Final Project for SW Engineering CSC648-848 Spring 2021

Team 01

HungryGators-19

Food Delivery and Restaurant Lookup Service

Available on Mobile and Desktop

Branden J. Nguyen – Team Lead – bnguye11@mail.sfsu.edu
Jasmine Jahan – Front End Lead
Yongjian Pan – Back End Lead
Robert Freeman – GitHub Master
Gurjot Singh – Team Member Front End
Tin Thu Zar Aye – Team Member Back End

URL of the demo: http://18.223.160.96:5002/

May 22, 2021

Table of Contents

I. Product Summary

HungryGators-19 is the name of our food delivery and restaurant lookup web application.

1. Homepage: This is the main page of the web application that contains a navigation bar with links that guide you to the other pages, as well as an integrated search bar. The web application name and logo are at the top, along with some marketing graphics.
2. Registration: SFSU students/staff/faculty, restaurant owners, and delivery drivers can create an account if you do not have one already.
3. Login: There are three different logins for SFSU students/staff/faculty (default), restaurant owners, and delivery drivers. You must be an SFSU student/staff/faculty to be able to place a food order. Restaurant owners can login and add menu items. Delivery drivers can login and view and accept restaurant orders to be delivered.
4. Search bar: Allows you to search for different food items or restaurants displayed in the search results.
5. Drop down tab: Allows you to filter your search results based on cuisine type.
6. Restaurant menus: Allows you to view the specific menu items for each restaurant and add these items to a cart.
7. Cart: Allows you to view the current order's items and total price and place the order.
8. Driver pickup, map of the campus: Delivery drivers can view all active orders placed with delivery details and choose an order to deliver. A delivery map is provided to assist with the delivery directions.
9. Upload restaurant: Restaurant owners can also add menu items to their restaurant(s) to be ordered.

URL to the application: http://18.223.160.96:5002/

II. A. Milestone 1 Document

SW Engineering CSC648/848 Spring 2021

HungryGators-19

Food Delivery and Restaurant Review Service

Available on Mobile and Desktop

Team 01

Branden J. Nguyen – Team Lead – bnguye11@mail.sfsu.edu
Jasmine Jahan – Front End Lead
Yongjian Pan – Back End Lead
Robert Freeman – GitHub Master
Gurjot Singh – Team Member Front End
Tin Thu Zar Aye – Team Member Back End

Milestone 1

March 5, 2021

Date Submitted	Date Revised
March 5, 2021	March 12, 2021

Table of Contents

1. Executive Summary:

San Francisco State University is a place of cultural diversity. There are many people including students, faculty, and staff from different parts of the world whose food habits are also different from one another. Though there are a variety of cafeterias and canteens in SFSU, they are not available for online delivery in the campus area. Most of the time students must go to the cafeteria to get food. This can be a hassle especially if students, faculty, and staff are having food in between classes, in which case, they may not have time to walk to the cafeteria, order the food, wait and be late for class. As well, members of SFSU must select food at their own risk without reading any reviews or ratings which could be misleading. There are some other applications to get food delivered from larger restaurants, but they are only available outside the SFSU campus area. The process is inconvenient and hard to navigate.

We, Team 01, are building a web-application named HungryGators-19 by which students, faculty and staff can view and leave credible reviews for restaurants, as well as order food to be delivered at SFSU from these establishments straight to their classroom door or dorm securely and efficiently. With this food delivery service students, faculty and staff will be able to order food by creating an account with their SFSU.edu email and name. If the person does not belong to the university, they cannot order from the app. For the person who is unable to use the app, they can get the phone number of the restaurants from the app and make a pickup order. The minimum viable product of our internet application can be described as a restaurant review and delivery service solely catered to San Francisco State University students, staff, and faculty. This app will not only solve the problem of convenient food delivery for members of SFSU who all have drastically varying schedules, but it will also save invaluable time which people within the university setting are always short of. Moreover, this web-application is a fantastic platform for new or established restaurants looking to spread their customer base and garner new business opportunities.

The application capitalizes on the vastly growing food delivery industry, as well as the ever-growing population of the SFSU community. The cornerstone of good execution of this business is the great variety of food options to choose from, and the convenient and safe delivery of the food. We would exceed our competitors by not only serving exclusively SFSU students, but also capitalize on the unique niche of visitors of the campus, which are

otherwise an untapped resource. With an emphasis on specific delivery drop-off locations, ease-of-use, as well as a great variety of restaurants, this service will appeal greatly to the university community. HungryGators-19 is an application which will revolutionize food delivery service for students at San Francisco State University.

II. Personae and Main Use Cases:

Personae:

John Full-Time staff at SFSU	About John: · He is a busy full-time staff at SFSU · He rarely cooks at home and he loves to eat the food from restaurants. Goals and Scenario: · Since is busy full time staff at SFSU, he orders the food from the app at least 4 times a week. · Every time he orders the food, he is more concerned about the rating of the restaurant than what kind of meal he wants because he thinks that the more rating the restaurant gets, the more tasty the food is. · Therefore, every time he orders, he is always looking for new suggestions about the restaurant deponse on his location.

Diane Full-Time student at SFSU	About Diane: · She is a full-time student at SFSU. · She goes to school all day and she works part time. · She doesn't have time to cook. · She orders the meal through the app regularly. Goals and Scenario: · Since she is a full-time student and works part-time, she doesn't have time to cook. · She orders the food through the app regularly, she becomes a premium member of the app. · Sometimes, she gets the coupons to get the discount. · Sometimes, she also gets the delivery fees waive. · Whenever she orders the meal, she is more concerned about the price than the food and nutrition.

Jack Full-Time Faculty at SFSU	About Jack: · He is a full time SFSU's faculty. · He works from 8:00am to 5:00pm · He is always ordering the food from the app for his breakfast and lunch because he doesn't have time to prepare in the morning. Goals and Scenario: · Since he is a SFSU's faculty, he gets not only a 10% discount but he also gets free delivery whenever he orders the food from the app. · He becomes the premium member of the app because he uses the app everyday.

Leo Admin of HungryGator-19	About Leo: · He is a SFSU student. · He also works as an admin of HungryGator-19. · He orders the food from the app. . He is a full time student and also works at school, he always orders the food from the app. Goals and Scenario: · Since he is not only SFSU's student but he also works as an admin of the app, he can easily order food from the app. · He can order the food as an admin of the app. . Since he is an admin of the app, he can manage the app. . He needs to review and approve the appropriate user of the app.

Jose Student/Driver of the HungryGator-19	About Jose: . He is not only student at SFSU but he also works as a driver of HungryGator-19's app. . He rarely cooks at home since he is busy with school and work. . He always orders the food from the app. Goals and Scenario: . Since he is working as a delivery-man for the app, he can easily order the food from the app by using his sfsu.edu account. . He is a delivery driver so he can easily access the SFSU campus' map

	. He can also able to view the order details from Restaurant food delivery to Customers.

Mary Retired faculty Of SFSU	About Mary: · She is a 70 years-old grandmother. · She used to work at SFSU. · She wants to order the pizza for her lunch. Goals and Scenario: · She rarely orders food through the app, she only has basic skills of using the app for ordering the food. · After she finds the restaurant, she looks through the menu and she wants to add in the cart but she isn't sure how to do it. · She just orders through the phone from the nearest restaurant and picks up the food.

Liam Undergraduate Student at SFSU	About Liam: · He is an undergraduate student at SFSU. · He is a vegetarian. · He is busy with his class schedule for the whole day. Goals and Scenario: · He gets time 1pm-2pm to eat lunch. · Since he is a vegetarian, he searches the vegetarian restaurant from the app to order lunch. · He gets only one hour to eat lunch, so he looks for the nearest vegetarian restaurant through the app. · It takes only 20 mins to get the food.

Calvin Restaurant owner at SFSU	About Calvin: . He is one of the restaurant owners at SFSU. . He is neither SFSU student nor SFSU staff. . He orders food from different restaurants at SFSU sometimes. Goals and Scenario: . He orders the food twice a week from the app. . When he orders the food from the app, he doesn't need to be a SFSU Student or he doesn't need to be a staff at SFSU. . He can just simply login as one of the SFSU's restaurant owners and order the food from the app. . He can remove and add the menu.

Main Use-Cases:

John: Full Time Student at SFSU

John is a busy full-time staff member at SFSU and he likes to order food through the app at least 4-5 times a week. Whenever he orders the food, he always looks at the review section and how many stars the restaurant gets. When he finds the restaurant which has good reviews, he orders right away without looking at the price. Moreover, he also likes to write the review and gives the rating after he tastes the food. Sometimes, when he writes the review, he forgets to login the account. As soon as he tried to submit, the application notified him to login the account to write the review. He always looks for the new suggestion whenever he orders the food because he doesn't want to try the same menu every day.

Diane: Full-Time Undergraduate Student at SFSU

Diane is a full-time undergraduate student at SFSU. Not only does she go to school all day but she also works part-time. She orders the food through the app regularly because she doesn't time to cook. The app provides a premium membership for the people who regularly use the app. Sometimes, the premium member gets coupons for the discount or free meal and sometimes they get the fee delivery.

Student who doesn't belong to the SFSU

The app is only for SFSU's students, staff and faculty. If someone who doesn't belong to the SFSU or someone who doesn't have the SFSU g-mail account, they can't create the account for the app to order the food. Moreover, the app also provides the phone number of the restaurant for the people who aren't familiar with using the app. After they order by phone, they can go and pick up the food from the restaurant. The app provides a 10% discount only for the SFSU's staff and faculty.

Vegetarian Customer

Liam is a vegetarian and he is a undergraduate student at SFSU. He loves to use the app because the app provides the option to search which kind of restaurant that the customers are looking for, for example, vegetarian restaurant, Chinese restaurant, Vietnamese restaurant or Italian restaurant. The customer also can see the estimated arrival time through the app and they also can track the delivery person's location.

Admin

Admin who can manage the app can also order the food from the app by using their sfsu.edu account. The admin also needs to review the user's profile and he can also remove from the user list if he finds the violent user in the app.

Delivery Driver

Delivery Driver has to be also SFSU student to work as a driver for the app. So, he can easily order the food by using his sfsu.edu email account. But when he delivers the food, he needs to change the delivery mood in the app. When he changes the delivery mood in the app, he can access the SFSU's map UI within the application for direction. He can also see the order detail of the customer and how long the restaurant takes to get ready to pick up by him.

Restaurant Owner

The restaurant owner should be or shouldn't be a SFSU student. If the restaurant owner is not an SFSU student, he can easily login as a restaurant owner and order the food from another restaurant. Restaurant owner also has authority to manage the app. He can update the menu or advertisement.

III. List of main data items and entities - data glossary/description

a. List of all entities
    i. takeout_order
        1. An order which includes several data fields which pertain to the kitchen order such as a listitem_id, item_price
    ii. receipt
        1. An itemized receipt of all items to be charged - visible to all actors
    iii. review
        1. written for a restaurant by a customer and viewable by employees and restaurant staff

b. List of all actors
    i. sfsu_customer
        1. a user of the application
    ii. person
        1. Shall be a parent entity to all users, which will hold information about a person non-specific to any role such as age, address, and other related data fields.
    iii. delivery_employee

1. Holds delivery employee information.

c. Data structures
   i. restaurant
      1. A list of all restaurants.
   ii. user_reg_record
      1. A list of all registered users, past and present.
   iii. delivery_driver_record
      1. A list of all drivers, past and present.
   iv. orders
      1. Handled first in, first out.

IV. Initial list of functional requirements:

1. Customers shall be able to register for the web application service.

   · Valid users shall be verified by checking if the email suffix contains '@sfsu.edu' to ensure exclusive use by SFSU students, staff, and faculty.

   · Customers shall be registered as a student/staff/faculty, restaurant owner, driver, or Admin.

   · Registration shall include name, SFSU email, and delivery address to account for estimation of delivery times and restaurant availability via building or apartment number.

2. Customers shall be able to login and logout to and from the web application service using verified email username credentials.

3. Customers shall be able to deregister their account.

4. Customers shall be able to search for restaurants based on different parameters.

   · Location, cuisine type, menu items, etc.

5. Customers shall be able to leave a rating or review of searchable restaurants.

   · Ratings may include a star rating from 0-5 with an optional comment review section included with the rating or photo attachment.

6. Customers shall be able to place a food order at a restaurant with a delivery or pickup option.

   · Once the order is placed, the order is automatically paid for (not simulated).

   · Unique features shall include the ability of the customer to leave specific delivery instructions.

7. Customers shall be able to view and make changes to their order via a cart or list.

8. Customers shall receive notifications about the status of a placed order.

   · Notifications shall be received through the web application.

9. Customers shall be able to track the status of their placed order via notifications.

10. Customers shall be able to cancel a placed order.

11. A web application-based notification system shall be implemented to send notifications about status of delivery to customers.

   - Notifications shall be sent when: an order is placed, a command has been entered to check the order status, a command has been entered to cancel the order, a restaurant is in the process of preparing the order, a restaurant has completed preparing the order, a delivery driver has been assigned to deliver the order, the delivery driver is on their way to pick up the order, the delivery driver has picked up the order, the delivery driver is enroute to the delivery address, the order has been successfully delivered, the restaurant is closed or no longer taking orders.

12. Customers shall be able to update their account and contact information.

13. Restaurants shall be able to register for the web application service.

14. Restaurants shall be able to login and logout of the web application using email credentials.

15. Restaurants shall be able to create their profile to be published by the Admin.

   · Create and update their name, description, profile image, and advertisement.

16. Restaurants shall be able to add or remove menu items.

17. Restaurants shall be able to add or remove or update menu item descriptions.

18. Restaurants shall be able to add or remove pictures from the menu or advertisement.

   · Unique feature shall be attaching photos to menu items or a photo carousel of popular menu items.

19. Restaurants shall be able to manage incoming placed orders.

   · This includes marking the completion of preparation of an order, order cancellation, and assigning a delivery driver to the order.

20. Restaurants shall be able to stop taking online orders.

   · This feature is available if somehow the restaurant is closed during the specific hours or is no longer in service.

21. Restaurants shall be able to de-list their profile if they do not wish to continue service.

22. The Admin shall be required to review and approve Restaurant profiles to be published that meet the terms of service.

23. The Admin shall be required to delete inappropriate items from Restaurant profiles that violate the terms of service.

24. The Admin shall be able to delete users that have violated the terms of service.

25. The Delivery Driver shall be able to access a SFSU campus map UI within the application for delivery directions.

26. The Delivery Driver shall be able to download and view order details to facilitate Restaurant food delivery to Customers.

V. List 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. Application delivery shall be from chosen cloud server.

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 must render well on mobile devices (specifics to be developed in consultation with users e.g. Petkovic).

4. Ordering and delivery of food shall be allowed only for SFSU students, staff and faculty.

5. Data shall be stored in the database on the team's deployment cloud server.

6. No more than 50 concurrent users shall be accessing the application at any time.

7. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.

8. The language used shall be English (no localization needed).

9. Application shall be very easy to use and intuitive.

10. Application should follow established architecture patterns.

11. Application code and its repository shall be easy to inspect and maintain.

12. Google analytics shall be used.

13. No email clients shall be allowed.

14. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.

15. Site security: basic best practices shall be applied (as covered in the class) for main data items.

16. Application shall be media rich (images, maps etc.). Media formats shall be standard as used in the market today.

17. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development.

18. The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2021 For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).

VI. Competitive analysis:

- There will be student discounts that other competitors don't usually provide.
- Advance search to pick between vegetarian options or Nonveg which makes it faster to order than our other competitors.
- Delivery will be only available for SFSU students, and staff.
- Delivery options will be provided based on their time and location.

Feature	**Competitor A (DoorDash)**	**Competitor B (GrubHub)**	**Our Future product**
SFSU Campus and housing	+	+	++
Food place search	+	+	+
Shopping cart	+	+	+
Delivery Options	+	+	++
Food Menu	+	+	+
Food option(eg: Veg, or nonveg)	+	-	++

+ feature exists; ++ superior; - does not exist

- There are a lot of restaurants which provide delivery service but we will only be focusing on delivering to SFSU students and staff/Facility . To order food, you need to have SFSU addresses and you need to have an SFSU account to order which will make it easier and faster for students and staff because the food will only be delivered to SFSU campus and housing. You can search nearby places and pick as usual like other restaurants. The shopping cart will be the same feature as others. There will be plenty of options for delivery for SFSU students and staff based on their schedule. They will have the priority to search between different kinds of food options like veg and nonveg which will make it easier and faster to checkout. Most of these options are not provided by other companions for SFSU students and staff.

VII. High-level system architecture and technologies used:

- **Server Host:** Amazon AWS Linux Ubuntu CPU 1 GB RAM
- **Operating System:** macOS Big Sur 11.2.1
- **Database:** MySQL Workbench80
- **Web Server:** NGINX 1.19.6

- **Server-Side Language:** Python
- **Additional Technologies:**
    - **Web Framework:** Flask
    - **IDE:** PyCharm

VIII. Team and roles:

- **Team Lead:** Branden Nguyen
- **Front End Lead:** Jasmine Jahan
- **Front End:** Gurjot Singh
- **Back End Lead:** Yongjian Pan
- **Back End:** Tin Thu Zar Aye
- **GitHub Master:** Robert Freeman

IX. Checklist:

a. So far all team members are engaged and attending ZOOM sessions when required

1. **OK**

b. Team found a time slot to meet outside of the class Back end, Front end leads and Github master chosen

1. **DONE**

c. 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

1. **DONE**

d. Team lead ensured that all team members read the final M1 and agree/understand it before submission

1. **ON TRACK**

e. Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)

1. **DONE**

B. Milestone II Document

SW Engineering CSC648/848 Spring 2021

HungryGators-19

Food Delivery and Restaurant Review Service

Available on Mobile and Desktop

Team 01

Branden J. Nguyen – Team Lead – bnguye11@mail.sfsu.edu
Jasmine Jahan – Front End Lead
Yongjian Pan – Back End Lead
Robert Freeman – GitHub Master
Gurjot Singh – Team Member Front End
Tin Thu Zar Aye – Team Member Back End

Milestone 2

March 19, 2021

Date Submitted	Date Revised
March 19, 2021	March 26, 2021

Table of Contents

I. Executive Summary

San Francisco State University is a place of cultural diversity. There are many people including students, faculty, and staff from different parts of the world whose food habits are also different from each other. Though there are a variety of cafeterias/canteens in SFSU, they are not available for online delivery in the campus area. Most of the time students have to go to the cafeteria to get the food. This can be a hassle especially if the students are having food in between classes, in which case, they may not have time to walk to the cafeteria, order the food, wait and be late for class. As well as the students have to select the food randomly without reading any review or rating which could be misleading. There are some other applications to get the food delivered from larger restaurants but available only outside the SFSU campus area. The process is clunky and hard to navigate.

We, Team 1, are building an application by which the students, faculty, and staff could see the review and order food within the campus, delivered straight to their classroom door or dorm. With this food delivery service students, faculty and staff will be able to order food by creating an account with their sfsu.edu email and name. If the person does not belong to the university they cannot order from the app. For the person who isn't familiar with using the app, they can get the phone number from the app and can pick it up. The minimum viable product of our internet application can be described as a restaurant review and delivery service solely catered to San Francisco State University students and staff. This app will not only solve the problem of food delivery for students of SFSU but also save time for the students to study.

The application would capitalize on the gold rush that is the student-delivery industry. This cornerstone of good execution of this business shall be the options to choose from, and the form delivery of the food. We would exceed our competitors by not only serving exclusively SFSU students but also monopolize students at the school, which are otherwise untapped resources. This emphasis on delivery addresses ease-of-use as well as delivery options and more restaurants, this service will have speedy delivery and a ton of options. HungryGators-19 is an application that will revolutionize food delivery services for students at San Francisco State University.

