SW Engineering CSC648/848 Spring 2023

Milestone 5

Final Project for SW Engineering Class CSC 648-848 Spring 2023

Food Delivery App: GatorGrub

Team 3

Shauhin Pourshayegan: spourshayegan@mail.sfsu.edu	Team Lead, Github Master, Document Editor
Hieu Ma: hma8@sfsu.edu	Front End Lead
Preetham Ramesh: pramesh1@sfsu.edu	Back End Lead
Lin Tun: ltun1@mail.sfsu.edu	Front End Developer
Derrick Liang: dliang5@mail.sfsu.edu	Back End Developer

URL: http://34.82.124.237:3001/

Date Submitted:	5/24/2023

2) Product summary:

I. Name of the product

GatorGrub

II. List of ALL major committed functions (Final P1 Funtions)

Unregistered users can

- Unregistered users shall be able to search by restaurant category.
- Unregistered users shall be able to search for food and restaurants.
- Unregistered users shall be able to search the restaurant map from their location.
- Unregistered users shall be able to sort restaurants by price.
- Unregistered users shall be able to select menu items and add menu items to their cart.
- Unregistered users shall be able to choose an order's delivery location.
- Unregistered users shall be able to read restaurant menus.
- Unregistered users shall be able to register an account with SFSU email.
- Unregistered users shall be able to register an account as a restaurant owner
- Unregistered users shall be able to register an account as a driver

SFSU students, faculty and staff can:

- SFSU Customers shall inherit all unregistered user functional requirements.
- SFSU Customers shall be able to log into their account.
- SFSU Customers shall be able to submit an order.

Restaurant owners can:

- Restaurant owners shall be able to register for an account.
- Restaurant owners shall be able to log into their account.
- Restaurant owners shall be able to upload menu item pictures and prices.
- Restaurant owners shall be able to submit restaurant details. (e.g. hours of operation, avg. order time.)
- Restaurant owners shall be able to choose to accept or reject an order.
- Restaurant owners shall confirm when orders are ready. This updates the order progress.

Drivers for GatorGrub can:

- Drivers shall be able to register as a driver.
- Drivers shall be able to log in as a driver.
- Drivers shall be able to accept or reject an order.

- Drivers shall confirm when food is picked up. This updates the order progress.
- Drivers shall confirm when food is delivered. This updates the order progress.
- Drivers shall be able to see the address of the pickup point.
- Drivers shall be able to see the address of the delivery point.

III. What is Unique in GatorGrub

GatorGrub is a web application that has been specifically designed to cater to the needs of students, faculty, and staff at SFSU. Our platform offers a convenient way for our customers to order snacks and food with just a few clicks. With our class to class delivery feature, our customers no longer have to worry about wasting time walking or commuting between classes, carrying heavy bags, or waiting in restaurants. Instead, they can simply place an order on GatorGrub and have it delivered directly to their next class location. This ensures that our customers can enjoy their meals without any hassle and without the fear of getting late for their classes.

IV. URL to GatorGrub

Deployment Server: Google Compute Engine

URL: http://34.82.124.237:3001/

3) Milestone documents – M1-M4

Contains files for M1, M2, M3 feedback summary report, and M4 (M4 is submitted here for the first time). Simply concatenate individual M1-M4 files starting from M1.

SW Engineering CSC648-848 Spring 2023

GatorGrub

Team 3

Shauhin Pourshayegan:	Team Lead, Github Master, Document Editor
Hieu Ma:	Front End Lead
Preetham Ramesh:	Back End Lead
Lin Tun:	Front End Developer
Derrick Liang:	Back End Developer

Milestone 1

Date Submitted:	3/13/2023
Date Revised:	3/20/2023

1. Executive Summary:

People at universities often find themselves stuck when searching for their next meal. Food options on campus are likely few and far between, and have the potential to taste like the ever-dreaded dining hall food. Students and instructors alike find issues managing their food habits when heavy workloads and packed schedules take precedent. As students of San Francisco State University, we find ourselves trekking back and forth to Stonestown Galleria just so we can get access to more options, wasting our time when it's most valuable to us. Other than the inconvenience of food locations and availability, the food available is not always affordable, which is a major issue for students and important for everyone. The inconvenience of our current options and the lack of affordable options are the primary driving forces of our team project.

As students, we wanted to provide a solution to these issues when we came to discuss what GatorGrub will be. We propose a service which will strengthen the connection between SFSU students, faculty and staff to their local food scene, as well as offer this primary user base with features which make their lives easier and solve the problems related to getting food on campus. GatorGrub considers the needs of our local restaurants by helping them reach their customers and persuade customers on the edge with exclusive deals on their next meal. GatorGrub aims to provide these users with a selection of local restaurants offering delivery.

Customers who don't have the time to leave their classroom or study area can get their food delivered to any room on campus, allowing their workshops and group study sessions to be uninterrupted by rumbling tummies. Customers can also decide whether they want their food delivered to one of several safe pickup spots throughout SFSU's campus. This feature benefits students who want to get food safely, without worrying about who they will encounter on the long walk to Stonestown or further out. GatorGrub also enables its users to get deals on different local restaurants. This will incentivise more sales, making restaurant owners and delivery drivers content and strengthening GatorGrub's infrastructure.

Our team is dedicated to this project, because of the convenience it could bring directly to us and our community. We are well structured, with a Team Lead at the top to allocate work among the section leads and overlook the entire project. The Back End Lead is responsible for developing the back end with the help of one Back End Developer. Similarly, the Front End Lead is responsible for developing the front end with the help of one Front End Developer. We're concerned with the issues related to student health and happiness, and we know that food is one of the factors which can strongly affect it. We think GatorGrub is the next step forward for the SFSU community.

2. Personae and Use Cases:

Personae:

SFSU Student:

About Kate:

- 20 y/o SFSU Junior
- Lives on campus
- Grocery store employee
- Hobbies: Digital art, Hiking, Anime

Kate is a serious student and an equally serious eater. She spends most of her day chasing deadlines and working on homework assignments, unless she's in class or back at her dorm to rest. Kate also works part-time at Trader Joes in Stonestown to make sure she can support her living expenses. She already has tuition and her dorm paid for, but everything else like the internet, her phone plan and food is her responsibility, so she has to be mindful with the money she spends.

Goals:

- Create an optimal study environment, with food easily and quickly accessible.
- Find ways to save on everyday expenses like food.

Frustrations:

When Kate's trying to get a project done, food is the last thing she wants to think about. However, she knows that if she isn't satisfied, then she can't perform as well in the classroom or at her desk studying. As a student she also doesn't have enough money to pay for food delivery every time she has a serious due date coming up.

Scenario:

One night Kate has a presentation she has to prepare for the following day's class. It's getting late and she needs something to eat for dinner. Before she could leave the

building, her friend, who lives two doors down, calls her name. She told Kate about GatorGrub, a new food ordering and delivery app targeted at the SFSU community. She recommended it because it offers students a cheap and easy way to access food.

Kate had planned to study that evening and prepare instant ramen if she really got hungry, but after checking out GatorGrub, she changed her mind. She was able to order her food to a safe location, which was convenient enough to pick up without wasting too much study time.

SFSU Faculty:

About Jane:

- Age: 30
- Location: Works on campus
- Profession: Works full time as a professor in SFSU
- Hobbies: Reading, growing plants

Jane feels inspired when she's able to teach her students something new. She teaches three sections of an introductory course for her field, because she wants to give new students a solid foundation they can build up from. On top of this she's also managing a graduate level research project which requires its own fair share of attention, both for her students and her personal research goals. Managing lectures, class notes and project deadlines sometimes puts an overwhelming burden on Jane.

Goals:

- To better her craft as an instructor
- To improve her workflow so she doesn't have to work overtime

Frustrations:

Jane struggles with time management and grading, so she wants to spend as much time as she can on grading assignments and responding to emails while she's on campus.

This leaves her without time for herself and her own needs such as lunchtime, where she only has a fifteen minute break to get fed before her next section begins.

Scenario:

Jane, with the help of the SFSU food delivery application, she can now order food from the SFSU campus/sfsu approved restaurants and have it delivered to her building and room number. So she no longer has to spend time walking to an eatery and ordering.

SFSU Staff:

About Mary:

- Age: 34
- Location: Works on campus
- Profession: Academic Adviser
- Hobbies: Cooking, hiking, watching sports

Mary is an academic adviser for students at SF State. She's great at her job because she loves interacting with all the professors in the different colleges and hearing about their studies. Mary is always helping students figure out which path is right for them and often finds herself going over her time allowance to make sure the students get the support they need.

Goals:

- Eat healthy
- Maintain a specific diet
- Help her advisees succeed in their chosen path.

Frustrations:

Oftentimes Mary has a hard time finding food that fits her diet and can't spend time commuting to the ones that do while working. She drinks coffee to keep her up in the morning, but if she can't get some real food in her system by 2:00 PM, she's going to

crash. This means she has trouble focusing on her students and makes her less enthusiastic to hear about the interesting studies of each college.

Scenario:

Long days in the office are typically paired with coffee, because no one has the time to step out for too long and pick up a meal. Mary finds herself with the dilemma of needing to decide whether to help students or take her lunch break. Her coworker Roy tells her about a new food ordering app some students came up with which benefits SFSU staff members too. With GatorGrub, Mary is able to find food more easily and it's deliverable to her work location.

GatorGrub Admin:

About Emma:

- Age: 23
- Location: Works remotely at home in Milpitas
- Profession: Graduate student in CS
- Hobbies: Lap swimming, brunch with friends, board games

Emma has already completed her bachelors and is pursuing a higher degree at San Jose State University. Emma's lifestyle is fairly sedentary while she's in between jobs, affording her consistency in her daily schedule. To get out of the house, she often travels to different universities and meets new people who work in her field, as she's devoted to her studies, and is always hearing about new projects popping up. This is especially true, because Emma has been going to a lot of job fairs at these various universities she visits. Emma knows a couple of friends that are always working on projects and she helps them network.

Goals:

- Supporting herself while pursuing her graduate degree.
- Finding an easy gig that can make her some money.

- Doing something that aligns with the career path she's chosen for herself.

Frustrations:

Emma finds difficulty in finding jobs around where she lives, since her options typically expect more time than she's able to give, considering the time she needs to spend studying for her upcoming degree.

Scenario:

Emma takes up an offer from her friends at SF State who are developing a new food ordering service. Other than the school and extracurriculars, Emma's schedule is fairly open for new job opportunities. Something that requires little time, but also allows her to practice her skills and be productive with the rest of her time.She's been told that all she has to do is run the back end restaurant verification process. The steps were simply explained to her and she just had to confirm the business actually exists and that the menu items provided resemble the ones from the menu of the location. After she's satisfied, she can give the green light for new sat

GatorGrub Driver:

About Jared:

- Age: 22
- Location: Lives on campus
- Profession: Graduate Student
- Hobbies: Online video games, Fixie bicycles, collecting funko pops

Jared works hard and plays hard. His primary focus is his journalism degree and internship. He devotes most of his time to these two, because he sees himself as a successful entertainment journalist in the future. He's fueled by his interests in gaming, pop culture and entertainment, inspiring him to push his journalism career towards a focus on the gaming and related tech industry. Sometimes he overspends on his hobbies, as fixie parts and funko pop figures are expensive.

Goals:

- To graduate with his degree in Journalism
- To be healthier

Frustrations:

- Jared lives off campus in an apartment in San Francisco. With rent and general QOL being so high, he needs an extra source of income to make up the difference. However, his schedule is unpredictable and he cannot commit to a fixed shift-schedule. As a result, he has found it hard to become a part-time employee.

Scenario:

- By becoming a delivery driver for GatorGrub, Jared can work whenever he has time and doesn't need to commute long distances to an office building. He can use his bike to bypass rush hour traffic and even achieve his fitness goals on the many hills of San Francisco.

Restaurant owner/manager:

About Marcus:

- Age: 47
- Location: Restaurant is near SFSU
- Profession: Restaurant Owner
- Hobbies: Cooking, Swimming, Reading

Marcus has over 30 years of restaurant experience. He started cooking with his parents at age 9 and officially started working for his dad's restaurant at age 16. Starting from the kitchen, he learned management skills in college and from his father. After acquiring his father's business and helping him retire, Marcus developed his restaurant and got it to function as efficiently as a well-oiled machine. His devotion to food made him want to step away from the restaurant and run a food truck that visits SFSU campus, where he can be the chef.

Goals:

- To be able to increase business sales and profits
- To give more exposure to their restaurant
- To create food people enjoy

Frustrations:

- Since there are hundreds of restaurants all over the city of San Francisco, it is hard for Marcus to gain more exposure to his restaurant. He can't park his food truck in a spot that's easily noticeable to most students. Marcus wants a new way to gain and retain customers while on campus.

Scenario:

- One day, Marcus hears news of a new app called GatorGrub for food ordering services. He decides to take a look at what features and benefits there are. He sees that he can register his restaurant to be mainly focused on catering to people from SFSU. He determines that this could be a great business opportunity to gain more traction to his restaurant.

Use cases:

Study session safe pickup:

1. Kate (**SFSU student**) wanted to grab a quick bite to eat without interrupting her study time. She turned to GatorGrub, a food ordering, delivery, and pickup service, to help her find a convenient solution. With GatorGrub, **Kate was able to search for nearby restaurants on campus, narrowed down by cuisine and price. She then placed her order online, selecting a safe pickup location close to her dorms.** This not only saved her time and money but also provided peace of mind when picking up her food. By using GatorGrub's safe pickup location on campus, Kate was able to quickly and easily get her food without having to leave the university or worry about her safety. She could then return to her studies feeling satisfied and focused.

Classroom delivery with lazy registration:

2. Long days answering student questions can make managing Jane's (**SFSU faculty/staff**) lunch difficult. It makes it easy when her food can be delivered straight to her. When she wants something for lunch that's easy to manage, Jane opens up GatorGrub. **She used GatorGrub's search feature to filter by location, cuisine type, price range, and other preferences. Once she found a restaurant that she liked, she could view the menu, prices, and reviews to decide on her order. After searching for and selecting the food she wants for lunch, she can log in or register using her university email (lazy registration). Then she's able to set up the delivery location to be in her classroom, by selecting a building and room number, which she has to teach in for the next two and a half hours.** After placing her order, she's able to wait for it to be delivered straight to her. With lunch out of the way and her stomach satisfied, Jane can go back to enjoying teaching class.

Grad starts delivery grind:

3. Jared (**GatorGrub driver**) is looking for additional ways to make money while still being able to manage his classes. He realizes that he can leverage his access to different modes of transportation, depending on his class schedule, to earn extra cash. To do so, he decides to register as a driver for GatorGrub, a food delivery service on campus. One of the benefits of being a GatorGrub driver is that Jared has complete control over the orders he accepts. **He can choose which orders to accept based on his availability, preferred transportation mode, and other factors. If he sees an order that he can complete, he can accept it and begin the delivery process. When Jared accepts an order, he will receive all the necessary information about the order, including the restaurant from which he needs to pick up the food and the delivery address.** He can then use his transportation method of choice to complete the delivery quickly and efficiently. By becoming a driver for GatorGrub, Jared is able to earn extra money while maintaining the flexibility he needs to balance his studies and other commitments.

Restaurant owner reaches new customers:

4. Marcus (**Restaurant owner**) is looking for ways to increase the exposure of his restaurant and attract more customers. He realizes that by partnering with GatorGrub, he can cater to a specific group of customers: the students and staff at San Francisco State University. By registering his restaurant with GatorGrub, Marcus can make his food easily accessible to this group of people without having to compete against hundreds of other restaurants on other food service apps. **He will need to provide details about his restaurant, such as the menu pdf, selecting operating hours, and selecting location. To register his restaurant with GatorGrub, Marcus needs to create a restaurant owner account. Once he has completed the registration process, Marcus will be informed that it may take up to 24 hours for his restaurant registration to be approved.** Once his restaurant is approved, Marcus can start offering his menu to the SFSU community through GatorGrub. He can take advantage of the platform's features, such as promotion codes and targeted marketing campaigns, to attract new customers and increase sales. With GatorGrub, Marcus can easily manage orders and deliveries, making the food delivery experience convenient for both himself and his customers.

Admin's daily routine:

5. Emma works as a **GatorGrub admin**, so she knows the daily ins and outs of running the restaurant verification process. When she wakes up she can check the newly submitted applications on her laptop. One by one, **each submission is verified to see if the restaurant is real and if the food on the submitted menu is similar enough to the food served by the restaurant in question. Certain details should be directly matching, like the address and phone number of the restaurant, while menu items and prices may be different due to the restaurant's policy on take-out and delivery.** After she processes the restaurant, a notification is sent to the **restaurant owner**, letting them know the result of their restaurant's review.

3. Data Glossary:

List of main data items and entities:

- SFSUCustomer Entity:
    - SFSUCustomer ID Number (PKey)
    - SFSUCustomerName
    - SFSUCustomerEmail
    - SFSUCustomerPhone Number
    - SFSUCustomerPassword
- Driver Entity:
    - Driver ID Number (PK)
    - DriverName

    - DriverEmail
    - DriverPhone Number
    - DriverPassword
- RestaurantOwner Entity:
    - RestaurantOwner ID Number (PK)
    - RestaurantOwnerName
    - RestaurantOwnerEmail
    - RestaurantOwnerPhone Number
    - RestaurantOwnerPassword
- Restaurant Entity
    - Restaurant ID Number (PK)
    - RestaurantOwner ID Number (FK)
    - RestaurantName
    - RestaurantEmail
    - RestaurantPhone Number
    - RestaurantPassword
    - RestaurantAddress
    - RestaurantCuisineType
    - RestaurantPriceTier (updated by trigger)
    - RestaurantHours (Hours of operation, specifically for pick-up/delivery)
    - DeliveryTime (time for order to get to door)
- Restaurant Menu Entity
    - Restaurant Menu ID (PK)
    - Restaurant ID (FK)
- Menu Item Entity
    - Menu Item ID (PK)
    - Menu ID (FK)
    - Menu Item Name
    - Menu Item Picture
    - Menu Item Price
    - Menu Item Description
- Order Entity
    - Order ID Number (PK)
    - Customer ID Number (FK)
    - Driver ID Number (FK)
    - Restaurant ID Number (FK)
    - OrderTime
    - DeliveryTime (OrderTime + RestaurantPrepTime + Google Maps Journey Time)
    - DeliveryLocation
        - Shall be either a desired address, building and room, or a “safe pickup spot”

    - OrderSelections (CSV, menu item: ID, quantity, notes)
    - OrderNotes
    - OrderDiscounts
    - OrderPrice
- Review Entity
    - Review ID (PK)
    - Restaurant ID (FK)
    - User ID (FK)
    - Message
    - Rating Number

4. Functional Requirements:

Unregistered Users:

1. Unregistered users shall be able to search for food and restaurants.
2. Unregistered users shall be able to search the restaurant map from their location.
3. Unregistered users shall be able to read restaurant menus.
4. Unregistered users shall be able to read restaurant reviews.
5. Unregistered users shall be able to select menu items and add menu items to their cart.
6. Unregistered users shall be able to choose an order's delivery location.

SFSU Users:

7. SFSU Users shall inherit all unregistered user functional requirements.
8. SFSU Users shall register their account with an SFSU email.
9. SFSU Users shall be able to submit an order.
10. SFSU Users shall be able to see their order's delivery status.
11. SFSU Users shall be able to write feedback on the order.
12. SFSU Users shall be able to upload pictures.
13. SFSU Users shall be able to contact drivers.
14. SFSU Users shall be able to send and receive messages to customer service.

Admins:

15. Admins shall be required to approve all restaurant applications before they go live on the app.
16. Admins shall review restaurant applications and provide the appropriate response.
17. Admins shall be able to view the user account.
18. Admins shall be able to warn users and restrict orders from the user account, if a user violates GatorGrub policy with the account.

Restaurant Owners:

19. Restaurant owners shall be able to upload menu item pictures and prices.
20. Restaurant owners shall be able to submit restaurant details. (e.g. hours of operation, avg. order time.)
21. Restaurant owners shall be able to respond to the feedback of their restaurant.
22. Restaurant owners shall be able to choose to accept or reject an order.
23. Restaurant owners shall be able to contact their customers and drivers.
24. Restaurant owners shall confirm when orders are ready. This updates the order progress.
25. Restaurant owners shall be able to send and receive messages to customer service.

Driver Users:

26. Driver users shall be able to accept or reject an order.
27. Driver users shall confirm when food is picked up. This updates the order progress.
28. Driver users shall confirm when food is delivered. This updates the order progress.
29. Driver users shall be able to see a map of the pickup point.
30. Driver users shall be able to see a map of the delivery point.
31. Driver users shall be able to contact customers.
32. Driver users shall be able to send and receive messages to customer service.

5. Non-functional Requirements:

List of non-functional requirements (performance, expected load, security requirements, storage, availability, fault tolerance...):

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0
2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers
3. All or selected application functions shall render well on mobile devices
4. Data shall be stored in the database on the team’s deployment server.
5. No more than 50 concurrent users shall be accessing the application at any time
6. Privacy of users shall be protected
7. The language used shall be English (no localization needed)
8. Application shall be very easy to use and intuitive
9. Application shall follow established architecture patterns
10. Application code and its repository shall be easy to inspect and maintain
11. Google analytics shall be used
12. No email clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application

13. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
14. Site security: basic best practices shall be applied (as covered in the class) for main data items
15. Media formats shall be standard as used in the market today
16. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development
17. The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2023. For Demonstration Only” at the top of the WWW page nav bar. (Important so as to not confuse this with a real application).

