Final Project for SW Engineering Class CSC 648-848 Spring 2020

Team 04

05/21/2020

SFStateEats

http://3.12.102.223:4000/

Team Members

Name	Email	Role
Rachit Joshi	rjoshi@mail.sfsu.edu	Team Lead
Samhita Brigeda	pandu.barigeda@gmail.com	Front End Lead
Pedro Souto	Pedro.Souto.SFSU@gmail.com	Back End Lead
John Pham	JohnPhamDeveloper@hotmail.com	Github Master
Vincent Tran	vtran6@mail.sfsu.edu	Database Lead
Khang Tran	ktran26@mail.sfsu.edu	Frontend Developer

Milestone 2 Version History

Milestone Version	Date
Milestone 5 Version 1	05/21/2020
Milestone 4 Version 2	05/19/2020
Milestone 4 Version 1	05/11/2020
Milestone 3 Version 2	04/30/2020
Milestone 3 Version 1	04/23/2020
Milestone 2 Version 2	04/09/2020
Milestone 2 Version 1	03/22/2020
Milestone 1 Version 2	03/10/2020
Milestone 1 Version 1	02/24/2020

Table of Contents

Section 2: Product Summary

Name of Product

SFStateEats

Committed Functionalities

User

- New users shall be able to create an account.
- Users shall be able to login into their account.
- Users shall be able to rate restaurants on campus.
- Users shall be able to rate an event on campus.
- Users shall be able to write reviews.
- Users shall be able to view reviews.
- Users shall be able to see the menu for each restaurant.
- Users shall be able to see the price for each item on the menu.
- Users shall be able to see the address of the restaurant on the profile page.
- Users shall be able to see hours of business for each restaurant.
- Users shall be able to apply filters to their search.
- Users shall be able to flag a review as inappropriate for the application.
- Users shall be able to sort reviews by date.
- Users shall be able to sort reviews by popularity.
- Users shall be able to delete a review they have written.
- Users shall be able to see all reviews they have written.
- Users shall be able to see all reviews another user has written.
- Users will be able to see a visual representation of the ratings, which shall be displayed on a scale of 0-5 stars.
- Users shall be able to reply to other user's reviews.[(Unique Feature)]

Business

- New business owners shall be able to register a business account.
- Business owners shall be able to login into their account.
- Business owners shall be able to add their restaurant to the website.
- Business owners shall be able to create a menu for their restaurant.
- Business owners shall be able to upload photos of their restaurant.
- Business owners shall be able to write descriptions of their restaurant on the profile page.
- Business owners shall be able to change prices on the menu.

- Business owners shall be able to change hours of the restaurant.
- Business owners shall be able to flag a review as inappropriate for the website.
- Business owners shall be able to request removal of their restaurant from the website.
- Business owners shall be able to upload photos of each item on the menu.

Restaurant

- The restaurant profile page shall display photos.
- The restaurant profile page shall display ratings.
- The restaurant profile page shall display tags.
- The restaurant profile page shall display a description.
- The restaurant profile page shall display address.
- The restaurant profile page shall display a menu.
- The restaurant profile page shall display hours.
- The restaurant profile page shall display reviews.
- The restaurant profile page shall display the owner.
- The restaurant profile page shall display phone numbers.
- The restaurant profile page shall display the various payment methods the restaurant accepts.

System Admin

- System Administrators shall have the privilege to ban any users from the website for misuse.
- System Administrators shall have the privilege to delete restaurants from the platform.
- System Administrators shall have the privilege to change restaurant information.
- System Administrators shall have the privilege to delete reviews from the platform.

System

- The system shall show new restaurants on the main page.

Unique Features

On our campus, there is a lack of information on smaller restaurants and food locations. **SFStateEats** will be a service that allows its users to rate, view, and discover all food places on San Francisco State University Campus. It will also allow restaurant owners to add their restaurants to the platform.

The unique feature that our product offers is the ability to reply and leave comments to **reviews written by other users.** This will add an immense amount of user active time, since the review section of many restaurants will foster discussions. This is a great natural way to keep users on the website for longer.

This functionality also presents a benefit for the user. They are able to start discussions with someone who has left a review, and ask them to specify further what factors went into the decision of their review. For instance, if ‘Alice’ leaves a negative review of Taco Bell, ‘Bob’ is able to discuss this review with ‘Alice’ further. He may find that ‘Alice’ left a negative review due to the fact that her burger was cold. ‘Bob’ may then opt to go to the restaurants anyways, but not order a burger for himself.

URL to Product

http://3.12.102.223:4000/

Section 3: Milestone Documents M1-M4

Starting on the next page, Milestones 1-4 will be copied in their entirety. This includes everything from the cover page to the last page. These milestones have not been revised after they were frozen. They can be considered “snapshots in time”.

Milestone One

SW Engineering CSC648/848 Spring 2019
Team 04
02/24/2020

SFStateEats

Team Members

Name	Email	Role
Rachit Joshi	rjoshi@mail.sfsu.edu	Team Lead
Samhita Brigeda	pandu.barigeda@gmail.com	Front End Lead
Pedro Souto	Pedro.Souto.SFSU@gmail.com	Back End Lead
John Pham	JohnPhamDeveloper@hotmail.com	Github Master
Vincent Tran	vtran6@mail.sfsu.edu	Database Lead
Khang Tran	ktran26@mail.sfsu.edu	Frontend Developer

Milestone 1 Version History

Version Name	Date
Milestone 1 Version 2	03/10/2020
Milestone 1 Version 1	02/24/2020

Table of Contents

Section 1: Executive Summary

Currently, San Francisco State does not have an organized and easy method to view food locations on campus. There are many applications that are used to find great restaurants and popular food spots. For example, *Yelp* and *Google Restaurants* are greatly used outside of campus and for personal use. On our campus, there is a lack of information on smaller restaurants and food locations. Some food locations like cafes, restaurants, and food courts are too small to show up on some of the previously mentioned services. ***SFStateEats*** will be a service that allows its users to rate, view, and discover all food places on San Francisco State University Campus. It will also allow restaurant owners to add their restaurants to the platform.

Fact is, trends catch on. New and hot apps spread between users, and what better place to spread a new service than a close knit school campus? Our service will increase in users via a network effect, as more and more students begin to experience SFStateEats. The reach that this web application can have is enormous, with San Francisco State University being just the tip of the iceberg.

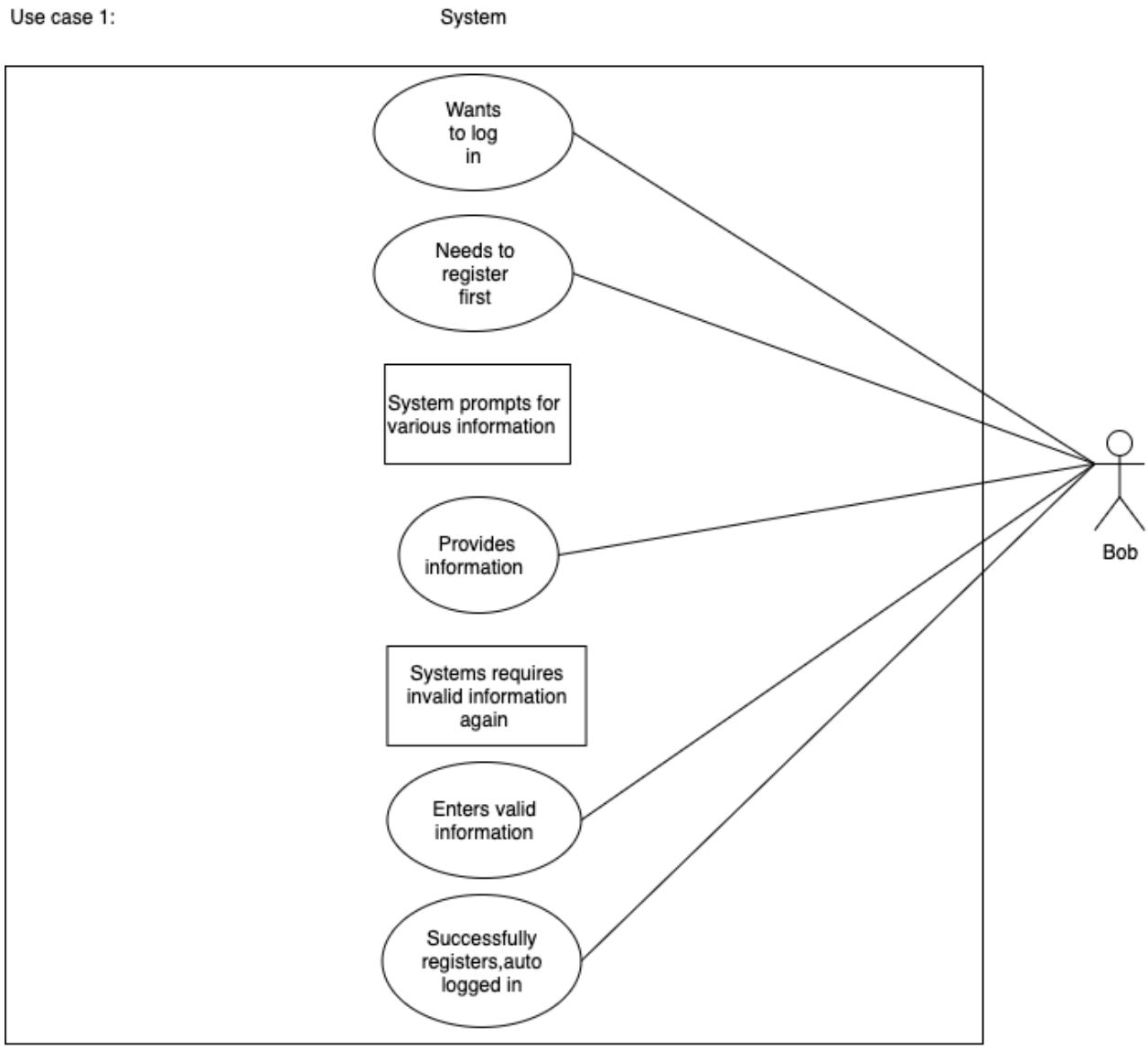
From day one, our application will be free to use. In order to survive in this competitive market, we shall implement ads throughout the application. These ads will have a much higher cost per impression (CPM) due to the fact that they are highly targeted, since the demographics is exclusively San Francisco State University students. Thus companies whose targeted demographic is college students have a great incentive to purchase ads in our application. These ads will allow us to monetize our products, without increasing the barrier to entry for new customers.

We plan to launch our web application in the beginning of summer 2020, where every student of San Francisco State University will be able to join us. When this happens, all features will go live.

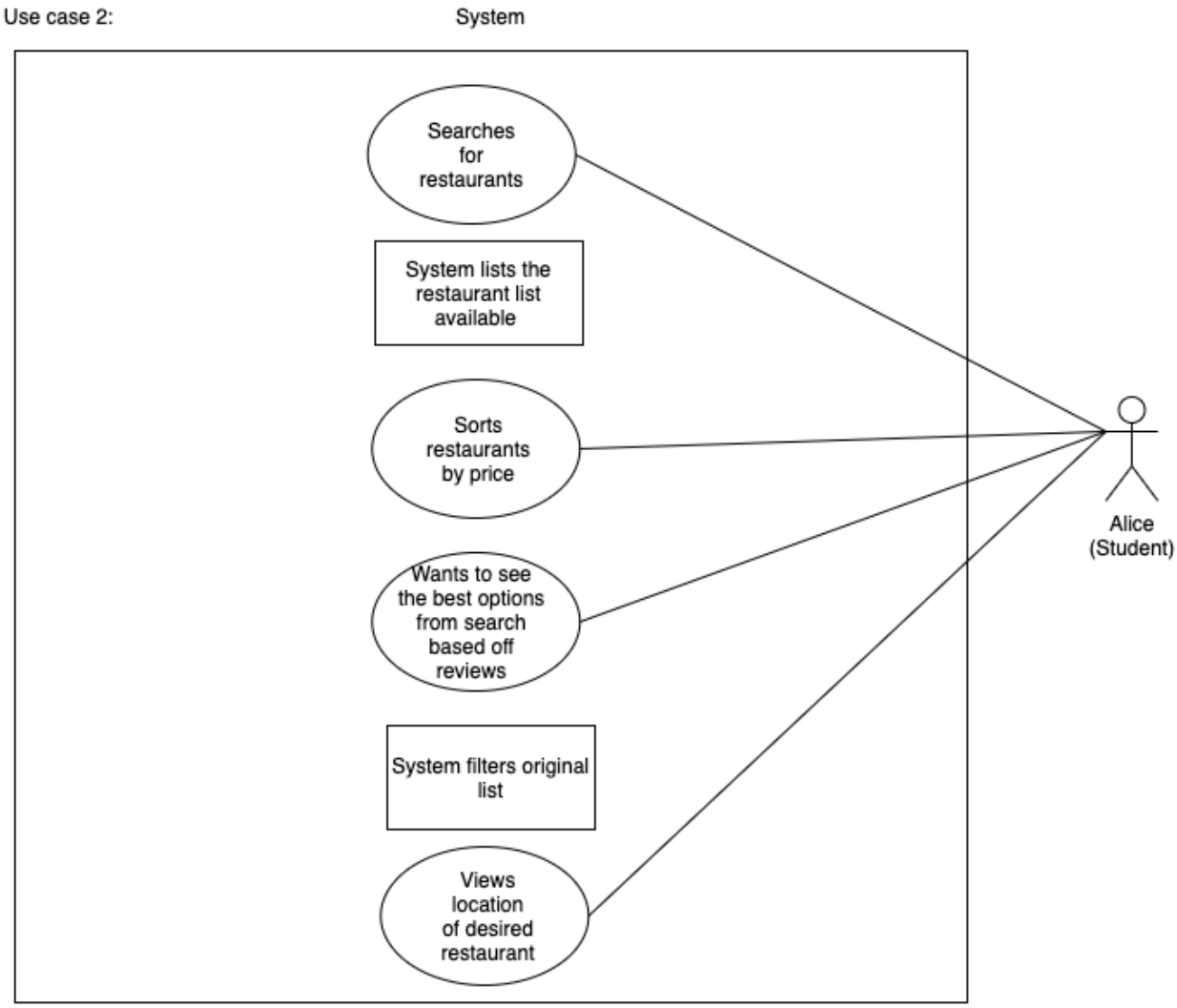
*“Our mission at SFStateEats is to make sure **every** student has access to **all** information of **every** restaurant on campus.”*

Section 2: Main Use Cases

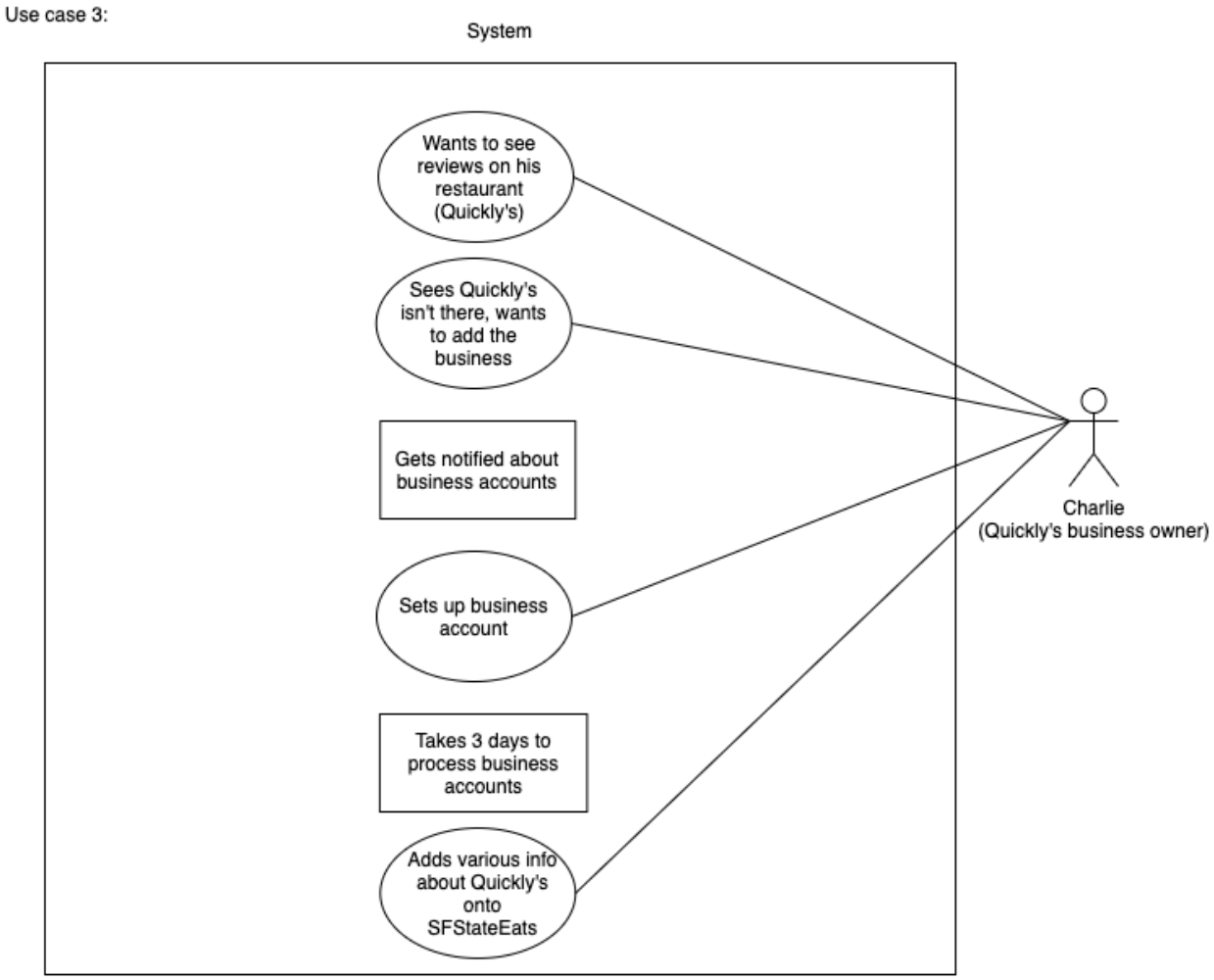
1. Bob is a Computer Science major student at San Francisco State University, attending his first semester after transferring from a local Community College. He just visited Caffe Rosso on campus for the first time and had an amazing breakfast sandwich. He wished that he knew about this place earlier, and decides to find someplace to leave a review so that others can learn about Caffe Rosso too. Bob learns about SFStateEats, and opens the website to give the restaurant a 5 star review. The SFStateEats application first asks Bob to login before he can rate a restaurant, but Bob does not have an account. Bob attempts to register for a regular user account, but the username he's trying to register with is already taken. Bob is given a prompt to try a different username instead. Once Bob has finished the registering process, he is given a message stating his registration is successful and is automatically logged in. He then proceeds to write a good review for Cafe Rosso.



2. Alice is a third year transfer from a Community College in Los Angeles, and this is her first time living alone. She has received financial aid and taken out student loans, but is still struggling to pay all her bills as San Francisco is very expensive. Due to this fact, she has been focusing on spending as little money as possible. She is on a budget and wants to find a cheap place to eat on campus before her next class. She opens the SFStateEats application and searches for every food place on campus. Since she wants the cheapest options, she sorts the list by price, then looks for a location with good reviews. She sees that Nizario's Pizza is located in the basement of the Cesar Chavez Student Center building. She also sees that this food place has 4.5 / 5 stars, and decides to try it out.

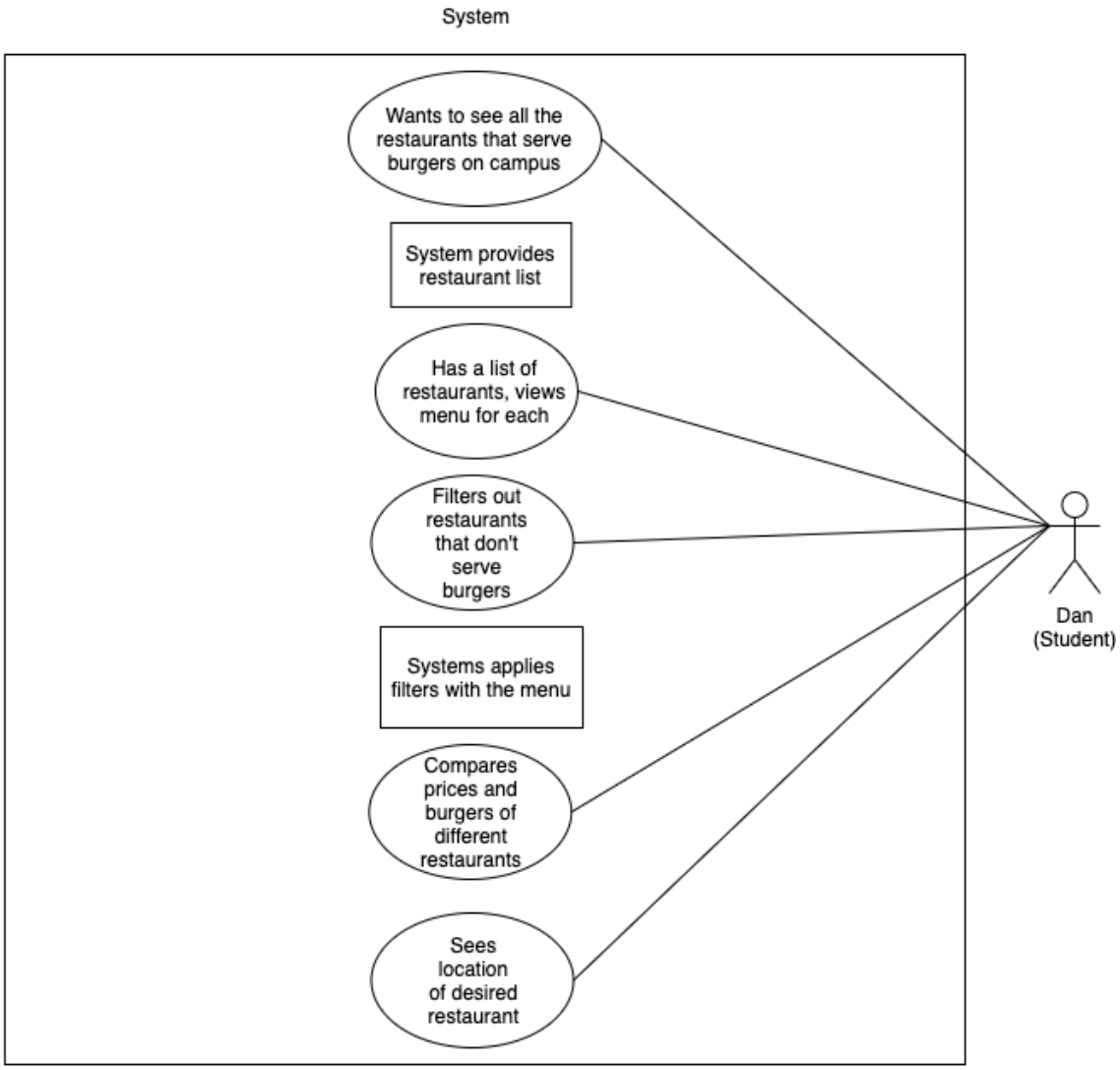


3. Charlie is not a student at San Francisco State University, but he is there almost every day because he is the owner of a food place called Quickly's on campus. He has owned this small restaurant for 8 years, and knows many of the customers personally. A few of them that he knows particularly well said they wanted to leave a great review on this new application, SFStateEats. Charlie is curious about the application. He looks up Quickly's on SFStateEats, and realizes this restaurant is not listed on the web app. He tries to add his restaurant onto the platform, but is notified that he needs a business account in order to add a new place to the platform. He starts the process to create his business account, and after 3 days, it is verified. With his business account, he adds Quickly's to the list of campus restaurants, along with the menu, hours, and prices.

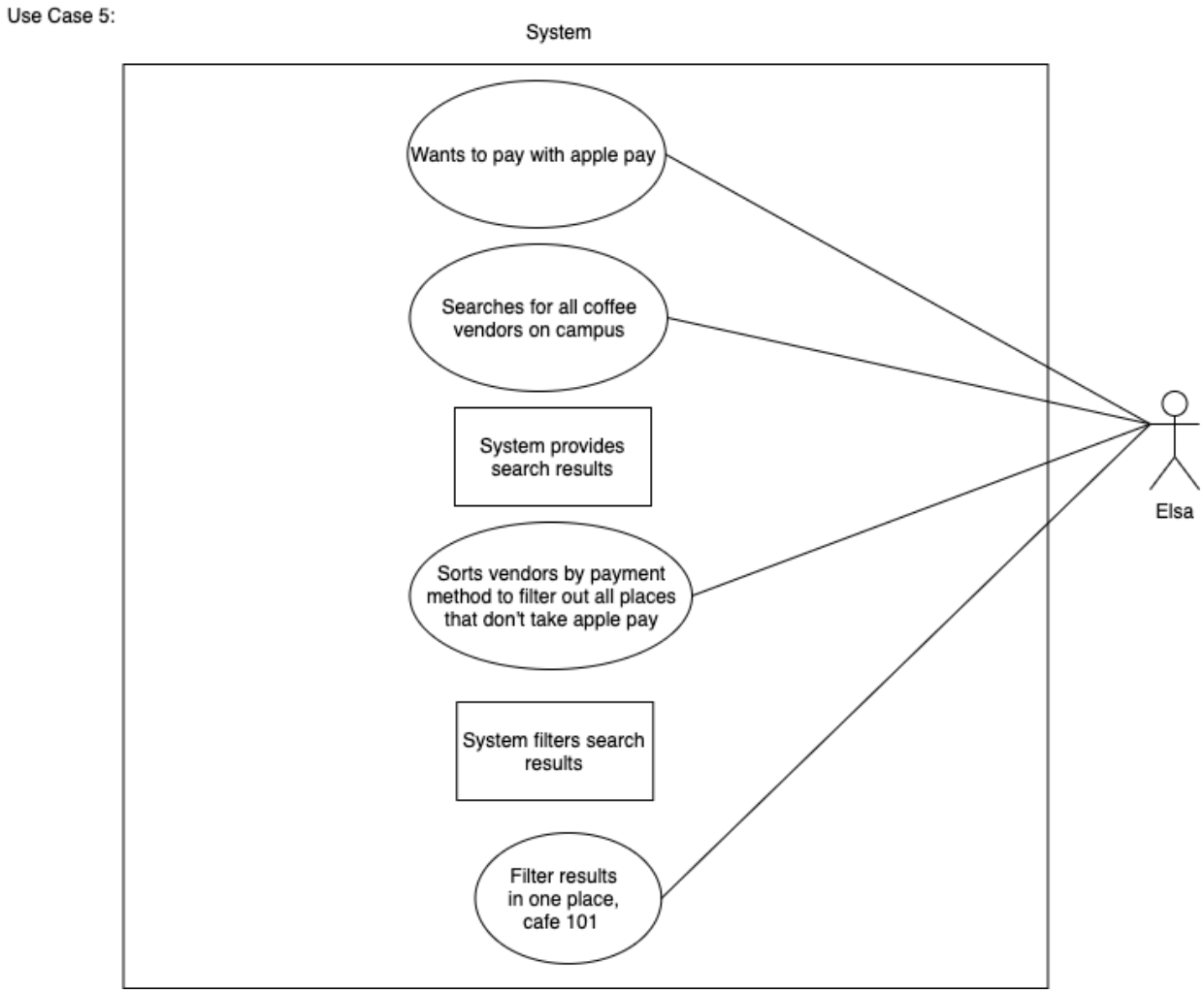


4. Dan is a new graduate student at San Francisco State University and loves trying out new food. He attended San Jose State University for 4 years prior, attaining his Bachelor's Degree in physics. While going to school at San Jose State University, he learned all the local food places, but now that he is a new student on a new campus, he feels lost on where to go. After asking some classmates, he was informed that most people on campus use SFStateEats to scout out new places, so he decides to visit the website. He is looking for a good burger for lunch today and wants to see all the burgers on campus. He opens the list of all restaurants, and pulls up the menu for each of them. He then filters out all the menus that do not have a burger as an item on it, and is left with a handful of menus that do. He compares the prices and description for all the burgers, and decides to go with the burger on the third menu. He is able to see that it belongs to a place called Burger Express, located on the first floor of the Ceasar Chavez building.

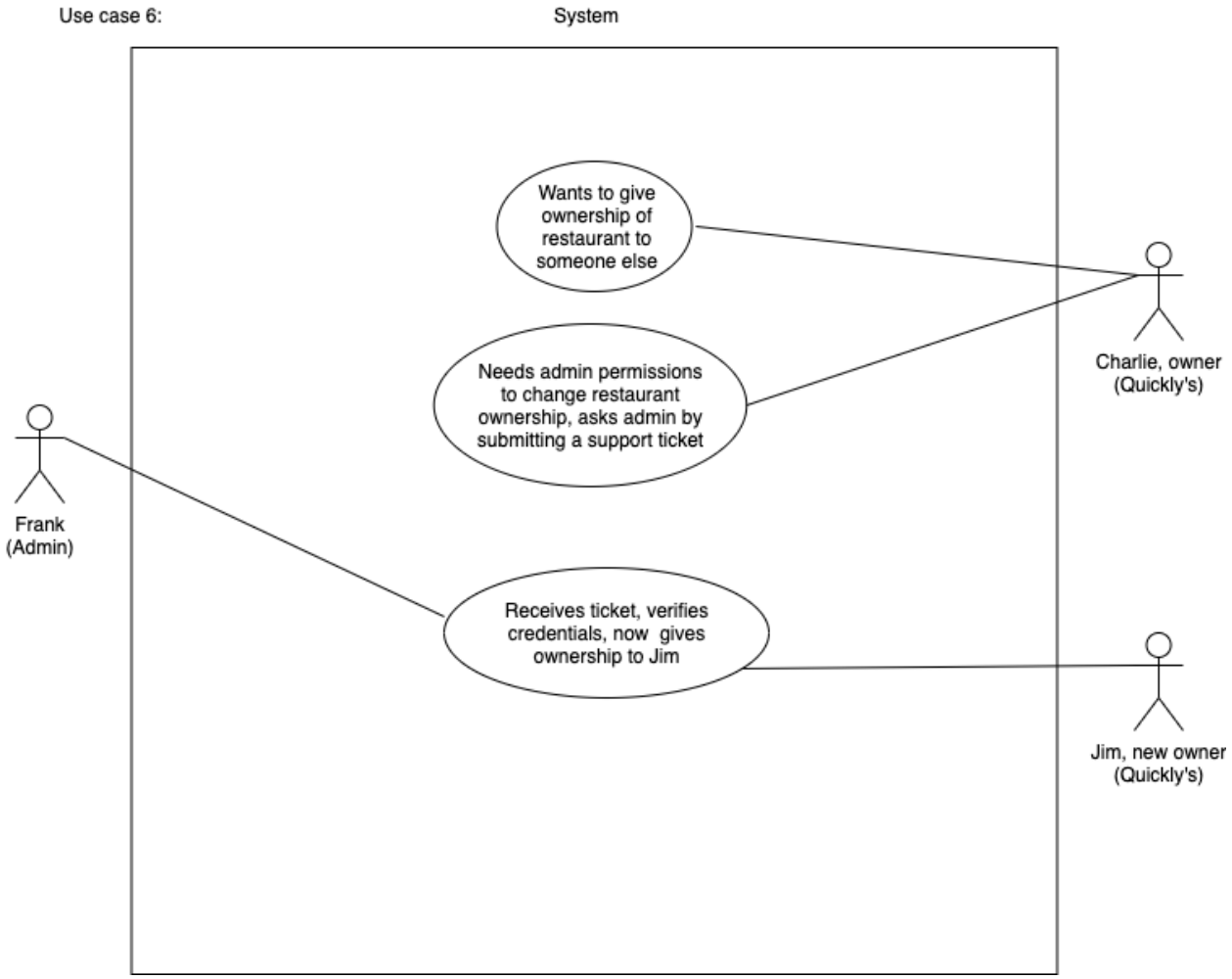
Use case 4:



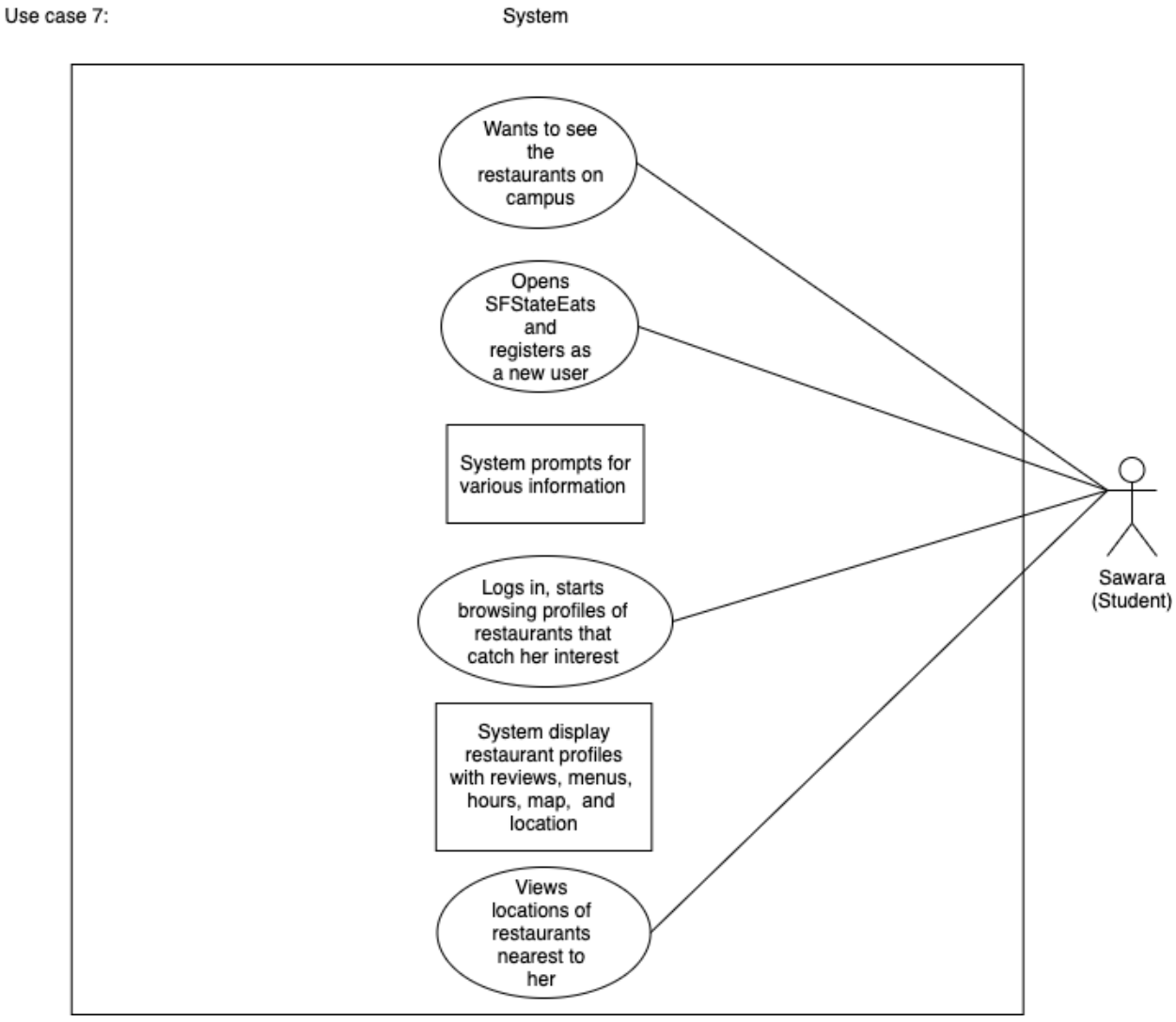
5. Elsa is a third year student attending San Francisco State University pursuing a degree in Sociology. Every semester she does great in her classes, and tries to offer as much support to her peers as possible. This semester, she has Tuesday and Thursday free, and leads a study session for Sociology 304 every Tuesday at 9:00 AM. In her rush to get out of the door in the morning, she forgot her wallet at home, and only has her iPhone with her. She needs a coffee to get her day started, but does not have any cash or credit cards with her. She opens SFStateEats, and pulls up all the vendors that have coffee. She then sorts it by payment method to filter out all the places that do not accept ApplePay. Luckly, one place did, so she decides to go to Cafe 101.