II. List of main data items and entities

KEY: PK=Primary Key, FK=Foreign Key; the non-filled circle is a table, a black square is a

- customer_orders
    - idcustomer_orders
        - PK
    - customer
        - FK: idsfsu_customer from sfsu_customer table
    - receipt
        - FK: idcustomer_receipts from customer_receipt table
- customer_receipts
    - idcustomer_receipts
        - PK
    - customer
        - FK: idsfsu_customer from sfsu_customer table
    - receipt
        - FK: idcustomer_receipts from customer_receipt table
- delivery_driver_record
    - iddelivery_driver_record
        - PK
    - name
    - vehicle
        - VARCHAR(45)
- delivery_employee

    - iddelivery_employee
        - PK
    - target_address
    - delivery_time
    - pickup_address
- person
    - idperson
        - PK
    - address
    - age
    - sfsu_customer_id
        - FK from sfsu_customer table
    - delivery_employee
        - FK: iddelivery_employee from delivery_employee table

- receipt
    - idreceipt
        - PK
    - item_price
    - item_name
- records
    - idrecords
        - PK

        - restaurant
            - FK from idrestaurant from restaurant table
        - delivery_driver_record
            - FK iddelivery_driver_record from delivery_driver_record table
        - user_reg_record
            - FK iduser_reg_record from user_reg_record table
- restaurant
    - idrestaurant
        - PK
    - address
    - phone_number
    - zip_code

- restaurant_person
    - idrestaurant_person
        - PK
    - idrestaurant
        - FK idrestaurant from restaurant table
    - idperson
        - FK idperson from person table
- review
    - idreview

        - PK
    - review
    - sfsu_customer
        - FK idsfsu_cutomer from sfsu_customer table
- sfsu_customer
    - idsfsu_customer
        - PK
    - name
    - address
    - zip_code

- takeout_order
    - idtakeout_order
        - PK
    - item
    - price
- user_reg_record
    - iduser_reg_record
        - PK
    - username
    - email

III. Prioritized Functional Requirements

Priority 1

Unregistered Users

1. Shall be able to search for restaurants based on different parameters.

    1.1 Location, restaurant name, menu items, etc.

2. Shall be able to view, edit, or cancel their order via a cart or list.

3. Shall be able to register and login.

    3.1 SFSU email shall be verified by checking the email suffix string '@sfsu.edu'.

    3.2 Shall be registered as 'student/staff/faculty' profile.

    3.3 Registration shall include name and SFSU email.

SFSU Customers

4. Shall be able to perform all functions of Unregistered Users.

5. Shall be able to logout of their account.

6. Shall be able to place a food order at a restaurant with a delivery or pickup option.

    6.1 Once the order is placed, the order is automatically paid for (not simulated).

    6.2 Unique features shall include the ability to leave specific delivery instructions.

7. Shall be able to view, edit, or cancel their order via a cart or list.

Restaurant Owners

8. Shall be able to create an account using their email.

    8.1 Shall be registered as ‘restaurant owner’ profile.

9. Shall be able to login and logout of their account.

10. Shall be able to manage incoming placed orders.

    10.1 This includes marking the completion of preparation of an order, order cancellation, and assigning a delivery driver to the order.

Delivery Drivers

11. Shall be able to create an account using their email.

    11.1 Shall be registered as ‘delivery driver’ profile.

12. Shall be able to login and logout of their account.

13. Shall be able to download and view order details to facilitate restaurant food delivery to SFSU Customers.

14. Shall be able to access a SFSU campus map UI for delivery directions.

Priority 2

Unregistered Users

15. Shall be able to browse existing restaurant reviews.

SFSU Customers

16. Shall be notified of the status of a placed order.

17. Shall be able to leave a rating or review of searchable restaurants.

    17.1 Ratings shall include a star rating from 0-5 with an optional comment review section or photo attachment.

18. Shall be able to browse existing restaurant reviews.

Restaurant Owners

19. Shall be able to add or remove pictures from their menu or advertisement.

    19.1 Unique feature shall include the attachment of photos to menu items or a photo carousel of popular menu items.

20. Shall be able to add or remove menu items.

21. Shall be able to add, remove, or update menu item descriptions.

Priority 3

SFSU Customers

22. Shall be notified in real time about the exact status of their order through a notification system.

    22.1 Notifications shall be sent when: an order is placed, a command has been entered to check the order status, a command has been entered to cancel the order, a restaurant is in the process of preparing the order, a restaurant has completed preparing the order, a delivery driver has been assigned to deliver the order, the delivery driver is on their way to pick up the order, the delivery driver has picked up the order, the delivery driver is en route to the delivery address, the order has been successfully delivered, the restaurant is closed or no longer taking orders.

23. Shall be able to deregister their account.

24. Shall be able to update their account and contact information.

Restaurant Owners

25. Shall be able to de-list their profile if they do not wish to continue service.

26. Shall be able to deregister their account.

27. Shall be able to update their account and contact information.

28. Shall be able to stop taking online orders.

28.1 This feature is available if somehow the restaurant is closed during the specific hours or is no longer in service.

Admin

29. Shall be able to login and logout of their account.

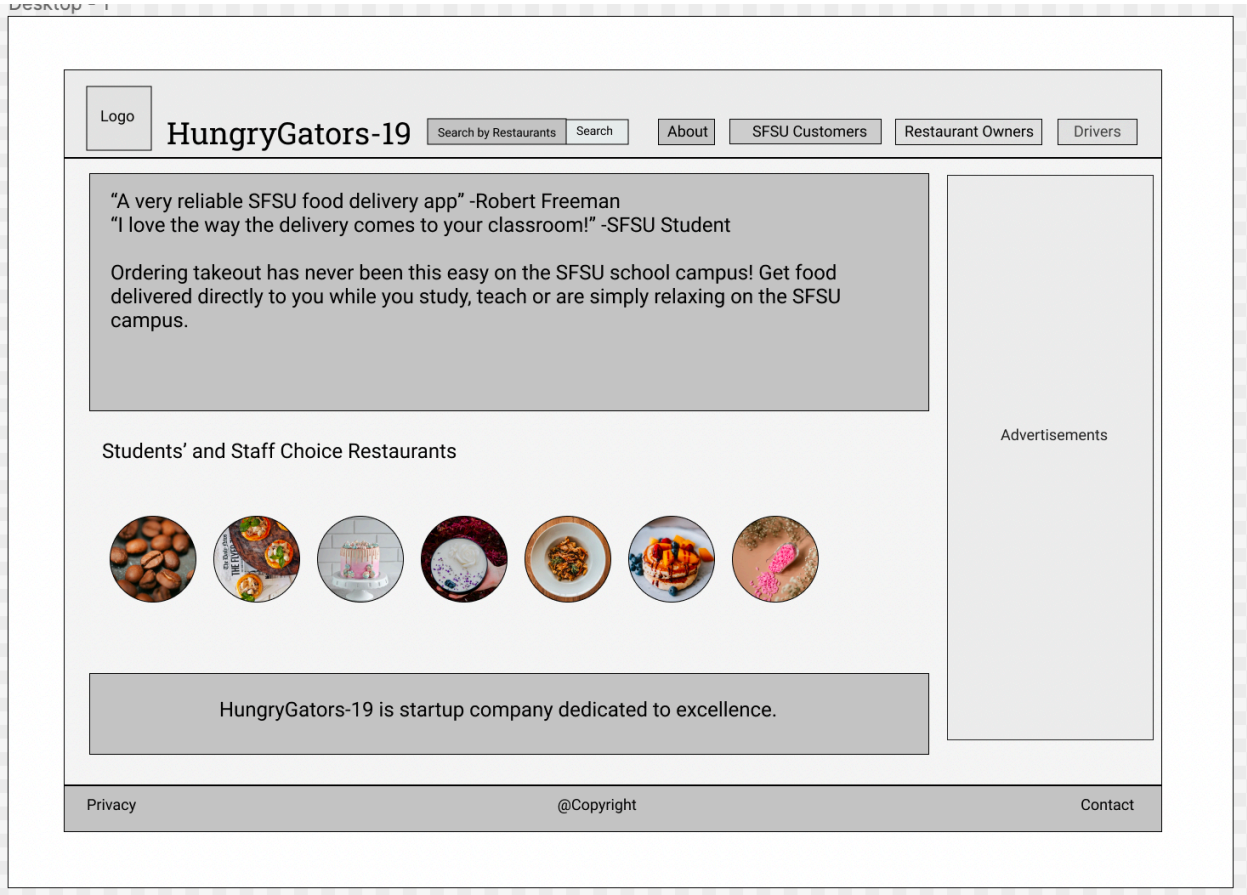
30. Shall be required to review and approve restaurant profiles to be published that meet the terms of service.

31. Shall be required to delete inappropriate items from restaurant profiles that violate the terms of service.

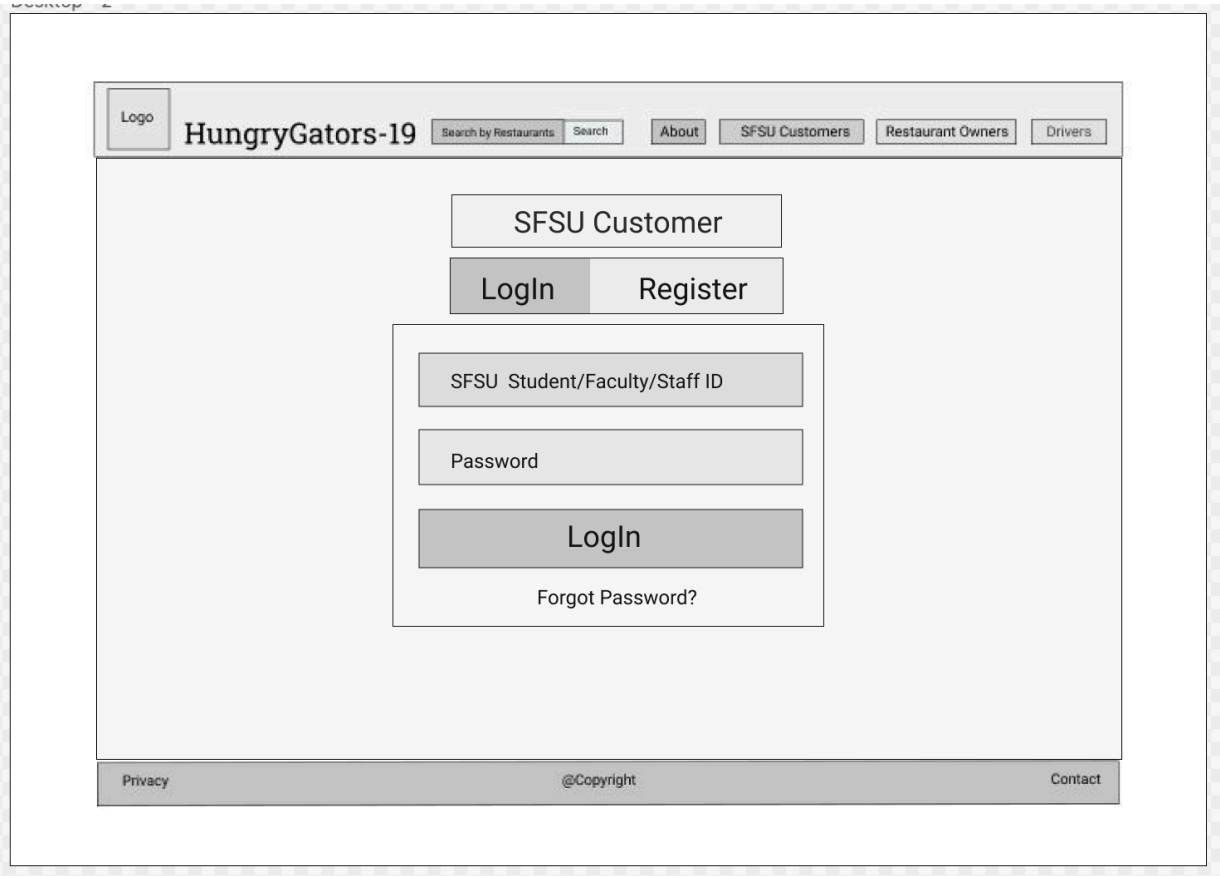
32. Shall be able to delete SFSU Customers, Restaurant Owners, or Delivery Drivers that have violated the terms of service.

IV. UI Mockups and Storyboards

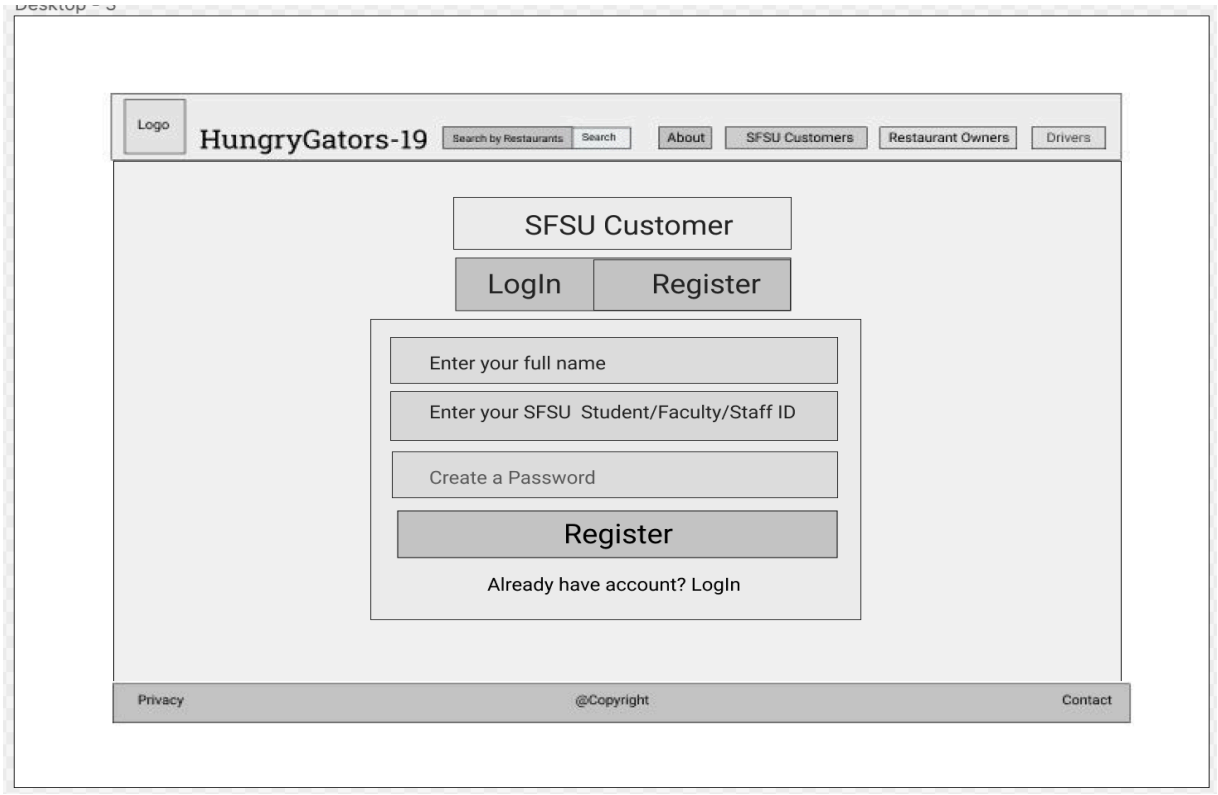
Use Case 1: Diane, a full-time student at SFSU. She orders her meals through the app regularly. Diane opens the web application to order her breakfast. Diane signed in with her SFSU ID which took her to the Checkout page. She really liked that easy Sign in the system, it saves her time always.



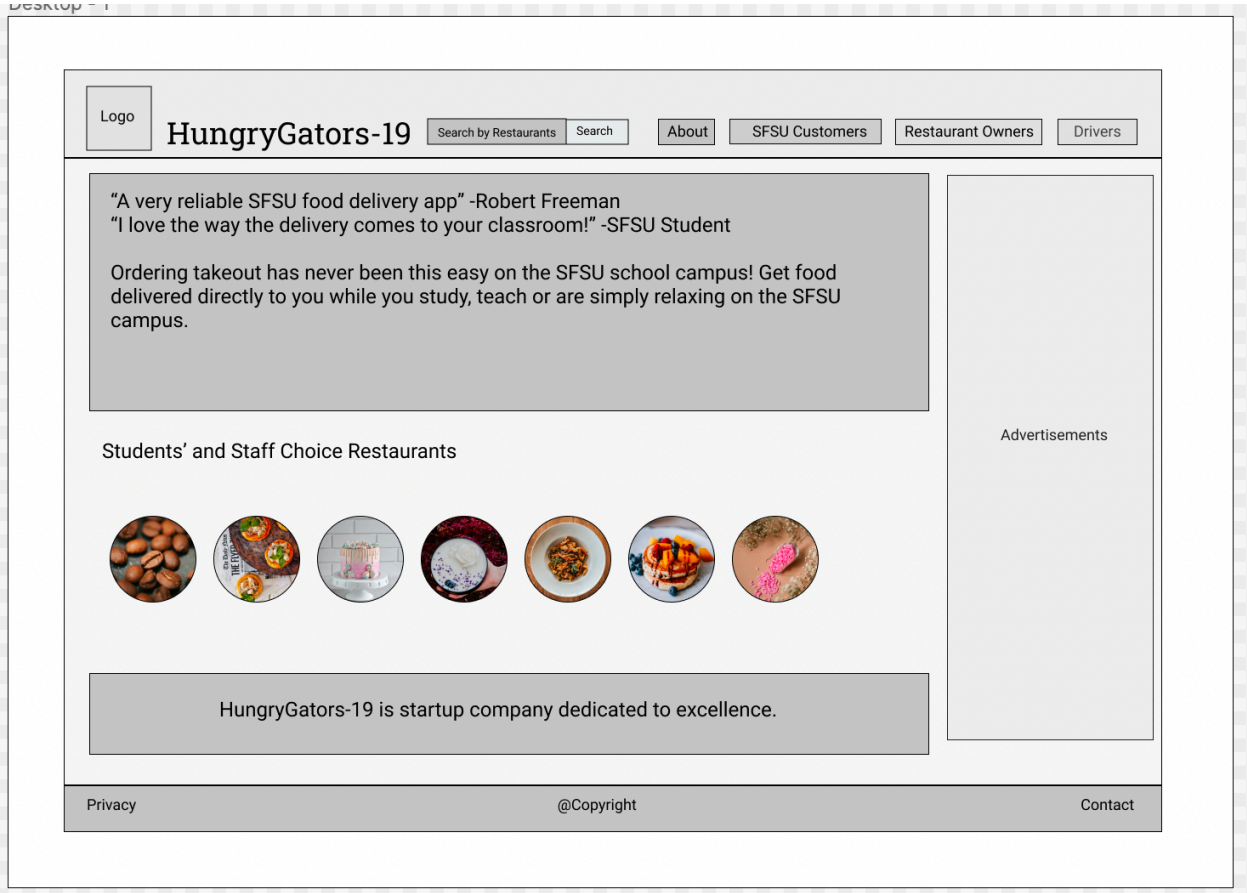
After Diana clicked the SFSU customers from the home page, the app takes to the customer login



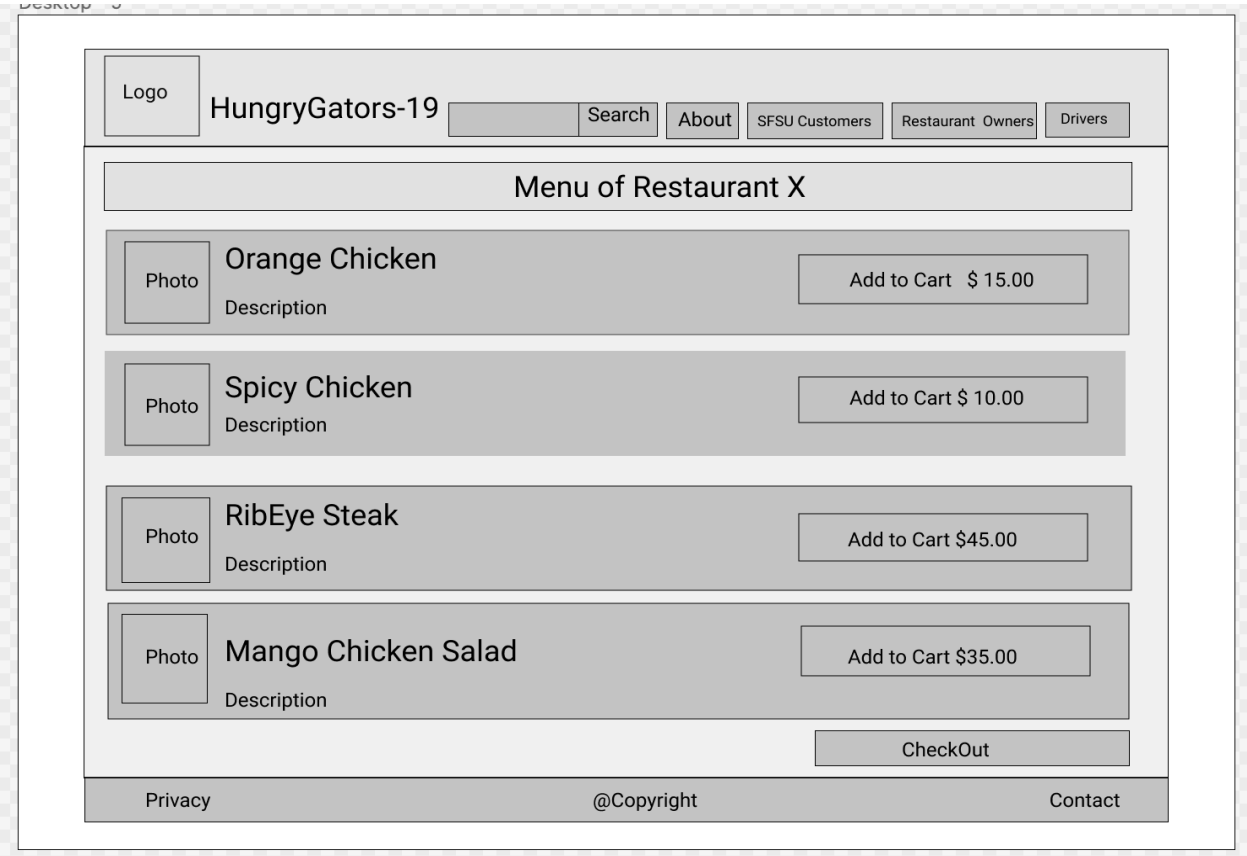
If she doesn't have account or forget password, the app is going to take her to Register page