6. Competitive Analysis:

Analysis of features across competition:	**Ez Cater**	**Grubhub**	**Uber Eat**	**DoorDash**	**GatorGrub**
Easy Map Direction	✘	✘	✔	✔	✔
Class to Class Delivery	✘	✘	✘	✘	✔
University Student / Faculty Discount	✘	✘	✘	✘	✔
Safe Pickup Spot	✘	✘	✘	✘	✔

✔ = has feature, ✘ = missing feature

GatorGrub, our delivery app, will provide a variety of benefits to SFSU students, faculty and staff members. For starters, it will offer SFSU students, faculty and staff members exclusive discounts on a variety of products and services. As a result, our student customers will be able to save money while also having their purchases delivered directly to their classroom. Faculty and staff can save time on their lunch breaks and make the most of their time. This incentivises more sales, satisfying both restaurant owners and GatorGrub drivers looking for jobs.

Another benefit of GatorGrub is that it will provide class-to-class delivery. This feature will be especially useful for our students and faculty customers who have classes in various buildings or locations on campus. Our customers will no longer have to waste time walking or commuting between classes, nor will they have to worry about carrying heavy bags or waiting in restaurants. With GatorGrub, our customers can simply place an order and have it delivered directly to their next class location.

Additionally, patrons can get their food delivered to a safe pickup spot on campus. For students and staff who want to save money on delivery and are willing to take a little walk within campus grounds, safe pickup locations are available in well traveled and secure parts of campus so people can get their food at a more reasonable price, with peace of mind knowing they'll be safe. (Potential locations: Caesar Chavez Student Center, Mashouf Wellness Center, Village Plaza (Dorms), UPN (near Stonestown), etc.)

7. High-level system architecture and technologies used:

- Server Host, Instance size (CPU and RAM):
    - Google Compute Engine, 1vCPU, 2 GB RAM
- Operating System and Version Number:
    - AlmaLinux version 9.1
- Database and Version Number:
    - MySQL version 8.0.32
- Web Server and Version Number:
    - Express.js version 4.18.2
- Server-Side language and Version Number:
    - Node.js version 18.14.1
- API's:
    - Google Analytics Reporting API v4
    - Google Maps JavaScript API
    - Google Routes API

Search functionality implementation:

Our search functionality will be fairly simple. Our database will store each restaurant along with metadata such as location, cuisine, price information, restaurant name, and price tier. This way we can organize restaurants by categories for users to browse. When a user wants to search using an input field, we will take the input, sanitize it and plug it into SQL queries for retrieval. We could use the LIKE function to perform a fuzzy search. We can implement indexing and implement pagination to optimize search functions.

8. Team and Roles:

Shauhin Pourshayegan	Team Lead, Github Master
Hieu Ma	Front End Lead
Preetham Ramesh	Back End Lead
Lin Tun	Front End Developer
Derrick Liang	Back End Developer

Work distribution:

- M0 edits: Hieu
- Executive summary: Shauhin
- Personas & use cases: Shauhin, Derrick, Hieu, Preetham, Lin
- Data glossary & search functionality: Preetham, Derrick
- Competitive analysis: Lin

9. Team Lead Checklist:

- ☑ So far all team members are engaged and attending team sessions when required
- ☑ Team found a time slot to meet outside of the class
- ☑ Back end, Front end leads and Github master chosen
- ☑ Team ready and able to use the chosen back and front end frameworks and those who need to learn are working on learning and practicing
- ☑ Team lead ensured that all team members read the final M1 and agree/understand it before submission
- ☑ Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)
- ☑ NEW: Use of any GenAI tool like ChatGPT: say if you used ChatGPT or like and how and for what segment of Milestone 1 (brief paragraph). As per class policy: this is allowed as a help BUT you cannot copy and paste its output and claim it is your own text, you need to put it in quotes or modify it, and only for short sentences You also are responsible for accuracy of your submission, so any ChatGPT content needs to be checked by you.

ChatGPT was considered for certain creative choices for our application, however after discussion and personal testing, we've decided as a group we don't want to use it yet. We reserve the option to use it further along in our project, but for now it has not been used.

SW Engineering CSC648-848 Spring 2023

GatorGrub

Team 3

Shauhin Pourshayegan: spourshayegan@mail.sfsu.edu	Team Lead, Github Master, Document Editor
Hieu Ma:	Front End Lead
Preetham Ramesh:	Back End Lead
Lin Tun:	Front End Developer
Derrick Liang:	Back End Developer

Milestone 2

Date Submitted:	3/31/2023
Date Revised:	4/6/2023

1. Executive Summary:

People at universities often find themselves stuck when searching for their next meal. Food options on campus are likely few and far between, and have the potential to taste like the ever-dreaded dining hall food. Students and instructors alike find issues managing their food habits when heavy workloads and packed schedules take precedent. As students of San Francisco State University, we find ourselves trekking back and forth to Stonestown Galleria just so we can get access to more options, wasting our time when it's most valuable to us. Other than the inconvenience of food locations and availability, the food available is not always affordable, which is a major issue for students and important for everyone. The inconvenience of our current options and the lack of affordable options are the primary driving forces of our team project.

Being students, we wanted to provide a solution to these issues when we came to discuss what GatorGrub will be. We propose a service which will strengthen the connection between SFSU students, faculty and staff to their local food scene, as well as offer this primary user base with features which make their lives easier and solve the problems related to getting food on campus. GatorGrub considers the needs of our local restaurants by helping them reach their customers and persuade customers on the edge with exclusive deals on their next meal. GatorGrub aims to provide these users with a selection of local restaurants offering delivery.

Customers who don't have the time to leave their classroom or study area can get their food delivered to any room on campus, allowing their workshops and group study sessions to be uninterrupted by rumbling tummies. Customers can also decide whether they want their food delivered to one of several safe pickup spots throughout SFSU's campus. This feature benefits students who want to get food safely, without worrying about who they will encounter on the long walk to Stonestown or further out. GatorGrub also enables its users to get deals on different local restaurants. This will incentivise more sales, making restaurant owners and delivery drivers content and strengthening GatorGrub's infrastructure.

Our team is dedicated to this project, because of the convenience it could bring directly to us and our community. We are well structured, with a Team Lead at the top to allocate work among the section leads and overlook the entire project. The Back End Lead is responsible for developing the back end with the help of one Back End Developer. Similarly, the Front End Lead is responsible for developing the front end with the help of one Front End Developer. We're concerned with the issues related to student health and happiness, and we know that food is one of the factors which can strongly affect it. We think GatorGrub is the next step forward for the SFSU community.

2. Data Glossary:

List of main data items and entities:

- UnregisteredUser (Does not require any data items other than temporary session and order information. Therefore this information will be stored in cookies)
    - SessionID(PK)
    - OrderID(FK)
- SFSUCustomer:
    - SFSUCustomerID (PK)
    - SFSUCustomerName
    - SFSUCustomerEmail
    - SFSUCustomerPhone
    - SFSUCustomerPassword
- Driver:
    - DriverID (PK)
    - DriverName
    - DriverEmail
    - DriverPhone
    - DriverPassword
- RestaurantOwner:
    - RestaurantOwnerID (PK)
    - RestaurantOwnerName
    - RestaurantOwnerEmail
    - RestaurantOwnerPhone
    - RestaurantOwnerPassword
- Restaurant:
    - RestaurantID (PK)
    - RestaurantOwnerID (FK)
    - RestaurantRegistered (boolean for admin review)
    - RestaurantName
    - RestaurantPhone
    - RestaurantPassword
    - RestaurantAddress
    - RestaurantCuisine
    - RestaurantPriceTier (FK)
    - RestaurantHours (Hours of operation, specifically for pick-up/delivery)
    - PrepTime (Avg. time for orders ready for takeout)
    - RestaurantImage
- RestaurantMenu:
    - RestaurantMenuID (PK)
    - RestaurantID (FK)

- MenuItem:
    - MenuItemID (PK)
    - RestaurantMenuID (FK)
    - MenuItemName
    - MenuItemPicture
    - MenuItemPrice
    - MenuItemType (e.g. appetizer, entrée, sides, dessert, drinks)
    - MenuItemDescription
- Order:
    - OrderID (PK)
    - CustomerID (FK)
    - DriverID (FK)
    - RestaurantID (FK)
    - OrderTime
    - DeliveryTime (OrderTime + RestaurantPrepTime + Google Maps Journey Time)
    - DeliveryAddress (only customer's address)
    - DeliveryBuilding
    - DeliveryRoom
    - PickupSpotID (includes restaurant location and safe pickup spots)
    - OrderSelections (CSV: MenuItemID, quantity, notes)
    - OrderNotes
    - OrderDiscounts
    - OrderPrice
- Review (Priority 2):
    - ReviewID (PK)
    - RestaurantID (FK)
    - UserID (FK)
    - Message
    - Rating

3. Functional Requirements Prioritized:

- **Priority 1**
    - Unregistered Users:
        - Unregistered users shall be able to search by restaurant category.
        - Unregistered users shall be able to search for food and restaurants.
        - Unregistered users shall be able to search the restaurant map from their location.

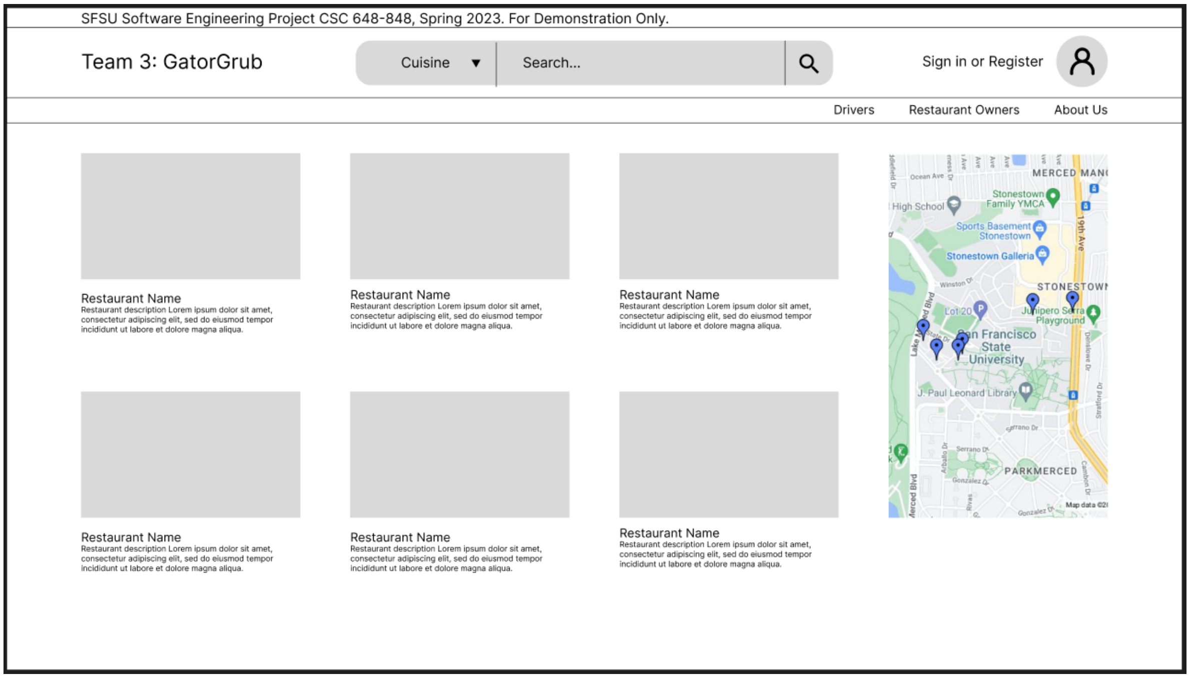
        - Unregistered users shall be able to sort restaurants by delivery time.
        - Unregistered users shall be able to sort restaurants by price.
        - Unregistered users shall be able to select menu items and add menu items to their cart.
        - Unregistered users shall be able to choose an order's delivery location.
        - Unregistered users shall be able to read restaurant menus.
        - Unregistered users shall be able to register an account with SFSU email.
        - Unregistered users shall be able to register an account as a restaurant owner
        - Unregistered users shall be able to register an account as a driver
    - SFSU Customers:
        - SFSU Customers shall inherit all unregistered user functional requirements.
        - SFSU Customers shall be able to log into their account.
        - SFSU Customers shall be able to submit an order.
    - Admins:
        - Admins shall be required to approve all restaurant applications before they go live on the app.
        - Admins shall review restaurant applications and provide the appropriate response.
        - Admins shall be able to view the user account.
        - Admins shall be able to warn users and restrict orders from the user account, if a user violates GatorGrub policy with the account.
        - Admins shall be able to remove users.
        - Admins shall be able to remove restaurants.
    - Restaurant Owners:
        - Restaurant owners shall be able to register for an account.
        - Restaurant owners shall be able to log into their account.
        - Restaurant owners shall be able to upload menu item pictures and prices.
        - Restaurant owners shall be able to submit restaurant details. (e.g. hours of operation, avg. order time.)
        - Restaurant owners shall be able to choose to accept or reject an order.
        - Restaurant owners shall confirm when orders are ready. This updates the order progress.
        - Restaurant owners shall be able to review and manage driver tasks
    - Drivers:
        - Drivers shall be able to register as a driver.
        - Drivers shall be able to log in as a driver.

        - Drivers shall be able to accept or reject an order.
        - Drivers shall confirm when food is picked up. This updates the order progress.
        - Drivers shall confirm when food is delivered. This updates the order progress.
        - Drivers shall be able to see a map of the pickup point.
        - Drivers shall be able to see a map of the delivery point.
- **Priority 2**
    - SFSU Customers:
        - SFSU Customers shall be able to write feedback on the order.
        - SFSU Customers shall be able to send and receive messages to customer service.
        - SFSU Customers shall be able to see their order's delivery status.
    - Restaurant Owners:
        - Restaurant owners shall be able to respond to the feedback of their restaurant.
        - Restaurant owners shall be able to send and receive messages to customer service.
- **Priority 3**
    - Unregistered Users:
        - Unregistered users shall be able to read restaurant reviews.
    - SFSU Customers:
        - SFSU Customers shall be able to contact drivers.
    - Restaurant Owners:
        - Restaurant owners shall be able to contact their customers and drivers.
    - Drivers:
        - Drivers shall be able to send and receive messages to customer service.

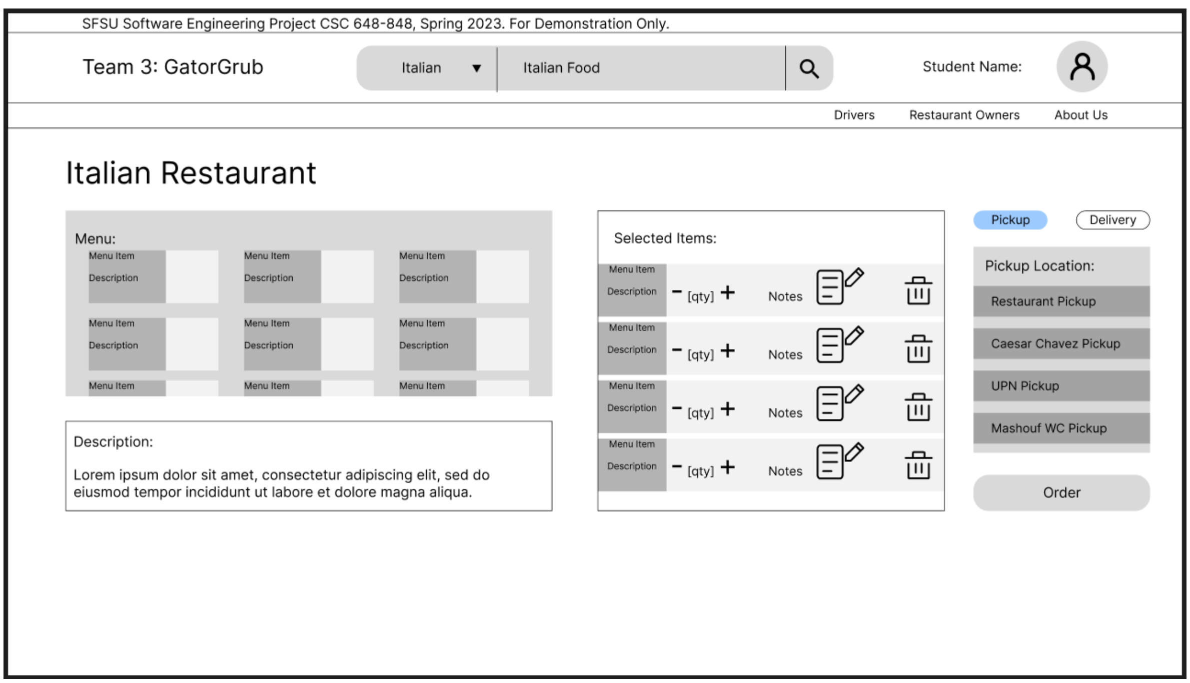
4. UI Storyboards Per Use Case:

Case 1: Kate (**SFSU student**) wanted to grab a quick bite to eat without interrupting her study time. She turned to GatorGrub, a food ordering, delivery, and pickup service, to help her find a convenient solution.

1. Kate opens the site and sees home page:

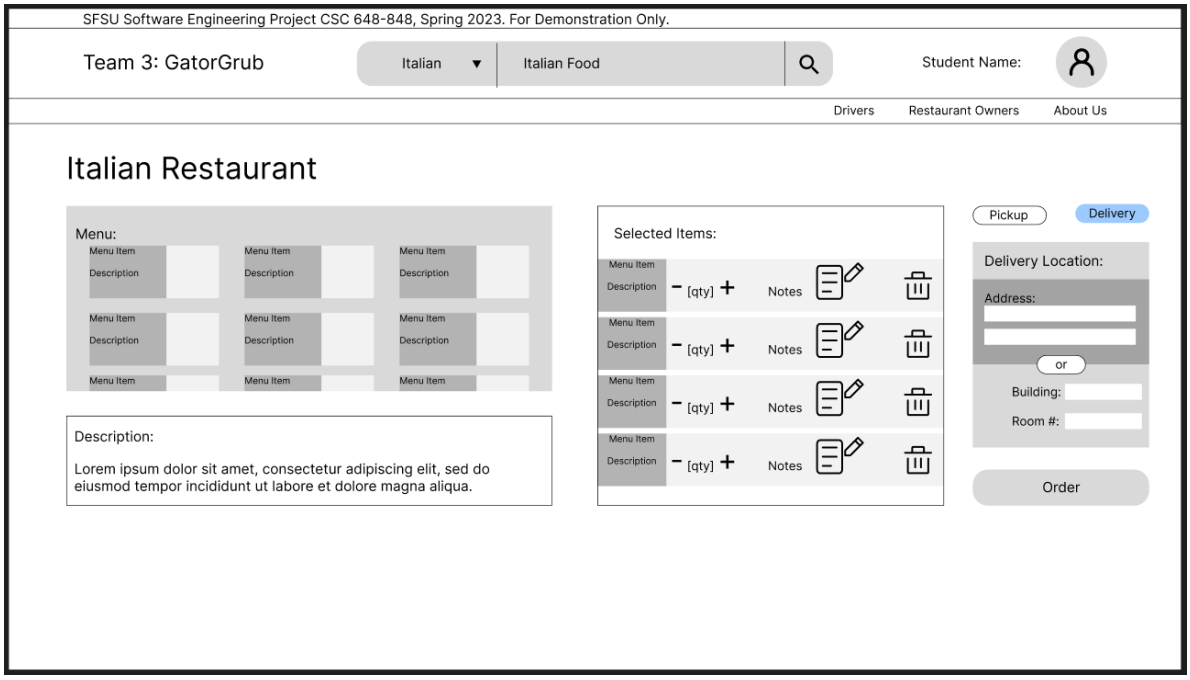


2. Kate then clicks on a restaurant, fills in her order, selects a pickup location, then clicks order to finish.

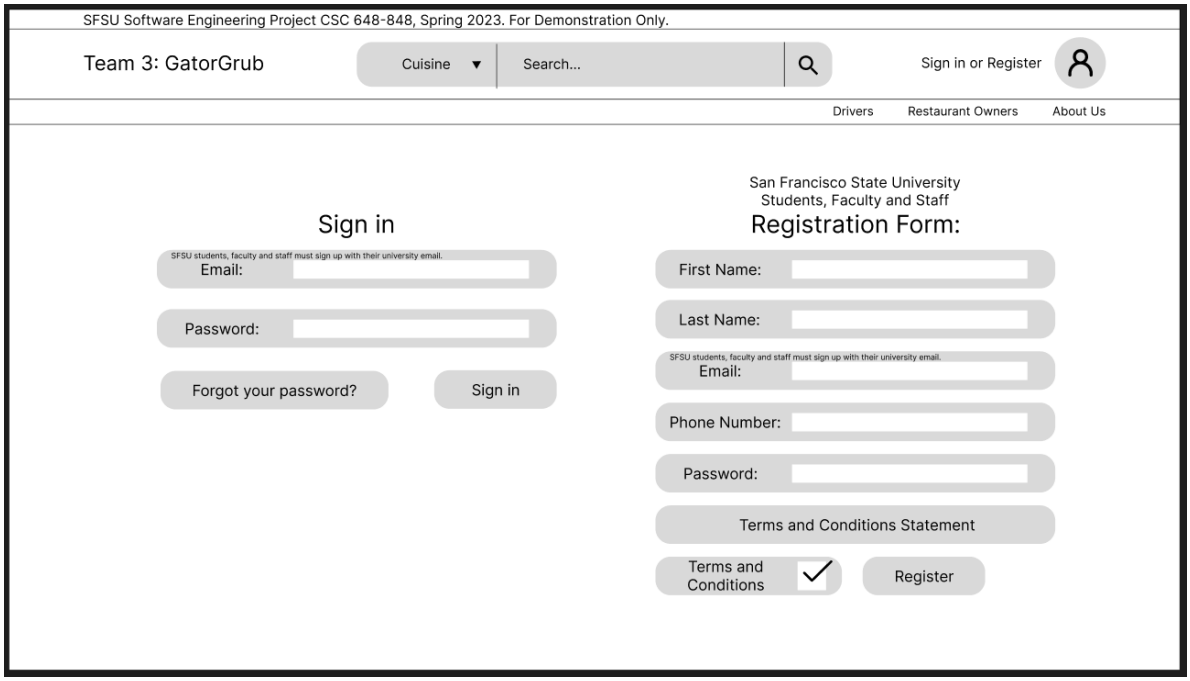


Case 2: Long days answering student questions can make managing Jane's (**SFSU faculty/staff**) lunch difficult. It makes it easy when her food can be delivered straight to her. When she wants something for lunch that's easy to manage, Jane opens up GatorGrub.

