

SW Engineering CSC648 / 848 Spring 2022

BusyGator

Team 04		
Role	Name	Email
Team Lead	Samantha Saxton-Getty	ssaxtongetty@mail.sfsu.edu
Github Lead	Vishal Ramanand Sharma	vsharma5@mail.sfsu.edu
Front End Lead	Elyssa Mari Tapawan	etapawan1@mail.sfsu.edu
Back End Lead	Aaron Carlson	acarlson8@mail.sfsu.edu
Front End	Abdullah Sharaf	fabdullah1@mail.sfsu.edu
Front End	Siqi Guo	sguo4@mail.sfsu.edu
Back End	Janvi Patel	jpatel6@mail.sfsu.edu

March 19, 2022

Milestone 02

Date	Version
03.19.22	Finished Version 01 submitted
04.08.22	Revision completed per instructor feedback Version 02

Table of Contents

Executive Summary	3
Main Data Items and Entities - Glossary / Definitions	4
List of Functional Requirements - Prioritized	5
UI Storyboards for Each Main Use Case	7
High Level Architecture and Database Summary	11
Key Risks	13
Project Management	14

1. Executive Summary

There are many students who cannot find the resources they need to properly study, work out, or research. Our application, BusyGator, provides various materials to help SFSU students, staff, and faculty find everything they are looking for on campus all online. BusyGator is an e-commerce application that allows students, staff, and faculty to buy different items, from school materials to gym equipment, as well as list items for others to buy. This will motivate the SFSU community to use these products offered by the application to better their education and health.

The BusyGator will have numerous functionalities and services. The main service the application will provide is allowing SFSU students, staff and faculty to post and /or buy textbooks, sports gear, research devices, and much more all online. Each product that can be bought will show how much is in stock during the time of browsing, the price of the material, and the user selling the item. The BusyGator will provide a map pin-pointing various safe areas to pick up the product all around the SFSU campus. Other important functionalities the application will include are a search option and categories to look for more specific items, and a sign up / sign in option to confirm users are SFSU students or faculty. These services will provide users with an easy and accessible way to find various resources all in one place. BusyGator gives SFSU an opportunity to help their community by managing their resources within the application. Instead of going to different locations on campus to see if they can buy materials in person, the BusyGator will make this task a lot easier. It saves students, staff, and faculties the time and hassle in their already busy schedules. Also, this allows alumnis to give textbooks they no longer need to newer students who will utilize them more for classes. The unique aspect of BusyGator is that all of the features and services in buying and selling material will be in one place and in one press of a button, where anyone can easily follow.

We are Team 4 from the Software Engineering class of Spring 2022 consisting of Aaron Carlson, Siqi Guo, Janvi Patel, Samantha Saxton-Getty, Abdullah Sharaf, Vishal Ramanand Sharma, and Elyssa Mari Tapawan, with Samantha being the team lead. Other roles consist of Aaron being the back-end lead, Elyssa being the front-end lead, and Vishal being the GitHub lead. The back-end operates the parts of the application that aren't accessed by a user such as data organization. The front-end works on the parts of the websites that allow users to interact with the web features, such as the application design. The GitHub lead ensures that there are no errors and that the application is functional. We are a group of aspiring students learning the aspects of team software development by making an application ourselves. Our goal is to work together to create the BusyGator to help us experience and prepare ourselves for our future careers.

2. Main Data Items and Entities - Glossary / Definitions

1. user:

Definition: stores the login and personal information of user accounts

Sub-Items:

- 2.1 user_id: primary key used for database identification
- 2.2 first_name: first name of the user
- 2.3 last_name: last name of the user
- 2.4 email: email address of the user used for login and confirmation of official SFSU status
- 2.5 password: protected string created by the user for login
- 2.6 administrator: flag reporting whether the user is an administrator
- 2.7 date_created: date that the user account was created

2. product:

Definition: stores descriptive information regarding the application's products

Sub-Items:

- 5.1 product_id: primary key used for database identification
- 5.2 seller_id: foreign key identifier of the user selling the product
- 5.3 category: foreign key identifier of the category the product falls under
- 5.4 location: foreign key identifier of the location the product is to be exchanged at
- 5.5 title: name of the product
- 5.6 description: description given to the product
- 5.7 image: image showcasing the product
- 5.8 image_thumbnail: image showcasing thumbnail version of product
- 5.9 price: price value attached to the product
- 5.10 approved: flag reporting approval status
- 5.11 date_created: date that the product post was created