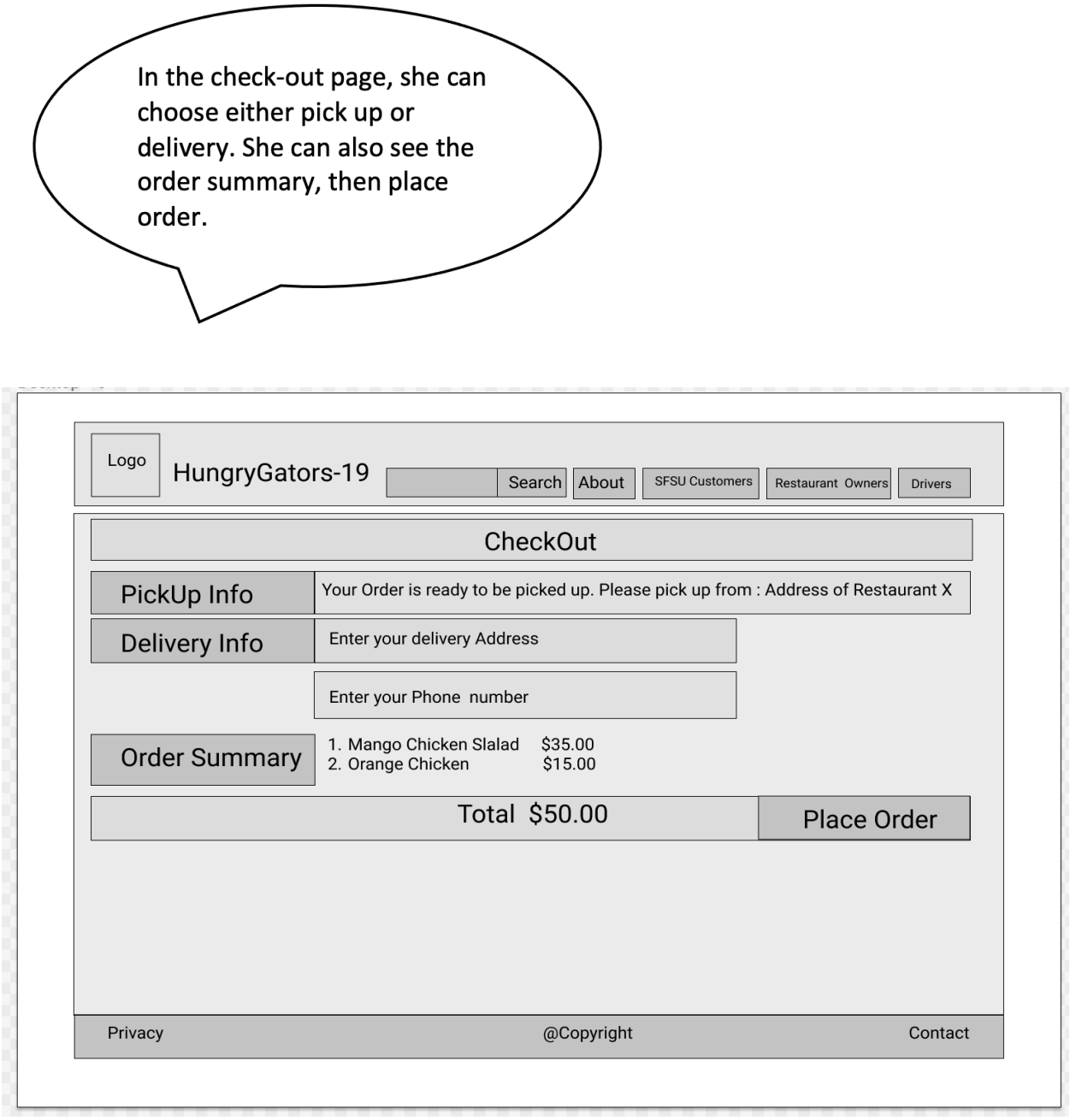


After she login successfully, she can select the restaurant from the app

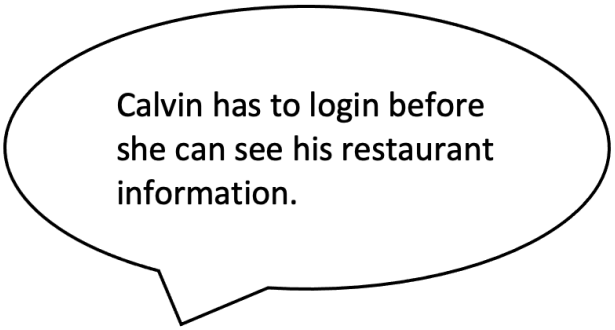


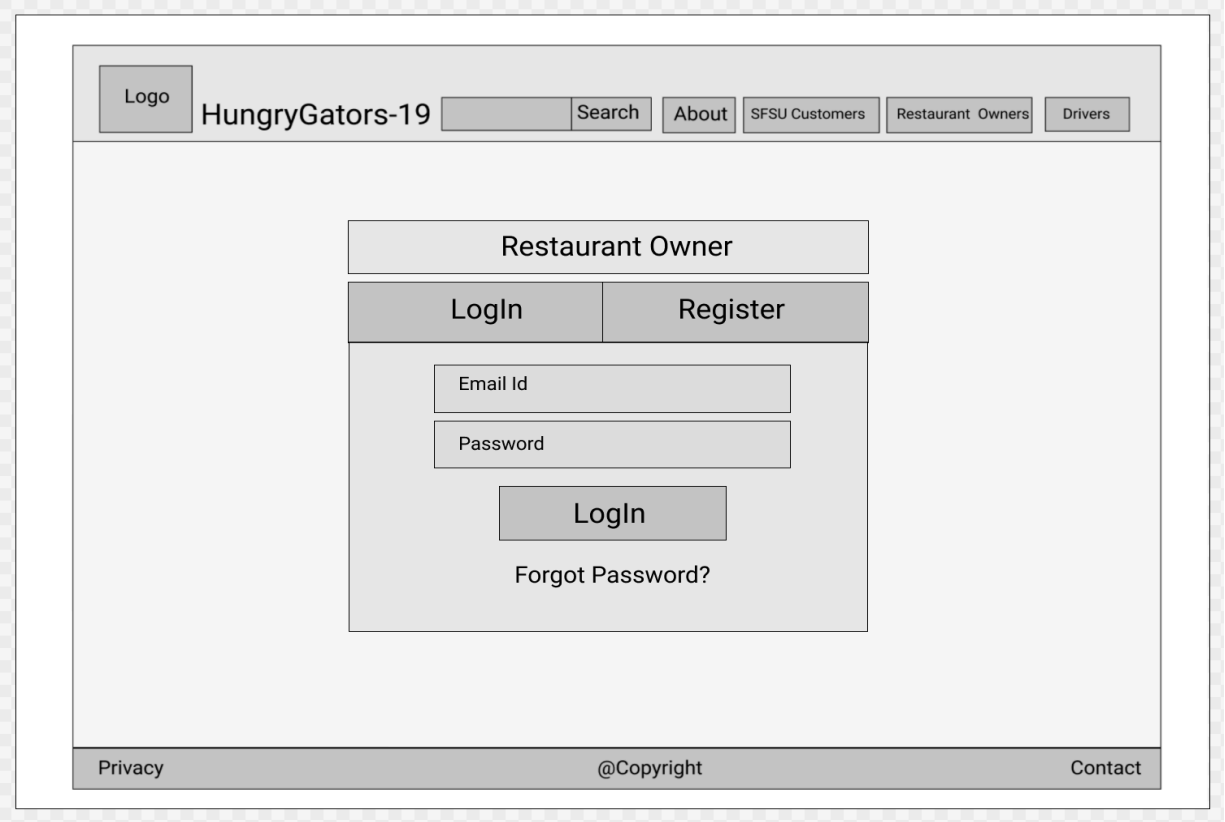
She added 4 items, and check out

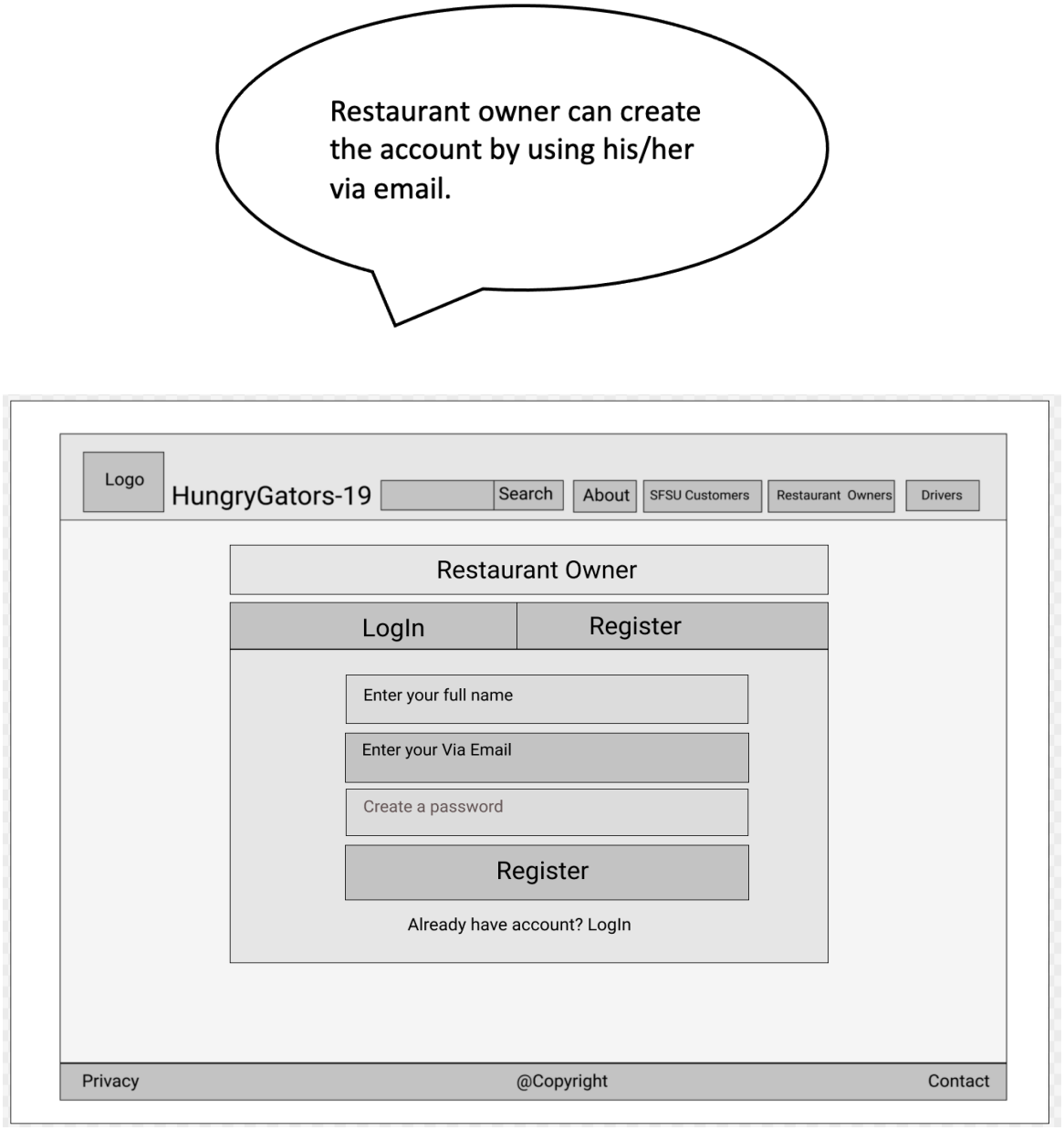




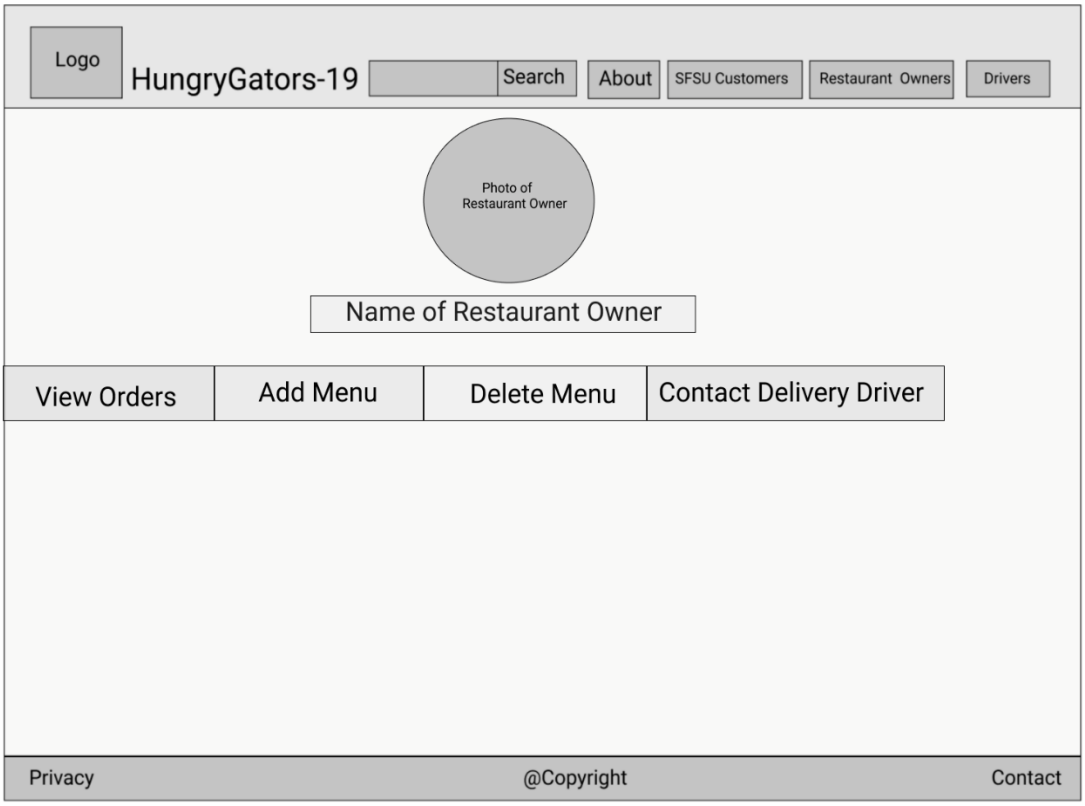
Use Case 2: Calvin is one of the restaurant owners at SFSU neither an SFSU student nor an SFSU staff.


Calvin has to login before she can see his restaurant information.

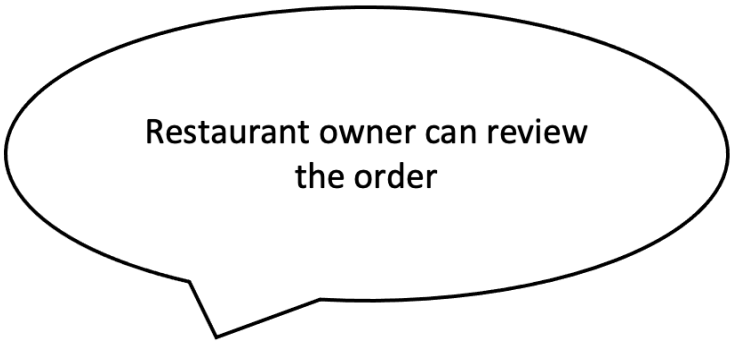

Logo
HungryGators-19
Search
About
SFSU Customers
Restaurant Owners
Drivers
Restaurant Owner
LogIn
Register
Email Id
Password
LogIn
Forgot Password?
Privacy
@Copyright
Contact


Restaurant owner can create the account by using his/her via email.
Logo
HungryGators-19
Search
About
SFSU Customers
Restaurant Owners
Drivers
Restaurant Owner
LogIn
Register
Enter your full name
Enter your Via Email
Create a password
Register
Already have account? LogIn
Privacy
@Copyright
Contact

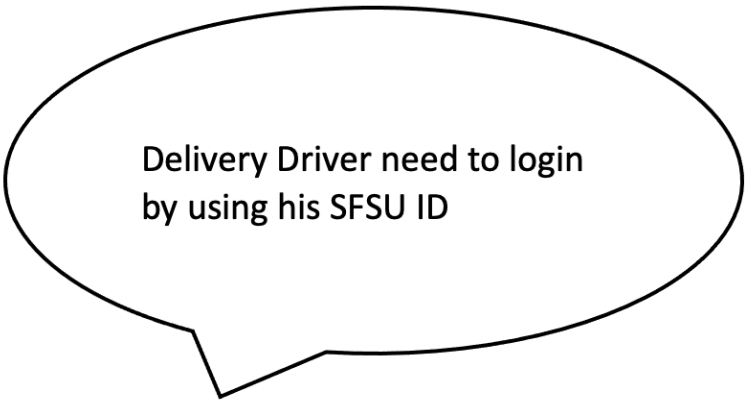
Restaurant owner can review
“view order, add menu,
delete menu and contact
deliver driver in his profile.

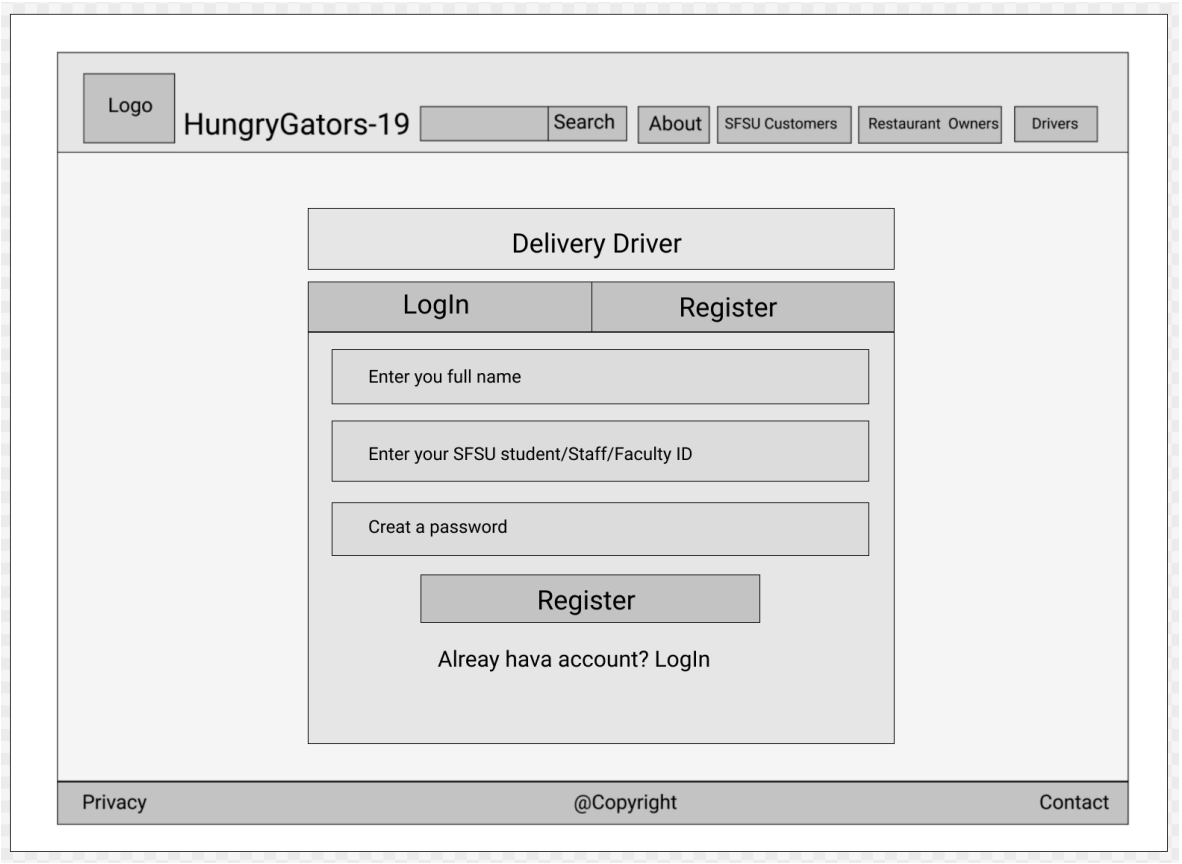


Restaurant owner can add the menu.

