

SW Engineering CSC648/848 Spring 2023

Milestone 2 part I: More Detailed Requirements, Specs,
Architecture, UI mock-ups

Food Feast

Team 01

Elahe Bashiri (Ebashiri@sfsu.edu): Team lead

Abbas Mahdavi: Back-end lead

Megan Lew: Front-end lead

Nathan Rennacker: GitHub master

Jed Graves: Developer

Alexander Diaz: Developer

Revision History

Date Submitted	Date Revised
3/31/2023	

1. Executive Summary:

Our team is developing a web application for restaurant search, order, and pickup called Food Feast. We are looking to address the issue of hungry SFSU students, staff, and faculty while supporting nearby local businesses. Our application is developed for exclusive use for the SFSU community. Many students and faculty usually have busy schedules and a limited amount of time to leave campus to search for food. It is important to have daily nutritious meals, but it may be difficult to find meals or have them delivered in time before the next class.

Food Feast will help users simplify the process of finding meals and having them delivered on campus or to their dorms. Our application will feature search/reviews of local restaurants, the ability of restaurants to register/advertise menus, the ability of restaurants to manage orders/delivery, and order meals. The delivery service will have access to a campus map for faster delivery times to classrooms and campus facilities. This makes our service unique because food delivery services usually deliver on a nearby street. Food delivery services are usually costly, our app will have a student verification system and offer students a 20% discount. We will also aim to provide a secure, fast payment system. Students can purchase their meals by linking their gator passes.

Our team consists of six members, and we are a student startup team at SFSU.

2. List of main data items and entities (expand as necessary):

1. Users:

- **Unregistered users:** Anyone visiting the website that has not created an account, who has limited access to the website but can still browse restaurants and items and create an account.
- **Registered SFSU users:** SFSU students, staff, and faculty who have full access to the website (Aside from admin, rest-reg, and driver section). Can log in and add items to the cart and place an order. Req Record: username, email, password
- **Registered Restaurant owners:** Users who have created a restaurant account while enrolling in their restaurant. Can manage their restaurant: delete, add/remove items, etc. Req Record: username, email, password, restaurant_id
- **Registered Drivers (Not a priority):** Users that created a driver's account, can pick up active orders and deliver them, can manage order status, and have access to restaurant addresses and delivery locations via a map to guide their delivery. Req Record: username, email, password

2. Account: The registered account of a user, including their data such as a username, email ending with @sfsu.edu (not required for restaurant owners and drivers), and password

3. Restaurants: The registered business by restaurant owners:

- Will include a unique ID, name, price range (1-5: \$), cuisine, description, rating (1-5), estimate delivery time, address, and main restaurant pictures

4. Menu Items: The list of the food for each specific restaurant, including the food item's unique id, name, description, price, corresponding restaurant id, and image.

5. An order: The record of a placed order that includes the user's info, the items in the order and their info, and the delivery location.

6. Delivery: The status of the order, and the driver's info

7. Location: The location of the restaurant, the building of the delivery, and the room number of the user

8. Map: A map available to the driver to help them deliver the food.

9. Payment: A selection when ordering to use Student ID or in-person cash!

10. User App features: Ability to see order status and 'Favorite' Restaurants/Food Items in particular restaurants

3. Functional Requirements - prioritized:

Priority 1

- **Unregister Users**

1. Register - Unregistered users shall be able to register for an account (inputting sfsu.edu information if applicable)
2. Browse Food Items - Unregistered users shall be able to see what restaurants have which items available for ordering.
3. Map - Unregistered users shall be able to see a map of the area with where the restaurants are in relation to you.
4. Filter by different categories - Unregistered users shall be able to click on different filters (price, cuisine, distance) above the list of restaurants, the ordering of the list will be changed depending on delivery time
5. Filing the restaurant form- Unregistered users shall be able to see the needed information and the policy for filing the form as a restaurant owner, it will be part of the registration choice.
6. Search Bar - Unregistered users shall be able to type in the name of a restaurant and its plans to it on the map. This search bar could autofill with suggestions.

