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

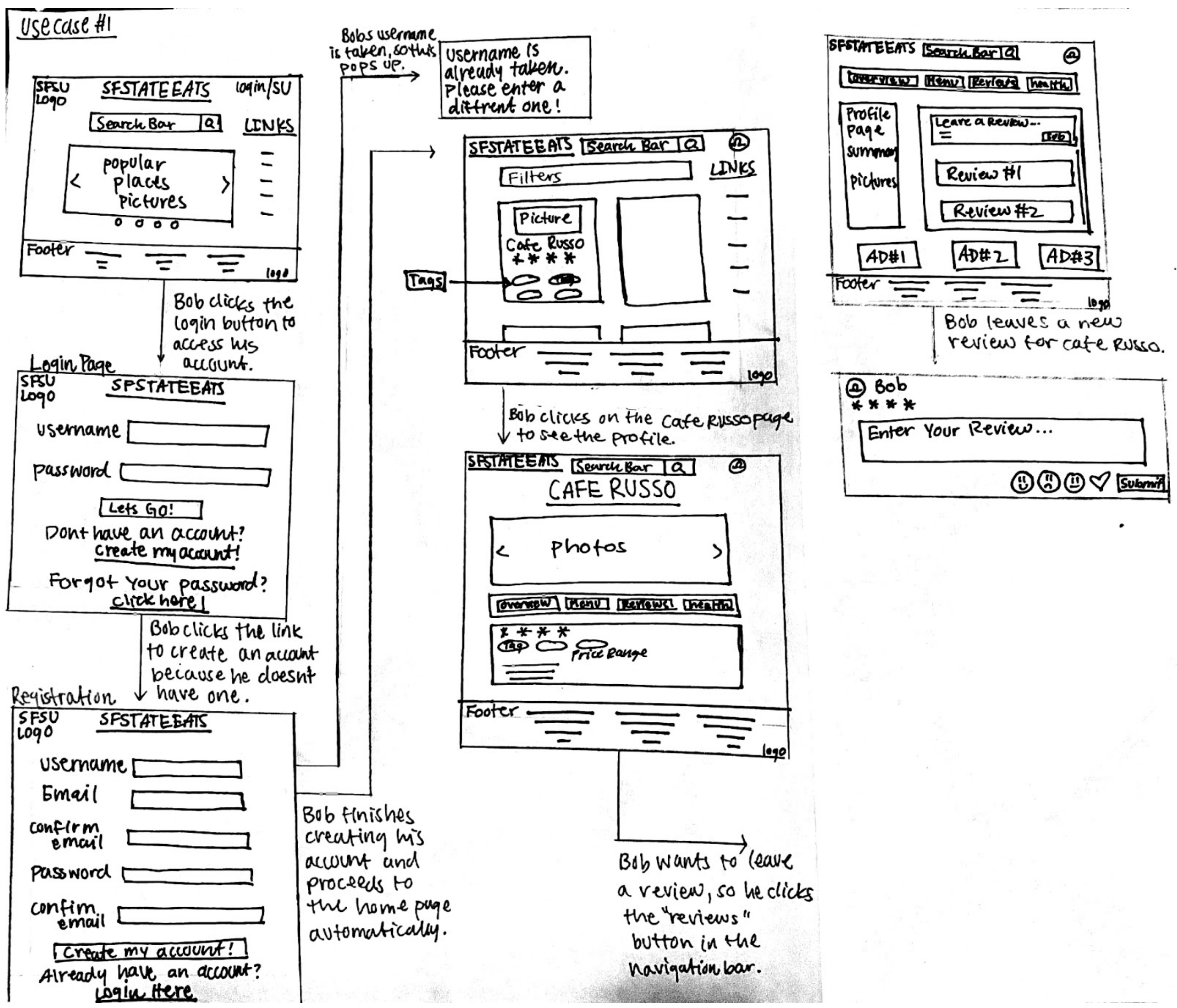
66. System Administrators shall be able to cancel orders placed at the customer's request.
    66.1. Customer cancellation requests shall be required to have a reason for cancelation.
67. System Administrators shall be able to refund the customer if the order is not what they wanted.

System

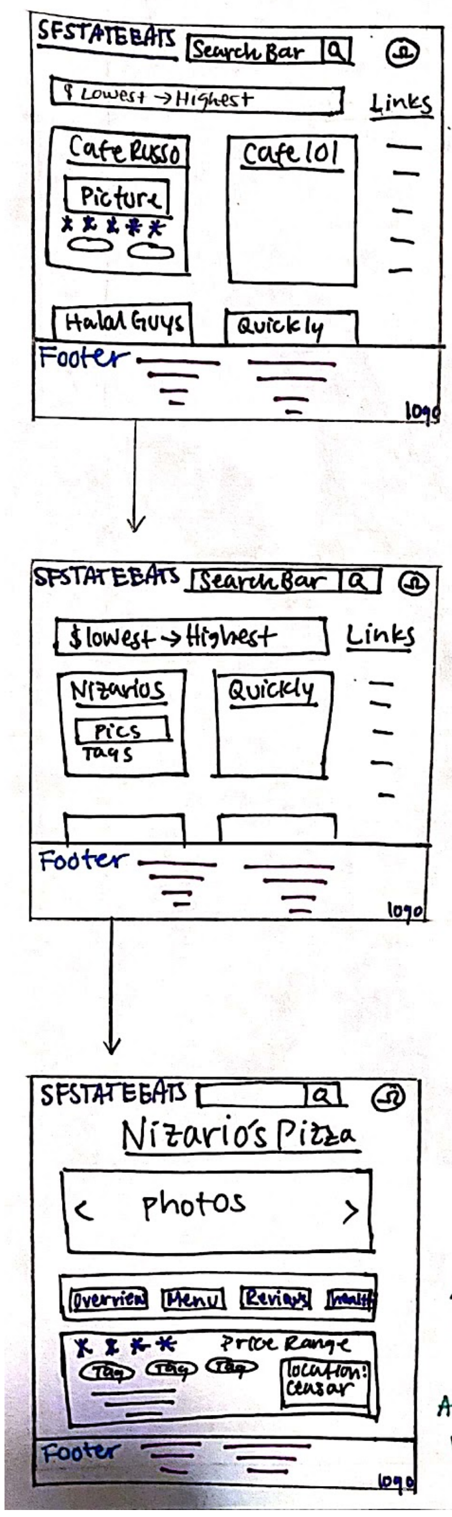
68. The system shall show trending restaurants on the main page.
    68.1. Trending restaurants are based on check-in's and new positive reviews.
    68.2. Trending restaurants are updated every 3 days.
    68.3. Display new reviews on the page.