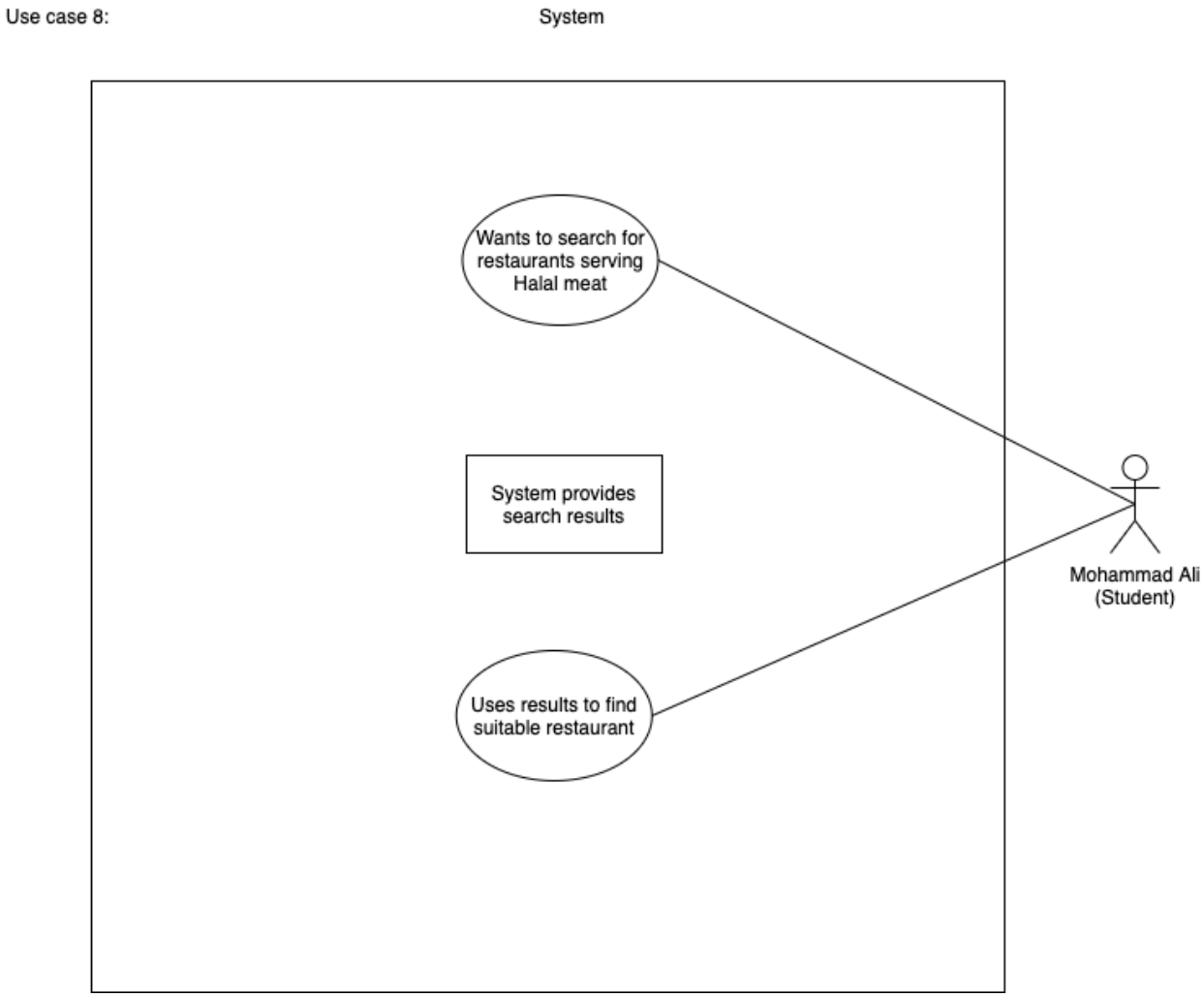
6. Frank is a senior at San Francisco State University, and has run many clubs in the past. This year he is the leader of ‘Google Codes’, a club for Computer Science majors sponsored by Google. Due to his experience working with people and his proven track record, the creators of SFStateEats offered him a position as an admin for the website, which Frank accepted. Frank has been an admin on the SFStateEats app for 3 weeks, and has been excelling so far. When users encounter a problem on the app, they can open a support ticket to receive help. Charlie, who is the owner of Quickly’s is selling his restaurant to move to another state. He needs to change the owner on the Quickly’s account to someone else, so he opens a support ticket to have an admin do it. Frank receives a support request, and after verifying all the proper paperwork, Frank manually changes the owner of Quickly’s to the new owner, Jim.



7. Sawara has been a student at San Francisco State University for 2 years. When out riding her bike, she fell and broke her leg, and now has to use crutches for the next 2 months. Normally when she has time between her classes, she walks around exploring food places on campus. Now that her leg is broken she has limited mobility, but still wants to explore and try out new food places. She learns about an app called SFStateEats that lists all the restaurants on campus, along with a host of useful information, and thinks this might be the perfect solution to her problem. She downloads the app and registers for a new account. After registering, she logs into the app and starts exploring the profile pages of all the restaurants that she is interested in. Then she starts seeing the locations on the map to see which is closest to her. Based on her search and interests, she choses to get an item at Cafe Russo.



8. Mohammad Ali is an international student at San Francisco State University. Back in his home country, United Arab Emirates, it is easy to find Halal food since many adhere to Islamic Law. After transferring to schools, he has struggled to find Halal on campus, and looks for a new way to find it. After doing some Googling, he comes across a website called SFStateEats, which is exactly what he was looking for. Mohammad logs into SFStateEats and uses halal meat in the search options. After skimming through the list of restaurants serving halal meat, he sees a place called Crave that serves sub sandwiches, and decides to try it out. After eating there, he was glad it was Halal, but thought the quality was not amazing. He leaves a 3 star rating along with his review explaining his thoughts.



Section 3: List of Main Data Items and Entities

Entity or Data Item	Description
System Administrator	Account for those users who are employed or trusted by the application owners. These accounts have access to moderator tools that no other account has access to. This may include the ability to remove reviews or change business information.
Unregistered User	A user who visits the website to use its features, but does not create an account.
Registered User	A user who uses the website and has created an account. This allows them to access features that an unregistered user may not have.
Registered Business Owner	A user who owns a business that is listed on SFStateEats. This account has access to features relating to business pages that other users do not.
Restaurant	An entity that contains all the information about a restaurant. This will be the page that has reviews, menus, hours, map, and location.
Review	A data item relating to restaurants. Reviews can be left by registered users and can be seen by any user. Reviews may include a section of text, photos, and ratings.
Menu	A data item relating to restaurants. A menu lists all of the items that the restaurant servers. This may include prices along with other information.
Map / Address	A data item relating to restaurants. A location will be information regarding where the restaurant can be found.

Section 4: Initial List of Functional Requirements

User

1. New users shall be able to create an account.
2. Users shall be able to login into their account.
3. Users shall be able to rate restaurants on campus.
4. Users shall be able to rate food places on campus.
5. Users shall be able to write and view reviews.
6. Users should be able to post pictures in their reviews.
7. Users shall be able to see the menu for each restaurant.
8. Users shall be able to see the price for each item on the menu.
9. Users shall be able to see the location of the restaurant on the profile page.
10. Users shall be able to see all the businesses that are involved in the farmers market. //P2
11. Users shall be able to see the map locations of each restaurant on campus. //P2
12. Users shall be able to add photos of each item on the website.
13. Users shall be able to check-in to restaurants. //P3
14. Users shall be able to see busy times for restaurants. //P3
15. Users shall be able to see hours of business for each restaurant.
16. Users shall be able to apply filters to their search.
17. Users shall be able to flag a review as inappropriate for the application.
18. Users shall be able to sort reviews by date.
19. Users shall be able to sort reviews by popularity.
20. Users shall be able to delete a review they have written.
21. Users shall be able to edit a review they have written. //P2
22. Users shall be able to see all reviews they have written.
23. Users shall be able to see all reviews another user has written.
24. Users shall be able to reply to other user's reviews. //P2
25. Users shall be able to share restaurant information and profiles to each other. //P2
26. Users will be able to connect their facebook and instagram to their accounts. //P3
27. Users will be able to login with their sfsu id that is linked to ilearn. //P3
28. Users will be able to see a visual representation of the ratings.
29. Users who are visually impaired will be able to use an automated system (such as siri) to navigate through the app. //P3

Business

30. New business owners shall be able to register a business account.
31. Business owners shall be able to login into their account.
32. Business owners shall be able to add their restaurant to the website.
33. Business owners shall be able to create a menu for their restaurant.

34. Business owners shall be able to upload photos of their restaurant.
35. Business owners shall be able to tag their restaurant into categories. // P2
36. Business owners shall be able to write descriptions of their restaurant on the profile page.
37. Business owners shall be able to upload an approved health score. // P2
38. Business owners shall be able to change prices on the menu.
39. Business owners shall be able to change hours of the restaurant.
40. Business owners shall be able to petition for a change of owner. //P2
41. Business owners shall be able to flag a review as inappropriate for the website.
42. Business owners shall be able to remove their restaurant from the website.
43. Business owners shall be able to see a chart of how their restaurant is doing based on the ratings. //P3

Restaurant

44. The restaurant profile page shall display photos.
45. The restaurant profile page shall display ratings.
46. The restaurant profile page shall display tags.
47. The restaurant profile page shall display a description.
48. The restaurant profile page shall display location.
49. The restaurant profile page shall display a menu.
50. The restaurant profile page shall display hours.
51. The restaurant profile page shall display reviews.
52. The restaurant profile page shall display the owner.
53. The restaurant profile page shall display phone number.
54. The restaurant profile page shall display the option to order online. //P3
55. The restaurant profile page shall display an option to pick up the order. //P3
56. The restaurant profile page will show any nutritional facts of their restaurant. //P2
57. The restaurant profile page shall show the customers if they serve halal, vegan, vegetarian, and non-vegetarian meals. // P2
58. The restaurant profile page shall display the various payment methods the restaurant accepts.
59. The restaurant profile page shall display the popularity/traffic of the restaurant. //P3

System Administrator

60. System Administrators shall have the privilege to ban any users from the website for misuse.
61. System Administrators shall have the privilege to delete restaurants from the platform.
62. System Administrators shall have the privilege to add restaurants to the platform.
63. System Administrators shall have the privilege to change restaurant information.
64. System Administrators shall have the privilege to verify new business accounts.
65. System Administrators shall have the privilege to delete reviews from the platform.

66. System Administrators shall be able to cancel orders placed at the customer's request.//P3
67. System Administrators shall be able to refund the customer if the order is not what they wanted. //P3

System

68. The system shall show trending restaurants on the main page. //P3
69. The system shall show recent reviews on the main page. //P2
70. The system shall show new restaurants on the main page .

Section 5: List of Non-Functional Requirements

Security

1. Login shall be required to leave reviews.
2. Login shall be required to create a business listing.
3. A gateway service will redirect traffic to other services to prevent the discovery of the end URL.
4. Email shall not be from a temporary email host provider.
5. Password shall contain at least 1 number, 1 special character, 1 uppercase letter, and 1 lowercase letter.
6. Users shall be required to change password every 12 months.
7. Change of password shall require email verification.
8. Users shall have 5 login attempts before account lockout.
9. Account unlock shall require email verification before unlocking.

Audit

10. New business listing shall be approved by the system administrator.
11. Change in business owner shall be approved by the system administrator.
12. Flagged reviews shall be reviewed by system administrator.
13. Inappropriate reviews shall be removed by the system administrator.

Performance

14. Each service should be hosted in its own server to prevent overwhelming one server.
15. The web application should restart if it is abruptly shut down to prevent downtime.
16. System shall respond visually within 5 seconds.
17. Animations shall be kept to a minimum to accommodate for slow performing devices.

Data Integrity

18. Images shall only be in the format of jpg, jpeg, and png.
19. Images shall be saved on the server.
20. Images uploaded shall be at most 2mb.
21. Images shall be saved in the original size.
22. Images shall be resized and displayed via CSS formatting, thus avoiding displaying and resizing after.
23. Reviews shall not consist of special characters or emojis.
24. Display name shall not consist of special characters or emojis.
25. Email shall not consist of emojis.
26. Password shall not consist of emojis.

27. Databases shall be backed up every 24 hours.
28. Databases shall be able to be backed up on command by a system administrator.

Compatibility

29. The site shall be compatible for all mobile screen sizes.
30. The site shall be compatible for all monitor screen sizes.
31. The site shall be able to scale to high resolution (2K, 4K).
32. The site shall be compatible for Safari on version 12.1.2.
33. The site shall be compatible for Firefox on version 73.0.
34. The site shall be compatible for Chrome on version 80.0.3987.106.

Conformance with Coding Standards

35. The whole production cycle of the site shall be finished in at least 5 days before the delivery date.
36. Development console logging shall be disabled for production.
37. The DOM tree shall have meaningful semantic for every element.
38. Components shall be created to allow for reusable code.

Look and Feel Standards

39. The site shall have an interface that is intuitive to navigate at first glance.
40. The site shall use clean fonts and colors.
41. The site shall meet modern design standards.

Internalization / Localization Requirements

42. The site shall be internationalized in the English language

Web Site Policies

43. The site shall not allow illegal content.
44. The site shall not allow harassment of other users.
45. The site shall not allow nudity.
46. The site shall not allow business owners to review their own site.
47. The site shall not allow self-promotion or malicious links.
48. Reviews and replies are subject to the website's community guidelines.

Section 6: Competitive Analysis

Feature/Company	Yelp	OpenTable	Google Reviews
Strengths	Sleek and intuitive UI	Unintrusive UI	More accurate restaurant information
Weaknesses	Restaurant information not always up to date	Commenting Section not fully fledged out	UI is not easy to navigate
Customer Support	Responsive customer support	Responsive customer support	Does not offer customer support
Featured Businesses	All businesses	Food businesses only	All businesses
Brand Recognition	Highly popular. Mildly accessible.	Less popular. Mildly accessible.	Mildly popular. Highly accessible
Onboard Experience	Fast and seamless browsing.	Small review base, but organized UI.	Large review base, but unorganized reviews.

Feature	Yelp	OpenTable	Google Reviews	SFStateEats
Reviews	+	+	+	++[1]
Reservation	+	+	-	-
Online Order	+	-	-	-
Campus Event Announcements	-	-	-	+[2]
Location and Hours	+	+	+	+

1: While other competitors also have reviews, none of them have threads within those reviews. SFStateEats shall have the ability to leave comments and start a thread on existing reviews. All reviews a restaurant has will show up in a list that users can browse. If a user thinks a review is good, bad, missing something, etc…, he is able to reply to this review. This will start a self-contained thread, that anyone can comment to add to. As more users comment, the thread gets bigger, but these chains of comments will never crowd the original reviews tab of the restaurant. This is a benefit because people can discuss the topics mentioned in a specific review, as opposed to just the restaurant as a whole. It allows for more user engagement, without cluttering the experience for other users.

2: Yelp, OpenTable, and Google Reviews all lack the ability to add food related events that are occuring on campus. One example of such an event is when a conference occurs, they often give out free food. SFStateEats shall allow organizers of these events to announce their events. This allows users to be on the lookout for food events simply by using the application. When a new event is added by a host, all users of SFStateEats who are subscribed and meet the criteria will be notified of this event. This is also great for hosts, because they are able to attract more people to their events.

Section 7: High Level System Architecture and Technologies Used

Server Host: Amazon EC2 server vCPUs 1 RAM 1GiB
Operating System: Ubuntu 18.04
Database: PostgreSQL v10.12
Web Server: EC2 t2.micro
Server-Side Language: JavaScript
Web Framework: ReactJS
IDE: IntelliJ, WebStorm, VS Code
HTML: 5
CSS: 3
React: 0.61
Node: 13.8.0
Supported browsers: Chrome(ver 80.0.3987.106), Safari(ver 12.1.2) and Firefox(ver 73.0) .

Section 8: Team

Name	Email	Role
Rachit Joshi	rjoshi@mail.sfsu.edu	Team Lead
Samhita Brigeda	pandu.barigeda@gmail.com	Front End Lead
Pedro Souto	Pedro.Souto.SFSU@gmail.com	Back End Lead
John Pham	JohnPhamDeveloper@hotmail.com	Github Master
Vincent Tran	vtran6@mail.sfsu.edu	Database Lead
Khang Tran	ktran26@mail.sfsu.edu	Frontend Developer

Section 9: Checklist

Task	Status
Team found a time slot to meet outside of the class	DONE
Github master chosen	DONE
Team decided and agreed together on using the listed SW tools and deployment server	DONE
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	ON TRACK
Team lead ensured that all team members read the final M1 and agree/ understand it before submission	DONE
Github organized as discussed in class (e.g. master branch, development branch, folder for milestone documents etc.)	DONE

Milestone Two

SW Engineering CSC648/848 Spring 2019
Team 04
04/09/2020

SFStateEats

Team Members

Name	Email	Role
Rachit Joshi	rjoshi@mail.sfsu.edu	Team Lead
Samhita Brigeda	pandu.barigeda@gmail.com	Front End Lead
Pedro Souto	Pedro.Souto.SFSU@gmail.com	Back End Lead
John Pham	JohnPhamDeveloper@hotmail.com	Github Master
Vincent Tran	vtran6@mail.sfsu.edu	Database Lead
Khang Tran	ktran26@mail.sfsu.edu	Frontend Developer

Milestone 2 Version History

Milestone Version	Date
Milestone 2 Version 2	04/09/2020
Milestone 2 Version 1	03/22/2020
Milestone 1 Version 2	03/10/2020
Milestone 1 Version 1	02/24/2020

Table of Contents

Section 1: Data Definitions V2

System Administrator

Description

Account for those users who are employed or trusted by the application owners. These accounts have access to moderator tools that no other account has access to. This may include the ability to remove reviews or change business information.

Restrictions

- Due to the fact that these accounts are not public, system administrators do not have any data items related to public profiles. This includes: profile picture, description, date of birth, reviews, ratings, etc.

Sub-Data Items

- Email
- Username
- Password

Registered User

Description

A user who uses the website and has created an account. This allows them to access features that an unregistered user may not have.

Restrictions

- Registered users shall not be younger than 13 years old.
- Registered users shall use an email that is not already in use in the application.

Sub-Data Items

- Email
- Username
- Password
- Profile Picture (Photo)
- Date of Birth

Registered Business Owner

Description

A user who owns a business that is listed on SFStateEats. This account has access to features relating to business pages that other users do not.

Restrictions

- Registered business owners shall be older than 13 years old.
- Registered business owners shall use an email that is not already in use in the application.
- Registered business owner’s phone number shall not be public.
- Registered business owner’s addresses shall not be public.

Sub-Data Items

- Email
- Username
- Password
- Phone Number
- Address

Restaurant

Description

An entity that contains all the information about a restaurant. This will be the page that has reviews, menus, hours, map, and location

Restrictions

- Restaurant shall not be listed unless it has been verified by a system administrator.
- Restaurants shall not upload more than 15 photos.
- Restaurant shall be located on campus.

Sub-Data Items

- Description
- Photo
- Review
- Menu
- Hours
- Address

Review

Description

A data item relating to restaurants. Reviews can be left by registered users and can be seen by any user. Reviews may include a section of text, photos, and ratings. A review consists of 3 parts, and may also include photos if the user wishes.

Restrictions

- A review shall have at most 3 photos uploaded.
- A review shall not require all sub data items to be filled out, as long as one of them is. For instance, if a user wants to only leave a text review, reaction/rating/photos may be left blank to achieve that.

Sub-Data Items

- Review Text
- Rating
- Photo

Menu

Description

A data item relating to restaurants. Restaurants may have a menu. A menu lists all of the items that the restaurant offers.

Restrictions

- A restaurant shall have at most 1 menu.

Sub-Data Items

- Menu Item

Menu Item

Description

A data item relating to a menu. A menu can have multiple menu items. Menu items are the individual items that a restaurant offers.

Restrictions

- A menu item shall have at most 1 photo.
- A menu item shall have an item name.
- A menu item shall have a price.

Sub-Data Items

- Description
- Price
- Photo
- Item Name

Item Name

Description

A data item relating to menu item. Every item on a menu must have a name. This is the text data for that name.

Restrictions

- An item name shall have at least 1 character.
- An item name shall have at most 50 characters.
- An item name shall not contain an emojis.

Sub-Data Items

- None

Price

Description

A data item relating to menu item. Every item on a menu must have a price. This is the numerical data for price.

Restrictions

- A price shall be denoted in USD.
- A price shall not be null.

Sub-Data Items

- None

Address

Description

A data item relating to restaurants. An address will be written information regarding where the restaurant can be found.

Restrictions

- An address shall only consist of text.
- An address shall have at most 100 characters.
- An address shall not have any emojis.

Sub-Data Items

- None

Email

Description

An email will be a stored piece of information that allows for contact, verification, and login for the user's of the application.

Restrictions

- An email shall contain only characters.
- An email shall not be longer than 50 characters.
- An email shall have an @ symbol, and end with a valid extension.
- An email shall not contain any emojis.

Sub-Data Items

- None

Username

Description

A username will be a stored piece of information that allows for distinction and login for the user's of the application.

Restrictions

- A username shall contain only characters.
- A username shall not be longer than 25 characters.
- A username shall not contain any emojis.
- A username shall be unique.

Sub-Data Items

- None

Password

Description

A password will be a stored piece of information that allows for account login validation for the user's of the application.

Restrictions

- A password shall contain only characters.
- A password shall not be longer than 250 characters.
- A password shall not be shorter than 8 characters.
- A password shall not contain any emojis.
- A password shall contain at least 1 number, 1 uppercase letter, 1 special character, and 1 lowercase letter.
- A password shall be encrypted when stored.

Sub-Data Items

- None

Photo

Description

A photo will be a stored piece of media. Photos have many uses such as profile pictures, restaurant photos, and review photos.

Restrictions

- A photo shall not have a file size larger than 2MB.
- A photo shall not have dimensions larger than 1920 x 1080 pixels.
- A photo must be in one of the following formats: .JPG, .JPEG, .PNG, .HEIC.

Sub-Data Items

- None

Date of Birth

Description

A date of birth will be a text piece of data. It shall be used for verification of age when creating an account.

Restrictions

- A date of birth shall follow the MM-DD-YYYY format.
- A date of birth shall have exactly 8 positive numeric characters.
- A date of birth shall have a year greater than 1900.
- A date of birth shall have a day greater than 0 and less than 32.
- A date of birth shall have a month greater than 0 and less than 13.

Sub-Data Items

- None

Description

Description

A description will be a text piece of data. It shall be used for describing various entities on the application.

Restrictions

- A description shall contain no emojis.
- A description shall be no longer than 500 characters.

Sub-Data Items

- None

Phone Number

Description

A phone number will be a text piece of data. It shall be used to have contact information for various users of the application.

Restrictions

- A phone number shall follow the (###)###-#### format.
- A phone number shall have exactly 10 positive numeric characters.
- A phone number shall not have any negative numbers.

Sub-Data Items

- None

Hour

Description

An hour will be a text piece of data. It shall be used by restaurants to display the operating hours in which they are open to the public.

Restrictions

- An hour shall follow the *“DAY”* HH:MM *“AM/PM”* format.
- The *“DAY” string shall exactly match one of the seven days of the week.*
- The *“AM/PM”* string shall exactly match either “AM” or “PM”.
- The HH:MM shall contain only exactly 4 positive numeric values.

Sub-Data Items

- None

Review Text

Description

A review text shall be a text piece of data. It shall be used when a registered user leaves a review for a restaurant.

Restrictions

- A review text shall contain no emojis.
- A review text shall be no longer than 2500 characters.

Sub-Data Items

- None

Rating

Description

A rating is a score created when a registered user rates his experience. This rating is from 0-5, 0 being the worst, and 5 being the best. This rating can then be aggregated and displayed visually.

Restrictions

- A rating shall be a number between 0 and 5, inclusive.

Sub-Data Items

- None

Section 2: Functional Requirements V2

Priority One

User

1. New users shall be able to create an account.
2. Users shall be able to login into their account.
    2.1. A user shall only be logged in to one session at any given time.
3. Users shall be able to rate restaurants on campus.
    3.1. A user shall only rate a restaurant once.
    3.2. A user shall only rate a restaurant if they are reviewing it.
4. Users shall be able to rate an event on campus.
    4.1. A user shall only rate and event once.
    4.2. A user shall only rate an event if they are reviewing it.
5. Users shall be able to write reviews.
    5.1. A user shall be able to write a review only once per restaurant.
    5.2. A user shall be logged in to write a review.
6. Users shall be able to view reviews.
7. Users should be able to post pictures in their reviews.
    7.1. A user shall upload a maximum of 3 images per review.
8. Users shall be able to see the menu for each restaurant.
9. Users shall be able to see the price for each item on the menu.
10. Users shall be able to see the address of the restaurant on the profile page.
    10.1. An address shall be a text description of where the restaurant is located on campus.
15. Users shall be able to see hours of business for each restaurant.
    15.1. Hours shall be updated for holidays and special occasions.
    15.2. Hours shall be formatted using the HH:MM format.

16. Users shall be able to apply filters to their search.
    16.1. A user shall be able to choose from one or many of the following filters:
        16.1.1. Recommendation
        16.1.2. Rating
        16.1.3. Reviews
        16.1.4. Price
        16.1.5. Hours
        16.1.6. Address / Distance
        16.1.7. Categories
        16.1.8. Payment Methods

17. Users shall be able to flag a review as inappropriate for the application.
    17.1. A user can only flag 10 reviews per day.
    17.2. The amount of flags a review has shall not be public.
18. Users shall be able to sort reviews by date.
    18.1. Users shall be able to sort by newest, oldest or specific date.
19. Users shall be able to sort reviews by popularity.
    19.1. A review's popularity is based on the number of positive reactions a review has.
    19.2. A user shall be able to sort popularity by most popular or least popular.
20. Users shall be able to delete a review they have written.
    20.1. A review can only be deleted within a specified (24 hours) time frame.
22. Users shall be able to see all reviews they have written.
23. Users shall be able to see all reviews another user has written.
28. Users will be able to see a visual representation of the ratings.
    28.1. The visual representation shall use a star format.
    28.2. A rating shall be on a scale from 0-5, in half increments.

Business

30. New business owners shall be able to register a business account.
    30.1. A business shall be verified by a system administrator.
    30.2. Business account verification shall take up to 3 business days.
    30.3. Business account verification shall require additional details (proof of ownership, etc).
31. Business owners shall be able to login into their account.

31.1. A business account shall only be logged in to one session at any given time.

32. Business owners shall be able to add their restaurant to the website.
    32.1. A new restaurant shall be verified by a system administrator.
    32.2. New restaurant verification shall take up to 3 business days.
    32.3. New restaurant verification shall require additional details (proof of ownership, etc).
33. Business owners shall be able to create a menu for their restaurant.
    33.1. A menu shall have the option for subsections.
    33.2. All menu items shall be required to have a price
    33.3. A menu item shall have the option to have categories.
34. Business owners shall be able to upload photos of their restaurant.
    34.1. A restaurant shall have at most 15 pictures uploaded.
36. Business owners shall be able to write descriptions of their restaurant on the profile page.
    36.1. Restaurant descriptions shall be at most 500 characters.
38. Business owners shall be able to change prices on the menu.
39. Business owners shall be able to change hours of the restaurant.
41. Business owners shall be able to flag a review as inappropriate for the website.
    41.1. A review flagged by a business shall have more weight than flags by users.
    41.2. A business can only flag 25 reviews per day.
    41.3. The amount of flags a review has shall not be public.
42. Business owners shall be able to request removal of their restaurant from the website.
    42.1. A request for deletion shall be required to have an explanation.
    42.2. A request for deletion shall be verified by the system administrator.

71. Business owners shall be able to upload photos of each item on the menu.
    71.1. A menu item shall only have 1 picture uploaded

Restaurant

44. The restaurant profile page shall display photos.
    44.1. Photos shall not be inappropriate.

45. The restaurant profile page shall display ratings.
46. The restaurant profile page shall display tags.
    46.1. Restaurant shall display at most 10 tags.
47. The restaurant profile page shall display a description.
    47.1. Description shall not contain any emojis or inappropriate language.
48. The restaurant profile page shall display address.
49. The restaurant profile page shall display a menu.
50. The restaurant profile page shall display hours.
51. The restaurant profile page shall display reviews.
52. The restaurant profile page shall display the owner.
    52.1. Restaurant owners shall decide if the restaurant displays the owner's account.
53. The restaurant profile page shall display phone numbers.
    53.1. A restaurant shall have at most 2 phone numbers.
    53.2. A business owner shall not have the same phone number as a restaurant phone number.
58. The restaurant profile page shall display the various payment methods the restaurant accepts.
    58.1. Payment methods shall be updated as systems change.

System Admin

60. System Administrators shall have the privilege to ban any users from the website for misuse.
    60.1. A ban shall be required to have a reason logged.
61. System Administrators shall have the privilege to delete restaurants from the platform.
    61.1. A deletion shall be required to have a reason logged if the deletion was in result of a ban.
    61.2. A deletion may also happen as a result of a restaurant removal request ticket.
62. System Administrators shall have the privilege to add restaurants to the platform.
    62.1. All restaurants added to the platform shall have relevant documents for proof of ownership.
63. System Administrators shall have the privilege to change restaurant information.
64. System Administrators shall have the privilege to verify new business accounts.
    64.1.New business accounts shall undergo review.
65. System Administrators shall have the privilege to delete reviews from the platform.

65.1. A review that receives too many flags shall open a ticket for further investigation.
65.2. A review that is removed shall be required to have a reason logged.

System

70. The system shall show new restaurants on the main page.
70.1. New restaurants shall be determined based on when they were approved on the platform.

Priority Two

User

10. Users shall be able to see all the businesses that are involved in the farmers market.
10.1. Businesses involved in the farmer's market shall have a special restaurant tag.
11. Users shall be able to see the map locations of each restaurant on campus.
11.1. Map location shall be based on Google API.
11.2. Map shall be able to pan and zoom.
11.3. Map shall provide GPS like directions to restaurants.
21. Users shall be able to edit a review they have written.
21.1. A review shall only be able to be edited within a certain time frame (2 hours).
21.2. A review shall display to all users that it has been edited.
21.3. A review shall only be edited at most 3 times.
24. Users shall be able to reply to other user's reviews.
24.1. A reply shall have the option to be a reaction.
24.2. A reply shall have the option to be text.
24.3. A reply shall have the option to be rating.
24.4. A text reply shall have a maximum of 500 characters.
25. Users shall be able to share restaurant information and profiles to each other.
25.1. Share shall be done over via a direct link to the restaurant.

Business

35. Business owners shall be able to tag their restaurant into categories.
35.1. A tag shall be chosen from a large list of predefined tags.
35.2. A restaurant shall have at most 15 tags.
37. Business owners shall be able to upload an approved health score.

37.1. A health score shall be uploaded as an image for verification.

37.2. Health score image shall contain all information required for proof validity.

40. Business owners shall be able to petition for a change of owner.

40.1. Petitions for change of owner shall be required to have a reason.

40.2. New business owner must have either a verified business account or a new business account under review.

Restaurant

56. The restaurant profile page will show any nutritional facts of the menu items.

57. The restaurant profile page shall show the customers if they serve halal, vegan, vegetarian, and non-vegetarian meals.

57.1. These options shall be categorized as tags and appear in filters.

System

69. The system shall show recent reviews on the main page.

Priority Three

User

13. Users shall be able to check-in to restaurants.

13.1. Check-in shall not use GPS for verification.

14. Users shall be able to see busy times for restaurants.

14.1. Busy times shall be calculated based on check-in's.

26. Users will be able to connect their facebook and instagram to their accounts.

26.1. Connections allow for users to share links directly to their social media feeds.

27. Users will be able to login with their sfsu id that is linked to ilearn.

27.1. User ID and password shall be identical to the university login.

29. Users who are visually impaired will be able to use an automated system (such as voice commands) to navigate through the app.

29.1. Websites shall use standard disability formats for this .

Business

43. Business owners shall be able to see a chart of how their restaurant is doing.
    - 43.1. Charts shall display information such as check-in's, positive reviews versus negative reviews, etc.

Restaurant

54. The restaurant profile page shall display the option to order online.
55. The restaurant profile page shall display an option to pick up the order.

System Admin

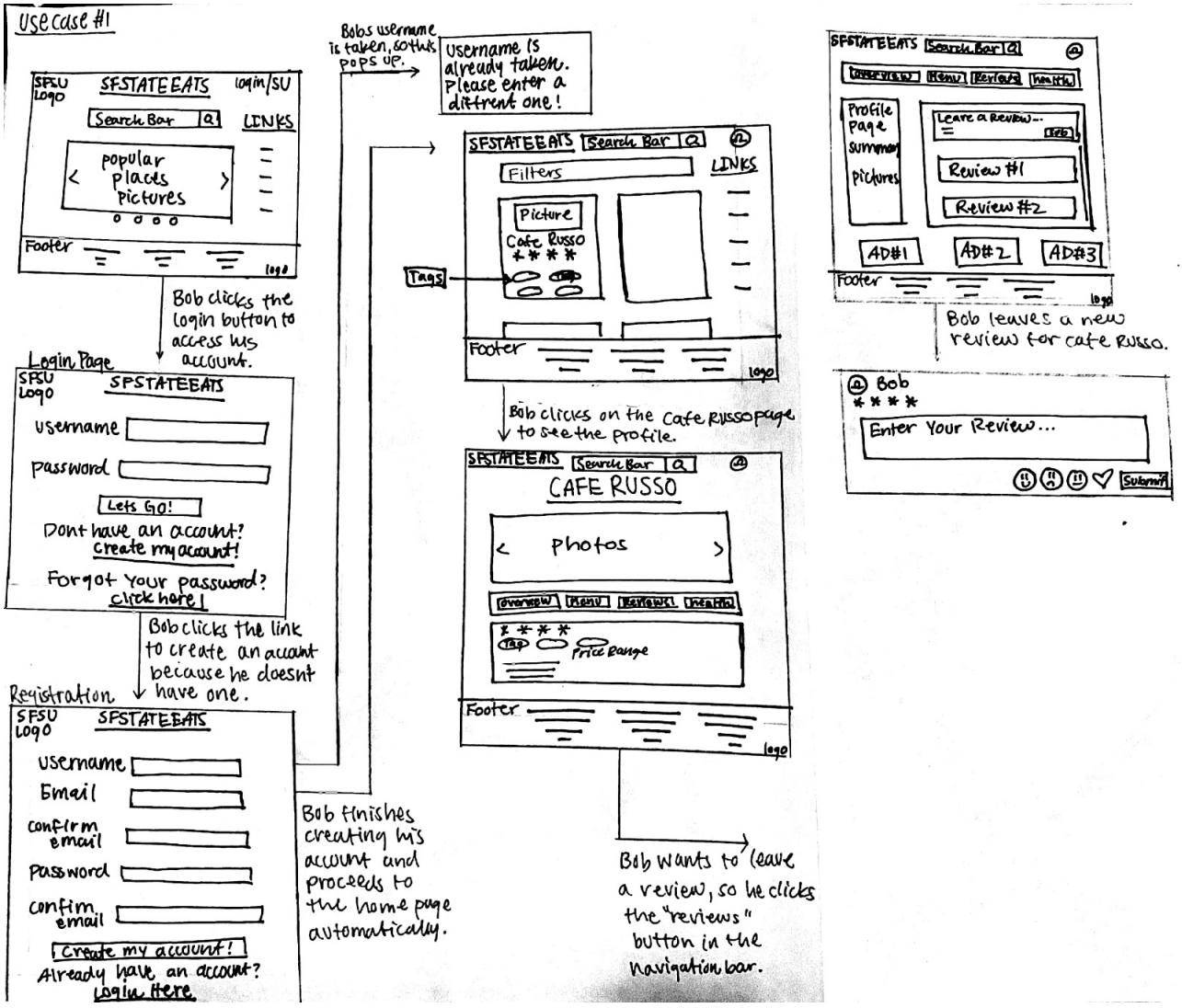
66. System Administrators shall be able to cancel orders placed at the customer's request.
    - 66.1. Customer cancellation requests shall be required to have a reason for cancelation.
67. System Administrators shall be able to refund the customer if the order is not what they wanted.