1. After Jane selects a restaurant, she fills in her order, selects a location for delivery, and then clicks order to finish.



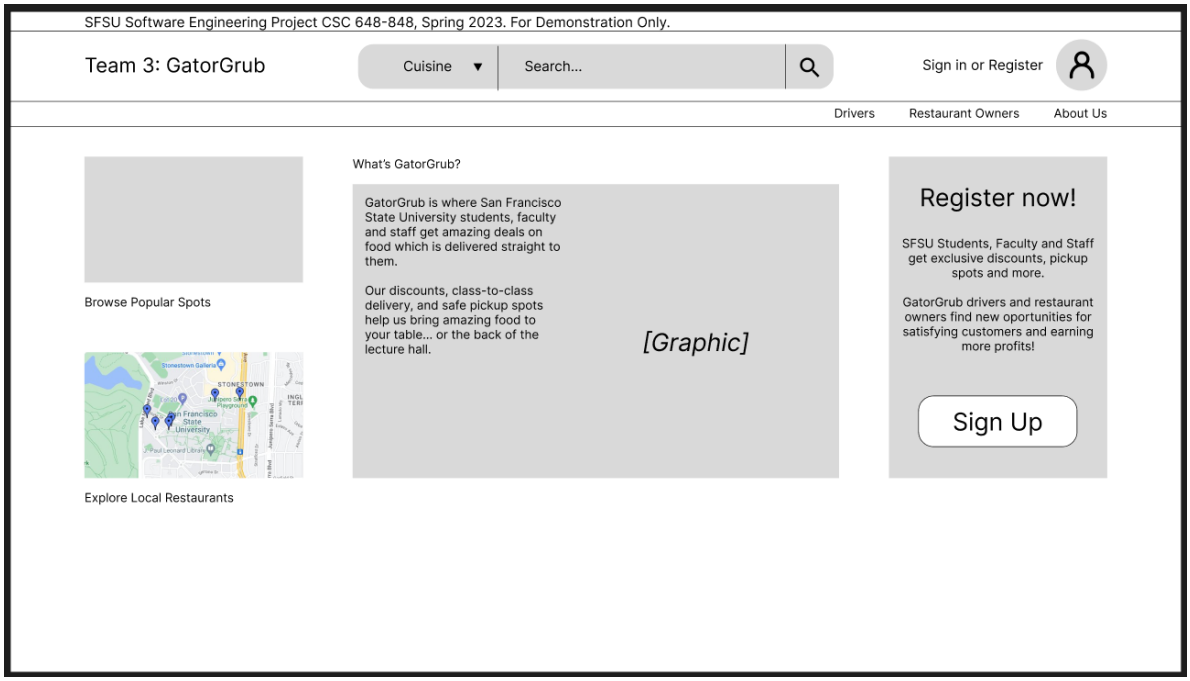
2. After Jane submits her order she's prompted to either sign in or register her account, so the order can be made from an SFSU customer.



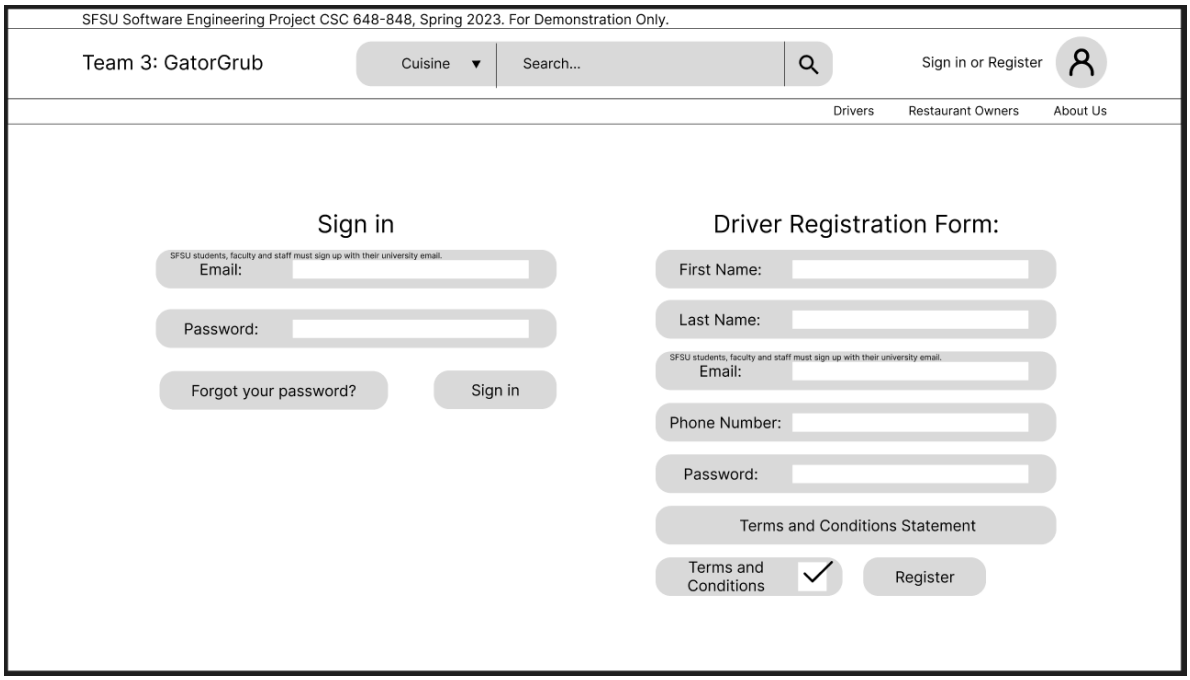
Case: 3 Jared (**Driver**) is looking for additional ways to make money while still being able to manage his classes. He realizes that he can leverage his access to different modes of transportation, depending on his class schedule, to earn extra cash. To do so, he decides

to register as a driver for GatorGrub, a food delivery service on campus. One of the benefits of being a GatorGrub driver is that Jared has complete control over the orders he accepts.

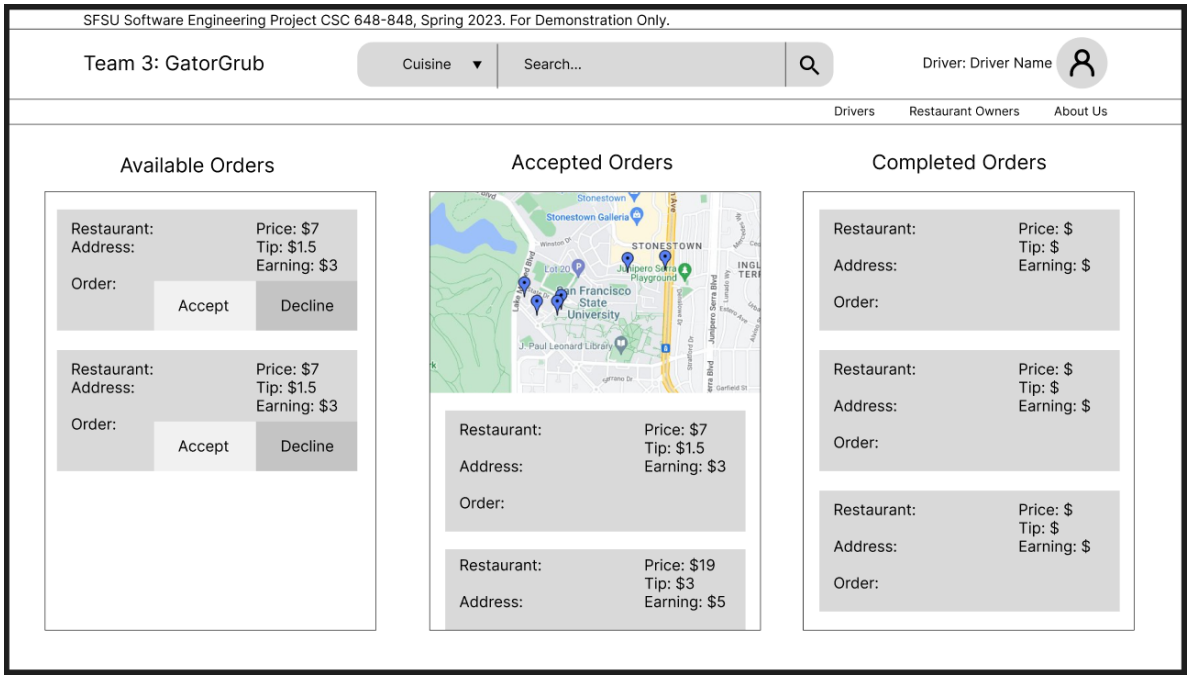
1. Jared visits the homepage to start registering as a driver. Jared clicks the sign up option.



2. The driver registration form appears and now after going through the process of registering, Jared now can take delivery requests.

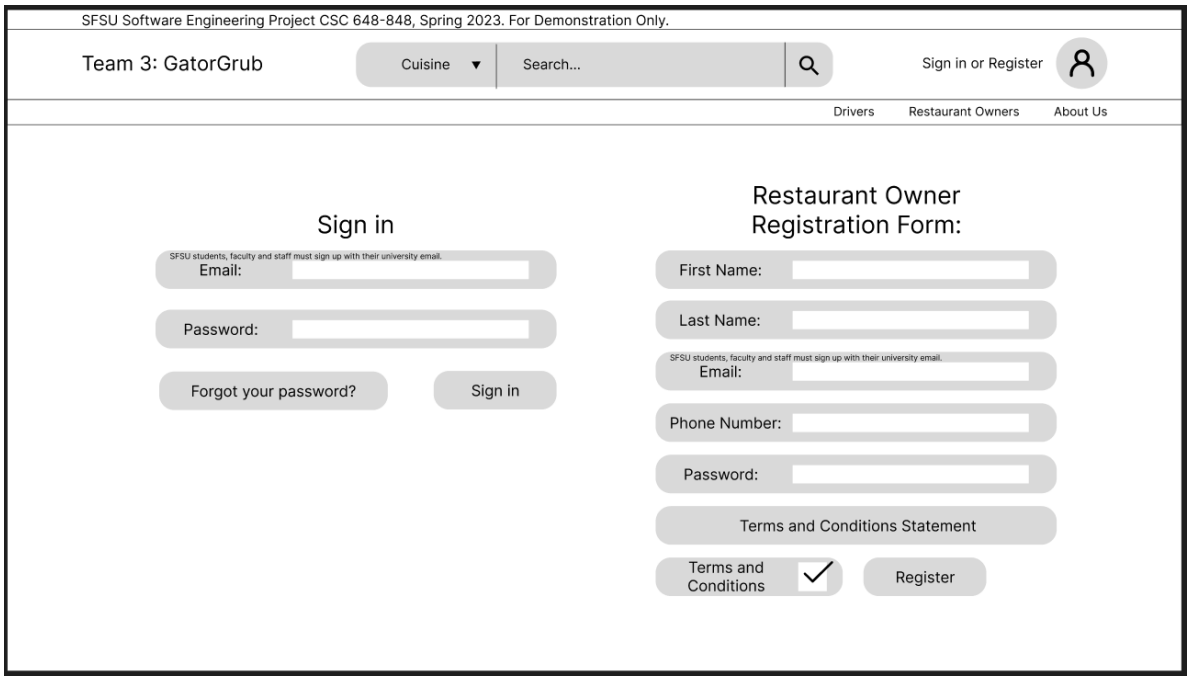


3. After clicking on the driver profile on the top right, it leads to this page for the driver, where they can accept/decline orders:



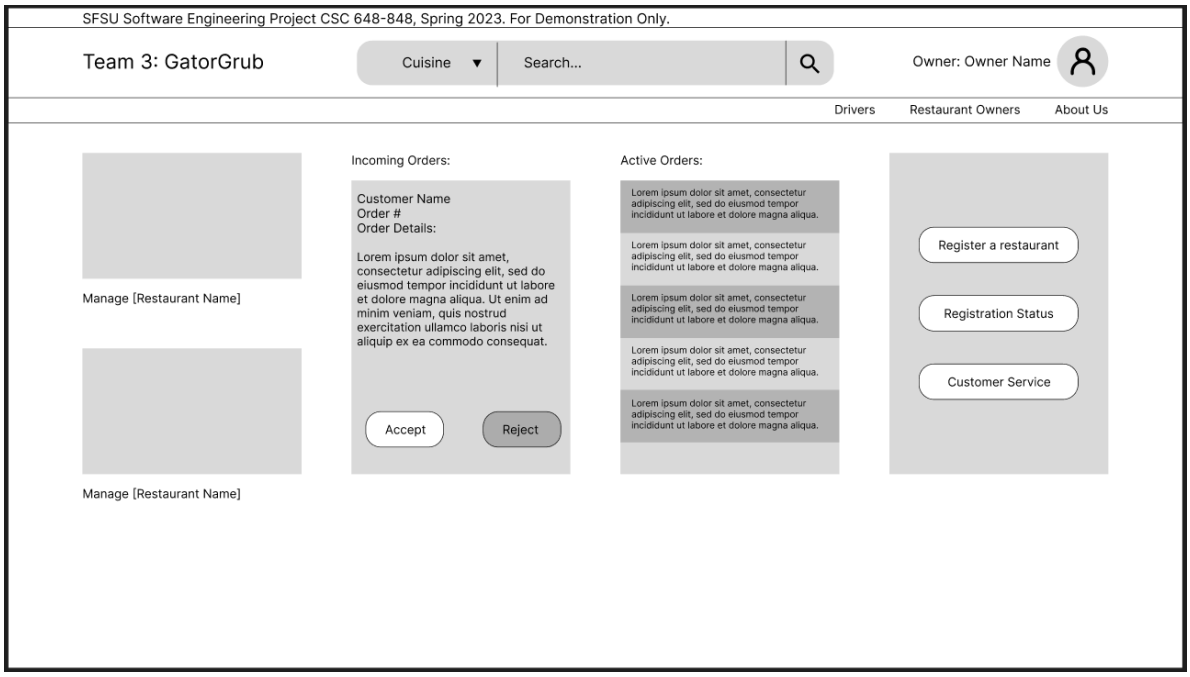
Case 4: Marcus (**Restaurant owner**) is looking for ways to increase the exposure of his restaurant and attract more customers. He realizes that by partnering with GatorGrub, he can cater to a specific group of customers: the students and staff at San Francisco State University.

1. After the restaurant owner signs up their account through the home page,



2. They then can click their icon and then this restaurant registration form will appear. He will need to provide details about his restaurant, such as the menu pdf, selecting operating hours, and selecting location. To register his restaurant with GatorGrub, Marcus needs to create a restaurant owner account.

2. After their restaurant is approved by the website admin, they can see incoming orders and online orders through clicking their profile icon.



5. High-level architecture and DB Organization Summary:

DB Organization:

List of main data items and entities:

- SFSUCustomer:
  - SFSUCustomerID (PK)
  - SFSUCustomerName
  - SFSUCustomerEmail
  - SFSUCustomerPhone
  - SFSUCustomerPassword
- Driver:
  - DriverID (PK)
  - DriverName
  - DriverEmail
  - DriverPhone
  - DriverPassword
- RestaurantOwner:
  - RestaurantOwnerID (PK)

    - RestaurantOwnerName
    - RestaurantOwnerEmail
    - RestaurantOwnerPhone
    - RestaurantOwnerPassword
- Restaurant:
    - RestaurantID (PK)
    - RestaurantOwnerID (FK)
    - RestaurantRegistered (boolean for admin review)
    - RestaurantName
    - RestaurantEmail
    - RestaurantPhone
    - RestaurantPassword
    - RestaurantAddress
    - RestaurantCuisineType
    - RestaurantPriceTier (FK)
    - RestaurantHours (Hours of operation, specifically for pick-up/delivery)
    - PrepTime (Avg. time for orders ready for takeout)
    - RestaurantImage
- RestaurantMenu:
    - RestaurantMenuID (PK)
    - RestaurantID (FK)
- MenuItem:
    - MenuItemID (PK)
    - RestaurantMenuID (FK)
    - MenuItemName
    - MenuItemPicture
    - MenuItemPrice
    - MenuItemType (e.g. appetizer, entrée, sides, dessert, drinks)
    - MenuItemDescription
- Order:
    - OrderID (PK)
    - CustomerID (FK)
    - DriverID (FK)
    - RestaurantID (FK)
    - OrderTime
    - DeliveryTime (OrderTime + RestaurantPrepTime + Google Maps Journey Time)
    - DeliveryAddress (includes customer’s address, room &building, restaurant location and safe pickup spots)
    - OrderSelections (CSV: MenuItemID, quantity, notes)
    - OrderNotes
    - OrderDiscounts

    - OrderPrice
- Review (Priority 2):
    - ReviewID (PK)
    - RestaurantID (FK)
    - UserID (FK)
    - Message
    - Rating

Media Storage:

Our media will be stored in the file system of the server. It is not advisable to store media in a database as it can put an unneeded strain on it. Our database will instead either store a path to a resource, or a link to that resource stored on a CDN.

Search functionality implementation:

Our search functionality will be fairly simple. Our database will store each restaurant along with metadata such as location, cuisine, price information, restaurant name, and price tier. This way we can organize restaurants by categories for users to browse. When a user wants to search using an input field, we will take the input, sanitize it and plug it into SQL queries for retrieval. We could use the LIKE function to perform a fuzzy search. We can implement indexing and implement pagination to optimize search functions.

SW tools changes:

We have switched our primary operating system from AlmaLinux to Ubuntu 20.04 LTS. AlmaLinux was a miscommunication and Ubuntu is a more standard OS.

6. Identifying Key Risks To Our Project:

Schedule Risks:

In-person meetings are difficult for our team to complete due to conflicting schedules. We have had times when some team members have not been present at a meeting. Keeping track of specific requirements and details for our submission becomes hard, when people aren't all on the same page.

We have addressed this in the past and plan to keep an eye on it by encouraging in-person meetings when possible, but also opening up to online meetings, which has the added benefit of screen sharing and clear discussion. We also host more casual meetings to check in with each other. Team Lead makes sure to post helpful meeting notes so that

members have something to look back on and review. After feedback we'll be trying even harder to catch all of our mistakes as we continue the assignment.

Technical Risks:

Keeping track of Google Compute Engine (GCE) has been difficult and has left us with a little less time available than we initially thought we would have at this stage in the project. If the issue is not dealt with it can cause our team to spend money (against the requirements).

We will continue to keep an eye on the GCE and make sure we have as much time as possible, but we have to come to an agreement on how to move forward. We intend not to spend money on this project, so we have to consider migrating our server to another account and stopping our service with the previous account so the individual group member doesn't get charged. After feedback, we've opted for migrating to a new account and refactoring places where our server details in the credentials file, so we can connect our other components to the new server.

7. Project Management:

Milestone 2 was pretty straightforward to manage for our team. After the class where we discussed milestone 2, our team lead was able to set up a Trello workspace for our team. After the milestone sections had been added to it, everyone was invited to see what we had on our plate. A meeting took place when we had set up the Trello workspace, in order to make sure everyone was on the same page. Each section of the milestone 2 document was briefly outlined and then assigned to the different sections of the team (back end & front end). The team lead also had some sections to fill out independently. Our M2 part 1 document was held on google drive and everyone has access to edit whatever they'd like. For anyone who may have had to miss the team meeting, the meeting notes were posted on a dedicated Discord text channel on the team's Discord server. (Discord is a team based messaging platform built for small professional teams and gaming enthusiasts, very similar to Slack.) The meeting notes specified instructions for the team, including goals, requirements and things to get done. These are typically specified for the two subteams, front end and back end. If we get close to a deadline, individual members are asked to complete specific parts of the assignment so that we can stay on track. Our meeting after the M2 reveal consisted of reviewing CEO notes and editing the M1 document. We then discussed any thoughts and concerns we had for each section of the M2 document. Team lead is responsible for the whole document being presentable and concise, whereas front end leads and back end leads should discuss directly with their partner how much work, assigned by the team lead, each is responsible for completing. When the rough draft is done, all group members reconvene to discuss the state of the submission document, as well as any other concerns they have with upcoming requirements for M2 part 2. After

everyone is able to make their edits and share their thoughts, everyone makes the appropriate changes and the team lead is responsible for finalizing the document for submission. After the finalized edited document is prepared, the team comes to review it one last time before submission to make sure they're happy with their work. These reviews usually take place over a Discord voice call so we can point out specific parts of the submission while sharing our screens for sharing notes.