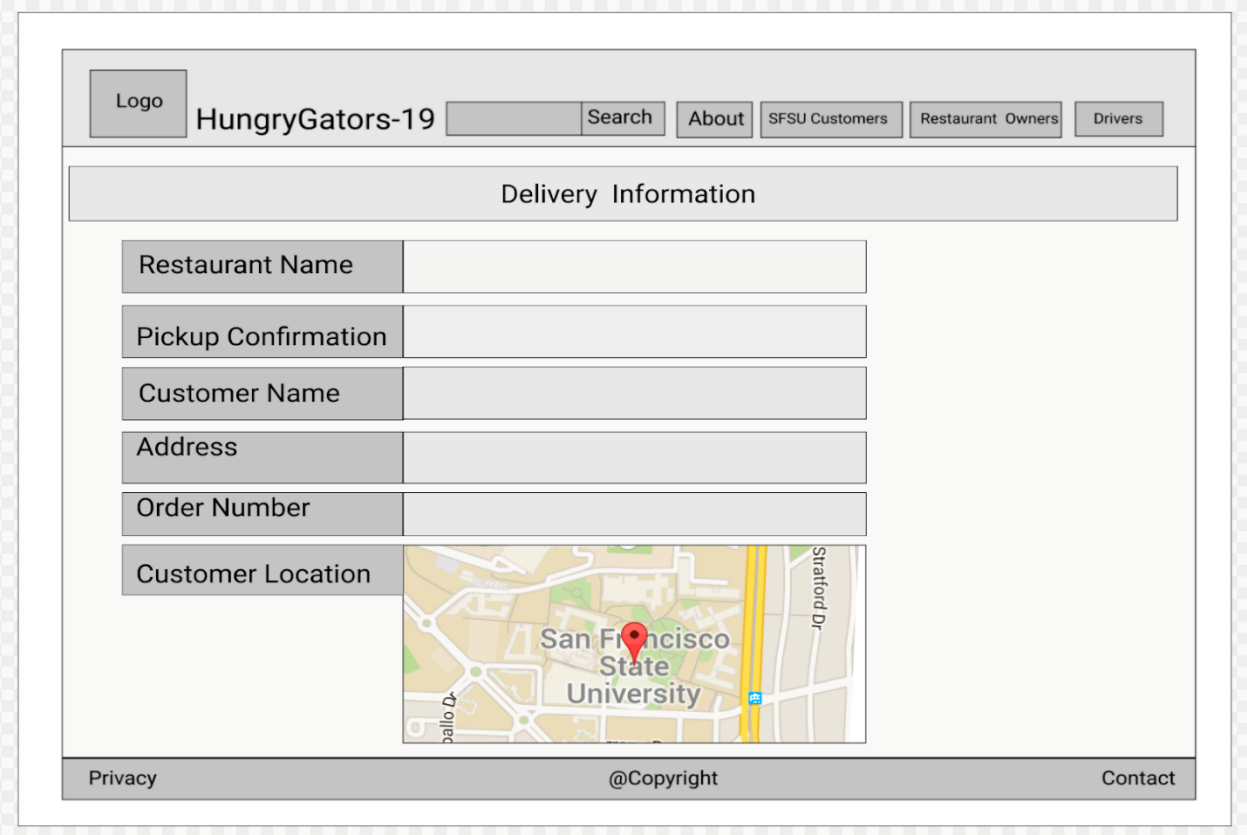


Use Case 3: Jose is a full-time student at SFSU but now he also wants to work as a driver at HungryGator-19. He opened the app and found “Driver” on the home page.


Delivery Driver need to login
by using his SFSU ID


Logo
HungryGators-19
Search
About
SFSU Customers
Restaurant Owners
Drivers
Delivery Driver
LogIn
Register
Enter you full name
Enter your SFSU student/Staff/Faculty ID
Creat a password
Register
Alreay hava account? LogIn
Privacy
@Copyright
Contact

Delivery Driver can review restaurant name, customer name, address, order number and customer location.



V. High-level Architecture, Database Organization Summary

- **Media storage:** Images and video/audio will be kept in the file system. There are no other special data format requirements for video/audio/GPS etc.
- **Search/filter architecture and implementation**: Using the LIKE % method. (Tutorial we will follow found here: https://medium.com/@joseortizcosta/search-utility-with-flask-and-mysql-60bb8ee83dad)
- **Your own APIs**:
    - Search Utility with Flask and MySQL
        - Flask
        - flask-MySQL
    - Bootstrap
- **Any significant non-trivial algorithms**:
    - A significant and non-trivial algorithm will be queuing takeout orders and assigning them to drivers dynamically. This shall be done by traversing a list of takeout_order objects which have not yet been assigned a driver. Then, any available delivery drivers (and of course the restaurant) should be alerted about this new order first in, first out. The delivery driver shall be alerted of the new order and the restaurant shall start making the order. No time estimate will be given.
- No changes have been made to any SW tools or frameworks. None were added. We are continuing as planned.

column (attribute), and the dark circles are notes about the attributes.

- **Data**
  - customer_orders
    - idcustomer_orders
      - PK
    - customer
      - FK: idsfsu_customer from sfsu_customer table
    - receipt
      - FK: idcustomer_receipts from customer_receipt table
- customer_receipts
  - idcustomer_receipts
    - PK
  - customer
    - FK: idsfsu_customer from sfsu_customer table
  - receipt
    - FK: idcustomer_receipts from customer_receipt table
- delivery_driver_record
  - iddelivery_driver_record
    - PK
  - name
  - vehicle
    - VARCHAR(45)
- delivery_employee
  - iddelivery_employee

        - PK
    - target_address
    - delivery_time
    - pickup_address
- person
    - idperson
        - PK
    - address
    - age
    - sfsu_customer_id
        - FK from sfsu_customer table
    - delivery_employee
        - FK: iddelivery_employee from delivery_employee table

- receipt
    - idreceipt
        - PK
    - item_price
    - item_name
- records
    - idrecords
        - PK
    - restaurant

        - FK from idrestaurant from restaurant table
    - delivery_driver_record
        - FK iddelivery_driver_record from delivery_driver_record table
    - user_reg_record
        - FK iduser_reg_record from user_reg_record table
- restaurant
    - idrestaurant
        - PK
    - address
    - phone_number
    - zip_code

- restaurant_person
    - idrestaurant_person
        - PK
    - idrestaurant
        - FK idrestaurant from restaurant table
    - idperson
        - FK idperson from person table
- review
    - idreview
        - PK

        - review
        - sfsu_customer
            - FK idsfsu_cutomer from sfsu_customer table
- sfsu_customer
    - idsfsu_customer
        - PK
    - name
    - address
    - zip_code

- takeout_order
    - idtakeout_order
        - PK
    - item
    - price
- user_reg_record
    - iduser_reg_record
        - PK
    - username
    - email

VI. Key Risks

- **Skill Risks:** Being able to learn and apply the right skills to build the prototype by the due date (This key risk is due to the level of expertise and time needed to build the prototype by the due date and solution may be to self-learn and search for online resources and ask around for help. Team members can study on their own as needed. Keeping the design simple also helps. And finally, backend can use VP to educate other team members)
- **Schedule Risks:** Being able to launch the prototype remotely by the due date (This key risk is due to the amount of time needed to launch the prototype remotely and the solution may be to begin working on individual parts without waiting for teammates' part to complete by keeping the scope of P1 list to a viable minimum)
- **Technical Risks:** Being able to deploy the prototype locally and remotely by due date (This key risk is due to the level of expertise needed in building, running, debugging, and deploying the prototype remotely by the due date and solution may be to ask for help if necessary but doing M2 VP early should help as well)
- **Teamwork Risks:** Being able to designate and coordinate work in a timely manner (This key risk is mainly due to communication and skills needed to coordinate the efforts effectively and solution may be to communicate in a timely manner if problem arises and use PM tools like Trello and follow up on progress)
- **Legal/Content Risks:** Being able to obtain legal content by the due date (This risk is due to obtaining online content but not giving citation needed and solution may be to ensure giving citation when obtaining online contents unless they are stated for free or make addresses fictitious or make our own images as necessary)

VII. Project Management

To Plan M2, we meet on Friday and Sunday to go over the document requirements and assign the work to everyone. We also set the due date for everyone, so we have the document ready before the deadline, and everyone goes over the document the following Friday to review. Once we finish reviewing the document, we fix the mistakes, if there are any. We are using Zoom for meetings and Trello for organizing our work tasks. We are currently focusing on the search menu, restaurants, food menu, Driver sign-in, Admin sign-in, restaurant owner sign-in, customer sign-in, cart, and food review. In the future, we will work on text notification, campus map, working Sign-in/out tab, and adding removing order.

C. Milestone 3 Document

Summary of Milestone 3 ZOOM meeting review with Prof. Petkovic and plans for further development

Team Number: 01
Meeting date: April 21, 2021
Summary of feedback on UI:

1. 2-3 lines of text explaining home page/app purpose/functionality.
2. Nav. bar order: About, Restaurant Owner (force login), Delivery Driver (force login), Login (by default SFSU students/staff/faculty).
    - Nav. bar should have the same .css across all pages.
3. Move search bar slightly to the right.
4. Center page slider should not move in a loop automatically. Include option for user to manually slide through the images.
5. Highlight Hungry Gators-19 logo, include class ID somewhere 'above the top bar'.
6. Add 1 more graphic to Home page to include Restaurant Owners.
    - Order of Graphics: Order food, Become a Restaurant Owner, Become a Delivery Driver
7. Search bar must have a persistent pull-down menu to notify user of what filters are active.
    - Number of search results found should always be displayed.
    - Search field should only accept max of 40 alpha numeric characters, should display an error message telling the user when they enter an invalid query.
    - Search bar should have the same .css across all pages.
8. For each restaurant search results could have additional information: $$ pricing rating of restaurants, Covid-safe/Contactless Delivery
9. Utilize godaddy free certificate security login (https: encrypt password).
10. Images can be omitted for restaurant menus add to cart feature.
    - Make open cart/confirm order UI/buttons obvious.
11. Registration page: use .sfsu email as primary username for login (clarify to user that .sfsu email will be primary login method).
    - Each registration page should have the proper title header: Login as SFSU Student/Faculty/Staff, Login as Restaurant Owner, Login as Delivery Driver.
12. Create password should have an additional field: Enter password again to confirm the field.
13. Login page should have 'Are you new? Register here!' link.

Summary of feedback on code and architecture:

1. Link up front-end pages with the back-end.
2. Keep it simple and clean, focus on P1 priorities only: Feature Freeze State.

Summary of feedback on github usage:

1. Number of commits should not be the main focus, but everyone should be actively working towards the goal and actively contributing to the project.
2. Use concise, yet purposefully descriptive commits so it is easy to identify at first glance.

Summary of feedback on DB:

1. List of tables: User (student/staff/faculty) table, Restaurant table, Order table, Menu table, Cuisine table
    a. Read all data from tables to save time.

Summary of feedback on teamwork:

1. It is the Homestretch! Everyone should be actively collaborating and engaged.

Agreed upon P1 list of features:

1. Homepage
2. Registration
3. Search Bar
4. Order Food
5. Restaurant Menus
6. Cart
7. Driver Pickup, map of campus
8. Upload Restaurant
9. Rob = Admin from workbench ‘flip bit’ *clarify

CEO Checklist:

*** Field validations

- search text input - max 40 alphanumeric characters
- enforce mandatory user action on "agree to terms" (opt in) in reg form (“agree to terms” can be dummy link)
- if you have dedicated registration for SFSU customers ensure their e-mail string contains "sfsu.edu" at the end (no other verification necessary)
- Enforce all mandatory fields in all forms
- (The rest is optional - my goal is to teach you how to do some validation, and save some work (once you do a few you know how to do the rest)_

*** Error messages

If field validation or any form fails provide meaningful error message that YOU design (not the system) to help the user. In error messages two things are key: a) explain what the error is; b) advise user how to correct it"

*** UX

Have the same CSS in ALL pages, and it must contain search widget and class ID text

Search parameters entered by the user must be persistent

Each -age shall have its meaningful title

In UX, search and forms design adhere to key design principles we covered in the class

In restaurant data upload next to SUBMIT button have a text "it may take up to 24 hours for posting to be approved by admin"

D. Milestone 4 Document

SW Engineering CSC648/848 Spring 2021

HungryGators-19

Food Delivery and Restaurant Review Service

Available on Mobile and Desktop

Team 01

Branden J. Nguyen – Team Lead – bnguye11@mail.sfsu.edu
Jasmine Jahan – Front End Lead
Yongjian Pan – Back End Lead
Robert Freeman – GitHub Master
Gurjot Singh – Team Member Front End
Tin Thu Zar Aye – Team Member Back End

Milestone 4

May 14, 2021

Date Submitted	Date Revised
May 14, 2021	May 21, 2021

Table of Contents

I. Product summary

- Hungry Gators-19 is the name of our application.
- Homepage: This is the main page of the website which will guide you to other pages.
- Registration: This feature takes you to the Registration page where you can create an account if you don't have one with us.
- Login (by default SFSU students/staff/faculty): There are three different logins for Driver, Restaurant owner, and student/staff. You have to login to checkout and restaurant owners have to login to add items on to the website.
- Search Bar: This feature helps you to search anything you are looking for on the website.
- Drop down tab: This feature includes different restaurants you can visit.
- Restaurant Menus : This feature takes you to the menus based on the restaurant you pick.
- Cart: This feature helps you to check out all the food items you picked.
- Driver Pickup, map of the campus: This feature is for the drivers where drivers look up for the orders and the map is provided for them so they can drop off the order.
- Upload Restaurant: This feature is for the restaurant owner so they can upload the items they have available for their restaurant.
- **(Link to the webpage)** **http://18.223.160.96:5002/**

II. Usability test plan

- Test objectives:
    - Search bar and dropdown options are being tested which are currently working for our application. All the cuisines are categorized in the dropdown option and you can also access all the data that have been stored in the database by typing into the Search bar.

      For Example:
    - Search: Input: Burger

      Output: There are 3 restaurants who have a burger name field.
    - DropDown: Input: American

      Output: There are 2 American restaurants under American name field.

- Test background and setup:
    - System setup: we are using pycharm locally and then we push it to Amazon AWS to deploy our application.
    - Starting point: Home page
    - Intended Users: SFSU students/staff, drivers, and restaurant owners.

- Usability Task description:
    - Order takeout from your favorite American restaurant nearby.
        - Once you open the HungryGators-19 Homepage go to Search and type in “American food”.
        - Then you will see # Items.

- Then, you select the restaurant and it will display the menu items.
- Then, you update the quantity of the item that you want to order from the menu and click the checkout button which takes you to the cart.
- From there you will see the updated cart page and click on place order.
- Once the order is placed it takes you back to the homepage.

- Evaluation of Effectiveness:

  The user was able to accurately use the search bar in less than 10 seconds and got all the results. The results are displayed and the user is able to navigate throughout the site.

- Evaluation of Efficiency:

  The average time to perform a search query for any food item was 10-30 seconds. The average number of options to click to perform the search is 3 including 'select the cuisine', 'search', and 'see menu from the search restaurants'. A total of 3 page screens are displayed throughout the entire search process.

- Evaluation of user satisfaction:

Please take a moment to answer these questions. Thank you	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
1. The content was just right (not too simple or complete)		•			
2. The Application was easy to use			•		
3. How was your overall experience with our application	•				

III. QA Test Plan

- Test Objectives

The main purpose of the quality assurance test plan is to make sure that the search bar is working. To ensure that the search bar has a persistent pull-down menu to notify users of what filters are active. Moreover, after clicking the search button, the number of search results found should be displayed. Not only can the user search the restaurant by typing the restaurant name in the search bar but the user can also search the restaurant by filtering the cuisines.

- HW and SW setup (including URL)

Server host: Amazon AWS Linux CPU 5 GB RAM

Operating System : macOS Big Sur 11.2.1

Web Browser : Safari, Latest Version of Google Chrome

Database: MySQL Workbench80

Web Server: NGINX 1.19.6

Server-Side Language: Python

Additional Technologies: Web Framework: Flask IDE: PyCharm

- Feature to be tested

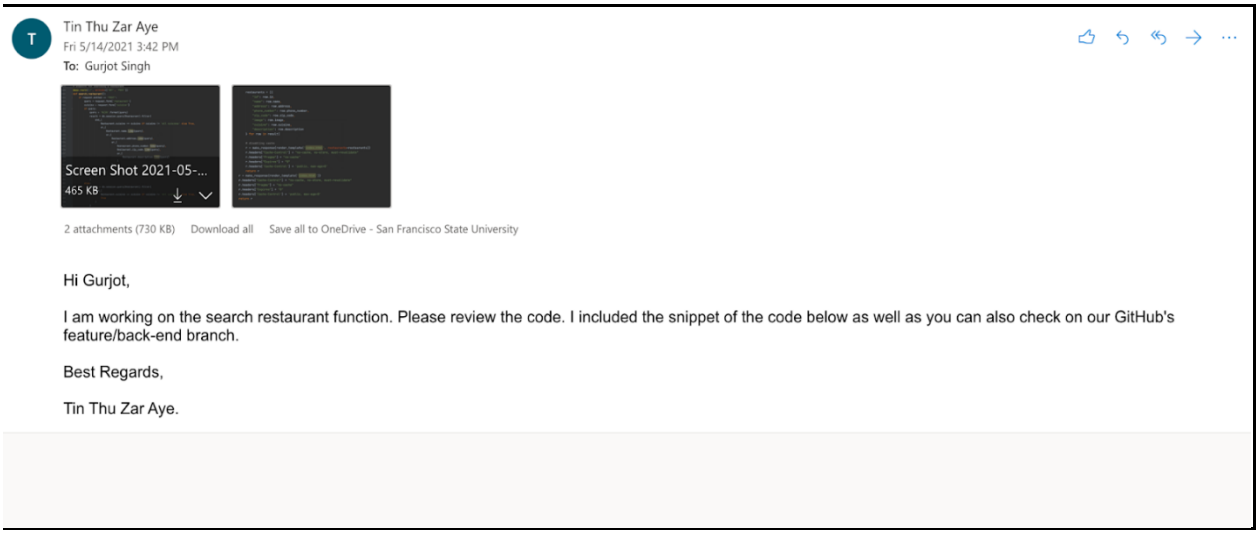
In this quality assurance test plan, the feature to be tested is that the search field should only accept a maximum of 40 alphanumeric characters. If the user enters the over 40 alphanumeric

characters, it should display the error message that is telling the users that they are entering the invalid query.

Test#	Title	Description	Input	Output	Result
1	Search Bar Testing	Click the search button, then 8 restaurants will be displayed	Choose “All Cuisines” and Click the search button	8 restaurants will be displayed for all cuisines which are American, Asian, European, African	PASS (Safari) PASS (Google Chrome)
2	Filtering the Cuisines	Test pull down menu for each cuisine	Users choose the specific cuisine from the “All Cuisines” drown down bar. Then the user needs to click the search button. Users choose “American” and clicked search button	3 American restaurants displayed	PASS (Safari) PASS (Google Chrome)
3	Searching with restaurant’s name	In the search bar, user type the restaurant to find the restaurant	Users type “Cabbage Mix” in the search bar and click the search button	Display the “Cabbage Mix” restaruant.	PASS (Safari) PASS (Google Chrome)

IV. Code Review

One team member sent the email to another team member to review the code.