System

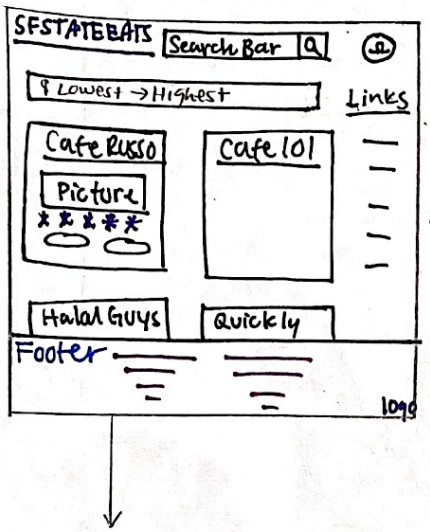
68. The system shall show trending restaurants on the main page.
    - 68.1. Trending restaurants are based on check-in's and new positive reviews.
    - 68.2. Trending restaurants are updated every 3 days.
    - 68.3. Display new reviews on the page.

Section 3: UI Mockups and Storyboards

WireFrame for Use Case 1

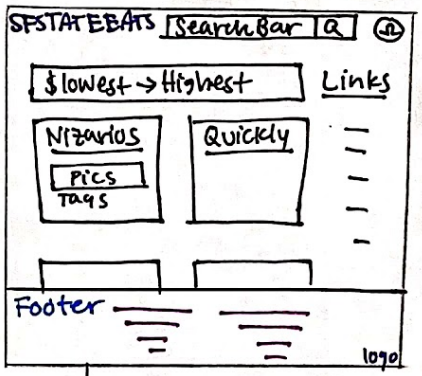


WireFrame for Use Case 2



Alice opens the SFSTATEEATS application and it opens to the home page, because she is already logged in.

Alice choses the filter that sorts restaurants from cheapest to expensive by price.



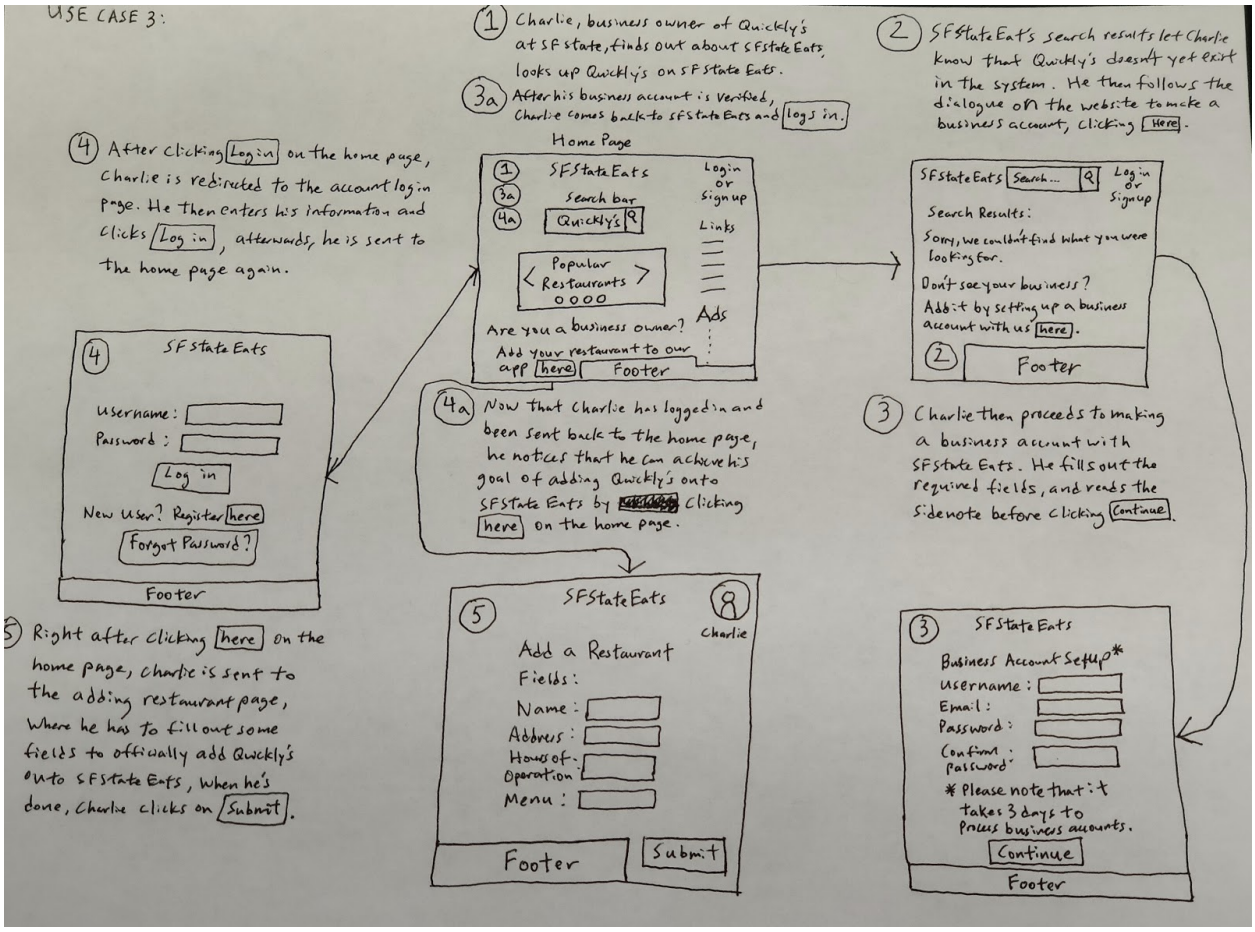
After sorting the restaurants, the web page shows the restaurants with cheapest prices.

Alice sees the restaurants and chooses the Nizario's Pizza restaurant.

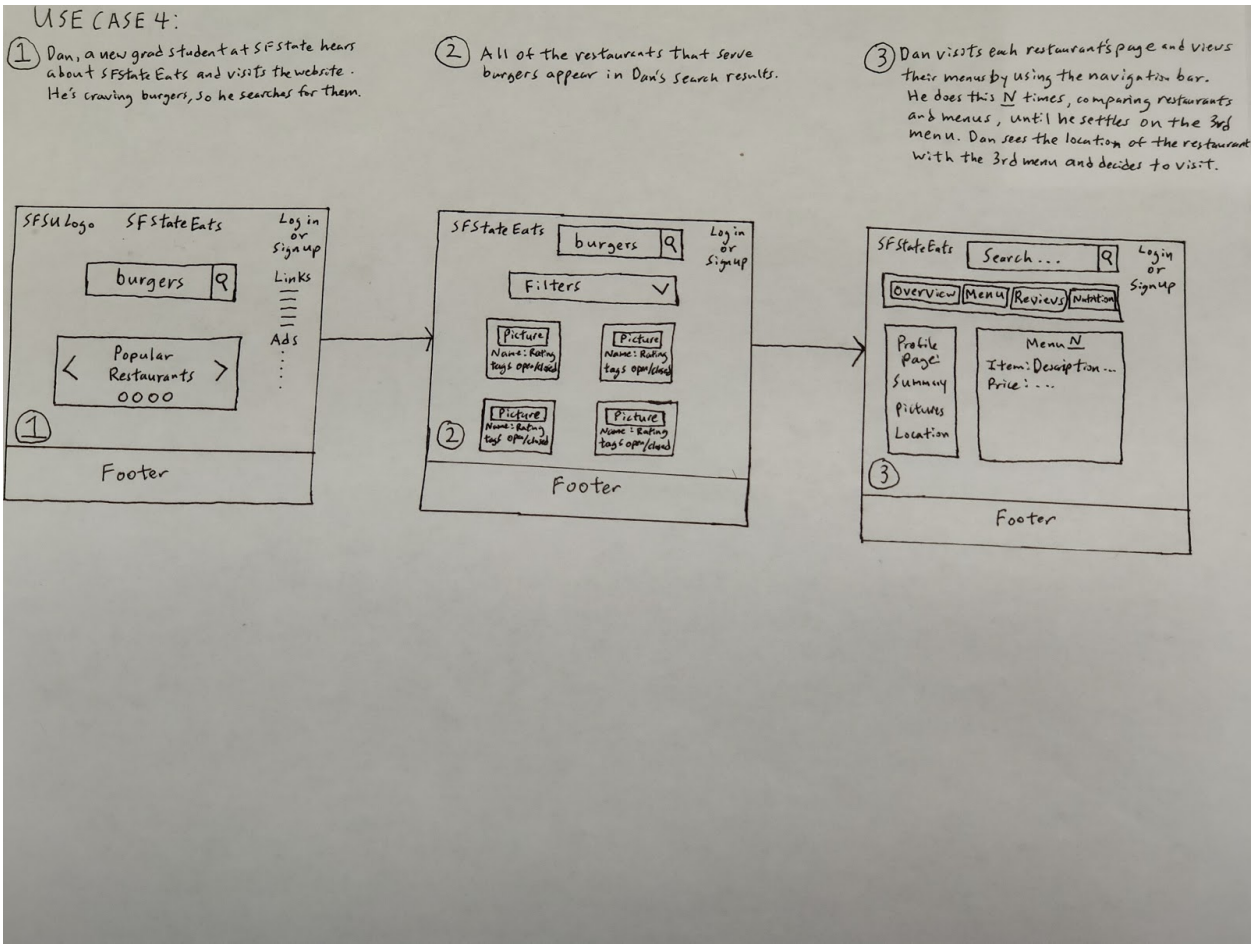
Alice also sees the location of the restaurant, ceasar chavez student center.

Alice also sees the ratings of the restaurant.

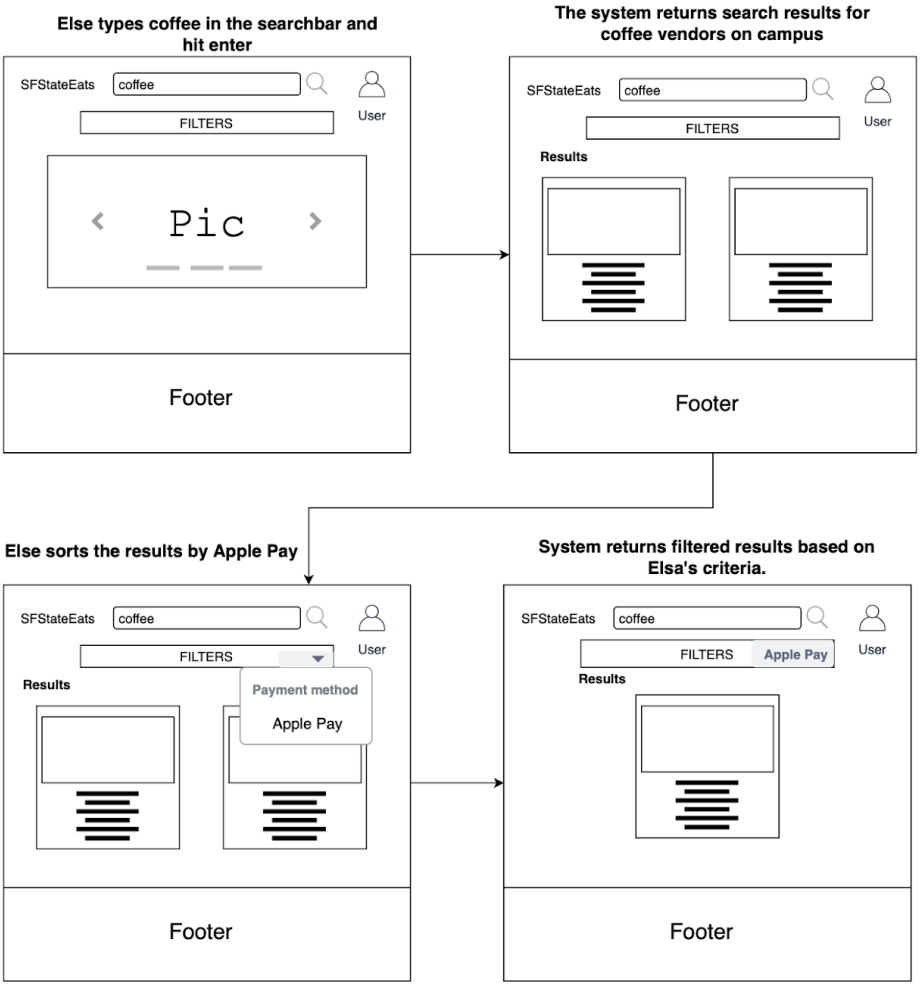
WireFrame for Use Case 3



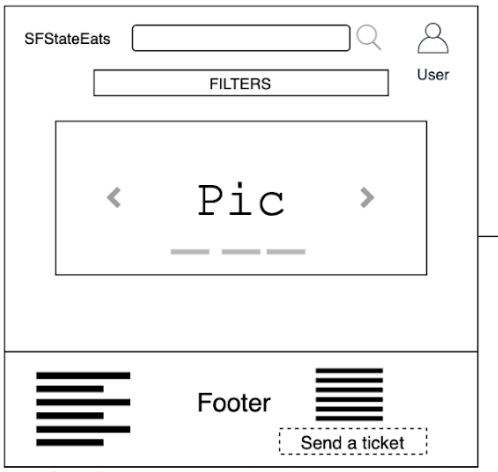
WireFrame for Use Case 4



WireFrame for Use Case 5

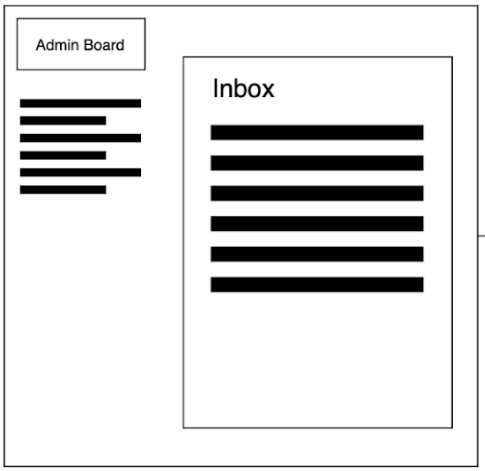


WireFrame for Use Case 6

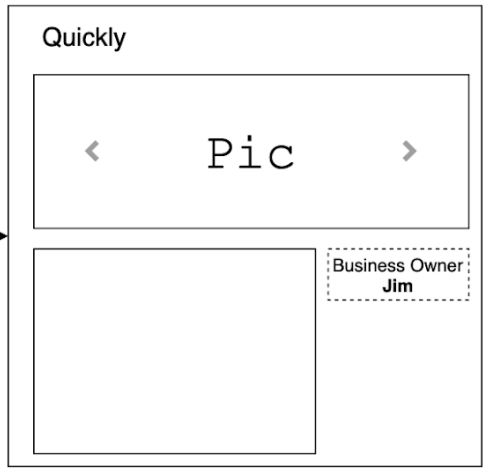


Charlie wants to send a ticket to sysadmin to update Quickly's owner to the new owner, Jim.

Charlie enters the details and upload necessary documents

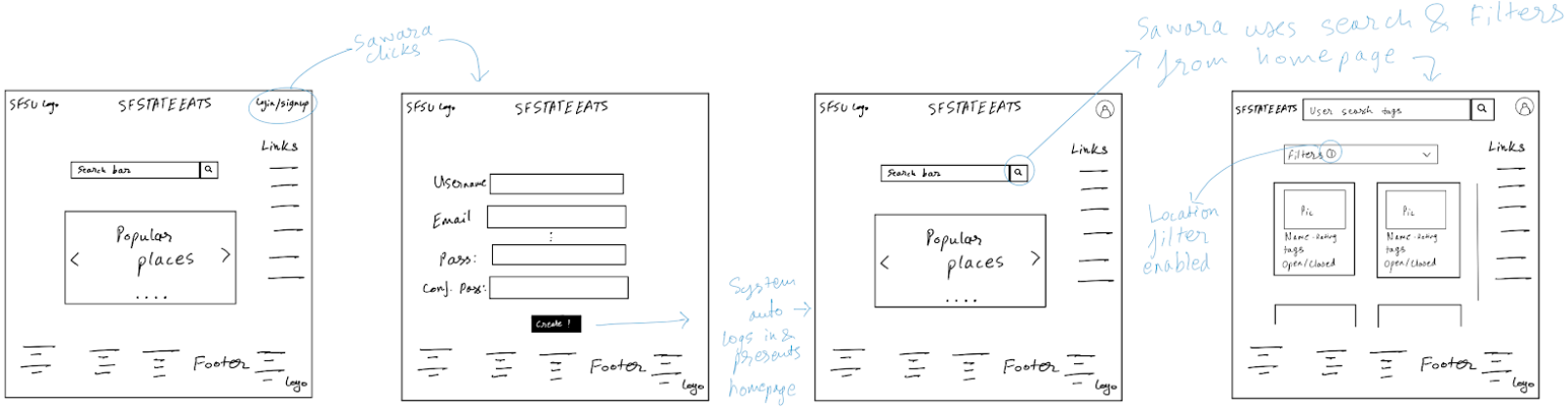


The sysadmin sees Charlie's message in the admin dashboad



The sysadmin updates Quickly page to reflect new updated info

WireFrame for Use Case 7



WireFrame for Use Case 8

Section 4: High Level Database Architecture and Organization

Description

- PostgreSQL will be used as the DBMS due to its ability to have table inheritance
- Search terms will be organized in alphanumerical order. Key terms to search are the restaurants and accounts.
- Images stored as URLs

Business Rules

1. Each user can create one business account
2. Each user can create one registered user account
3. Each user can search for many restaurants
4. Each business account can create a single restaurant
5. Each user can use with one payment type
6. Each admin can transfer many restaurant
7. Each registered user can submit many support tickets
8. Each registered user can review many restaurants
9. Each restaurant can create a single menu
10. Each restaurant can accepts many payment types
11. Each admin can delete many restaurants
12. Each menu can have many food

Entities

User

- userId: PK
- name: composite (first name, last name)
- dob: multivalue (month, day , year)
- age: derived
- photo

Account

- accountId: PK
- username
- password

Admin

- adminId: PK

Restaurant

- restaurantId: PK
- healthScore
- name
- description
- photo
- Popularity
- openHours
- tags
- location
- expires

Business account

- businessAccountId: PK

Registered user

- registeredUserId: PK

Review

- reviewId: PK
- title
- description
- stars

Payment

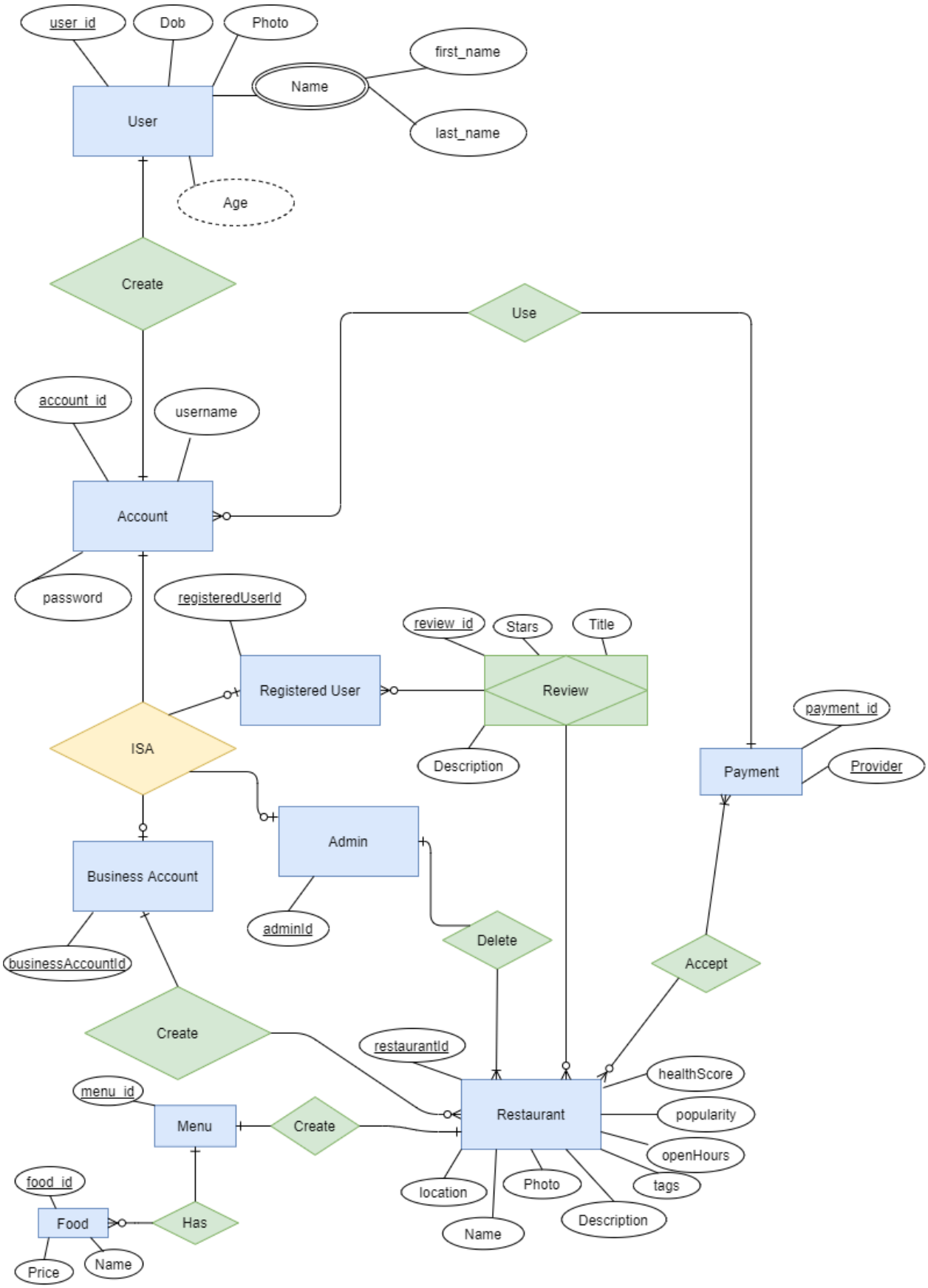
- paymentId: PK
- provider

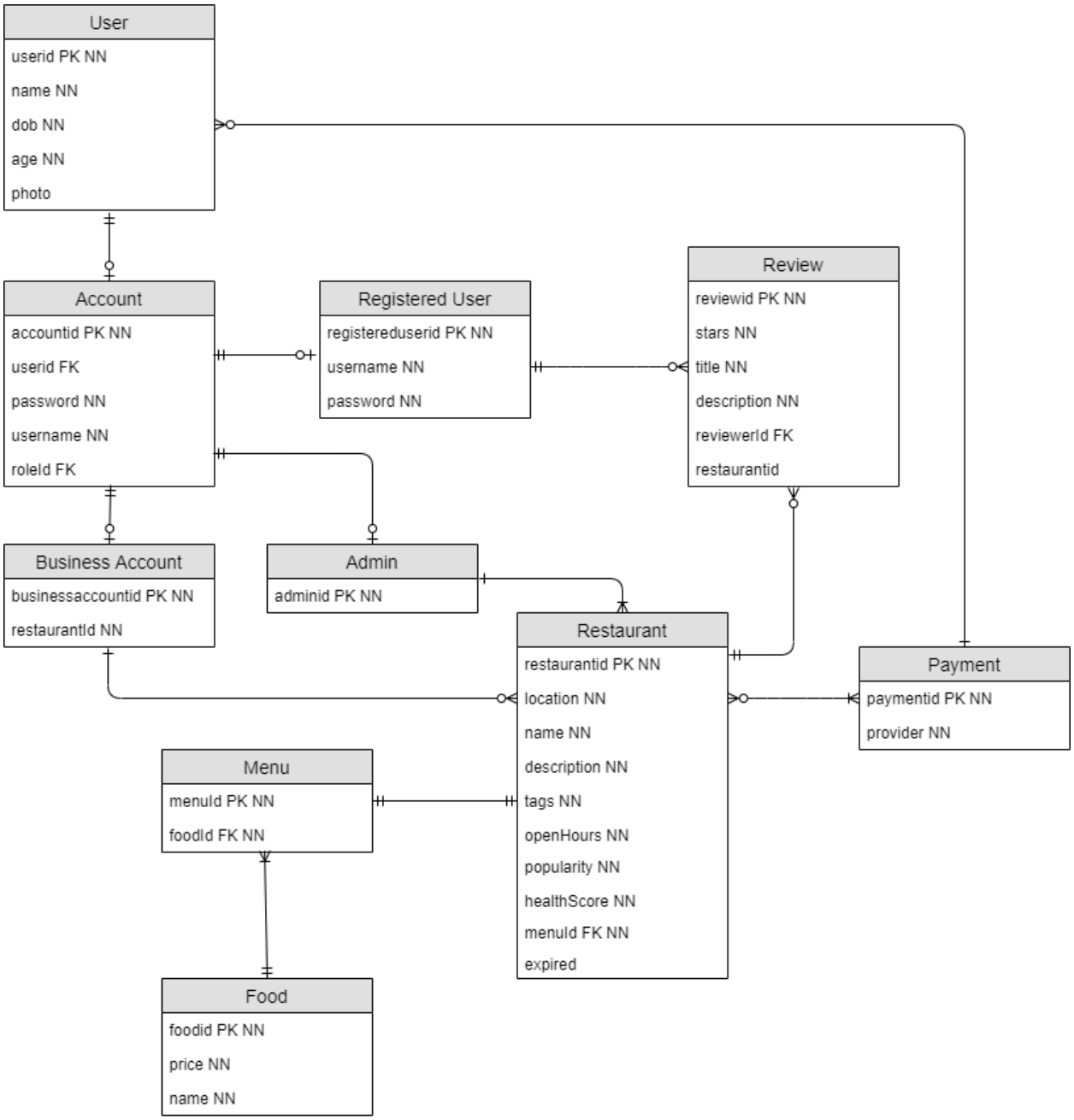
Ticket

- ticketId: PK
- record: DATETIME

Menu

- menuId: PK
- foodId: PK


user_id
Dob
Photo
first_name
Name
last_name
User
Age
Create
Use
account_id
username
Account
password
registeredUserId
review_id
Stars
Title
Registered User
Review
Description
payment_id
Provider
Payment
ISA
Admin
Business Account
adminId
Delete
Accept
businessAccountId
Create
restaurantId
menu_id
Menu
Create
Restaurant
healthScore
popularity
openHours
tags
location
Photo
Description
Name
food_id
Food
Has
Price
Name


User
userid PK NN
name NN
dob NN
age NN
photo
Account
accountid PK NN
userid FK
password NN
username NN
roleId FK
Registered User
registereduserid PK NN
username NN
password NN
Review
reviewid PK NN
stars NN
title NN
description NN
reviewerId FK
restaurantid
Business Account
businessaccountid PK NN
restaurantId NN
Admin
adminid PK NN
Restaurant
restaurantid PK NN
location NN
name NN
description NN
tags NN
openHours NN
popularity NN
healthScore NN
menuId FK NN
expired
Payment
paymentid PK NN
provider NN
Menu
menuId PK NN
foodId FK NN
Food
foodid PK NN
price NN
name NN

Media Storage

- Images will be kept in the file system on the server.
    - Images are used in profile, restaurant descriptions, and reviews.
    - A photo shall not have a file size larger than 2MB.
    - A photo shall not have dimensions larger than 1920 x 1080 pixels.
    - A photo must be in one of the following formats: .JPG, .JPEG, .PNG, .HEIC.

Search Filter Architecture

- Searching restaurant names with exact match
    - /api/restaurant?name=Mc+Donald&contains=0
    - This query is used when a user wants to find a restaurant by an exact name. This will only return the restaurant if the user enters the exact name. Any misspelling will cause no restaurants to be returned.
    - The database term being searched in this query shall be restaurant name only.
    - The SQL query for this query will be a basic select query, where we are only selecting restaurants with an exact match.

- Searching restaurant names with word inside
    - /api/restaurant?name=d&contains=1
    - This query is used when a user wants to find a restaurant where the name contains a specific string. This is used as the more traditional search user's expect, where they might not know the exact name of the restaurant.
    - The database term being searched in this query shall be restaurant name only.
    - The SQL query for this query will be a select query, where we use %like to find restaurants where the name is simply contained (versus an exact match).

- Gets restaurants that have less than 50 popularity (not inclusive) ascending order
    - /api/restaurant?filterBy=popularity&filterValue=50&filterCondition=lt&orderBy=asc
    - This filter is used to sort restaurants by popularity, where the user can choose between ascending and descending values.
    - The database terms being searched in this query are: restaurant name and number of reviews.
    - The SQL query has multiple conditions. The first condition will be a simple select condition. The second condition will be 'WHERE' condition which we use to filter restaurants with popularity.

Section 5: High Level API’s and Main Algorithms

API’s

Login

A basic API used for user authentication. This API will require both the user name and password to be sent in a JSON format. The password shall only be sent encrypted.

The server shall compare the information received with the information stored on the database. If all information received is valid and matches the stored information, the API endpoint shall return a successful login response, along with a token for that specific user. If the information does not match the database or is not valid, the API shall return an unsuccessful response along with a reason. Response shall be in the JSON format.

Register Regular User Account

A basic API used for regular user account registration. This shall be a different API endpoint than Register Business Account. This API shall receive information in a JSON format. The JSON shall contain all information required to register for a regular user account. This information includes: email, username, password, first name, last name, and date of birth.

The server shall do checks to make sure this information is valid, such as: email validation, date of birth validation, and password validation. It shall also ensure that the username is not taken. Only after doing all of these checks shall the server create an account for the user, after which, the API endpoint shall return a response stating if the registration was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Register Business Account

A basic API used for business account registration. This shall be a different API endpoint than Register Regular User Account. This API shall receive information in a JSON format. The JSON shall contain all information required to register for a business account. This information includes: email, username, password, first name, last name, date of birth, name of business, and proof of ownership.

The server shall do checks to make sure this information is valid, such as: email validation, date of birth validation, and password validation. It shall also ensure that the username is not taken. If all information is valid, this business account shall be added to a list of Business Accounts that must be verified manually by a system administrator and the API endpoint shall return a response stating that the registration was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Post Review

API used to post a review for a restaurant. This API shall receive information in a JSON format. The JSON shall contain all information required to post a review. This information includes: user token, restaurant ID, and review text.

The server shall do checks to make sure this information is valid, such as: user token validation, restaurant ID validation, and review text validation. These validations will ensure that the user is real and logged in, that the restaurant exists, and that the review text does not contain invalid characters. If all information is valid, the review shall be added to the database and the API endpoint shall return a response stating that the post was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Remove Review

API used to remove a review from a restaurant. This API shall receive information in a JSON format. The JSON shall contain all information required to remove a review. This information includes: user token, review ID.

The server shall do checks to make sure this information is valid, such as: user token validation and review ID validation. These validations will ensure that the user is real and logged in and that the review exists. If all information is valid, the review shall be removed from the database and the API endpoint shall return a response stating that the review removal was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Flag Review

API used to flag a review for a restaurant. This API shall receive information in a JSON format. The JSON shall contain all information required to post a review. This information includes: user token and review ID.

The server shall do checks to make sure this information is valid, such as: user token validation and review ID validation. These validations will ensure that the user is real and logged in and that the review exists. If all information is valid, the review shall be flagged and the API endpoint shall return a response stating that the flag was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Post Rating

API used to leave a rating for a restaurant. This API shall receive information in a JSON format. The JSON shall contain all information required to leave a rating. This information includes: user token, restaurant ID, and rating.

The server shall do checks to make sure this information is valid, such as: user token validation, restaurant ID validation, and rating validation. These validations will ensure that the user is real and logged in, that the restaurant exists, and that the rating is a number between 0-5. If all information is valid, the rating shall be added to the database and the API endpoint shall return a response stating that the rating was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Remove Rating

API used to remove a rating from a restaurant. This API shall receive information in a JSON format. The JSON shall contain all information required to remove a rating. This information includes: user token, rating ID.

The server shall do checks to make sure this information is valid, such as: user token validation and rating ID validation. These validations will ensure that the user is real and logged in and that the rating exists. If all information is valid, the rating shall be removed from the database and the API endpoint shall return a response stating that the rating removal was successful. If it was not successful, it shall also return a reason. Response shall be in the JSON format.

Significant Non-trivial Algorithms

Restaurant Rating

Restaurant ratings shall be a combination of all ratings that a restaurant has from all users, weighed equally. This means that every time a user posts or removes a rating from a restaurant, the restaurant's rating must be updated. In short, the rating for a restaurant will be an *average* of all the ratings for that specific restaurant. In order to update the rating, you simply iterate over all ratings keeping track of the summation of all ratings. Once you have the count, simply divide by count the maximum number of possible stars.

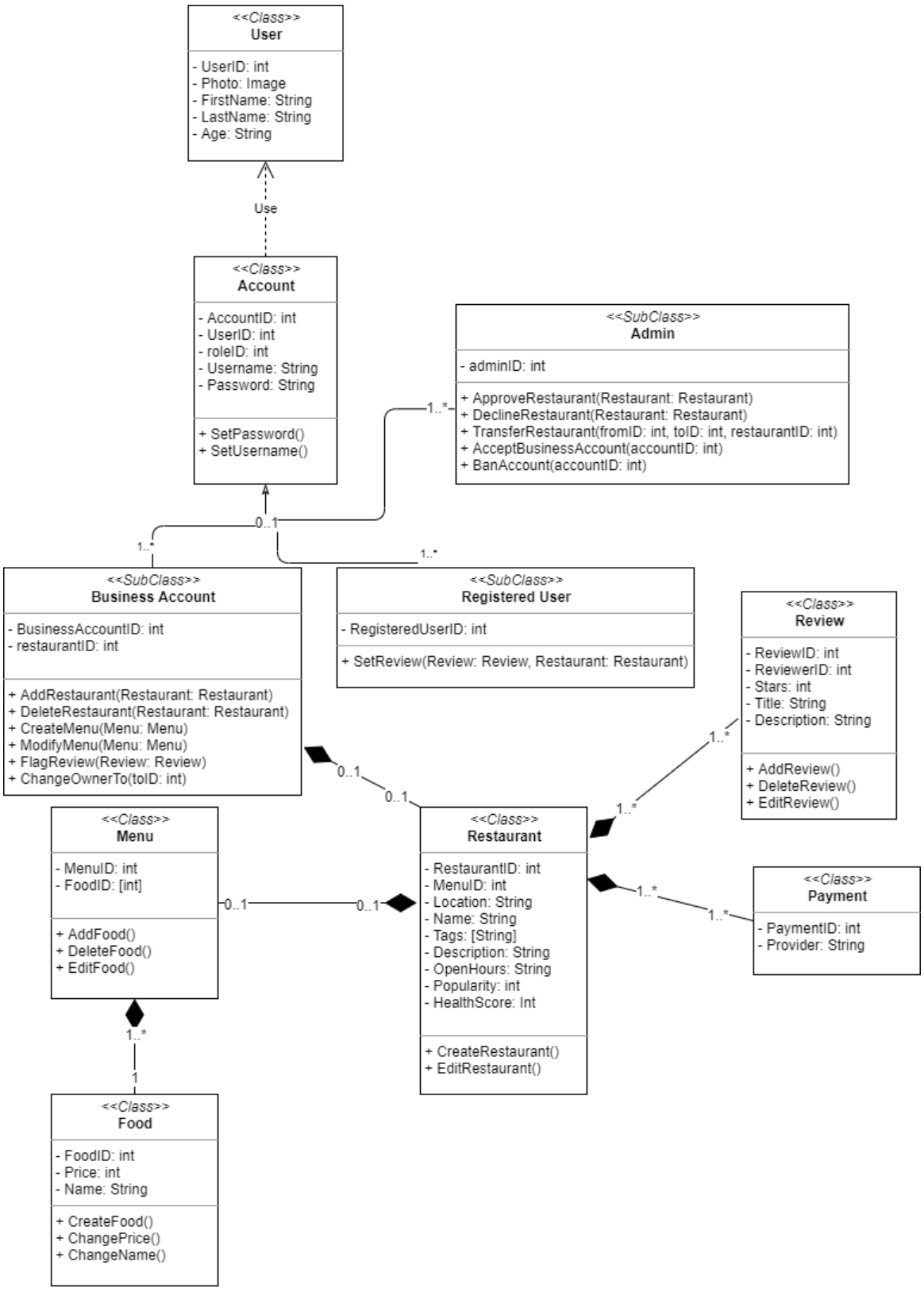
New Restaurants List

On the main page of the application, there will be a list of new restaurants that were added recently. In order to display these, we will keep an ordered list utilizing a stack structure. Everytime a new restaurant is added to the platform, it must also be added to the stack of new restaurants, which will keep track of an arbitrary number of new restaurants. By doing this, we ensure that we do not have to iterate over all restaurants by date, every time a user opens the main page.

