**Sprint Plan Template - CMPS 115 – Software Methodology**

At the end of your team's sprint planning meeting, the team needs to turn in a sprint plan. This document needs to be typewritten (or the team needs to use a web-based agile planning tool and provide the TA/tutor access to the tool to view the project) and have the following elements:

∙ **Heading:** Document name ("Sprint {number} Plan"), product name, team name, sprint completion date, revision number & revision date.

Hungry Slugs

Sprint 3 Completion Date: 5/21/24

Revision Number: 1.1

Revision Date: 5/13/24

∙ **Goal:** Short, 1-2 sentence description of the high-level goal(s) for the sprint. ∙ **Task listing, organized by user story:** This section lists the user stories, in priority order from most important (top) to least important (bottom). Within each user story, there needs to be a list of tasks required to implement the user story, along with the time estimate for each tasks (preferably less than or equal to 6 ideal hours). This should look like:

Old user stories:

US1: As a student, I want to see other people’s ratings of the food so that I know if the specific food is good. [8]

1. Create a login page where users log in/sign up using a Google account. (6 hrs)
   1. A separate page or place on the screen that allows people to log into their account
   2. Login in with google button or already signed and takes to profile page
   3. Create a blank profile page
   4. <https://django-oauth-toolkit.readthedocs.io/en/latest/index.html>
   5. <https://www.hacksoft.io/blog/google-oauth2-with-django-react-part-1>
2. Create a user profiles page that displays current known information about the user. Use backend functions to get data on the user. (3 hrs)
   1. Create a basic figma design
   2. Make a post request that can get the data from the backend and then display it on the front end from the user collection
   3. Display the user’s past reviews
   4. Display the images the user has posted
   5. Display the user’s favorite meals
   6. Display the allergies and allow users to change them
      1. Add backend functions to allow users get/change allergies
      2. Create a ui for users to choose which allergies they have.
3. Add a UI to rate the dining hall foods and connect this to the backend. (6 hrs)
   1. Have a drop down menu to allow the user to click on the rating they want to give (perhaps like myAnimeList).
   2. Create a food object on the db from a post request or fetch the food from the db if it already exists. Only fetch the ratings portion of the food.
   3. Show their rating on the meal after they have given the rating in the menu ui
      1. Add the allergies for the food if they are new or the food doesn’t exist
   4. Create/connect buttons to functions created on the backend for meals

US2: As a student with allergies, I want to be able to filter out the dishes that do not contain my dietary restrictions, so that I can find food options that fit my dietary preferences.

1. Add more search options like a check box or drop down menu to filter types of food. (3 hrs)
   1. Create a basic figma ui or drawn outline
   2. Implement the ui on the search page
   3. Implement the ui on the food page as mocked up in Figma
2. If we save the user’s allergies on the backend, we can automatically fill in certain boxes if the user wants. (2 hrs)
   1. Create backend functions to receive the users allergies and preferences
   2. Load these on the frontend
3. Take data already on the frontend for the allergies and display icons next to each food for what the allergies are. (1 hrs)
   1. Add all allergy data to the backend
   2. Associate an icon or image with each string for an allergy

New user stories:

US3: As a user, I want to be able to post pictures of food I get at the dining hall and share it, so that I can show the quality of the food being served. [8]

1. Be able to click on a food name in the menu which leads to a page that has tabs for comments, ratings, and images. (6 hrs)
   1. Figma design for the food page
   2. If needed create a basic photo
   3. Create a new food page with tabs for comments, images, and ratings data
      1. Be able to find these food pages on the dining hall menu page
      2. Take the string from the locations collection and add it to the food db if it does not exists
      3. The foods collection will store the data for comments, images, and ratings

2. Display/upload images in the image tab. (6 hrs)

* 1. In the images tab have a photo album
  2. Button to upload photo to album
  3. Function to load photo to backend
     1. Store the photo in mongodb
  4. Function to load photo from backend
  5. Hash each image to a tag in the food objects.
  6. grid FS
  7. Store file names in food object

US4: As a user, I want to be able to post comments on the food I get at the dining hall, so I can communicate with other users about the dining hall. [8]

1. Temp fixes to not rely on US1 (1 hr)
   1. Create a dummy account on each page load, generate a random name
2. Create a page that is similar to the reddit threads UI for comments.

