

KaChow Now App - Final Report

Fall Term 2022

Group #17

Team Leader: Nathan Le (#300227651)

Aaditya Shah(#300236230)

Bo Lin Chen(#300236550)

Kevin Dang(#300241133)

Sami Hassan(#300169285)

Siva Senthilkumaran(#300230364)

Table of Contents

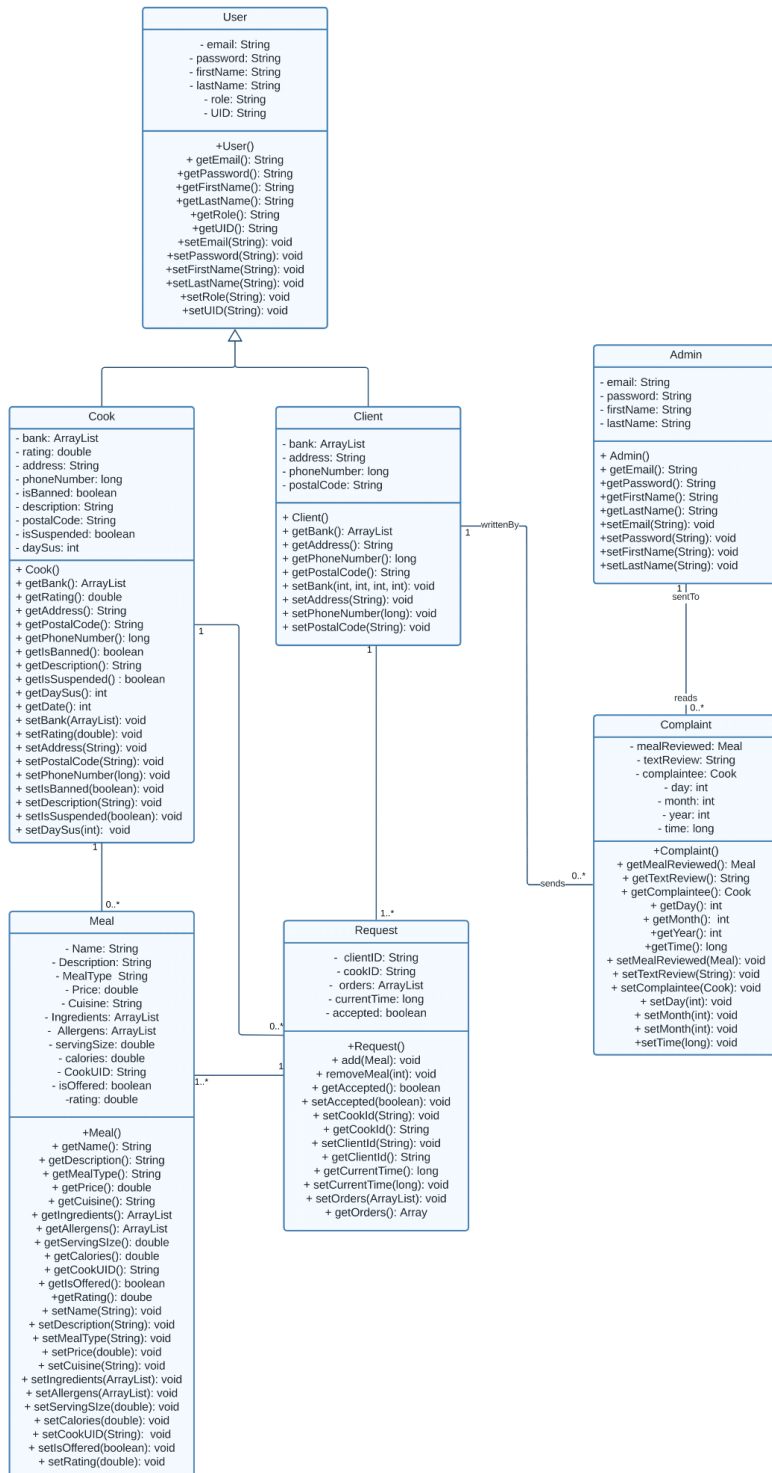
Introduction:	3
UML Diagram:	4
Contributions:	5-9
Deliverable 1:	5
Nathan Le's Contributions:	5
Aaditya Shah Contributions:	5
Bo Lin Chen Contributions:	5
Kevin Dang's Contributions	5
Siva Senthilkumaran's Contributions:	5
Sami Hassan's Contributions	5
Deliverable 2:	6-7
Nathan Le's Contributions:	6
Aaditya Shah's Contributions:	6
Bo Lin Chen Contributions:	6
Kevin Dang's Contributions	7
Siva Senthilkumaran's Contributions:	7
Sami Hassan's Contributions:	7
Deliverable 3:	7-8
Nathan Le's Contributions:	7
Aaditya Shah's Contributions:	7
Bo Lin Chen's Contributions:	7
Kevin Dang's Contributions	8
Siva Senthilkumaran's Contributions:	8
Sami Hassan's Contributions:	8
Deliverable 4:	8-9
Nathan Le's Contributions:	8
Aaditya Shah's Contributions:	8
Bo Lin Chen's Contributions:	8
Kevin Dang's Contributions	9
Siva Senthilkumaran's Contributions:	9
Sami Hassan's Contributions:	9
Screenshots:	10-21
Lessons Learned:	22-23