- **Register Users**

7. Login -users shall be able to log in
8. Place order for delivery - users shall be able to select items from a menu and place an order with those items. This order will be conveyed to the restaurant with a driver specified so that it can be delivered to you.

- **Restaurant Owner**

9. Register for the service - Restaurants shall be able to opt-in to the service so that they will be displayed on the website as orderable. Will need specific information from them to ensure that they're legitimate.

- **Driver**

10. Delivery interface for drivers - Drivers shall be able to get their own orders which contain a map marked with the order number, etc. Drivers have a specific interface.

- **Admin**

11. Approve pending requests from restaurants - Admin shall be required to approve pending requests from restaurants that are available to get orders from the website.

12. Reject pending requests from restaurants - Admin shall be required to reject the requests which are inappropriate.

Priority 2

- **Unregister Users**

13. View the popularity of restaurants - Unregistered users shall be able to view which restaurants are popular at which times depending on how many people order from a specific restaurant at different times of the day (could also tap into external analytics, but not sure of the logistics of it)

14. Estimated time for delivery vs. pickup - Unregistered users shall be able to see the estimate of the delivery. Each restaurant has an estimated time to complete an order, and then the system will use that for pickup time or add on additional delivery time for delivery.

- **Register Users**

15. View Order Status - users shall be able to check a page to see the status of their order, any delivery updates, and the estimated time it will arrive.

16. Place Order for pickup - users shall be able to select items from a menu from a specific restaurant (who has provided their menu already) and place an order with those items. This order will be conveyed to the restaurant, which will have it ready for you to pick up.

17. Update Order Status - users shall be able to cancel or add things to their order (this times out depending on how far along the restaurant is with completing their order).

- **Restaurant Owner**

18. Update menu - Restaurants shall be able to upload and update their menu in an easy-to-use form (that can include pictures), which will then allow users to select from it.

19. Instructions to delivery drivers - Restaurants shall be able to give instructions to drivers, including a list of items that can include a lot of information or very little. This typically will include a specific list of the items that the customer ordered so that the driver can ensure that everything is present.

- **Admin**

20. Control the list of approved restaurants - Admin shall be required to delete restaurants If sees an inappropriate username or receives a complaint from someone about anything.

Priority 3

- **Register Users**

21. Write a review - users shall be able to get feedback. After you have ordered from somewhere, you will have the ability to give the restaurant a rating and include a written review if you choose.

22. Save restaurants - users shall be able to save their favorite restaurants. A little star will appear next to restaurants that you can click on, and it will add it to your list of saved restaurants (a list that you can easily access when you want to see what you like)

- **Restaurant Owner**

23. Advertising - Restaurants shall be able to advertise on the website would include things like promoted restaurants that prominently display their food in the specific well has seen places on the website. You can click on these advertisements, and it will take you to the typical page where you can order from them.

