SW Engineering CSC648-848-05 Fall2023

Project: BiteRate
Team number: Team 5
Team Lead: Alec Nagal | anagal1@mail.sfsu.edu
Frontend Lead: Ali Alsharif
Backend Lead: Gerry Putra
Doc Master: Gabriella Arcilla
Database Master: Kenneth Pang
Backend Assistant: Robel Ayelew
Team: James Lu, Robin Reyes
Milestone 2
Date: September 21, 2023

Milestone/Checkpoint #	**Date Submitted**	**Revision #**
Milestone 1v1 Checkpoint #1	09/10/2023	Revision #0
Milestone 1v1 Checkpoint #2	09/21/2023	Revision #0
Milestone 1v2	10/12/2023	Revision #1
Milestone 2v1	10/12/2023	Revision #0

Table of Contents

Data Definitions

1. ***User***
    a. A user is a registered user who can create many accounts and be authenticated by the system to verify their identity before granting full access to the application. In addition, registered users have the perks of rating restaurants, and reviewing as well as recommending restaurants to other registered users. Furthermore, registered users shall be able to search for restaurants, registered users, and dislike restaurants and cuisines. Most importantly, registered users who rate restaurants shall have points accumulated as their score. Also, registered users shall be able to see all their past reviews, past restaurants that they rated, and the restaurants they visited. Importantly, registered users shall have full power to edit or manage their profiles, add or follow other registered users as well and bookmark restaurants to their favorite list. Last but not least, registered users shall be able to send private messages to registered users to exchange information on restaurants that they visited.

2. ***Account***
    a. Account shall be created by one user and the user shall have a type, meaning a user can be basic, admin, or restaurant. Accounts created for the first time by any type shall need verification upon registration. To get into the details, basic is a new customer who just joined, admin is an employee who has complete control of the system, and restaurant is a business that wants to get their customers' feedback. Most importantly, users who accumulated a certain amount of points or scores shall receive ranks such as silver, gold, or platinum. Finally, a restaurant shall be able to create many accounts and only the accounts created by the restaurants shall be able to post restaurants.

3. ***Admin***
    a. An admin is a type of user who has full control of the system, meaning this type of user shall be able to see all the existing users and have access to their account information. Most importantly, the admin shall be able to add a new restaurant and temporarily or permanently remove existing restaurants if necessary.

4. ***Restaurant***
    a. A restaurant is a business that shall have information such as brand name, address, location, website, and rating. This type of user shall show or provide the kinds of cuisines that the restaurant is serving, therefore to showcase all the cuisines that the restaurant is providing to its customers, it shall have its own profile page that shows its brand image, products, rating, customer reviews, and

photos.

5. ***Friends***
    a. Friends in BiteRate are registered users who follow each other, meaning registered user A followed registered user B, and user B followed user A, therefore users A and B are friends at this point. The perks of these types of users are that they shall be able to see each other activities such as restaurants they visited, restaurants they liked, and the restaurants they reviewed.

6. **Point / Score**
    a. Point or Score is information that BiteRate uses to assign ranks to the registered users, therefore this information shall be shown as the main information in a user profile page. The points or scores will be given to users who rated, reviewed, or recommended restaurants. In addition, users who earned points shall be listed on the leaderboard in ascending order, meaning the user who earned the most points will be listed in the first place.

7. ***Rating***
    a. Rating shall be uniquely associated with a single restaurant, and the overall rating of a restaurant will be calculated by averaging the rating that the restaurant received from all its customers. Most importantly, the rating shall be able to be modified by the user, and shall not be deleted, meaning a user can modify his or her rate but cannot delete his or her rating.

8. ***Review***
    a. A review shall be uniquely associated with a single restaurant, and a review shall allow the reviewer to embed images. In addition, the review shall have a maximum capacity for the amount of characters it can take. Last but not least, the review shall be able to be updated, modified, or deleted by the author.

9. ***Recommendation (Recs)***
    a. The system shall keep track of the high-rated restaurants, meaning it shall have a form of storage where it keeps the high-rated restaurants, therefore the system can use this “storage” to recommend restaurants to registered users.

10. ***User Activity/History***
    a. The user activity or history is a page that stores all the activities of the user such as the restaurant he or she rated, reviewed, and recommended. Most importantly, certain activities shall be visible to followers or friends.

11. ***ChatAssistance (chatbot customer service)***
    a. ChatAssistance is a virtual helper that serves many users. This virtual assistance shall be able to answer about registered restaurants and cuisines and recommend restaurants to registered users based on the cuisine, restaurant name, or location. 573 South Willard Ave, San Jose, CA 95126

12. Address
    a. An Address shall have information such as street number, name, city, state, and zip code. The address shall have one city and it shall belong to one address.

13. ***Cities***
    a. A city shall have information such as city name, county, state, and zip code. Also, the city shall belong to one address.

14. ***Notifications***
    a. A Notification shall notify registered users if they receive messages, friend requests, restaurant recommendations, and updates about their reviews.

Prioritized Functional Requirements

Priority 1:

1. ***User***

1.1. A user shall be able to create many accounts.
1.2. A user shall be authenticated by the system to verify their identity before granting full access to the application.
1.3. A registered user shall be able to rate a restaurant.
1.4. A registered user shall be able to review a restaurant.
1.5. A registered user shall be able to recommend a restaurant to other registered users.
1.6. A registered user shall be able to search for any restaurants.
1.7. A registered user shall be able to search for any other registered users.
1.8. A registered user shall be able to dislike a restaurant.
1.9. A registered user shall be able to dislike a cuisine.
1.10. A registered user who rates a restaurant shall have 'points' accumulated as their 'score'.
1.11. A registered user shall be able to see all their past reviews.
1.12. A registered user shall be able to see all their past restaurant ratings that they rated.
1.13. A registered user shall be able to see all the restaurants that they have visited.
1.14. A registered user shall be able to edit/manage their profile page.
1.15. A registered user shall be able to send a personal message to another registered user.
1.16. A registered user shall be able to add/follow other users.
1.17. A registered user shall be able to bookmark restaurants to their favorites list

2. ***Account***

2.1. An account shall be created by one user.
2.2. An account shall be verified by the user.
2.3. An account shall have a type, 'basic', dedicated for new customers that just join
2.4. An account shall have a type, 'admin', dedicated for employees.
2.5. An account shall have a type, 'restaurant', dedicated for restaurant owners.
2.6. A 'basic' account type user shall receive a 'rank' once the user accumulated a certain amount of points/score.
2.7. A 'basic' account type shall be able to receive rank of 'silver', 'gold', or 'platinum' once the user accumulated a certain amount of points/score.
2.8. An account shall be created as 'restaurant' to be able to post a restaurant.
2.9. A 'restaurant' account shall be able to post many restaurants.
2.10. An account shall need verification upon registering for the first time.

3. ***Admin***

3.1. An account with a type of ‘admin’ shall be able to view all existing users and access to their account information.

3.2. An account with a type of ‘admin’ shall be able to add new restaurants.

3.3. An account with a type of ‘admin’ shall be able to temporarily/permanently remove existing restaurants.

4. ***Restaurant***

4.1. A restaurant shall have information such as brand name, address, location, website.

4.2. A restaurant shall have a rating.

4.3. A restaurant shall declare the type of cuisine they are serving.

4.4. A restaurant shall have its own profile page showcasing their brand image, products, rating, customer reviews and photos.

5. ***Friend/Followers***

5.1. Friends shall be able to see each other’s activities

6. ***Rating***

7.1. A user rating shall be uniquely associated with a single restaurant.

7.2. The overall rating of a restaurant shall be determined by averaging the ratings from all individual users.

7.3. A rating shall be able to be modified by the user.

7.4. A rating shall not be able to be deleted.

7. ***Review***

8.1. A review shall be uniquely associated with a single restaurant.

8.2. A review shall allow users to embed images.

8.3. A review shall have a maximum capacity for the amount of characters it can take.

8.4. A review shall be able to be updated or deleted by the author.

8. ***User Activity/History***

10.1. User activities like reviewing, rating, or recommending shall be assigned a dedicated point/score value.

10.2. User activities on each user shall be recorded.

10.3. Certain activities shall be visible to their friends/followers.

9. ***Address***

14.1. An address shall have basic address information such as street name, city name, phone number, zip code, country.

14.2. An address shall have one city.

10. ***Cities***

15.1. A city shall belong to one address.

11. ***Notifications***

14.1 A user shall receive notifications for receiving new messages

14.2 A user shall receive notifications for friend requests

14.3 A user shall receive notifications for restaurant recommendations.

14.4 A user shall receive notifications on updates about their reviews

Priority 2:

1. ***Point/Score***

    6.1. Point/Score shall be shown as the main information in a user profile page.

    6.2. Point/Score of every user in the platform shall be listed by index as ‘leaderboard’.

    6.3. Point/Score value shall be assigned to certain activities in the app.

    6.4. A certain amount of Points/Score shall be provided to a user who rated a restaurant.

    6.5. A certain amount of Points/Score shall be provided to a user who reviewed a restaurant.

    6.6. A certain amount of Points/Score shall be provided to a user who recommended a restaurant.

2. ***Recommendation (Recs)***

    9.1. High rated restaurants shall be recommended to users

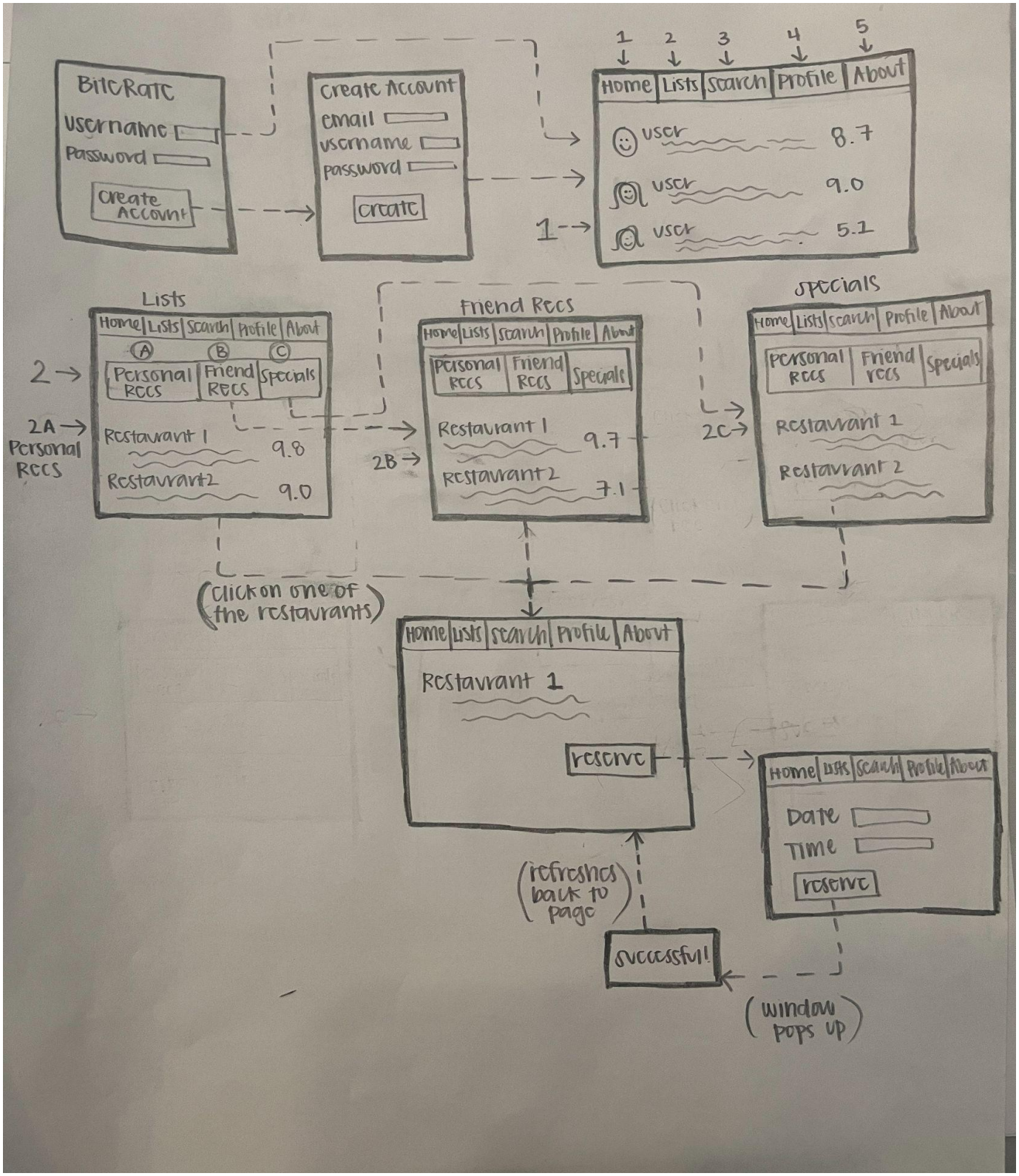
3. ***ChatAssistance (chatbot customer service)***