Section 3: UI Mockups and Storyboards

WireFrame for Use Case 1


Use case #1
SFSU Logo
SFSTATE EATS
login/SU
Search Bar
LINKS
popular places pictures
Footer
logo
Bob clicks the login button to access his account.
Login Page
SFSU Logo
SFSTATEEATS
username
password
Lets Go!
Dont have an account? create my account!
Forgot Your password? click here!
Bob clicks the link to create an account because he doesnt have one.
Registration
SFSU Logo
SFSTATEEATS
username
Email
confirm email
Password
confirm email
Create my account!
Already have an account? login Here
Bob finishes creating his account and proceeds to the home page automatically.
Bobs username is taken, so this pops up.
Username is already taken. Please enter a diffrent one!
SFSTATEEATS
Search Bar
Filters
LINKS
Picture
Cafe Russo
Tags
Footer
logo
Bob clicks on the Cafe Russo page to see the profile.
SFSTATEEATS
Search Bar
CAFE RUSSO
Photos
overview
Menu
Reviews
health
Tag
Price Range
Footer
logo
Bob wants to leave a review, so he clicks the "reviews" button in the navigation bar.
SFSTATEEATS
Search Bar
overview
Menu
Reviews
health
Profile Page summary
pictures
Leave a Review...
Review #1
Review #2
AD#1
AD#2
AD#3
Footer
logo
Bob leaves a new review for cafe Russo.
Bob
Enter Your Review...
Submit

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.

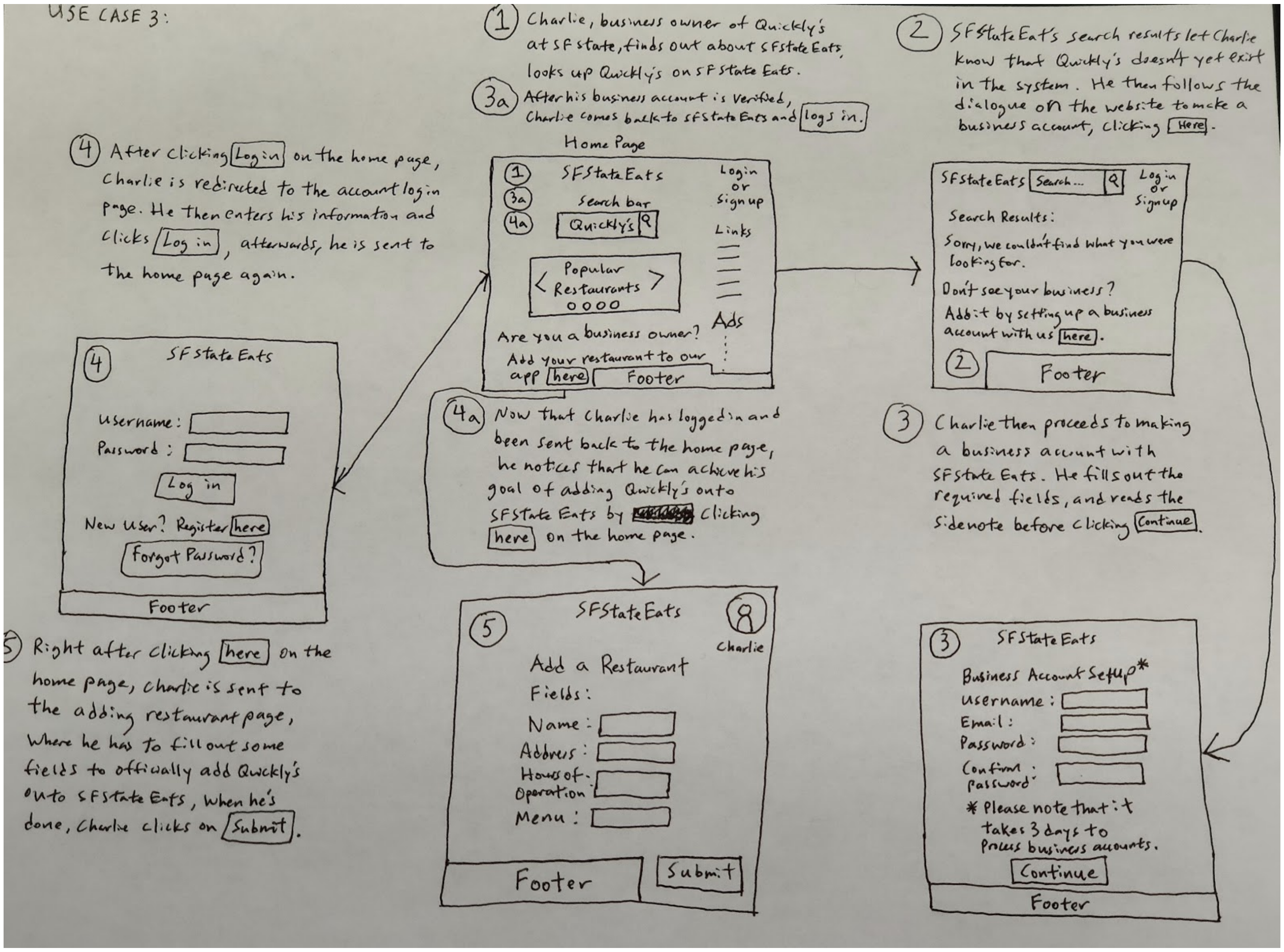
After sorting the restaurants, the webpage shows the restaurants with cheapest prices.

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

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

Alice also sees the ratings of the restaurant.

WireFrame for Use Case 3



(1) Charlie, business owner of Quickly's at SF state, finds out about SFState Eats, looks up Quickly's on SFState Eats.