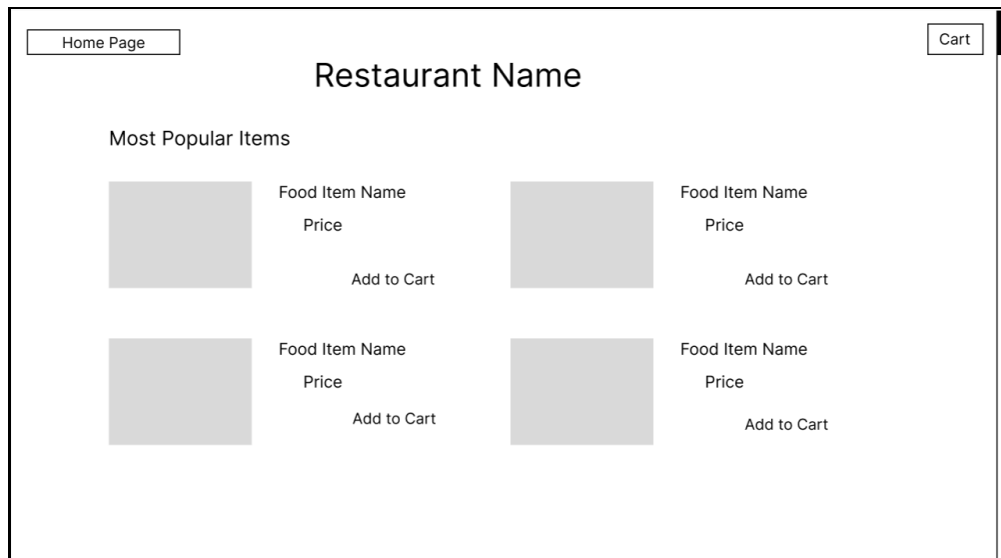
- **Driver**

24. Report information to customer - Drivers shall be able to update the customer with any extra information that the customer may need, including. I'm here with your food, Increased wait time, etc.

4. UI Storyboards for each main use case (low-fidelity B&W wire diagrams only):

- **Ordering Food**

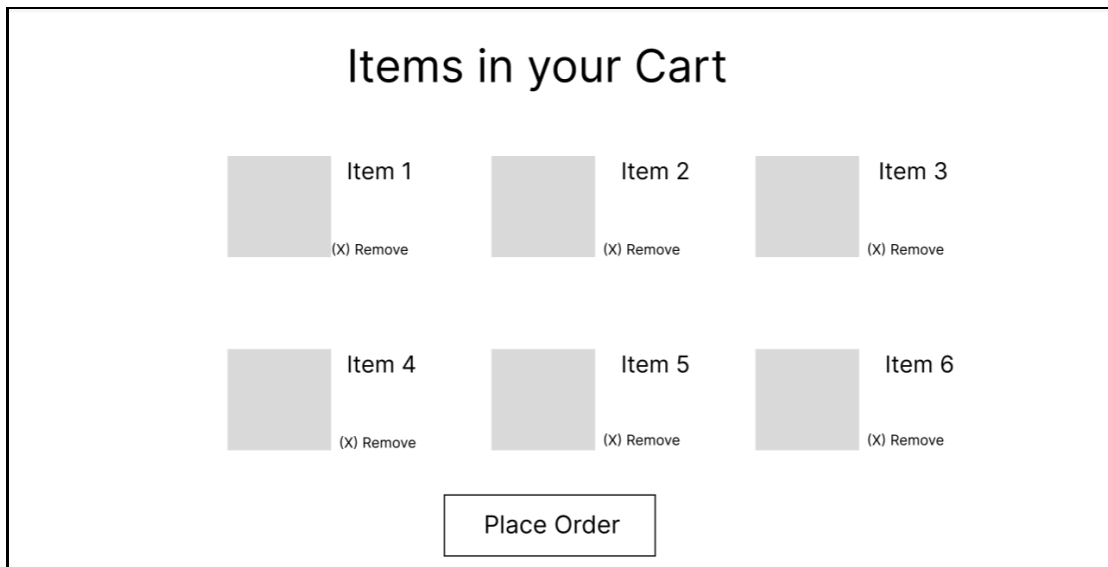
Dr. Demir is a Professor of Psychology at San Francisco State University. He has an extremely busy schedule, combining meetings, lectures, and research sessions. After his morning class, he has a meeting with the department head and only a short break in between. He knows that he will not be at his peak performance for the meeting and afternoon class if he doesn't eat, the break is not long enough to go off campus for food, and being lunchtime, it is also a busy time for most delivery apps. He knows, however, that the campus food delivery app delivers only from nearby restaurants using drivers that are in the immediate area. He also knows he can have the food delivered directly to his office. He opens the app and is immediately greeted with the home page. He sees a lot of options and promotions but doesn't want anything really heavy, like pasta or pizza, because that might make him tired. He sees the option for sandwiches and immediately selects it. He is presented with several local, high-rated options and selects one that looks good. Being pressed for time, he decides to select the featured sandwich and a coffee. Tapping the checkout button, he is directed to the login page. Now that he is logged in, his payment information is saved in the app, so checkout is a breeze. It arrives at his door shortly after he returns to his office. This will keep him alert and his blood sugar up through the afternoon.





Clicking on any food image on the front page or when searching through our restaurant list will take you to a restaurant specific page that lists out items that users can add to their cart.