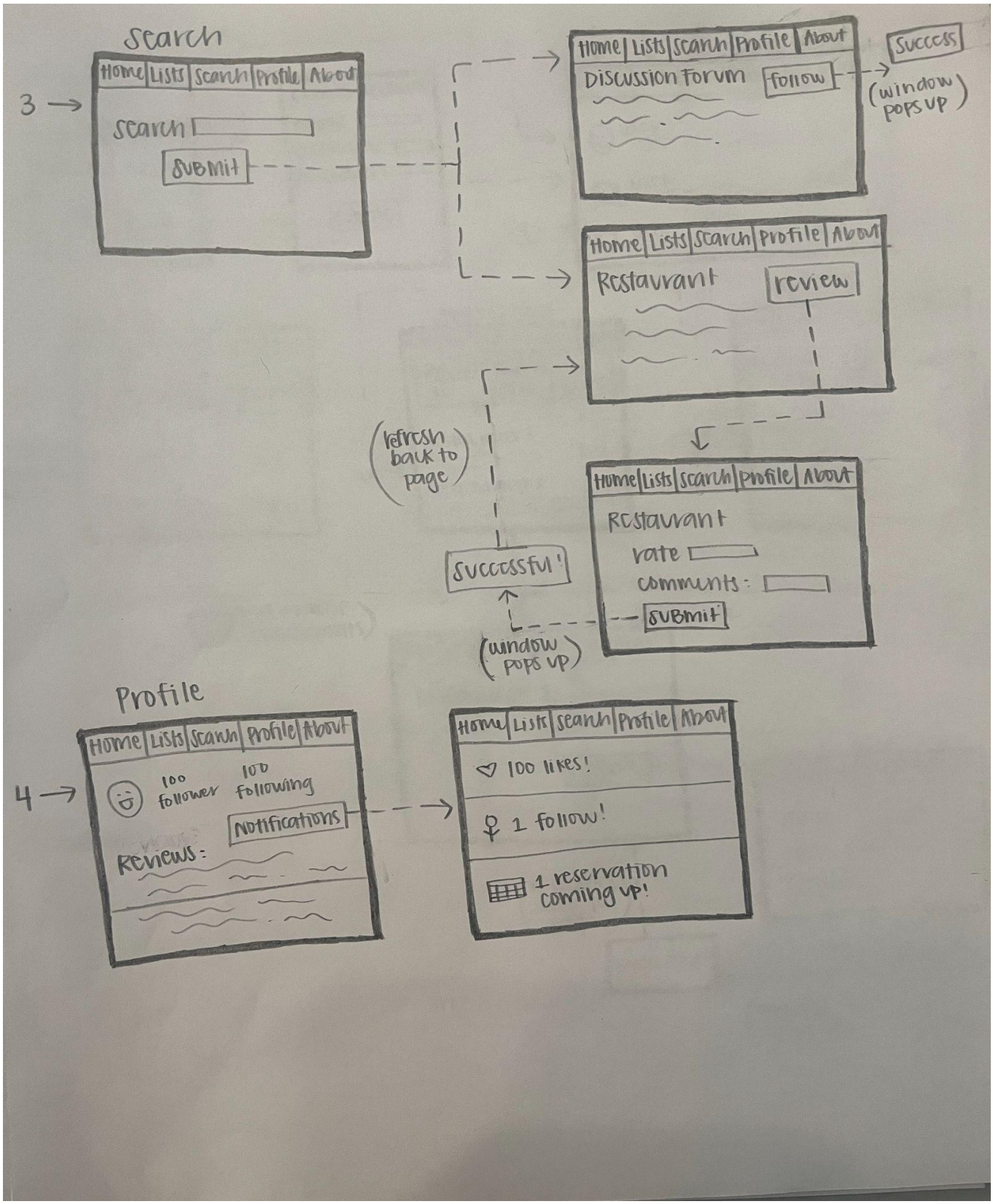
    13.1. A chat assistance agent shall be able to serve many users.

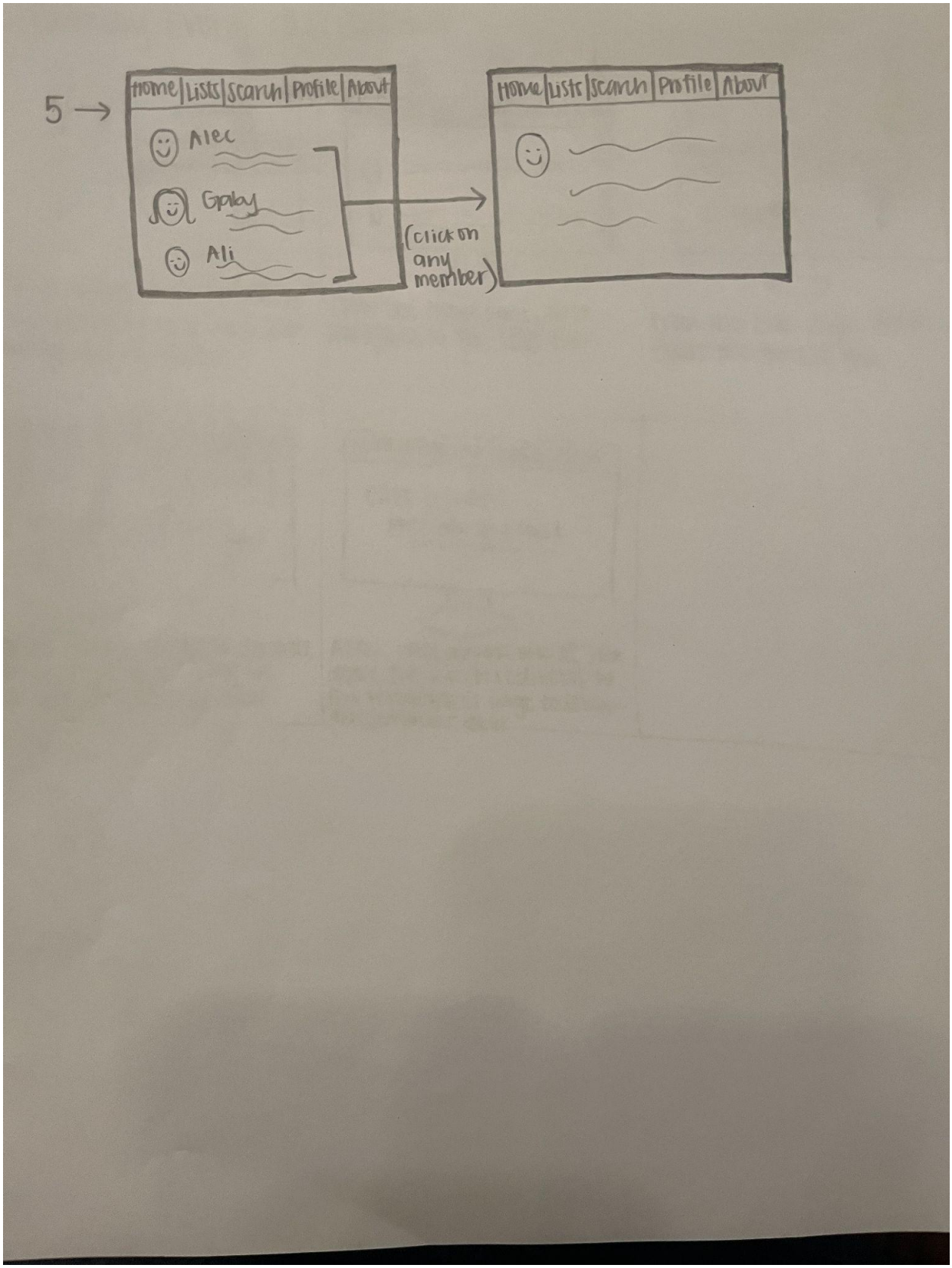
    13.2. A chat assistance service shall allow users to ask about restaurants and/or the cuisine they like.

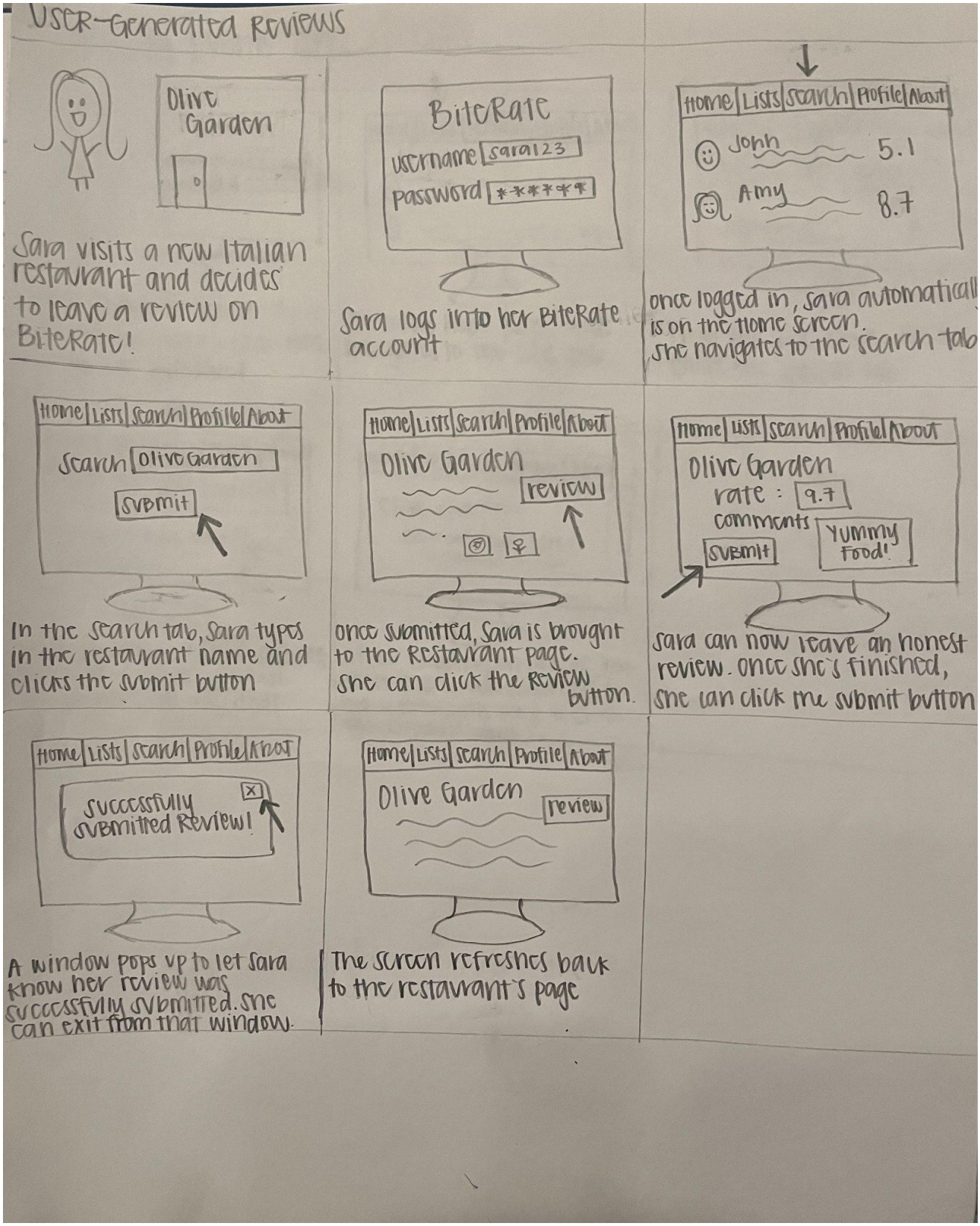
    13.3. A chat assistance service shall be able to recommend restaurants to users based on cuisine, restaurant name, or location.