3. location:

Definition: stores information regarding the locations used for exchanges

Sub-Items:

- 6.1 location_id: primary key used for database identification
- 6.2 name: name of the exchange location

4. category:

Definition: stores information regarding the product categories

Sub-Items:

- 7.1 category_id: primary key used for database identification

7.2 name: name of the product category

7.3 description: description given to the product category

5. message:

Definition: stores information regarding the sending of in-site messages

Sub-Items:

9.1 message_id: primary key used for database identification

9.2 creator_id: foreign key identifier of the message's sender

9.3 receiver_id: foreign key identifier of the message's receiver

9.4 subject: subject of the message being sent

9.5 message_body: body of the message being sent

9.6 date_created: date that the message was created

3. List of Functional Requirements - Prioritized

Priority 1:

Unregistered User:

1.1 Unregistered Users will be able to search posts.

1.2 Unregistered Users shall be able to filter search results based on categories. This function will help to narrow down posts that fit what the user is looking for.

1.3 Unregistered Users shall be able to register for an account. The registration email can only be a SFSU email.

1.4 Unregistered Users shall not contact the seller. They will not be able message the seller unless they have a registered account.

Registered User:

2.1 Registered Users can perform all functions that Unregistered User can along with some additional functionalities.

2.2 Registered Users can log in to their accounts to buy / sell items.

2.3 Registered Users can request to reset their password in case they forget their password.

2.4 Registered Users can create posts. This function is required to help Users sell their products.

2.5 Registered Users shall be able to edit the post after it has been submitted.

2.6 If the product is not available anymore then the User can delete the post.

2.7 Registered Users can log out of their accounts. This is to enhance security in case they are accessing the account from a public place.

2.8 Registered Users can contact the seller of the post via in-site messaging after buy. This function is required for transactions between users.

Administrator:

3.1 Administrators will be able to delete / ban a User if they consistently fail to comply with the terms of service.

3.2 Administrators shall be able to delete or edit posts made by the User, if they do not follow the terms of service or they are offending in any capacity.

3.3 Administrators will be able to approve a post before it goes live, so that none of the posts go against the terms of service.

Priority 2:

Registered User:

2.1 The product can be reported to the Administrator if the product isn't accurate to the description provided / images displayed. In this case the seller shall be reported to the Administrator.

Administrator:

3.1 Administrators can perform all functions that a Registered User can.

Priority 3:

Registered User:

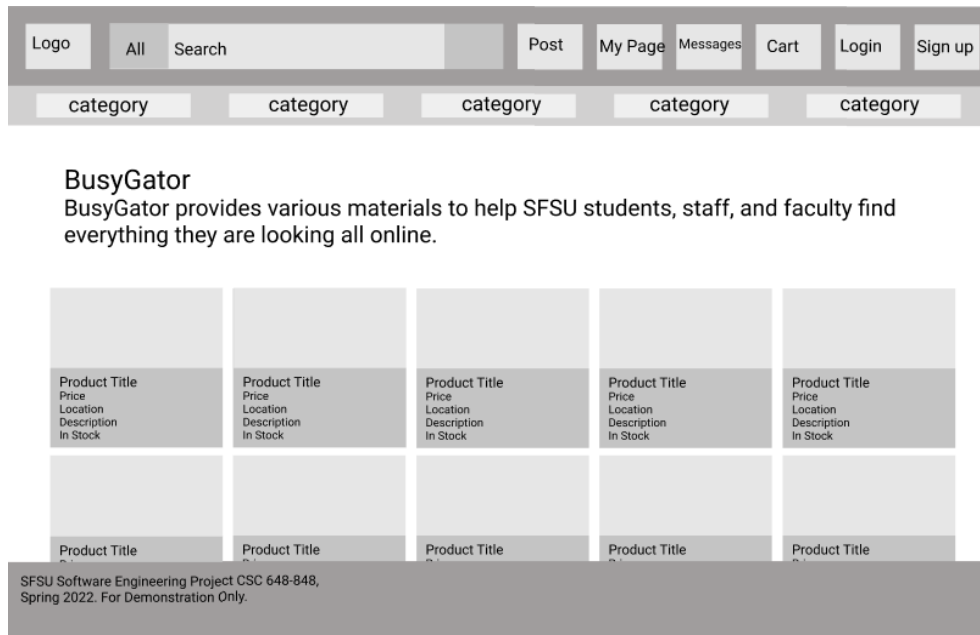
2.1 Registered Users can propose the meeting location and time to complete the transaction.