- **Viewing Cart/Placing Order**

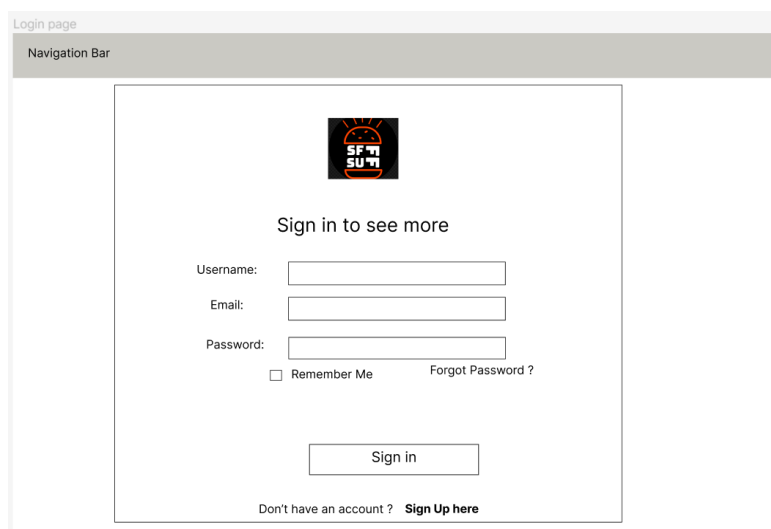
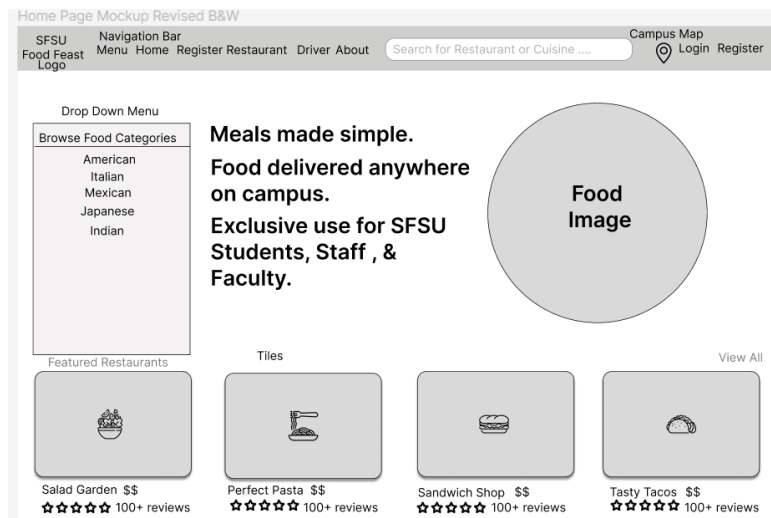


Clicking on the cart icon will take you to the current items you have selected to order, here you can remove items if you wish and also place an order.

- **Login/ Registration**


A new user visits the Food Feast's home page and decides to log in for the first time. When the user clicks on the sign-in button, they are redirected to the login page. However, since the user is not yet registered, they are prompted to create an account first.

On the sign-in page, the user sees a link that says, "Doesn't have an account? Sign Up here". When the user clicks on this link, they are taken to the sign-up page where they can enter their personal information. Once the user has successfully created their account, they are automatically redirected back to the home page. Then the user can access the features of the Food Feast application.



Registration page

Navigation Bar



Sign Up to have an account

Enter Username:

Enter Email:

Enter Password:

Re-Enter Password:

Already have an account ? [Sign In here](#)

Home Page Mockup Revised B&W

SFSU Food Feast Logo

Navigation Bar

Menu Home Register Restaurant Driver About

Search for Restaurant or Cuisine ...

Campus Map

Login Register

Drop Down Menu

Browse Food Categories

- American
- Italian
- Mexican
- Japanese
- Indian


Meals made simple.

Food delivered anywhere on campus.

Exclusive use for SFSU Students, Staff , & Faculty.