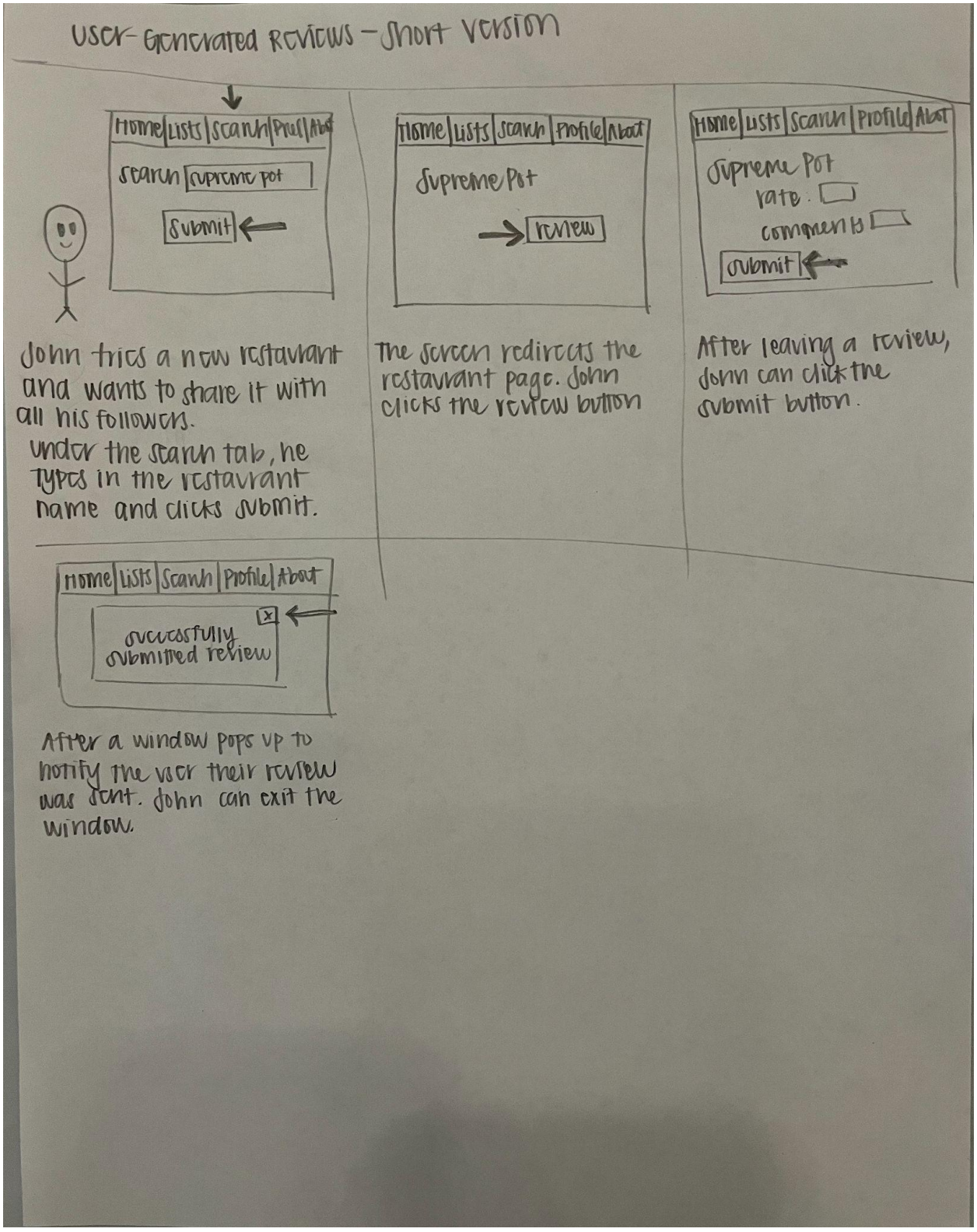
UI Mockups and Storyboards

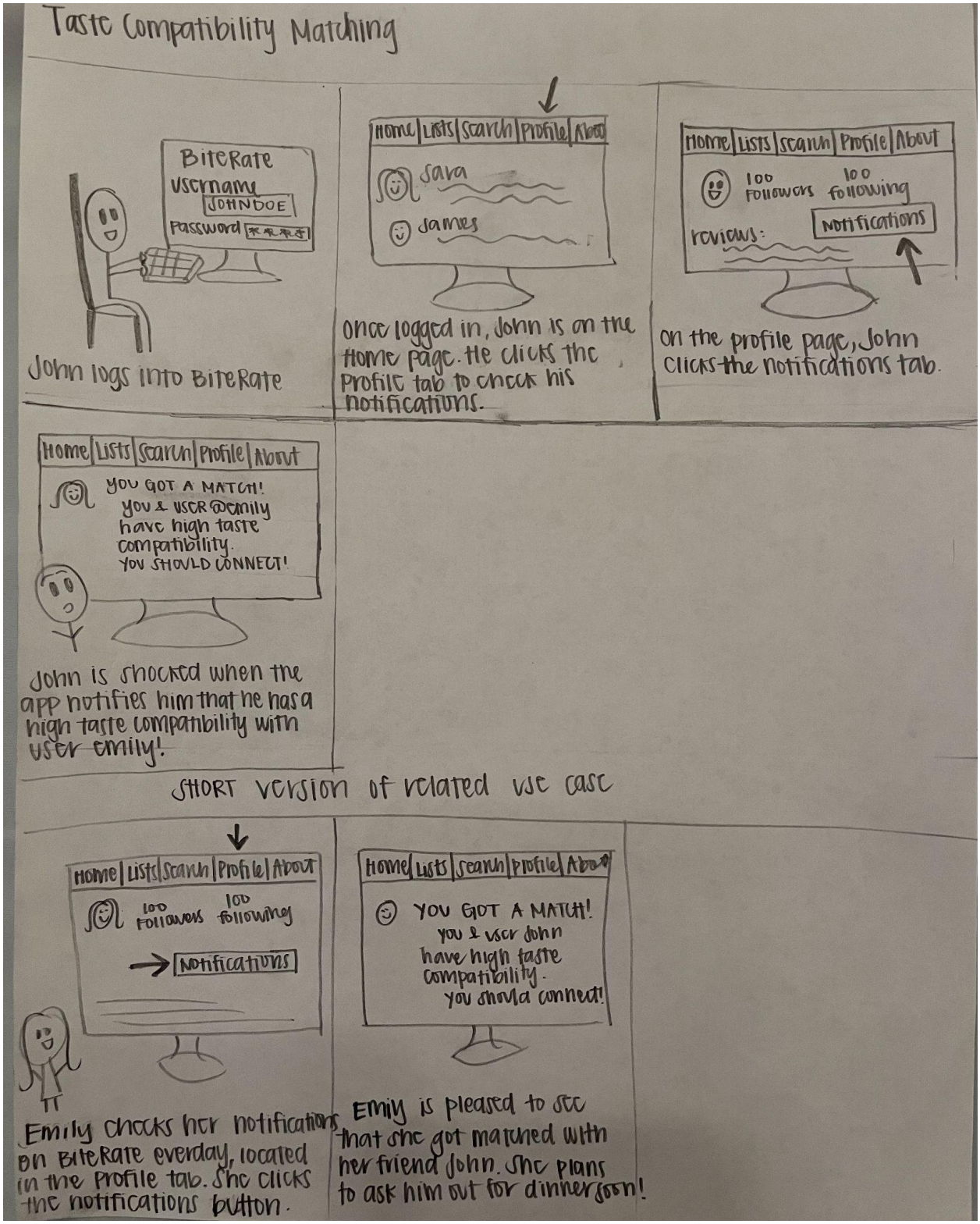


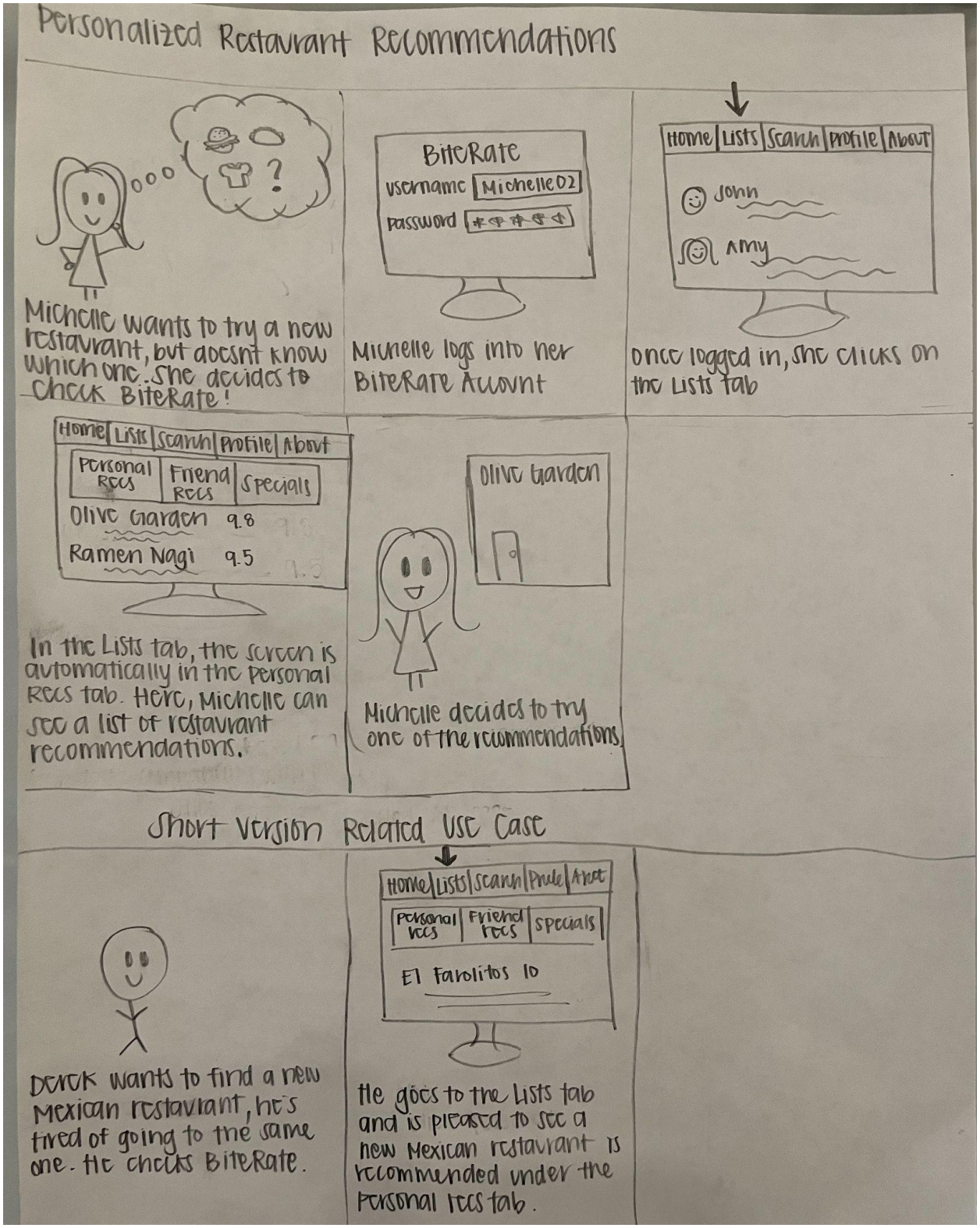

Search
Home Lists Search Profile About
3
search
submit
Home Lists Search Profile About
Discussion forum
follow
success
(window pops up)
Home Lists Search Profile About
Restaurant
review
(refresh back to page)
Home Lists Search Profile About
Restaurant
rate
comments:
submit
successful!
(window pops up)
Profile
Home Lists Search Profile About
4
100 followers
100 following
Notifications
Reviews:
Home Lists Search Profile About
100 likes!
1 follow!
1 reservation coming up!