SW Engineering CSC648-848 Spring 2023

GatorGrub

Team 3

Shauhin Pourshayegan: spourshayegan@mail.sfsu.edu	Team Lead, Github Master, Document Editor
Hieu Ma:	Front End Lead
Preetham Ramesh:	Back End Lead
Lin Tun:	Front End Developer
Derrick Liang:	Back End Developer

Milestone 3

Date Submitted:	4/28/2023

1. Milestone 3 Meeting Summary:

- **Summary of feedback on UI (record all pages that need to be resubmitted in a pdf):**

There are many things we need to fix on our UI pages. The first one that the professor wanted us to fix is on our homepage. We were told to change our homepage to be more attractive, in other words he wanted us to make the homepage more creative because right now our homepage is mostly boxes and white in the background. Another thing the professor wanted us to fix is on our navbar of how we structure on our register/login to the right side and make each category with hover. He wants all the driver, restaurant owner, and SFSU customer register/login buttons. Beside that he also wanted us to have a logo on the navbar, so it is more lookable. On our register and login pages he was hoping we would be able to make a button that can direct back to the homepage(User feedback).

Furthermore, the professor would like our restaurant result page to have the search number result amount show up on the left corner so customers know how many results were found. On our restaurant register form, the professor wanted the site to be more simple, which meant he did not like to have a lot of different boxes in one page. He wanted us to put the main information that the restaurant owner should put on the restaurant register form. Beside that on the restaurant result page, the professor wanted to see the distance of how far the restaurant will be to SFSU for each restaurant. In the Driver page, the professor also did not want us to overcomplicate our design. The two features that professor wants to see right now is the accept orders page from restaurant and completion of the SFSU customer delivery page.

- **Summary of feedback on code and architecture:**

Our code was less criticized, but there were chances to make our files more friendly to our developers. CTO recommended shortly after the meeting that we have a place to improve routing in our file structure and this can be addressed fairly easily.

- **Summary of feedback on GitHub usage:**
    - Github usage is satisfactory, however we could work harder to make our commit messages more meaningful. CTO also mentioned we could reduce the small commits and try to make large changes at once
- **Summary of feedback on DB:**
    - Although the DB is mostly satisfying, it still needs a table for restaurant cuisines. This will allow users to select from a changing list and allows restaurants to add new cuisines to the list.
- **Summary of feedback on Teamwork and risk management:**
    - Leadership issues with assigning and tracking work to other group members in a timely manner.

    - In order to address this issue our team lead is responsible for a review of our progress so far, and with the knowledge of the milestone's team meeting, will
- **Architecture review COMPLETE**

2. Functional Requirements Prioritized:

- **Priority 1**
    - Unregistered Users:
        - Unregistered users shall be able to search by restaurant category.
        - Unregistered users shall be able to search for food and restaurants.
        - Unregistered users shall be able to search the restaurant map from their location.
        - Unregistered users shall be able to sort restaurants by delivery time.
        - Unregistered users shall be able to sort restaurants by price.
        - Unregistered users shall be able to select menu items and add menu items to their cart.
        - Unregistered users shall be able to choose an order's delivery location.
        - Unregistered users shall be able to read restaurant menus.
        - Unregistered users shall be able to register an account with SFSU email.
        - Unregistered users shall be able to register an account as a restaurant owner
        - Unregistered users shall be able to register an account as a driver
    - SFSU Customers:
        - SFSU Customers shall inherit all unregistered user functional requirements.
        - SFSU Customers shall be able to log into their account.
        - SFSU Customers shall be able to submit an order.
    - Admins:(This will be from the workbench)
        - Admins shall be required to approve all restaurant applications before they go live on the app.
        - Admins shall review restaurant applications and provide the appropriate response.
        - Admins shall be able to view the user account.
        - Admins shall be able to warn users and restrict orders from the user account, if a user violates GatorGrub policy with the account.
        - Admins shall be able to remove users.
        - Admins shall be able to remove restaurants.

    - Restaurant Owners:
        - Restaurant owners shall be able to register for an account.
        - Restaurant owners shall be able to log into their account.
        - Restaurant owners shall be able to upload menu item pictures and prices.
        - Restaurant owners shall be able to submit restaurant details. (e.g. hours of operation, avg. order time.)
        - Restaurant owners shall be able to choose to accept or reject an order.
        - Restaurant owners shall confirm when orders are ready. This updates the order progress.
        - Restaurant owners shall be able to review and manage driver tasks
    - Drivers:
        - Drivers shall be able to register as a driver.
        - Drivers shall be able to log in as a driver.
        - Drivers shall be able to accept or reject an order.
        - Drivers shall confirm when food is picked up. This updates the order progress.
        - Drivers shall confirm when food is delivered. This updates the order progress.
        - Drivers shall be able to see a map of the pickup point.
        - Drivers shall be able to see a map of the delivery point.

3. High-level architecture Revisions:

- We intend to use node.bcrypt.js 5.1.0 library to implement the encryption for all user passwords. Approved by CTO.
- We have previously updated our fuzzy search to use Fuse.js 6.6.2 library. Approved by CTO.
- Our database model will now include a table for restaurant cuisines so that they may be displayed and updated dynamically.

4. Identifying Key Risks To Our Project:

Schedule Risks:

Some team members have overbearing assignments from other classes and this prevents them from completing assigned tasks. This causes the team to fall behind with checkpoints and functional requirements.

These delays usually take place when multiple completed components are integrated together and something breaks. We want to allocate more time to address issues like this, which cause the most stress when nearing due dates. We're working harder to break down the tasks we need to get done and schedule more reasonable and in-depth tasks for each person. These tasks will be scheduled to be completed by a point that gives team members more time to address issues, which often come up with each task.

7. Checkpoint Due Thursday 5/4:

Before Thursday 5/4 we need to have:

- PDF of all missing and revised pages (screenshots of website)
    - Homepage
    - Search results
        - Have est. time to deliver to SFSU.
    - Almost all login and registration pages need minor reworks
    - Owner Dashboard
    - Restaurant application/registration
    - Restaurant menu & order page.
    - Driver dashboard

SW Engineering CSC648/848 Spring 2023
Food Delivery App: GatorGrub
Team 3

Shauhin Pourshayegan: spourshayegan@mail.sfsu.edu	Team Lead, Github Master, Document Editor
Hieu Ma:	Front End Lead
Preetham Ramesh:	Back End Lead
Lin Tun:	Front End Developer
Derrick Liang:	Back End Developer

Milestone 4
Product description, Beta Launch, QA and Usability Testing

Date Submitted:	5/23/2023

1) Product summary

GatorGrub:

Gator grub is a food delivery app which offers SFSU students, faculty and staff convenient and affordable ways to get food delivered to them. People at universities often find themselves stuck when searching for their next meal. Food options on campus are likely few and far between, and have the potential to taste like the ever-dreaded dining hall food. Students and instructors alike find issues managing their food habits when heavy workloads and packed schedules take precedent. As students of San Francisco State University, we find ourselves trekking back and forth to Stonestown Galleria just so we can get access to more options, wasting our time when it's most valuable to us. Other than the inconvenience of food locations and availability, the food available is not always affordable, which is a major issue for students and important for everyone. The inconvenience of our current options and the lack of affordable options are the primary driving forces of our team project.

Being students, we wanted to provide a solution to these issues when we came to discuss what GatorGrub will be. We propose a service which will strengthen the connection between SFSU students, faculty and staff to their local food scene, as well as offer this primary user base with features which make their lives easier and solve the problems related to getting food on campus. GatorGrub aims to provide these users with a selection of local restaurants offering delivery. Customers who don't have the time to leave their classroom or study area can get their food delivered to any room on campus, allowing their workshops and group study sessions to be uninterrupted by rumbling tummies. Customers can also decide whether they want their food delivered to one of several safe pickup spots throughout SFSU's campus. This feature benefits students who want to get food safely, without worrying about who they will encounter on the long walk to Stonestown or further out. GatorGrub also enables its users to get deals on different local restaurants. This will incentivise more sales, making restaurant owners and delivery drivers content and strengthening GatorGrub's infrastructure.

On GatorGrub, anyone who joins can:

- Unregistered users shall be able to search by restaurant category.
- Unregistered users shall be able to search for food and restaurants.
- Unregistered users shall be able to search the restaurant map from their location.
- Unregistered users shall be able to sort restaurants by delivery time.

- Unregistered users shall be able to sort restaurants by price.
- Unregistered users shall be able to select menu items and add menu items to their cart.
- Unregistered users shall be able to choose an order's delivery location.
- Unregistered users shall be able to read restaurant menus.
- Unregistered users shall be able to register an account with SFSU email.
- Unregistered users shall be able to register an account as a restaurant owner
- Unregistered users shall be able to register an account as a driver

SFSU students, faculty and staff can:

- SFSU Customers shall inherit all unregistered user functional requirements.
- SFSU Customers shall be able to log into their account.
- SFSU Customers shall be able to submit an order.

Restaurant owners can:

- Restaurant owners shall be able to register for an account.
- Restaurant owners shall be able to log into their account.
- Restaurant owners shall be able to upload menu item pictures and prices.
- Restaurant owners shall be able to submit restaurant details. (e.g. hours of operation, avg. order time.)
- Restaurant owners shall be able to choose to accept or reject an order.
- Restaurant owners shall confirm when orders are ready. This updates the order progress.
- Restaurant owners shall be able to review and manage driver tasks.

Drivers for GatorGrub can:

- Drivers shall be able to register as a driver.
- Drivers shall be able to log in as a driver.
- Drivers shall be able to accept or reject an order.
- Drivers shall confirm when food is picked up. This updates the order progress.
- Drivers shall confirm when food is delivered. This updates the order progress.
- Drivers shall be able to see a map of the pickup point.
- Drivers shall be able to see a map of the delivery point.

2) Usability test plan for selected function

Selected Function for Testing Usability : Searching

I. Test objectives:

The search functionality serves as a fundamental component within our web application, as it equips users with a valuable means to refine their results. The efficacy and user-friendliness of the search feature, as well as its accuracy in addressing user input, will establish the initial and most crucial impression.

Furthermore, input validation is essential and must undergo thorough testing prior to the final release. On one hand, it is important to limit the length and content of user input to prevent any unsuitable code injections. Conversely, when input is deemed valid, we ought to display items that closely align with the user's search query.

Ultimately, the testing objectives outlined above aim to gather user feedback, empowering our development team to enhance the user experience for both current and prospective customers.

II. Test background and setup

(a). System Setup

The system setup process is both swift and straightforward. From the user's standpoint, there is no need for installation. To test the search function, users simply need to launch a web browser on their computer or laptop and ensure that they are connected to the internet. We currently support all the latest versions of web browsers, including Google Chrome, Firefox, Microsoft Edge, and more. By entering the URL into their web browser, users can easily access our food delivery website, "GatorGrub."

(b). Staring Point

The starting point for the search function is the homepage, which can be accessed via the URL provided below. The search bar, which is used to initiate the search function, is located just above the navigation bar. If the user's input is valid in terms of character(s) and length, the corresponding search results will be displayed on the same page. However, if the input is invalid, the search result will be blank, with no restaurants displayed.

(c). Intended User

We understand that users may simply want to search our website without the need to register or log in. As such, for the search function, users of all types, including unregistered, registered, and admin users, are intended users. We aim to provide a seamless and hassle-free experience for anyone who wishes to search for items on GatorGrub, without any limitations or restrictions. Therefore, users can search for items on our website without the need to register or log in.

(d). Task

Task #	Task for Usability Test
1	Find All the restaurants.
2	Find Chinese restaurants.
3	Find a restaurant called Bevande Venue.

(e). URL to be Tested
http://34.82.124.237:3001/

III. Questionnaire

Below is a Likert-Scale Questionnaire with questions inquiring user's satisfaction after performing the above tasks. Each "question" is a statement with five choices (from **Strongly Agree** to **Strongly Disagee**) scaling user's response.

	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
Category browsing is intuitive.					
Search function is convenient to use.					
The user interface for finding restaurants is pleasant to use.					
The search function was user-friendly.					
Overall, I am satisfied with the search function and category browsing on Gator Grub.					

Comments: **(Your feedback are especially appreciated)**
_____ _____

_____ _____

3) QA test plan and QA testing - max 2 pages (please consult class slides on this topic)

a) For the same function you chose for the usability test, write a QA test plan (check class slides), with brief and separate sections as follows:

- **HW and SW setup** (including URL):
- **Feature to be tested:**
- **QA Test plan: in table format**: This is the plan to be given to QA tester to execute your QA test plan. Contains min of 3 test cases and results of testing them on your system: appr. 1 page. You must provide QA test plan in a separate section in the easy to read tabular form allowing easy reading and analysis by management e.g. like presented in the class slides on SW QA.

I. Test Objectives

Beyond user satisfaction, the QA test for the search function primarily focuses on quality, specifically the accuracy of results and deployment. In order to assess the correctness of the search function, each search result is examined to determine if the item title closely corresponds to the input text. Furthermore, the total number of results is tallied to ensure that all matching records in the database have been successfully retrieved.

II. HW and SW Setup

(a). Server Host

- The server host has been installed on the operating system of AlmaLinux version 9.1.
- Google Compute Engine has been installed on the Windows System, with the free-tier services selected for our web application.
- Google Compute Engine Server has been allocated with 1vCPU, 2 GB RAM.

(b). Software Setup (need to ask and add by back-end)

- MySQL 8.0.32 has been successfully setup, which is required for running the database.
- Express.js 4.18.2 has been successfully setup, which is required for running the web server.
- Node.js 18.14.1 has been successfully setup, which is required for running the server-side language.
- Node.bcrypt.js 5.1.0 library has been successfully setup, which is required for

implement the encryption for all user password.
- Fuse.js 6.6.2 library has been successfully setup, which is required for serach function.

(c). APIs
- Google Analytics Reporting API v4
- Google Maps JavaScript API
- Google Routes API

(d).Web Browser
- In this QA testing, QA tester will use on the web browser of the latest 2 versions, Google Chrome and Microsoft Edge browsers.
- Most of the computers have these browser either installed or automatically upgraded. If not available, the QA tester may have to download them and install.
- The URL for QA test is http://34.82.124.237:3001/

III. Features to be tested
Below will be the features of the search function, and the table of QA test plan.

(a). Selection group of categories
- Upon selecting the "All" category, users will be able to view all the restaurants listed on the website. Notably, this category is also the default setting when users first visit our homepage.
- When users switch to other categories, the search results should correspond to the specific category selected. This ensures that users are presented with relevant search results that align with their intended search criteria.

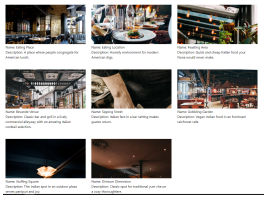
(b). Approximate matching
- For text search, the results do not have to perfectly match the user's input. Instead, approximate matching will be executed.

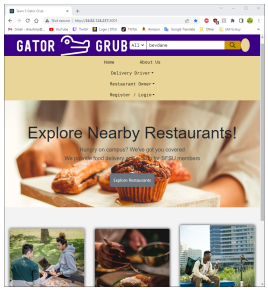
(c). Responsiveness
- By expanding or shrinking the browser, the list page of search results is responsive to the change of browser size.

IV. Test Plan
To enhance the readability of the test plan, a tabular format is provided below. This format allows users to easily comprehend the details included in the QA test.

The columns in the QA table have been structured according to both standard and recommended formats, which include the following: test number, test description, test input, expected output, and test results (either PASS or FAIL) on both Google Chrome and Microsoft Edge (the two latest versions for each browser).

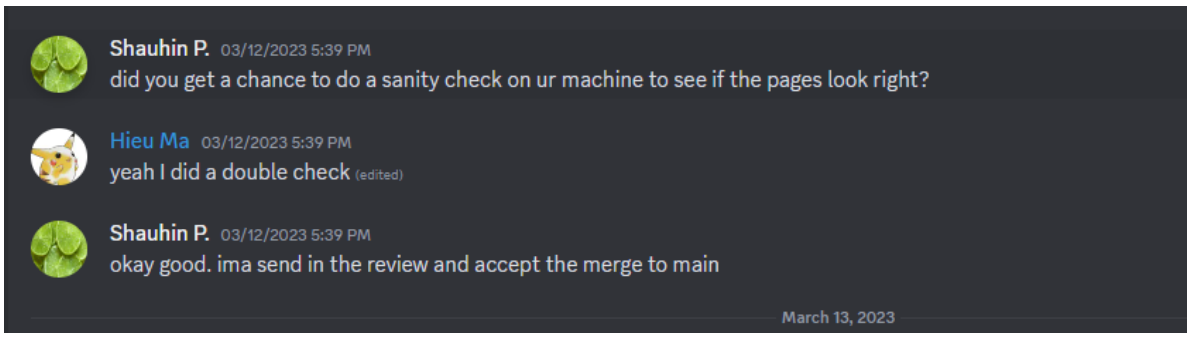
This format ensures that all necessary information is included in the table and that the test results are consistent across different browsers and versions.

Test Number	Test Title and Description	Test Input	Expected Output	Test Results (Chrome)	Test Results (Edge)
1	Selection group of categories - Test for the Search bar: Upon selecting the "All" category, users will be able to view all the restaurants listed on the website.	• On the homepage of GatorGrub, navigate to the search bar at the top of the screen. "**All**" should already be selected. • Click on the search icon to the left. • Observe the results on the search results page.	 An empty string means all restaurants. Under "**All**" category, there will be 8 restaurants.	Pass	Pass
2	Selection group of categories - Search specific category: Users expect to be able to search for food by cuisine. The test confirms that the search function should allow users to do this.	• On the homepage of GatorGrub, navigate to the search bar at the top of the screen. "All" should already be selected. • Click on the cuisine. • Change the selected cuisine to "Italian". • Click on the search icon to the left. • Observe the results on the search results page.	 Results: Name: Sipping Street Description: Italian fare in a bar setting makes guests return. Name: Feasting Area Description: Quick and cheap Italian food your Nona would never make. Name: Bevande Venue Description: Classic bar and grill in a lively commercial alleyway with an amazing Italian	Pass	Pass

			cocktail selection.		
3	Approximate matching search: There must be some allowance for typos in search queries. The test must ensure the search can function regardless of simple typos.	• On the homepage of GatorGrub, navigate to the search bar at the top of the screen. “All” should already be selected. • Click on the search bar input field. • Enter or type “bevdane avenue” in this field. • Click on the search icon to the left. • Observe the results on the search results page.	“bevdane avenue” is a misspelling of “Bevande” from the restaurant “Bevande Venue”. Results:  Name: Bevande Venue Description: Classic bar and grill in a lively commercial alleyway with an amazing Italian cocktail selection. Name: Bevande Venue Description: Classic bar and grill in a lively commercial alleyway with an amazing Italian cocktail selection.	Pass	Pass
4	Responsiveness: By expanding or shrinking the browser, the list page of search results should be responsive to the change of browser size.	• Have the browser window maximized. • Be on the homepage of GatorGrub. • Drag the window to the side of the screen to collapse it horizontally. • Observe the results.		Pass	Pass

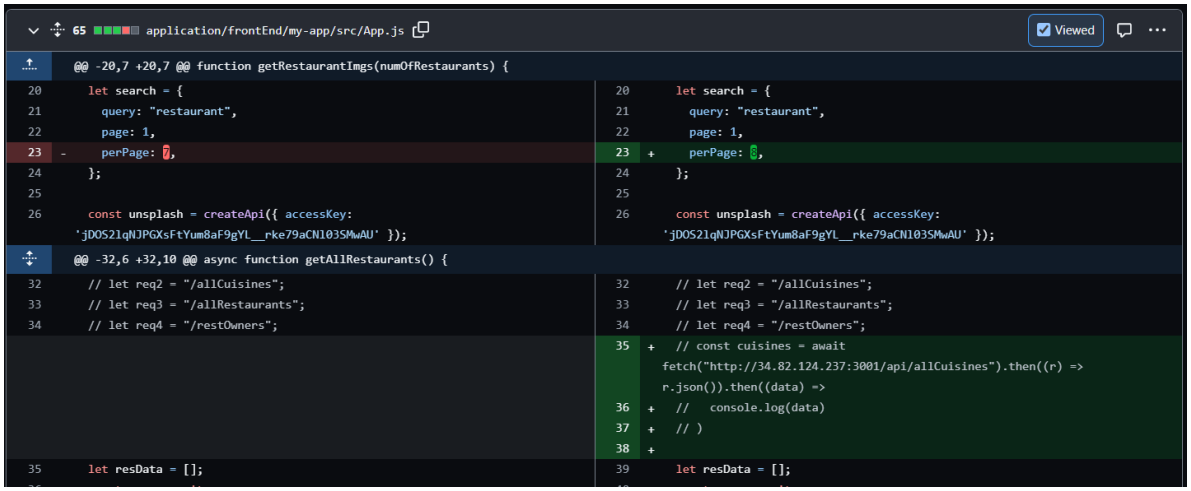
4) Peer Code Review:

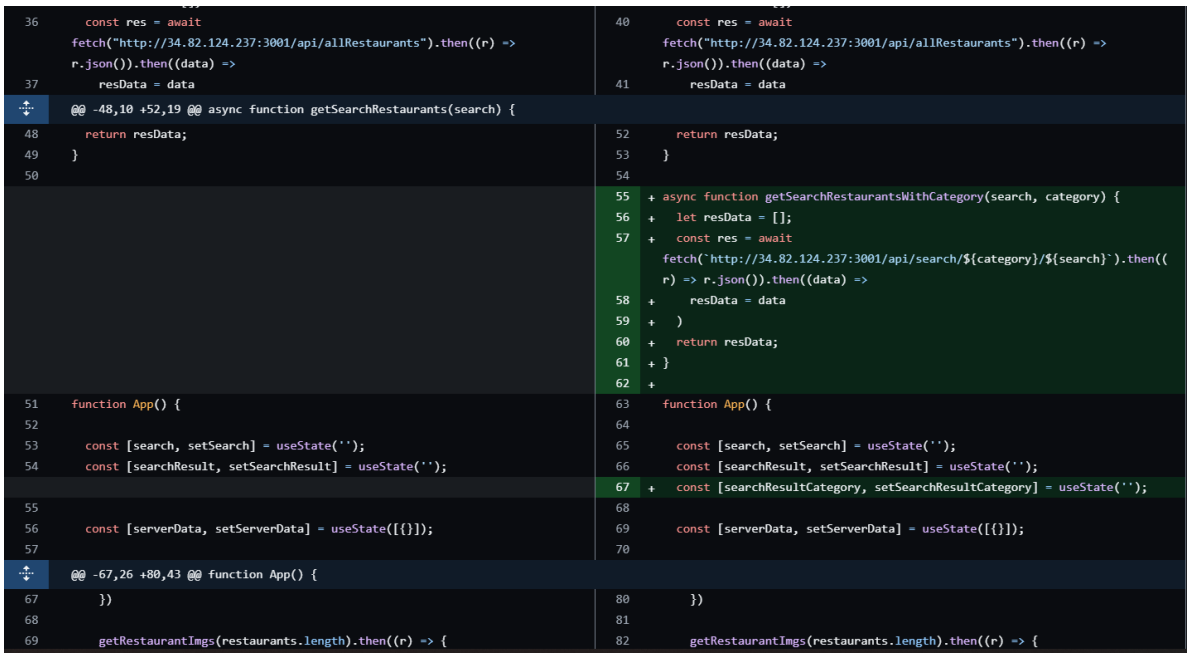
Person who code is being reviewed sends e-mail to reviewer with pointer to the code and asks for review:



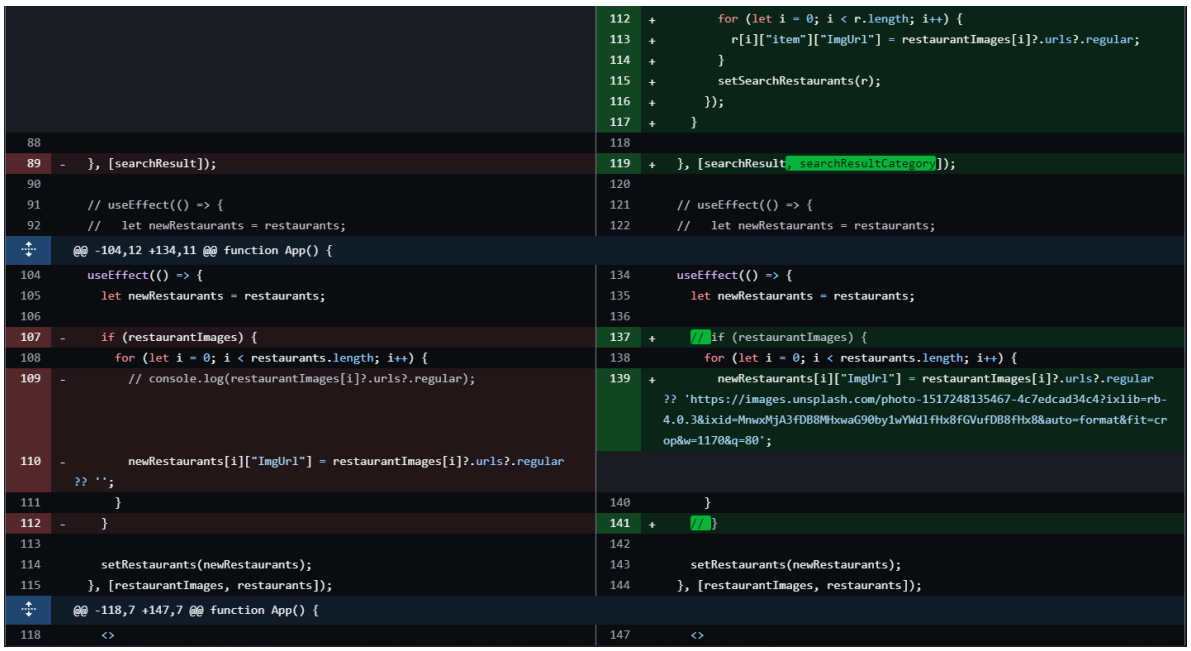
Reviewer reviews the code in whatever way is most practical for you (e.g. commenting on code in repository, or using github review options):

For this code review, the reviewer was on call with the reviewee. They skimmed through code together before the reviewer had to review independently with the reviewee still listening to hear feedback immediately.





```
  const res = await fetch("http://34.82.124.237:3001/api/allRestaurants").then((r) => r.json()).then((data) =>
    resData = data
@@ -48,10 +52,19 @@ async function getSearchRestaurants(search) {
  return resData;
}

+ async function getSearchRestaurantsWithCategory(search, category) {
+   let resData = [];
+   const res = await fetch(`http://34.82.124.237:3001/api/search/${category}/${search}`).then((r) => r.json()).then((data) =>
+     resData = data
+   )
+   return resData;
+ }
+
function App() {

  const [search, setSearch] = useState('');
  const [searchResult, setSearchResult] = useState('');
+   const [searchResultCategory, setSearchResultCategory] = useState('');

  const [serverData, setServerData] = useState([{}]);

@@ -67,26 +80,43 @@ function App() {
    })

    getRestaurantImgs(restaurants.length).then((r) => {
-     setRestaurantImages(r.response.results);
    })

  }, []);

  // Search Use Effect
  useEffect(() => {
    let newRestaurants = [];
-     console.log("GOT RESPONSE", searchResult);

-     getSearchRestaurants(searchResult).then((r) => {
-       console.log(r)
-       for (let i = 0; i < r.length; i++) {
-         r[i]["item"]["ImgUrl"] = restaurantImages[i].urls.regular;
-       }
-       console.log("search restaurants use effect", r)
-       setSearchRestaurants(r);
-     });
+     if(searchResultCategory !== 'all') {
+       getSearchRestaurantsWithCategory(searchResult, searchResultCategory).then((r) => {
+         for (let i = 0; i < r.length; i++) {
+           r[i]["item"]["ImgUrl"] = restaurantImages[i]?.urls?.regular;
+         }
+         console.log("search restaurants use effect", r)
+         setSearchRestaurants(r);
+       });
+     } else if (searchResultCategory === 'all' && searchResult === '') {
+       getAllRestaurants().then((r) => {
+         let newRestaurants = [];
+         for(let i = 0; i < r.length; i++) {
+           newRestaurants.push({item: r[i]});
+         }
+         for (let i = 0; i < newRestaurants.length; i++) {
+           newRestaurants[i]["item"]["ImgUrl"] = restaurantImages[i]?.urls?.regular;
+         }
+         setSearchRestaurants(newRestaurants);
+       })
+     } else {
+       getSearchRestaurants(searchResult).then((r) => {
```



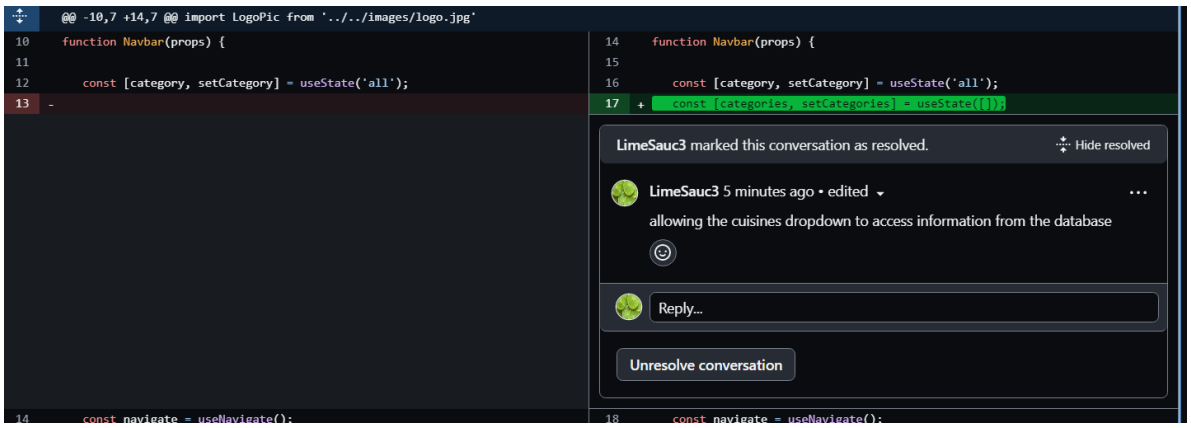
```
                                                        +         for (let i = 0; i < r.length; i++) {
                                                        +           r[i]["item"]["ImgUrl"] = restaurantImages[i]?.urls?.regular;
                                                        +         }
                                                        +         setSearchRestaurants(r);
                                                        +       });
                                                        +     }
88                                                      
-   }, [searchResult]);                                 +   }, [searchResult, searchResultCategory]);
90                                                      
91      // useEffect(() => {                            121     // useEffect(() => {
92      //   let newRestaurants = restaurants;          122     //   let newRestaurants = restaurants;
@@ -104,12 +134,11 @@ function App() {
104     useEffect(() => {                               134     useEffect(() => {
105       let newRestaurants = restaurants;             135       let newRestaurants = restaurants;
106                                                     136
-       if (restaurantImages) {                       +       //if (restaurantImages) {
108       for (let i = 0; i < restaurants.length; i++) { 138       for (let i = 0; i < restaurants.length; i++) {
-         // console.log(restaurantImages[i]?.urls?.regular); +         newRestaurants[i]["ImgUrl"] = restaurantImages[i]?.urls?.regular
                                                        +   ?? 'https://images.unsplash.com/photo-1517248135467-4c7edcad34c4?ixlib=rb-
                                                        +   4.0.3&ixid=MnwxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8&auto=format&fit=cr
                                                        +   op&w=1170&q=80';
-         newRestaurants[i]["ImgUrl"] = restaurantImages[i]?.urls?.regular
-   ?? '';
111       }                                             140       }
-     }                                               +       //}
113                                                     142
114       setRestaurants(newRestaurants);               143       setRestaurants(newRestaurants);
115     }, [restaurantImages, restaurants]);            144     }, [restaurantImages, restaurants]);
@@ -118,7 +147,7 @@ function App() {
118       <>                                            147       <>
```

```
119         <div className="App">                       148         <div className="App">
120           <header className="App-header">           149           <header className="App-header">
-           <Navbar search={search} setSearch={setSearch} setSearchResult= +           <Navbar search={search} setSearch={setSearch} setSearchResult=
    {setSearchResult} />                                    {setSearchResult} setSearchResultCategory={setSearchResultCategory} />
122           </header>                                 151           </header>
123         </div>                                      152         </div>
124         <Routes>                                    153         <Routes>
```

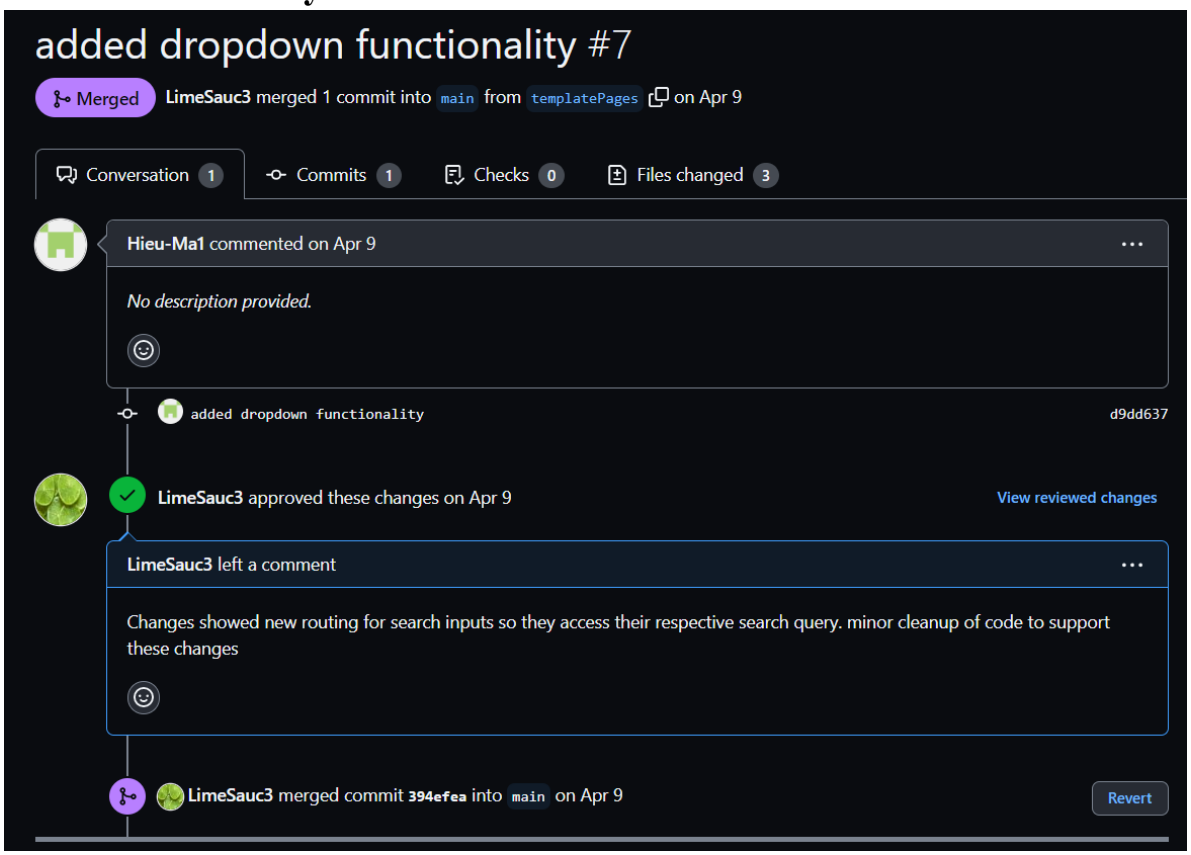
```
    '../../components/loginRegisterModal/LoginRegisterModal';   '../../components/loginRegisterModal/LoginRegisterModal';
12                                                      16
-   async function submitOrder(orderPrice, location, restaurantID, userName, +   async function submitOrder(orderPrice, location, restaurantID, userName,
    setModalShow, selectedItems) {                          setModalShow, selectedItems, userID) {
14      console.log("submit order login username", userName)   18      console.log("submit order login username", userName)
15      if (userName) {                                 19      if (userName) {
16        // console.log("submitted order");            20        // console.log("submitted order");
@@ -23,8 +27,8 @@ async function submitOrder(orderPrice, location, restaurantID, userName, setModa
23              'Content-Type': 'application/json'      27              'Content-Type': 'application/json'
24            },                                        28            },
25            body: JSON.stringify({                    29            body: JSON.stringify({
-             CustomerID: 0,                            +             CustomerID: userID,
-             DriverID: 0,                              +             DriverID: 0,
```

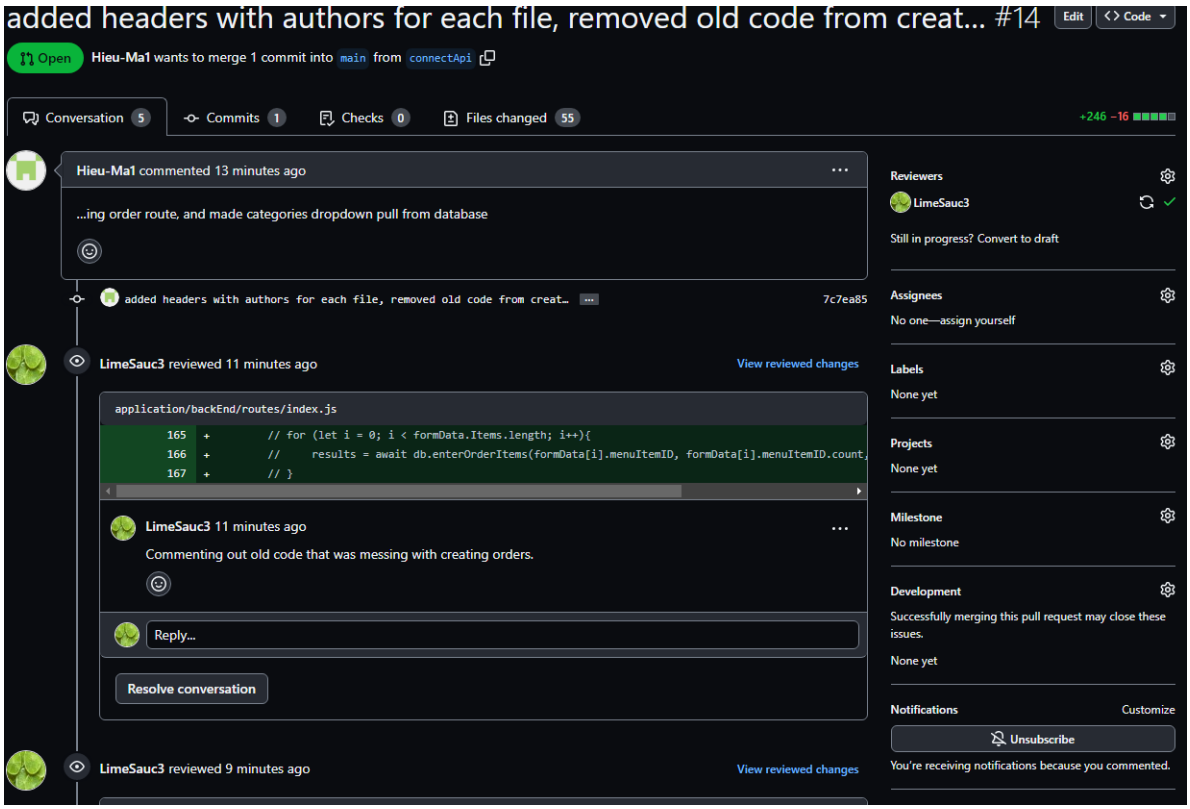


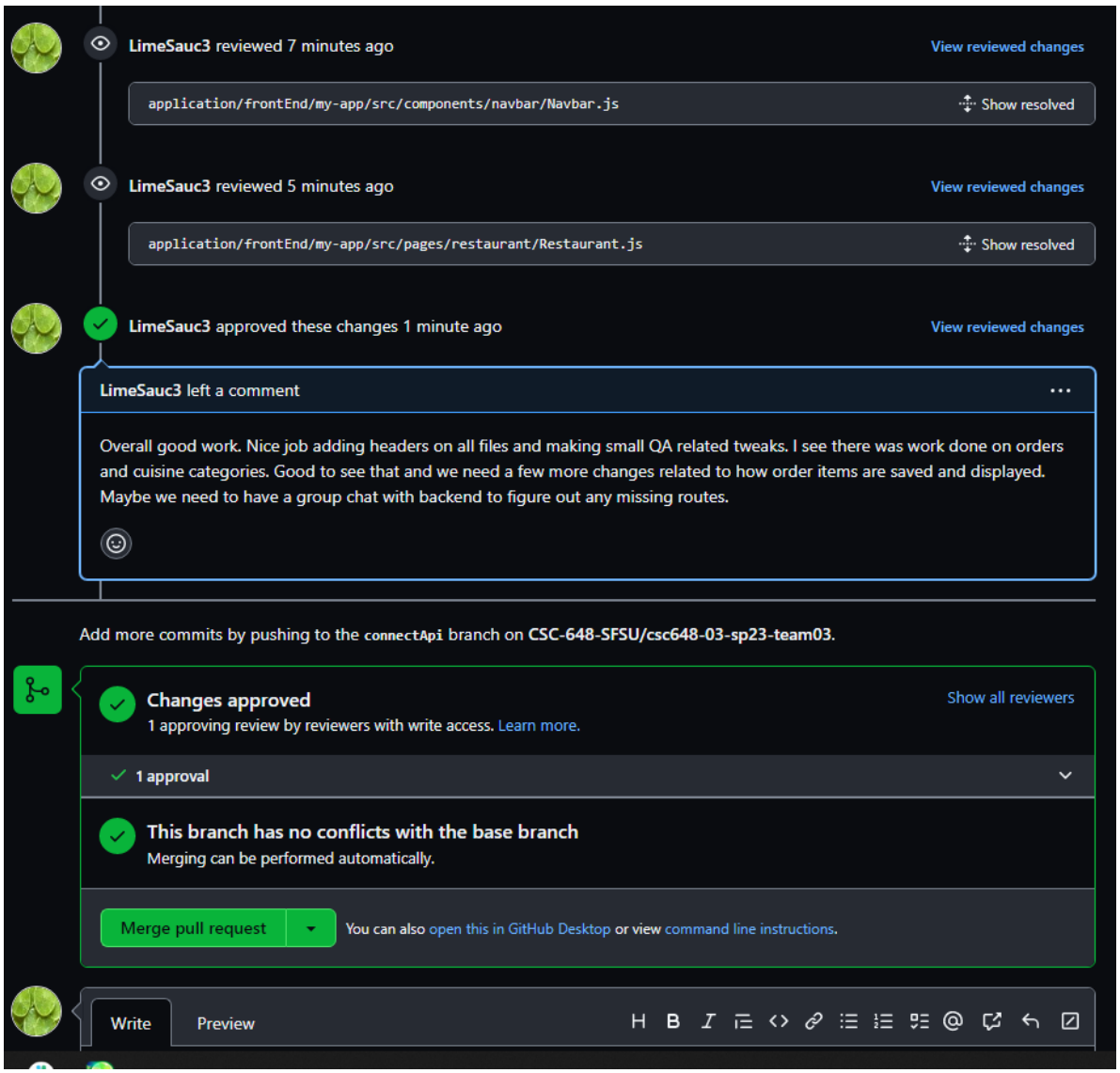
```
28              RestaurantID: restaurantID,             32              RestaurantID: restaurantID,
```



Reviewer sends summary of review in email back to coder:




added headers with authors for each file, removed old code from creat... #14
Edit
Code
Open
Hieu-Ma1 wants to merge 1 commit into main from connectApi
Conversation 5
Commits 1
Checks 0
Files changed 55
+246 −16
Hieu-Ma1 commented 13 minutes ago
...ing order route, and made categories dropdown pull from database
added headers with authors for each file, removed old code from creat…
7c7ea85
LimeSauc3 reviewed 11 minutes ago
View reviewed changes
application/backEnd/routes/index.js
165 + // for (let i = 0; i < formData.Items.length; i++){
166 + // results = await db.enterOrderItems(formData[i].menuItemID, formData[i].menuItemID.count
167 + // }
LimeSauc3 11 minutes ago
Commenting out old code that was messing with creating orders.
Reply...
Resolve conversation
LimeSauc3 reviewed 9 minutes ago
View reviewed changes
Reviewers
LimeSauc3
Still in progress? Convert to draft
Assignees
No one—assign yourself
Labels
None yet
Projects
None yet
Milestone
No milestone
Development
Successfully merging this pull request may close these issues.
None yet
Notifications
Customize
Unsubscribe
You're receiving notifications because you commented.