Tin Thu Zar Aye
Fri 5/14/2021 3:42 PM
To: Gurjot Singh

Screen Shot 2021-05-...
465 KB

2 attachments (730 KB) Download all Save all to OneDrive - San Francisco State University

Hi Gurjot,

I am working on the search restaurant function. Please review the code. I included the snippet of the code below as well as you can also check on our GitHub's feature/back-end branch.

Best Regards,

Tin Thu Zar Aye.

```python
# endpoint for searching a restaurant
@app.route('/', methods=['GET', 'POST'])
def search_restaurant():
    if request.method == "POST":
        query = request.form['restaurant']
        cuisine = request.form['cuisine']
        if query:
            query = '%{}%'.format(query)
            result = db.session.query(Restaurant).filter(
                and_(
                    Restaurant.cuisine == cuisine if cuisine != 'all cuisines' else True,
                    or_(
                        Restaurant.name.like(query),
                        or_(
                            Restaurant.address.like(query),
                            or_(
                                Restaurant.phone_number.like(query),
                                Restaurant.zip_code.like(query),
                                or_(
                                    Restaurant.description.like(query)
                                )
                            )
                        )
                    )
                )
            )
        else:
            result = db.session.query(Restaurant).filter(
                and_(
                    Restaurant.cuisine == cuisine if cuisine != 'all cuisines' else True,
                    True
                )
            )
```

```
212
213         restaurants = [{
214             "id": row.id,
215             "name": row.name,
216             "address": row.address,
217             "phone_number": row.phone_number,
218             "zip_code": row.zip_code,
219             "image": row.image,
220             "cuisine": row.cuisine,
221             "description": row.description
222         } for row in result]
223
224         # disabling cache
225         r = make_response(render_template('index.html', restaurants=restaurants))
226         r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
227         r.headers["Pragma"] = "no-cache"
228         r.headers["Expires"] = "0"
229         r.headers['Cache-Control'] = 'public, max-age=0'
230         return r
231     r = make_response(render_template('index.html'))
232     r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
233     r.headers["Pragma"] = "no-cache"
234     r.headers["Expires"] = "0"
235     r.headers['Cache-Control'] = 'public, max-age=0'
236     return r
237
```

Another Team member reviewed the code and made the comments.

Code Review

2

GS

Gurjot Singh
Fri 5/14/2021 4:03 PM
To: Tin Thu Zar Aye

It looks good but you need to add some comments for the code.

Line 182: add the comment
#request for processing a submitted form

Line 183: add the comment
#retrieve post form data from each field

Line 185: add the comment
#search restaurant by parameters

Line 187: add the comment
#filter search results by each filed

...

From: Tin Thu Zar Aye <taye1@mail.sfsu.edu>
Sent: Friday, May 14, 2021 3:42 PM
To: Gurjot Singh <gsingh13@mail.sfsu.edu>
Subject: Code Review

Hi Gurjot,

I am working on the search restaurant function. Please review the code. I included the snippet of the code below as well as you can also check on our GitHub's feature/back-end branch.

Best Regards,

Tin Thu Zar Aye.

Reply | Forward

V. Self-check on best practices for security

Asset to be protected	Type of possible/expected attacks	Strategy to mitigate/protect the assets
Registration for user and delivery driver's email to include "sfsu.edu"	User can't register if they don't have "sfsu.edu" email address	If the user/delivery driver tries to register with the email which is not ending with sfsu.edu, the email string will be validated and an error will be displayed.
Password confirmation for the registration	User can't register if they enter the different password for the "confirm password" field when they register	Password will be saved as encrypted string
Check box for "Team and Conditions"	User can't register if they don't check in the check box for "Team and Condition".	The check box is implemented by HTML and the user can't register if they forgot to check in the check box.
User Login	User can't login if they don't have "sfsu.edu" email	To order food from the restaurant, user require to login with their "sfsu-edu" email address. if not, the email string will be validated and an error will be displayed
User password	User can't login if they put the wrong password or forgot the password.	User's password will be stored in the database and if they forgot the password to login, they could click the "forgot Password?" to create new password.
User's name	User can't order if they forgot to put their name in the Check-out page.	Users can't place the order and will get an error if they forgot to put their name in the Checkout page.
User's Phone Number	User can't order if they forgot to put their phone number in the Check-out page.	User can't place the order and will get an error if they forgot to put their name in the Checkout page.
User Address	User can't order if they forgot to put their address in the Check-out page.	Users can't place the order and will get an error if they forgot to put their address in the Checkout page.

Restaurant Name	User can't find the restaurant name if they enter the non-existent restaurant name.	All the restaurant names are storing in the MySQL database and the users can't find the restaurant if there is not storing in the database.
Menu items	User can't see the menu item before searching the restaurant.	All the menu items are storing in the MySQL database.
Delivery Driver Login	Delivery Driver can't login if they don't have "sfsu.edu" email	To see the order detail and where to deliver the order, delivery driver need to login with the "sfsu.edu" email address
Search bar input validation	Over 40 alphanumeric characters will display the error message	User needs to input up to 40 valid alphanumeric character which does not include special characters
Home Page's Images	Importing the image in the different directory when implementing the code will not display an image.	Implementing with the correct directory for the images in homepage's HTML will be protected the images from being massed up.

VI. Self-check: Adherence to original Non-functional specs

<u>High-level non-functional specifications (how the app is delivered and other constraints) that MUST be adhered to (not negotiable)</u>

1. Application shall be developed, tested and deployed using tools and servers approved by Class CTO and as agreed in M0. Application delivery shall be from chosen cloud server
ON TRACK

2. Application shall be optimized for standard desktop/laptop browsers e.g. must render correctly on the two latest versions of two major browsers
ON TRACK

3. All or selected application functions must render well on mobile devices (specifics to be developed in consultation with users e.g. Petkovic)
DONE

4. Ordering and delivery of food shall be allowed only for SFSU students, staff and faculty
ON TRACK

5. Data shall be stored in the database on the team's deployment cloud server.
ON TRACK

6. No more than 50 concurrent users shall be accessing the application at any time
DONE

7. Privacy of users shall be protected and all privacy policies will be appropriately communicated to the users.
ON TRACK

8. The language used shall be English (no localization needed)
DONE

9. Application shall be very easy to use and intuitive
ON TRACK

10. Application should follow established architecture patterns
ON TRACK

11. Application code and its repository shall be easy to inspect and maintain
ON TRACK

12. Google analytics shall be used
ON TRACK

13. No e-mail clients shall be allowed.
DONE

14. Pay functionality, if any (e.g. paying for goods and services) shall not be implemented nor simulated in UI.
DONE

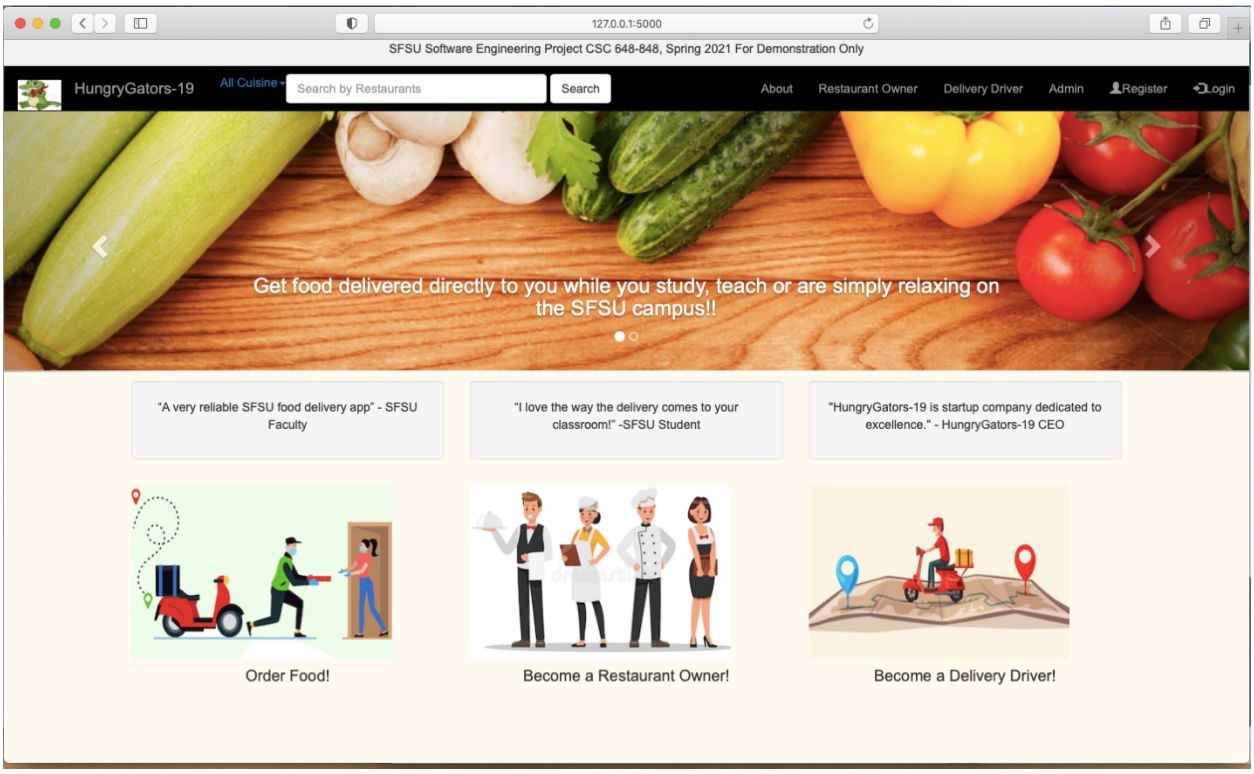
15. Site security: basic best practices shall be applied (as covered in the class) for main data items
ON TRACK

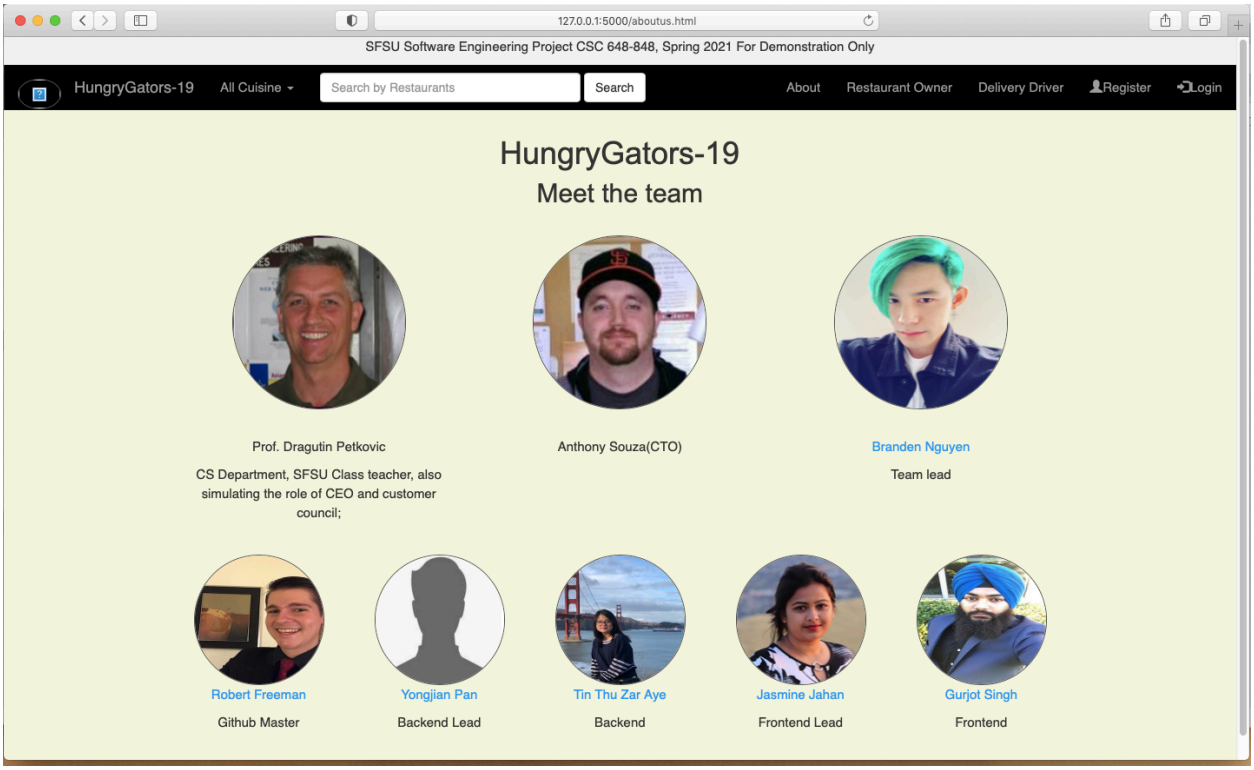
16. Application shall be media rich (images, maps etc.). Media formats shall be standard as used in the market today
ON TRACK

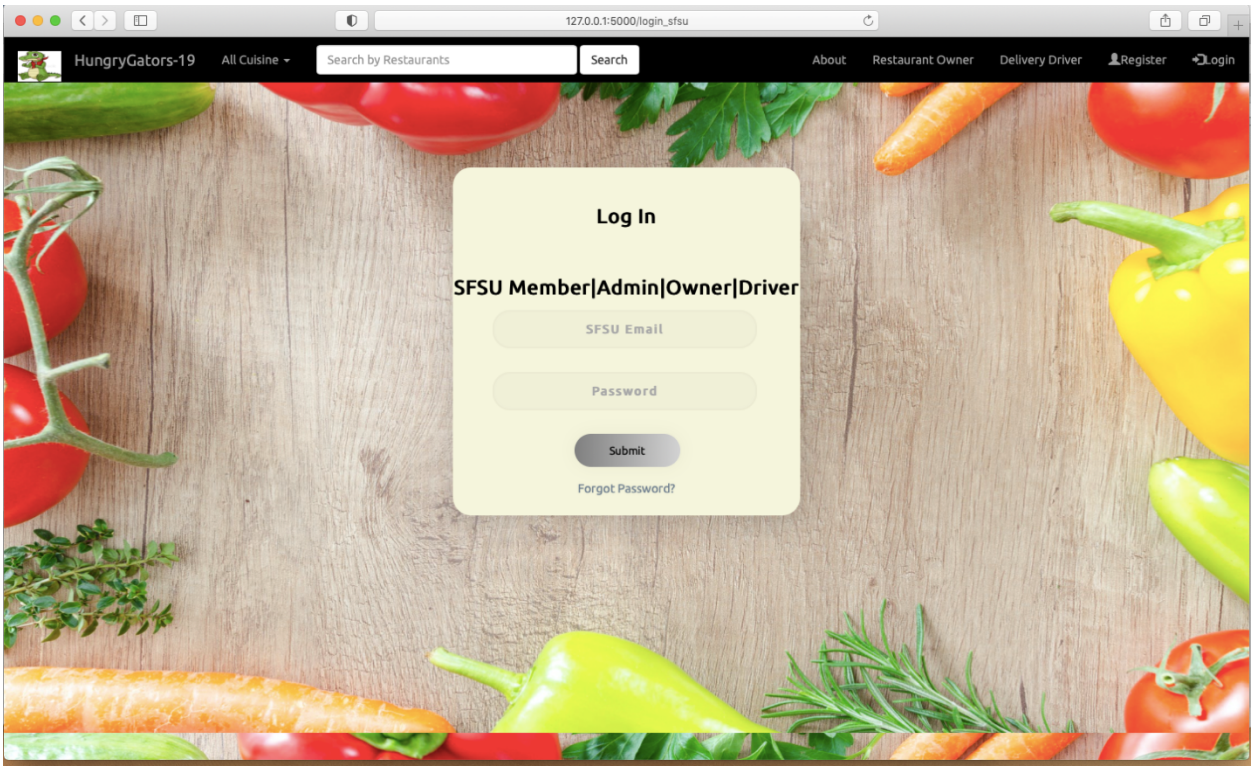
17. Modern SE processes and practices shall be used as specified in the class, including collaborative and continuous SW development
ON TRACK

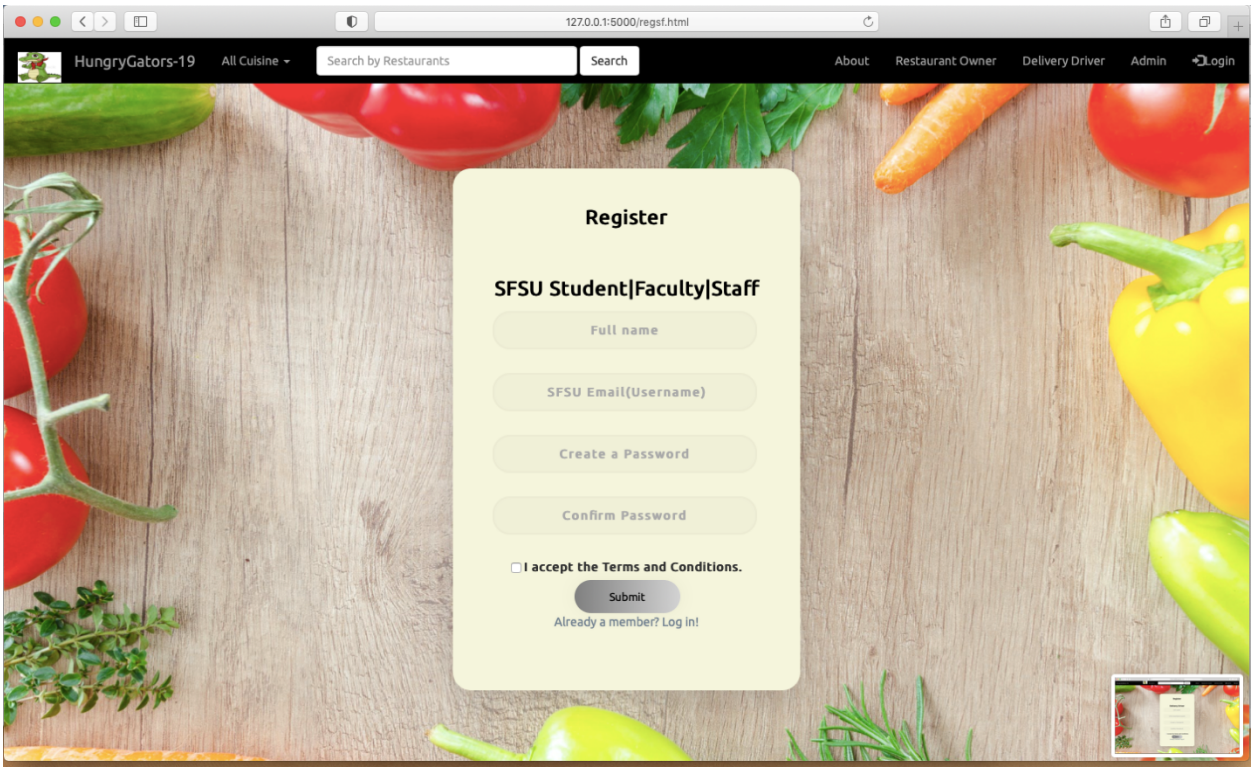
18. The application UI (WWW and mobile) shall prominently display the following exact text on all pages "SFSU Software Engineering Project CSC 648-848, Spring 2021 For Demonstration Only" at the top of the WWW page. (Important so as to not confuse this with a real application).
ON TRACK