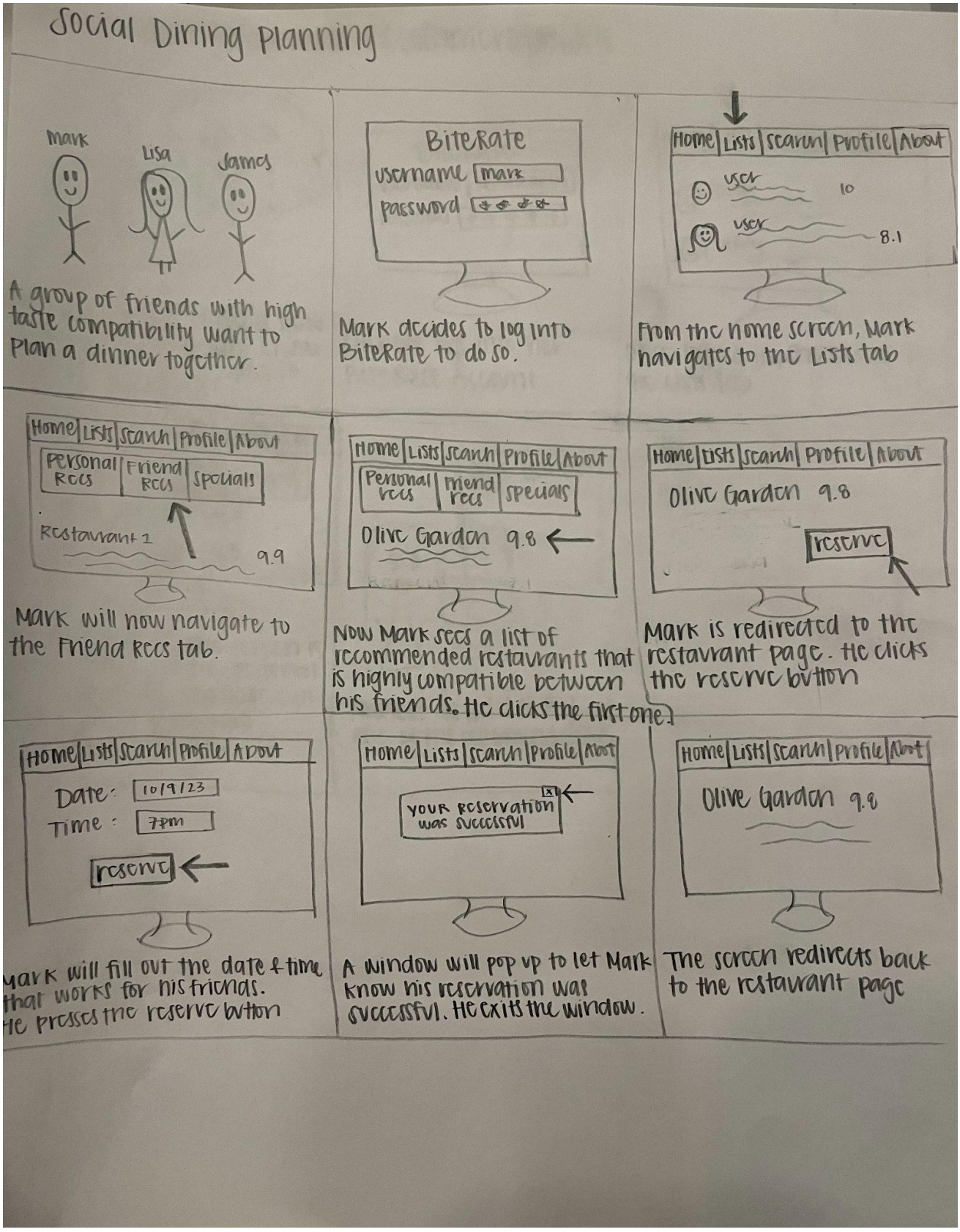

5
Home
Lists
Search
Profile
About
Alec
Gaby
Ali
(click on any member)
Home
Lists
Search
Profile
About

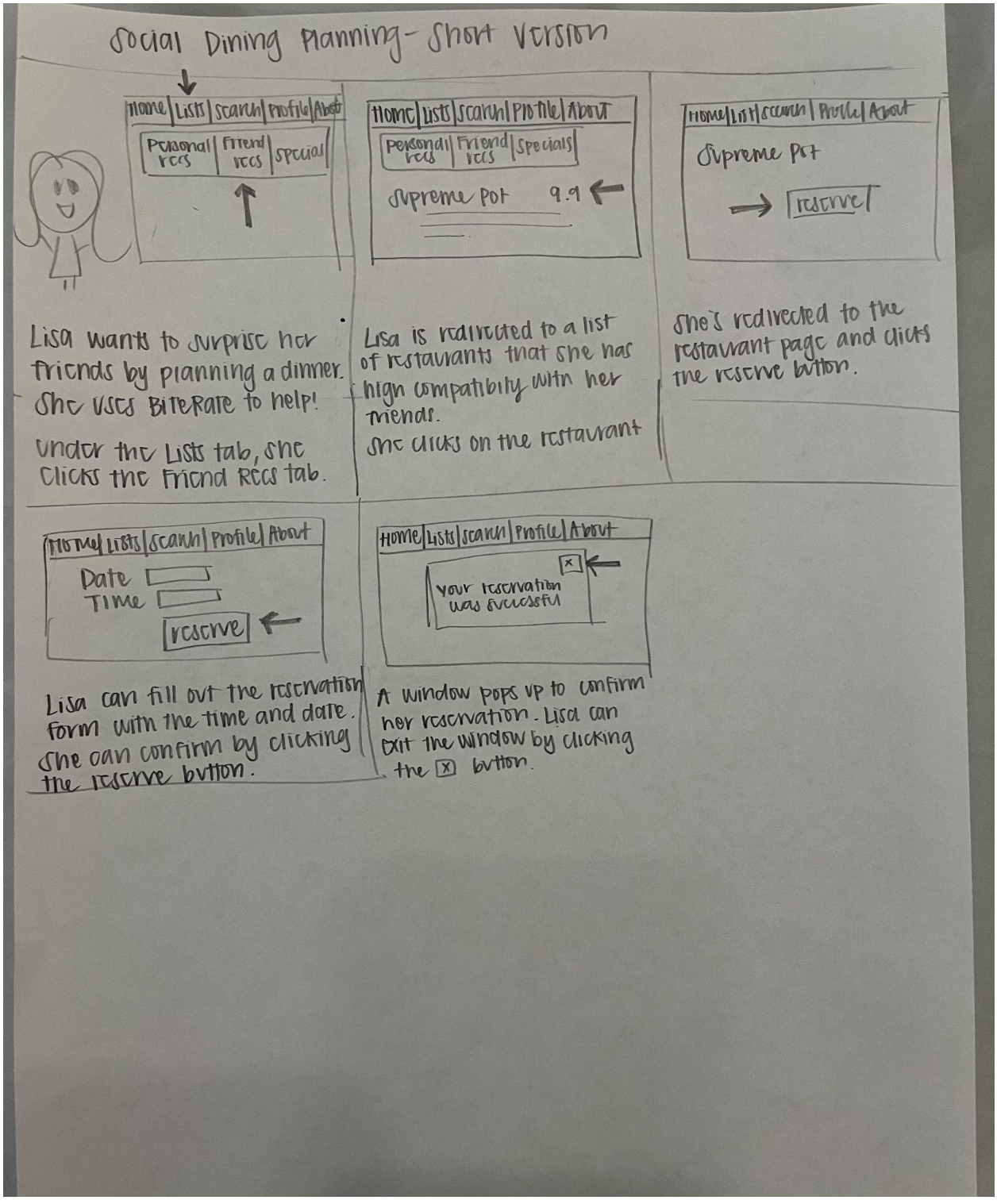

User-Generated Reviews
Olive Garden
Sara visits a new Italian restaurant and decides to leave a review on BiteRate!
BiteRate
username sara123
password ******
Sara logs into her BiteRate account
Home Lists Search Profile About
John 5.1
Amy 8.7
once logged in, sara automatically is on the Home screen. She navigates to the Search tab
Home Lists Search Profile About
Search Olive Garden
Submit
In the Search tab, Sara types in the restaurant name and clicks the submit button
Home Lists Search Profile About
Olive Garden
review
once submitted, Sara is brought to the Restaurant page. She can click the Review button.
Home Lists Search Profile About
Olive Garden
rate : 9.7
comments
Yummy Food!
Submit
Sara can now leave an honest review. Once she's finished, she can click the submit button
Home Lists Search Profile About
Successfully submitted Review!
A window pops up to let Sara know her review was successfully submitted. She can exit from that window.
Home Lists Search Profile About
Olive Garden
review
The screen refreshes back to the restaurant's page