(3a) After his business account is verified, Charlie comes back to SFState Eats and logs in.

(2) SFStateEat's search results let Charlie know that Quickly's doesn't yet exist in the system. He then follows the dialogue on the website to make a business account, clicking Here.

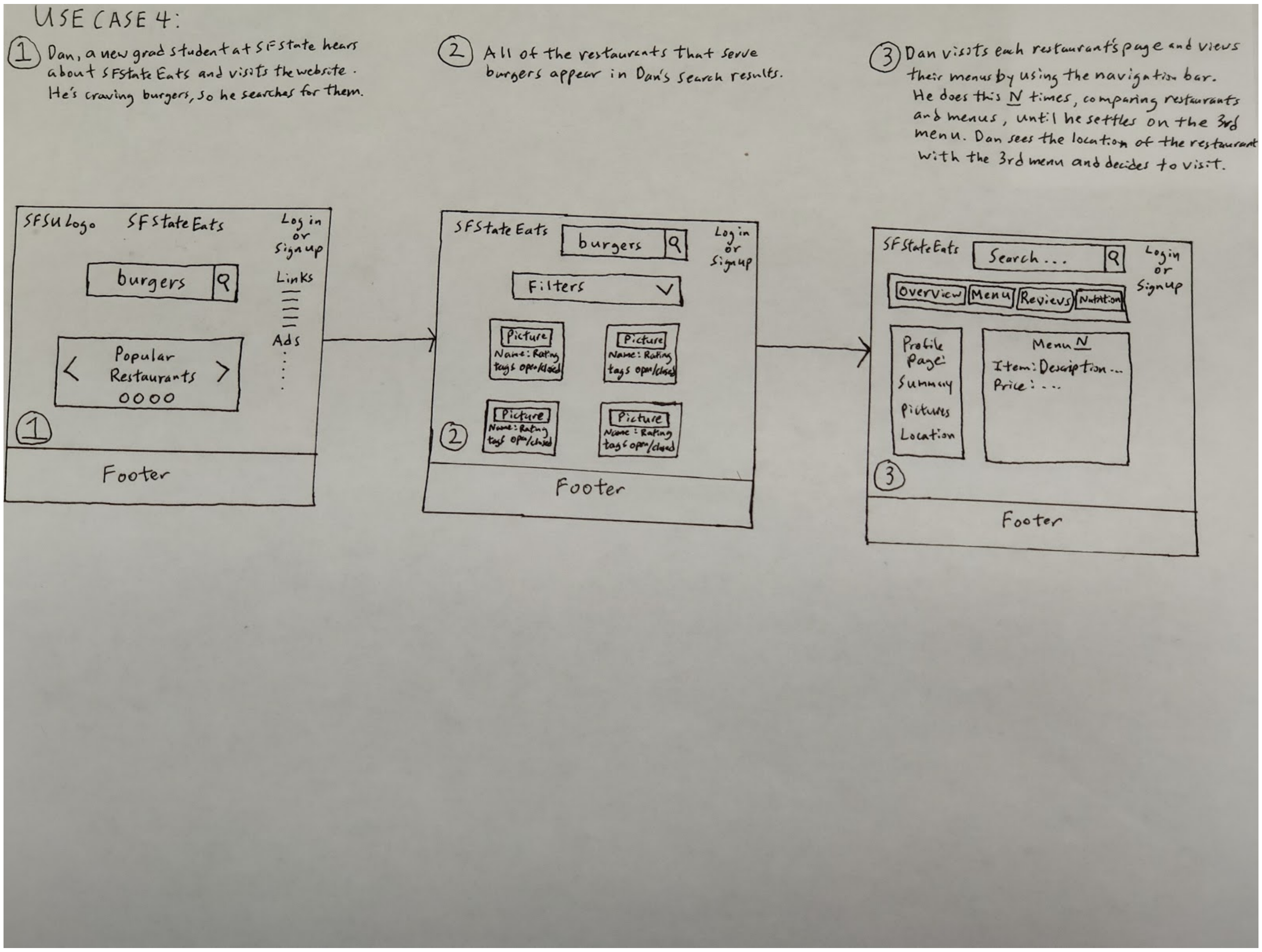
(4) After clicking Log in on the home page, Charlie is redirected to the account login page. He then enters his information and clicks Log in, afterwards, he is sent to the home page again.

(4a) Now that Charlie has logged in and been sent back to the home page, he notices that he can achieve his goal of adding Quickly's onto SFState Eats by [illegible] clicking here on the home page.

(3) Charlie then proceeds to making a business account with SFState Eats. He fills out the required fields, and reads the sidenote before clicking Continue.

(5) Right after clicking here on the home page, Charlie is sent to the adding restaurant page, where he has to fill out some fields to officially add Quckly's onto SFState Eats, when he's done, Charlie clicks on Submit.