2.2 Registered Users can look at their transaction history in case they want to revisit a post.

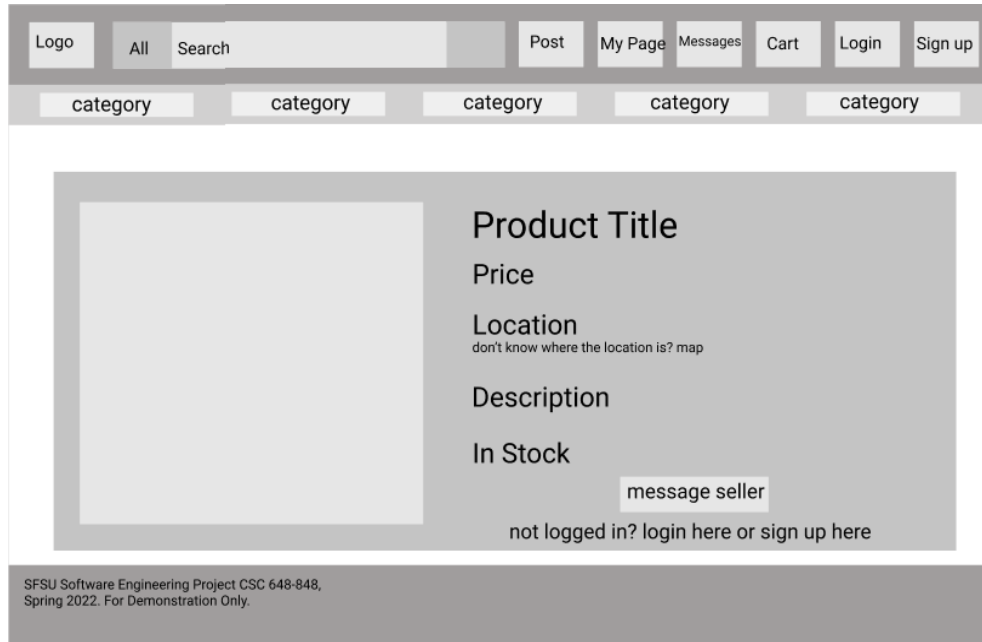
4. UI Storyboards for Each Main Use Case

Use Case 1: SFSU Student/Faculty (Buyer)

John is a student and a part-time worker at SFSU. He is looking for a textbook for his class on BusyGator. He enters what he is looking for in the search bar. The results show up on the “BusyGator” page.

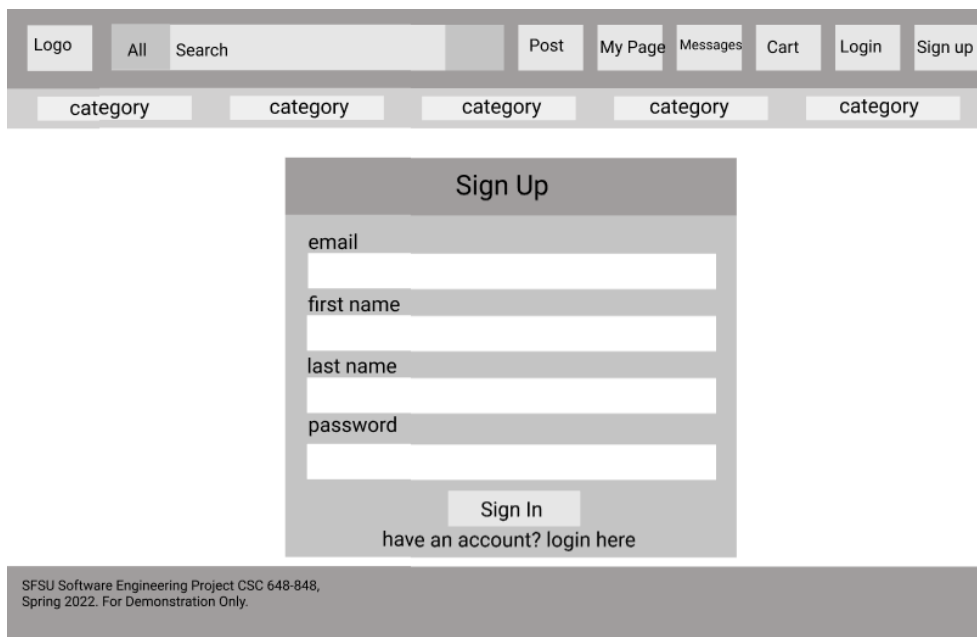


John looks on the page of all the results. He clicks on the product and sees the details of the textbook. He tries to message the seller, but it prompted to log in or sign up for an account before he can message the seller about the product.