Recent Reviews List

On the main page of the application, there will be a list of recent reviews that users have posted. In order to display these, we will keep an ordered list utilizing a stack structure. Everytime a new review is added to the platform, it must also be added to the stack of new reviews, which will keep track of an arbitrary number of recent reviews. By doing this, we ensure that we do not have to iterate over all reviews by date, every time a user opens the main page. In order to make sure that these reviews are actually recent, we can iterate over this small stack every hour, and remove all reviews left more than an hour ago.

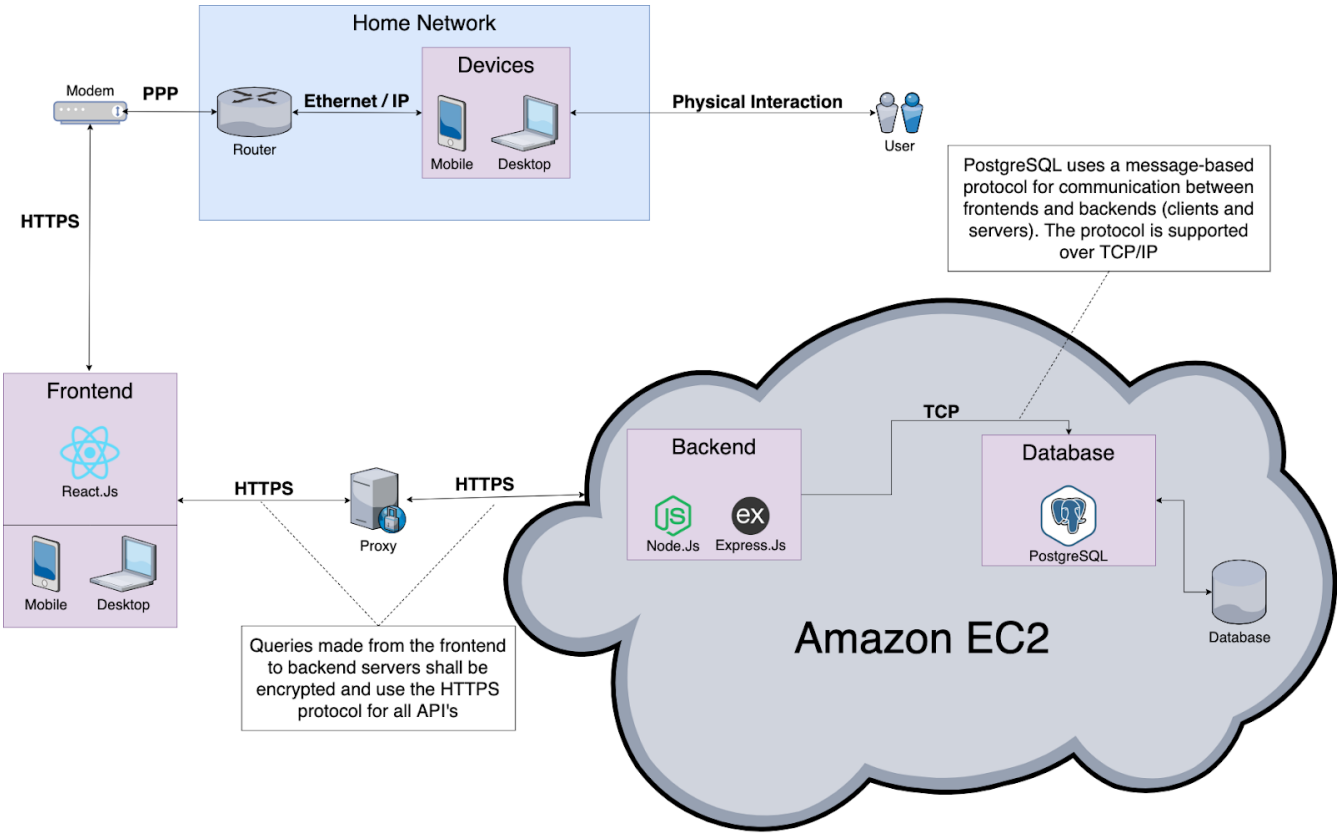
Section 6: High Level UML



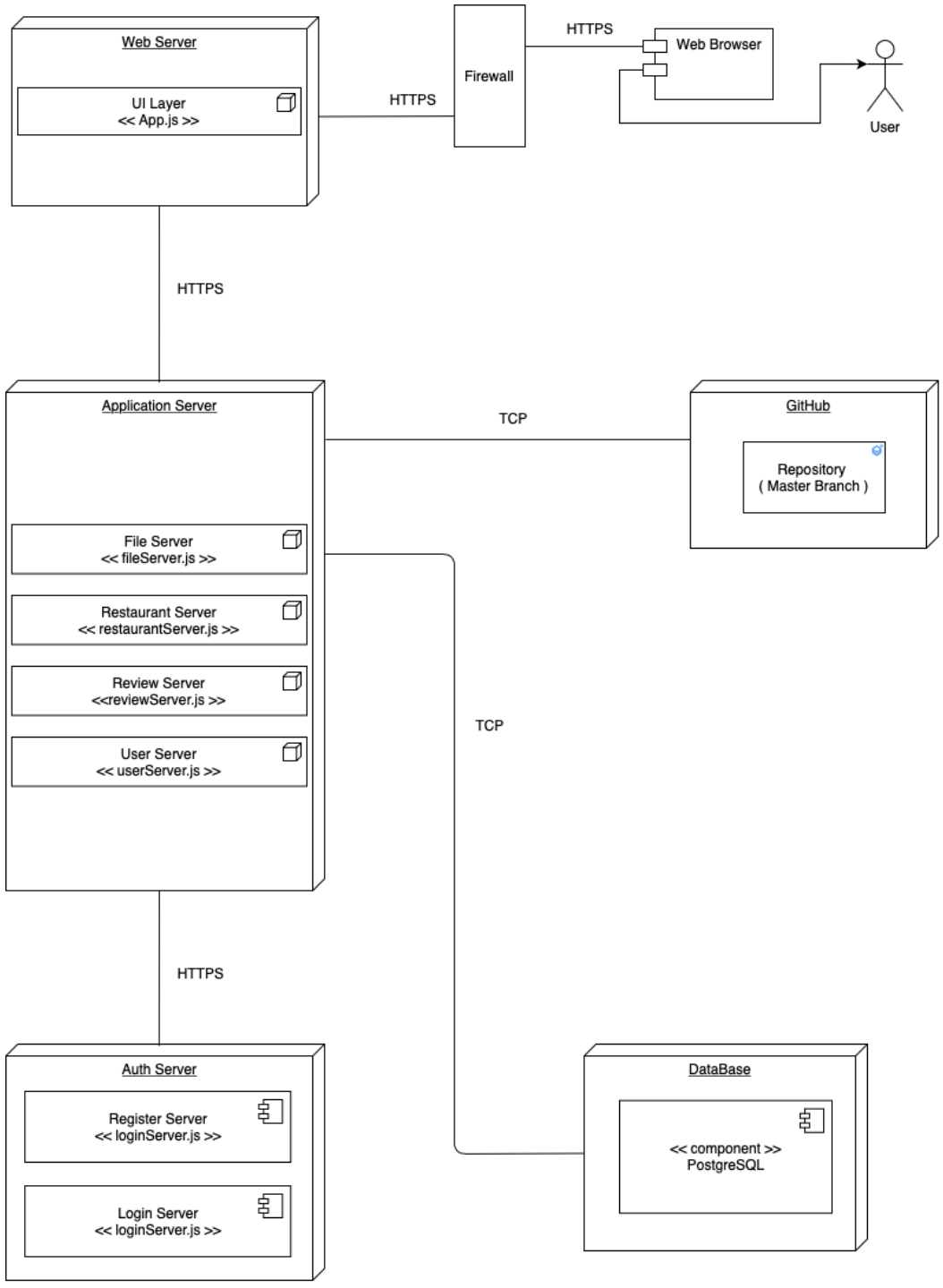
Diagrams

Section 7: High Level Application Network and Deployment Diagrams

Network Diagram



Deployment Diagram



Section 8: Key Risks of this Project

Technical risks

- EC2 or AWS failure will lead to downtime. There is no way to mitigate this other than a possible backup host.
- Increase in users will lead to an increase in database size which will increase server costs. In the event that this happens, an additional funding request will be made to the CEO.

Teamwork risks

- With the recent spread of coronavirus, we have shifted to an online format which has caused a gap in communication. Online meetings are now conducted more frequently in order to stay on track. We assume that after an initial adjustment period, the team will get more comfortable with the new format.
- With a small team it is possible that if the scope of the application gets too broad, we will spend more time maintaining and less time implementing new features. If this becomes the case, then we will first focus on priority 1 features.

Schedule risks

- With coronavirus and the associated teamwork risks, it could have an affect on the delivery of the application. It might affect every member in a different way. This is where the flexibility of online meetings can be used as an advantage. Also, as the situation develops, focus on priorities may change.

Skill risks

- The lack of experience is mainly due to the fact that there is no one in the team who can provide direction in terms of security infrastructure. The plan is to network with other people and manage it in-house till a more favourable solution is found.
- Constantly changing technology means that the software stack might become outdated on the line. Our technology stack being modular mitigates this problem to an extent.

Section 9: Project Management

Management of Tasks

In order to break down tasks for Milestone 2, our group first sat together and reviewed all sections of the Milestone together. For each section, we determined which category it belonged to: Front end, Back end, or Both. After having all sections in its own category, we went back and estimated total time to complete each section. Finally, we assigned each person one or multiple sections, such that the workload would be balanced while also ensuring that each person has a high probability to get a task for the team they were on. After everyone had a task, we agreed to meet every other day from 2:00 - 4:00 PM, in order to discuss progress we had made. This constant cycle of meeting and updating made sure we all progressed in our tasks.

For future tasks, we will use a similar approach. This approach worked well because it ensured that each task was given to the person who would perform the best at it. A small improvement we will utilize form here on out is the use of Trello. Trello will allow us to have real time check lists along with their progress, thus we would not have to wait to meet every other day to see this progress.

Tools Used

Since shifting to online format, team communication is being done on the internet entirely. Regular meetings are conducted on Discord which is a chat client that allows us to create separate voice and text channels for each team/task. Team collaboration on documents happens over google docs which allows simultaneous editing and commenting. Tracking and managing tasks shall be done through trello, starting with Vertical Prototype.

Milestone Three

SW Engineering CSC648/848 Spring 2019
Team 04
04/30/2020

SFStateEats

Team Members

Name	Email	Role
Rachit Joshi	rjoshi@mail.sfsu.edu	Team Lead
Samhita Brigeda	pandu.barigeda@gmail.com	Front End Lead
Pedro Souto	Pedro.Souto.SFSU@gmail.com	Back End Lead
John Pham	JohnPhamDeveloper@hotmail.com	Github Master
Vincent Tran	vtran6@mail.sfsu.edu	Database Lead
Khang Tran	ktran26@mail.sfsu.edu	Frontend Developer

Milestone 2 Version History

Milestone Version	Date
Milestone 3 Version 2	04/30/2020
Milestone 3 Version 1	04/23/2020
Milestone 2 Version 2	04/09/2020
Milestone 2 Version 1	03/22/2020
Milestone 1 Version 2	03/10/2020

Milestone 1 Version 1	02/24/2020

Table of Contents

Section 1: Data Definitions V3

System Administrator

Description

Account for those users who are employed or trusted by the application owners. These accounts have access to moderator tools that no other account has access to. This may include the ability to remove reviews or change business information.

Restrictions

- Due to the fact that these accounts are not public, system administrators do not have any data items related to public profiles. This includes: profile picture, description, date of birth, reviews, ratings, etc.

Sub-Data Items

- Email
- Username
- Password

Registered User

Description

A user who uses the website and has created an account. This allows them to access features that an unregistered user may not have.

Restrictions

- Registered users shall not be younger than 13 years old.
- Registered users shall use an email that is not already in use in the application.

Sub-Data Items

- Email
- Username
- Password
- Profile Picture (Photo)
- Date of Birth

Registered Business Owner

Description

A user who owns a business that is listed on SFStateEats. This account has access to features relating to business pages that other users do not.

Restrictions

- Registered business owners shall be older than 13 years old.
- Registered business owners shall use an email that is not already in use in the application.
- Registered business owner's phone number shall not be public.
- Registered business owner's addresses shall not be public.

Sub-Data Items

- Email
- Username
- Password
- Phone Number
- Address

Restaurant

Description

An entity that contains all the information about a restaurant. This will be the page that has reviews, menus, hours, map, and location

Restrictions

- Restaurant shall not be listed unless it has been verified by a system administrator.
- Restaurants shall not upload more than 15 photos.
- Restaurant shall be located on campus.

Sub-Data Items

- Description
- Photo
- Review
- Menu
- Hours
- Address

Review

Description

A data item relating to restaurants. Reviews can be left by registered users and can be seen by any user. Reviews may include a section of text, photos, and ratings. A review consists of 3 parts, and may also include photos if the user wishes.

Restrictions

- A review shall have at most 3 photos uploaded.
- A review shall not require all sub data items to be filled out, as long as one of them is. For instance, if a user wants to only leave a text review, reaction/rating/photos may be left blank to achieve that.

Sub-Data Items

- Review Text
- Rating
- Photo

Menu

Description

A data item relating to restaurants. Restaurants may have a menu. A menu lists all of the items that the restaurant offers.

Restrictions

- A restaurant shall have at most 1 menu.

Sub-Data Items

- Menu Item

Menu Item

Description

A data item relating to a menu. A menu can have multiple menu items. Menu items are the individual items that a restaurant offers.

Restrictions

- A menu item shall have at most 1 photo.
- A menu item shall have an item name.
- A menu item shall have a price.

Sub-Data Items

- Description
- Price
- Photo
- Item Name

Item Name

Description

A data item relating to menu item. Every item on a menu must have a name. This is the text data for that name.

Restrictions

- An item name shall have at least 1 character.
- An item name shall have at most 50 characters.
- An item name shall not contain an emojis.

Sub-Data Items

- None

Price

Description

A data item relating to menu item. Every item on a menu must have a price. This is the numerical data for price.

Restrictions

- A price shall be denoted in USD.
- A price shall not be null.

Sub-Data Items

- None

Address

Description

A data item relating to restaurants. An address will be written information regarding where the restaurant can be found.

Restrictions

- An address shall only consist of text.
- An address shall have at most 100 characters.
- An address shall not have any emojis.

Sub-Data Items

- None

Email

Description

An email will be a stored piece of information that allows for contact, verification, and login for the user's of the application.

Restrictions

- An email shall contain only characters.
- An email shall not be longer than 50 characters.
- An email shall have an @ symbol, and end with a valid extension.
- An email shall not contain any emojis.

Sub-Data Items

- None

Username

Description

A username will be a stored piece of information that allows for distinction and login for the user's of the application.

Restrictions

- A username shall contain only characters.
- A username shall not be longer than 25 characters.
- A username shall not contain any emojis.
- A username shall be unique.

Sub-Data Items

- None

Password

Description

A password will be a stored piece of information that allows for account login validation for the user's of the application.

Restrictions

- A password shall contain only characters.
- A password shall not be longer than 250 characters.
- A password shall not be shorter than 8 characters.
- A password shall not contain any emojis.
- A password shall contain at least 1 number, 1 uppercase letter, 1 special character, and 1 lowercase letter.
- A password shall be encrypted when stored.

Sub-Data Items

- None

Photo

Description

A photo will be a stored piece of media. Photos have many uses such as profile pictures, restaurant photos, and review photos.

Restrictions

- A photo shall not have a file size larger than 2MB.
- A photo shall not have dimensions larger than 1920 x 1080 pixels.
- A photo must be in one of the following formats: .JPG, .JPEG, .PNG, .HEIC.

Sub-Data Items

- None

Date of Birth

Description

A date of birth will be a text piece of data. It shall be used for verification of age when creating an account.

Restrictions

- A date of birth shall follow the MM-DD-YYYY format.
- A date of birth shall have exactly 8 positive numeric characters.
- A date of birth shall have a year greater than 1900.
- A date of birth shall have a day greater than 0 and less than 32.
- A date of birth shall have a month greater than 0 and less than 13.

Sub-Data Items

- None

Description

Description

A description will be a text piece of data. It shall be used for describing various entities on the application.

Restrictions

- A description shall contain no emojis.
- A description shall be no longer than 500 characters.

Sub-Data Items

- None

Phone Number

Description

A phone number will be a text piece of data. It shall be used to have contact information for various users of the application.

Restrictions

- A phone number shall follow the (###)###-#### format.
- A phone number shall have exactly 10 positive numeric characters.
- A phone number shall not have any negative numbers.

Sub-Data Items

- None

Hour

Description

An hour will be a text piece of data. It shall be used by restaurants to display the operating hours in which they are open to the public.

Restrictions

- An hour shall follow the *“DAY”* HH:MM *“AM/PM”* format.
- The *“DAY” string shall exactly match one of the seven days of the week.*
- The *“AM/PM”* string shall exactly match either “AM” or “PM”.
- The HH:MM shall contain only exactly 4 positive numeric values.

Sub-Data Items

- None

Review Text

Description

A review text shall be a text piece of data. It shall be used when a registered user leaves a review for a restaurant.

Restrictions

- A review text shall contain no emojis.
- A review text shall be no longer than 2500 characters.

Sub-Data Items

- None

Rating

Description

A rating is a score created when a registered user rates his experience. This rating is from 0-5, 0 being the worst, and 5 being the best. This rating can then be aggregated and displayed visually.

Restrictions

- A rating shall be a number between 0 and 5, inclusive.

Sub-Data Items

- None

Section 2: Functional Requirements V3

Priority One

User

1. New users shall be able to create an account.
2. Users shall be able to login into their account.
    2.1. A user shall only be logged in to one session at any given time.
3. Users shall be able to rate restaurants on campus.
    3.1. A user shall only rate a restaurant once.
    3.2. A user shall only rate a restaurant if they are reviewing it.
4. Users shall be able to rate an event on campus.
    4.1. A user shall only rate and event once.
    4.2. A user shall only rate an event if they are reviewing it.
5. Users shall be able to write reviews.
    5.1. A user shall be able to write a review only once per restaurant.
    5.2. A user shall be logged in to write a review.
6. Users shall be able to view reviews.
8. Users shall be able to see the menu for each restaurant.
9. Users shall be able to see the price for each item on the menu.
10. Users shall be able to see the address of the restaurant on the profile page.
    10.1. An address shall be a text description of where the restaurant is located on campus.
15. Users shall be able to see hours of business for each restaurant.
    15.1. Hours shall be updated for holidays and special occasions.
    15.2. Hours shall be formatted using the HH:MM format.
16. Users shall be able to apply filters to their search.
    16.1. A user shall be able to choose from one or many of the following filters:
        16.1.1. Recommendation
        16.1.2. Rating
        16.1.3. Reviews
        16.1.4. Price
        16.1.5. Hours
        16.1.6. Address / Distance
        16.1.7. Categories
        16.1.8. Payment Methods
17. Users shall be able to flag a review as inappropriate for the application.
    17.1. A user can only flag 10 reviews per day.
    17.2. The amount of flags a review has shall not be public.
18. Users shall be able to sort reviews by date.
    18.1. Users shall be able to sort by newest, oldest or specific date.

19. Users shall be able to sort reviews by popularity.
    19.1. A review's popularity is based on the number of positive reactions a review has.
    19.2. A user shall be able to sort popularity by most popular or least popular.
20. Users shall be able to delete a review they have written.
    20.1. A review can only be deleted within a specified (24 hours) time frame.
22. Users shall be able to see all reviews they have written.
23. Users shall be able to see all reviews another user has written.
28. Users will be able to see a visual representation of the ratings.
    28.1. The visual representation shall use a star format.
    28.2. A rating shall be on a scale from 0-5, in half increments.
29. Users shall be able to reply to other user's reviews.
    29.1. A reply shall have the option to be a reaction.
    29.2. A reply shall have the option to be text.
    29.3. A reply shall have the option to be rating.
    29.4. A text reply shall have a maximum of 500 characters.

Business

30. New business owners shall be able to register a business account.
    30.1. A business shall be verified by a system administrator.
    30.2. Business account verification shall take up to 3 business days.
    30.3. Business account verification shall require additional details (proof of ownership, etc).
31. Business owners shall be able to login into their account.
    31.1. A business account shall only be logged in to one session at any given time.
32. Business owners shall be able to add their restaurant to the website.
    32.1. A new restaurant shall be verified by a system administrator.
    32.2. New restaurant verification shall take up to 3 business days.
    32.3. New restaurant verification shall require additional details (proof of ownership, etc).
33. Business owners shall be able to create a menu for their restaurant.
    33.1. A menu shall have the option for subsections.
    33.2. All menu items shall be required to have a price
    33.3. A menu item shall have the option to have categories.
34. Business owners shall be able to upload photos of their restaurant.
    34.1. A restaurant shall have at most 15 pictures uploaded.
36. Business owners shall be able to write descriptions of their restaurant on the profile page.
    36.1. Restaurant descriptions shall be at most 500 characters.
38. Business owners shall be able to change prices on the menu.
39. Business owners shall be able to change hours of the restaurant.

41. Business owners shall be able to flag a review as inappropriate for the website.
    41.1. A review flagged by a business shall have more weight than flags by users.
    41.2. A business can only flag 25 reviews per day.
    41.3. The amount of flags a review has shall not be public.
42. Business owners shall be able to request removal of their restaurant from the website.
    42.1. A request for deletion shall be required to have an explanation.
    42.2. A request for deletion shall be verified by the system administrator.
71. Business owners shall be able to upload photos of each item on the menu.
    71.1. A menu item shall only have 1 picture uploaded

Restaurant

44. The restaurant profile page shall display photos.
    44.1. Photos shall not be inappropriate.
45. The restaurant profile page shall display ratings.
46. The restaurant profile page shall display tags.
    46.1. Restaurant shall display at most 10 tags.
47. The restaurant profile page shall display a description.
    47.1. Description shall not contain any emojis or inappropriate language.
48. The restaurant profile page shall display address.
49. The restaurant profile page shall display a menu.
50. The restaurant profile page shall display hours.
51. The restaurant profile page shall display reviews.
52. The restaurant profile page shall display the owner.
    52.1. Restaurant owners shall decide if the restaurant displays the owner's account.
53. The restaurant profile page shall display phone numbers.
    53.1. A restaurant shall have at most 2 phone numbers.
    53.2. A business owner shall not have the same phone number as a restaurant phone number.
58. The restaurant profile page shall display the various payment methods the restaurant accepts.
    58.1. Payment methods shall be updated as systems change.

System Admin

60. System Administrators shall have the privilege to ban any users from the website for misuse.
    60.1. A ban shall be required to have a reason logged.
61. System Administrators shall have the privilege to delete restaurants from the platform.

61.1. A deletion shall be required to have a reason logged if the deletion was in result of a ban.
61.2. A deletion may also happen as a result of a restaurant removal request ticket.

62. System Administrators shall have the privilege to add restaurants to the platform.
62.1. All restaurants added to the platform shall have relevant documents for proof of ownership.
63. System Administrators shall have the privilege to change restaurant information.
64. System Administrators shall have the privilege to verify new business accounts.
64.1.New business accounts shall undergo review.
65. System Administrators shall have the privilege to delete reviews from the platform.
65.1. A review that receives too many flags shall open a ticket for further investigation.
65.2. A review that is removed shall be required to have a reason logged.

System

70. The system shall show new restaurants on the main page.
70.1. New restaurants shall be determined based on when they were approved on the platform.

Priority Two

User

10. Users shall be able to see all the businesses that are involved in the farmers market.
10.1. Businesses involved in the farmer's market shall have a special restaurant tag.
11. Users shall be able to see the map locations of each restaurant on campus.
11.1. Map location shall be based on Google API.
11.2. Map shall be able to pan and zoom.
11.3. Map shall provide GPS like directions to restaurants.
21. Users shall be able to edit a review they have written.
21.1. A review shall only be able to be edited within a certain time frame (2 hours).
21.2. A review shall display to all users that it has been edited.
21.3. A review shall only be edited at most 3 times.
25. Users shall be able to share restaurant information and profiles to each other.
25.1. Share shall be done over via a direct link to the restaurant.
7. Users should be able to post pictures in their reviews.
7.1. A user shall upload a maximum of 3 images per review.

Business

35. Business owners shall be able to tag their restaurant into categories.
    35.1. A restaurant shall have at most 15 tags.
37. Business owners shall be able to upload an approved health score.
    37.1. A health score shall be uploaded as an image for verification.
    37.2. Health score image shall contain all information required for proof validity.
40. Business owners shall be able to petition for a change of owner.
    40.1. Petitions for change of owner shall be required to have a reason.
    40.2. New business owner must have either a verified business account or a new business account under review.

Restaurant

56. The restaurant profile page will show any nutritional facts of the menu items.
57. The restaurant profile page shall show the customers if they serve halal, vegan, vegetarian, and non-vegetarian meals.
    57.1. These options shall be categorized as tags and appear in filters.

System

69. The system shall show recent reviews on the main page.

Priority Three

User

13. Users shall be able to check-in to restaurants.
    13.1. Check-in shall not use GPS for verification.
14. Users shall be able to see busy times for restaurants.
    14.1. Busy times shall be calculated based on check-in's.
26. Users will be able to connect their facebook and instagram to their accounts.
    26.1. Connections allow for users to share links directly to their social media feeds.
27. Users will be able to login with their sfsu id that is linked to ilearn.
    27.1. User ID and password shall be identical to the university login.
29. Users who are visually impaired will be able to use an automated system (such as voice commands) to navigate through the app.
    29.1. Websites shall use standard disability formats for this.

Business

43. Business owners shall be able to see a chart of how their restaurant is doing.
    43.1. Charts shall display information such as check-in's, positive reviews versus negative reviews, etc.

Restaurant

54. The restaurant profile page shall display the option to order online.
55. The restaurant profile page shall display an option to pick up the order.

System Admin

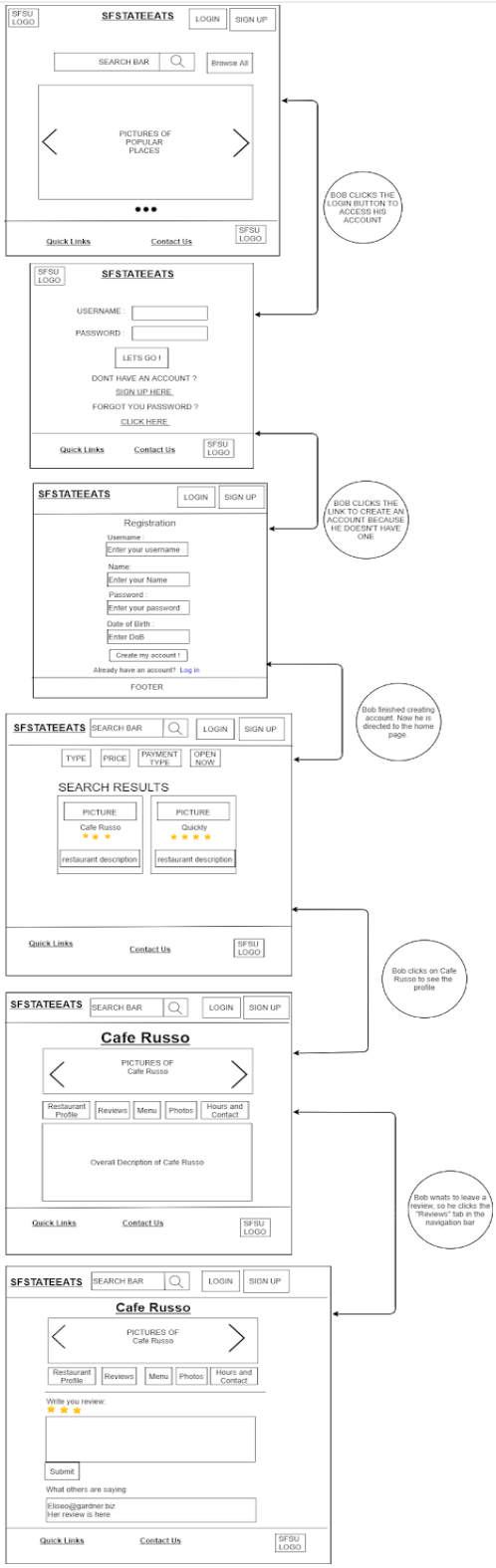
66. System Administrators shall be able to cancel orders placed at the customer's request.
    66.1. Customer cancellation requests shall be required to have a reason for cancelation.
67. System Administrators shall be able to refund the customer if the order is not what they wanted.

System

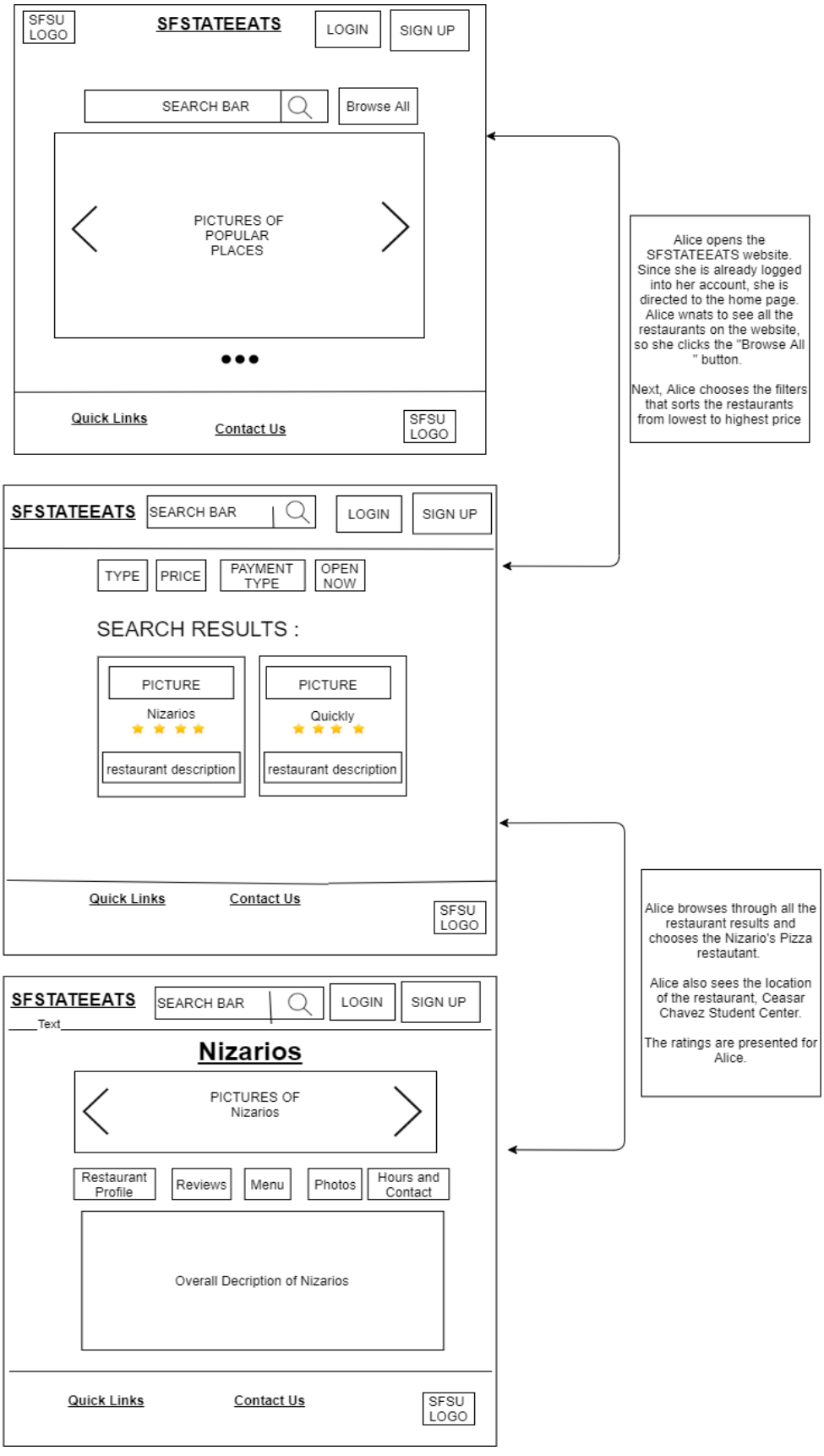
68. The system shall show trending restaurants on the main page.
    68.1. Trending restaurants are based on check-in's and new positive reviews.
    68.2. Trending restaurants are updated every 3 days.
    68.3.Display new reviews on the page.