Food Image

Featured Restaurants




Salad Garden \$\$

☆☆☆☆ 100+ reviews




Perfect Pasta \$\$

☆☆☆☆ 100+ reviews



Sandwich Shop \$\$

☆☆☆☆ 100+ reviews



Tasty Tacos \$\$

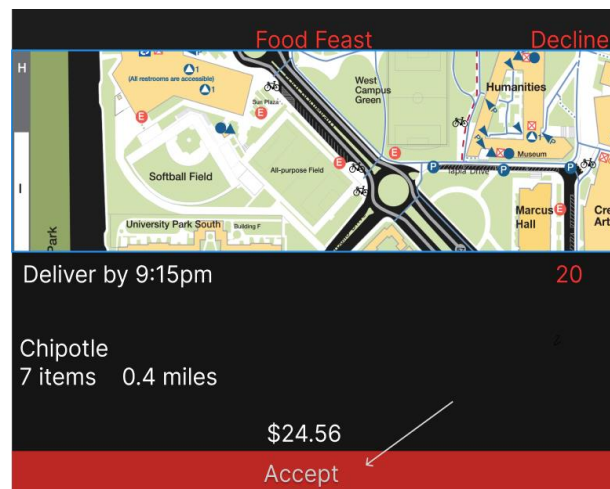
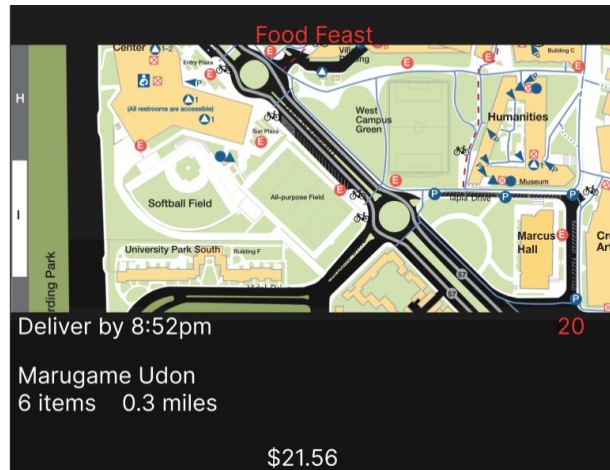
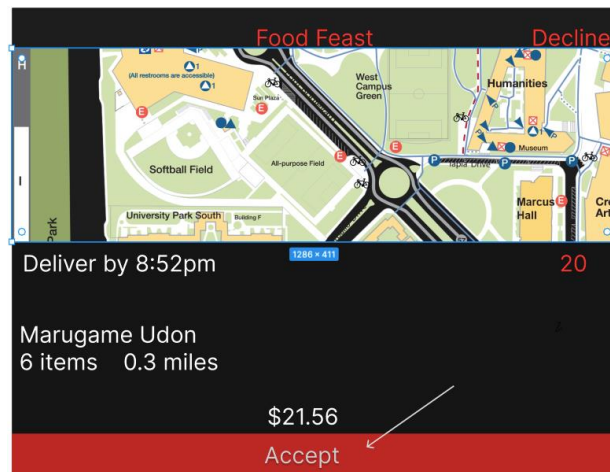
☆☆☆☆ 100+ reviews

View All

- **Driver**

Kylie is a 20-year-old university undergraduate student who is majoring in communications and hoping to pursue a career in public relations or marketing. She hangs out with her friends and orders food online for everyone, but delivery apps charge too much, and she does not want to break the bank and so she is looking for extra income. She then found the campus delivery app and decided to deliver food to students from nearby restaurants. The setup would be very simple. After she has registered for the app, she will click on “Drivers,” to which she would be shown orders in the area. It would show a map, the time to deliver the food, the restaurant and the number of items from the restaurant, the distance, the pay she would get upon completing the order, a accept/decline button, and a timer countdown in a circle showing how much time she has to decide. The decline would be on the top right of the screen, while the accept button would be on the bottom to avoid any mistakes and/or accidents.

If the declines the offer, the next available order will replace the previous, and so on and so forth. If she accepts the offer, she would have the same screen, except the “accept” and “decline” options would disappear.