The mockup shows a web interface for a product. At the top is a navigation bar with a 'Logo' button, an 'All' button, a search input field, and buttons for 'Post', 'My Page', 'Messages', 'Cart', 'Login', and 'Sign up'. Below this is a category filter bar with five 'category' buttons. The main content area features a large placeholder image on the left and a product details section on the right. The details section includes fields for 'Product Title', 'Price', 'Location' (with a subtext 'don't know where the location is? map'), 'Description', and 'In Stock'. A 'message seller' button is positioned below the 'In Stock' field. At the bottom of the details section, it says 'not logged in? login here or sign up here'. The footer contains the text: 'SFSU Software Engineering Project CSC 648-848, Spring 2022. For Demonstration Only.'

John doesn't have an account yet, so he clicks on the signup button and creates an account on the website by inputting his SFSU email, first name, last name, and password.



The mockup shows a 'Sign Up' form. At the top is a navigation bar identical to the one in the previous image. Below it is a category filter bar with five 'category' buttons. The main content area is dominated by a 'Sign Up' form. The form has a title 'Sign Up' and four input fields labeled 'email', 'first name', 'last name', and 'password'. Below these fields is a 'Sign In' button and a link that says 'have an account? login here'. The footer contains the text: 'SFSU Software Engineering Project CSC 648-848, Spring 2022. For Demonstration Only.'

After the account has been made, John logs in and can now message the seller of the product to obtain the product. We will implement the cart feedback in the future.

The screenshot displays a web application interface. At the top, there is a navigation bar with buttons for 'Logo', 'All', 'Search', 'Post', 'My Page', 'Messages', 'Cart', 'Login', and 'Sign up'. Below this is a row of five 'category' buttons. The main content area is titled 'messaging seller'. It contains a text input field with the placeholder '@buyer_name' and a 'Date Time' label. Below the input field, the text 'hello!' is visible. A message status bar indicates 'message has been sent!'. At the bottom of the messaging area is a 'home' button. The footer contains the text: 'SFSU Software Engineering Project CSC 648-848, Spring 2022. For Demonstration Only.'

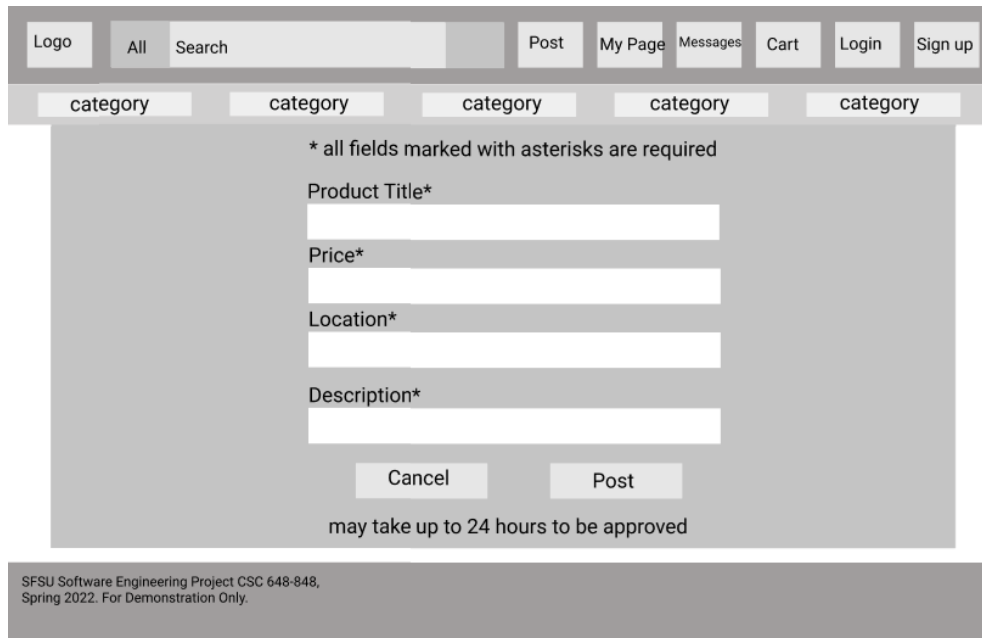
Use Case 2: SFSU Student (Seller)