Section 3: Wireframes Based on your Mockups/Storyboards V2 (detailed)

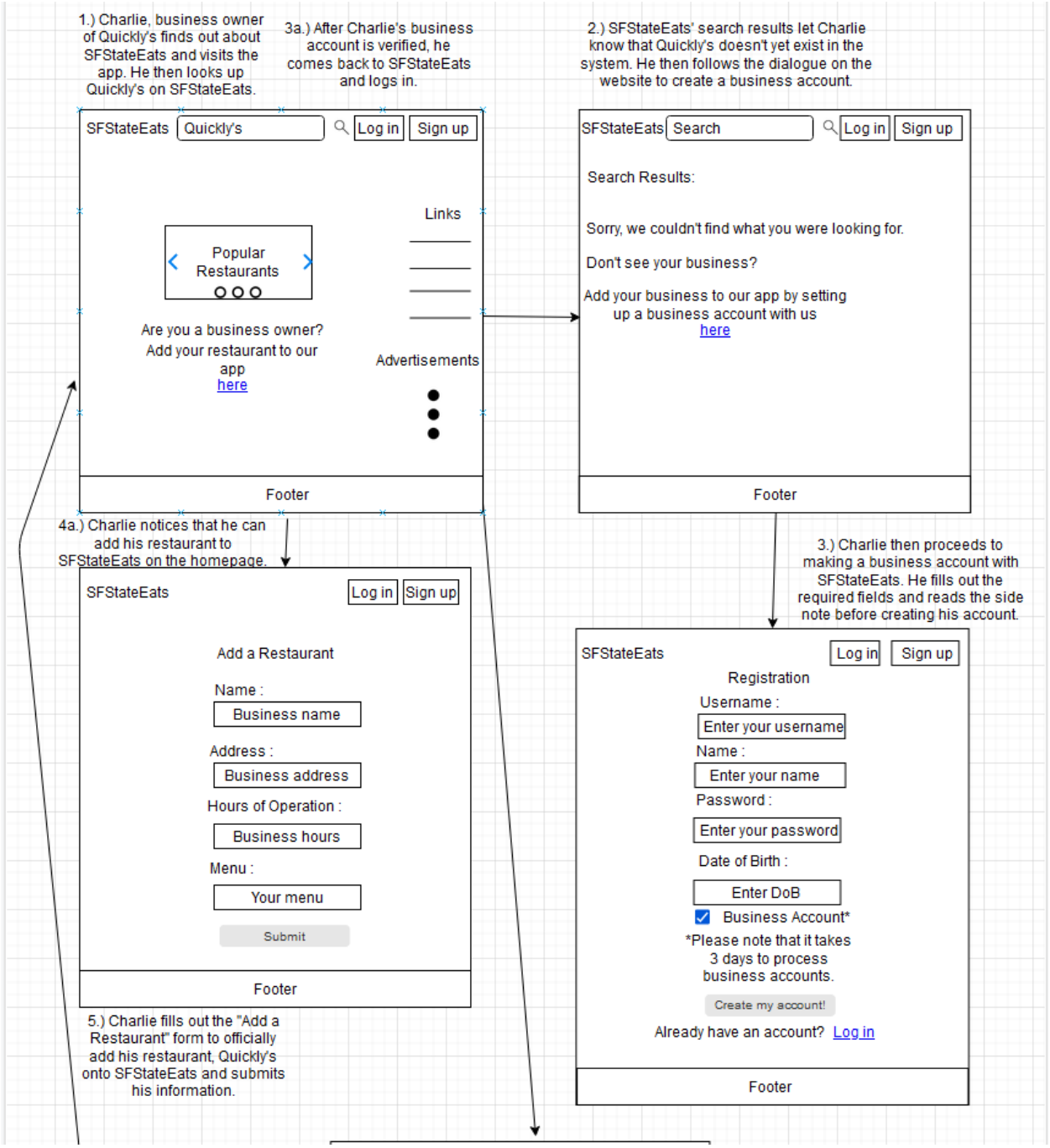
WireFrame for Use Case 1

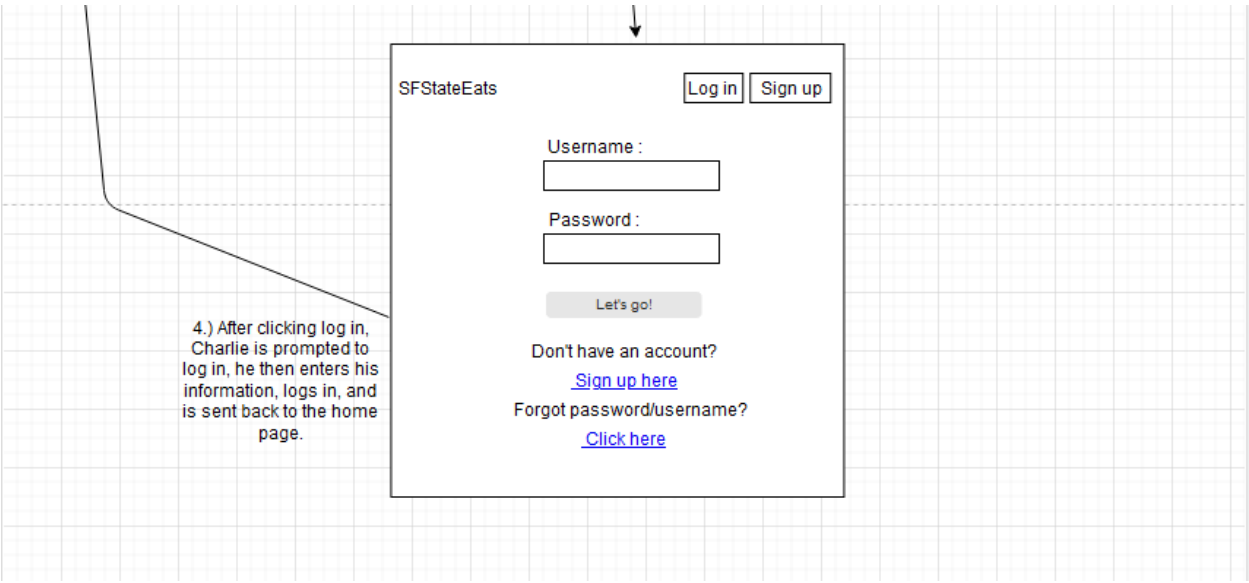


WireFrame for Use Case 2

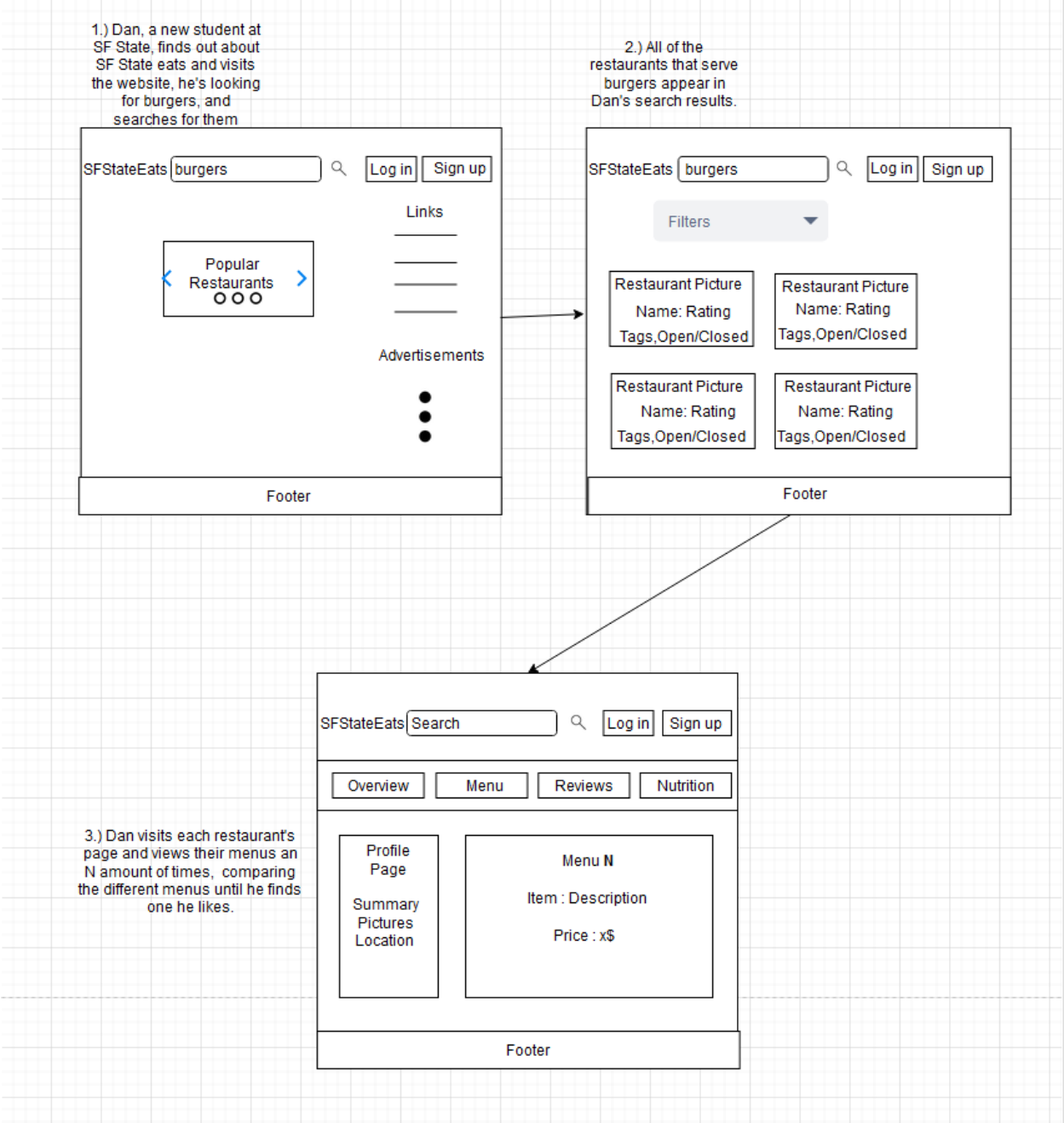


WireFrame for Use Case 3

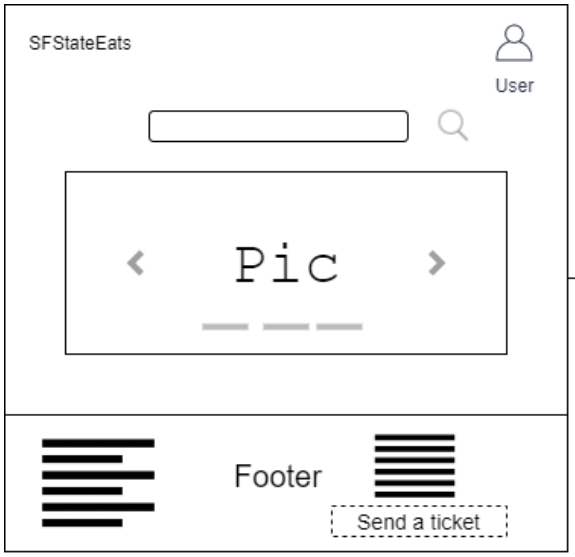



SFStateEats
Log in
Sign up
Username :
Password :
Let's go!
Don't have an account?
Sign up here
Forgot password/username?
Click here
4.) After clicking log in, Charlie is prompted to log in, he then enters his information, logs in, and is sent back to the home page.

WireFrame for Use Case 4

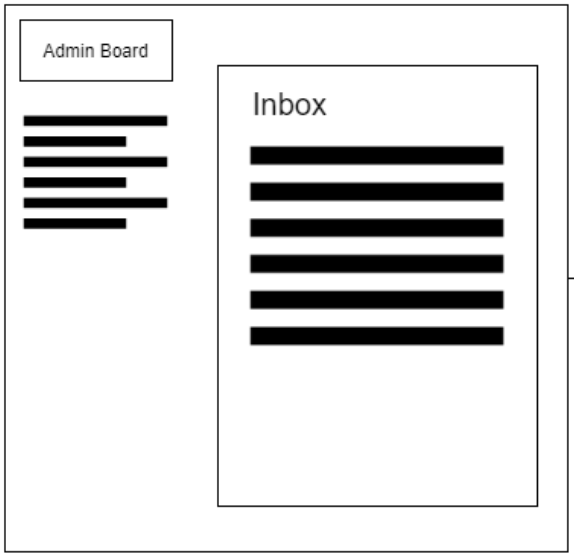


WireFrame for Use Case 5

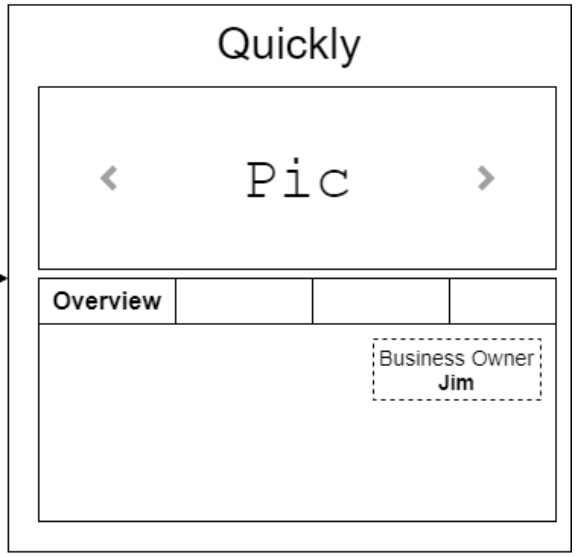


Charlie wants to send a ticket to sysadmin to update Quickly's owner to the new owner, Jim.

Charlie enters the details and upload necessary documents

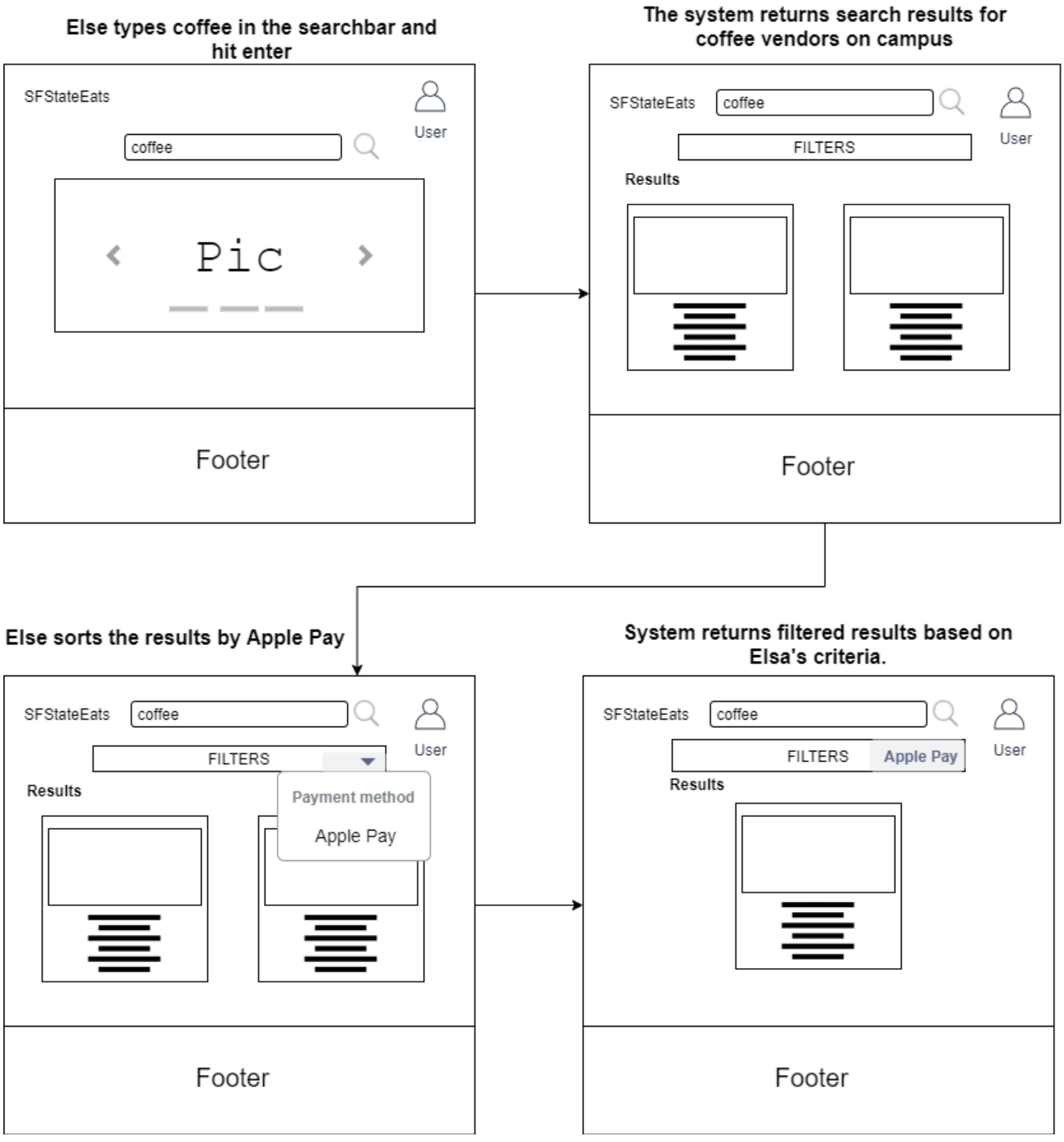


The sysadmin sees Charlie's message in the admin dashboad

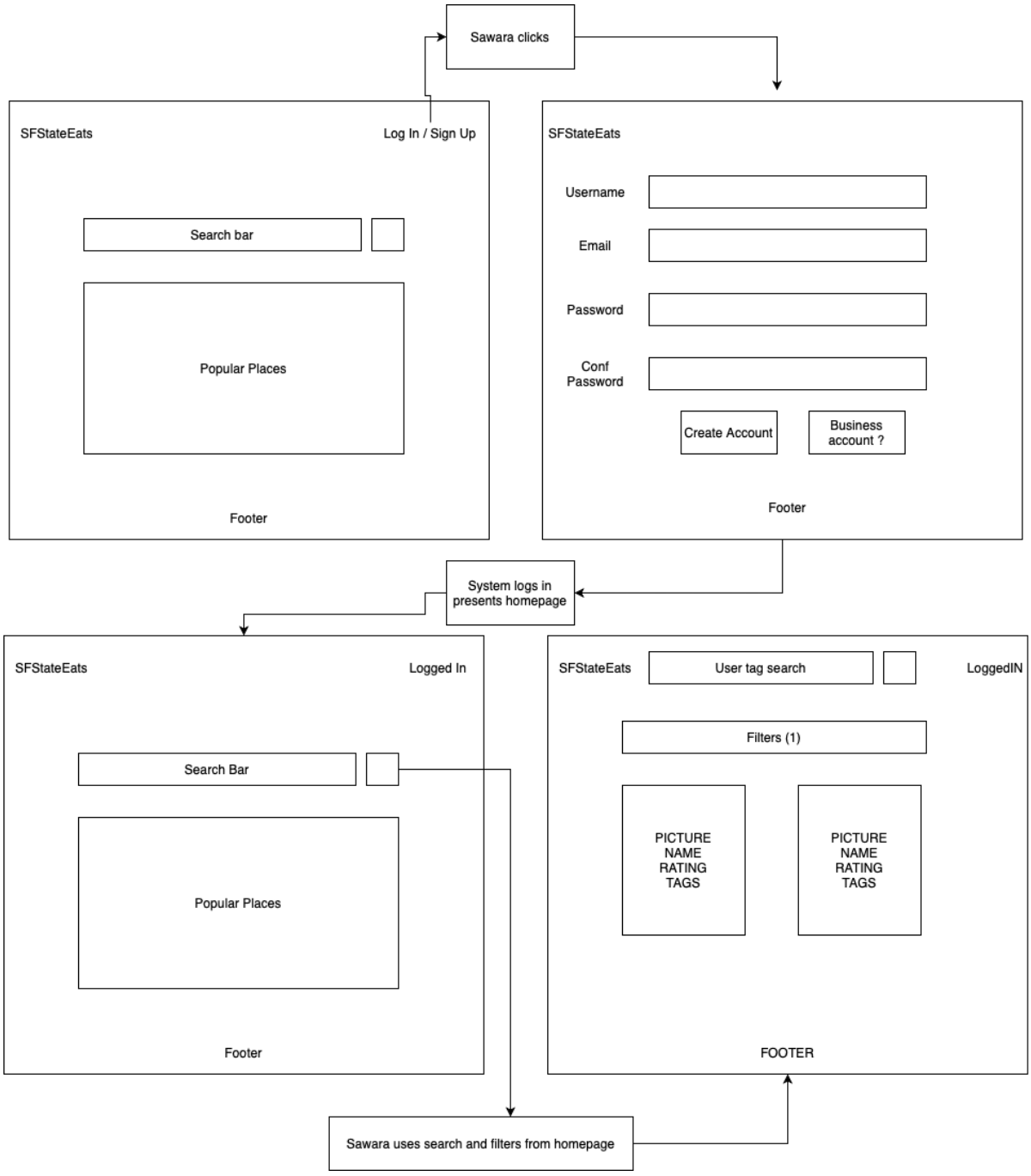


The sysadmin updates Quickly page to reflect new updated info

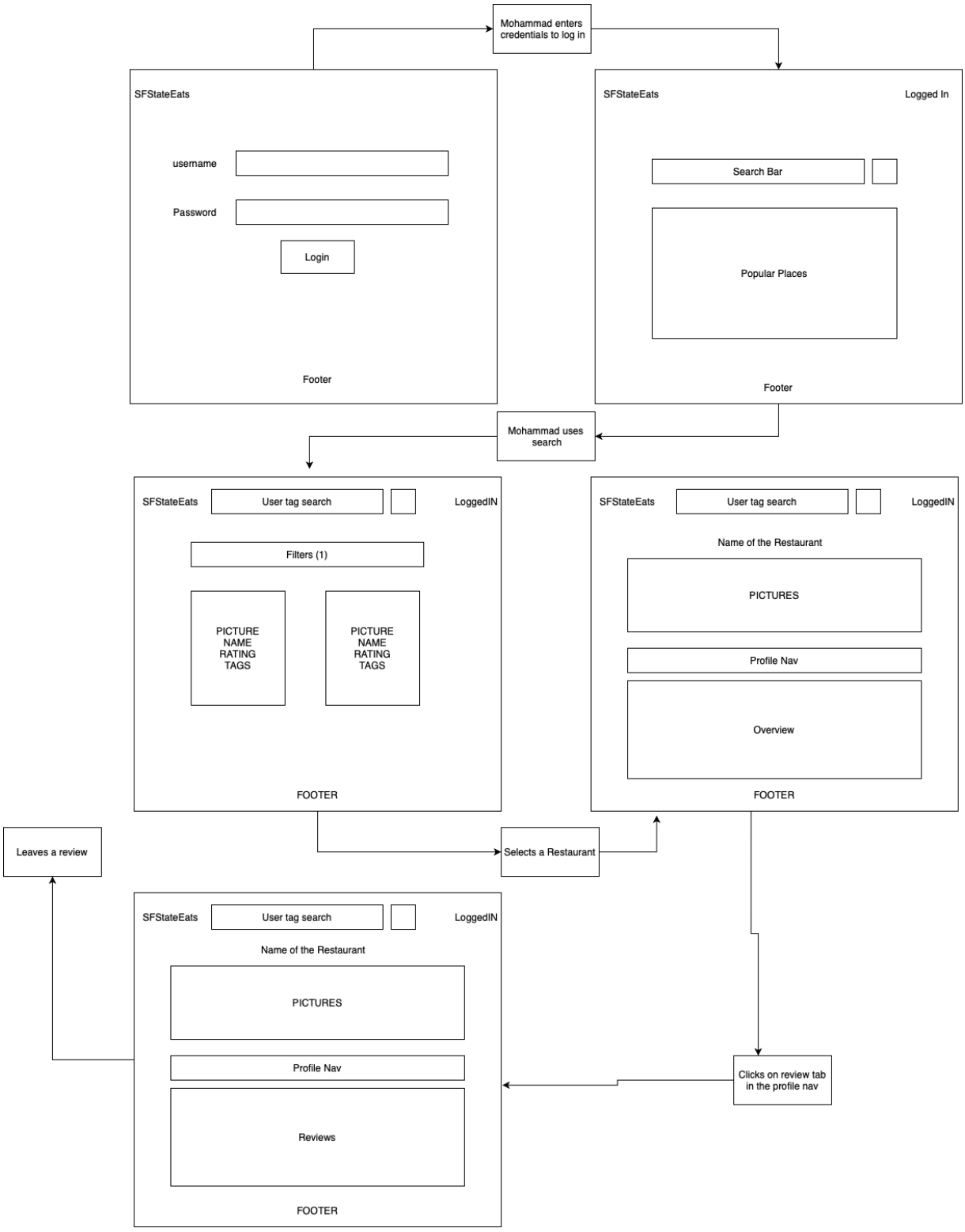
WireFrame for Use Case 6



WireFrame for Use Case 7

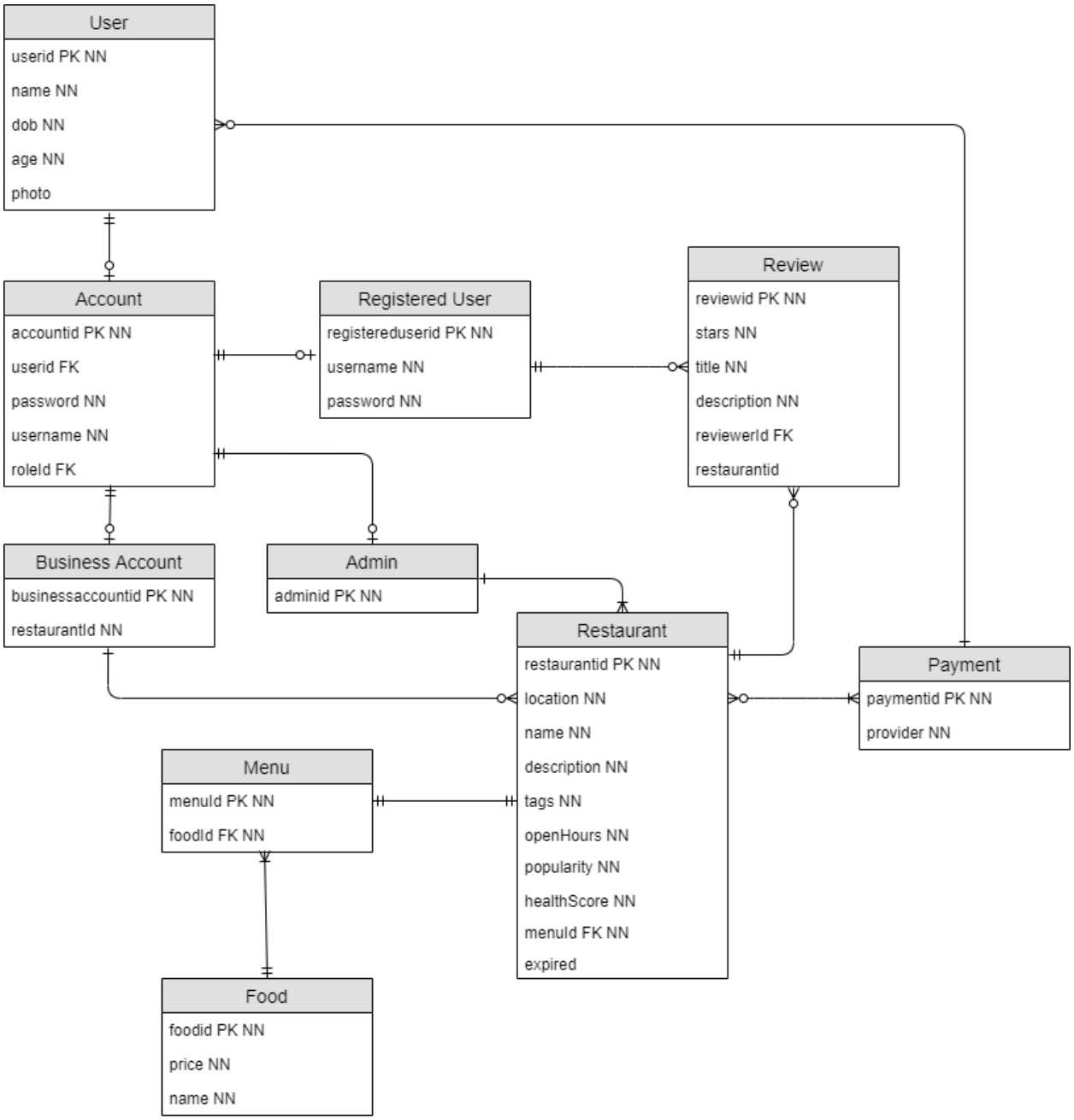


WireFrame for Use Case 8



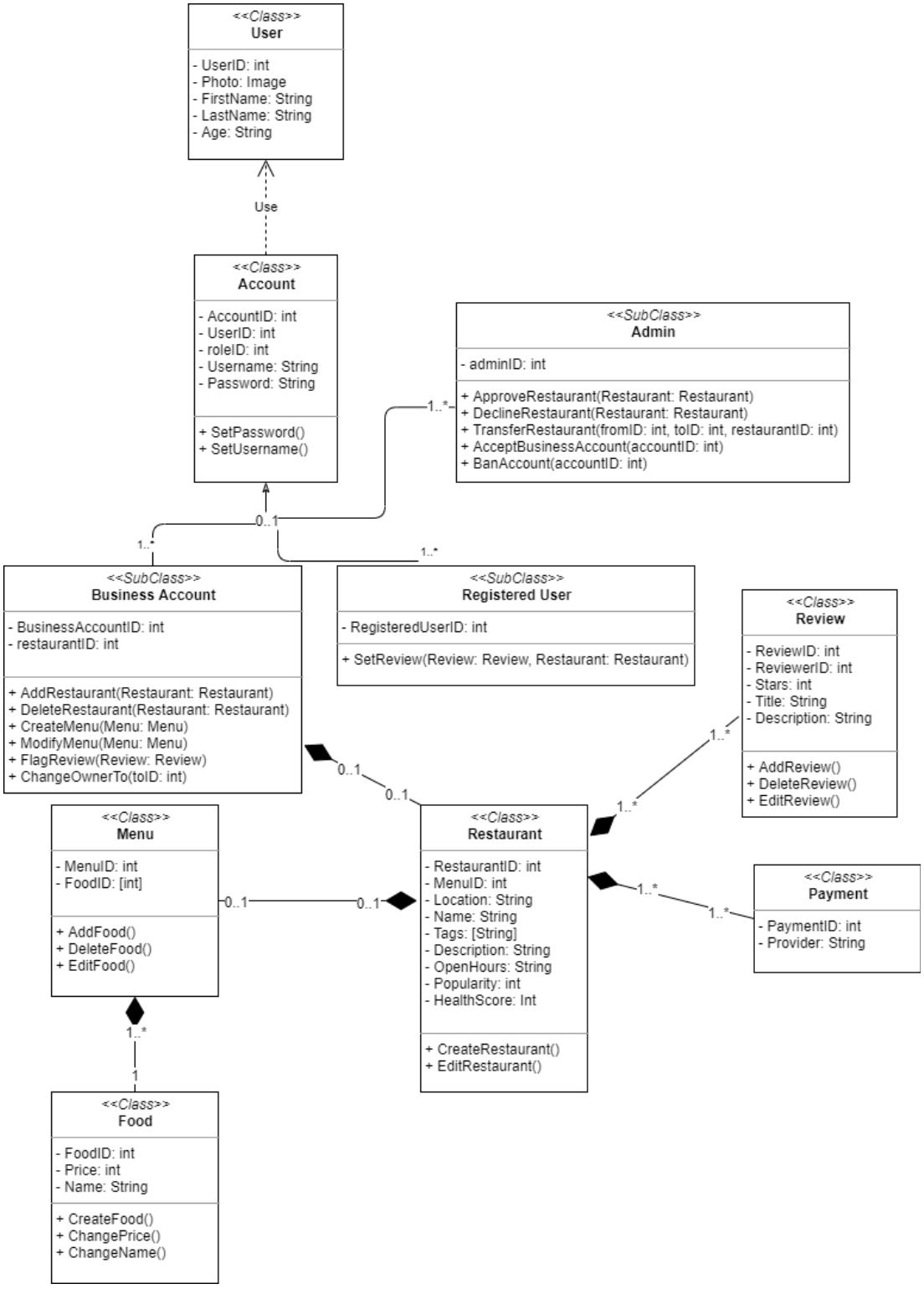
Section 4: High level database architecture and organization V2

Database Model

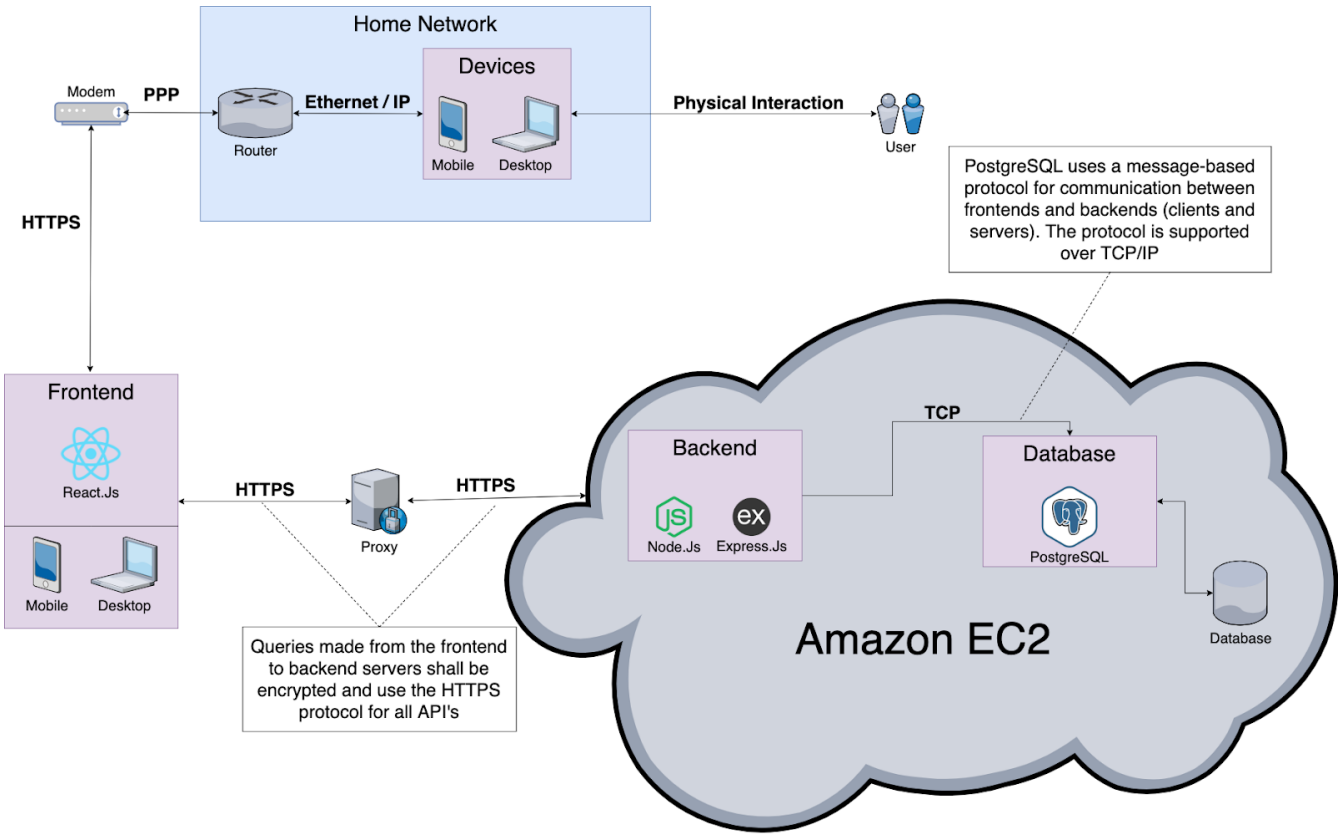


Section 5: High Level Diagrams V2

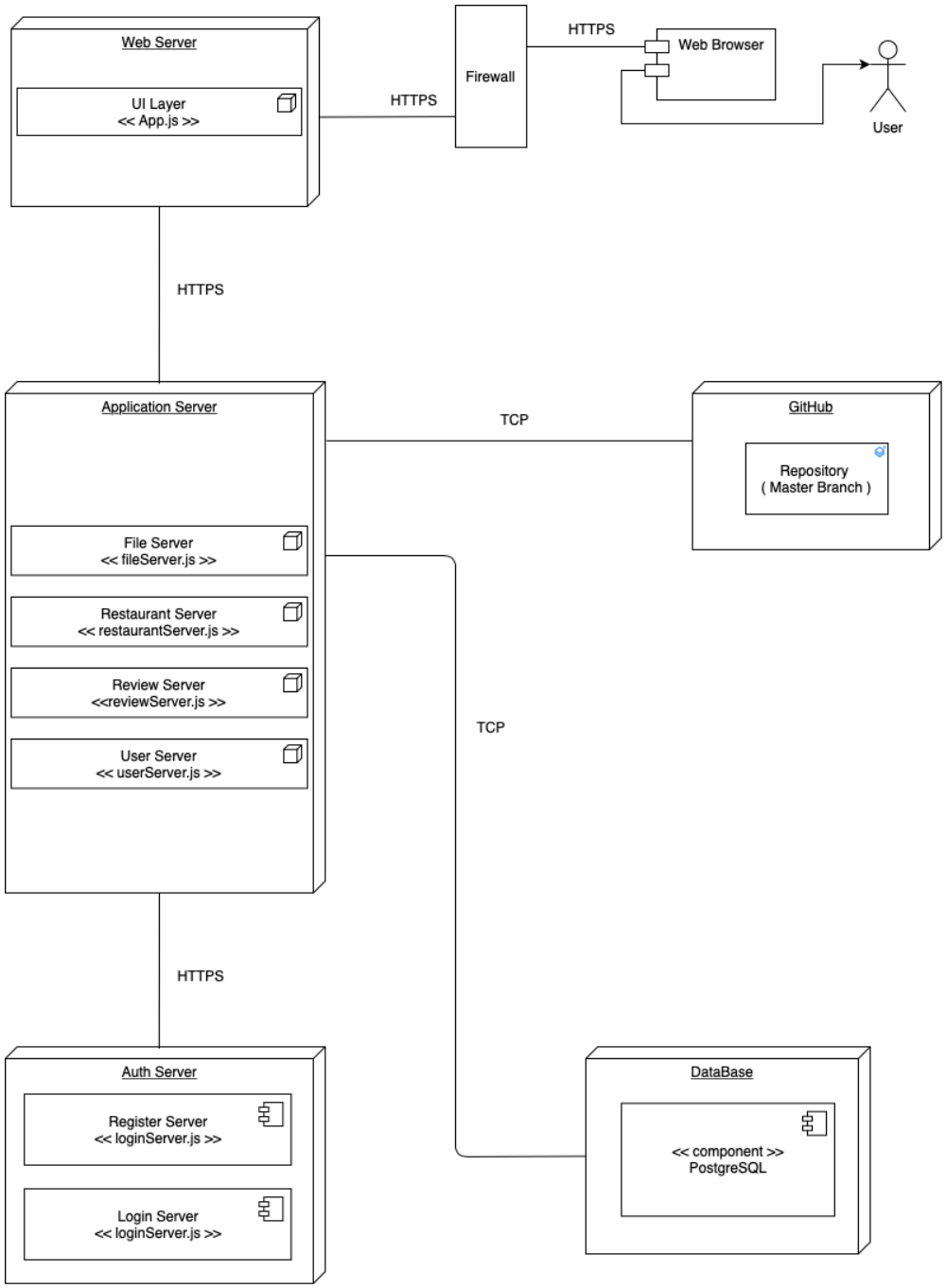
UML Diagram



Network Diagram



Deployment Diagram



Section 6: Customer Feedback

- Picture on the landing page should be smaller. As it stands, it overwhelms the user. As it stands, the picture slide is so large that it extends off the screen.
- Menus need to have an explanation of what they do. The “ticket” page for instance, needs to explain that a ticket is used to contact the support/admin of the website with any issues.
- On the restaurant info page, the navigation bar needs to be moved down with a larger font size. At the moment, it is hard for the user to easily see where to go for navigation within that page.
- On the restaurant info page, in the reviews tab, the submit button needs to be patted down. As it stands, the button is flush with the review text box, which does not look good.
- On the search page, the results need to use a larger portion of the window. As it stands right now, the results are squeezed into the middle 50% of the screen, but should be closer to 75% to make results clearer.
- Footer needs to be a bit smaller, because it dominates the website currently.
- The picture in the bottom right of the footer needs to have a clear background for a more professional look and feel.