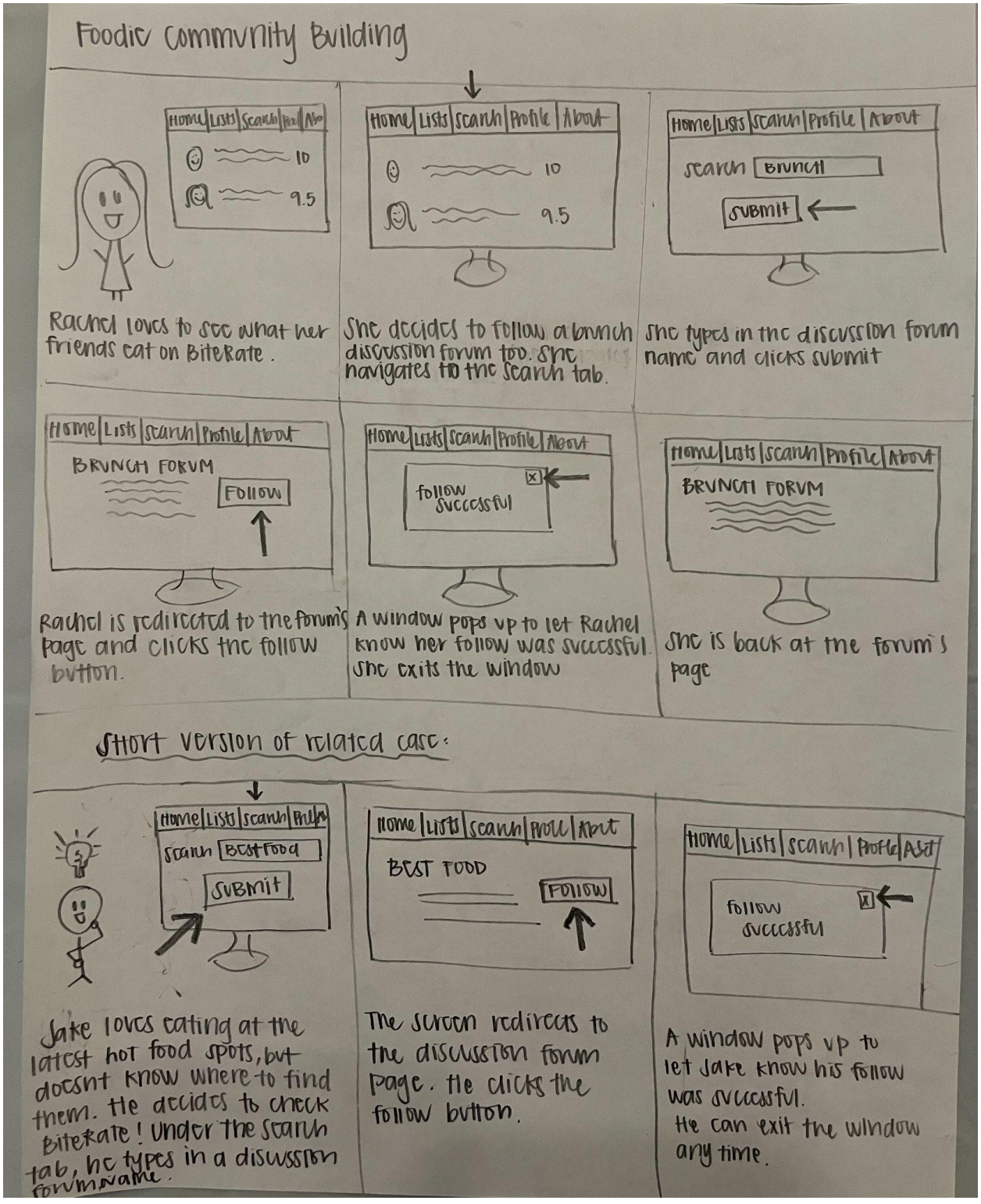

User-Generated Reviews - Short Version
Home Lists Search Profile About
Search Supreme Pot
Submit
John tries a new restaurant and wants to share it with all his followers.
under the search tab, he types in the restaurant name and clicks submit.
Home Lists Search Profile About
Supreme Pot
review
The screen redirects the restaurant page. John clicks the review button
Home Lists Search Profile About
Supreme Pot
rate
comments
submit
After leaving a review, John can click the submit button.
Home Lists Search Profile About
successfully submitted review
After a window pops up to notify the user their review was sent. John can exit the window.

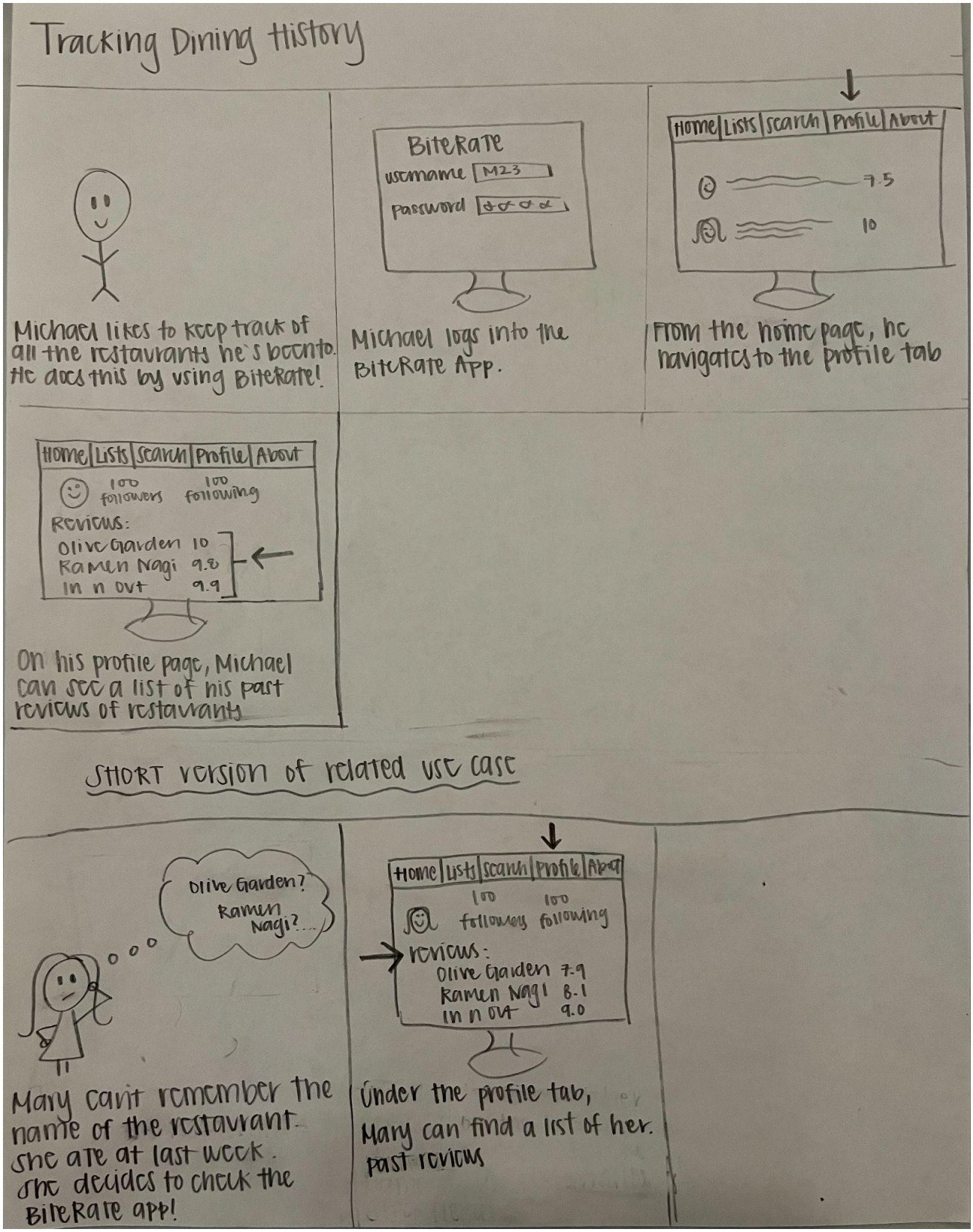

Taste Compatibility Matching
BiteRate
Username
JOHNDOE
Password
John logs into BiteRate
Home Lists Search Profile About
Sara
James
Once logged in, John is on the Home page. He clicks the Profile tab to check his notifications.
Home Lists Search Profile About
100 Followers
100 Following
Reviews:
Notifications
On the profile page, John clicks the notifications tab.
Home Lists Search Profile About
YOU GOT A MATCH!
You & User @Emily have high taste compatibility.
You should connect!
John is shocked when the app notifies him that he has a high taste compatibility with user Emily!
SHORT version of related use case
Home Lists Search Profile About
100 Followers
100 Following
Notifications
Emily checks her notifications on BiteRate everday, located in the Profile tab. She clicks the notifications button.
Home Lists Search Profile About
YOU GOT A MATCH!
You & User John have high taste compatibility.
You should connect!
Emily is pleased to see that she got matched with her friend John. She plans to ask him out for dinner soon!

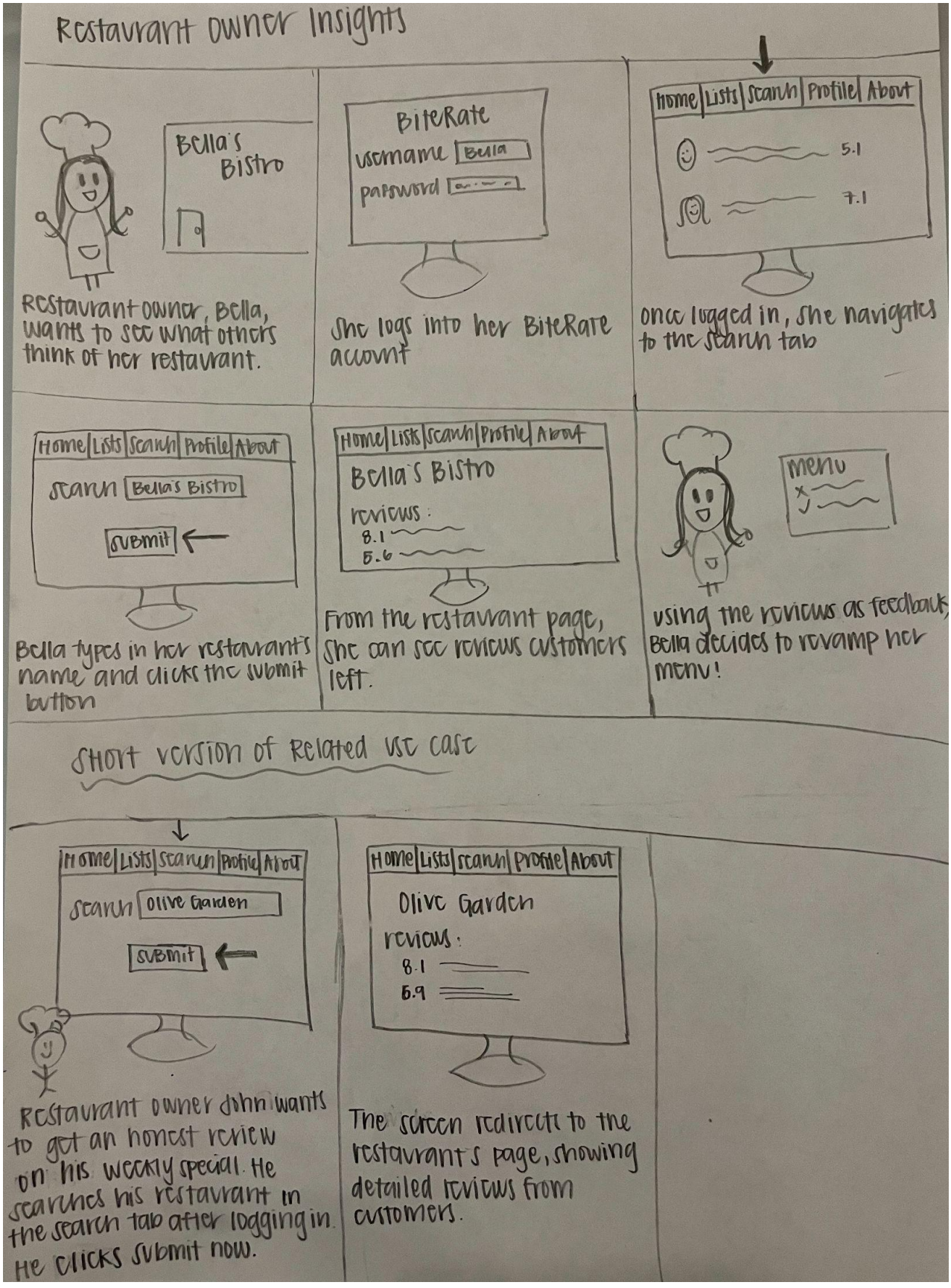

Personalized Restaurant Recommendations
Michelle wants to try a new restaurant, but doesn't know which one. She decides to check BiteRate!
BiteRate
username MichelleD2
password
Michelle logs into her BiteRate Account
Home Lists Search Profile About
John
Amy
Once logged in, she clicks on the Lists tab
Home Lists Search Profile About
Personal Recs
Friend Recs
Specials
Olive Garden 9.8
Ramen Nagi 9.5
In the Lists tab, the screen is automatically in the personal Recs tab. Here, Michelle can see a list of restaurant recommendations.
Olive Garden
Michelle decides to try one of the recommendations
Short Version Related Use Case
Derek wants to find a new Mexican restaurant, he's tired of going to the same one. He checks BiteRate.
Home Lists Search Profile About
Personal Recs
Friend Recs
Specials
El Farolitos 10
He goes to the Lists tab and is pleased to see a new Mexican restaurant is recommended under the Personal Recs tab.