Jennifer is a full-time SFSU student looking to list a textbook to sell. To list an item for sale, Jennifer needs to have an account on the BusyGator. Jennifer repeats the same process as use case 1 to log in, where she can now list a product. To list items, Jennifer needs to fill out the product's description when she is making a new post. After she finishes, she clicks the post button to have it listed.

The screenshot displays a web application interface. At the top, there is a navigation bar with buttons for 'Logo', 'All', 'Search', 'Post', 'My Page', 'Messages', 'Cart', 'Login', and 'Sign up'. Below this is a row of five 'category' buttons. The main content area is titled 'New Post'. It contains a form with the following elements:

- 'add image:' label
- 'choose file' button
- 'no image chosen' text
- A checkbox labeled 'accept use policy for uploading image'

 Below the form, there is a note: '*all fields marked with asterisks are required'. The footer contains the text: 'SFSU Software Engineering Project CSC 648-848, Spring 2022. For Demonstration Only.'



Logo All Search Post My Page Messages Cart Login Sign up

category category category category category

* all fields marked with asterisks are required

Product Title*

Price*

Location*

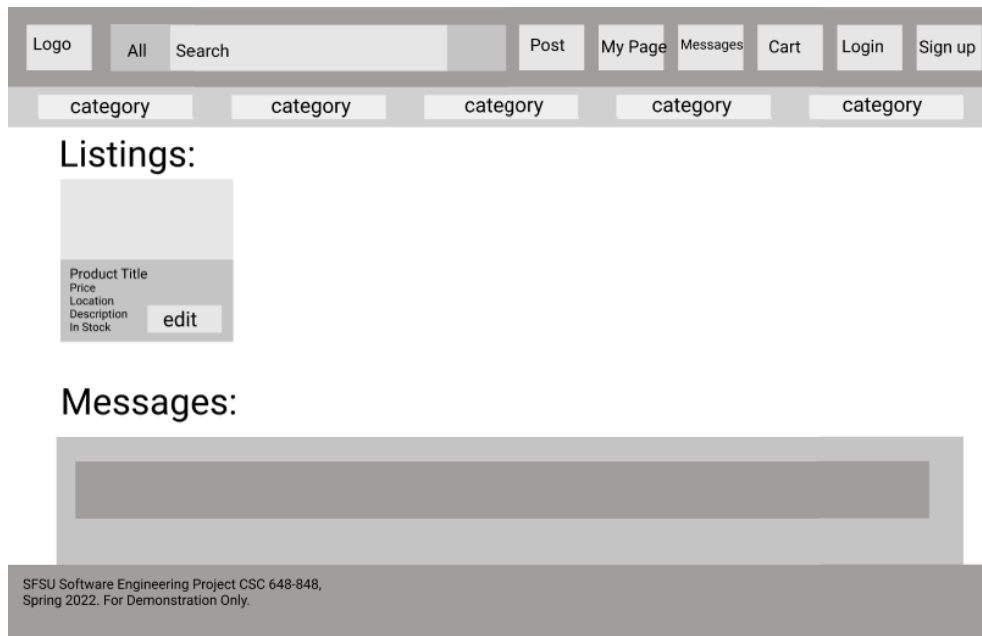
Description*

Cancel Post

may take up to 24 hours to be approved

SFSU Software Engineering Project CSC 648-848,
Spring 2022. For Demonstration Only.

Jennifer wants to update the price for an item she has already listed. She presses the my page button on the navigation bar to see her listings and messages and clicks the one she wants to edit. To edit the listing, she presses the edit button to update the prices for the listed item. She updates the price and saves the listing.



Logo All Search Post My Page Messages Cart Login Sign up

category category category category category

Listings:

Product Title	Price	Location	Description	In Stock	
					edit

Messages:

SFSU Software Engineering Project CSC 648-848,
Spring 2022. For Demonstration Only.

Jennifer finds out that she lost the textbook and wants to close the listed item. She clicks the listings button again on the top. She presses the edit button and presses the delete button where it will prompt “Are you sure you want to delete this listing?” She presses yes and deletes the listing.