Milestone Four

SW Engineering CSC648/848 Spring 2019 - Team 04
05/19/2020

SFStateEats

Team Members

Name	Email	Role
Rachit Joshi	rjoshi@mail.sfsu.edu	Team Lead
Samhita Brigeda	pandu.barigeda@gmail.com	Front End Lead
Pedro Souto	Pedro.Souto.SFSU@gmail.com	Back End Lead
John Pham	JohnPhamDeveloper@hotmail.com	Github Master
Vincent Tran	vtran6@mail.sfsu.edu	Database Lead
Khang Tran	ktran26@mail.sfsu.edu	Frontend Developer

Milestone 4 Version History

Milestone Version	Date
Milestone 4 Version 2	05/19/2020
Milestone 4 Version 1	05/11/2020
Milestone 3 Version 2	04/30/2020
Milestone 3 Version 1	04/23/2020
Milestone 2 Version 2	04/09/2020
Milestone 2 Version 1	03/22/2020
Milestone 1 Version 2	03/10/2020
Milestone 1 Version 1	02/24/2020

Table of Contents

Section 1: Product Summary

Name of Product

SFStateEats

Committed Functionalities

User

- New users shall be able to create an account.
- Users shall be able to login into their account.
- Users shall be able to rate restaurants on campus.
- Users shall be able to write reviews.
- Users shall be able to view reviews.
- Users shall be able to see the menu for each restaurant.
- Users shall be able to see the price for each item on the menu.
- Users shall be able to see the address of the restaurant on the profile page.
- Users shall be able to see hours of business for each restaurant.
- Users shall be able to apply filters to their search.
- Users shall be able to flag a review as inappropriate for the application.
- Users shall be able to sort reviews by date.
- Users shall be able to sort reviews by popularity.
- Users shall be able to delete a review they have written.
- Users shall be able to see all reviews they have written.
- Users shall be able to see all reviews another user has written.
- Users will be able to see a visual representation of the ratings, which shall be displayed on a scale of 0-5 stars.
- Users shall be able to reply to other user's reviews.[(Unique Feature)]

Business

- New business owners shall be able to register a business account.
- Business owners shall be able to login into their account.
- Business owners shall be able to add their restaurant to the website.
- Business owners shall be able to create a menu for their restaurant.
- Business owners shall be able to upload photos of their restaurant.
- Business owners shall be able to write descriptions of their restaurant on the profile page.
- Business owners shall be able to delete a menu for his restaurant.
- Business owners shall be able to change hours of the restaurant.

- Business owners shall be able to flag a review as inappropriate for the website.
- Business owners shall be able to request removal of their restaurant from the website.

Restaurant

- The restaurant profile page shall display photos.
- The restaurant profile page shall display ratings.
- The restaurant profile page shall display tags.
- The restaurant profile page shall display a description.
- The restaurant profile page shall display address.
- The restaurant profile page shall display a menu.
- The restaurant profile page shall display hours.
- The restaurant profile page shall display reviews.
- The restaurant profile page shall display the owner.
- The restaurant profile page shall display phone numbers.
- The restaurant profile page shall display the various payment methods the restaurant accepts.

System Admin

- System Administrators shall have the privilege to ban any users from the website for misuse.
- System Administrators shall have the privilege to delete restaurants from the platform.
- System Administrators shall have the privilege to change restaurant information.
- System Administrators shall have the privilege to delete reviews from the platform.

Unique Features

On our campus, there is a lack of information on smaller restaurants and food locations. **SFStateEats** will be a service that allows its users to rate, view, and discover all food places on San Francisco State University Campus. It will also allow restaurant owners to add their restaurants to the platform.

The unique feature that our product offers is the ability to reply and leave comments to **reviews written by other users.** This will add an immense amount of user active time, since the review section of many restaurants will foster discussions. This is a great natural way to keep users on the website for longer.

This functionality also presents a benefit for the user. They are able to start discussions with someone who has left a review, and ask them to specify further what factors went into the decision of their review. For instance, if ‘Alice’ leaves a negative review of Taco Bell, ‘Bob’ is able to discuss this review with ‘Alice’ further. He may find that ‘Alice’ left a negative review due to the fact that her burger was cold. ‘Bob’ may then opt to go to the restaurants anyways, but not order a burger for himself.

URL to Product

http://3.12.102.223:4000/

Section 2: Usability Test Plan

Tasks

1) Write and view reviews

2) Reply to reviews

3) Apply filters to search

4) Add a restaurant

5) See hours of business for each restaurant

Test objectives:

In this test the users will test five main functions of the website. Users shall be able to write and view reviews, reply to reviews, apply filters to their search inquiry, add a new restaurant, and see hours of business for each restaurant. Writing and viewing reviews will be tested because that is one of the biggest functions in a website designated for food. Many users view reviews to see if they are interested in going to the restaurant or not. Next, we will test the function that will allow users to reply to other user's reviews. This will allow users to communicate about the restaurant and express their opinions. The third function that will be tested is the ability to apply filters to a search query because the users should be able to find the restaurants that match the specific needs of the user. Another function that we will be testing is to add a new restaurant. In this task the user will be able to add a new restaurant to the current list of restaurants, allowing other users to see information on this new restaurantFinally, the last function to be tested is the hours of operation. Users should be able to clearly see the hours of operation by clicking the Hours tab on the profile page of each restaurant.

System Setup:

In the system setup, we clone the entire project from GitHub repository, set up the databases, and start the server. Next, we will provide each user with a laptop with the updated Chrome Browser to open the website and start navigating.

Starting Point:

The starting point for all users will be the homepage, where users will be able to search for restaurants, login into their account, or sign up for a new account. Then users will be able to navigate through the website to write reviews, view reviews, reply reviews, apply search filters, and see for hours of operation for a certain restaurant.

Intended Users:

The SfStateEats website's main target user groups are Sf state students and faculty. This website will help new incoming students to find affordable, delicious, and great restaurants on campus. Students and faculty members will be able to find restaurants based on health needs, payment options, and price.

User Satisfaction Evaluation:

To measure user satisfaction, we will use Likert tests to analyze exactly where we need to improve our website. Users will be able to choose from strongly disagree to strongly agree on each task that they are asked to perform.

Usability Task description:

TASK	**DESCRIPTION**
Task #1	View reviews of a restaurant and then write a review.
Machine State	User is logged in.
Successful Completion Criteria	Successfully writes a review and sees other reviews.
Benchmark	Completed in **1 min.**

TASK	**DESCRIPTION**
Task #2	Reply to a review on a restaurant page.
Machine State	User is logged in.
Successful Completion Criteria	User clicks reply button and leaves a comment on another review
Benchmark	Completed in **1 min.**

TASK	**DESCRIPTION**
Task #3	Apply filters to search results.
Machine State	User is not logged in.
Successful Completion Criteria	User successfully chooses all filter to their query
Benchmark	Completed in **30 sec.**

TASK	**DESCRIPTION**
Task #4	Add a restaurant to the website.
Machine State	User logged into a business account.
Successful Completion Criteria	User successfully provides all details and presses the send button.
Benchmark	Completed in **30 sec.**

TASK	**DESCRIPTION**
Task #5	View hours of operation of a restaurant.
Machine State	User is logged in.
Successful Completion Criteria	User successfully sees the hours of operation of restaurant
Benchmark	Completed in **45 sec.**

- Usability Test Table

Test #1	**% completed**	**Errors**	**Comments**
User 1	**90 %**	**After I sign up for a new account, it does not take me back to the homepage.** **There should be an alert or pop up message i am not logged in, not another page.**	**The entire UI should be reevaluated. The beginning picture is too big, the profile page for restaurants has unnecessary big texts and slideshows. No filters for the reviews. Have thumbs up or down signs when writing reviews.**

Test #2	**% completed**	**Errors**	**Comments**
User 2	**85%**	**If I search for a restaurant and it is not listed in the restaurants, then there should be a message that tells me that restaurant does not exist.** **Filled in info for adding restaurant but it doesnt show me my restaurant on homepage.**	**Overall, pretty good but just change the looks of the website.**

Test #3	% completed	Errors	Comments
User 3	95%	Sign up page doesn't have ok alert. Comments take time to refresh.	Design needs changing.

Test #4	% completed	Errors	Comments
User 4	90%	Commenting on a review doesn't give proper feedback. Search result page is Blank when there is no result.	Redesign review interface.

Test #5	% completed	Errors	Comments
User 5	85%	Price filter not working. Posted review doesn't appear in the review section.	HomePage is too empty.

Test #6	% completed	Errors	Comments
User 6	80%	Images are too big Doesn't show if i am logged in or logged out	Home page is very blank

Test #7	% completed	Errors	Comments
User 7	75%	Cant see the menu Cant see the picture in profile page	Improve UI

Test #8	% completed	Errors	Comments
User 8	95%		Nice concept. Work on UI. seems very basic

Test #9	% completed	Errors	Comments
User 9	75%	Could not add my review Images are way too big and overpower the website.	Make small images

Test #10	**% completed**	**Errors**	**Comments**
User 10	**85%**	**Price filter not working.** **Posted review doesn't appear in the review section.**	**HomePage is too busy.**

- Questionnaire:

3 Likert scale questions were asked to the test subjects for evaluating user satisfaction.

S.No.	**Task**	**Strongly Disagree**	**Disagree**	**Neutral**	**Agree**	**Strongly Agree**
1.	I am able to navigate through the website easily.			3	4	2
2.	I thought there was too much inconsistency in this system.		3	5	1	
3.	The process of writing a new review is efficient.		3	3	3	

Section 3: QA Test Plan

- **Test objectives:** Password shall contain at least 1 number, 1 special character, 1 uppercase letter, and 1 lowercase letter
- **HW and SW setup:** http://3.12.102.223:4000/signup
    - Hardware Setup:
        - macOS Catalina v.10.15.1
        - Macbook Pro (Retina, Mid 2012)
        - 2.3 GHz Quad-Core Intel Core i7
        - 8 GB 1600 MHz DDR3
        - NVIDIA GeForce GT 650M 1 GB
    - Begins when user enters every valid field except for password and then submits the form
- **Feature to be tested:** Signup
- **QA Test plan:**

Number	Description	Test Input	Expected Output	Pass/Fail
1	Password without special character	“Lake1”	error: “missing special character”	Fail
2	Password without upper case letter	“lake1@”	error: “missing upper case”	Fail
3	Password without number	“Lake@”	error: “missing number”	Fail
4	Password containing number, special character, uppercase letter, and lowercase letter	“Lake1@”	message: “user successfully registered”	Pass

- **Test objectives:** Images uploaded shall be in the format of jpg, jpeg, and png
- **HW and SW setup:** http://3.12.102.223:4000/addrestaurant
    - Hardware Setup:
        - macOS Cataline v.10.15.1
        - Macbook Pro (Retina, Mid 2012)
        - 2.3 GHz Quad-Core Intel Core i7
        - 8 GB 1600 MHz DDR3
        - NVIDIA GeForce GT 650M 1 GB
    - Begins when business owner enters the add restaurant page and attaches an image to their restaurant submission
- **Feature to be tested:** Image Upload
- **QA Test plan:**

Number	Description	Test Input	Expected Output	Pass/Fail
1	Uploading a GIF image	test.gif	error: “Image needs to be in jpg, jpeg, or png format	Fail
2	Uploading a TIFF image	test.tiff	error: “Image needs to be in jpg, jpeg, or png format	Fail
3	Uploading a png image	test.png	message: “image successfully uploaded”	Pass

- **Test objectives:** Username shall be stored as alphanumeric characters
- **HW and SW setup:** http://3.12.102.223:4000/signup
    - Hardware Setup:
        - macOS Cataline v.10.15.1
        - Macbook Pro (Retina, Mid 2012)
        - 2.3 GHz Quad-Core Intel Core i7
        - 8 GB 1600 MHz DDR3
        - NVIDIA GeForce GT 650M 1 GB
    - Begins when a user needs an account and navigates to the signup link to fill out the username field

- **Feature to be tested:** Signup
- **QA Test plan:**

Number	Description	Test Input	Expected Output	Pass/Fail
1	Storing username with an “@” symbol	“Bob@”	error: “username can only be stored as alphanumeric characters”	Fail
2	Storing username with an “*” symbol	“B*ob”	error: “username can only be stored as alphanumeric characters”	Fail
3	Storing username with only alphanumeric characters	“Billy123”	message: “user successfully registered”	Pass

- **Test objectives:** Only authorized users with bearer tokens shall be able to leave reviews
- **HW and SW setup:** http://3.12.102.223:4000/review
    - Hardware Setup:
        - macOS Cataline v.10.15.1
        - Macbook Pro (Retina, Mid 2012)
        - 2.3 GHz Quad-Core Intel Core i7
        - 8 GB 1600 MHz DDR3
        - NVIDIA GeForce GT 650M 1 GB
    - Begins when a user navigates to a restaurant page to create a review and submits the review for posting
- **Feature to be tested:** Review
- **QA Test plan:**

Number	Description	Test Input	Expected Output	Pass/Fail
1	Not logged in user trying to post review	{bearer: null}	error: “user is not logged in”	Fail
2	User trying to post review with an invalid bearer token (modified)	{bearer: “NotValidToken 123”}	error: “user has an invalid bearer token	Fail
3	User posting review with a valid bearer token	{bearer: “ValidToken123 ”}	message: “user authorized for this route”	Pass

- **Test objectives:** Password shall be stored as an encrypted value
- **HW and SW setup:** http://3.12.102.223:4000/signup
    - Hardware Setup:
        - macOS Cataline v.10.15.1
        - Macbook Pro (Retina, Mid 2012)
        - 2.3 GHz Quad-Core Intel Core i7
        - 8 GB 1600 MHz DDR3
        - NVIDIA GeForce GT 650M 1 GB
    - Begins when a user successfully signs up their account and information is send to backend to be encrypted
- **Feature to be tested:** Signup
- **QA Test plan:**

Number	Description	Test Input	Expected Output	Pass/Fail
1	Storing password as encrypted value	“lake”	“$2y$12$j3G.cl13WHTKRjyt5MH7Ne8zsHQjqKl0HL7NEvOkNJmoEUXyYv./i”	Pass
2	Attempting to store password as original password	“lake”	“lake”	Fail
3	Storing password as encrypted value	“Lake!2@a”	“$2y$12$um0OUbEFZYX5donYdI3GfeRZMlDgR9uziSbKnV3lXOKSnf6OPkTRW”	Pass

Section 4: Code Review

Coding Style

Bracket Indentation Convention

For our application, we choose to go with the "the one true brace style" (abbreviated as OTBS). This style places the first curly bracket in the same line as the function name, and the last curly bracket on the line after the last line of code. Some advantages of this style are that the starting brace needs no extra line alone; and the ending brace lines up with the statement it conceptually belongs to. (Source: https://en.wikipedia.org/wiki/Indentation_style#Variant:_1TBS_(OTBS))

Variable Naming Convention

For our application, we choose to go with an underscore (_) delimiter for all variables. This means that when a variable consists of multiple words, each word is separated by an underscore. For instance, a variable that represents an account id would be called **account_id**.

Code Review Information

- Code Written By: Pedro Souto
- Code Reviewed By: John Pham
- Code Review Method: Email
- Code Description: This piece of code is written inside of a javascript file named ‘account.controller.js’. This controller is responsible for all interactions with the account table inside our database. It implements functions which create, delete, and modify accounts. This particular snippet, which is pasted below, was used as the basis for the code review.

Code Review Key

- ** Positive Comment**
- ** Negative Comment**

Code Reviewed Snippet

```
const bcrypt = require('bcrypt');
var jwt = require('jsonwebtoken');
const saltRounds = 10;
const {addUser, deleteUserByID} = require("./user.controller");
   **I like this, good idea!**
/**
* These are the STRING representation for the column names in our database to
* help prevent misspelling
*/
const TableColumnKey = {
   TABLE_NAME: "account",
   ACCCOUNT_ID: "account_id",
   USER_ID: "users_id",
   USERNAME: "username",
   PASSWORD: "password",
};
   **Good Header**
// ** REGISTER AN ACCOUNT **
// Request URL: http://localhost:3000/account/register
// Required key-value parameters in POST body: username, password, name, dob
const registerAccount = async (req, res) => {
   ** Extraneous Comment, Consider Removing?**
   // Check to make sure all required parameters passed in
   if (req.body.username && req.body.password && req.body.name && req.body.dob)
{
         // Before we can create an account, verify that the username is free
```

```
        if (await usernameIsFree(req) == 0) {
            console.log("Failed to register new account. Username taken.");
                  res.status(400).send({ error: "Failed to register new account.
Username taken."});
            return;
        }
        **Good Comment, Explains Database Relationships**
        // An account needs a user as a FK. User stores the personal information
such as name, dob, and photo
        // First create a user entry in the user table, for this specific
account
        **Variable Name Follows Convention**
        var user_id = await addUser(req);

        // Check to make sure user was added successfully
        if (user_id != -1) {
            hashed_password = bcrypt.hashSync(req.body.password, saltRounds);
            try {
                ** Extraneous Comment, Consider Removing?**
                // Attempt to insert account
                const response = await req.DATABASE_CLIENT.query(`
                  INSERT INTO ${TableColumnKey.TABLE_NAME}
(${TableColumnKey.USER_ID}, ${TableColumnKey.USERNAME},
${TableColumnKey.PASSWORD})
                  VALUES (${user_id}, '${req.body.username}',
'${hashed_password}')
                `);
                console.log("Successfully registered new account.");
                res.send({ message: "Successfully registered new account." });
                return;
              } catch (e) {
                // If account creation was unsuccessful, we should delete the
user we just made as well
                req.body.id = user_id;
                await deleteUserByID(req);
                console.log(e);
                console.log("Failed to register new account.");
```

```
                res.status(500).send({ error: "Failed to register new
account."});
                return;
            }
        } else {
            console.log("Failed to add new user before account.");
            res.status(500).send({ error: "Failed to add new user before
account."});
            return;
        }
    } else {
        console.log("Failed to register new account. Missing Parameters.");
        res.status(400).send({ error: "Failed to register new account. Missing
Parameters."});
        return;
    }
};
```

Transcript of Email - Summary of Comments

Positives

- All variable names follow the agreed upon naming convention.
- All brackets follow the agreed upon indentation convention.
- Database object at the top of the file is very smart. It allows for quick change of names, without changing every instance of that table row inside all queries.
- Plenty of comments, especially in the confusing parts of the code.
- Functions have headers, which discusses both what the function does and what the function requires (parameters).

Negatives

- Although comments are great, there are a few comments that seem redundant and extra. Not every line needs a comment, only those sections or lines of code in which another programmer might be confused with.

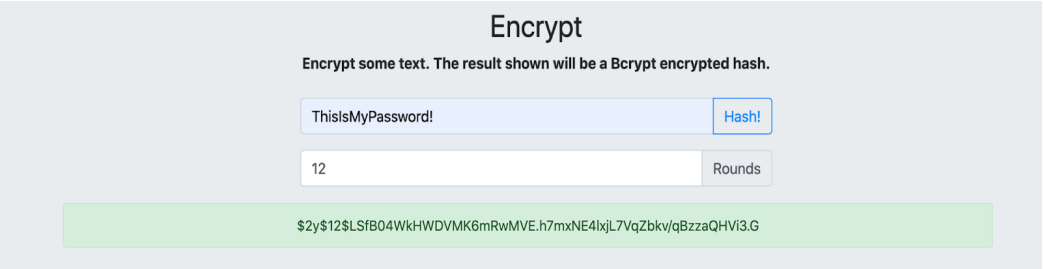
Section 5: Self-check: Best Practices for Security

List of Major Assets being Protected

- User Password
    - User passwords are being encrypted using the Bcrypt blowfish hashing algorithm. This will ensure that even after a security breach, hackers would not be able to login to any accounts.
- Uploaded Images
    - All uploaded images will be stored in an AWS service called S3. S3 is a bucket to store anything from images, to videos, to files. We will use it to store all media on our website. This is being protected via permissions, which are set up inside the AWS console. Only whitelisted IP addresses are able to access these images.
- Database Information
    - The database is being stored in an AWS service called RDS. RDS is a Relational Database System which stores a variety of SQL databases. We are using this service to store our PostgreSQL instance. By hosting this instance on AWS, we are able to leverage their experience and security. Via the AWS console, we are able to set permissions and whitelist specific IP addresses. By doing this, we ensure that even during a security breach, the only entity able to access these records is our server, thus reducing the potential damage.

Confirmation of Password Encryption

All user passwords are being encrypted using the open source node library called **'bcrypt'**. Bcrypt uses a variant of the Blowfish encryption algorithm's keying schedule, and introduces a work factor, which allows you to determine how expensive the hash function will be. Because of this, bcrypt can keep up with Moore's law. As computers get faster you can increase the work factor and the hash will get slower (Source: https://codahale.com/how-to-safely-store-a-password)



Above is an example of a password **'ThisIsMyPassword!'** which has been hashed using 12 salt rounds. A salt round determines how much work is needed to decrypt the password, and a common standard today is 12. Over time however, this number will need to increase due to computing capabilities increasing. The hash for this particular password is: **'$2y$12$LSfB04WkHWDVMK6mRwMVE.h7mxNE4lxjL7VqZbkv/qBzzaQHVi3.G'**.

```
hashed_password = bcrypt.hashSync(req.body.password, saltRounds);
```

Above is the actual hashing implementation in our code. This particular piece of code is used in the register function. This line hashes the password, and uses the **hashed password** to create an account for the user which is stored in the database.

```
var validPassword = bcrypt.compareSync(req.body.password,
      response.rows[0].password);
```

Above is the actual decryption implementation in our code. This particular piece of code is used in the login function. This line compares the password the user is entering, with the hashed password stored in the database. If the hash of the given password is equal to the stored hash in the database, then the password is valid. When the password is valid, this function returns a boolean of ‘True’, which allows for an easy comparison without **ever** storing the unhashed password of the user.

Confirmation of Input Validation

All inputs are first validated before being sent to an endpoint. This is done via a middleware. A middleware is a function that goes between an HTTP(S) endpoint, and the function behind that endpoint. This ensures that before any functions are called, all information being sent to that endpoint is first validated. This is beneficial in 2 ways. First, if a request is not valid due to improper input, the server does not need to waste resources executing commands, only to find out later that there was an error with the input. Secondly, it allows each endpoint to assume that all data is valid, which reduces copied code.

Account Details Validation

```
app.use((req, res, next) => {
 if (req.cookies['Token']) {
   var decoded = jwt.verify(req.cookies['Token'],
'csc648CookiesSecret');
   req.account_id = decoded.account_id;
   console.log("Call made by user: ", req.account_id);
 }
 next();
});
```

The above is the middleware function responsible for validating tokens (which is a user input that corresponds to identification). Before **any** endpoint is called, this middleware first ensures that the cookie is valid. If a cookie is not valid, it stops execution of that endpoint, thus not wasting any resources. If a cookie is valid however, this endpoint attaches the account_id to the request. This allows **every** endpoint to assume that account_id is valid!

Endpoint Parameters Validation

```
const registerAccount = async (req, res) => {
   // Check to make sure all required parameters passed in
   if (req.body.username && req.body.password && req.body.name &&
req.body.dob) {
       // CODE GOES HERE
}
```

The above is a sample endpoint. Every endpoint implements similar checks, to ensure that all pieces of data it needs were sent in the HTTP call. If a user makes a request to an endpoint, but does not send the correct information, then that request is simply ignored and does not cause a waste of resources on the server.

Section 6: Self-check: Adherence to Original Non-functional Specs

Progress Key

- DONE
- ISSUE
- ON TRACK

Security

- Login shall be required to leave reviews.
- Login shall be required to create a business listing.
- A gateway service will redirect traffic to other services to prevent the discovery of the end URL.
- Email shall not be from a temporary email host provider.
- Passwords shall contain at least 1 number, 1 special character, 1 uppercase letter, and 1 lowercase letter.
- Users shall be required to change password every 12 months.
- Change of password shall require email verification.
- Users shall have 5 login attempts before account lockout.
- Account unlock shall require email verification before unlocking.

Audit

- New business listings shall be approved by the system administrator.
- Change in business owner shall be approved by the system administrator.
- Flagged reviews shall be reviewed by system administrators.
- Inappropriate reviews shall be removed by the system administrator.

Performance

- Each service should be hosted in its own server to prevent overwhelming one server.
- The web application should restart if it is abruptly shut down to prevent downtime.
- System shall respond visually within 5 seconds.
- Animations shall be kept to a minimum to accommodate for slow performing devices.

Data Integrity

- Images shall only be in the format of jpg, jpeg, and png.
- Images shall be saved on the server.
- Images uploaded shall be at most 2mb.
- Images shall be saved in the original size.
- Images shall be resized and displayed via CSS formatting, thus avoiding displaying and resizing after.
- Reviews shall not consist of special characters or emojis.
- Display name shall not consist of special characters or emojis.
- Email shall not consist of emojis.
- Passwords shall not consist of emojis.
- Databases shall be backed up every 24 hours.
- Databases shall be able to be backed up on command by a system administrator.

Compatibility

- The site shall be compatible for all mobile screen sizes.
- The site shall be compatible for all monitor screen sizes.
- The site shall be able to scale to high resolution.
- The site shall be compatible for Safari on version 12.1.2.
- The site shall be compatible for Firefox on version 73.0.
- The site shall be compatible for Chrome on version 80.0.3987.106.

Conformance with Coding Standards

- The whole production cycle of the site shall be finished in at least 5 days before the delivery date.
- Development console logging shall be disabled for production.
- The DOM tree shall have meaningful semantics for every element.
- Components shall be created to allow for reusable code.

Look and Feel Standards

- The site shall have an interface that is intuitive to navigate at first glance.
- The site shall use clean fonts and colors.
- The site shall meet modern design standards.

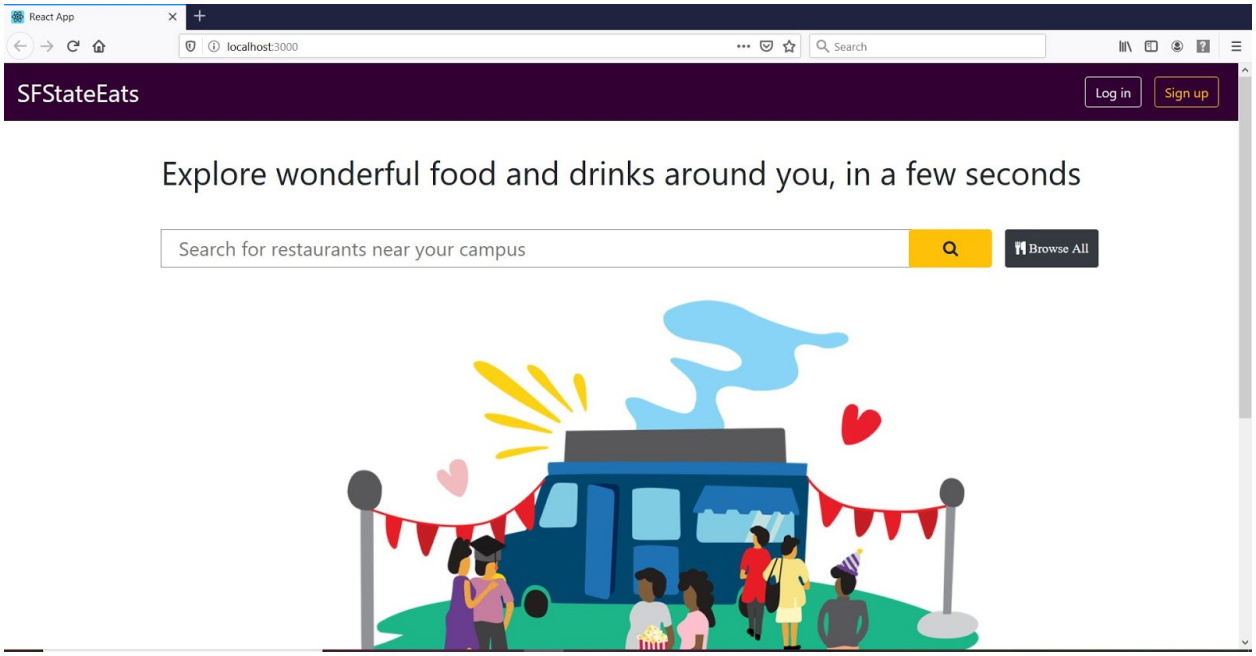
Internalization / Localization Requirements

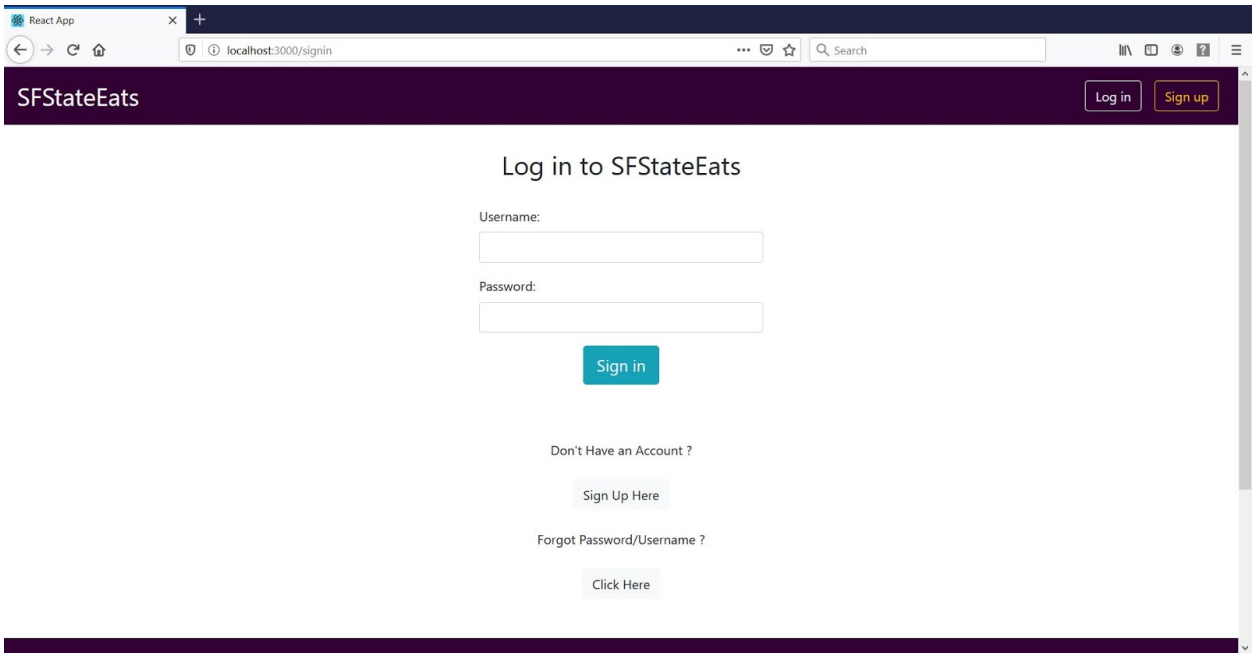
- The site shall be internationalized in the English language

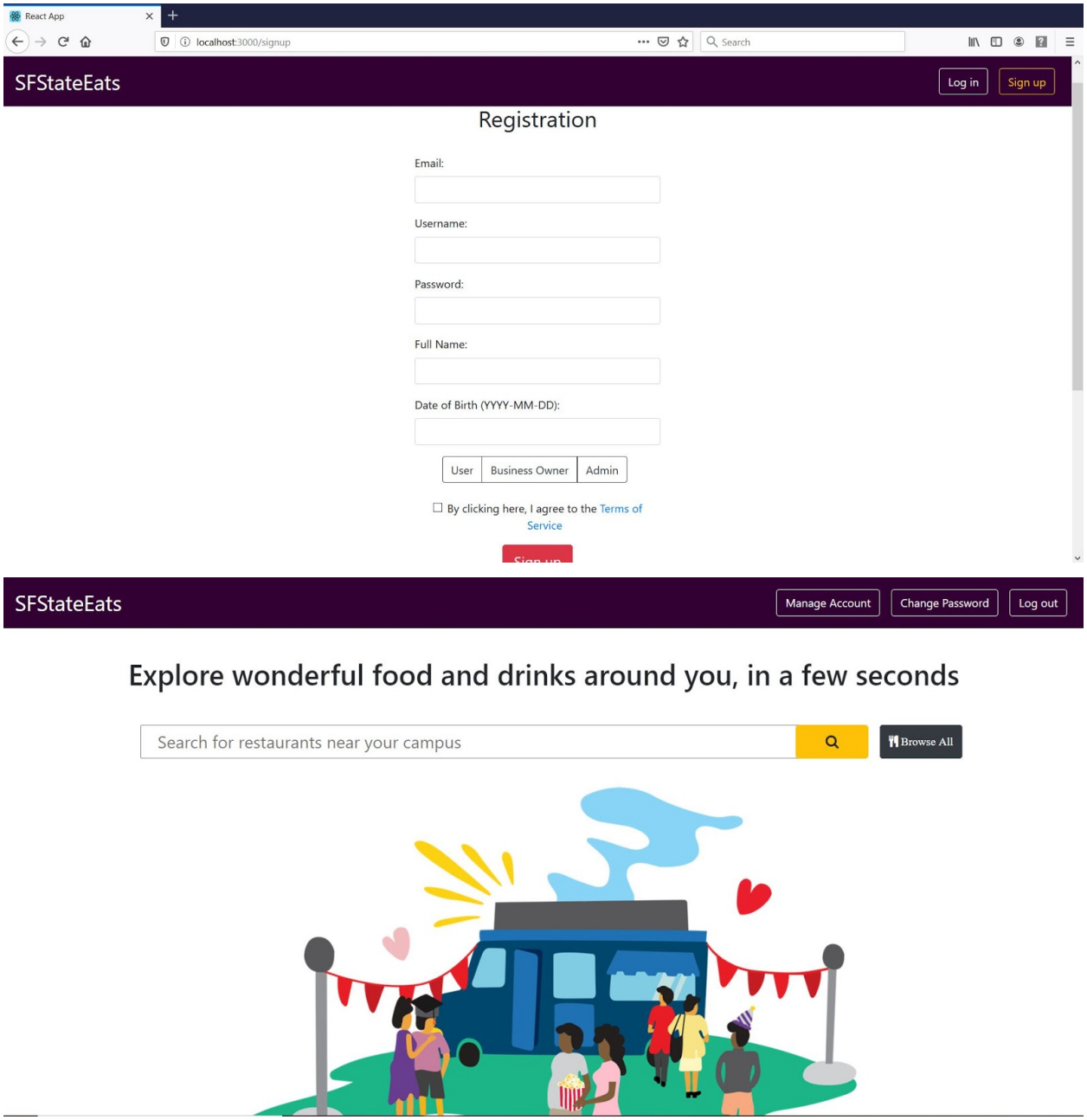
Web Site Policies

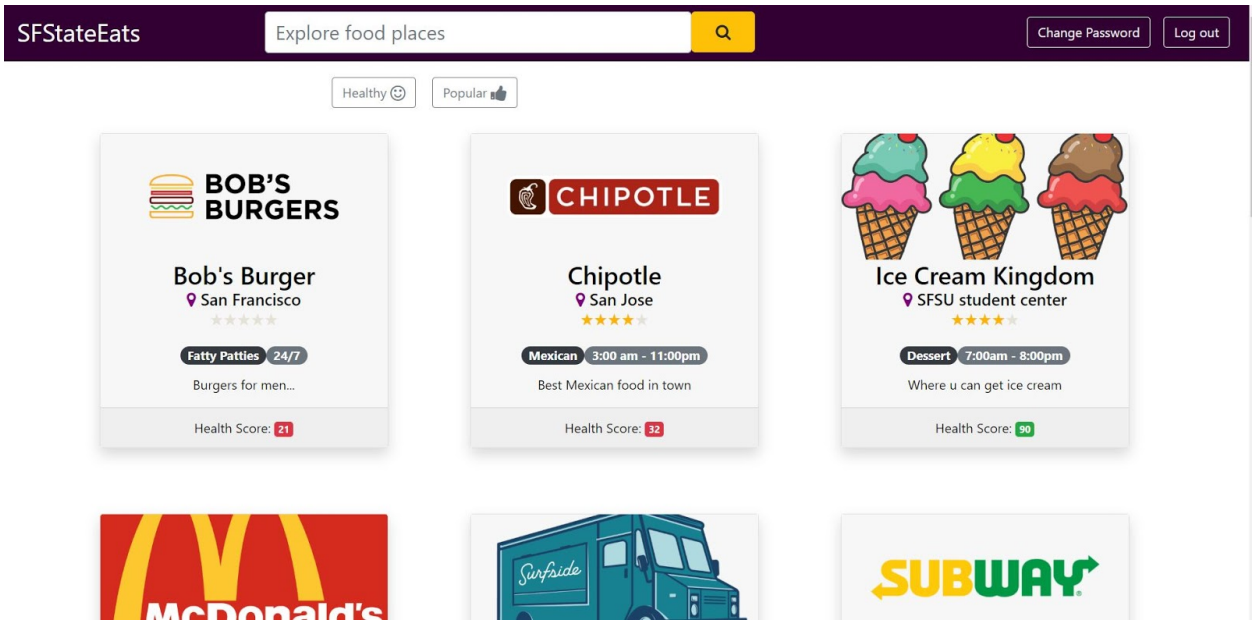
- The site shall not allow illegal content.
- The site shall not allow harassment of other users.
- The site shall not allow nudity.
- The site shall not allow business owners to review their own site.
- The site shall not allow self-promotion or malicious links.
- Reviews and replies are subject to the website's community guidelines.