5) Self-check on best practices for security – ½ page

Asset	Types of possible attacks	Mitigation strategy
User Information	Password Breach	We make sure to have

	Unauthorized user gains access to confidential data	**encrypted passwords. Require users to authenticate themselves. Track system usage.**
Database Systems	**SQL Injection** Unauthorized user makes system unavailable	**We have parameterized queries so users can't input their own sql statements**

6) Self-check of the adherence to original Non-functional specs – performed by team leads

Current Development Status of Non-Functional Requirements

1	Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0.	DONE
2	Application shall be optimized for standard desktop/laptop browsers, e.g. must render correctly on the two latest versions of two major browsers.	DONE
3	All or selected application functions shall render well on mobile devices, but no mobile native app is to be developed.	DONE
4	Data shall be stored in the database on the team's deployment server.	DONE
5	No more than 50 concurrent users shall be accessing the application at any time.	DONE
6	Privacy of users shall be protected.	DONE
7	The language used shall be English (no localization needed).	DONE

8	Application shall be very easy to use and intuitive.	DONE
9	Application shall follow established architecture patterns.	DONE
10	Application code and its repository shall be easy to inspect and maintain.	DONE
11	Google analytics shall be used.	ON TRACK
12	No email clients shall be allowed. Interested users can only message to sellers via in-site messaging. One round of messaging (from user to seller) is enough for this application.	DONE
13	Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.	DONE
14	Site security: basic best practices shall be applied (as covered in the class) for main data items.	DONE
15	Media formats shall be standard as used in the market today.	DONE
16	Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.	DONE
17	The application UI (WWW and mobile) shall prominently display the following exact text on all pages "*SFSU Software Engineering Project CSC 648-848, Spring 2023. For Demonstration Only.*" at the top of the WWW page nav bar. (Important so as to not confuse this with a real application).	DONE

4) Product Screen Shots:

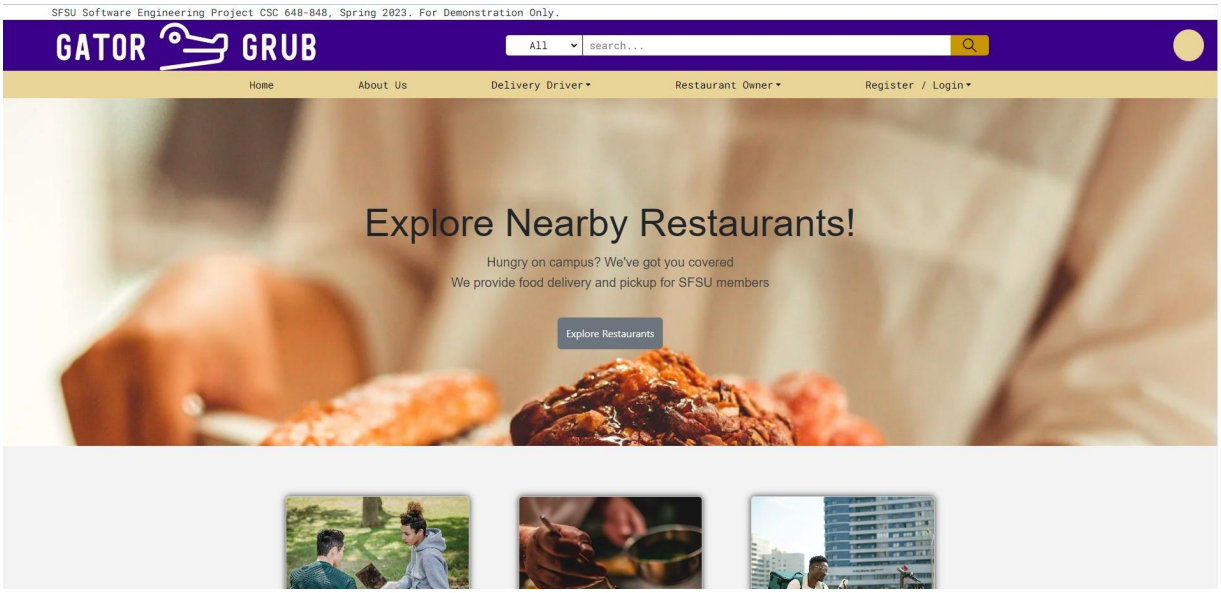


Figure. Home Page

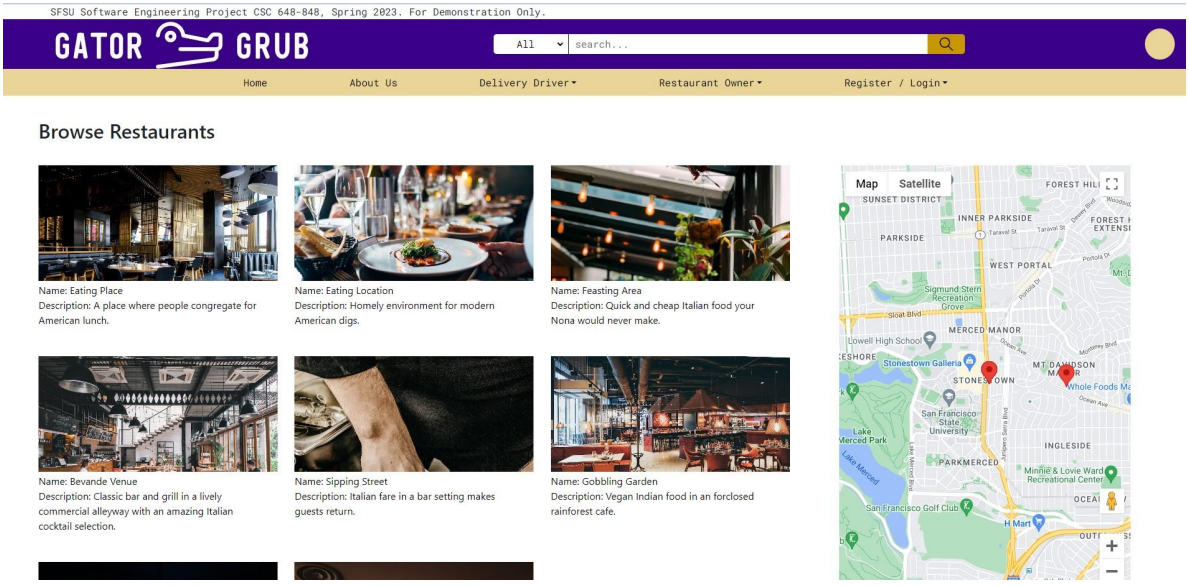


Figure. Explore Restaurants

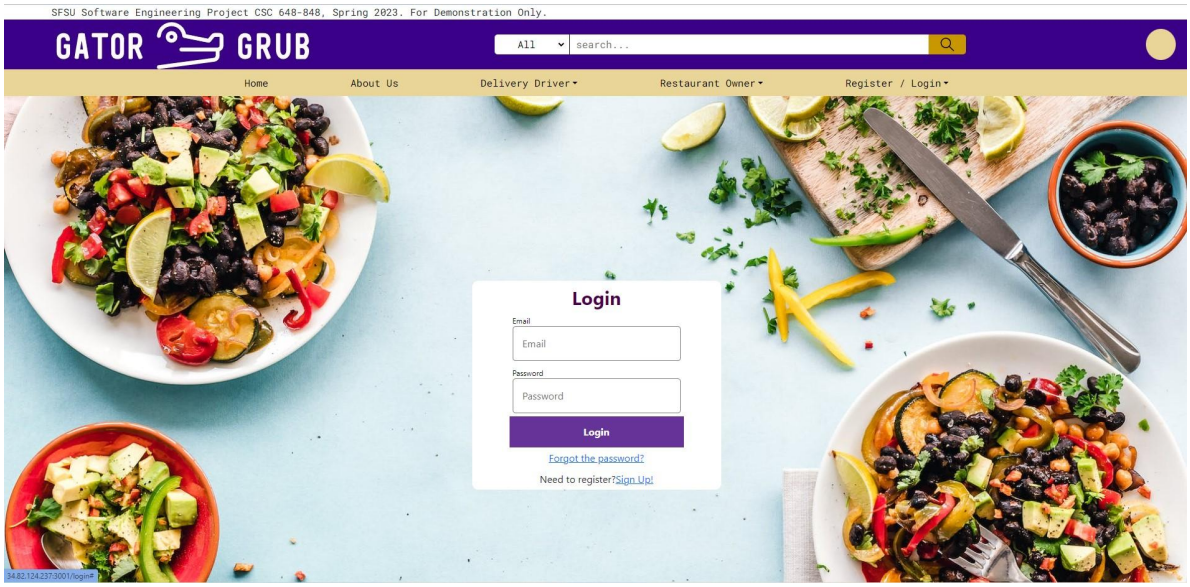


Figure. User's Login

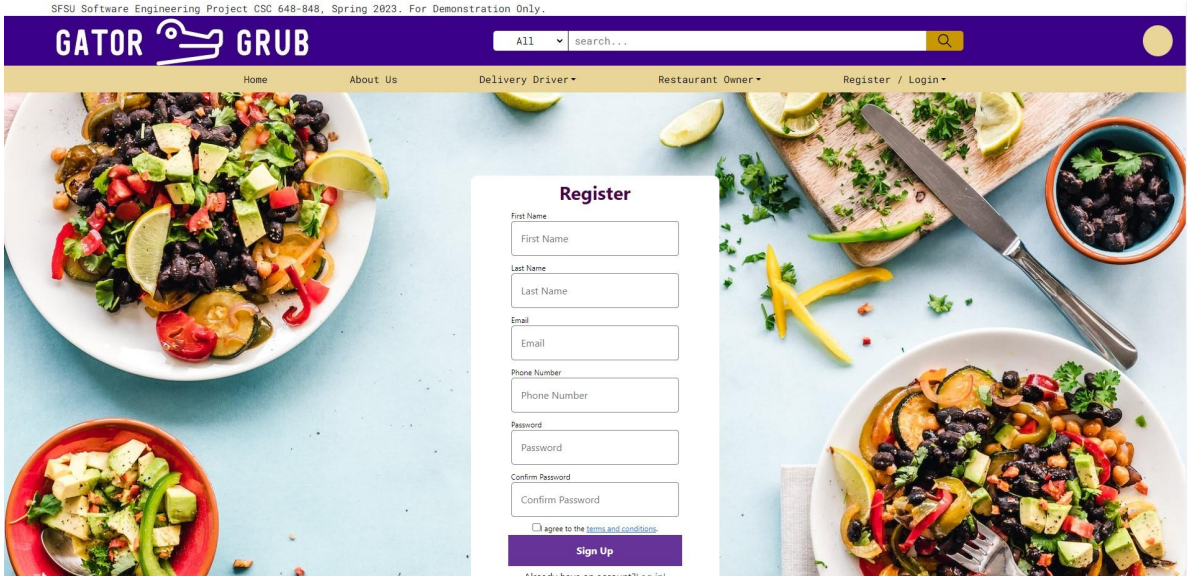


Figure. User's Register

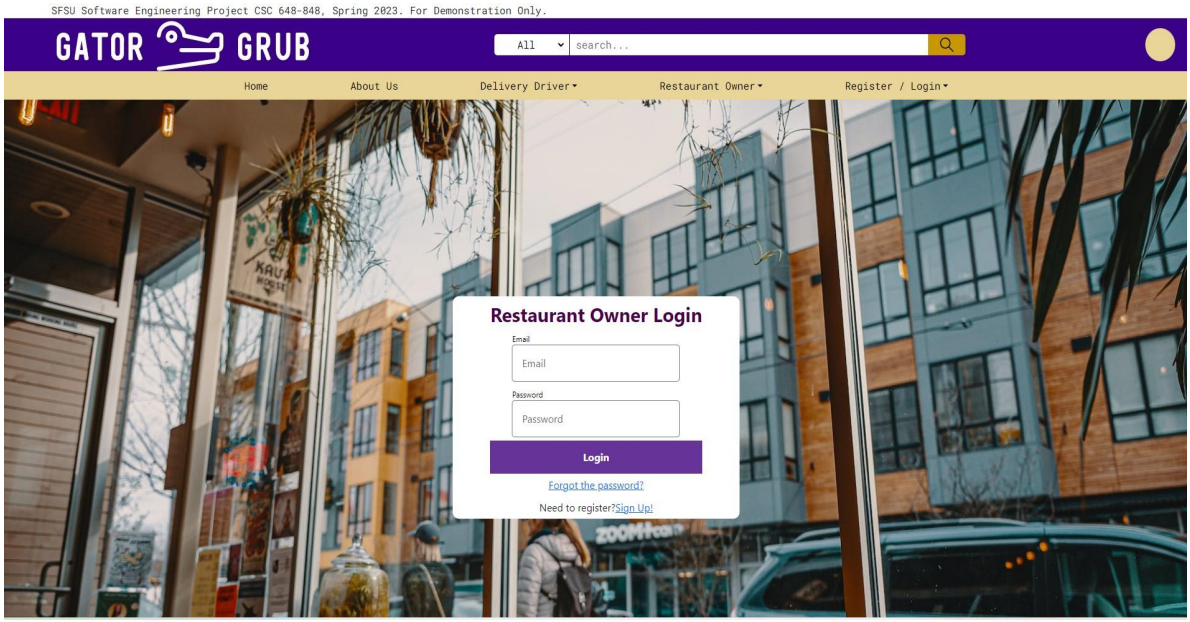


Figure. Restaurant Owner's Login

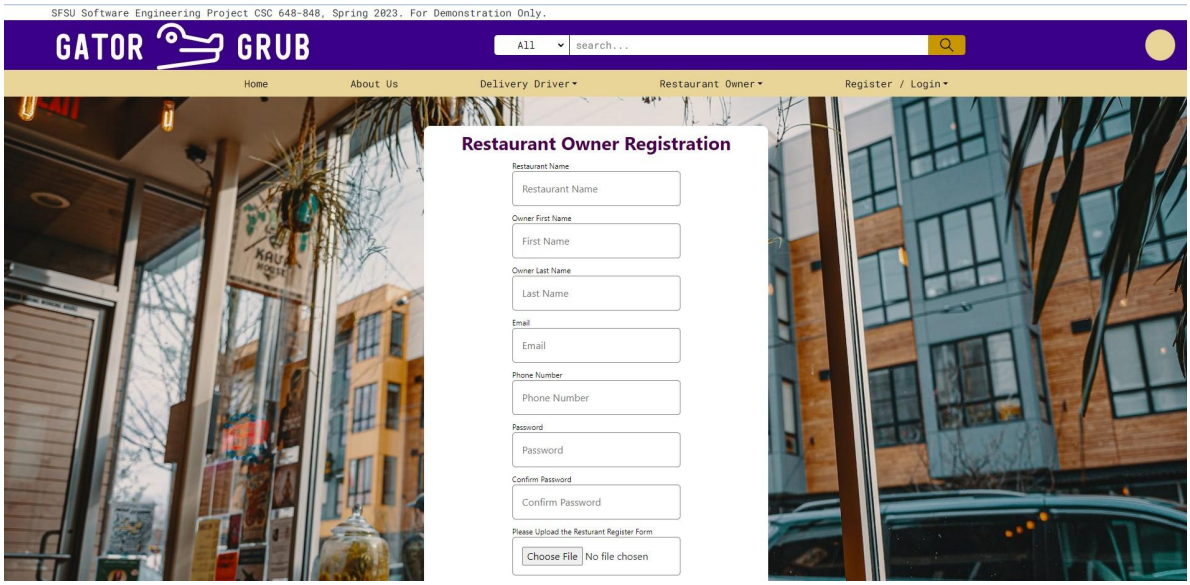


Figure. Restaurant Owner's Registration

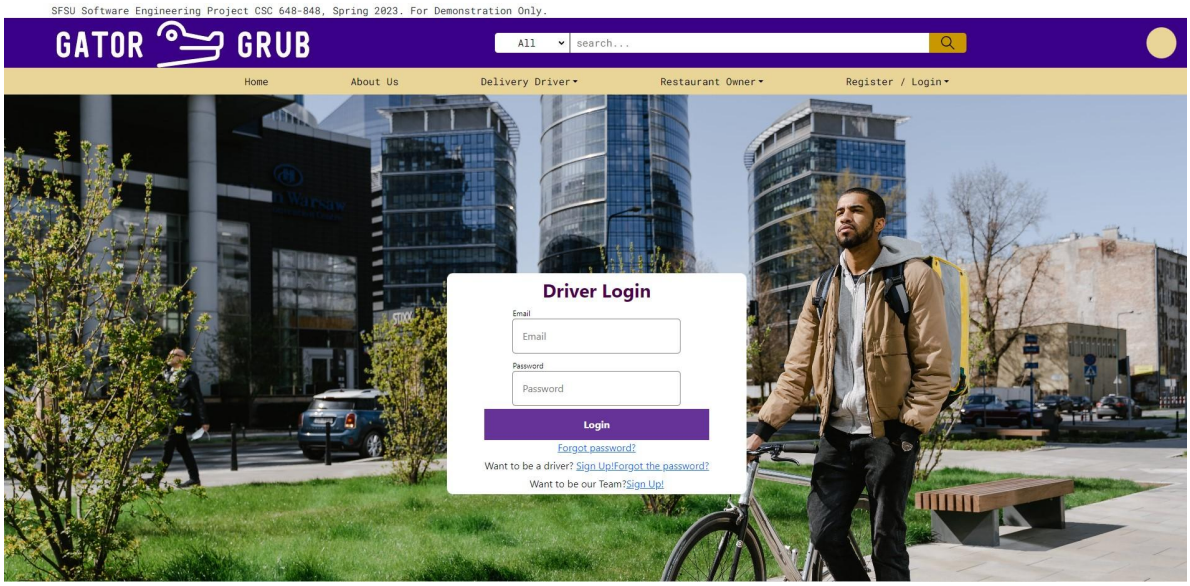


Figure. Delivery Driver's Login

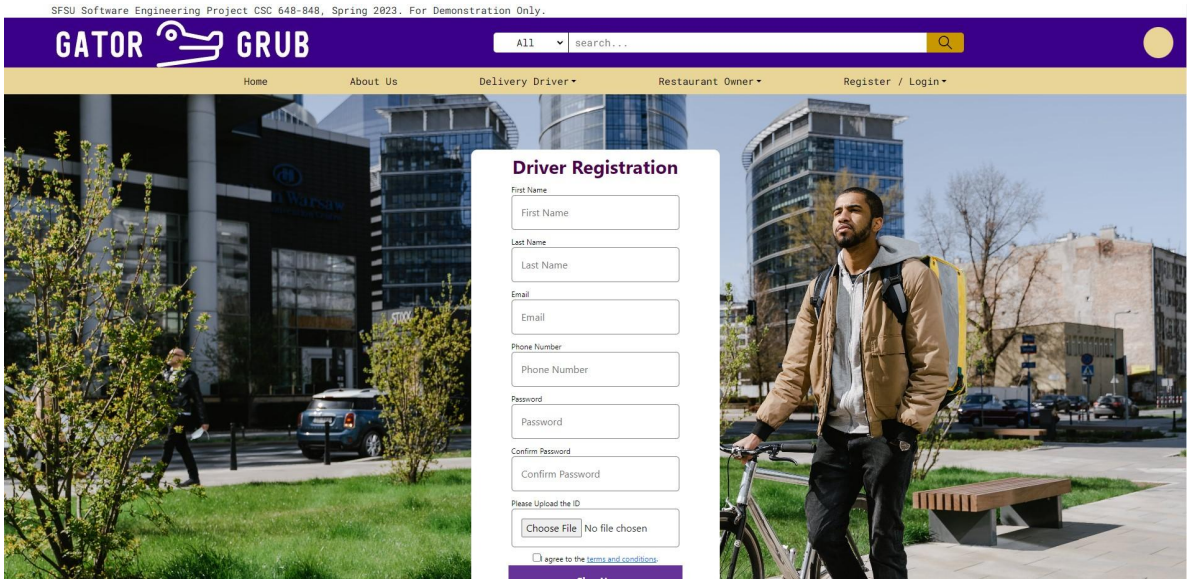


Figure. Delivery Driver's Registration

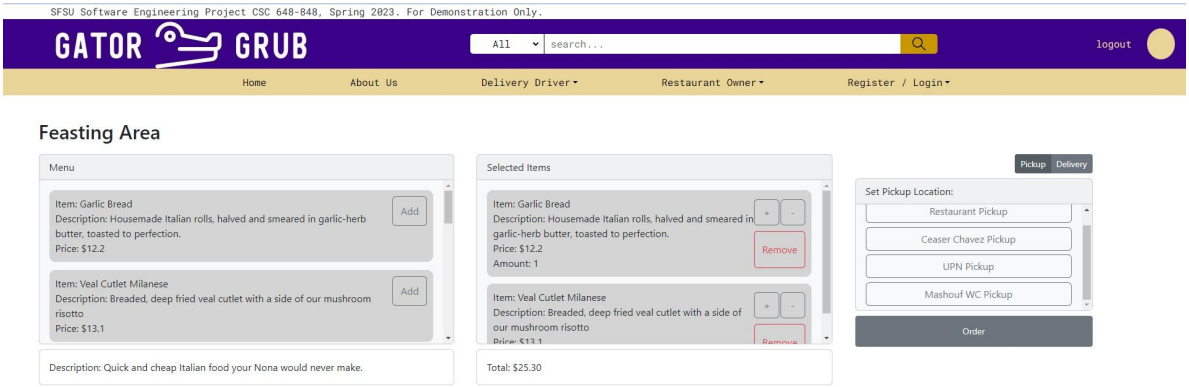


Figure. Restaurant Menu and Ordering interface

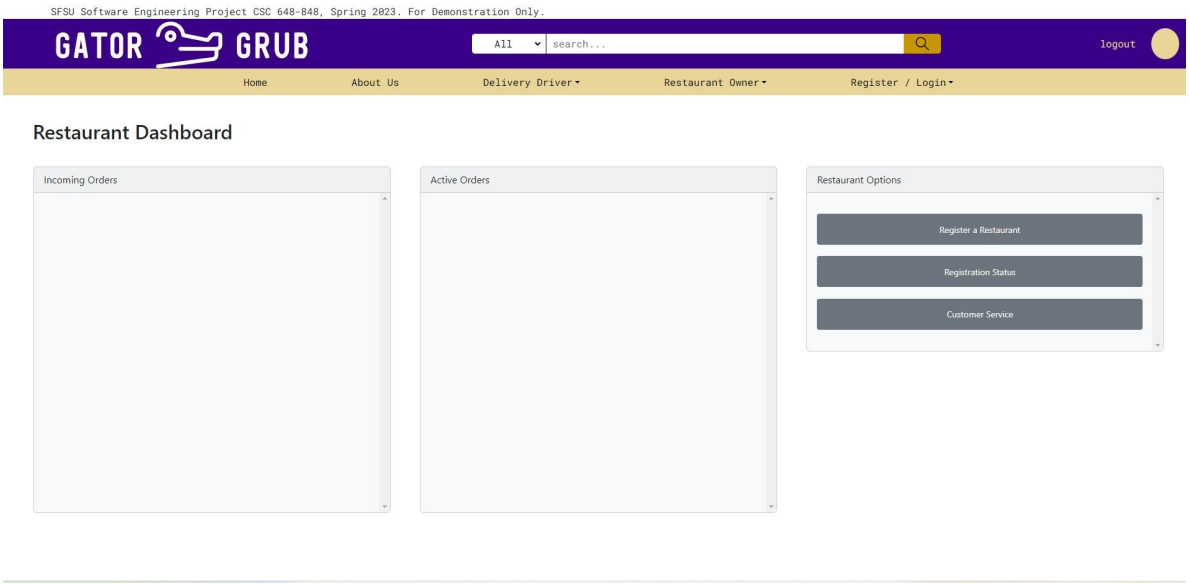


Figure. Restaurant Dashboard

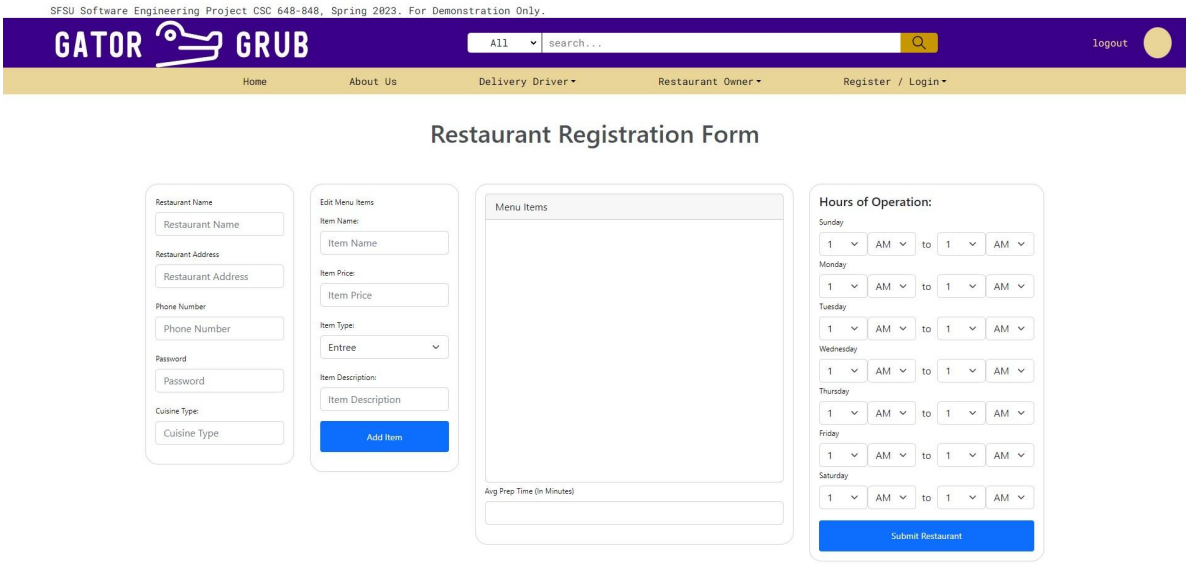


Figure. Restaurant Registration Form

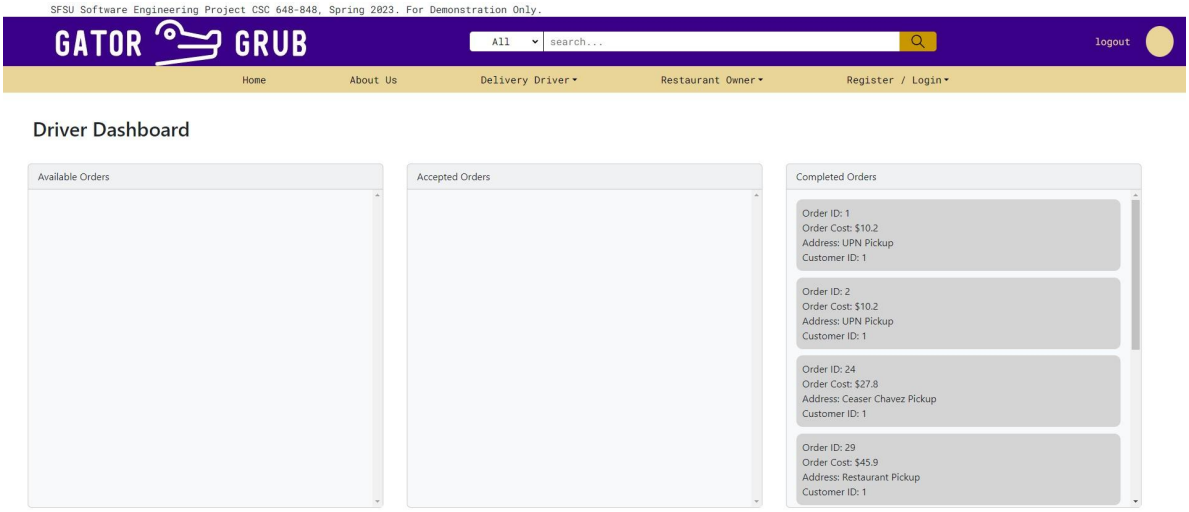


Figure. Driver Dashboard

5) Database Organization:

DDL for gatorGrubDB.SFSUCustomer

```
1   CREATE TABLE `SFSUCustomer` (
2     `SFSUCustomerID` int NOT NULL AUTO_INCREMENT,
3     `SFSUCustomerName` varchar(45) NOT NULL,
4     `SFSUCustomerEmail` varchar(45) NOT NULL,
5     `SFSUCustomerPhone` varchar(45) NOT NULL,
6     `SFSUCustomerPassword` varchar(60) NOT NULL,
7     PRIMARY KEY (`SFSUCustomerID`),
8     UNIQUE KEY `SFSUCustomerEmail_UNIQUE` (`SFSUCustomerEmail`),
9     UNIQUE KEY `SFSUCustomerID_UNIQUE` (`SFSUCustomerID`)
10  ) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8mb3
```

SFSUCustomerID	SFSUCustomerName	SFSUCustomerEmail	SFSUCustomerPhone	SFSUCustomerPassword
1	Balthazar McSquishy	mcsquishb@sfsu.edu	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
2	Gertrude Puddlesworth	puddle.gert@sfsu.edu	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
3	Barnaby McFluffernutter	nutfluff@sfsu.edu	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
4	Hortense Snickerdoodle	hors@sfsu.edu	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
5	Chuckleberry Finnegan	chuck.finn@sfsu.edu	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
6	Derrick	dl@sfsu.edu	123-456-7890	$2b$10$ZdORfxlEQx/yGw3I6WkfEeLenNlOcAn...
7	SDasdfa	asdf@sfsu.edu	111-222-3333	$2b$10$Fp0EL7/mlb13G1ZigVj5ZeyUuWH5gaIr...
9	Test	asdfasdf@sfsu.edu	111-222-3333	$2b$10$HAxa3vE8RFJQ3d50uHYtjOgF3z9Juh6...
11	James	jsauce@sfsu.edu	123-456-7890	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
12	D	dl1@sfsu.edu	123-213-1123	$2b$10$sJ1EMi67pi7TSC.wAhv9y.2zhju0cZ.uM...
NULL	NULL	NULL	NULL	NULL

Figure. SFSUCustomer DDL and current DB info

DDL for gatorGrubDB.Driver

```
1  CREATE TABLE `Driver` (
2    `DriverID` int NOT NULL AUTO_INCREMENT,
3    `DriverName` varchar(45) NOT NULL,
4    `DriverEmail` varchar(45) NOT NULL,
5    `DriverPhone` varchar(45) NOT NULL,
6    `DriverPassword` varchar(60) NOT NULL,
7    PRIMARY KEY (`DriverID`),
8    UNIQUE KEY `DriverID_UNIQUE` (`DriverID`),
9    UNIQUE KEY `DriverEmail_UNIQUE` (`DriverEmail`)
10 ) ENGINE=InnoDB AUTO_INCREMENT=13 DEFAULT CHARSET=utf8mb3
```

DriverID	DriverName	DriverEmail	DriverPhone	DriverPassword
4	Rolf Kleinfeldt	rolfkleinfeldt@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
5	Gabriella Tong	gabbytong@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
6	Ana Warner	anawarner@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
7	Hadas Curry	hadascurry@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
8	Bea Joosten	beajoosten@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
9	Anton Haight	antonhaight@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
10	Tonio Stenger	toniostenger@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
11	Catrina Oliver	catrinaoliver@gmail.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...
12		resttest1@gmail.com	111-222-3333	$2b$10$C9RCRf.b0609r1/SVwx4C.po3bkZ7xU...
NULL	NULL	NULL	NULL	NULL

Figure. Driver DDL and current DB info

DDL for gatorGrubDB.RestaurantOwner

```
1  CREATE TABLE `RestaurantOwner` (
2    `RestaurantOwnerID` int NOT NULL AUTO_INCREMENT,
3    `RestaurantOwnerName` varchar(45) NOT NULL,
4    `RestaurantOwnerEmail` varchar(45) NOT NULL,
5    `RestaurantOwnerPhone` varchar(45) NOT NULL,
6    `RestaurantOwnerPassword` varchar(60) NOT NULL,
7    `RestaurantID` int NOT NULL DEFAULT '0',
8    PRIMARY KEY (`RestaurantOwnerID`),
9    UNIQUE KEY `RestaurantOwnerEmail_UNIQUE` (`RestaurantOwnerEmail`),
10   UNIQUE KEY `RestaurantOwnerID_UNIQUE` (`RestaurantOwnerID`)
11 ) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=utf8mb3
```

RestaurantOwnerID	RestaurantOwnerName	RestaurantOwnerEmail	RestaurantOwnerPhone	RestaurantOwnerPassword	RestaurantID
1	Bob Smith	bobsmith@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	1
2	Billy Smith	billysmith@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	2
3	Billy Johnson	billyjohnson@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	3
4	Greg Elliot	gregelliot@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	4
5	Sarah Sanders	sarahsanders@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	5
6	Joe Rogen	joerogen@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	6
7	Johnjacobjingleheimer Schmidt	johnjacobjingleheimerschmidt@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	7
8	Joanna Zhang	joannazhang@sfsu.com	1231231234	$2b$10$Ep8fhLqqccwj/DQRRWnOyO7cDa1wW...	8
NULL	NULL	NULL	NULL	NULL	NULL

Figure. RestaurantOwner DDL and current DB info

DDL for gatorGrubDB.Restaurant

```
CREATE TABLE `Restaurant` (
  `RestaurantID` int NOT NULL AUTO_INCREMENT,
  `RestaurantOwnerID` int NOT NULL,
  `flag` tinyint NOT NULL DEFAULT '0',
  `RestaurantName` varchar(45) NOT NULL,
  `RestaurantPhone` varchar(45) NOT NULL,
  `RestaurantPassword` varchar(45) NOT NULL,
  `RestaurantAddress` varchar(75) NOT NULL,
  `RestaurantCuisine` int NOT NULL,
  `RestaurantPriceTier` int NOT NULL,
  `RestaurantHours` varchar(45) NOT NULL,
  `RestaurantPrepTime` int NOT NULL,
  `RestaurantImage` varchar(45) NOT NULL,
  `RestaurantCoordinates` varchar(45) NOT NULL,
  `RestaurantDescription` varchar(250) DEFAULT NULL,
  PRIMARY KEY (`RestaurantID`),
  UNIQUE KEY `RestaurantID_UNIQUE` (`RestaurantID`),
  UNIQUE KEY `RestaurantName_UNIQUE` (`RestaurantName`),
  KEY `r_owner_fk_idx` (`RestaurantOwnerID`),
  KEY `CuisineID_idx` (`RestaurantCuisine`),
  CONSTRAINT `CuisineID` FOREIGN KEY (`RestaurantCuisine`) REFERENCES `Cuisine` (`CuisineID`),
  CONSTRAINT `r_owner_fk` FOREIGN KEY (`RestaurantOwnerID`) REFERENCES `RestaurantOwner` (`RestaurantOwnerID`)
) ENGINE=InnoDB AUTO_INCREMENT=9 DEFAULT CHARSET=utf8mb3
```

RestaurantID	RestaurantOwnerID	flag	RestaurantName	RestaurantPhone	RestaurantPassword	RestaurantAddress	RestaurantCuisine
1	1	1	Eating Place	1234567890	password1	155 Winston Dr, San Francisco, CA 94132	1
2	2	1	Eating Location	1234567890	password2	155 Winston Dr, San Francisco, CA 94132	1
3	3	1	Feasting Area	1234567890	password3	155 Winston Dr, San Francisco, CA 94132	2
4	4	1	Bevande Venue	1234567890	password4	155 Winston Dr, San Francisco, CA 94132	2
5	5	1	Sipping Street	1234567890	password5	155 Winston Dr, San Francisco, CA 94132	2
6	6	1	Gobbling Garden	1234567890	password6	155 Winston Dr, San Francisco, CA 94132	3
7	7	1	Stuffing Square	1234567890	password7	155 Winston Dr, San Francisco, CA 94132	3
8	8	1	Dimsum Dimension	1234567890	password1	155 Winston Dr, San Francisco, CA 94132	4
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

RestaurantPriceTier	RestaurantHours	RestaurantPrepTime	RestaurantImage	RestaurantCoordinates	RestaurantDescription
3		25	./src/images/restaurant/12345	37.7266,-122.4744	A place where people congregate for American l...
5		25	./src/images/restaurant/12345	37.7266,-122.4744	Homely environment for modern American digs.
1		25	./src/images/restaurant/12345	37.7266,-122.4744	Quick and cheap Italian food your Nona would n...
2		25	./src/images/restaurant/12345	37.7266,-122.4744	Classic bar and grill in a lively commercial alleyw...
5		25	./src/images/restaurant/12345	37.7266,-122.4744	Italian fare in a bar setting makes guests return.
4		25	./src/images/restaurant/12345	37.7266,-122.4744	Vegan Indian food in an forclosed rainforest cafe.
2		25	./src/images/restaurant/12345	37.7266,-122.4744	This Indian spot in an outdoor plaza serves pani...
3		25	./src/images/restaurant/12345	37.726233, -122.463763	Classic spot for traditional yum cha on a cozy th...
NULL	NULL	NULL	NULL	NULL	NULL

Figure. Restaurant DDL and current DB info

6) Github organization:

Shauhin is our Github master and serves as the main administrator and maintainer for our main branch github repository. Other team members do not have permission to make changes or merge requests on the main repository. Instead, Shauhin will be the only one responsible for merging pull requests into the main branch. In this process, code is submitted via pull request, then it's reviewed by the Github master where it's either accepted and the merge is instantly approved or it is rejected with comments for revisions. This ensures that our codebase remains organized and that changes are properly reviewed and approved before being merged into the main repository. Subteam leads may have to merge changes if the Github master is unavailable.

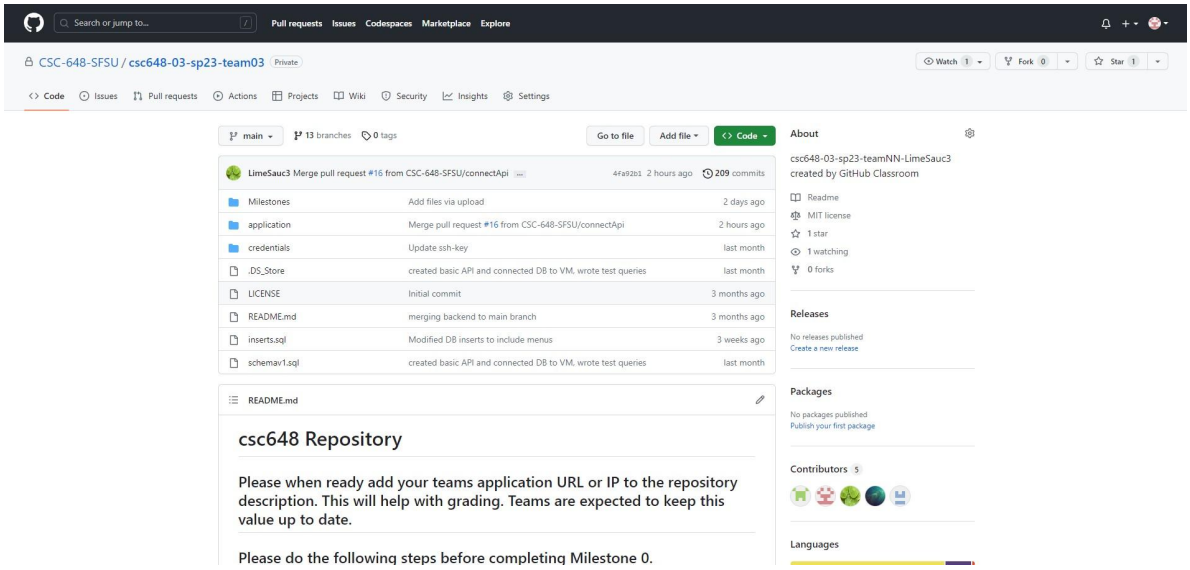


Figure. Github Home Page

7) Google analytics stats plot for your WWW site (1 page)

8) Project management:

For our project we were able to be effective as a team by two primary methods of task tracking: Discord and Trello. Discord is our primary means of communication as a team. Discord is an app which holds private message boards, online meeting rooms and various methods to organize, divide and manage teams. It is used as a place for the team to discuss the project either through the server's text or voice channels. This is where important text documentation can quickly be posted for the entire team to view or a place for private messages between users.

Every meeting is concluded with a post to the team-meetings text channel by the team lead with details on what exactly was discussed, where we were and where we had to be. Before tasks are assigned on Trello, they're announced on Discord by the team lead, so that everyone knows what to expect and that subteams (front-end and back-end) can keep track of their own tasks. After that, tasks are loaded onto Trello boards with different stages of development, ranging from "To-Do" to "Complete for Submission".