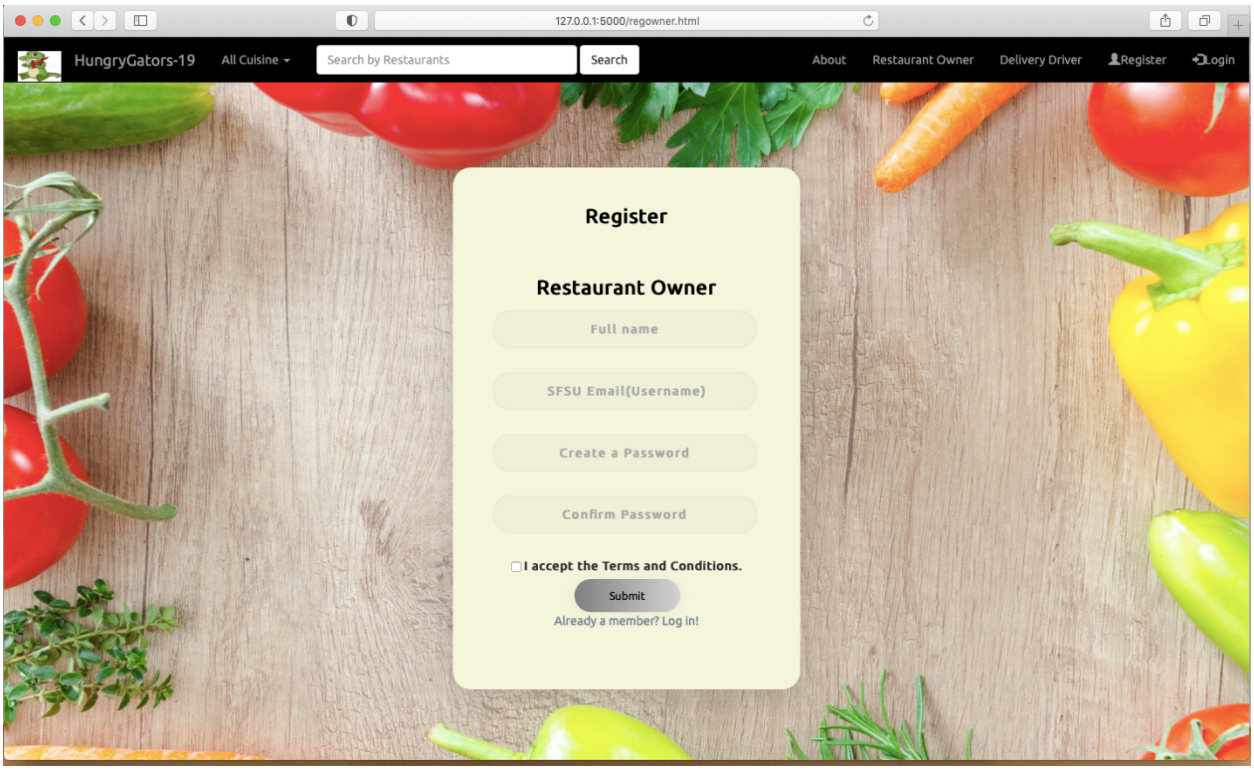
III. Product Screenshots: Screenshots of the actual final product as shown in the demo, one per page

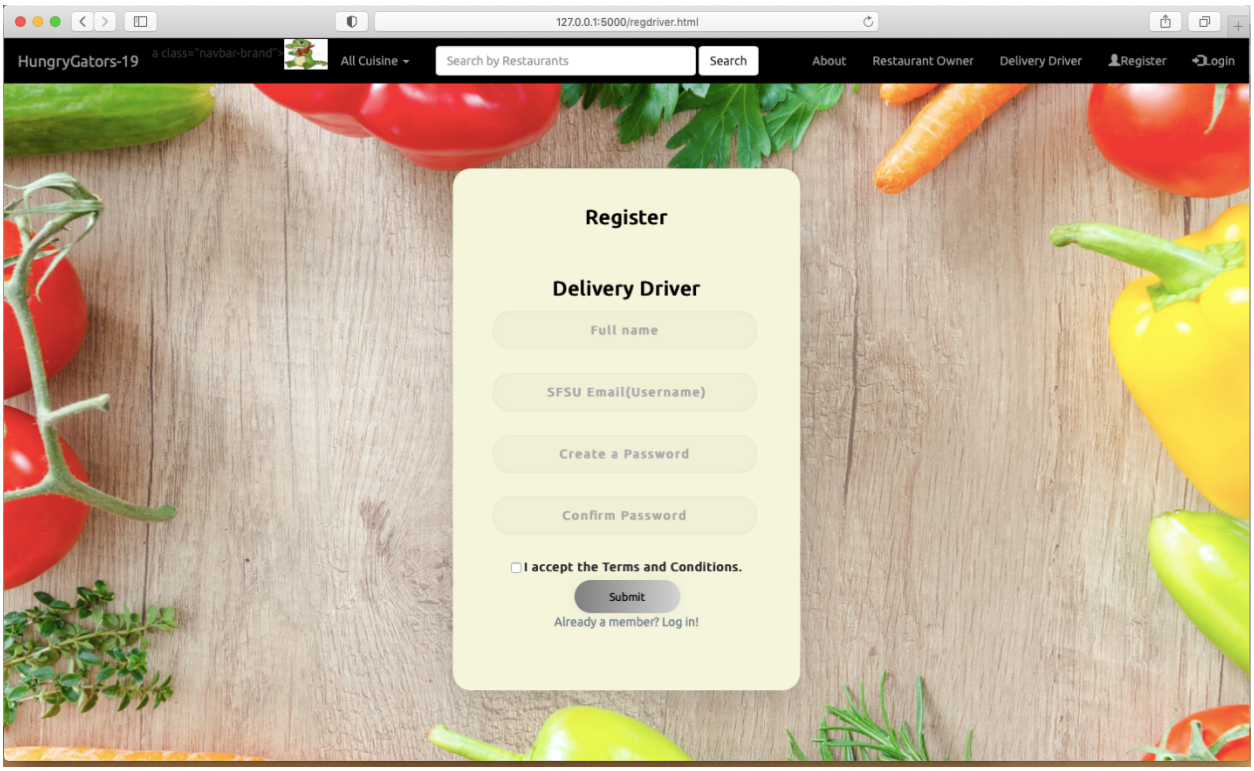


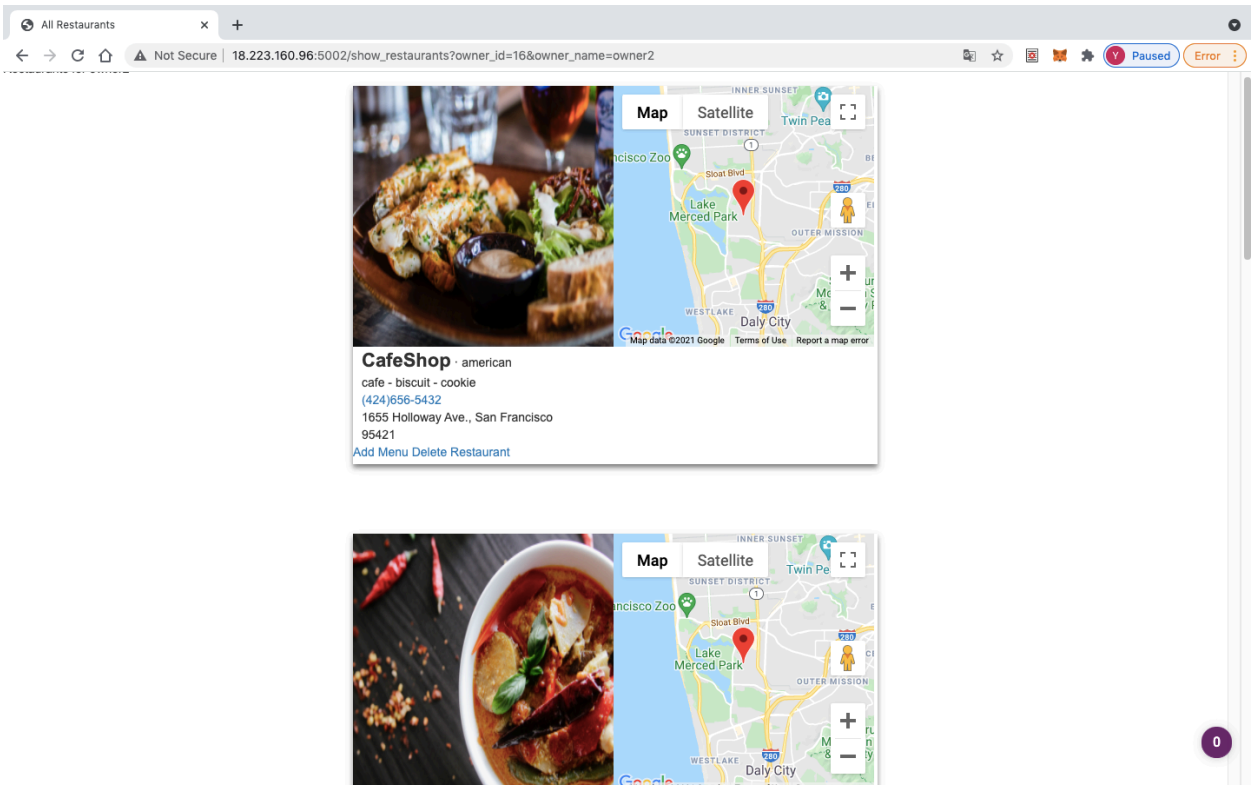


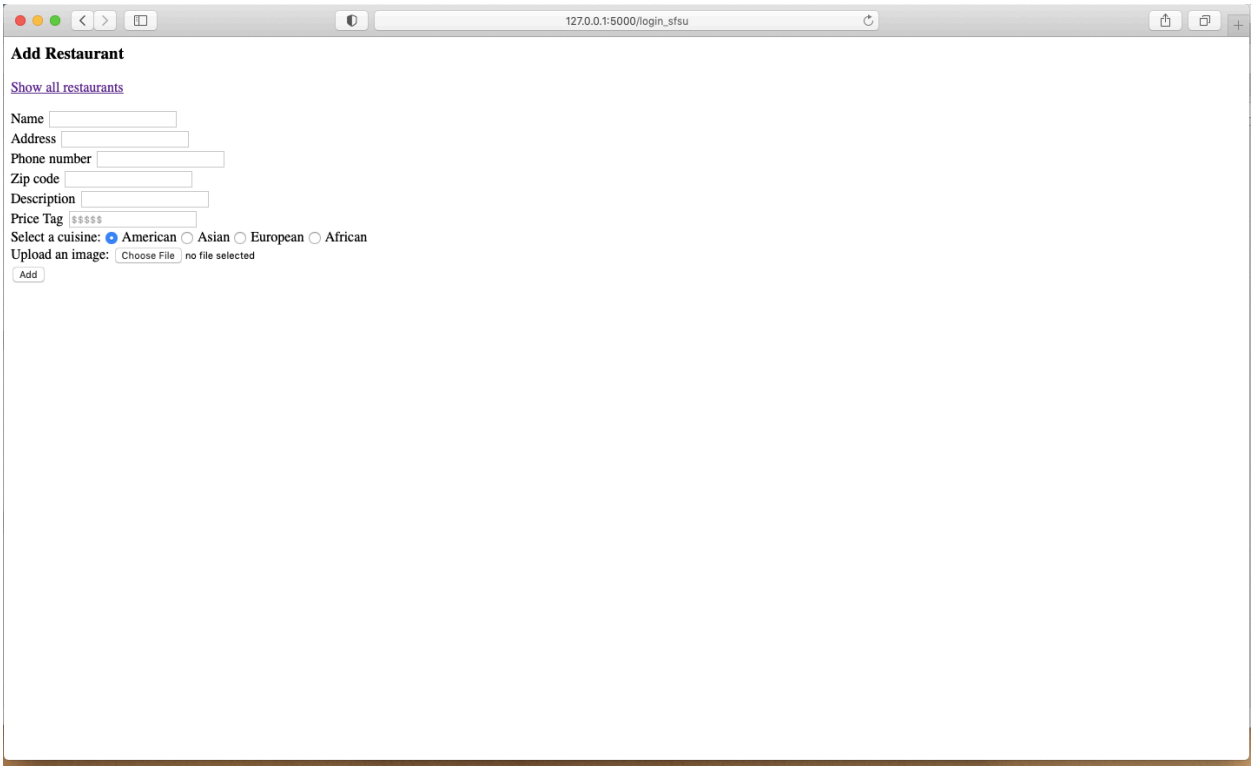

127.0.0.1:5000/login_sfsu
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
Log In
SFSU Member|Admin|Owner|Driver
SFSU Email
Password
Submit
Forgot Password?

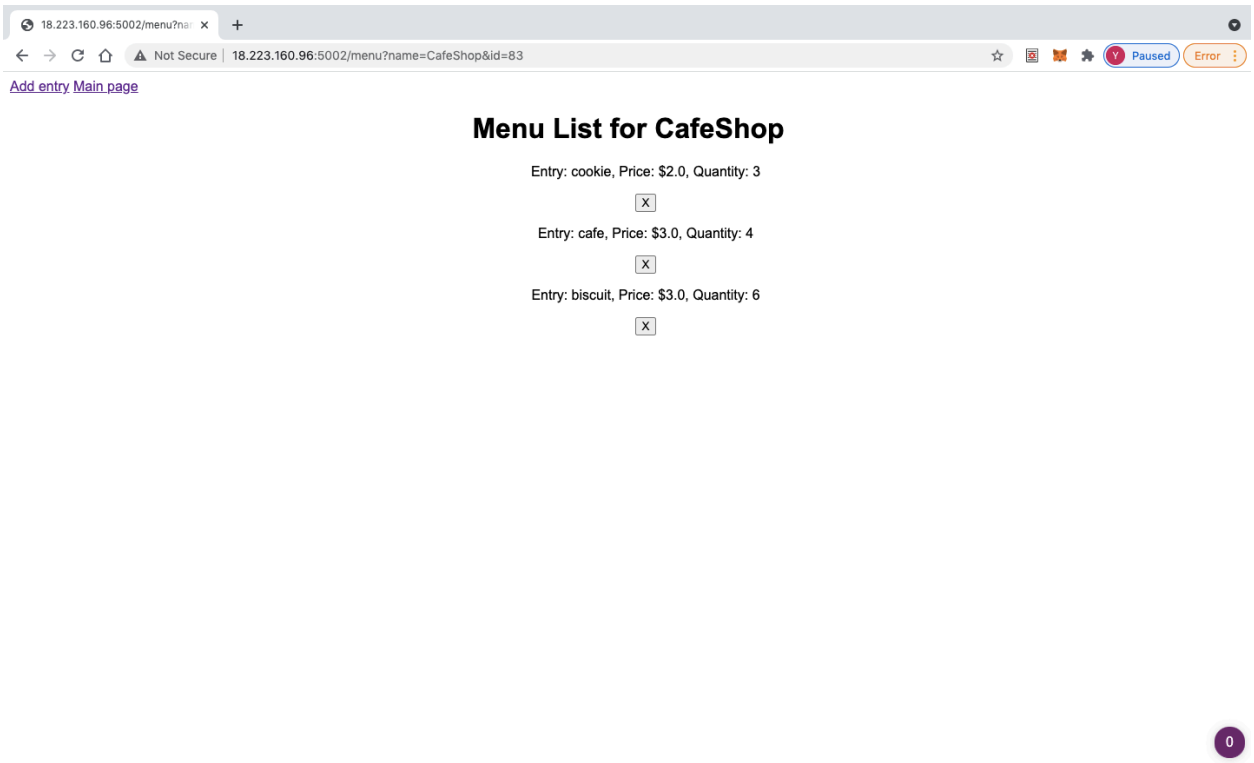

127.0.0.1:5000/regsf.html
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Admin
Login
Register
SFSU Student|Faculty|Staff
Full name
SFSU Email(Username)
Create a Password
Confirm Password
I accept the Terms and Conditions.
Submit
Already a member? Log in!

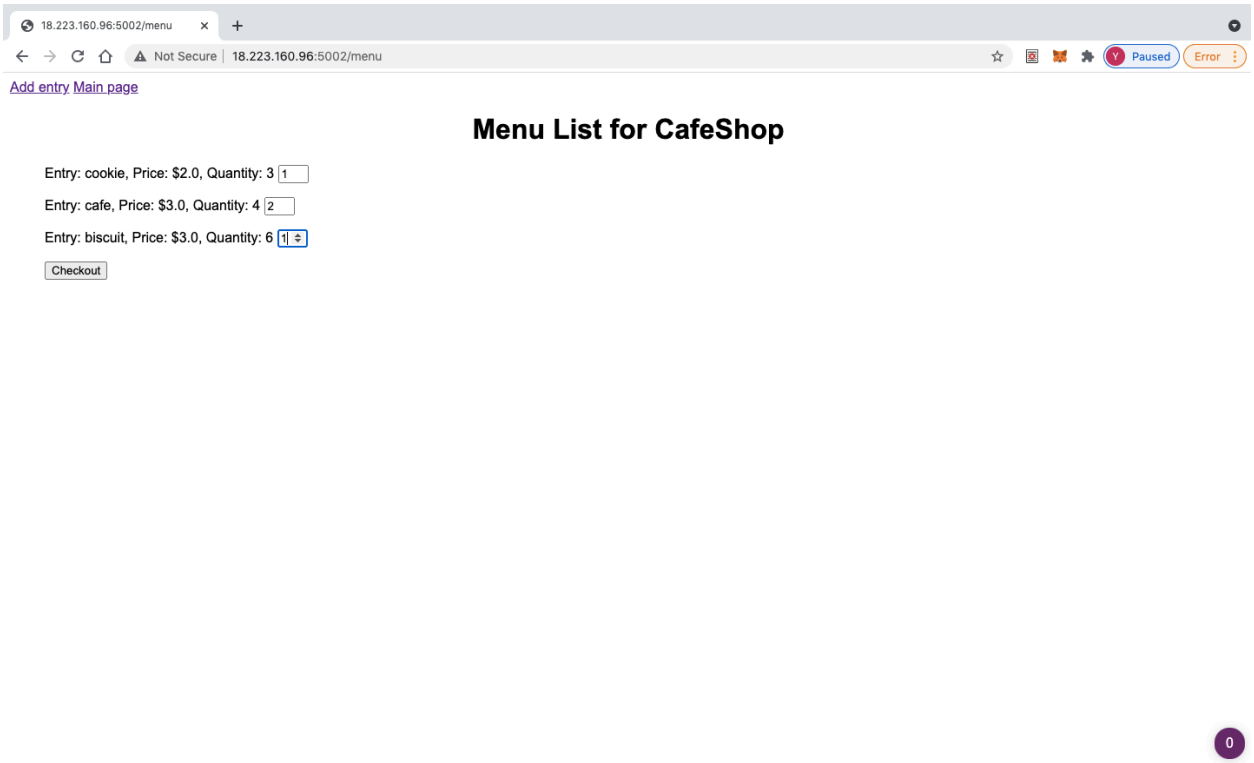

127.0.0.1:5000/regowner.html
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
Register
Restaurant Owner
Full name
SFSU Email(Username)
Create a Password
Confirm Password
I accept the Terms and Conditions.
Submit
Already a member? Log in!

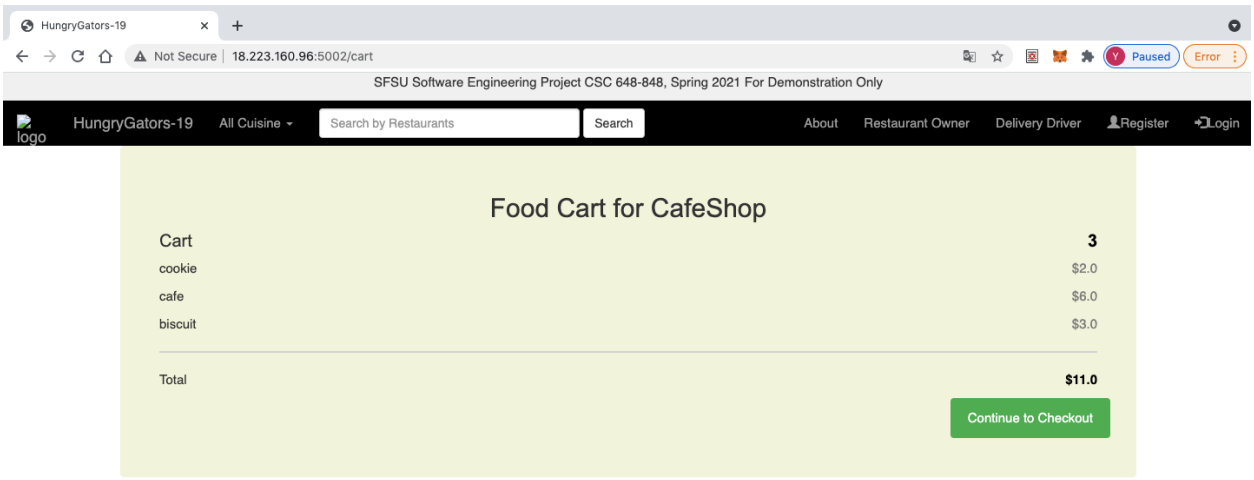

127.0.0.1:5000/regdriver.html
HungryGators-19
a class="navbar-brand"
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
Register
Delivery Driver
Full name
SFSU Email(Username)
Create a Password
Confirm Password
I accept the Terms and Conditions.
Submit
Already a member? Log in!