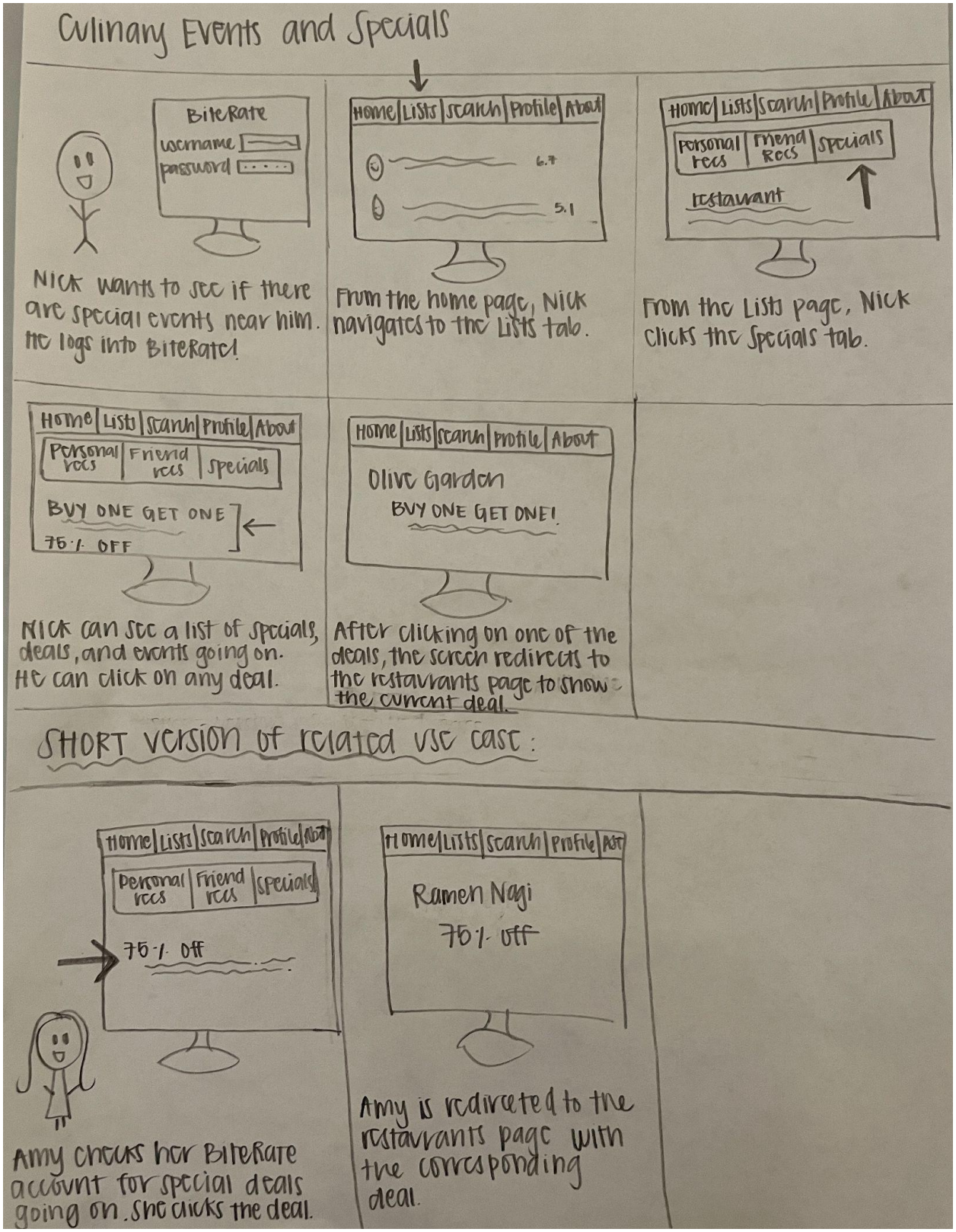

Social Dining Planning
Mark
Lisa
James
A group of friends with high taste compatibility want to plan a dinner together.
BiteRate
username mark
password
Mark decides to log into BiteRate to do so.
Home Lists Search Profile About
user 10
user 8.1
From the home screen, Mark navigates to the Lists tab
Home Lists Search Profile About
Personal Recs
Friend Recs
Specials
Restaurant 1 9.9
Mark will now navigate to the Friend Recs tab.
Home Lists Search Profile About
Personal recs
Friend recs
Specials
Olive Garden 9.8
Now Mark sees a list of recommended restaurants that is highly compatible between his friends. He clicks the first one.
Home Lists Search Profile About
Olive Garden 9.8
reserve
Mark is redirected to the restaurant page. He clicks the reserve button
Home Lists Search Profile About
Date: 10/9/23
Time: 7pm
reserve
Mark will fill out the date & time that works for his friends. He presses the reserve button
Home Lists Search Profile About
YOUR RESERVATION WAS SUCCESSFUL
A window will pop up to let Mark know his reservation was successful. He exits the window.
Home Lists Search Profile About
Olive Garden 9.8
The screen redirects back to the restaurant page


Social Dining Planning- Short Version
Home | Lists | Search | Profile | About
Personal Recs | Friend Recs | Specials
Supreme Pot 9.9
Supreme Pot
reserve
Lisa wants to surprise her friends by planning a dinner. She uses BiteRate to help!
Under the Lists tab, she clicks the Friend Recs tab.
Lisa is redirected to a list of restaurants that she has high compatibility with her friends.
She clicks on the restaurant
She's redirected to the restaurant page and clicks the reserve button.
Date
Time
reserve
Your reservation was successful
Lisa can fill out the reservation form with the time and date. She can confirm by clicking the reserve button.
A window pops up to confirm her reservation. Lisa can exit the window by clicking the x button.


Foodie community building
Home Lists Search Profile About
10
9.5
Rachel loves to see what her friends eat on BiteRate.
She decides to follow a brunch discussion forum too. She navigates to the search tab.
search BRUNCH
SUBMIT
She types in the discussion forum name and clicks submit
BRUNCH FORUM
FOLLOW
Rachel is redirected to the forum's page and clicks the follow button.
follow successful
A window pops up to let Rachel know her follow was successful. She exits the window
She is back at the forum's page
Short version of related case:
search BestFood
SUBMIT
Jake loves eating at the latest hot food spots, but doesnt know where to find them. He decides to check BiteRate! Under the search tab, he types in a discussion forum name.
BEST FOOD
FOLLOW
The screen redirects to the discussion forum page. He clicks the follow button.
follow successful
A window pops up to let Jake know his follow was successful.
He can exit the window any time.


Tracking Dining History
Michael likes to keep track of all the restaurants he's been to. He does this by using BiteRate!
BiteRate
username M23
password
Michael logs into the BiteRate App.
Home Lists Search Profile About
7.5
10
From the home page, he navigates to the profile tab
Home Lists Search Profile About
100 followers
100 following
Reviews:
Olive Garden 10
Ramen Nagi 9.8
In n Out 9.9
On his profile page, Michael can see a list of his past reviews of restaurants
SHORT version of related use case
Olive Garden?
Ramen Nagi?...
Mary can't remember the name of the restaurant she ate at last week. She decides to check the BiteRate app!
Home Lists Search Profile About
100 followers
100 following
Reviews:
Olive Garden 7.9
Ramen Nagi 8.1
In n Out 9.0
Under the profile tab, Mary can find a list of her past reviews


Restaurant owner insights
Bella's Bistro
Restaurant owner, Bella, wants to see what others think of her restaurant.
BiteRate
username Bella
password
She logs into her BiteRate account
Home Lists Search Profile About
5.1
7.1
once logged in, she navigates to the search tab
Home Lists Search Profile About
search Bella's Bistro
submit
Bella types in her restaurant's name and clicks the submit button
Home Lists Search Profile About
Bella's Bistro
reviews:
8.1
5.6
From the restaurant page, she can see reviews customers left.
menu
using the reviews as feedback, Bella decides to revamp her menu!
short version of Related use case
Home Lists Search Profile About
search Olive Garden
submit
Restaurant owner John wants to get an honest review on his weekly special. He searches his restaurant in the search tab after logging in. He clicks submit now.
Home Lists Search Profile About
Olive Garden
reviews:
8.1
5.9
The screen redirects to the restaurant's page, showing detailed reviews from customers.


Culinary Events and Specials
BiteRate
username
password
Nick wants to see if there are special events near him. He logs into BiteRate!
Home | Lists | Search | Profile | About
6.7
5.1
From the home page, Nick navigates to the Lists tab.
Home | Lists | Search | Profile | About
Personal Recs | Friend Recs | Specials
restaurant
From the Lists page, Nick clicks the Specials tab.
Home | Lists | Search | Profile | About
Personal Recs | Friend Recs | Specials
BUY ONE GET ONE
75% OFF
Nick can see a list of specials, deals, and events going on. He can click on any deal.
Home | Lists | Search | Profile | About
Olive Garden
BUY ONE GET ONE!
After clicking on one of the deals, the screen redirects to the restaurants page to show the current deal.
SHORT version of related use case:
Home | Lists | Search | Profile | About
Personal Recs | Friend Recs | Specials
75% off
Amy checks her BiteRate account for special deals going on. She clicks the deal.
Home | Lists | Search | Profile | About
Ramen Nagi
75% off
Amy is redirected to the restaurants page with the corresponding deal.

High-Level Database Architecture and Organization

Database Requirements

- A user shall be able to create zero or many accounts.
- A registered user shall be able to rate zero or many restaurants.
- A registered user shall be able to post zero or one review for each restaurant.
- A registered user shall be able to do zero or one recommendation for each restaurant.
- A registered user shall be able to follow zero or many users.
- A restaurant account shall be able to post zero or one restaurant.
- A restaurant shall be able to post zero or many images.

Entity, Attributes, and Relationship Description

1. ***User (Strong)***
    - UserID: key, numeric
    - LastVisit: multivalue, timestamp

    ` Related to Account by one-to-many relationship.

2. ***Account (Strong)***
    - AccountID: key, numeric
    - User: key, numeric
    - Password: alphanumeric
    - FirstName: alphanumeric
    - LastName: alphanumeric
    - Email: alphanumeric
    - Address: key, alphanumeric
    - Phone: numeric
    - DateCreated: timestamp

    Related to User by many-to-one relationship.
    Related to Address by many-to-one relationship.
    Related to ChatAssistance by many-to-many relationships.
    Related to Admin, BasicAccount, RestaurantAccount by an inheritance one-to-one relationship.

3. ***Admin (Strong)***
    - AdminID: key, numeric

- Account: key, numeric
- Role: alphanumeric
- StartDate: timestamp
- EndDate: timestamp

Related to Account by an inheritance one-to-one relationship.