Part 1 frontend: (6 hrs)

* 1. Create a basic figma ui that can show the comments (in comments tab of food page)
  2. Create a ui to see the comments
     1. Show username for person who posted the comment
     2. Show when the comment was posted
        1. The time can be received from django
     3. Create temporary means of linking to the user’s account
  3. Create a ui to post a comment
     1. Text box
     2. Button to post comment

Part 2 backend: (2 hrs)

* 1. Function to post a comment on the backend
  2. Function to get all comments from the backend (stored in foods db)

∙ **Team roles:** Give a listing of all team members. Next to the team member, list their role(s) for this sprint. Assign each person to at least one role (for example, this role might be "Developer"). This looks like:

*Team member 1: role 1 {, role 2, role 3}*

*Team member 2: role 1 {, role 2, role 3}*

*...*

*Team member N: role 1 {, role 2, role 3}*

* Ian Holloway: Product Owner / Developer
* Akshat Tiwari: Developer
* Anya Zhang: Scrum Master / Developer
* Akhil Senthil: Developer
* Noah Kim: Developer

∙ **Initial task assignment:** A listing of each team member, with their first user story and task assignment. This should look like:

*Team member 1: user story, initial task*

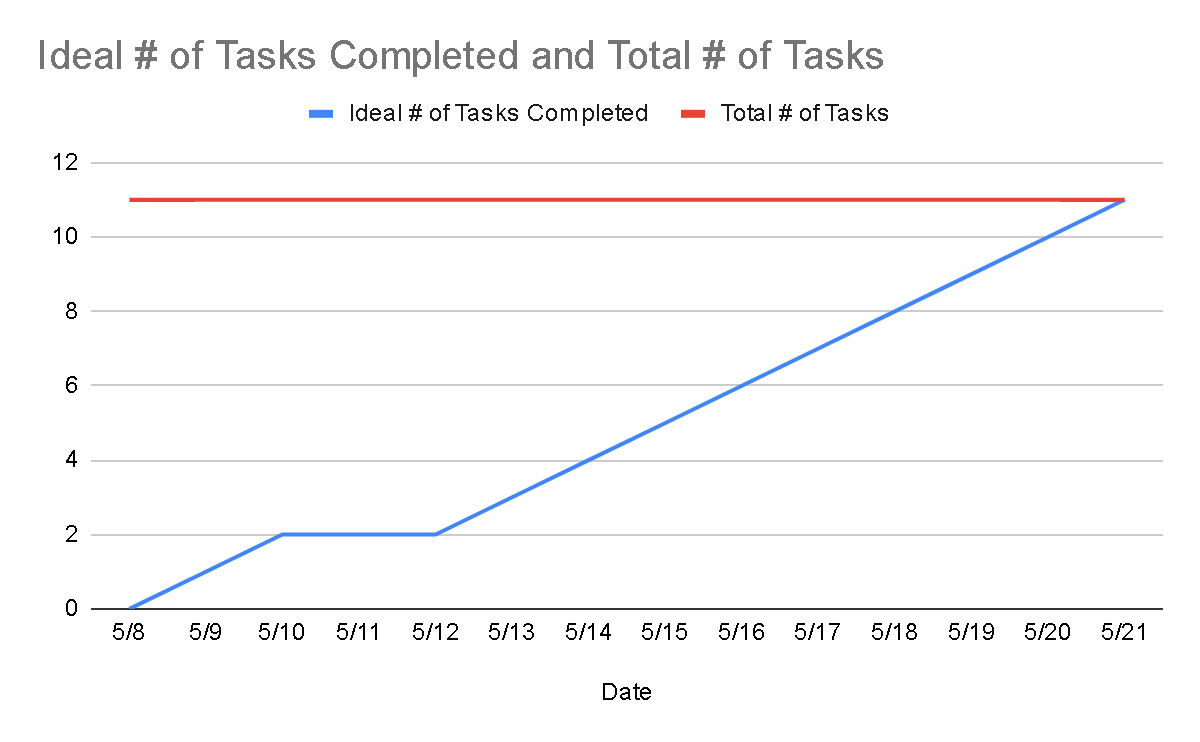
*Team member 2: user story, initial task*