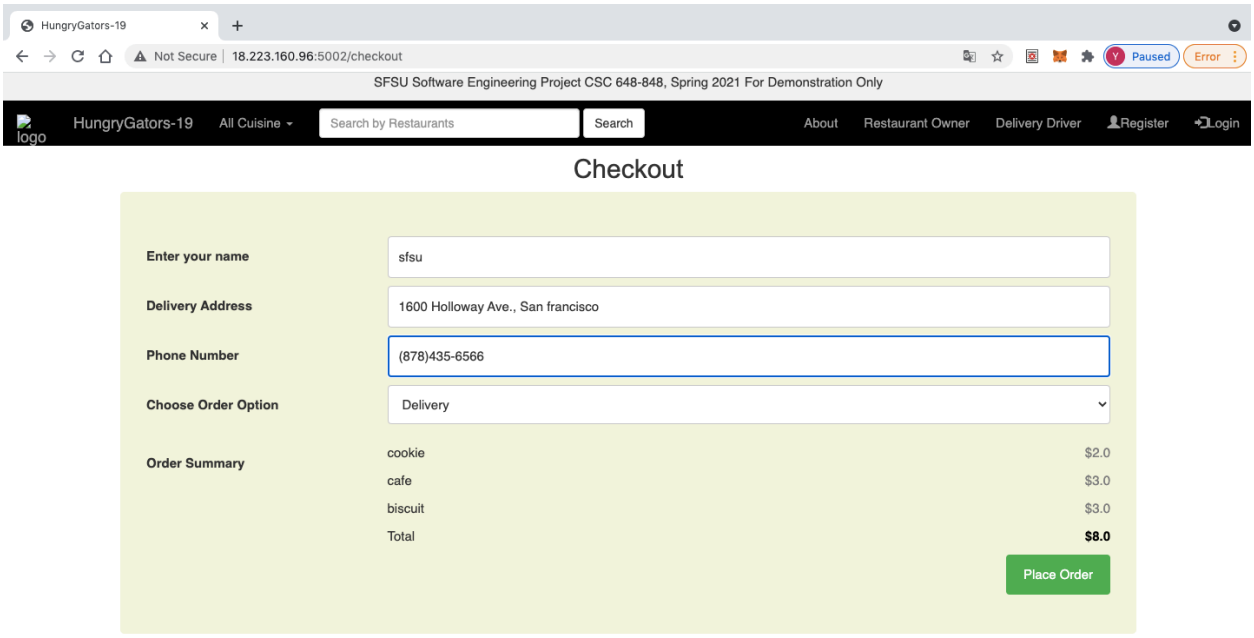

All Restaurants
Not Secure | 18.223.160.96:5002/show_restaurants?owner_id=16&owner_name=owner2
Paused
Error
Map
Satellite
Lake Merced Park
Daly City
Map data ©2021 Google
Terms of Use
Report a map error
CafeShop · american
cafe - biscuit - cookie
(424)656-5432
1655 Holloway Ave., San Francisco
95421
Add Menu Delete Restaurant
Map
Satellite
Lake Merced Park
Daly City

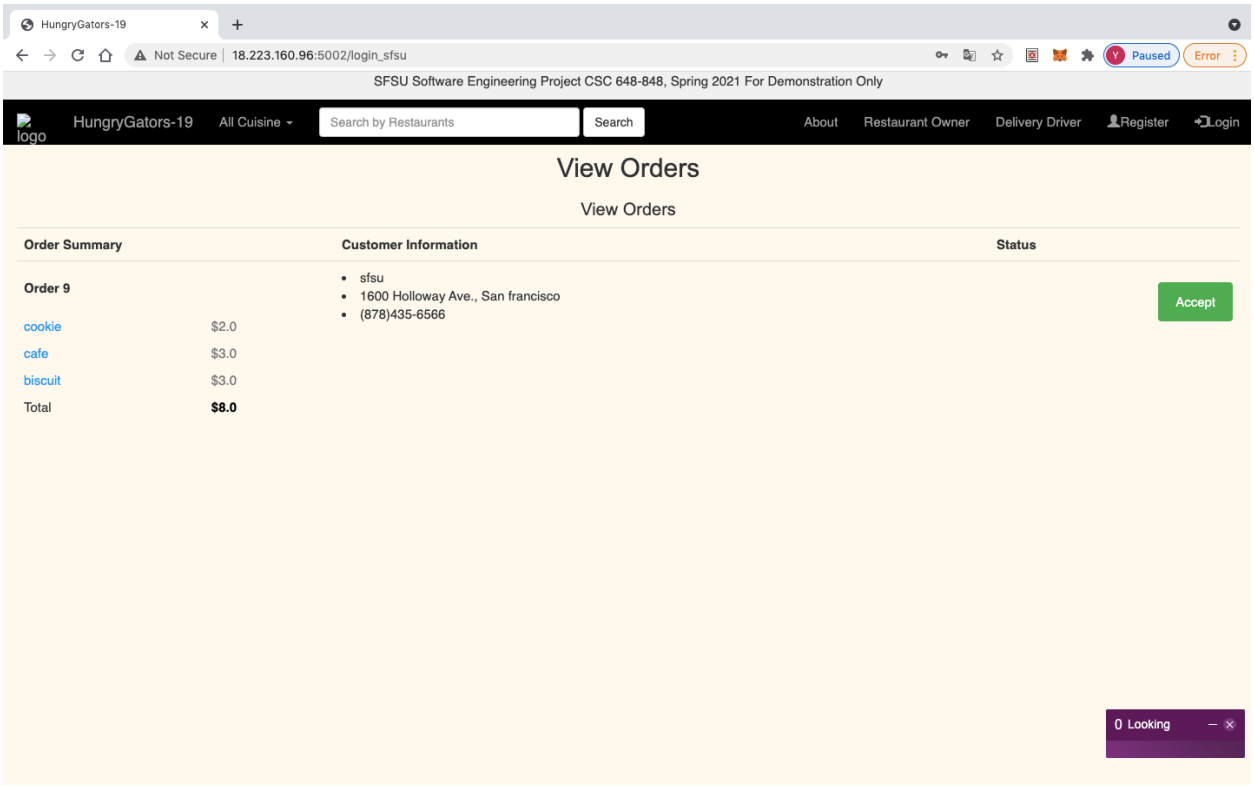

127.0.0.1:5000/login_sfsu
Add Restaurant
Show all restaurants
Name
Address
Phone number
Zip code
Description
Price Tag $$$$$
Select a cuisine: American Asian European African
Upload an image: Choose File no file selected
Add

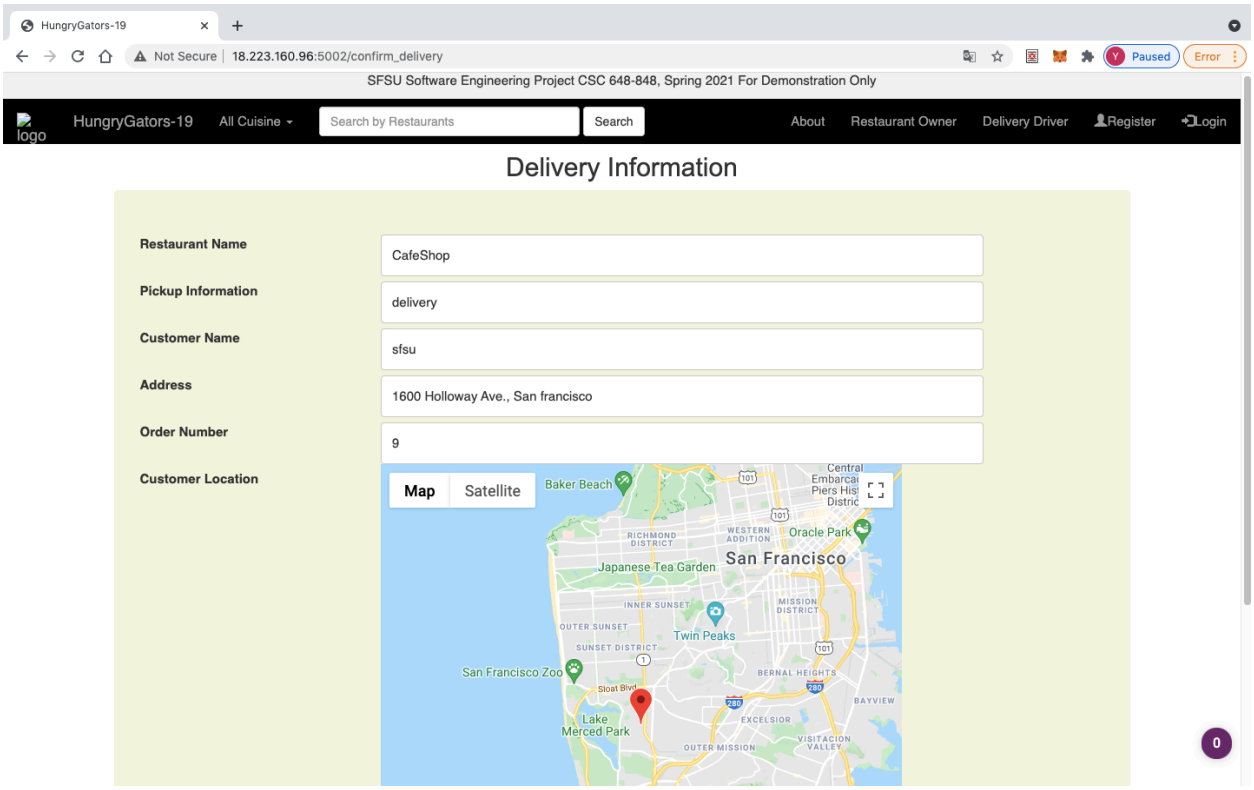

18.223.160.96:5002/menu?na
Not Secure | 18.223.160.96:5002/menu?name=CafeShop&id=83
Paused
Error
Add entry Main page
Menu List for CafeShop
Entry: cookie, Price: $2.0, Quantity: 3
X
Entry: cafe, Price: $3.0, Quantity: 4
X
Entry: biscuit, Price: $3.0, Quantity: 6
X
0


18.223.160.96:5002/menu
Not Secure | 18.223.160.96:5002/menu
Paused
Error
Add entry Main page
Menu List for CafeShop
Entry: cookie, Price: $2.0, Quantity: 3 1
Entry: cafe, Price: $3.0, Quantity: 4 2
Entry: biscuit, Price: $3.0, Quantity: 6 1
Checkout
0

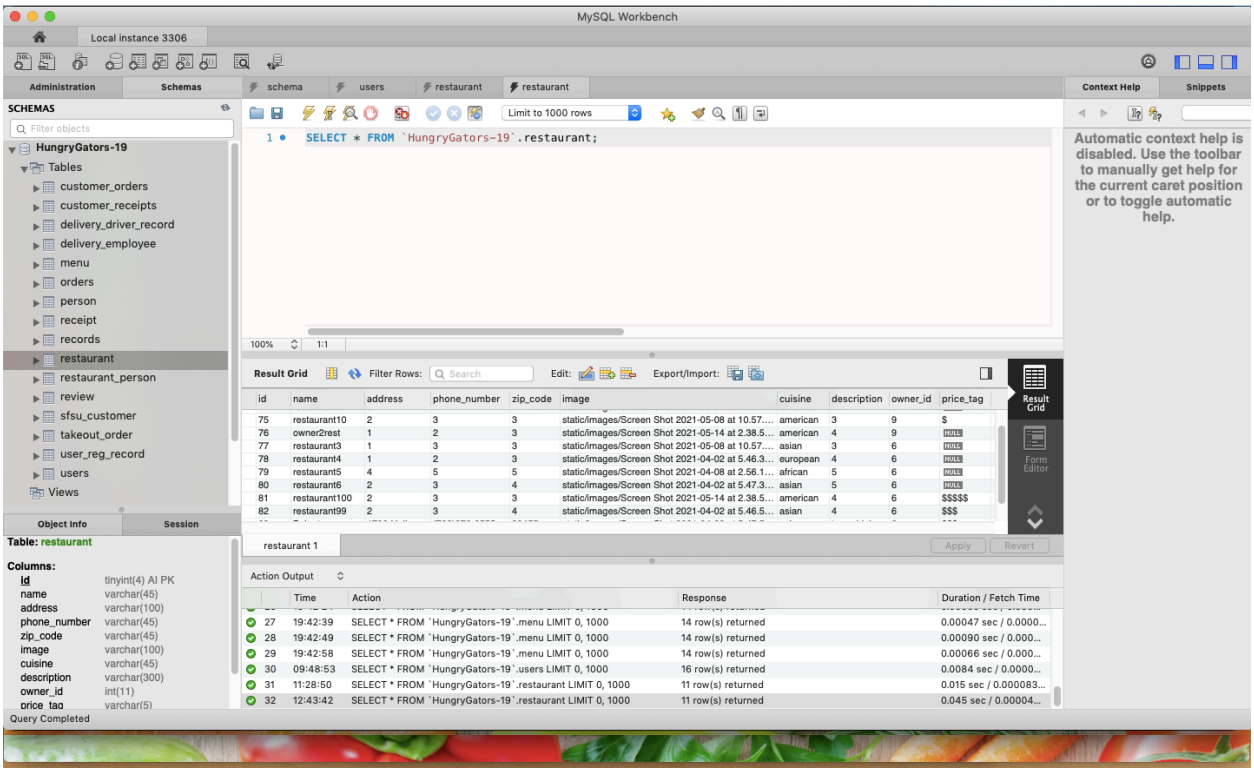

HungryGators-19
Not Secure | 18.223.160.96:5002/cart
Paused
Error
SFSU Software Engineering Project CSC 648-848, Spring 2021 For Demonstration Only
logo
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
Food Cart for CafeShop
Cart
3
cookie
$2.0
cafe
$6.0
biscuit
$3.0
Total
$11.0
Continue to Checkout

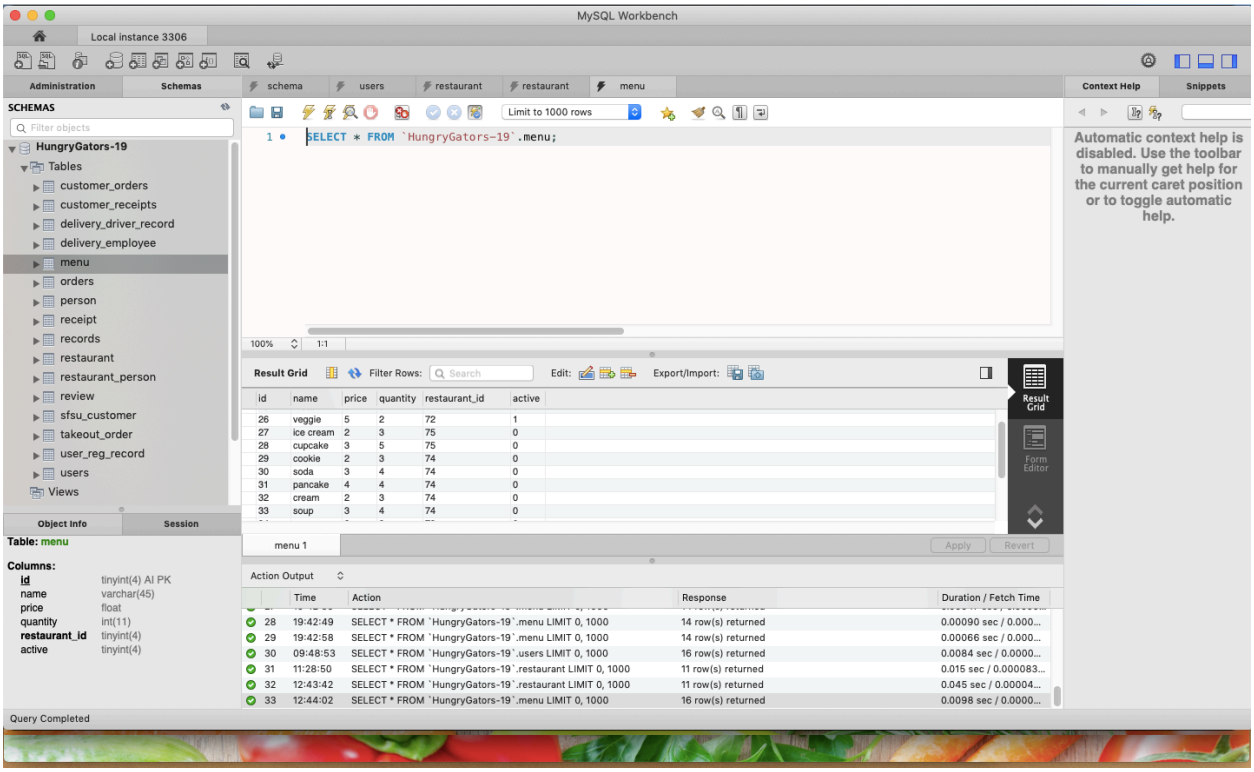

HungryGators-19
Not Secure | 18.223.160.96:5002/checkout
Paused
Error
SFSU Software Engineering Project CSC 648-848, Spring 2021 For Demonstration Only
logo
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
Checkout
Enter your name
sfsu
Delivery Address
1600 Holloway Ave., San francisco
Phone Number
(878)435-6566
Choose Order Option
Delivery
Order Summary
cookie
$2.0
cafe
$3.0
biscuit
$3.0
Total
$8.0
Place Order

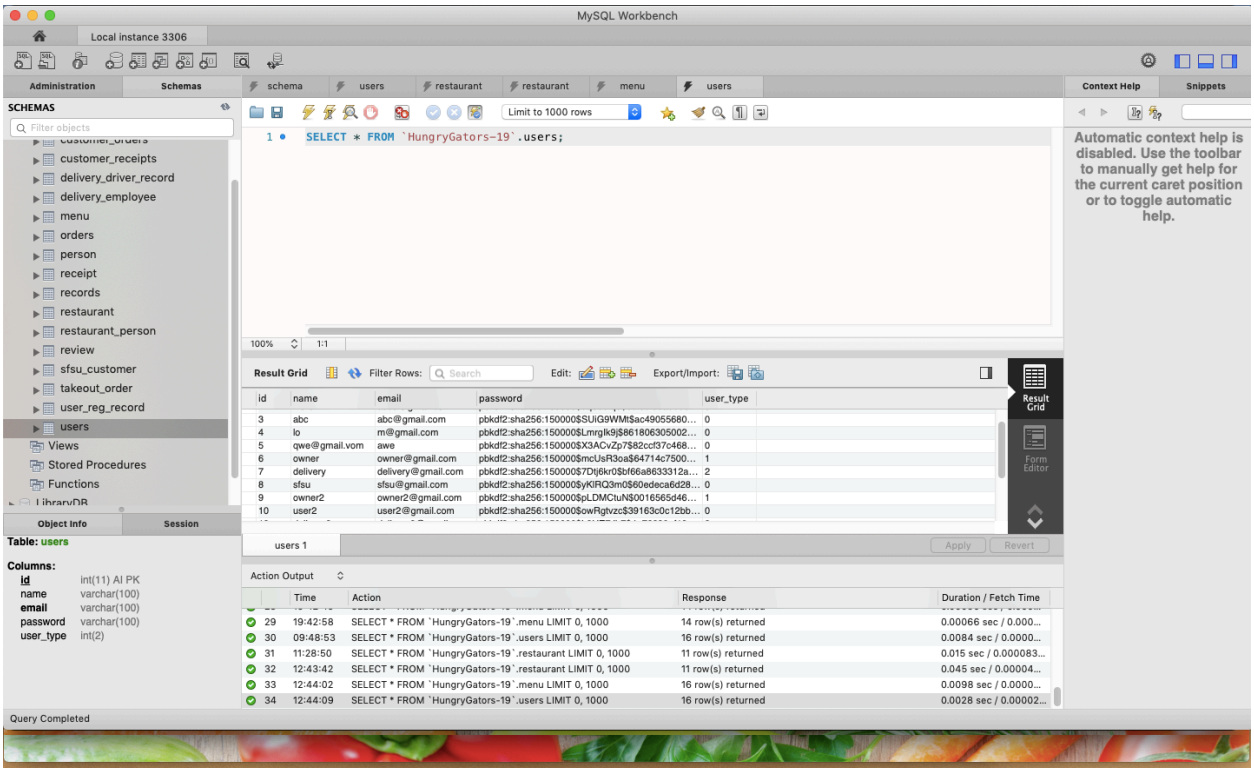

HungryGators-19
Not Secure | 18.223.160.96:5002/login_sfsu
Paused
Error
SFSU Software Engineering Project CSC 648-848, Spring 2021 For Demonstration Only
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
View Orders
View Orders
Order Summary
Customer Information
Status
Order 9
sfsu
1600 Holloway Ave., San francisco
(878)435-6566
Accept
cookie $2.0
cafe $3.0
biscuit $3.0
Total $8.0
0 Looking