WireFrame for Use Case 4



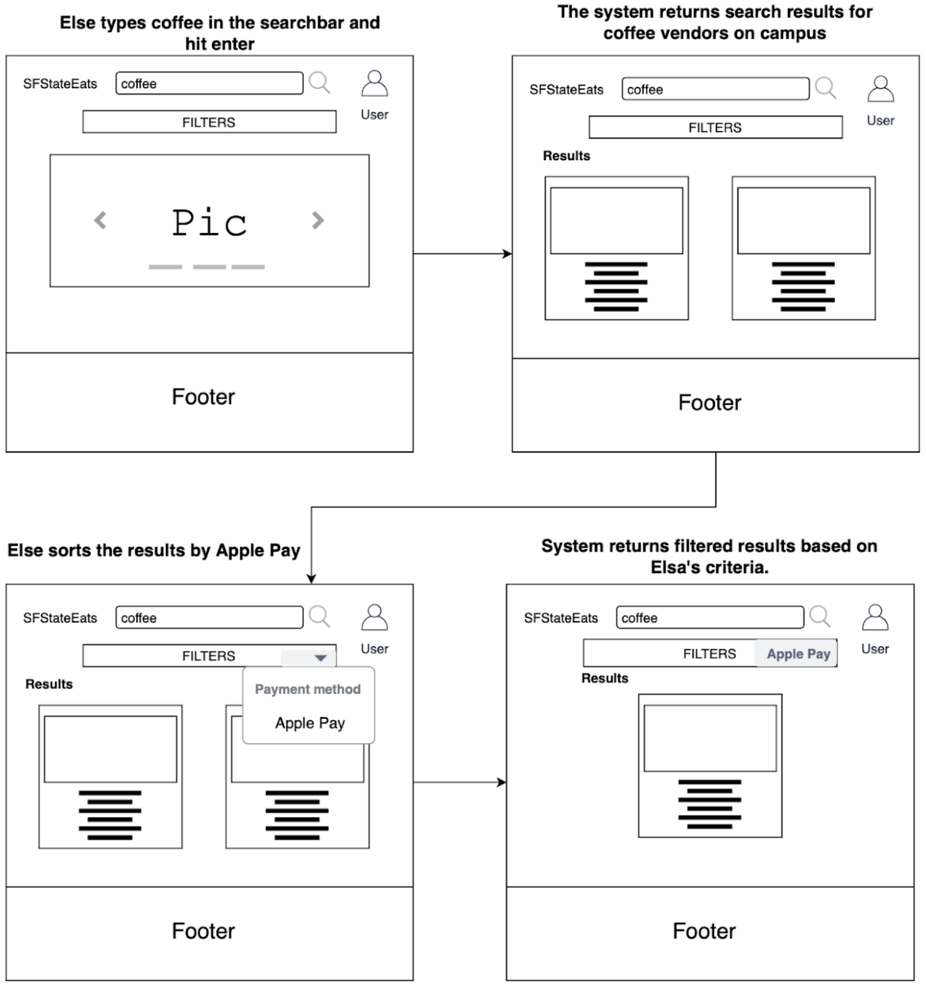
USE CASE 4:

1. Dan, a new grad student at SFState hears about SFState Eats and visits the website. He's craving burgers, so he searches for them.

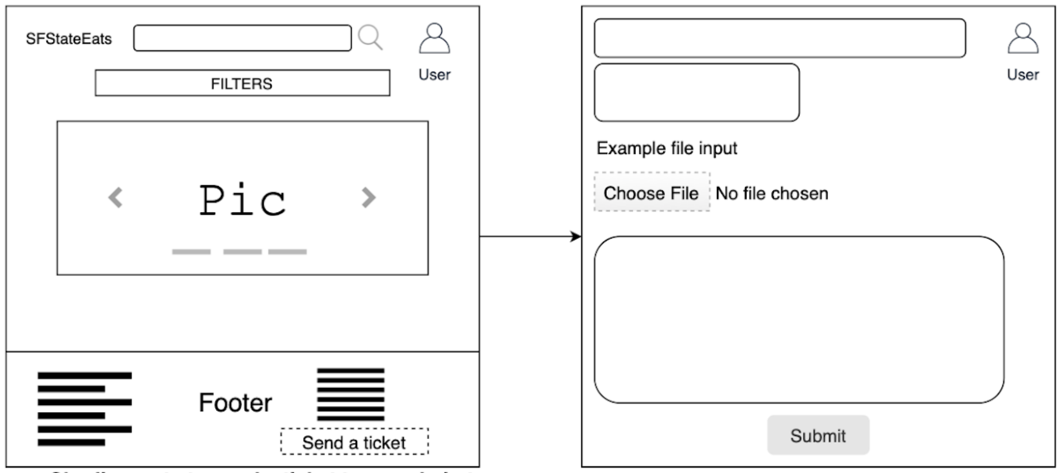
2. All of the restaurants that serve burgers appear in Dan's search results.

3. Dan visits each restaurant's page and views their menus by using the navigation bar. He does this N times, comparing restaurants and menus, until he settles on the 3rd menu. Dan sees the location of the restaurant with the 3rd menu and decides to visit.

WireFrame for Use Case 5

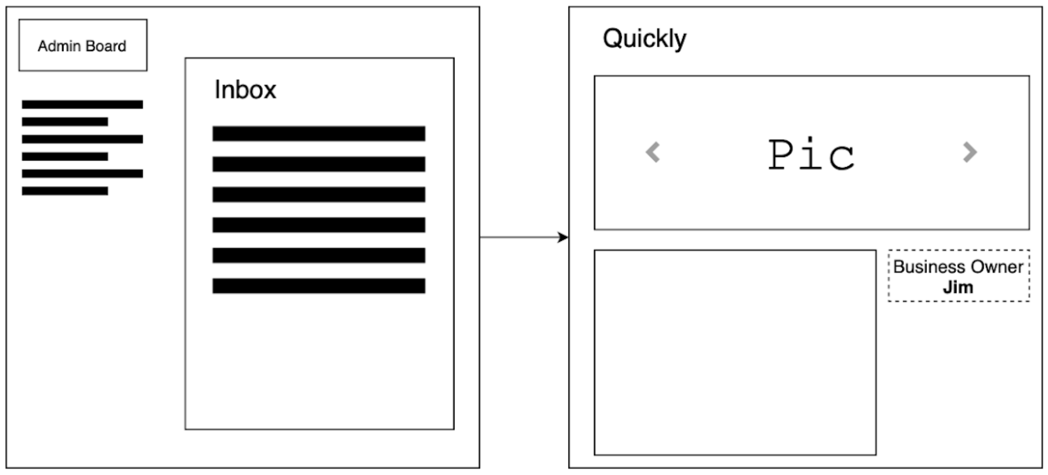

Else types coffee in the searchbar and hit enter
SFStateEats
coffee
User
FILTERS
Pic
Footer
The system returns search results for coffee vendors on campus
SFStateEats
coffee
User
FILTERS
Results
Footer
Else sorts the results by Apple Pay
SFStateEats
coffee
User
FILTERS
Results
Payment method
Apple Pay
Footer
System returns filtered results based on Elsa's criteria.
SFStateEats
coffee
User
FILTERS
Apple Pay
Results
Footer