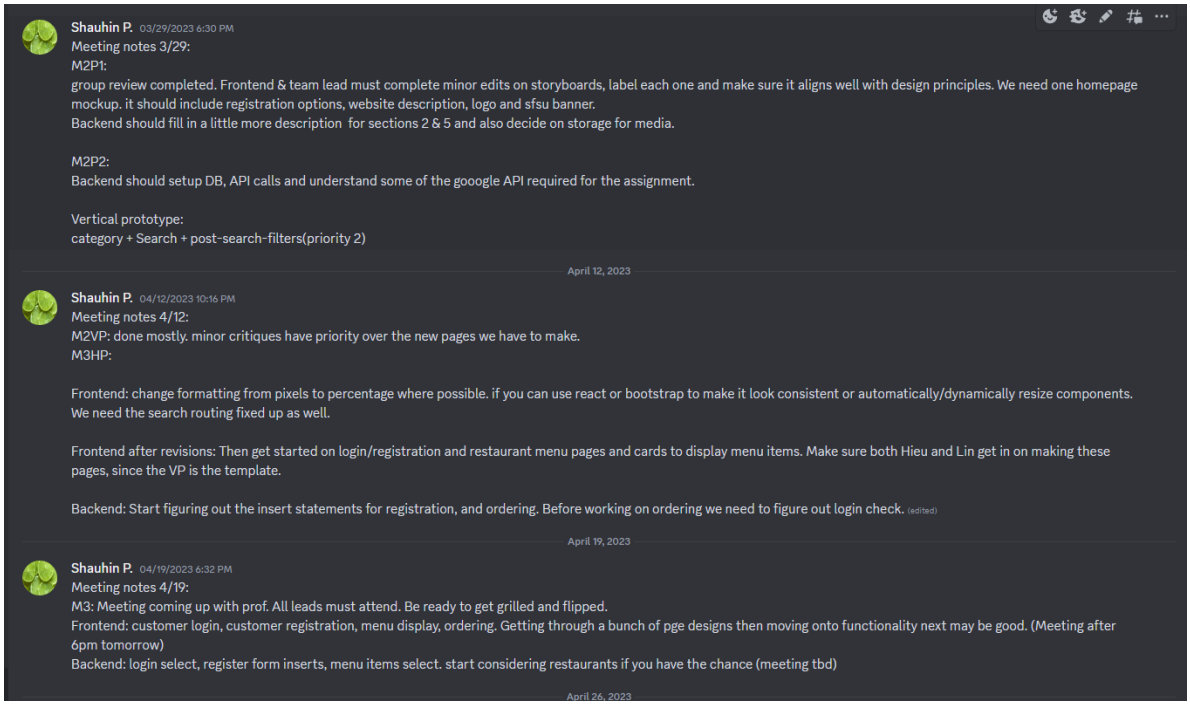


Figure. #meeting-notes text channel on Team 3 Discord, Posting meeting notes

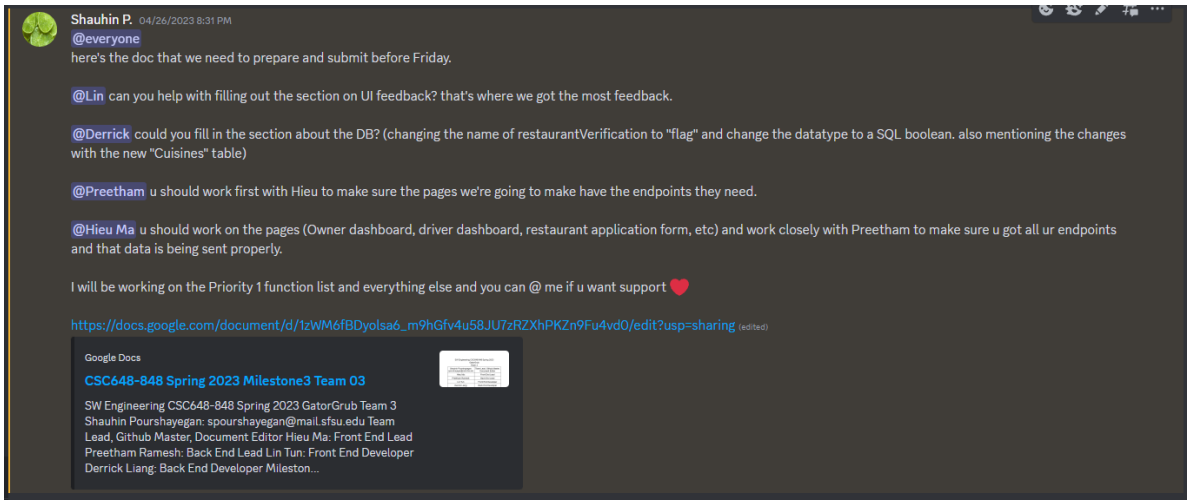


Figure. #general text channel on Team 3 Discord, Assigning tasks

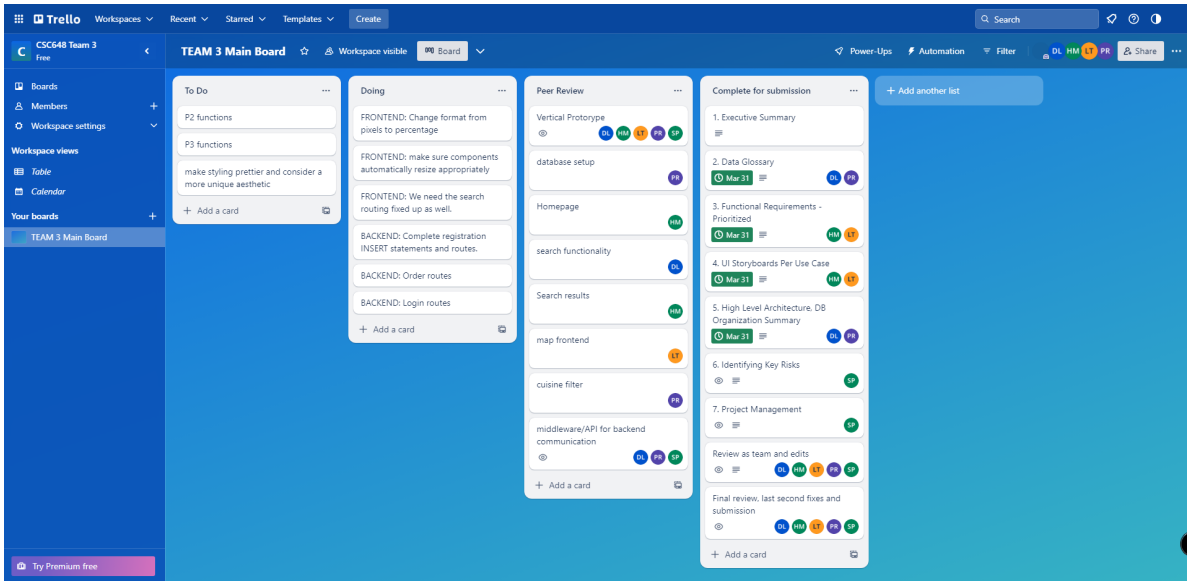
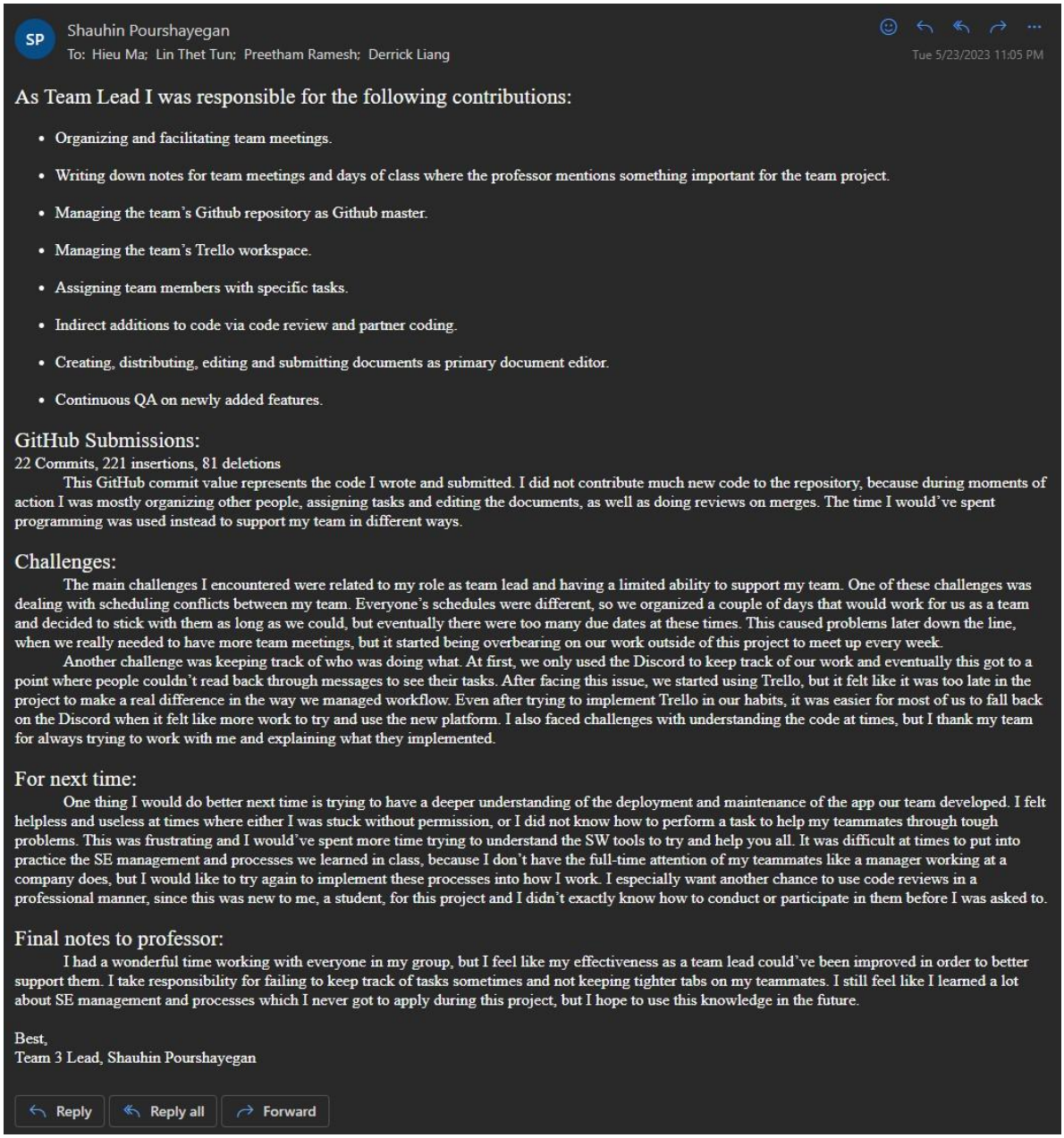


Figure. TEAM 3 Main Board Trello

9) Team member self assessment and contributions:

Shauhin's email:



SP Shauhin Pourshayegan
To: Hieu Ma; Lin Thet Tun; Preetham Ramesh; Derrick Liang
Tue 5/23/2023 11:05 PM

As Team Lead I was responsible for the following contributions:

- Organizing and facilitating team meetings.
- Writing down notes for team meetings and days of class where the professor mentions something important for the team project.
- Managing the team's Github repository as Github master.
- Managing the team's Trello workspace.
- Assigning team members with specific tasks.
- Indirect additions to code via code review and partner coding.
- Creating, distributing, editing and submitting documents as primary document editor.
- Continuous QA on newly added features.

GitHub Submissions:
22 Commits, 221 insertions, 81 deletions
This GitHub commit value represents the code I wrote and submitted. I did not contribute much new code to the repository, because during moments of action I was mostly organizing other people, assigning tasks and editing the documents, as well as doing reviews on merges. The time I would've spent programming was used instead to support my team in different ways.

Challenges:
The main challenges I encountered were related to my role as team lead and having a limited ability to support my team. One of these challenges was dealing with scheduling conflicts between my team. Everyone's schedules were different, so we organized a couple of days that would work for us as a team and decided to stick with them as long as we could, but eventually there were too many due dates at these times. This caused problems later down the line, when we really needed to have more team meetings, but it started being overbearing on our work outside of this project to meet up every week.
Another challenge was keeping track of who was doing what. At first, we only used the Discord to keep track of our work and eventually this got to a point where people couldn't read back through messages to see their tasks. After facing this issue, we started using Trello, but it felt like it was too late in the project to make a real difference in the way we managed workflow. Even after trying to implement Trello in our habits, it was easier for most of us to fall back on the Discord when it felt like more work to try and use the new platform. I also faced challenges with understanding the code at times, but I thank my team for always trying to work with me and explaining what they implemented.

For next time:
One thing I would do better next time is trying to have a deeper understanding of the deployment and maintenance of the app our team developed. I felt helpless and useless at times where either I was stuck without permission, or I did not know how to perform a task to help my teammates through tough problems. This was frustrating and I would've spent more time trying to understand the SW tools to try and help you all. It was difficult at times to put into practice the SE management and processes we learned in class, because I don't have the full-time attention of my teammates like a manager working at a company does, but I would like to try again to implement these processes into how I work. I especially want another chance to use code reviews in a professional manner, since this was new to me, a student, for this project and I didn't exactly know how to conduct or participate in them before I was asked to.

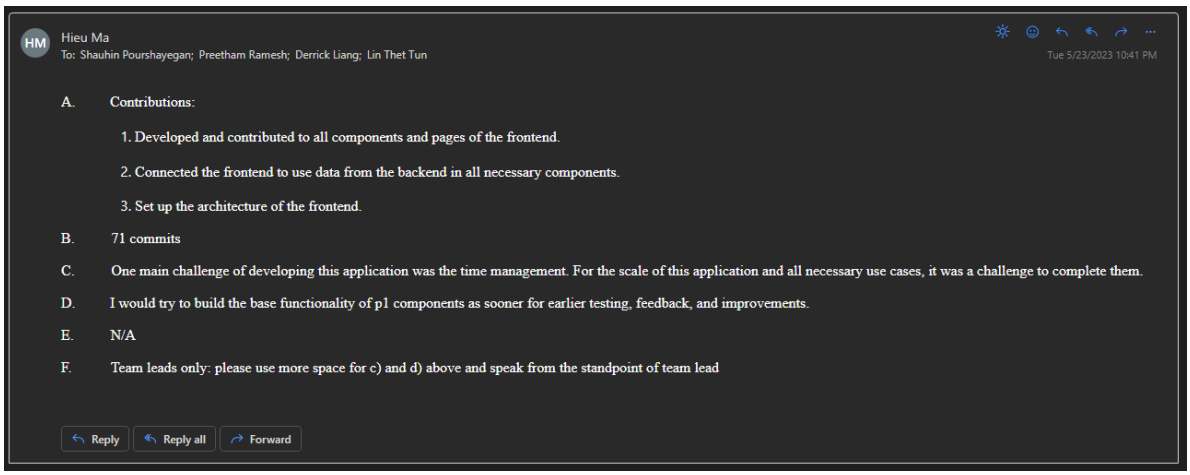
Final notes to professor:
I had a wonderful time working with everyone in my group, but I feel like my effectiveness as a team lead could've been improved in order to better support them. I take responsibility for failing to keep track of tasks sometimes and not keeping tighter tabs on my teammates. I still feel like I learned a lot about SE management and processes which I never got to apply during this project, but I hope to use this knowledge in the future.

Best,
Team 3 Lead, Shauhin Pourshayegan

Reply Reply all Forward

Preetham's email:

Heiu's email:



Hieu Ma
To: Shauhin Pourshayegan; Preetham Ramesh; Derrick Liang; Lin Thet Tun
Tue 5/23/2023 10:41 PM

A. Contributions:

1. Developed and contributed to all components and pages of the frontend.
2. Connected the frontend to use data from the backend in all necessary components.
3. Set up the architecture of the frontend.

B. 71 commits

C. One main challenge of developing this application was the time management. For the scale of this application and all necessary use cases, it was a challenge to complete them.

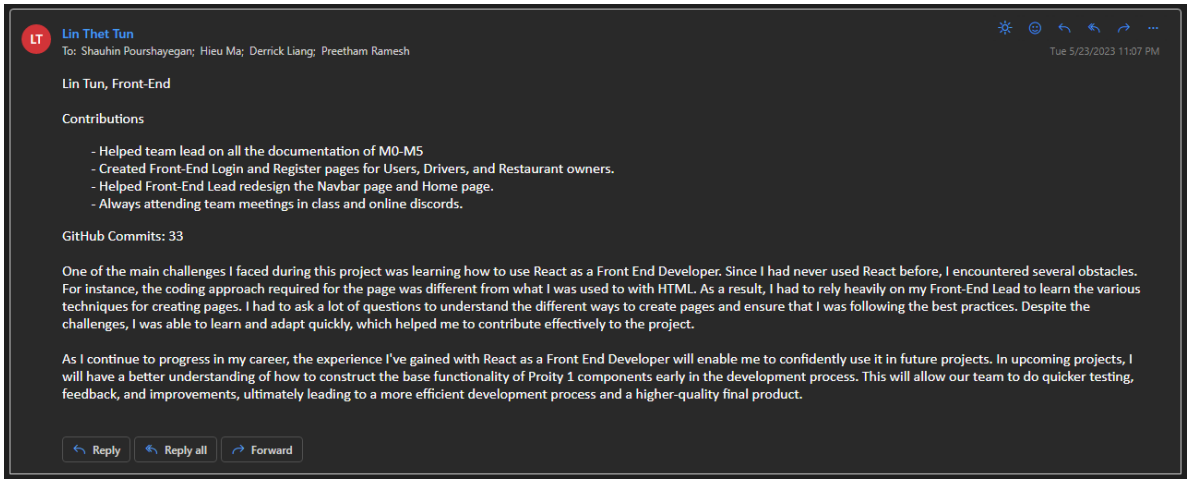
D. I would try to build the base functionality of p1 components as sooner for earlier testing, feedback, and improvements.

E. N/A

F. Team leads only: please use more space for c) and d) above and speak from the standpoint of team lead

Reply Reply all Forward

Lin's email:



Lin Thet Tun
To: Shauhin Pourshayegan; Hieu Ma; Derrick Liang; Preetham Ramesh
Tue 5/23/2023 11:07 PM

Lin Tun, Front-End

Contributions

- Helped team lead on all the documentation of M0-M5
- Created Front-End Login and Register pages for Users, Drivers, and Restaurant owners.
- Helped Front-End Lead redesign the Navbar page and Home page.
- Always attending team meetings in class and online discords.

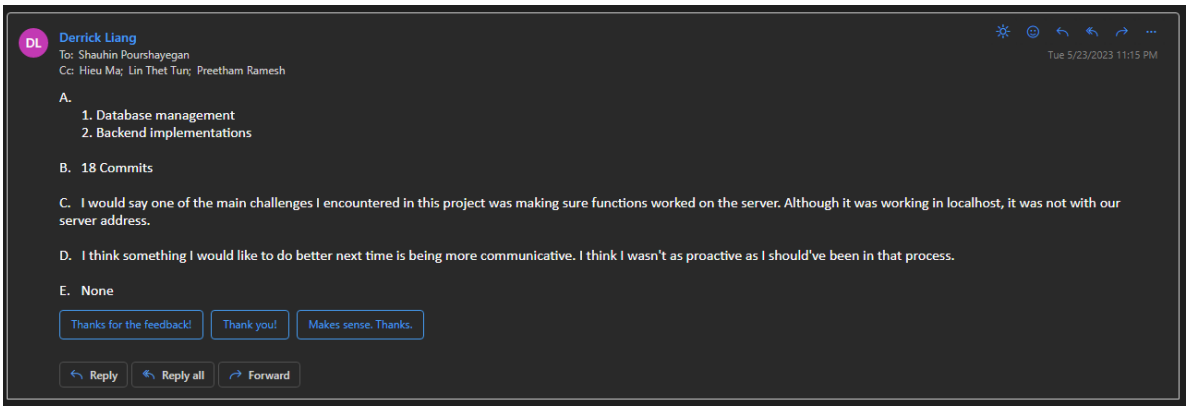
GitHub Commits: 33

One of the main challenges I faced during this project was learning how to use React as a Front End Developer. Since I had never used React before, I encountered several obstacles. For instance, the coding approach required for the page was different from what I was used to with HTML. As a result, I had to rely heavily on my Front-End Lead to learn the various techniques for creating pages. I had to ask a lot of questions to understand the different ways to create pages and ensure that I was following the best practices. Despite the challenges, I was able to learn and adapt quickly, which helped me to contribute effectively to the project.

As I continue to progress in my career, the experience I've gained with React as a Front End Developer will enable me to confidently use it in future projects. In upcoming projects, I will have a better understanding of how to construct the base functionality of Proity 1 components early in the development process. This will allow our team to do quicker testing, feedback, and improvements, ultimately leading to a more efficient development process and a higher-quality final product.

Reply Reply all Forward

Derrick's email:



Length of all of the above is max¾ of a page per team member (except for the team lead).

NOTE: In this section you must submit copies of raw (original) e-mails sent from each team member to all other team members also showing e-mail address info (sender and recipients). Screen shots OK but make them readable. Team members who fail to submit their self assessment and contributions will receive minimum 5 point reduction of their own team grade

Submission of M5 folder - each team please give hard copy bound document (one physical folder) to instructor just before your demo

Good luck!!!!!!!