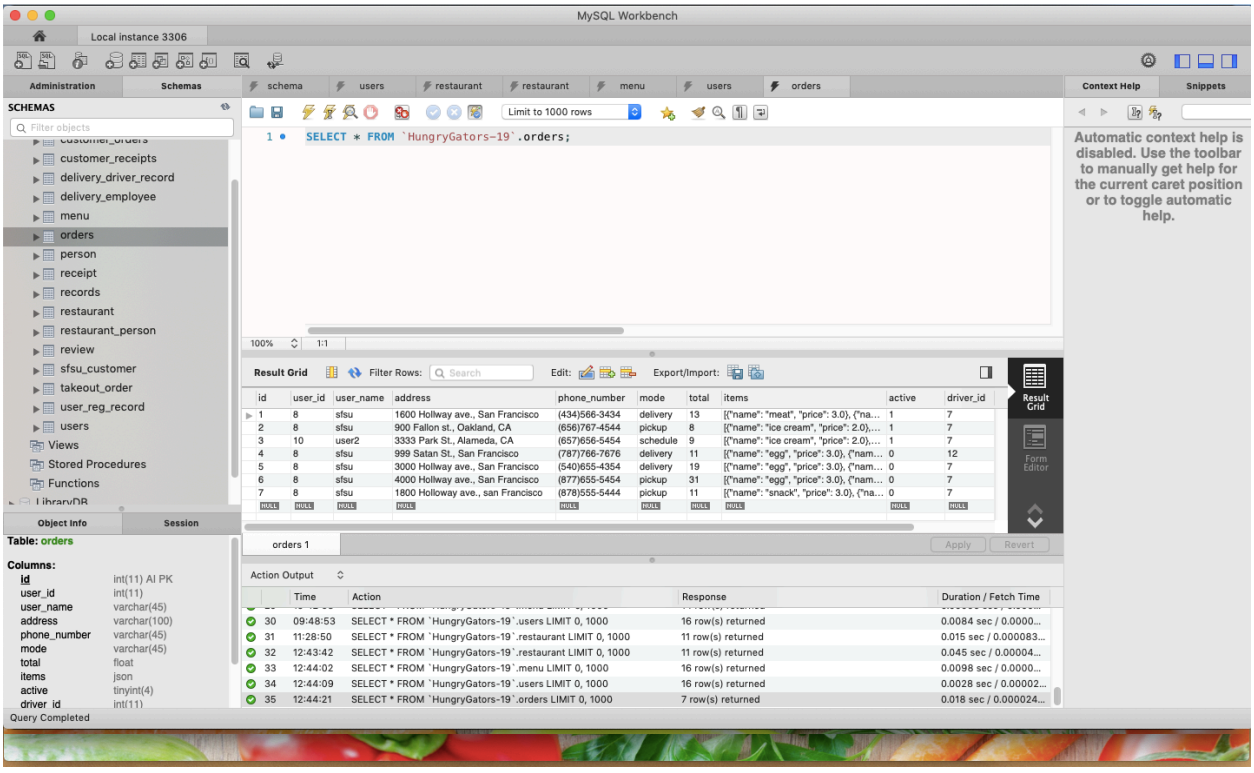

HungryGators-19
Not Secure | 18.223.160.96:5002/confirm_delivery
Paused
Error
SFSU Software Engineering Project CSC 648-848, Spring 2021 For Demonstration Only
HungryGators-19
All Cuisine
Search by Restaurants
Search
About
Restaurant Owner
Delivery Driver
Register
Login
Delivery Information
Restaurant Name
CafeShop
Pickup Information
delivery
Customer Name
sfsu
Address
1600 Holloway Ave., San francisco
Order Number
9
Customer Location
Map
Satellite
Baker Beach
San Francisco
Oracle Park
Japanese Tea Garden
Twin Peaks
San Francisco Zoo
Lake Merced Park
0

IV. Database Organization: Screenshots of key DB tables (1-2 pages)



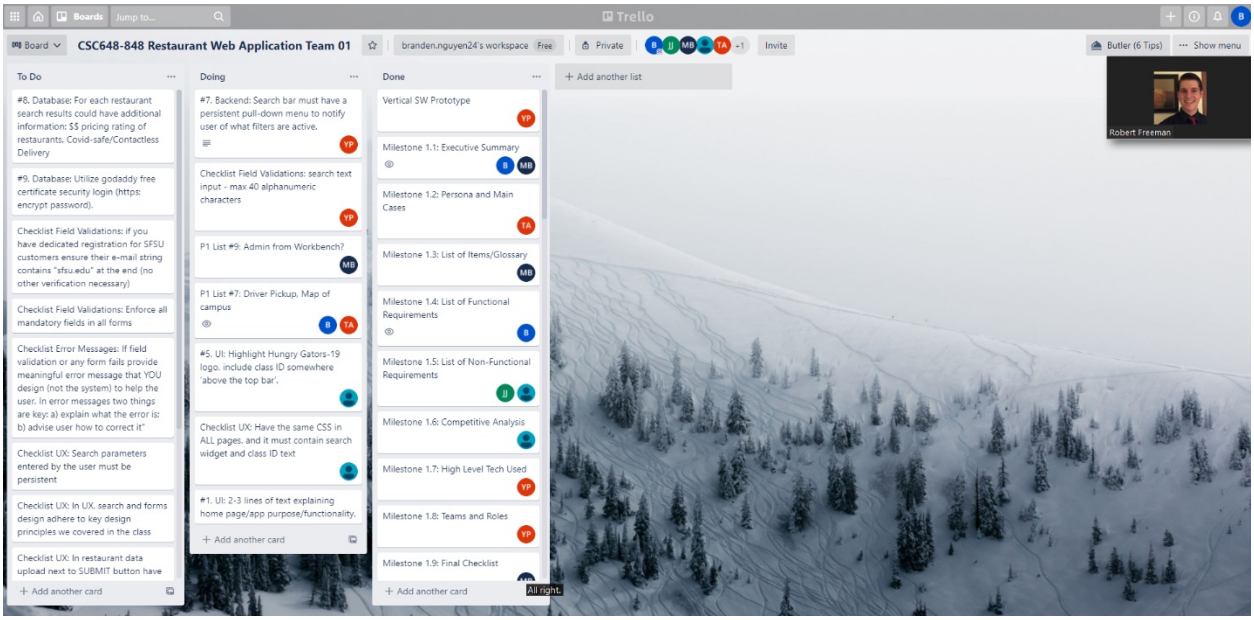


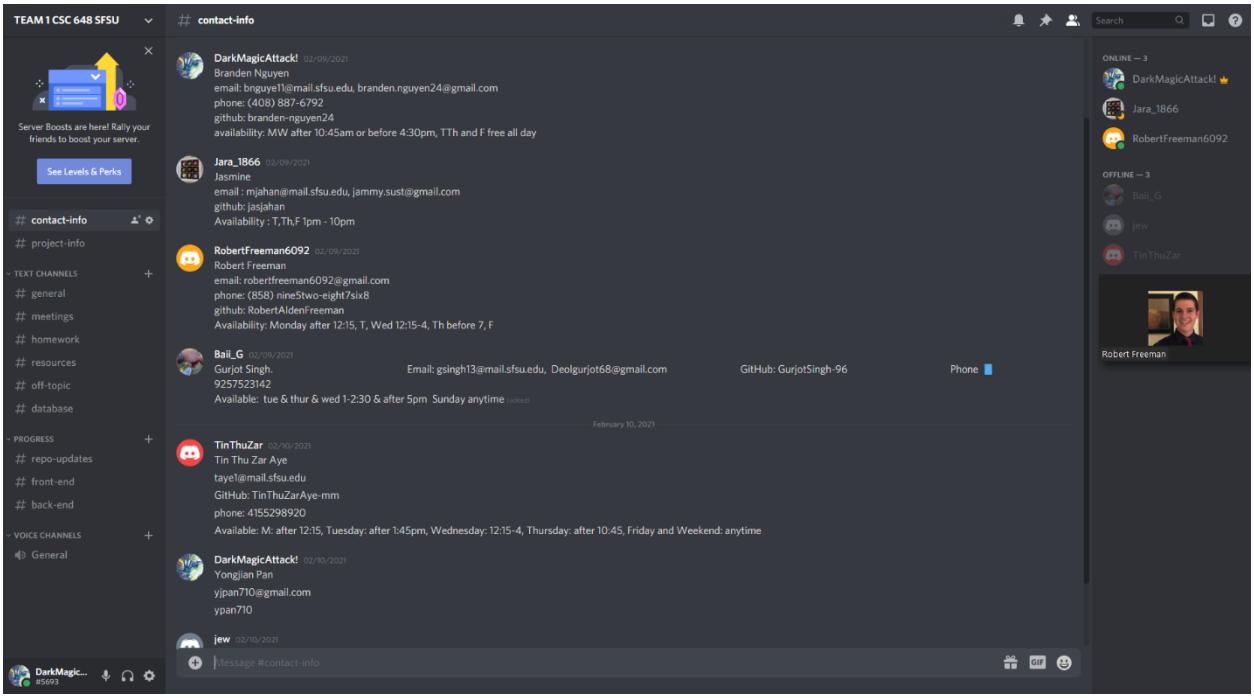

MySQL Workbench
Local instance 3306
SELECT * FROM `HungryGators-19`.users;
Result Grid
Action Output
Query Completed


MySQL Workbench
Local instance 3306
SELECT * FROM `HungryGators-19`.orders;
Result Grid
Action Output
Query Completed

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

VI. Project management: Screenshot(s) of your project management system (like Trello) showing snapshot(s) of your project management (show 2-3 screens)





VII. Team member self assessment and contributions

CSC 648 - Team Member Self Assessment and Contributions

RF

Robert Freeman <robertfreeman6092@gmail.com>
Fri 5/21/2021 6:01 PM

To:

- Gurjot Singh;
- Jasmine Jahan;
- Branden Justin Nguyen;
- Tin Thu Zar Aye;
- Yongjian Pan

a) My contributions to the teamwork:

- GitHub tutorials and branch structure setup
- Relational database with MySQL Workbench
- Merging others' work and fixing github problems and errors
- Frontend templates and styling
- Milestone paperwork several sections on all milestones

b) Number of submissions I made to the develop branch

- 58 commits
- 1,739,876 additions
- 2,707,617 deletions

c) The main challenges I faced were database creation, GitHub management (merging others' code) and interpersonal communication. It was a huge challenge to try to merge backend code to front end without manually moving files and folders which made my head spin.

d) Next time I think I would check in with teammates more and maybe merge develop back into their feature branches and tell them to pull. This would be better since pulling is such an important part of the project.

--
Robert Freeman

CSC 648 - Team Member Self Assessment and Contributions

Jasmine Jahan
Fri 5/21/2021 11:17 PM

To:

- Robert Alden Freeman;
- Branden Justin Nguyen;
- Gurjot Singh;
- Tin Thu Zar Aye;
- Yongjian Pan

a) My contributions to the teamwork:

- Made mockups in Figma.
- Created the initial workflow for the project.
- Designed User Interface considering page layout, consistency, and customer satisfaction.
- Created frontend page templates with Bootstrap, Flask, HTML, CSS, Google Maps(Embedded).
- Helped others to set up the project locally according to specific packages in PyCharm.
- Implemented inheritance coding style in Flask to avoid code duplication.
- Worked as a lead for the M2 document portion.

b) Number of submissions I made to the develop branch

- 20 commits
- 278,849 additions
- 1,408 deletions

c) The main challenges for me were fixing bugs, communicating with the team members, learning new technologies within a short period of time. Unfortunately, I lost some of my front-end work due to not pulling from the backend side. It was hard to connect front page templates to the backend since the URLs were not pushed into GitHub.

d) Next time I will try to maintain more work management products like Trello from the beginning of the project. So that every team member can collaborate equally for the project.

Regards,
Jasmine.

CSC 648 - Team member self assessment and contributions

GS

Gurjot Singh
Sat 5/22/2021 12:32 AM

To:

- Robert Alden Freeman;
- Branden Justin Nguyen;
- Jasmine Jahan;
- Tin Thu Zar Aye;
- Yongjian Pan

a) My contributions to the team project

- Helped front end team
- List of Non-Functional Requirements
- created the Html pages
- Helped to fix the links to the navbar
- Competitive Analysis
- Fixed Hungry Gators-19 logo, include class ID above the top bar
- Helped Front end Fixing CSS pages
- Worked on M4 Document

b) Number of submissions I made to GitHub team Dev. branch

- 11 Commits
- 683 additions
- 382 deletions

c) The Main challenge I encountered in a team project was that I had to fix bugs regarding class id and logo. Also, poor communication with the backend. Connecting frontend work with the backend was hard since the backend was barely pulling our work. Lack of communication problem.

d) Next Time, I would like to find a better way to connect with the whole team so everyone is on the same page. I would also like to make sure that everyone is having the updated files when someone pushes to the branch so, no one is working on old files.

CSC 648 - Team Member Self Assessment

BN

Branden Justin Nguyen
Sat 5/22/2021 2:11 AM

To:

- Jasmine Jahan;
- Robert Alden Freeman;
- Tin Thu Zar Aye;
- Gurjot Singh;
- Yongjian Pan

a) My contributions to the teamwork:

- Setup, organized, and maintained project management tools such as Trello and team Discord channel to promote team communication and collaboration.
- Scheduled and ran weekly class and outside of class team Zoom meetings.
- Distributed and organized weekly workflow tasks through SCRUM like focus groups via Trello.
- Maintained communication between Class CEO and CTO for Milestone submissions.
- Edited and collaborated on multiple sections of all Milestone documents--team scribe and wrote M3 and M5 documents.
- Ensured team members meet milestone and deliverable deadlines and adhere to project specifications and requirements.
- Assisted team members through focus groups of mockups, workflow, troubleshooting bugs, coding fixes, QA and testing of features.
- Assisted GitHub Master with refactoring of GitHub repository and bug testing.
- Attempted to keep morale and motivations high and deadlines firm, despite increasingly high amounts of anxiety, stress, frustrations, clashing personalities and work styles throughout multiple instances of the project experience for all members involved.

b) Number of submissions I made to the develop branch

- 11 commits
- 96 additions
- 53 deletions

c) The main challenge I faced as a team lead was being able to successfully unite six very different individuals in terms of personality, working styles, and learning habits, to organize and maintain an efficient workflow and collaboration process to achieve our goal. From the beginning of the semester, all of us started with high hopes and motivated hearts that were full of ambition and an eagerness to learn. However, all of that changed very quickly as soon as wave after wave of bugs started plaguing our workflow mainly in terms of our project structure and GitHub usage.

Every time a member of the team was able to learn or implement a certain feature and push, the resulting pull request caused a vast array of errors tracing back to the Pycharm project settings and virtual environment. Each time someone pulled, they would first have to resolve the same errors every time and reimport the entire project library and settings. It was extremely frustrating for all members involved. And that built up frustration soon created an atmosphere of independent learning. All of us started working individually to try and implement whatever features we could. The GitHub errors promoted non-GitHub usage until those errors could be fixed. That lead to us at one point having the obscenely ridiculous file structure of 5 different app.py files at one point within the root of our repository, not knowing what resources or source files were linked to what (even Pycharm couldn't tell what our proper file structure was, and I don't blame it!)

We miraculously kept up with the workflow of the class as best we could, while in the meantime attempting to troubleshoot and solve the bugs one at a time. Eventually, we were able to pinpoint the main cause of our errors to be the .gitignore file was not properly written to ignore .xml files as well as our terrible project structure. From there, me and our GitHub Master took an entire afternoon to say enough is enough and we went line by line, error by error, until we refactored the entire GitHub repository to finally have members pull successfully without encountering a single error. It was a huge accomplishment for us as a team, maybe finally we could have a real collaborative process without any more headaches.

Unfortunately, our problems were just beginning. Our real problem, the one that plagued us the most throughout the entire semester was our lack of effective communication and collaboration with certain team members. In the end, we had a dysfunctional workflow that made absolutely no sense and made it literally impossible to collaborate effectively and efficiently using the Agile SW processes that were required from this class and of this project and its members. The frontend and backend had their own independent implementations, and they were never able to be truly synced or brought together because unfortunately the backend could not or would not pull the project repository. Because of this misuse of GitHub protocol, the team was split and either frontend or backend would have to literally wait for the other at times to begin work. What made the situation even more demoralizing and demotivating for all members was that because the backend did not pull or fork on a new branch before pushing, there would be sections of work completely lost which would take even longer to try and retrieve through GitHub processes. This caused a lot of demotivation and despair amongst many team members. The entire collaborative workflow element was completely nonsensical to an almost comedic level and impossibly difficult to resolve. This

ultimately led to a lot of tensions, bitter frustrations, and anxiety throughout the entire process.

d) Coming into this class I thought based on my previous experience in my Multiplayer Game Development class as a team lead, I would be able to emulate my previous success. I understand now that I had much to learn and that every team is different. Every person has different personalities, needs, and communicates, and learns in different ways. I feel like I tried so hard to attempt to unite and motivate everyone to be as open, communicative, and collaborative as possible. However, clearly it was a struggle and maybe my efforts were not enough.

I think I should have attempted to schedule more one on one meetings with every team member to truly understand their main pain points and most importantly analyze the major setbacks keeping people from collaborating effectively with one another and come up with well-designed solutions. Moreover, I would attempt to be more demanding, without being a tyrannical babysitter, of team communication and accountability inside and outside of the classroom and group meetings. Everyone needs to be on the same page in a group project and if there is even one person who is not synced with the group, it can potentially cause serious problems for everyone. This relates to making sure everyone is aware of deadline dates, deliverable project and document requirements and ensuring everyone remains focused, engaged, and accountable.

I understand that throughout this semester I made some slip ups in terms of attention to detail myself, I was far from perfect. All I can say is that in the end, I've truly learned and grown from this experience as a computer scientist and as an individual. I learned so many things about being a project manager/team lead, responsibility, accountability, and the effectiveness of communicating and motivating others. I understand I have much to improve on for that I am truly sorry things did not go as smoothly as they could have, but I know in my heart I truly, genuinely gave it my all and I care about the well-being and success of every one of my team members.

Thank you everyone for all your efforts, pain, and sacrifice. I could not have done any of this without each of you. Thank you for everything you've taught me throughout this semester.

Best,

-Team 01, Branden J. Nguyen

CSC 648 - Team Member Self Assessment and Contributions

T

Tin Thu Zar Aye
Sat 5/22/2021 3:50 AM

To:

- Branden Justin Nguyen;
- Jasmine Jahan;
- Robert Alden Freeman;
- Gurjot Singh;
- Yongjian Pan

a) My contributions to the teamwork:

- List of Personas and Use Cases
- Worked on M2 document
- Helped for mockup on Figma
- Implemented for Delivery Driver Page
- Worked on M4 document

b) Number of submissions I made to the develop branch

- 4 commits
- 233 additions
- 50 deletions

c) The main challenge that I faced in this team project was communicating with team members and learning how Flask and Python work.

d) Next time I will try to collaborate and communicate with certain team member. I also try to have Scrum meeting with team members for every other day to make sure that we have on the same page and plan out for the task since the beginning of the project.

CSC 648 - Team Member Self-assessment and Contributions

YP

Yongjian Pan
Sat 5/22/2021 12:19 PM

To:

- Branden Justin Nguyen;
- Jasmine Jahan;
- Tin Thu Zar Aye;
- Gurjot Singh;
- Robert Alden Freeman

a) My contributions to the teamwork:

- Programmed in backend
- Integrated with frontend
- Connected web application to local database
- Made sure web application runs successfully locally
- Deploy web application on the cloud

b) Number of submissions I made to the develop branch

- 20 commits
- 5,406,332 additions
- 2,283,708 deletions

c) The main challenges for me were programming in backend, integrating with frontend, connecting the backend to the database, and deploying the web application on the schedule on time to meet milestones. In addition, there were challenges to pulling and pushing to Github and creating fork branches effectively. Finally, communicating with teammates are also challenging as I can't meet with them in-person and has through telecommunication tools like Zoom and Discord.

d) I will next time I will learn more about the web technologies and how to work in an online environment more effectively and efficiently.