4. ***Basic Account (Weak)***
    - BasicAccountID: key, numeric
    - Account: key, numeric
    - BiteScore: numeric

    Related to Account by an inheritance one-to-one relationship.
    Related to RestaurantRecommendation, RestaurantRating, and RestaurantReview by many-to-many relationships.
    Related to Activity by many-to-many relationships.
    Followed by many-to-many relationships.

5. ***RestaurantAccount (Weak)***
    - RestaurantAccountID: key, numeric
    - Account: key, numeric

    Related to Account by an inheritance one-to-one relationship.
    Related to Restaurant as an associative entity between Restaurant and BasicAccount.

6. ***RestaurantOwner (Weak)***
    - RestaurantOwnerID: key, numeric
    - RestaurantAccount: key, numeric
    - Restaurant: key, numeric

    Related to Restaurant and BasicAccount as an associative entity.

7. ***Restaurant (Strong)***
    - RestaurantID: key, numeric
    - Name: alphanumeric
    - Cuisine: key, numeric
    - Address: key, numeric
    - OpenDate: timestamp
    - PostDate: timestamp

- Images: Array???

  Related to Address by many-to-one relationship.
  Related to RestaurantRecommendation, RestaurantRating, and RestaurantReview by many-to-many relationship.
  Related to Cuisine by many-to-one relationship.

8. RestaurantRating (Weak)

- RestaurantRatingID: key, numeric
- BasicAccountID: key, numeric
- Restaurant: key, numeric
- Rating: numeric
- RateDate: timestamp

9. RestaurantReview (Weak)

- RestaurantReviewID: key, numeric
- BasicAccountID: key, numeric
- Restaurant: key, numeric
- ReviewComment: alphanumeric
- ReviewDate: timestamp

10. RestaurantRecommendation (Weak)

- RestaurantRecommendationID: key, numeric
- BasicAccountID: key, numeric
- Restaurant: key, numeric

11. RestaurantImages (Weak)

- Image: key, numeric
- Restaurant: key, numeric
- Timestamp: timestamp

12. ImageURL (Strong)

- ImageID: key, numeric
- ImageURL: alphanumeric

13. Cuisine (Strong)

- CuisineID: key, numeric
- Name: alphanumeric
- CountryOrigin: alphanumeric

Related to Restaurant by a one-to-many relationship.

14. Address (Strong)

- AddressID: key, numeric
- AddressLine1: alphanumeric
- AddressLine2: alphanumeric
- City: alphanumeric
- State: alphanumeric
- Country: alphanumeric
- PostalCode: numeric

Related to Account by one-to-many relationship.
Related to Restaurant by one-to-many relationship.

15. UserActivity (Weak)

- UserActivityID: key, numeric
- BasicAccountID: key, numeric
- Activity: key, numeric

Related to BasicAccount and Activity acting as an associative entity between them.

16. Activity (Strong)

- ActivityID: key, numeric
- Name: alphanumeric

Related to BasicAccount by many-to-many relationships.

17. Follow (Strong)

- FollowerID: key, numeric
- BasicAccountID: key, numeric
- FollowerBasicAccountID: key, numeric
- DateFollow: timestamp

Related to BasicAccount by many-to-many relationships.

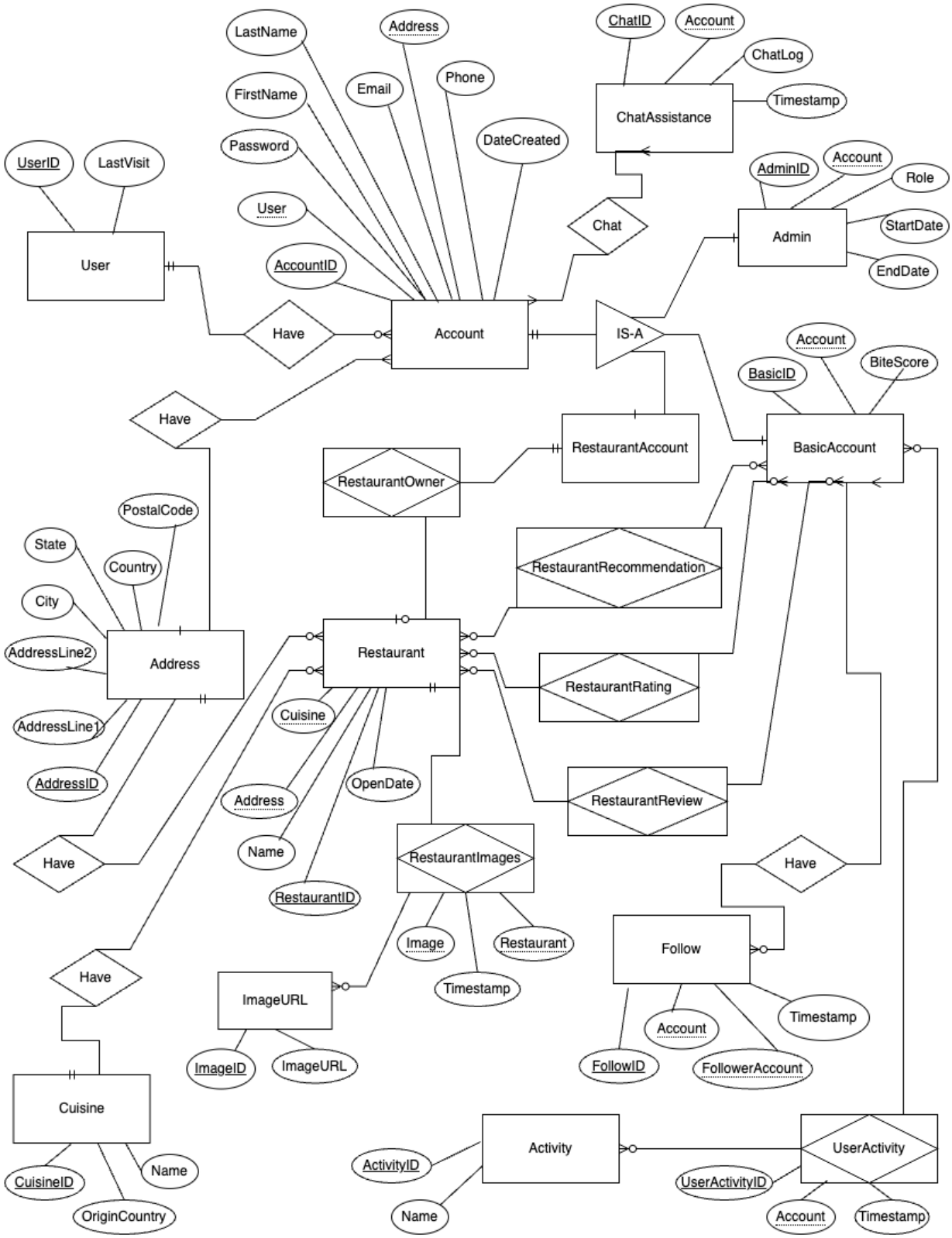
18. ChatAssistance (Strong)

- ChatID: key, numeric

- Account: key, numeric
- ChatLog: alphanumeric
- Timestamp: timestamp

  Related to Account by many-to-many relationship.

Entity Relationship Diagram (ERD)



What DBMS We Will Use

Based on what our team had discussed, all of us are more familiar with using mySQL for our relational database. And this is the main reason we decided to go with mySQL.

Media Storage

Based on what our team had discussed, we shall use DB BLOB to store images. But for this Milestone 2 checkpoint 2, Vertical SW Prototype, we have decided to keep it simple by storing images in the file system.

Search/Filter Architecture And Implementation

1. Search Algorithm
    - We shall mainly use mySQL 'LIKE' clauses for basic string/substring matching in our search.
2. Organization For User
    - We shall have a search bar on the home page to allow users to enter keywords to search for restaurants/users.
    - In addition to the search bar, we shall have optional filters next to the search bar for a more specific result.
3. Database Attributes For Search (Primary Search Field)
    - A user's first/last name.
    - A restaurant name.
    - A cuisine type.
    - A location or area.
4. MySQL Code Implementation/Examples

    When a user wants to search for a particular restaurant by name:

    ```
    SELECT * FROM restaurant WHERE name LIKE '%search_term%';
    ```

    When a user wants to search for a particular user by name:

    ```
    SELECT * FROM user WHERE name LIKE '%search_term%';
    ```

High-Level APIs and Main Algorithms

1. ***User Management API:***
- This API allows users to create accounts, log in, and manage their profiles.
- It includes features for user authentication, account creation, and profile editing.
- Additionally, it provides endpoints for searching and connecting with other users.

2. ***Restaurant Management API:***
- This API facilitates the creation, modification, and retrieval of restaurant-related information.
- Users can use this API to search for restaurants, view restaurant profiles, and submit ratings, reviews, and recommendations.

3. ***Point/Score System API:***
- The Point/Score API tracks user activities and calculates scores/points.
- It includes logic to assign scores for actions like rating, reviewing, and recommending restaurants.
- The API also provides a leaderboard to display user rankings.

4. ***Recommendation Engine:***
- This algorithm analyzes user behavior, such as ratings and reviews, to provide restaurant recommendations.
- It considers user preferences, historical data, and restaurant ratings to suggest suitable dining options.

5. ***User Activity Tracking API:***
- This API records user interactions and activities within the application.
- It stores data related to user actions, allowing users to see their own history and enabling the visibility of certain activities to friends/followers.

6. ***Chatbot Integration:***
- The chatbot integrates with a chat assistance service.
- It uses natural language processing (NLP) and algorithms to understand user queries about restaurants and cuisines.
- The chatbot then recommends restaurants based on user preferences and location.

7. ***Notification Service:***
- This API handles the generation and delivery of notifications to users.
- It sends notifications for events such as new messages, friend requests, restaurant recommendations, and updates on user reviews.

8. ***Ranking Algorithm:***

- The ranking algorithm calculates a restaurant's overall rating by averaging user ratings.
- It ensures that ratings are appropriately weighted and considers factors like recency and the number of ratings.

9. ***Geolocation API:***

- The geolocation API assists in determining a user's location and finding nearby restaurants.
- It may use geospatial algorithms and data to provide accurate location-based services.