WireFrame for Use Case 6



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

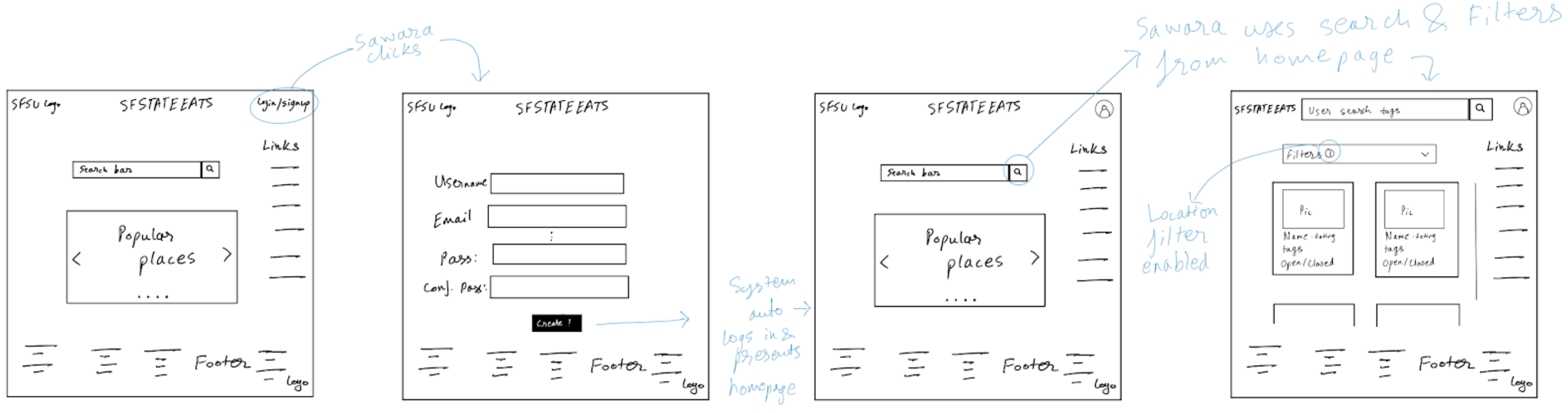
Charlie enters the details and upload necessary documents



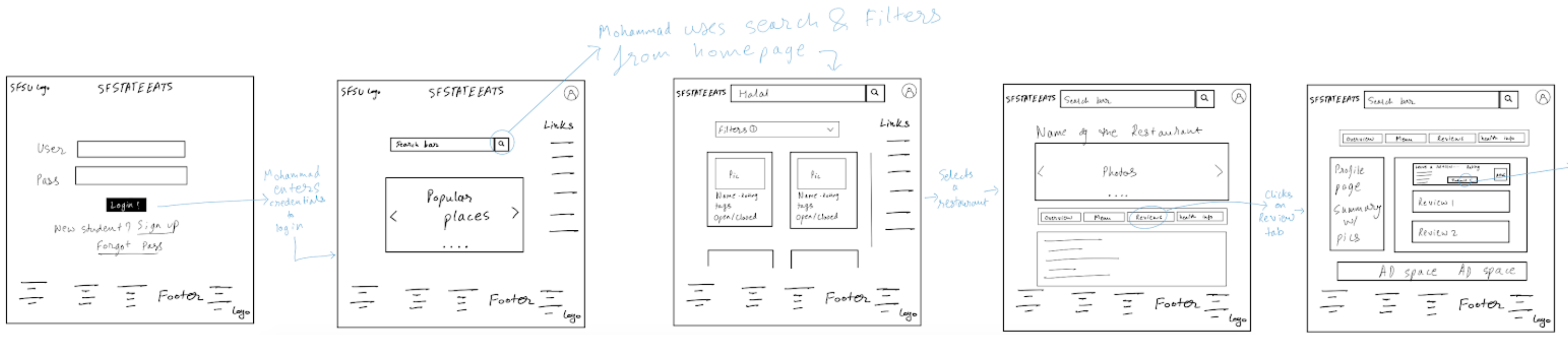
The sysadmin sees Charlie's message in the admin dashboad