- **Restaurant Manager**

Haru is the owner of a Japanese restaurant near the SFSU campus. He has only lived in the United States for 3 years, and his English is not yet fluent. His employees all speak Japanese as well, so that is not a problem for him. He recognizes the potential of marketing to the nearby campus but is not really sure how to do it. He tried setting up a delivery app service before, but it was confusing and frustrating, so he gave up quickly. Then he found the campus delivery app. It is exactly what he needs to reach hungry university students. He goes into his office and logs into his desktop computer. The setup is very simple and straightforward. He fills out the form and uploads some photos of the restaurant and the menu items. He clicks submit and is prompted to make an account. Once it's submitted, he is informed that it may take up to one business day for his application to be approved. The next day, his restaurant was listed on the app, and he got his first order. It was quickly picked up by a student delivery person. Haru is now very excited that he was able to expand his customer base.

5. High-level Architecture, Database Organization summary only:

DB organization:

Users

- Id (primary key)
- Username
- Email
- Password
- isAdmin
- isRestaurantOwner

Restaurants

- Id (primary key)
- Name
- Price
- Cuisine
- Description
- Rating
- Est_delivery_time
- Address
- picture

Menu_items

- Id (primary key)
- Name
- Price
- Restaurant_id
- Image

Media Storage:

Images are stored using DB BLOBS.

Search/Filter architecture/implementation:

Search is implemented with React Fuse.js dynamic search. It has fuzzy search capabilities and can handle misspelled words. It can specify three keys in our data that we want to search on which are name, cuisine, and description. Priority 2 is viewing the popularity of restaurants based on time. There can be a sorting algorithm to sort the restaurants with the most busy restaurant with the highest ranking.

6. Identify actual key risks for your project at this time:

Software Technical Ability

Broad - In a student-run team where we are required to make a full-stack web program, there might be issues in development where we aren't sure where to proceed in a specific part of our program due to a lack of expertise. This could be in the frontend or backend. This is probably more complicated to resolve up front as problems like these are encountered during development and can't necessarily be predicted, but resolving to ask for help when needed and *early* is important in combating the issue.

Insufficient Testing - If we don't do our due-diligence with testing we may find ourselves down the line with problems from merges many days prior that will hinder our future development. To counter this we will ensure that other members test each other's code to check for potential bugs.

Github Usage - Making sure that we manage our branches well is imperative, having too many branch conflicts can be a nightmare to resolve. Going over github practices and having the github person keep track of branches and merges goes a long way in mitigating this.

Program Requirements

Multiple Device Compatibility - Keeping track of how our website looks on multiple different browsers, in addition to a phone screen can be a daunting task. To combat this frequent check ins during development for how each browser/browser side window looks with each page is a pretty good first step in helping resolve the issue.

Security Vulnerabilities - Even though this website won't be really going live with sensitive individual information, having good security practices when managing backend data is imperative in creating good full stack services. Asking for help in this regard—someone to look over for vulnerabilities—might be a good step to take in order to shore up our software in this regard.

Team

Miscommunication/Poor communication - Our communication has been pretty good thus far, however that could change pretty rapidly depending on the situations presented to the team. Ensuring that we keep in contact every week on our specified meeting place—discord—and attending meetings will usually mean that this doesn't become an issue.

7. Project management:

As the team lead for our software engineering project, I have held regular team meetings every Friday to discuss project goals, expectations, and deadlines to ensure everyone is on the same page. During these meetings, I assign tasks and encourage team members to share their progress, concerns, and feedback so that we can identify potential issues early on and address them accordingly.

To help keep our team organized and focused, I create a google doc for every milestone that allows members to focus on the tasks, responsibilities, timelines, and resources needed to complete every milestone and future task. This google doc has been shared with all team members to ensure everyone understands their roles and responsibilities and working on them.

To monitor progress and track tasks, I will use a project management tool called Trello. This tool allows me to create boards for each project phase and assign tasks to team members with deadlines and priorities. It also enables me to monitor progress and identify potential delays.

In addition, I am also using communication tools such as Discord and Zoom to facilitate collaboration and communication among team members. These tools allow us to stay connected, share files and have real-time discussions about the project.