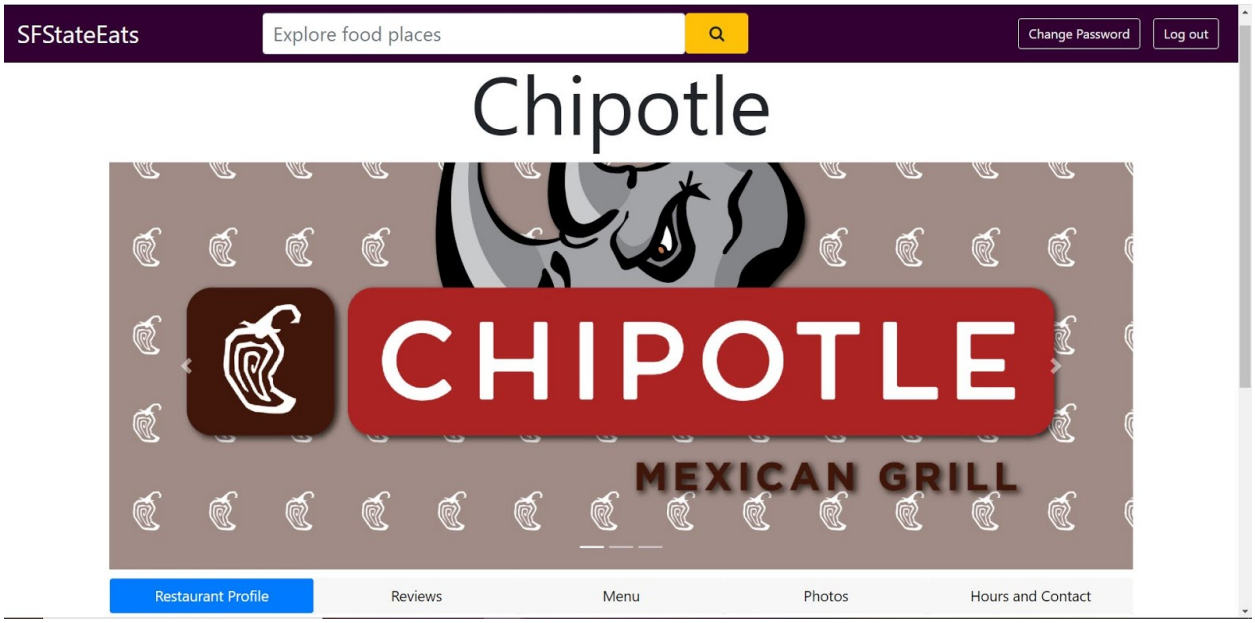
Section 4: Screenshots of Final Product

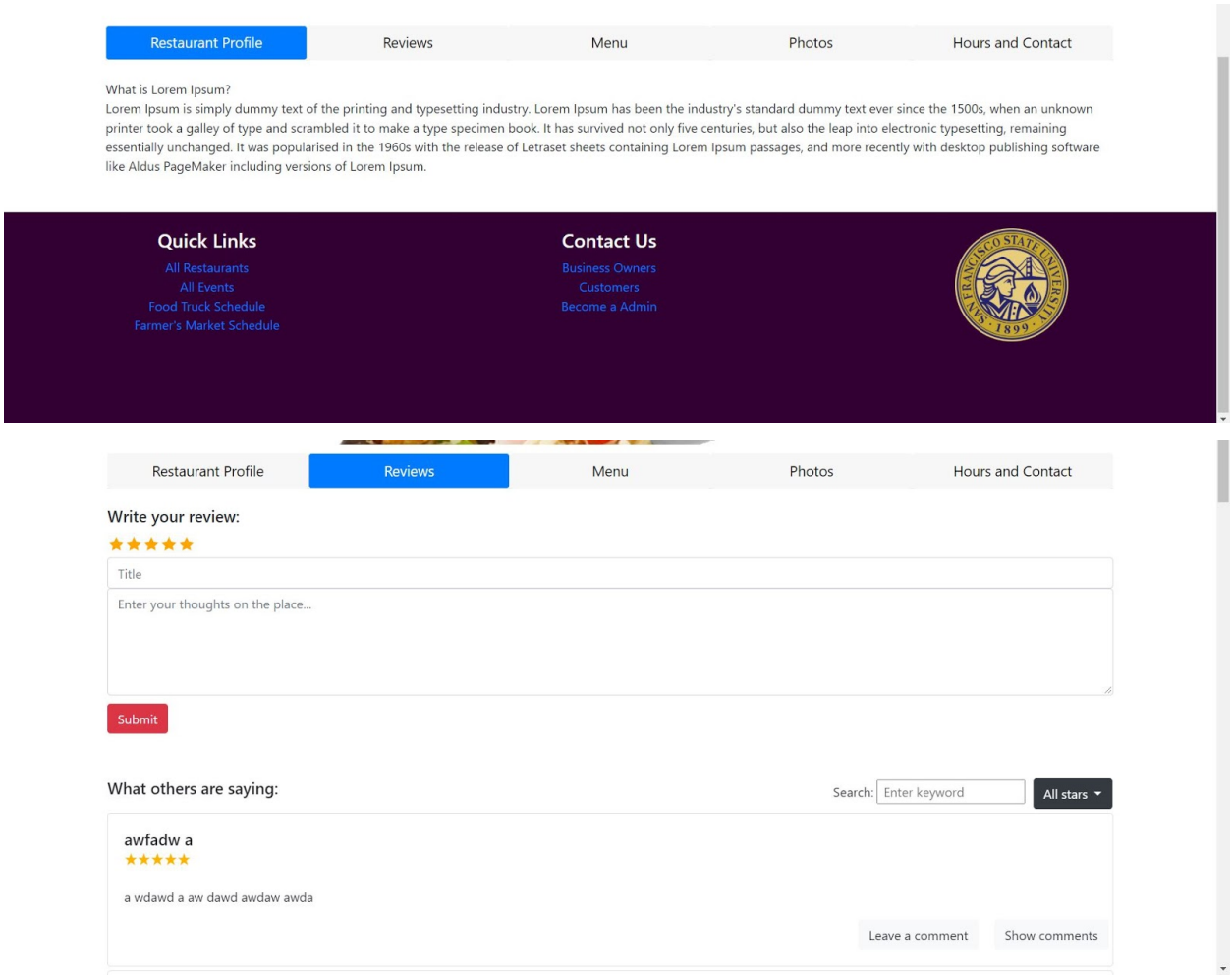


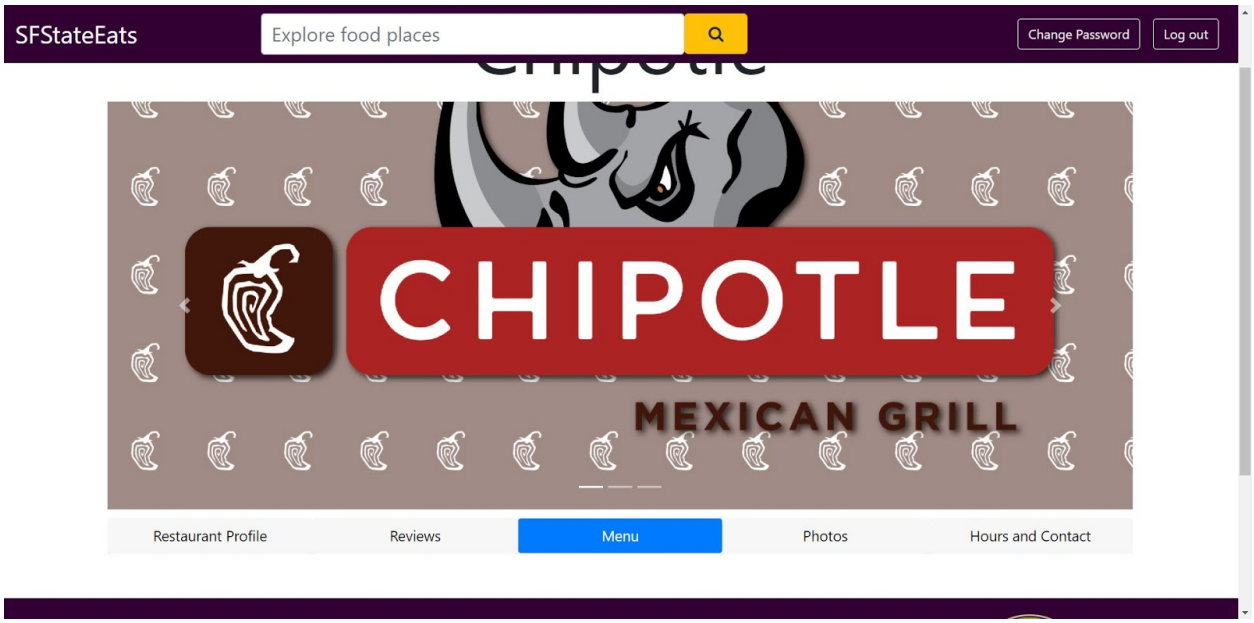


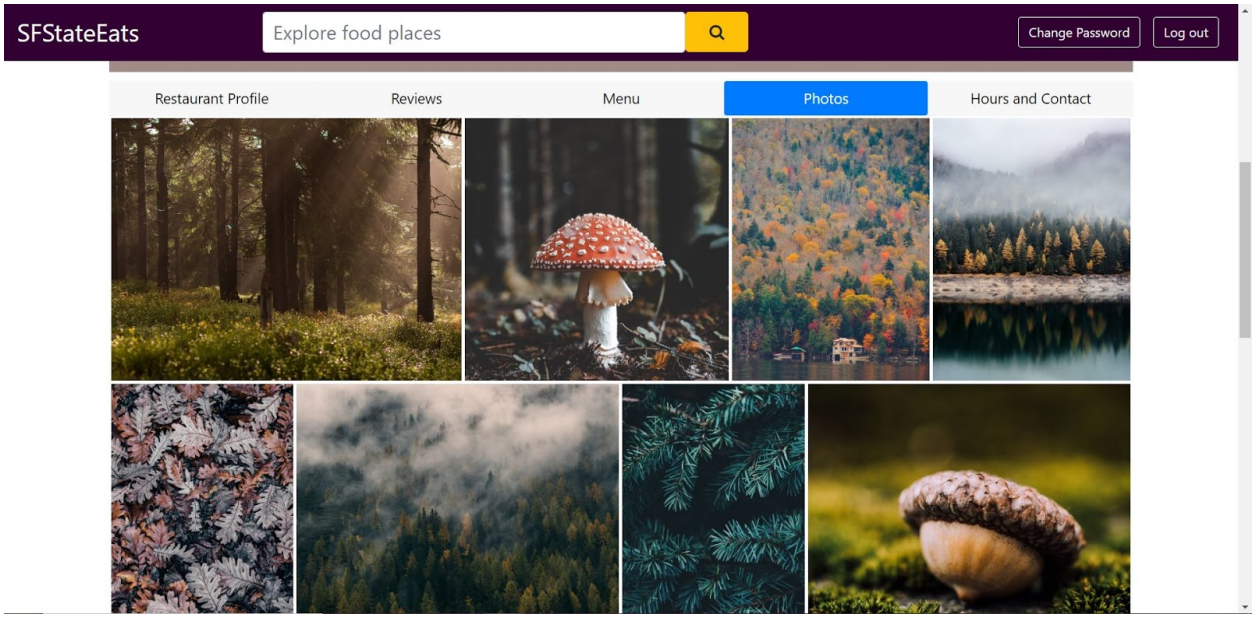

React App
localhost:3000/signup
Search
SFStateEats
Log in
Sign up
Registration
Email:
Username:
Password:
Full Name:
Date of Birth (YYYY-MM-DD):
User
Business Owner
Admin
By clicking here, I agree to the Terms of Service
Sign up
SFStateEats
Manage Account
Change Password
Log out
Explore wonderful food and drinks around you, in a few seconds
Search for restaurants near your campus
Browse All

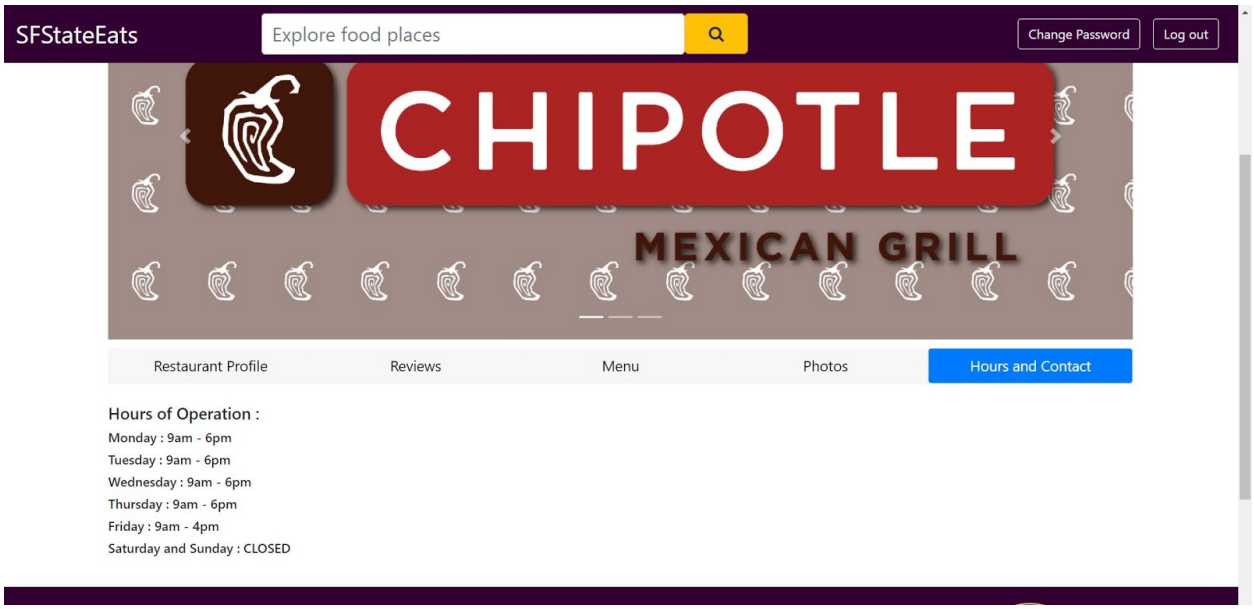

SFStateEats
Explore food places
Change Password
Log out
Healthy
Popular
BOB'S BURGERS
Bob's Burger
San Francisco
Fatty Patties
24/7
Burgers for men...
Health Score: 21
CHIPOTLE
Chipotle
San Jose
Mexican
3:00 am - 11:00pm
Best Mexican food in town
Health Score: 32
Ice Cream Kingdom
SFSU student center
Dessert
7:00am - 8:00pm
Where u can get ice cream
Health Score: 90
Surfside
SUBWAY

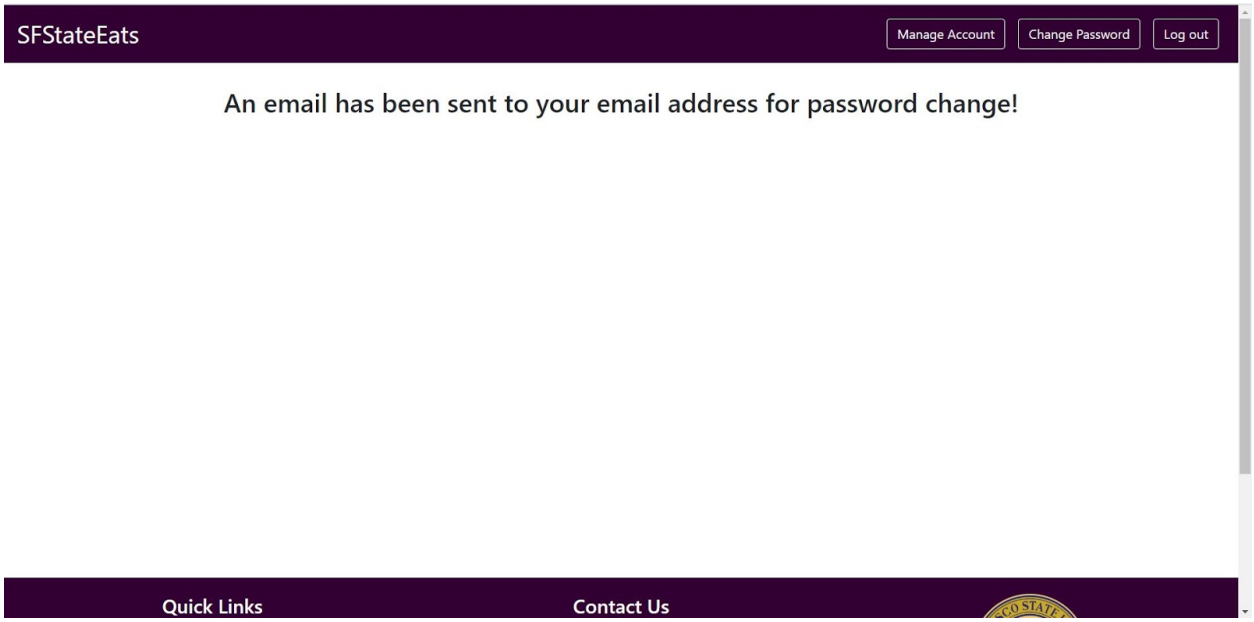

SFStateEats
Explore food places
Change Password
Log out
Chipotle
CHIPOTLE
MEXICAN GRILL
Restaurant Profile
Reviews
Menu
Photos
Hours and Contact

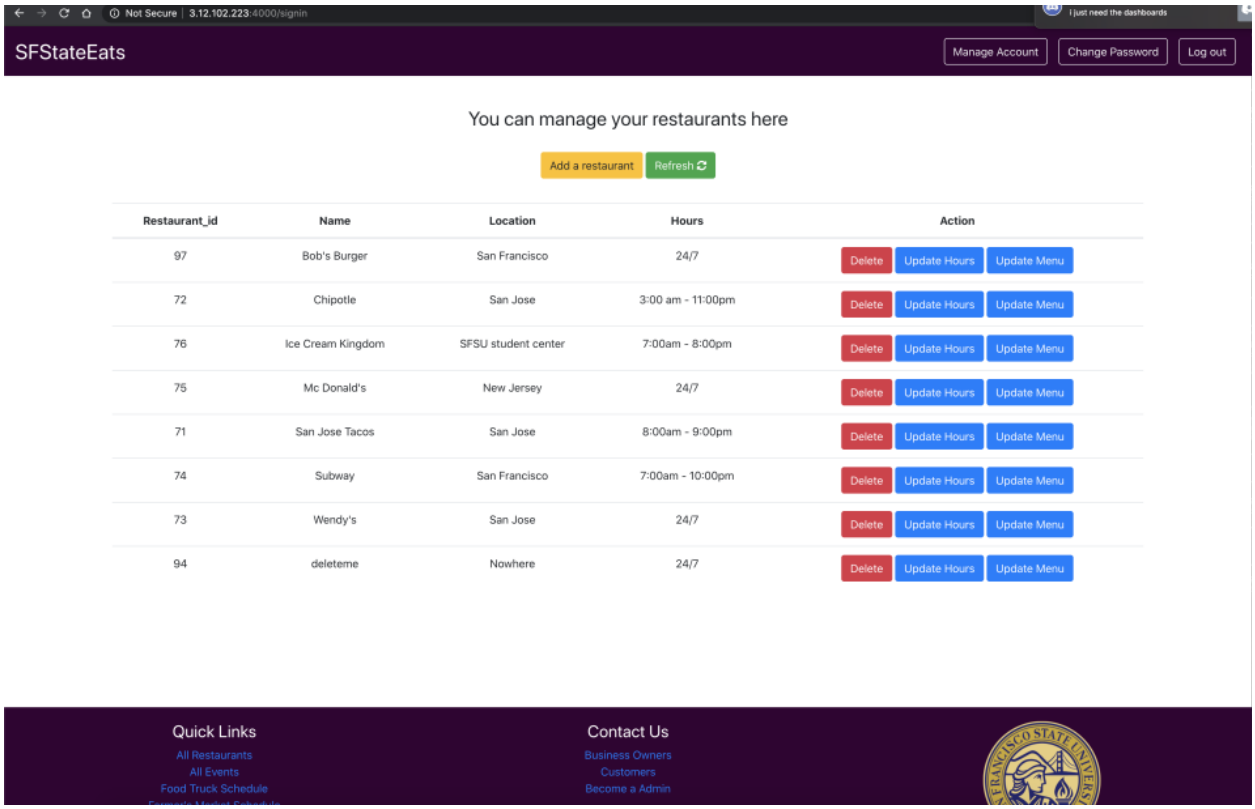

Restaurant Profile
Reviews
Menu
Photos
Hours and Contact
What is Lorem Ipsum?
Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.
Quick Links
All Restaurants
All Events
Food Truck Schedule
Farmer's Market Schedule
Contact Us
Business Owners
Customers
Become a Admin
Restaurant Profile
Reviews
Menu
Photos
Hours and Contact
Write your review:
Title
Enter your thoughts on the place...
Submit
What others are saying:
Search:
Enter keyword
All stars
awfadw a
a wdawd a aw dawd awdaw awda
Leave a comment
Show comments

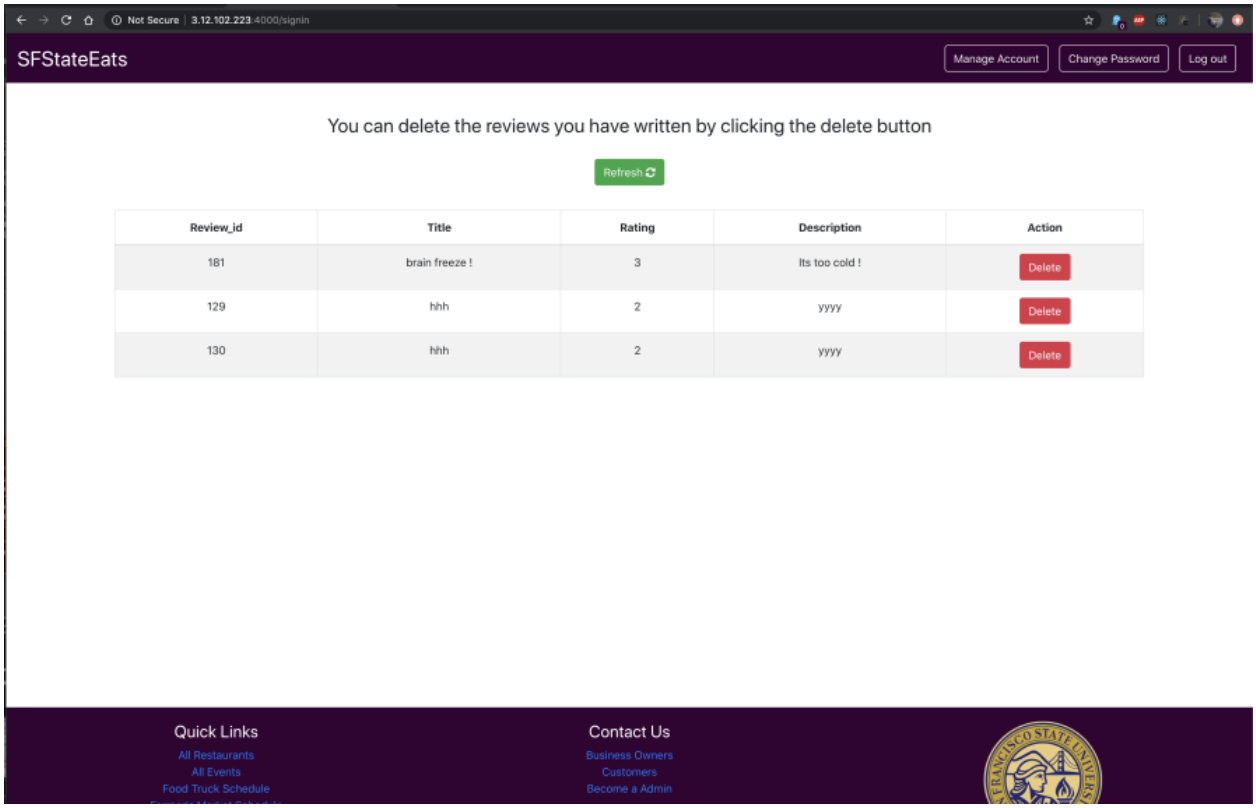

SFStateEats
Explore food places
Change Password
Log out
CHIPOTLE
MEXICAN GRILL
Restaurant Profile
Reviews
Menu
Photos
Hours and Contact

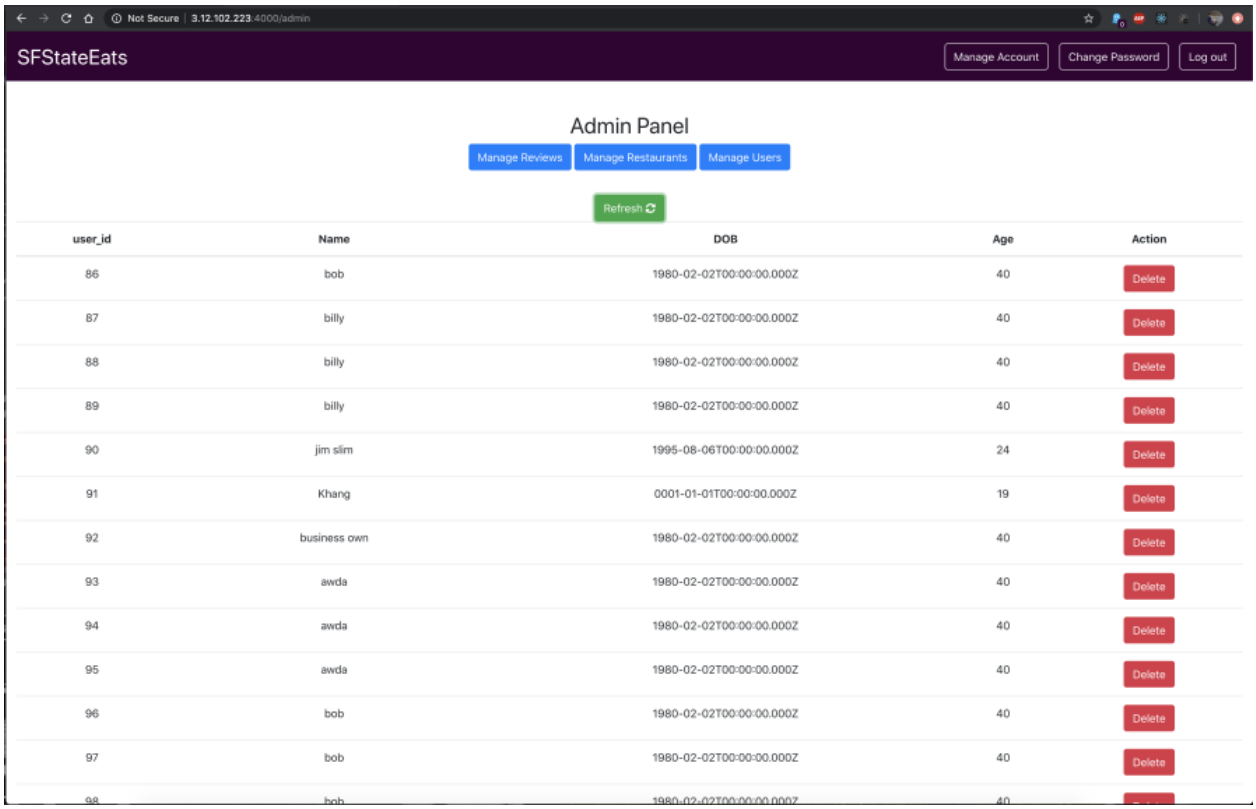

SFStateEats
Explore food places
Change Password
Log out
Restaurant Profile
Reviews
Menu
Photos
Hours and Contact


SFStateEats
Explore food places
Change Password
Log out
CHIPOTLE
MEXICAN GRILL
Restaurant Profile
Reviews
Menu
Photos
Hours and Contact
Hours of Operation :
Monday : 9am - 6pm
Tuesday : 9am - 6pm
Wednesday : 9am - 6pm
Thursday : 9am - 6pm
Friday : 9am - 4pm
Saturday and Sunday : CLOSED


SFStateEats
Manage Account
Change Password
Log out
An email has been sent to your email address for password change!
Quick Links
Contact Us


Not Secure | 3.12.102.223:4000/signin
SFStateEats
Manage Account
Change Password
Log out
You can manage your restaurants here
Add a restaurant
Refresh
Restaurant_id
Name
Location
Hours
Action
97 Bob's Burger San Francisco 24/7
72 Chipotle San Jose 3:00 am - 11:00pm
76 Ice Cream Kingdom SFSU student center 7:00am - 8:00pm
75 Mc Donald's New Jersey 24/7
71 San Jose Tacos San Jose 8:00am - 9:00pm
74 Subway San Francisco 7:00am - 10:00pm
73 Wendy's San Jose 24/7
94 deleteme Nowhere 24/7
Delete
Update Hours
Update Menu
Quick Links
All Restaurants
All Events
Food Truck Schedule
Contact Us
Business Owners
Customers
Become a Admin


Not Secure 3.12.102.223:4000/signin
SFStateEats
Manage Account
Change Password
Log out
You can delete the reviews you have written by clicking the delete button
Refresh
Review_id
Title
Rating
Description
Action
181
brain freeze !
3
Its too cold !
Delete
129
hhh
2
yyyy
Delete
130
hhh
2
yyyy
Delete
Quick Links
All Restaurants
All Events
Food Truck Schedule
Contact Us
Business Owners
Customers
Become a Admin


Not Secure 3.12.102.223:4000/admin
SFStateEats
Manage Account
Change Password
Log out
Admin Panel
Manage Reviews
Manage Restaurants
Manage Users
Refresh
user_id
Name
DOB
Age
Action
86
bob
1980-02-02T00:00:00.000Z
40
Delete
87
billy
1980-02-02T00:00:00.000Z
40
Delete
88
billy
1980-02-02T00:00:00.000Z
40
Delete
89
billy
1980-02-02T00:00:00.000Z
40
Delete
90
jim slim
1995-08-06T00:00:00.000Z
24
Delete
91
Khang
0001-01-01T00:00:00.000Z
19
Delete
92
business own
1980-02-02T00:00:00.000Z
40
Delete
93
awda
1980-02-02T00:00:00.000Z
40
Delete
94
awda
1980-02-02T00:00:00.000Z
40
Delete
95
awda
1980-02-02T00:00:00.000Z
40
Delete
96
bob
1980-02-02T00:00:00.000Z
40
Delete
97
bob
1980-02-02T00:00:00.000Z
40
Delete

Section 5: Screenshots of key DB Tables

Account

Data Output | Explain | Messages | Notifications

	account_id [PK] integer	users_id integer	username character varying (255)	password character varying (255)	email character varying (255)	expired_password boolean
1	63	86	bob	$2b$10$8EnwroeG.7skK8jJYy...	lolcake1000@hotmail.com	false
2	64	87	businessOwner1	$2b$10$Fqjh0ovhaEPrrwPBL...	bsoeseblkjekj@hotmail.com	false
3	65	88	notBusinessOwner1	$2b$10$CrE8DIydCLIFHjUoLG...	bsoeseblkjekj@hotmail.com	false
4	66	89	notBusinessOwner2	$2b$10$U64/2U17PMkxLpEv...	bsoeseblkjekj@hotmail.com	false
5	68	91	test	$2b$10$MHmtCVgmtBjKvbvU...	percytran96@gmail.com	false
6	67	90	jim	$2b$10$k7S8JOywqfCtxc.Pd...	jim1@gmail.com	false
7	69	92	businessOwner1me	$2b$10$IqaYlT26svU39xiKeia...	awandkisbzckizs@hotmail.com	false

Business Account

Data Output | Explain | Messages | Notifications

	business_account_id [PK] integer	account_id integer
1	23	63
2	26	64
3	27	71
4	28	73
5	29	74
6	30	76
7	31	79

Comment

Data Output | Explain | Messages | Notifications

	comment_id [PK] integer	account_id integer	comment character varying (255)
1	69	63	This is a comment 6
2	70	63	This is a comment 7
3	71	63	This is a comment 8
4	72	63	This is a comment 9
5	73	63	This is a comment 10

Comment Review

Data Output | Explain | Messages | Notifications

	comment_review_id [PK] integer	comment_id integer	review_id integer
1	57	69	126
2	58	70	126
3	59	71	126
4	60	72	126
5	61	73	126