*...*

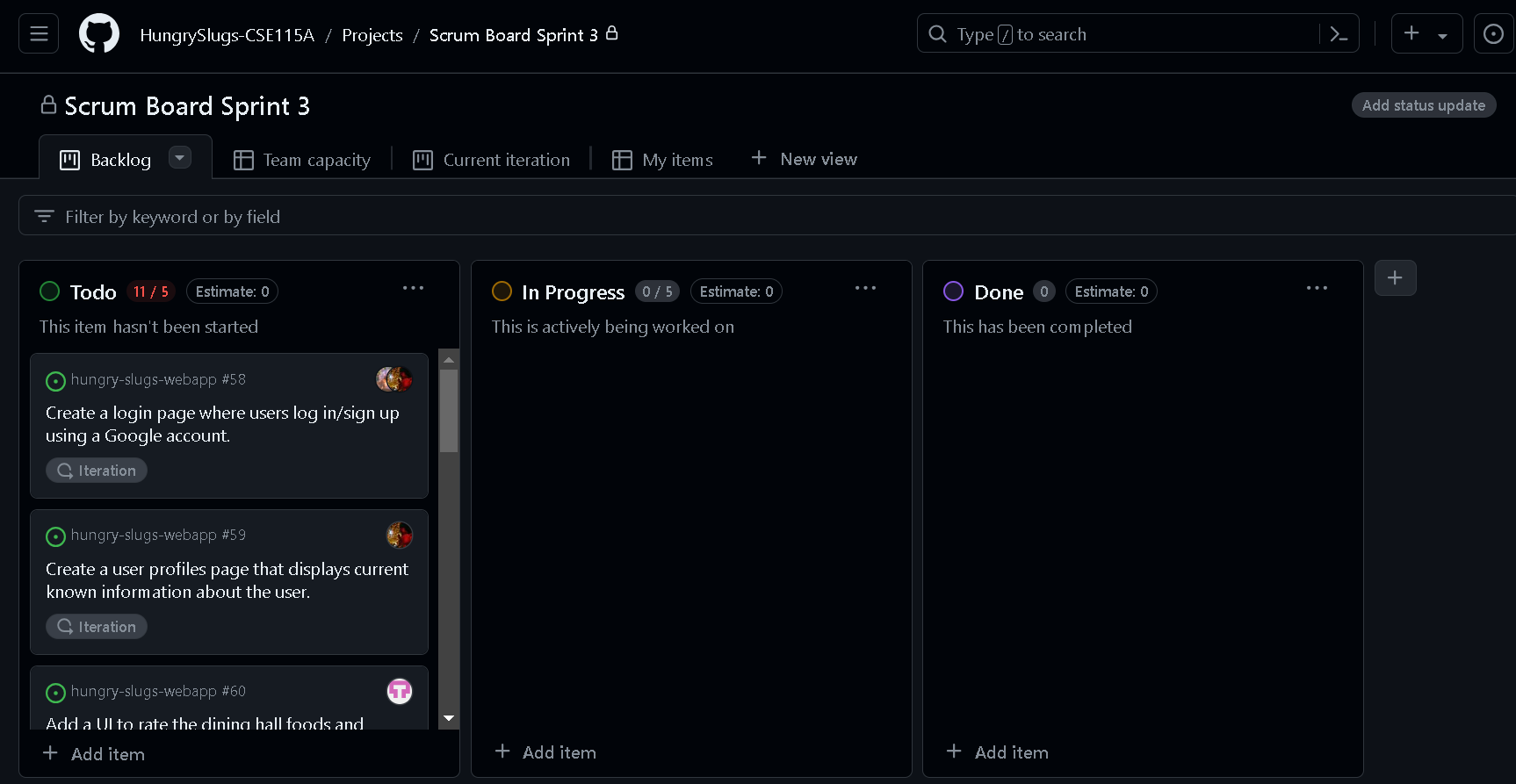
*Team member N: user story, initial task*

* Anya: US1: 1 helping, US4: Part 1 (frontend)
* Akshat : US2:1, US2:2, US2:3
* Akhil: US1:1, US1:2
* Ian: US4:1, US4:1 P2 (backend), US3:1
* Noah: US1:3 , US3

∙ **Initial burnup chart:** A graph giving the initial burnup chart for this sprint and is labeled as such with sprint number and project name and is located in the lab.



∙ **Initial scrum board:** Also known as a task board, the scrum board is a physical board and labeled as such with sprint number and project name and located in the lab. This board has four columns, titled user stories, tasks not started, tasks in progress, and tasks completed. Index cards or post-it notes representing the user stories and the tasks for this sprint should be placed in the user stories, tasks not started, and tasks in progress columns. Tasks associated with a user story should be placed in the same row as the user story.



∙ **Scrum times:** List at least the three days and times during the week when your team will meet and conduct Scrum meetings. Also, indicate which of these meetings will have the TA/tutor visit as arranged with the TA/tutor. It is expected the TA/tutor will visit during the Scrum meeting during your lab time.

1. Mondays 9:00am - 9:15am, with team
2. Mondays 11:00am - 12:00pm, with TA
3. Wednesdays 9:00am - 9:15am, with team
4. Fridays 9:00am - 9:15, with team

Backlog:

As a student, I want to get notified when a dining hall is serving a specific dish, so that I can know when and where my favorite food is being served. [5]

* Pull backend data of the user’s favorite meals
* Create a popup on each page to display which dining halls are serving the favorite foods