Sprint 3 Report

CSE 115A
TheGoodPlates
5/17/2022

- **Actions to stop doing:**
  - Adding more sections that introduce new functionality
    - Instead of adding more functionality, the team needs to start focusing on improving the existing functionality that we worked on
  - Stop underestimating the difficulty of implementing user stories
    - Be more realistic about requirements, as well as developer abilities

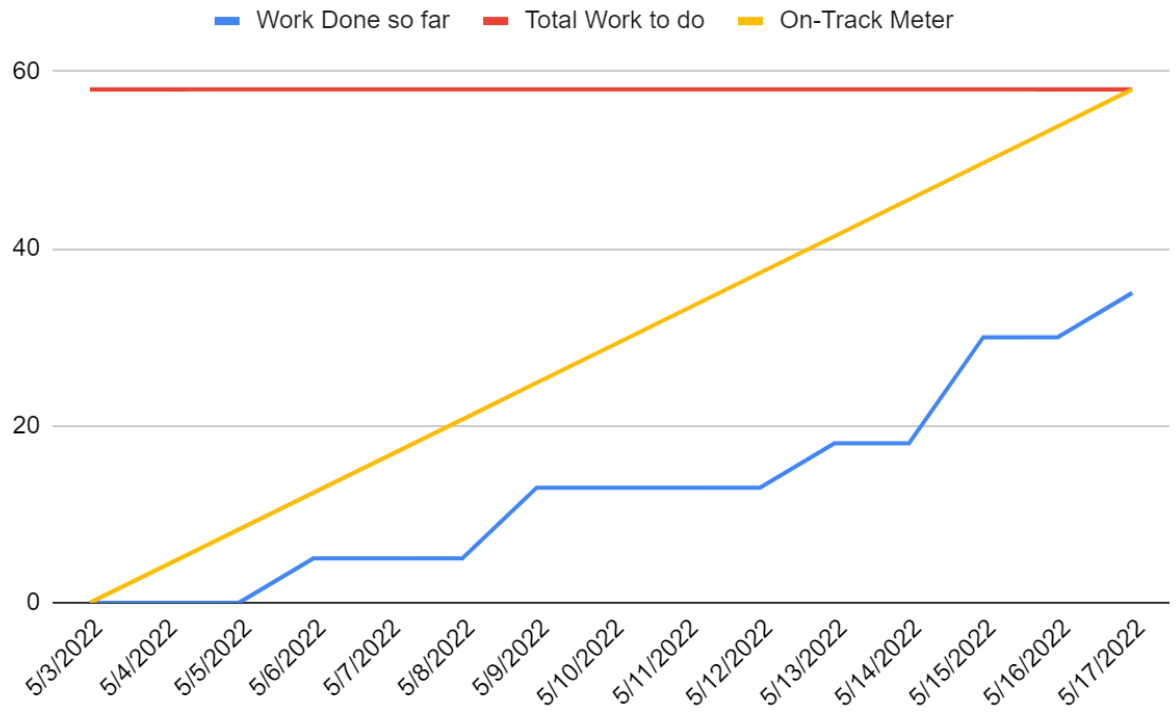
- **Actions to start doing:**
  - Organize the website functionality in a way that is more clear/beneficial to users
    - E.g. Organize the preference list, organize the list of restaurants that API is currently returning
  - Brainstorming detailed functionality of the project
    - We need to focus on the user stories that have higher priorities but also the details of each feature we have developed so far so that it benefits the better communication between users and the application
  - Apply the API features we want to use
    - It will help return actual useful/quality recommendations to users
  - Consider what might cause the criticism from the audience and try to solve/find the solutions for the problem
    - This will help prevent the problem that might happen after the application is released
    - For example, why does the team exclude the 'Rating' category from the Yelp API → because 'rating' is very personal and not trustworthy sometimes
  - Delete unnecessary codes that the team ends up not using
    - This will make the clean code which benefits the team to understand the code better together
    - This will help not use too much storage when the team executes the code
  - Everyone commits/pushes more often to the Github page
    - The team can share the update of the project as soon as other teammates finished
    - The team can keep track of the individual progress
  - Submit individual progress to Notion
    - Keep track of the progress and move to-do tasks
    - Can find all the documents in one place
  - Keep spikes in mind when developing user stories

        - Better time management and planning
    - Break down the user stories into smaller tasks
        - The tasks that have to be completed become clearer
        - Easier to assign individual tasks
        - Less pressure as well as less vague
        - Having a list of developer tasks

- **Actions to keep doing:**
    - weekly meetings
        - everyone is up to date and changes can be discussed
    - Pushing information to GitLab Master branch
        - helpful so everyone is on the same page
    - Sharing any tutorials/information on programming languages
        - helpful so everyone is on the same page
    - Everyone is putting in a great amount of work

- **Work completed/not completed:**
    - TASKS COMPLETED: User Story 1, 2, 5, 6
        - As a user, I want the recommended restaurants to cater to my preferences and location so that I can look through the restaurants based on that
        - As someone who cares about their online security, I would like to have a private profile to use the website
        - As a user, I would like my recommendations to evolve based on my likes/dislikes
        - As a user who has already chosen some restaurant types based on my preferences, I want to see the history of restaurants I have liked
    - NOT COMPLETED: User Story 3, 4
        - As someone who doesn't like repeating myself, I want my profile to save the information I input
        - As a user, I want to have access to my saved information to check if I input it correctly

- **Work completion rate:**
    - Total number of **user stories completed**: 4
    - Total number of **estimated ideal work hours completed** during the prior sprint: 32 hours (excluding sprint meeting)
    - Total number of **days** during the prior sprint: 14 days
    - The final sprint **burnup chart**:



User Story 23	To Do 29	In Progress 1	Done 37
Sprint 3 25			
User Story 1: As a user, I want the recommended restaurants to cater to my preferences and location so that I can look through the restaurants based on that.	(3) Create a container for every type of information we need to store.	(3) Create a database through IndexedDB	(1) Save location and preference information
User Story 2: As someone who cares about their online security, I would like to have a private profile to use the website.	(3) Use GraphQL Queries to actually store the information that the user inputs.	+ New	(1) Send the information to the Yelp API for a list of restaurants that are requested by the search bar
User Story 3: As someone who doesn't like repeating myself, I want my profile to save the information I input.	(4) Figure out how to use the session identifier to access the database for user info.		(1) Combine and format the list of preferences into one string to be sent to yelp API
User Story 4: As a user, I want to have access to my saved information to check if I input correctly.	(4) Use GraphQL queries to pull out the list of liked restaurants.		(1) Send the information to the Yelp API from the cuisine buttons to get a list of restaurants
User Story 5: As a user, I would like my recommendations to evolve based on my likes/dislikes.	(4) Use the information gained from the queries to customize what the user sees on the webpage (show their previously liked restaurants).		(1) Format the all the requested information in order
User Story 6: As a user who has already chosen some restaurant types based on my preferences, I want to see the history of restaurants I have liked.	(4) Use cookies using IndexedDB to get session identifiers from users		(2) Learn how to use IndexedDB and GraphQL
+ New	+ New		(2) Create another section on the page that includes simple user login and register UI
			(5) Figure out how to input the list of restaurants to the algorithm
			(5) Design the algorithm itself
			(6) Connect list of liked restaurants to "liked restaurants" page
			(6) Transfer liked restaurants from the recommendations component to the parent component
			(6) Transfer the information about the restaurants to the "liked restaurants" page
			+ New