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

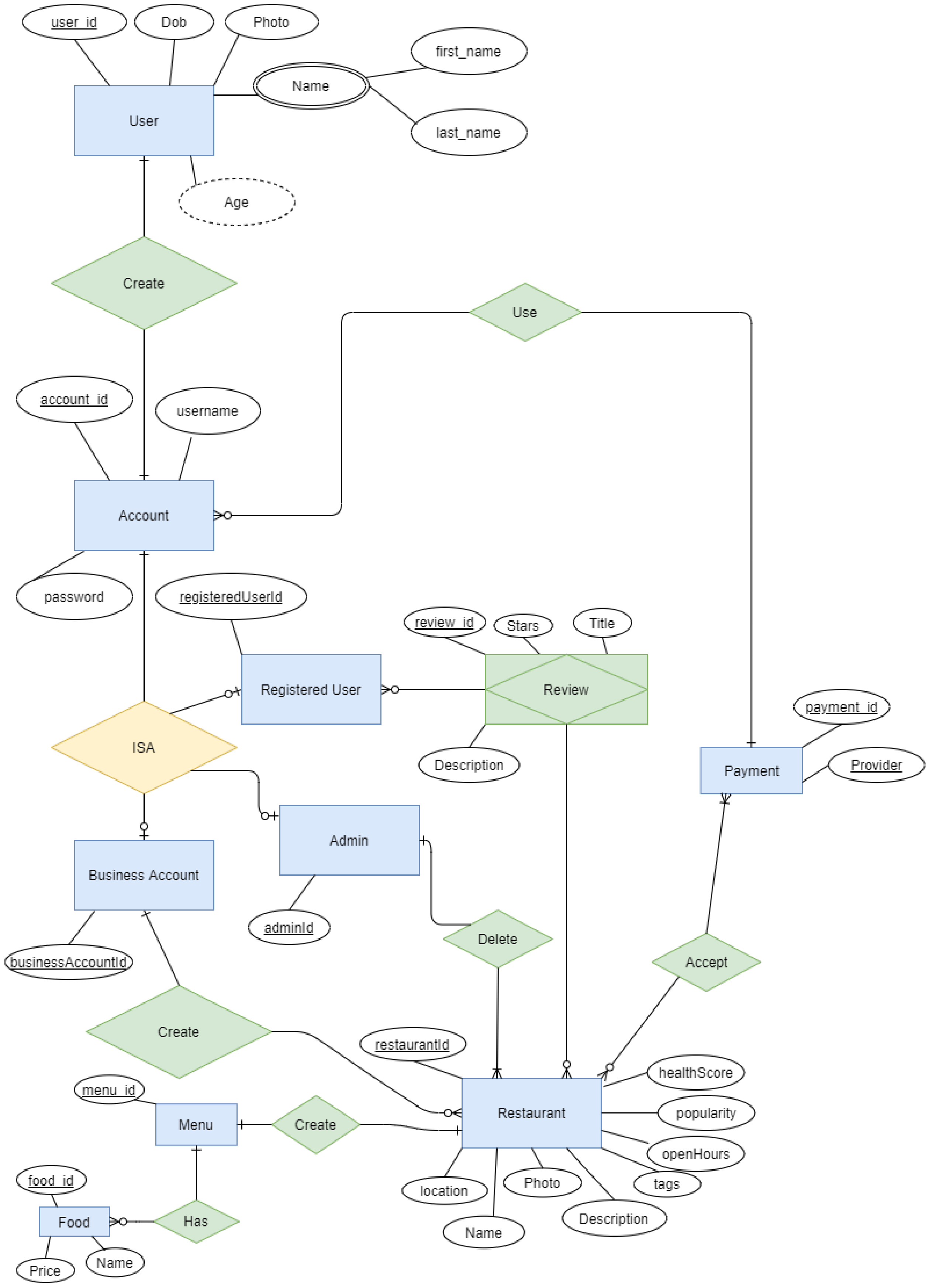
- paymentId: PK
- provider

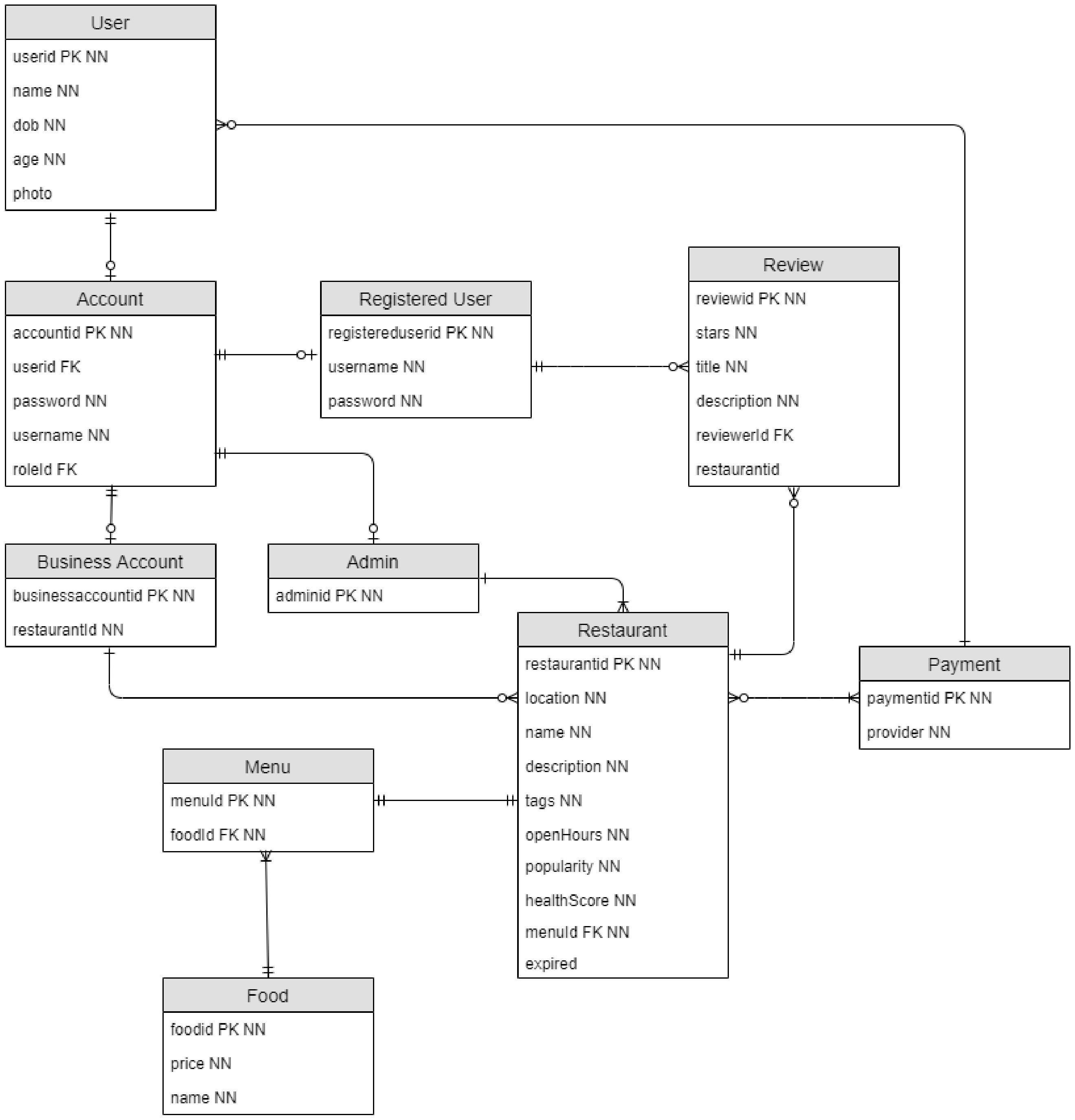
Ticket

- 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
payment_id
Provider
Payment
ISA
Description
Admin
Business Account
adminId
Delete
Accept
businessAccountId
Create
restaurantId
menu_id
healthScore
Menu
Create
Restaurant
popularity
openHours
tags
food_id
location
Photo
Description
Food
Has
Name
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.