The screenshot displays a web application interface. At the top, there is a navigation bar with a 'Logo' button, an 'All' button, a search input field, and several other buttons: 'Post', 'My Page', 'Messages', 'Cart', 'Login', and 'Sign up'. Below the navigation bar, there is a row of five 'category' buttons. The main content area is a form for editing a listing. It contains a 'Product Title' label and input field, a 'Price' label and input field, a 'Location' label and input field, and a 'Description' label and input field. At the bottom of the form, there are three buttons: 'Cancel', 'Editing Done', and 'Delete'. A footer at the bottom of the page reads: 'SFSU Software Engineering Project CSC 648-848, Spring 2022. For Demonstration Only.'

5. High Level Architecture and Database Summary

DB Organization:

BusyGators database organization is centered around two account types, user and administrator. Both account type tables hold information regarding the login while the user account also has more in-depth details regarding the users personal information.

The user interacts with products through transactions and posting, while the administrator interacts with products through approving user posts. Products are items with detailed information, including references to tables of location and category. In addition, the user interacts with other users through messages with the message recipient.

The naming convention utilized in this database is keyword based. If there is more than one keyword in the name, it is separated by an underscore. This keyword describes the data variable. For example, *keyword1_keyword2_keyword3*, *keyword1_keyword2* or *keyword*.

DB Tables:

1. user

- 1.1. user_id (PK)
- 1.2. first_name
- 1.3. last_name
- 1.4. email
- 1.5. password
- 1.6. administrator
- 1.7. date_created

2. product

- 2.1. product_id (PK)
- 2.2. seller_id (FK)
- 2.3. category (FK)
- 2.4. location (FK)
- 2.5. title
- 2.6. description
- 2.7. image
- 2.8. image_thumbnail
- 2.9. price
- 2.10. approved
- 2.11. date_created

3. location

- 3.1. location_id (PK)
- 3.2. name

4. category

- 4.1. category_id (PK)
- 4.2. name
- 4.3. description

5. message

- 5.1. message_id (PK)
- 5.2. creator_id (FK)
- 5.3. receiver_id (FK)
- 5.4. subject
- 5.5. message_body
- 5.6. date_created

Media Storage:

BusyGators media will be stored in a file system. The relative paths for each file will be stored in the image attribute within the product table. There will be a limit of maximum file size. User passwords will be encrypted and the file system will be protected using bcrypt to scramble file names.

Search / Filter Architecture and Implementation:

BusyGator will have a search bar that allows the user to search by item title and description. Users are able to then further filter by price and location. These filters will be created using the location and price attributes of the product table. If the user selects a category from the drop down or the sub navbar, that category is included with the search item. We will utilize SQL queries with %like% to match category and search terms together, or match the search term through all items if there is no category selected.

6. Key Risks

Skills Risk:

We will be working on different platforms and using different programs. Some of these programs are new to several of our group members. Our team members are guiding and helping each other knowing how it works.

How to Resolve:

For the members doing it for the first time, it will be difficult but our team shares resources, mini tutorials and guiding through Discord and Zoom meetings. We also do more research while using it.

7. Project Management

As soon as milestone 02 was released and reviewed in class, we immediately reviewed the document as a team to ensure everyone understood the milestone in its entirety. We focused heavily on revising milestone 01 for a smooth transition into milestone 02 since it would be a backbone for a lot of the content in the second milestone. Following this, I made a copy of our milestone 01 Google document and tailored it to the current milestone and its tasks. I added comments to each portion of the document assigning which member would complete each task, some including a brief note on expected work. During our meeting, we reviewed each assigned task and cleared up any questions before each member began their work. I utilized Discord to keep a list of “to-do” items that included meetings, deadlines for deliverables, and notes so that each team member could reference what would be going on during the given week. However, in the future I plan to use Trello as suggested by the professor for a more clean and concise area to divide work and display “to-do” items. Another tool we relied heavily on during meeting times was When2Meet. As a team we utilized this tool to quickly schedule follow up meetings that all team members were able to make in order to check-in on work progress, give feedback, and answer questions that weren’t covered over text on Discord. This milestone proved challenging in coordinating the front-end and back-end teams on their deliverables and concepts. Going forward, we will be utilizing Trello for project management and scheduling more frequent meeting times to ensure both teams are on the same page for fluid development.