Introduction:

KaChow Now is an application designed for local cooks to upload their meals and sell them to clients from home. The application supports three types of users: clients, cooks, and administrators. Clients are the ones who will order the meals from cooks. They are able to search for a specific meal and view the meal's description such as calories and allergens. Once they have ordered a meal, they can see the status of their order which can range from: pending, approved or rejected. If the client likes the meal, they can view the chef's profile and give them a rating or if they dislike the meal then, the client can send a complaint.

Cooks are able to add and delete meals to their menu. Also, cooks have the ability to offer their meals to clients for them to order. Once they order, cooks can view incoming purchase requests from clients and have the option to approve or deny them. Administrators are premade accounts that receive the complaints about the cooks from the clients and can either suspend cooks permanently or temporarily.

UML Diagram:



Contributions:

Deliverable 1:

Nathan Le's Contributions:

- Signup UI
- Displays different edit text boxes depending on which user is trying to sign up. If no role is selected, some text boxes are hidden
- Error checking on signup button, validation of each type of information, and restrictions on each edit text box
- Error checking for login button
- Created the signup button and implemented it to create a user account on click
- Created the login Button and made listeners
- Created User Class
- When entering postal code or phone number and when the text box reaches its limit, you would automatically start entering text in the adjacent box

Aaditya Shah Contributions:

- Started initial design on UML
- Added Firebase database and Firebase authentication dependencies
- Enabled client authentication information to be saved and validated using authentication API
- Enabled client and cook information to be saved and retrieved from the Firebase database
- Logged in clients in the backend
- Visualized potential Database design/hierarchy to allow clients and cooks to store information while not overlapping with each other

Bo Lin Chen Contributions:

- UI design for the necessary screens needed, for example the home screen.

Kevin Dang's Contributions

- Designed and updated the welcome screen to display the role of the person who logged in
- Setup client java class
- Updated UI colors so they all follow the same color scheme

Siva Senthilkumaran's Contributions:

- Made the spinner for the app
- Tested the UI Manually, looking for inconsistencies in font/styling/text
- Managed the website data entries on Firebase

Sami Hassan's Contributions

- Created the Cook, Client and Admin classes with their methods.
- Updated welcome screen for Deliverable 1 (scrapped for further deliverables).
- Designed UML class diagram.

- README file on Github.

Deliverable 2:

Nathan Le's Contributions:

- Created Admin Page activity and designed basic UI
- Set up logout button in admin page
- Created adding meal page with all text boxes
- Created Report Button
- Displaying meals on the cook profile from client account
- Created ComplaintList adapter and used it to display complaints in the admin page
- Error checking on complaints
- Implemented dismiss, ban, and suspend buttons
- Created the offered meals activity from the client account and set a listener for the chef icon to bring client to the cook profile
- Implemented suspension of cooks methods and updating login display according to if the cook is banned or suspended.
- Displaying to the suspended cook how many days they are suspended for and to the banned cook that they are banned

Aaditya Shah's Contributions:

- CircleCI integration
- Added Firebase storage dependency
- Enabled user and cooks to submit profile pictures
- Changed Cook, Client, Admin, Meal, and Complaints class to save it in the database more efficiently
- Created Recyclerview adapter to display Pictures of chefs horizontally that move you to cooks page where you can see meals
- Created list adapter for admin complaints and connected with its corresponding layout
- Designed hierarchy for Firebase storage organization
- Retrieved and saved pictures to Firebase Storage
- Amended Database design/hierarchy to allow Clients to save submitted complaints and to allow Admins to access
- Clients can rate cooks and the average of their ratings are stored in the Database
- Cooks can view their profile and rating
- Allowed admins to suspend or ban cooks by connecting complaints to specific cooks
- Created system to automatically unsuspend a cook after they have been suspended for that many days as well as not allow banned cooks to login or create new accounts
- Validated inputs to submit complaint page
- Saved Created meals in database using updated Database design/hierarchy

Bo Lin Chen Contributions:

- Made the admin inbox page where the list of complaints will show up

- Made necessary UI pages/dialogs that were needed.