Food

Data Output | Explain | Messages | Notifications

	food_id [PK] integer	name character varying (255)	price numeric	menu_id integer
1	4	Mc Griddle	2.99	1
2	5	Mc Muffin	1.99	1
3	1	Big Mac	7.99	2
4	2	Happy Meal	4.50	2
5	3	Quarter Pounder	8.50	2
6	6	Small Soda	2.50	2
7	7	Mc Flurry	2.55	3

Menu

Data Output | Explain | Messages | Notifications

	menu_id [PK] integer	restaurant_id integer	name character varying (255)
1	3	75	Dessert Menu
2	1	75	Breakfast Menu
3	2	75	Lunch and Dinner Menu

Registered User

Data Output | Explain | Messages | Notifications

	registered_user_id [PK] integer	account_id integer
1	1	65
2	2	66
3	3	69
4	4	70
5	5	72
6	6	75
7	7	77

Restaurant

Data Output | Explain | Messages | Notifications

	restaurant_id [PK] integer	name character varying (255)	description character varying (255)	health_score integer	popularity integer	open_hours character varying (100)
1	76	Ice Cream Kingdom	Where u can get ice cream	90	0	7:00am - 8:00pm
2	71	San Jose Tacos	This is a restaurant in san jose	73	10	8:00am - 9:00pm
3	73	Wendy's	Square patties is our thing!	80	40	1:00am - 2:00am
4	74	Subway	This is a nice restaurant.	82	60	7:00am - 10:00pm
5	75	Mc Donald	So unhealthy	42	90	24/7
6	72	Chipotle 2	Best Mexican food in town	32	20	3:00 am - 5:00pm

Data Output | Explain | Messages | Notifications

tags character varying[] (100)	expired timestamp with time zone	main_photo character varying[] (1000)	photos character varying[] (1000)	location character varyin
{"Ice Cream",Desert}	[null]	{https://sfsufinalprojectteam.s3....	{https://sfsufinalprojectteam.s3....	SFSU studen
{"Fast Food"}	[null]	{https://sfsufinalprojectteam.s3....	{https://sfsufinalprojectteam.s3....	San Jose
{Special,Good,Wow!}	[null]	{https://sfsufinalprojectteam.s3....	{https://sfsufinalprojectteam.s3....	San Jose
{"Nice View"}	[null]	{https://sfsufinalprojectteam.s3....	{https://sfsufinalprojectteam.s3....	San Francisc
{"Fast Food"}	[null]	{https://sfsufinalprojectteam.s3....	{https://sfsufinalprojectteam.s3....	New Jersey
{Cheap,Mexican}	[null]	{https://sfsufinalprojectteam.s3....	{https://sfsufinalprojectteam.s3....	San Jose

Review

Data Output | Explain | Messages | Notifications

	review_id [PK] integer	reviewer_id integer	restaurant_id integer	stars integer	title character varying (255)	description character varying (300)	flagged boolean
1	98	75	72	5	sdbsj sbs dlb sd bk	l lkd awld alkwdawd	false
2	99	75	72	5	sdbsj sbs dlb sd bk	l lkd awld alkwdawd as as	false
3	100	75	72	5	sdbsj sbs dlb sd bkas a	l lkd awld alkwdawd as as	false
4	101	75	72	5	sdbsj sbs dlb sd bkas a	l lkd awld alkwdawd as as as a	false
5	104	75	72	5	awfadw a	a wdawd a aw dawd	false
6	105	75	72	5	awfadw a	a wdawd a aw dawd awdaw	false
7	106	75	72	5	awfadw a	a wdawd a aw dawd awdaw a...	false

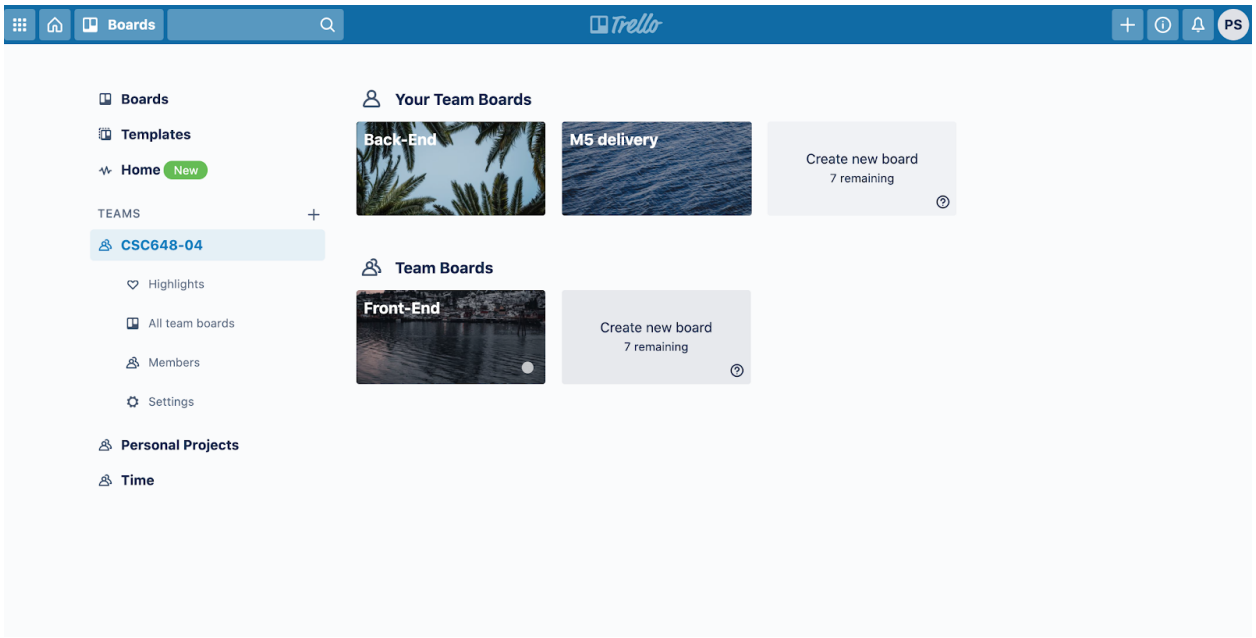
User

Data Output | Explain | Messages | Notifications

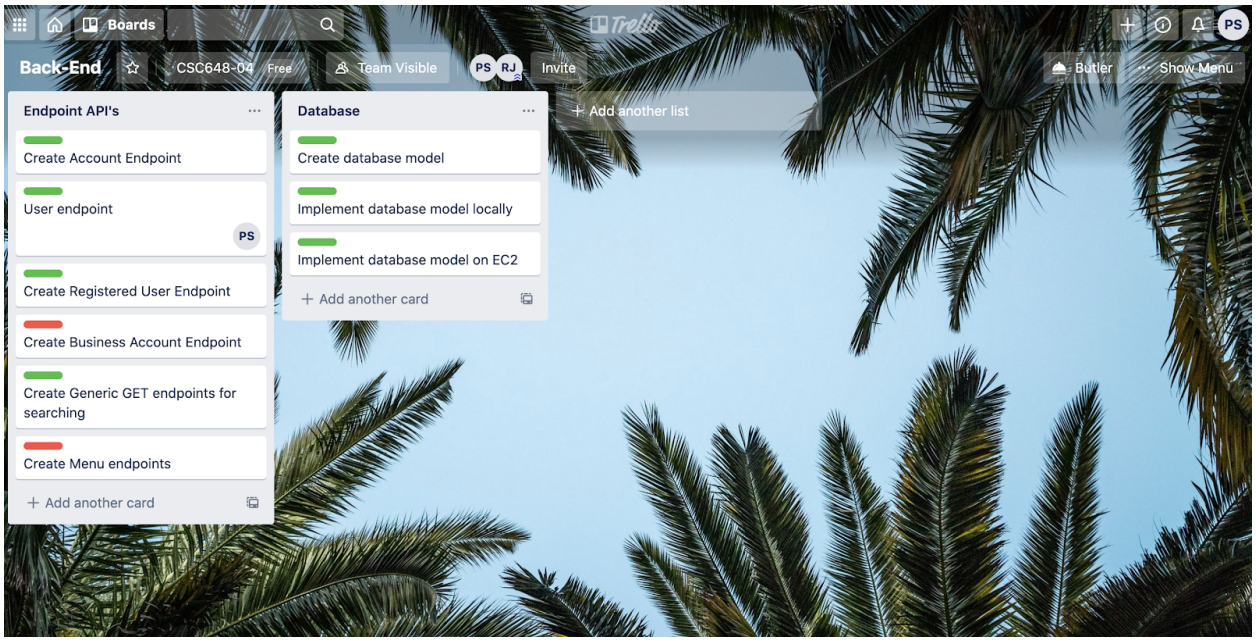
	users_id [PK] integer	name character varying (255)	dob date	age integer	photo character varying (255)
1	86	bob	1980-02...	40	[null]
2	87	billy	1980-02...	40	[null]
3	88	billy	1980-02...	40	[null]
4	89	billy	1980-02...	40	[null]
5	90	jim slim	1995-08...	24	[null]
6	91	Khang	0001-01...	19	[null]
7	92	business own	1980-02...	40	[null]

Section 6: Screenshots of Trello

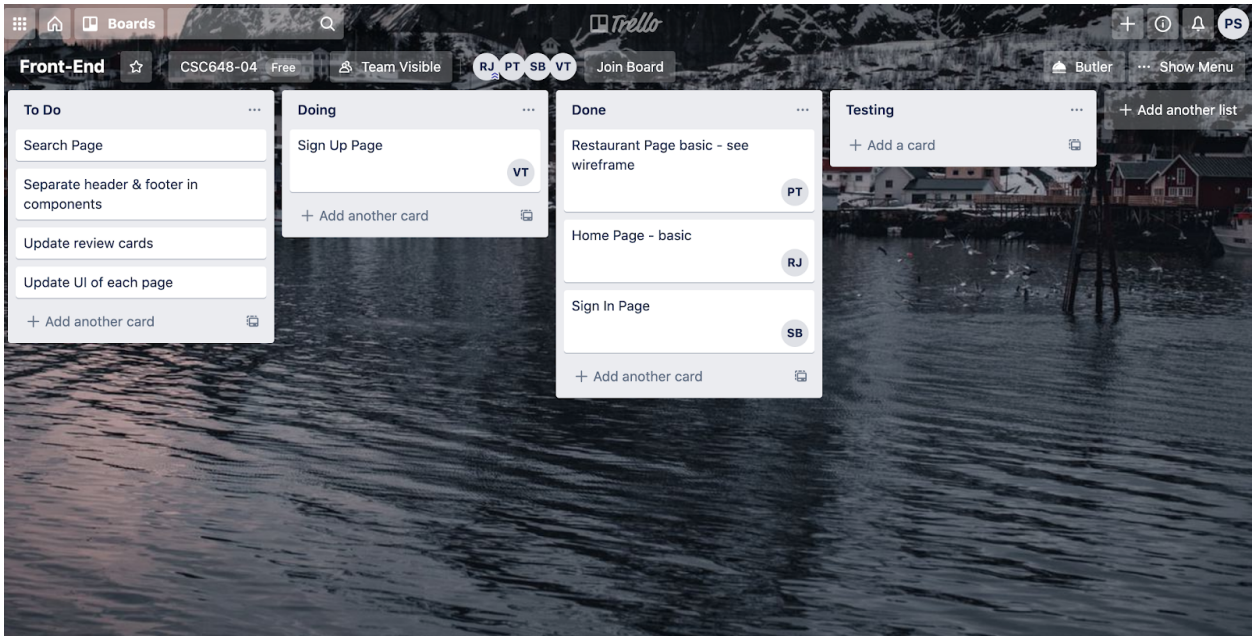
Trello Home (All Boards)



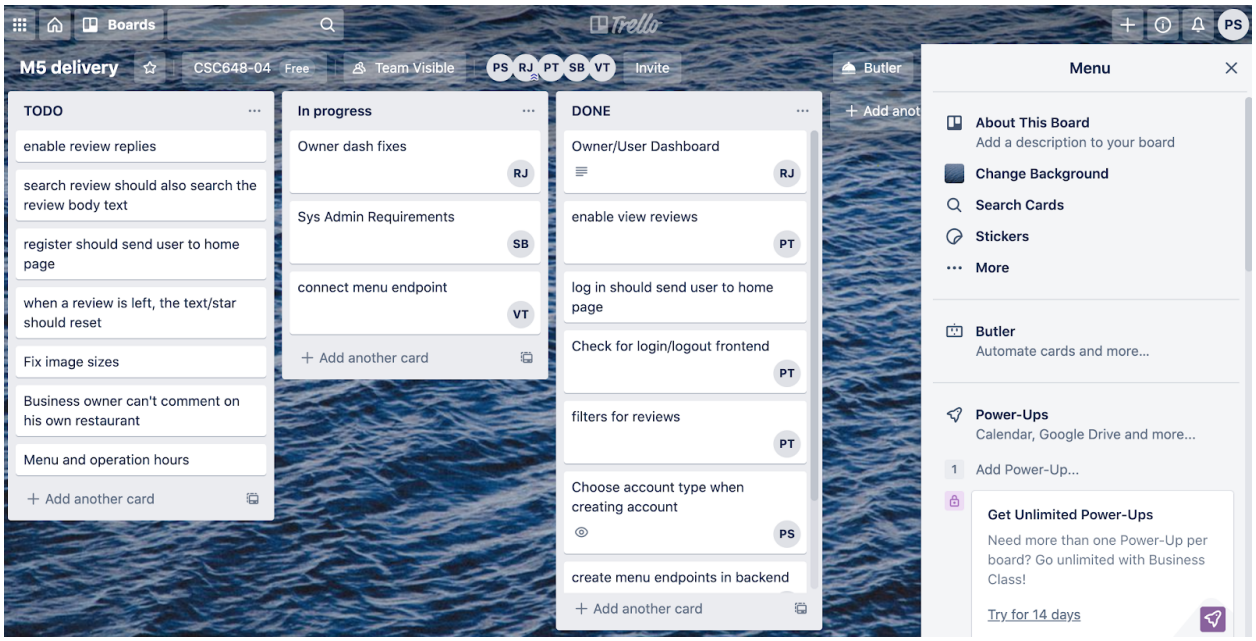
Backend Board



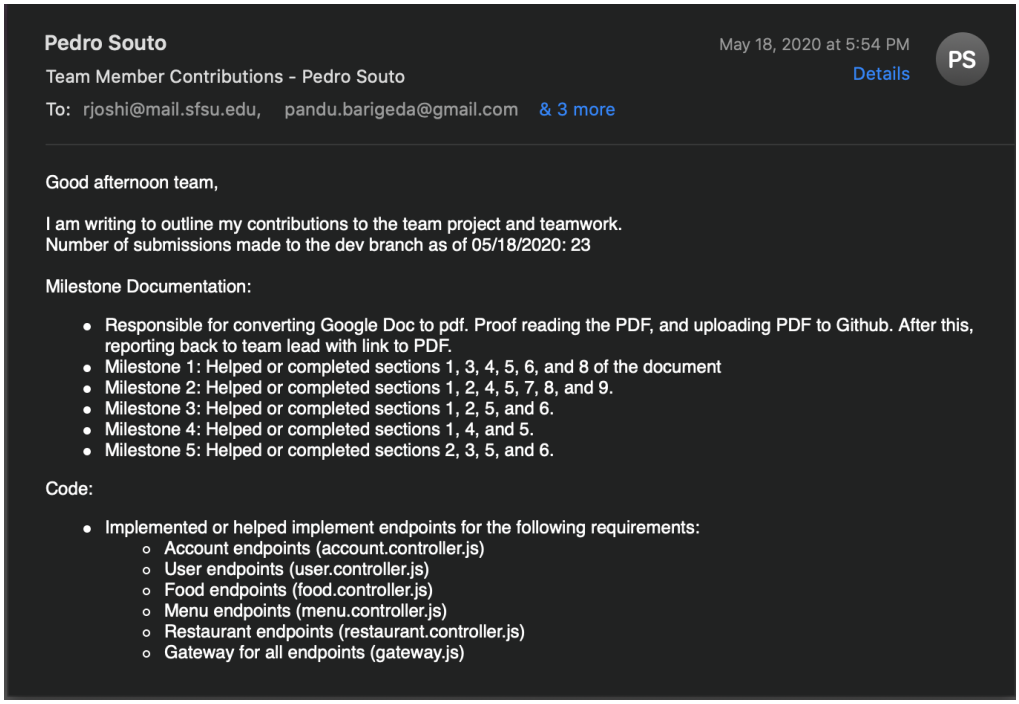
Frontend Board



Milestone 5 Delivery Board



Section 7: Team Member Contributions



Pedro Souto May 18, 2020 at 5:54 PM Details

Team Member Contributions - Pedro Souto

To: rjoshi@mail.sfsu.edu, pandu.barigeda@gmail.com & 3 more

Good afternoon team,

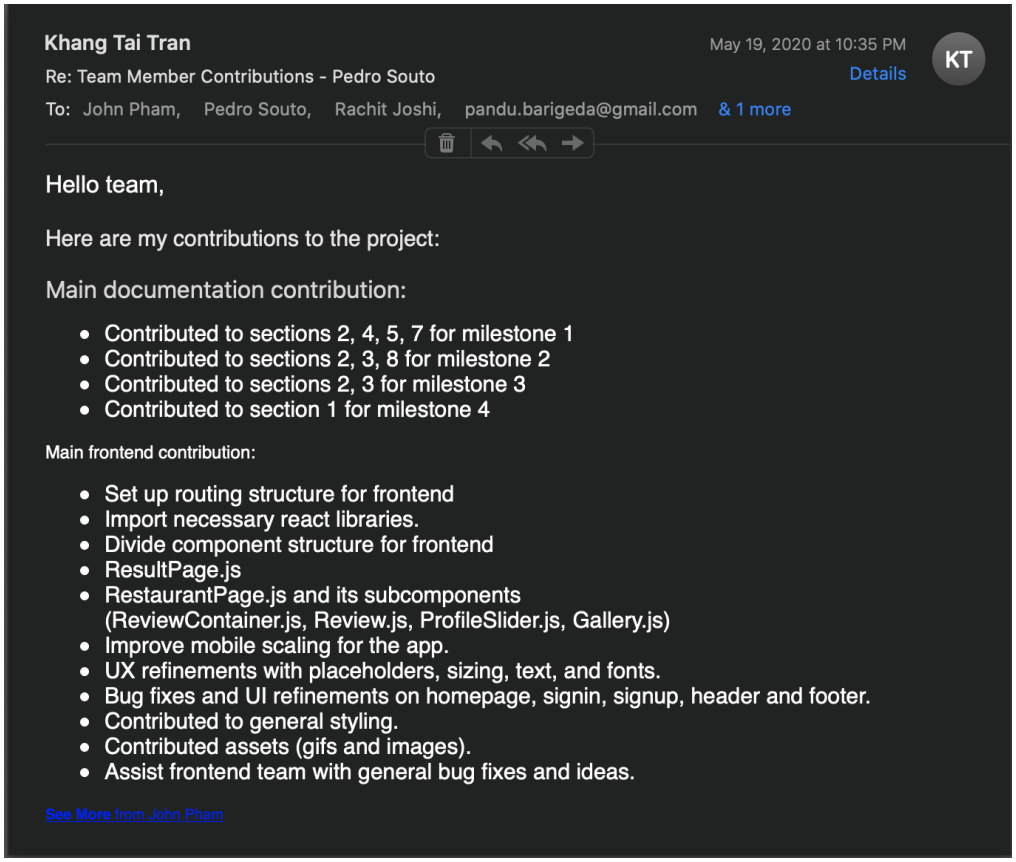
I am writing to outline my contributions to the team project and teamwork.
Number of submissions made to the dev branch as of 05/18/2020: 23

Milestone Documentation:

- Responsible for converting Google Doc to pdf. Proof reading the PDF, and uploading PDF to Github. After this, reporting back to team lead with link to PDF.
- Milestone 1: Helped or completed sections 1, 3, 4, 5, 6, and 8 of the document
- Milestone 2: Helped or completed sections 1, 2, 4, 5, 7, 8, and 9.
- Milestone 3: Helped or completed sections 1, 2, 5, and 6.
- Milestone 4: Helped or completed sections 1, 4, and 5.
- Milestone 5: Helped or completed sections 2, 3, 5, and 6.

Code:

- Implemented or helped implement endpoints for the following requirements:
  - Account endpoints (account.controller.js)
  - User endpoints (user.controller.js)
  - Food endpoints (food.controller.js)
  - Menu endpoints (menu.controller.js)
  - Restaurant endpoints (restaurant.controller.js)
  - Gateway for all endpoints (gateway.js)



Khang Tai Tran May 19, 2020 at 10:35 PM Details

Re: Team Member Contributions - Pedro Souto

To: John Pham, Pedro Souto, Rachit Joshi, pandu.barigeda@gmail.com & 1 more

Hello team,

Here are my contributions to the project:

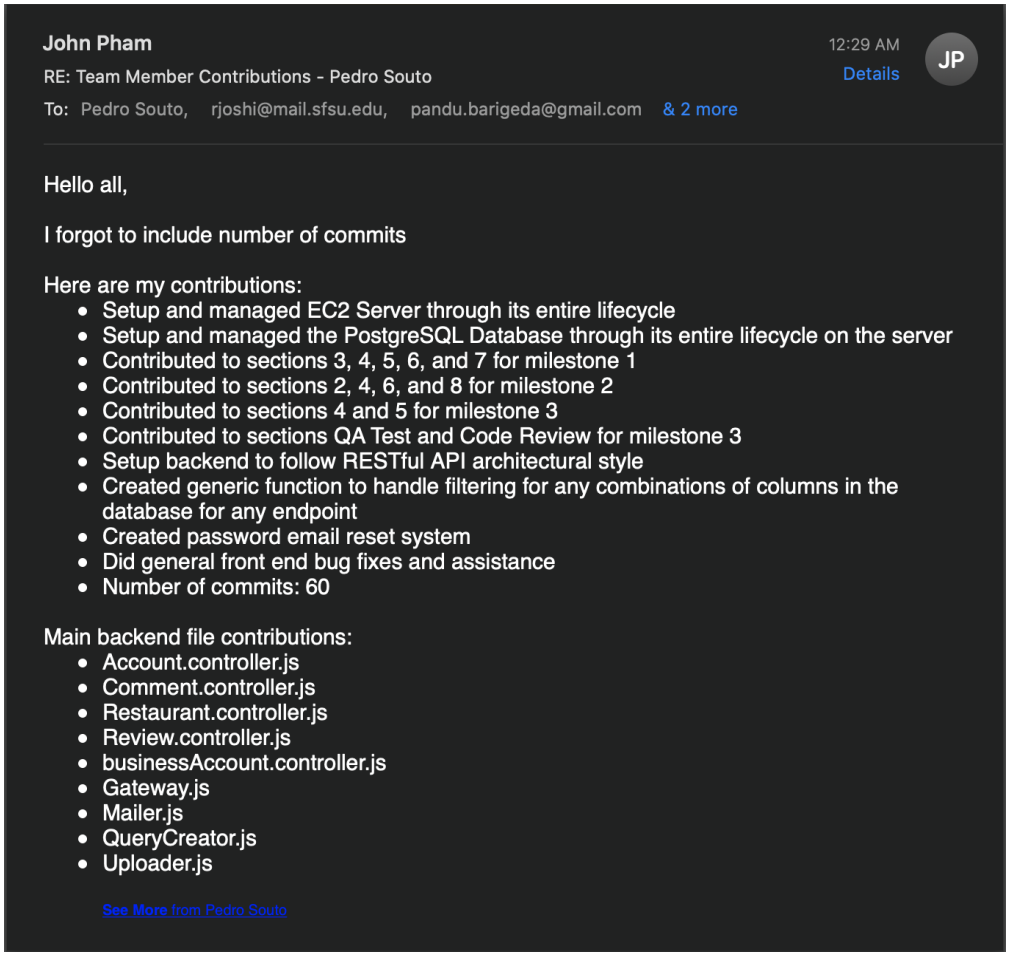
Main documentation contribution:

- Contributed to sections 2, 4, 5, 7 for milestone 1
- Contributed to sections 2, 3, 8 for milestone 2
- Contributed to sections 2, 3 for milestone 3
- Contributed to section 1 for milestone 4

Main frontend contribution:

- Set up routing structure for frontend
- Import necessary react libraries.
- Divide component structure for frontend
- ResultPage.js
- RestaurantPage.js and its subcomponents (ReviewContainer.js, Review.js, ProfileSlider.js, Gallery.js)
- Improve mobile scaling for the app.
- UX refinements with placeholders, sizing, text, and fonts.
- Bug fixes and UI refinements on homepage, signin, signup, header and footer.
- Contributed to general styling.
- Contributed assets (gifs and images).
- Assist frontend team with general bug fixes and ideas.

See More from John Pham



John Pham 12:29 AM
RE: Team Member Contributions - Pedro Souto Details
To: Pedro Souto, rjoshi@mail.sfsu.edu, pandu.barigeda@gmail.com & 2 more

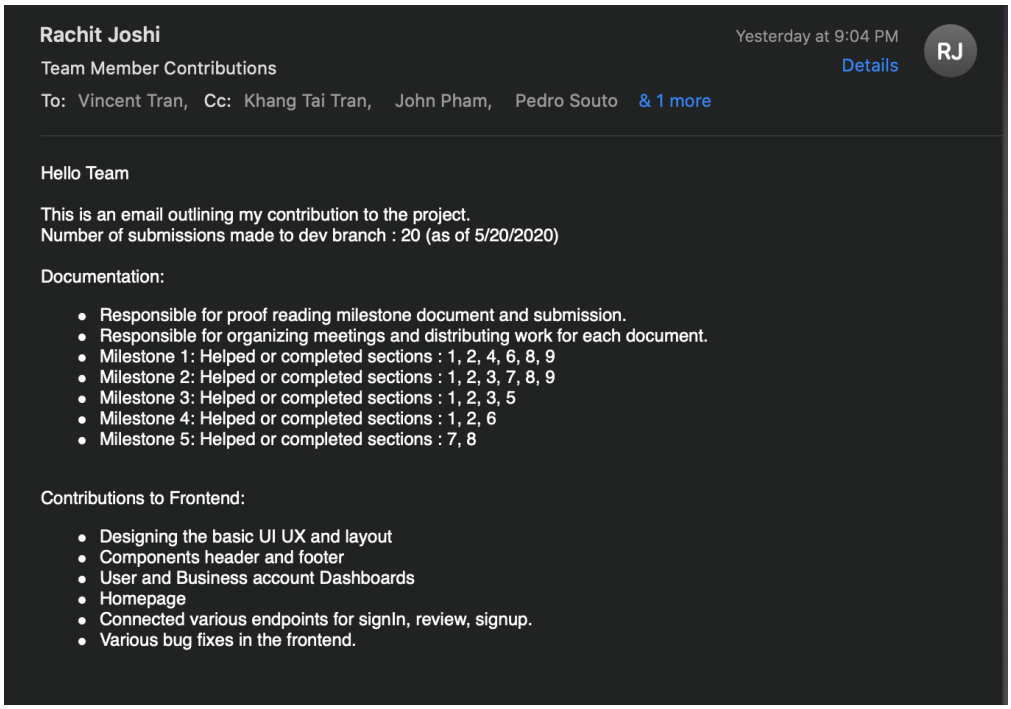
Hello all,

I forgot to include number of commits

Here are my contributions:
- Setup and managed EC2 Server through its entire lifecycle
- Setup and managed the PostgreSQL Database through its entire lifecycle on the server
- Contributed to sections 3, 4, 5, 6, and 7 for milestone 1
- Contributed to sections 2, 4, 6, and 8 for milestone 2
- Contributed to sections 4 and 5 for milestone 3
- Contributed to sections QA Test and Code Review for milestone 3
- Setup backend to follow RESTful API architectural style
- Created generic function to handle filtering for any combinations of columns in the database for any endpoint
- Created password email reset system
- Did general front end bug fixes and assistance
- Number of commits: 60

Main backend file contributions:
- Account.controller.js
- Comment.controller.js
- Restaurant.controller.js
- Review.controller.js
- businessAccount.controller.js
- Gateway.js
- Mailer.js
- QueryCreator.js
- Uploader.js

See More from Pedro Souto



Rachit Joshi Yesterday at 9:04 PM
Team Member Contributions Details
To: Vincent Tran, Cc: Khang Tai Tran, John Pham, Pedro Souto & 1 more

Hello Team

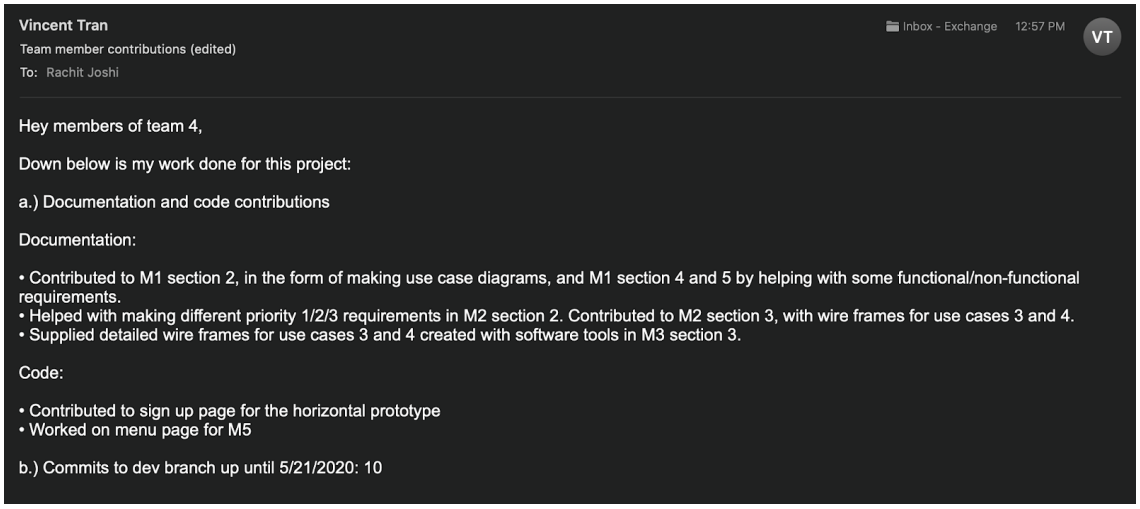
This is an email outlining my contribution to the project.
Number of submissions made to dev branch : 20 (as of 5/20/2020)

Documentation:

- Responsible for proof reading milestone document and submission.
- Responsible for organizing meetings and distributing work for each document.
- Milestone 1: Helped or completed sections : 1, 2, 4, 6, 8, 9
- Milestone 2: Helped or completed sections : 1, 2, 3, 7, 8, 9
- Milestone 3: Helped or completed sections : 1, 2, 3, 5
- Milestone 4: Helped or completed sections : 1, 2, 6
- Milestone 5: Helped or completed sections : 7, 8

Contributions to Frontend:

- Designing the basic UI UX and layout
- Components header and footer
- User and Business account Dashboards
- Homepage
- Connected various endpoints for signIn, review, signup.
- Various bug fixes in the frontend.



Vincent Tran

Inbox - Exchange 12:57 PM

VT

Team member contributions (edited)

To: Rachit Joshi

Hey members of team 4,

Down below is my work done for this project:

a.) Documentation and code contributions

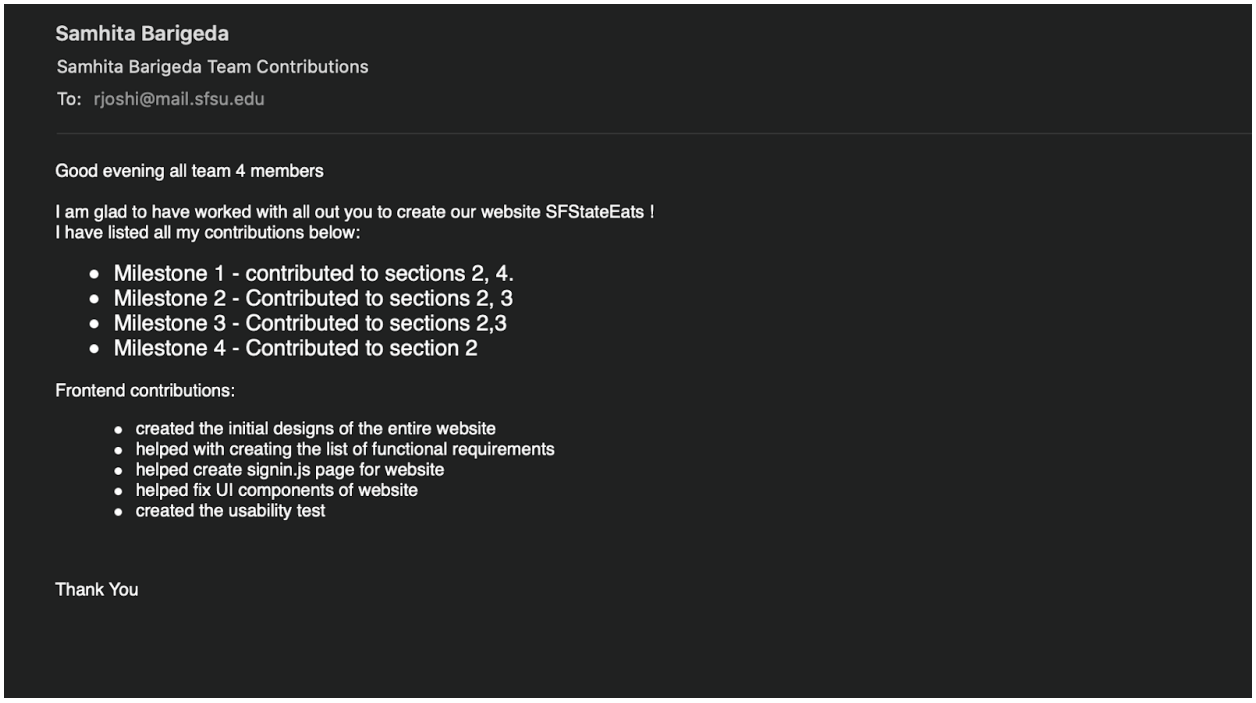
Documentation:

• Contributed to M1 section 2, in the form of making use case diagrams, and M1 section 4 and 5 by helping with some functional/non-functional requirements.
• Helped with making different priority 1/2/3 requirements in M2 section 2. Contributed to M2 section 3, with wire frames for use cases 3 and 4.
• Supplied detailed wire frames for use cases 3 and 4 created with software tools in M3 section 3.

Code:

• Contributed to sign up page for the horizontal prototype
• Worked on menu page for M5

b.) Commits to dev branch up until 5/21/2020: 10



Samhita Barigeda

Samhita Barigeda Team Contributions

To: rjoshi@mail.sfsu.edu

Good evening all team 4 members

I am glad to have worked with all out you to create our website SFStateEats !
I have listed all my contributions below:

- Milestone 1 - contributed to sections 2, 4.
- Milestone 2 - Contributed to sections 2, 3
- Milestone 3 - Contributed to sections 2,3
- Milestone 4 - Contributed to section 2

Frontend contributions:

- created the initial designs of the entire website
- helped with creating the list of functional requirements
- helped create signin.js page for website
- helped fix UI components of website
- created the usability test

Thank You

Section 8: Post Analysis

There were a couple of issues while designing this. These issues though not a huge problem by themselves, are exaggerated with the inability to conduct meetings regularly and in person. The technical challenges came in the form of CORS errors, Frontend organization, and issues deployment. The team was able to deal with these issues but the situation made it so that it took more time and effort to effectively communicate and resolve these errors. Another factor contributing to this was the fact that there was a learning curve to the technologies that we are using, as some team members were not confident with the libraries/platform. The communication issues meant that whenever one of the team members hit a roadblock, the issue reached other team members after a while.

While no one can change the situation, I think with better planning and use of technologies these issues can be mitigated. Frontend and backend teams can improve communication and have a greater degree of planning in the integration of endpoints and responses. React, being so complex, started causing problems as the website got bigger. Better planning of the hierarchical structure in react will facilitate better passing of props across multiple components. React also makes us stick with inbuilt components which are limiting. We also had issues with deployment. Code working on localhost throws CORS errors when ported. Resolving these bugs was tedious but now that the backend team is aware about what’s causing these bugs, they can be vigilant.