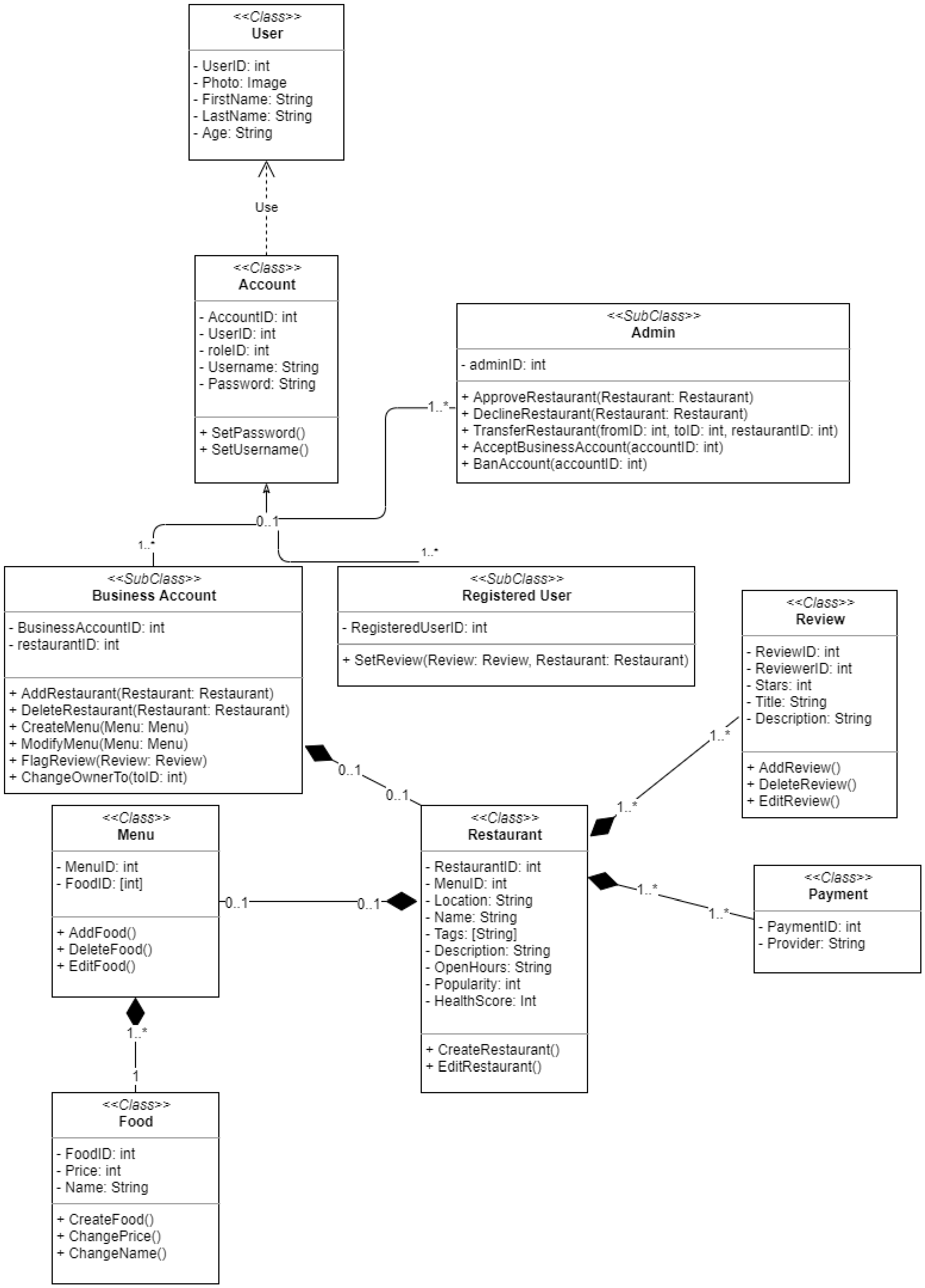
New Restaurants List

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

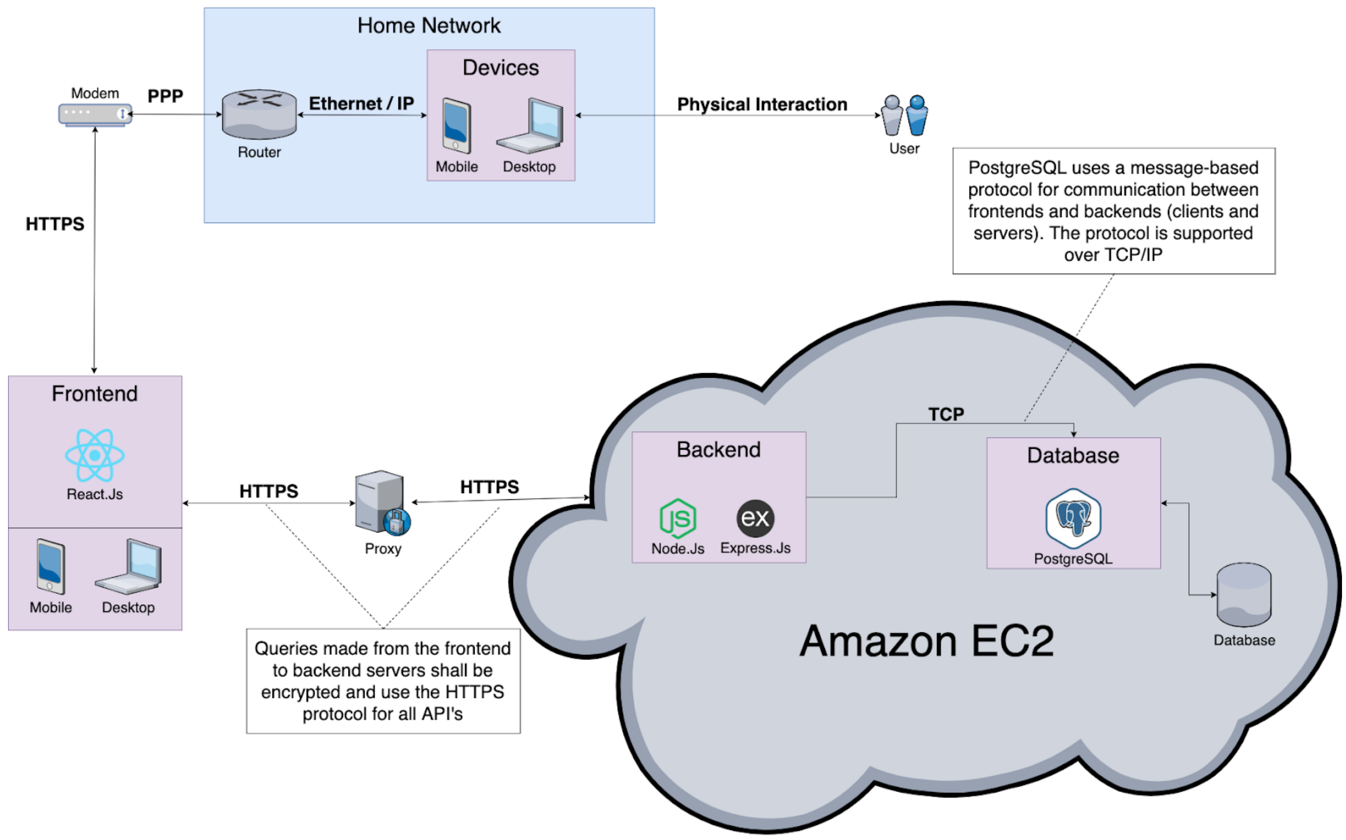
On the main page of the application, there will be a list of recent reviews that users have posted. In order to display these, we will keep an ordered list utilizing a stack structure. Everytime a new review is added to the platform, it must also be added to the stack of new reviews, which will keep 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

