Elaborate on current search.*
- **User story 8: "**As a user activating a search, I would like to have the chance to see/learn more so that I can gain more knowledge."

- **User story 9:** *See trending topic*
- **User story 9: "***As a frequent user, I would like to see what is relevant in the online world so that I can learn more about my social environment if I want to."*

- **User story 10:** *Adjust "safe content" in what I see.*
- **User story 10:** *"As a user, I want to be able to decide whether to see explicit content or not in my searches so that I am prepared for what I will see/read."*

Enter the user stories into your task management system in a **Product Backlog column/section/category**, or with a “Product Backlog” label/tag. Paste a **screenshot** below so that the grader can confirm you added the stories.


khairyf / Projects / CS 361 - Class Assignment (Fariss)
Type / to search
Add status update
View 1
New view
Filter by keyword or by field
Discard
Save

CS 361 - Class Assignment (Fariss)

	Title	Assignees	Status	Labels
1	Finish and turn in Assignment 2 document.	khairyf	In Progress	
2	CS 361 Test Task	khairyf	Done	
3	Show Image #1	khairyf	Todo	User Story
4	Read Text #2	khairyf	Todo	User Story
5	See Past Searches #3	khairyf	Todo	User Story
6	Security of Software Use #4	khairyf	Todo	User Story
7	Greeting from Software #5	khairyf	Todo	User Story
8	Farewell from Software #6	khairyf	Todo	User Story
9	Know the Number of Searches per Visit #7	khairyf	Todo	User Story
10	Elaborate on Current Search #8	khairyf	Todo	User Story
11	See Trending Topic #9	khairyf	Todo	User Story
12	Adjust "safe content" in What I See #10	khairyf	Todo	User Story

You can use Control + Space to add an item

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:**
  How often the software’s functions succeed or fail.

- **Quality attribute 2:** *Securability*
- **Quality attribute 2 definition:**
  The characteristic or degree of being securable (a resource being securable).

- **Quality attribute 3:** *Simplicity*
- **Quality attribute 3 definition:**
  The state or quality of being simple.

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

- **Why quality attribute 1 is relevant to your project:**
  Reliability was chosen because the user would want the data they receive to be familiar

with what they want, so the software has to be reliable in its execution.

- **Why quality attribute 2 is relevant to your project:**
  Securability was selected because the software and user experience is quite personal because the user will be willingly expressing what they are thinking, so the software needs to be trusted to not expose user's thoughts and expressions.
- **Why quality attribute 3 is relevant to your project:**
  Simplicity was chosen because the software is supposed to embed itself onto the user - the user can freely use the software to its desire, and the software's appearance and design should not be a burden to the user.

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

Next, move some user stories from your Product Backlog to your Sprint Backlog, 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.

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)

Create code that enables searches to the software.
Person can enter what they want to see/read and they will be able to receive data from software.

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.

First user story

(Front of index card)
Show Image *"As a user of the software, when I describe an image, I want to see it so that I can see what I want."*
(Back of index card) **Acceptance criteria** Functional requirements ● *Given an input when a user enters text to the screen, then an image will be shown of the user's input.* Quality attributes & Non-functional requirements ● *A relevant image (reliability) should be shown only to the user (only appears on the user' screen where the user expects it) (securability) in a simple manner - not showing too much or too little (simplicity).*

Second user story

(Front of index card)
Read text *"As a user of the software, when I describe something in text, I want to read about it so that I can read/understand what I want about something."*
(Back of index card) **Acceptance criteria** Functional requirements ● *Given an input when a user enters text to the screen, then output text will be shown regarding the user's input.* Quality attributes & Non-functional requirements ● *Accurate text (reliability) in relation to the user input will be shown, only to the user's screen (securabiility), in a concise manner where it will show medium-length (1 paragraph or less) and concise information (simplicity).*

Third user story

(Front of index card)

Greeting from software

"As a frequent user, I want to be greeted with the time, date, and weather when I log in so that I feel welcomed and feel like I am communicating with someone when using the software, rather than just typing to a blank screen."

(Back of index card)

Acceptance criteria

Functional requirements
- *Given an account access when a user uses the software, then the user will be greeted with the time, date, and weather.*

Quality attributes & Non-functional requirements
- *A welcome message will always be displayed after account access (reliability), only after successful account access trial (securability), in a short, easy-to-read manner (simplicity).*

Fourth user story

(Front of index card)

Farewell from software

"As a frequent user of the software, I want to be said my farewells (good-bye message) when exiting the software so that I can leave the software with good feelings and a time well spent."

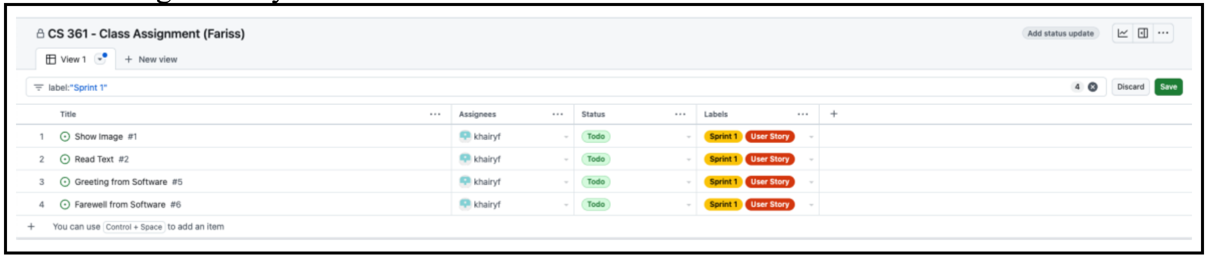
(Back of index card)

Acceptance criteria

Functional requirements
- *Given an end to software use when the user wants to close software, then a farewell message will be displayed.*

Quality attributes & Non-functional requirements
- *A farewell message will always be displayed upon end of software usage (reliability), only once the user software logs out of their credentials (securability), in a simple, easy manner (simplicity).*

3) Take a **screenshot** that shows you’ve moved these user stories into a Sprint Backlog in your task management system.



CS 361 - Class Assignment (Fariss)

label:"Sprint 1" 4

	Title	Assignees	Status	Labels
1	Show Image #1	khairyf	Todo	Sprint 1, User Story
2	Read Text #2	khairyf	Todo	Sprint 1, User Story
3	Greeting from Software #5	khairyf	Todo	Sprint 1, User Story
4	Farewell from Software #6	khairyf	Todo	Sprint 1, User Story

You can use Control + Space to add an item

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 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.

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.