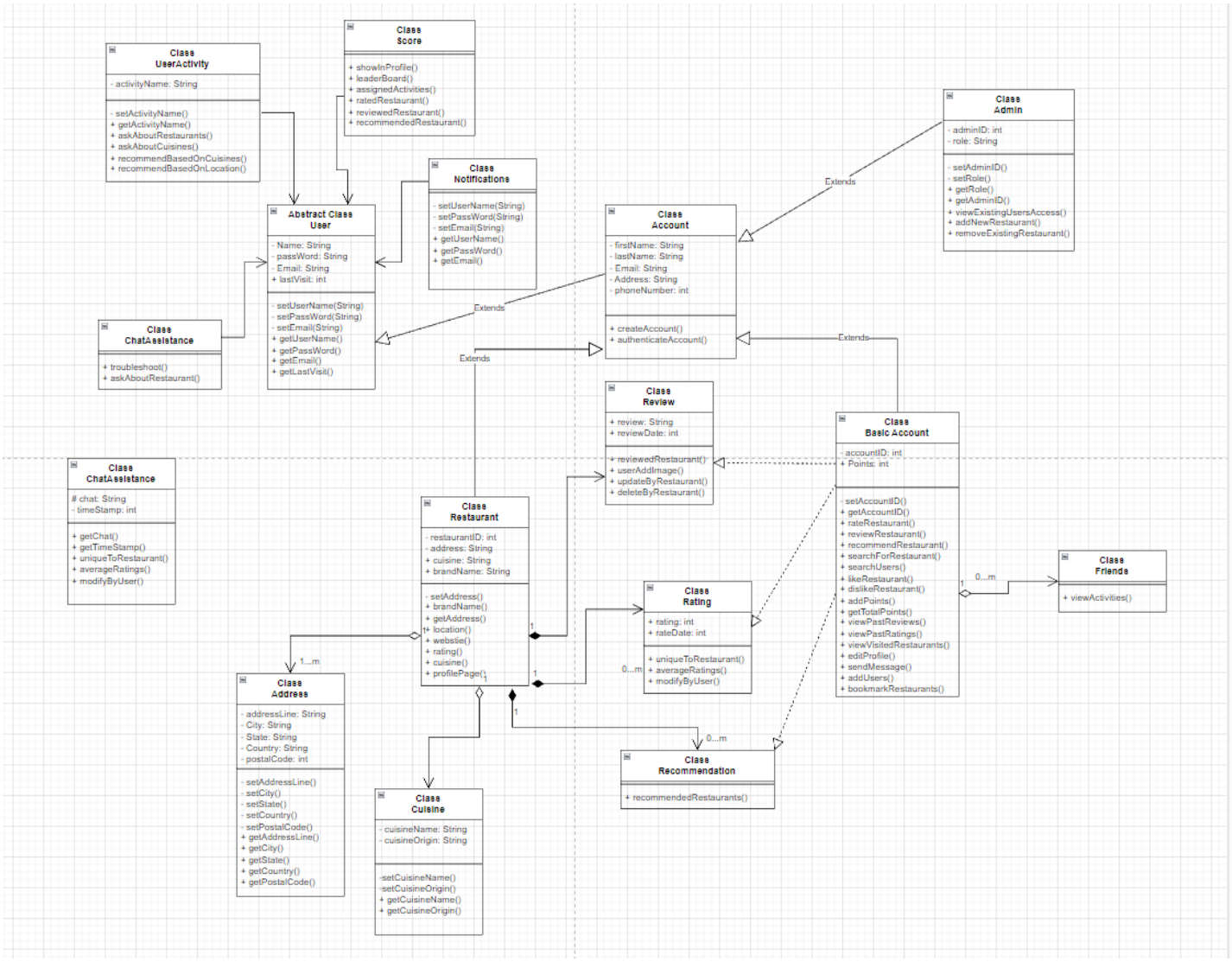
10. ***Data Storage and Retrieval:***

- The application may utilize MySQL (8.1) as the database system to store and retrieve user data, restaurant information, reviews, and more.

11. ***User Interaction API:***

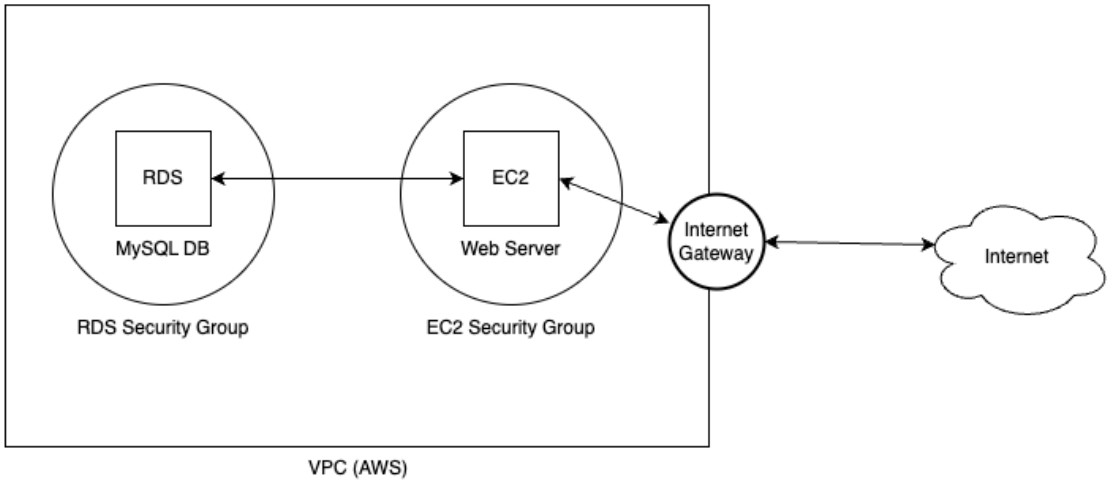
- This API manages interactions between users, including friend requests, messaging, and following other users.
- It facilitates communication and social networking within the application.

High-Level UML Diagrams

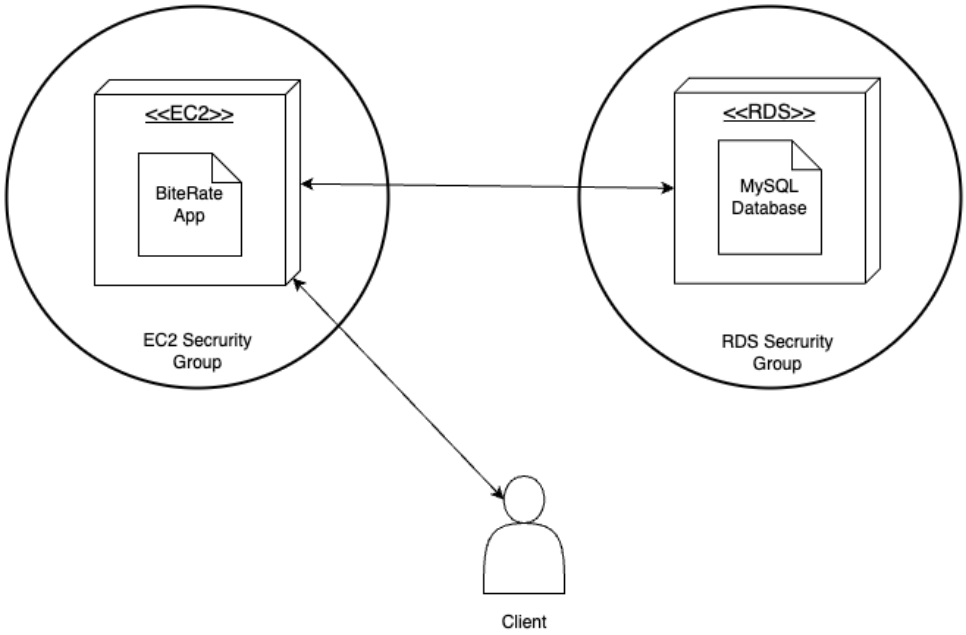


High-Level Application Network and Deployment Diagrams

Application Network Diagram



Deployment Diagram



Identify *Actual* Key Risks for the Project

Skills Risks

1. **Inadequate Documentation Skills**

Risk: Our team has a lack of experience in documenting, as for most of us, this is our first time documenting the progression of projects. This is an actual risk since we don't possess the right skills to give adequate documentation, which can lead to a deduction of points on our milestone.

Solution: To address this issue, our team is using the feedback given on Milestone 1 to evaluate the mistakes made in our past work. By doing so, we will improve in the future. We are also carefully reading the instructions on the Milestone 2 assignment, to ensure we will not miss any requirements.

2. **Using an Unfamiliar Framework**

Risk: The framework our team is using for the frontend of our web application is Angular. This particular framework is unfamiliar to most of our team as we mostly worked with React. This can cause bugs/errors in our program since we don't have the knowledge to implement a web application using Angular.

Solution: Our team made the decision to learn the basics of Angular by watching tutorials online and reading the necessary documentation. Due to the project's time constraint, our team decided to learn only the basic functionalities of Angular, instead of diving deep into all its complexities. By doing so, we have enough knowledge to get our project into production while keeping in mind the project's time constraint.

Schedule Risks

1. **Schedule Conflicts Related to School:**

Risk: Most of the team members have different class schedules which causes our free time to overlap. This makes it difficult to schedule additional team meetings to work on the project. This is an actual risk because this can cause a delay in production for our project.

Solution: The team has a set meeting time every Wednesday, which is mandatory. For additional team meetings, our team decided to meet via Discord as needed. This ensures that we have enough time to finish our project.

2. **Underestimating Time Needed**

Risk: For Milestone 1, our team made the mistake of underestimating the time needed to finish the milestone. Our team is at risk of making the same mistake, where we do not allocate

enough time to finish all parts of the project. This is an actual risk because it can cause a deduction of points in our Milestone.

Solution: Our team regularly updates each other on the progress of their work through Discord. Also, our team lead assigned a one-week deadline for the documentation, to ensure that we have enough time to work on the prototype. By managing our time wisely, we have enough time to finish the project.

Technical Risks

1. **AWS/MySQL Connection Issue**

Risk: Our team has recurring issues connecting our AWS host to the MySQL database. For this reason, we had a project delay since the database needed to be connected before we could start integrating other parts.

Solution: Our team discussed the issue during class, through Discord, and during team meetings. We accessed the issue, pinpointed the problem, and came up with ideas to resolve it.

2. **Integration Risks**

Risk: For the prototype, our team split up the work between the backend and frontend. This may cause problems in integrating the separated software into one, causing bugs and errors.

Solution: Although we cannot predict integration errors, we can try our best to solve them earlier if it happens. Our team decided to start integrating the frontend and backend early before the due date, to solve any errors if they arise. By doing so, we will have enough time to resolve it.

Teamwork Risks

1. **Missing Team Meetings**

Risk: During team meetings, we exchange important information regarding the project. Missing a team meeting can throw the members out of sync causing miscommunication on the project's objectives and individual responsibilities.

Solution: Our team agreed on a mandatory team meeting every Wednesday. We also hold extra team meetings as needed. In addition to team meetings, our group is in constant contact to relay any questions, concerns, or information we may have.

2. **Failing to Finish Responsibilities**

Risk: Each member is given their own task to finish by the due date, but there is a potential risk of someone missing their individual deadline. This will cause a delay in the production of the project.

Solution: Our team lead assigned roles and tasks early on in the project, so each individual has a clear idea of the assignment. Our team also messages each other via Discord on the progress of our tasks. We also hold weekly team meetings to discuss the overall progression of our project.

Legal/Content Risks

1. **Copyright Violations**

Risks: Our team is at risk of copyright issues when gathering the necessary resources for the project. One example could be gathering pictures to display on our web application.

Solution: Our team decided to be careful when downloading images from the internet as most pictures are copyrighted. To do this, we used websites like Unsplash and Pexels to download copyright-free images.

Project Management

For M2 tasks, we primarily utilized Notion, a project management and collaboration platform. Notion allowed us to create a structured workspace where we could document project goals, functional requirements, and task lists. It served as a central hub for organizing and tracking our progress throughout Milestone 2 (M2).

We used Notion's features, such as tables, Kanban boards, and task databases, to maintain a clear overview of our tasks, assign responsibilities, and monitor the status of each task. This provided us with a collaborative environment to discuss and prioritize work efficiently. Moving forward, we recognize the importance of adopting a unified task management system that provides a holistic view of all tasks and their statuses. We plan to also integrate Trello.

Trello offers an intuitive dashboard view that aligns well with our task management needs. It enables us to create boards for different aspects of the project, assign tasks to team members, set deadlines, and track progress visually. By implementing Trello, we aim to streamline task management and improve transparency across the team, ensuring that future tasks are planned, executed, and monitored effectively.

Detailed List of Contributions

Alec Nagal: I worked on API/Main Algorithms, Project management, and the frontend of our application

Ali Alsharif: Worked on UML diagram and frontend of our application 10/10

Gerry Putra: Worked on High level database, network/ Deployment, backend of application 10/10

Gabriella Arcilla: UI mockups, identify risks, revisions for milestone 1 document and milestone 2 document 10/10

Kenneth Pang: Prioritize functional requirements, database 10/10

Robel Ayelew: Data definitions and frontend of application 10/10

James Lu: Revision milestone 1 document 10/10

Robin Reyes: Revision milestone 1 document, main use case diagram 10/10