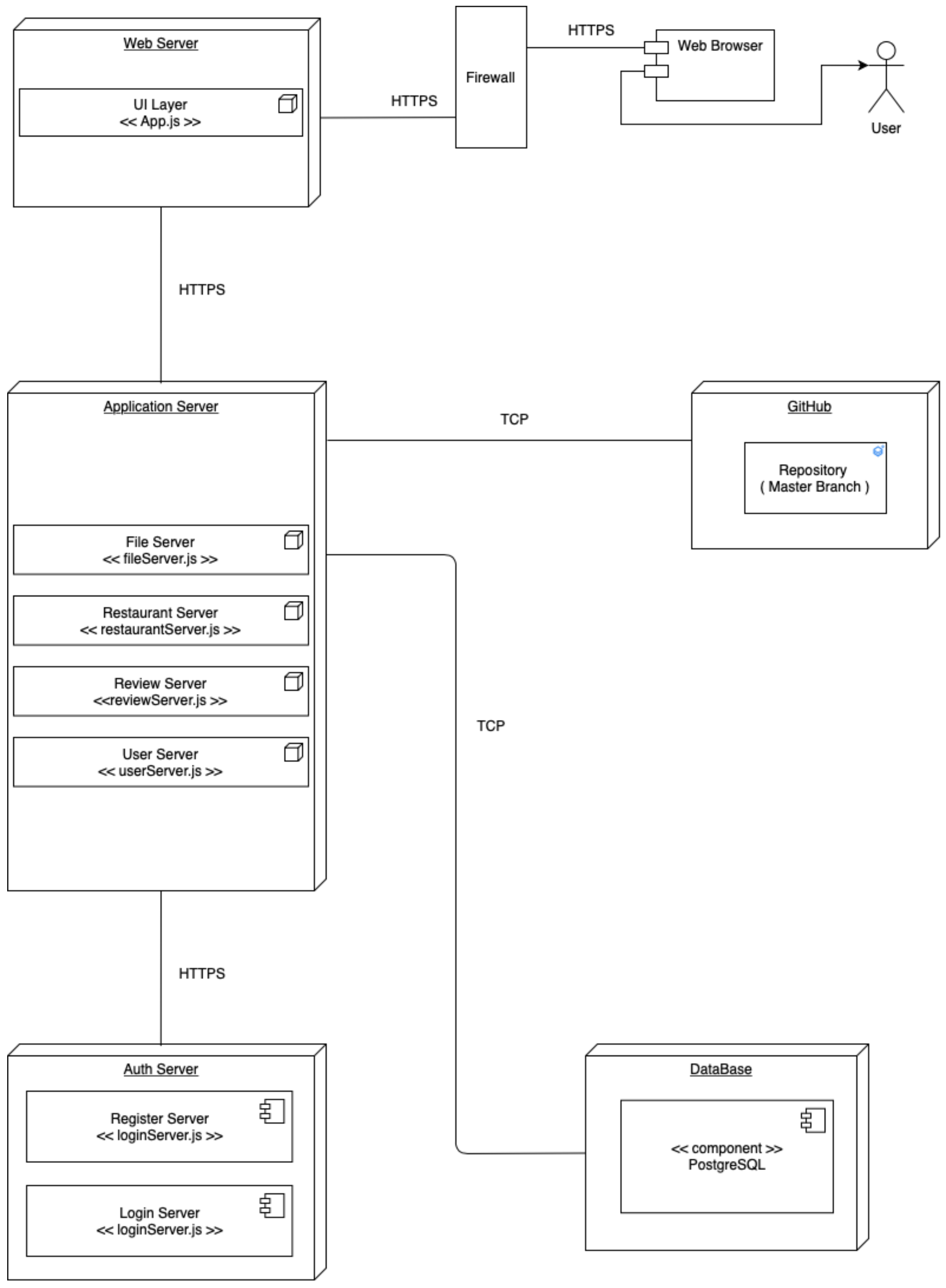


Section 7: High Level Application Network and Deployment Diagrams

Network Diagram


Home Network
Devices
Modem
PPP
Router
Ethernet / IP
Mobile
Desktop
Physical Interaction
User
HTTPS
PostgreSQL uses a message-based protocol for communication between frontends and backends (clients and servers). The protocol is supported over TCP/IP
Frontend
React.Js
Mobile
Desktop
HTTPS
Proxy
HTTPS
TCP
Backend
Node.Js
Express.Js
Database
PostgreSQL
Database
Amazon EC2
Queries made from the frontend to backend servers shall be encrypted and use the HTTPS protocol for all API's

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.