Kevin Dang's Contributions

- Designed the cook's menu page
- Implemented scroll view function on the cook's menu page
- Created the popup dialog for which admins will see when banning and suspending cooks

Siva Senthilkumaran's Contributions:

- Managed the Java Unit Tests for the classes in the app
- Tested the UI Manually, looking for inconsistencies in font/styling/text
- Managed the website data entries on Firebase

Sami Hassan's Contributions:

- Created Report class and methods for hardcoded reports required for the deliverable.
- Minor UI touch ups
- Designed UML class diagram.
- Updated README file on Github.

Deliverable 3:

Nathan Le's Contributions:

- Created Meal Class
- Created MealList adapter
- Listeners for each meal entry on the cook profile which displays information of each meal to the client
- Created the display that shows the user all offered meals from the cook selected
- Display meal information when cook clicks on their meal
- Cook can toggle if their meal is offered or not and UI displays accordingly
- Cook can delete meal from their menu only when it is not currently offered
- Cook can add meal to their menu through database
- Validation of all information while adding meal

Aaditya Shah's Contributions:

- Enabled cooks to be able to offer meals by changing the meal class
- Amended Database design/hierarchy to accommodate cooks offering or not offering meals
- Allow cooks to remove meals from the database
- Added pictures for uploaded meals
- Validated fields for multiple inputs as well as changed the layouts for certain pages

Bo Lin Chen's Contributions:

- Made the necessary UI pages/dialogs that were needed. (client home page, cook menu(offered meal list), dialog for removing a meal/ toggling offered or not)
- Log out button for cook

Kevin Dang's Contributions:

- Corrected button IDs on admin complaints page
- Created and updated suspend button on admin page
- Created and implemented the ability for cook to offer meals to clients
- Connected offering meals implementation to UI

Siva Senthilkumaran's Contributions:

- Managed the Java Unit Tests for the classes in the app
- Tested the UI Manually, looking for inconsistencies in font/styling/text
- Managed the website data entries on Firebase
- Fixed an error with Firebase, and updated the rules of the database for the application

Sami Hassan's Contributions:

- Created the submit report screen and its activity file, fields inputted create an instance of the report class and sent to admin.
- Ensured fields and buttons were valid, sending toasts to users for invalid fields.
- Developed 1 out of 4 of the deliverable's unit tests.
- Created UML class diagram.
- Created the Meal class and methods.
- Updated README file on Github.

Deliverable 4:

Nathan Le's Contributions:

- Created Request class
- Requests will funnel into pending request list and accepted request list
- If a request is pending, cooks can accept or reject it. If a request is accepted they can complete or still reject it.
- Handling of requests in database
- Back button on searching for meals as a client
- Error checking when adding a quantity while ordering meals
- UI displays the status of orders for clients to view. It displays the time the order was made, the cook from which they ordered, and the status (pending, accepted, or complete).

Aaditya Shah's Contributions:

- Amended Database design/hierarchy to allow Clients to submit orders and to allow cooks to access orders
- Enabled cooks to accept or reject meals based on their discretion as well as enabling them to complete orders
- Amended Meals class to allow a rating to be saved per meal
- Developed rating system where clients can rate the meal after the cook completes the order
- Implemented SearchView for users to find meals from any cook as long as they aren't suspended

- Added Padding to all UI elements
- Created new theme and color palette for dark and light layouts
- Implemented android material 3 library to re-design default elements
- Followed android Material 3 guidelines in simplifying layouts and creating a more cohesive UI

Bo Lin Chen's Contributions:

- Made the necessary UI pages that were needed. (client home page, cook profile, cook profile client side, search, initial design for submitting report)
- Helped rebrand the UI design (changing the colors scheme, and text)

Kevin Dang's Contributions

- Redesigned the UI to follow the updated color scheme (textcolor, button color, background colors)
- Made the pop-up dialog for rating cooks
- Implemented the rate cook function and connected it to the UI
- Validated inputs for rate cook function
- Developed the notification builder which allowed clients to see if their order was approved or denied
- Wrote the entire final report except for other member's contributions

Siva Senthilkumaran's Contributions:


- Developed the original UI for the Submit Rating page
- Managed the Java Unit Tests for the classes in the app
- Tested the UI Manually, looking for inconsistencies in font/styling/text
- Managed the website data entries on Firebase

Sami Hassan's Contributions:

- UML class diagram.
- Updated README file on Github
- Revamping the UI, altering many screens.
- Updated the cook's profile screen on the client side

Screenshots:

5:09 [status icons] 77%



Email

Password

Login

Sign up

5:10 [status icons] 76%

Registration

First Name

Last Name

Password

Email

Phone: -

Address, Click again for AutoComplete

Postal Code:

Client

Card Number

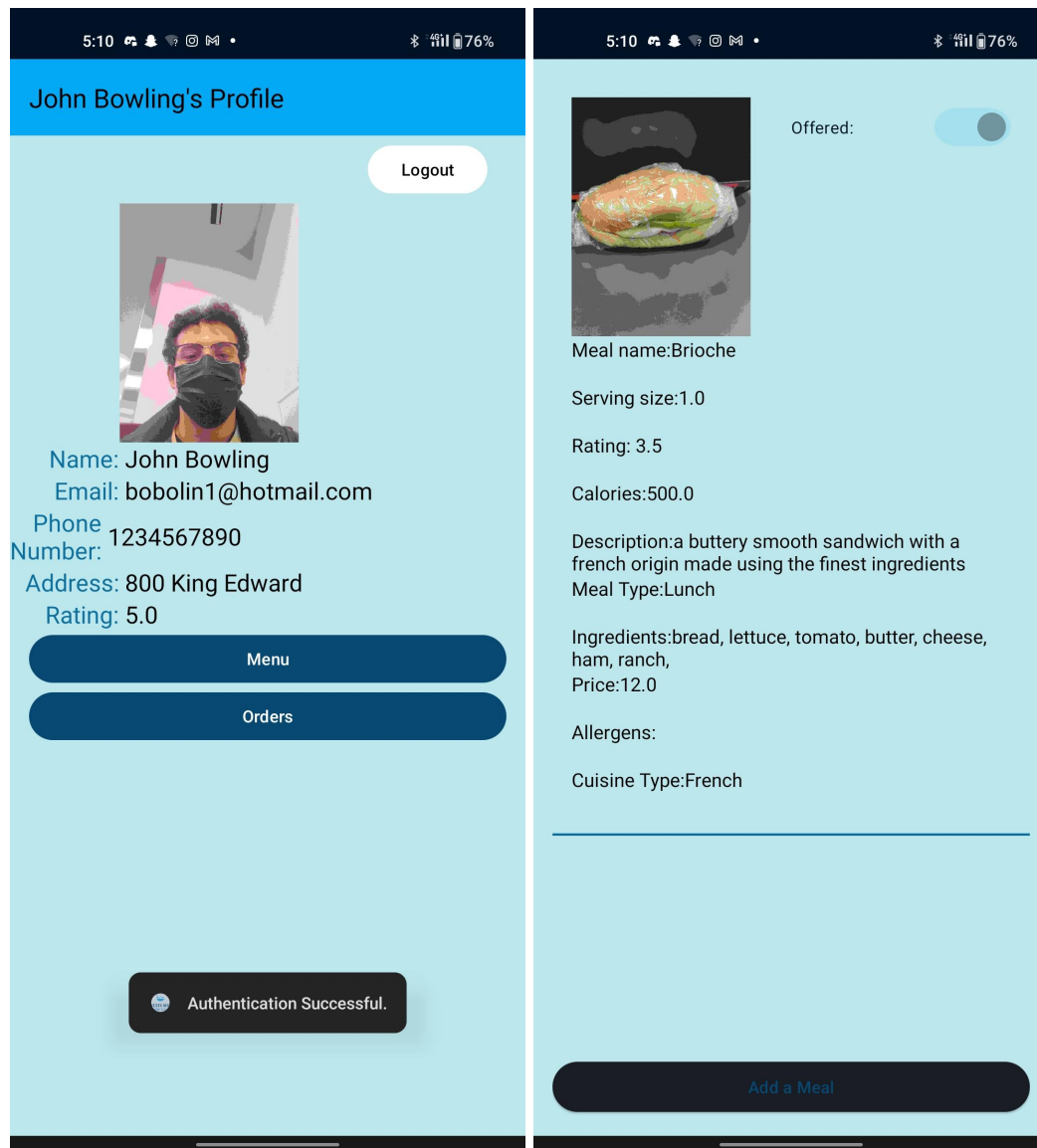
CCV

Enter Card Expiry:


Month Year


UPLOAD PROFILE PICTURE...


Register!




5:11









76%

Meal Name

Meal Type

Price

Cuisine

Serving Size

Calories

Enter Description...

Enter Ingredients...

ADD INGREDIENTS






Enter Allergens...

ADD ALLERGENS

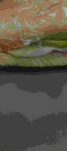
Upload Meal Photo...

ADD MEAL!

5:11

76%



Offered: ☒

Meal name:Brioche

Serving size:1.0

Rating: 3.5

Calories:500.0

Information about Brioche

Toggle Offer

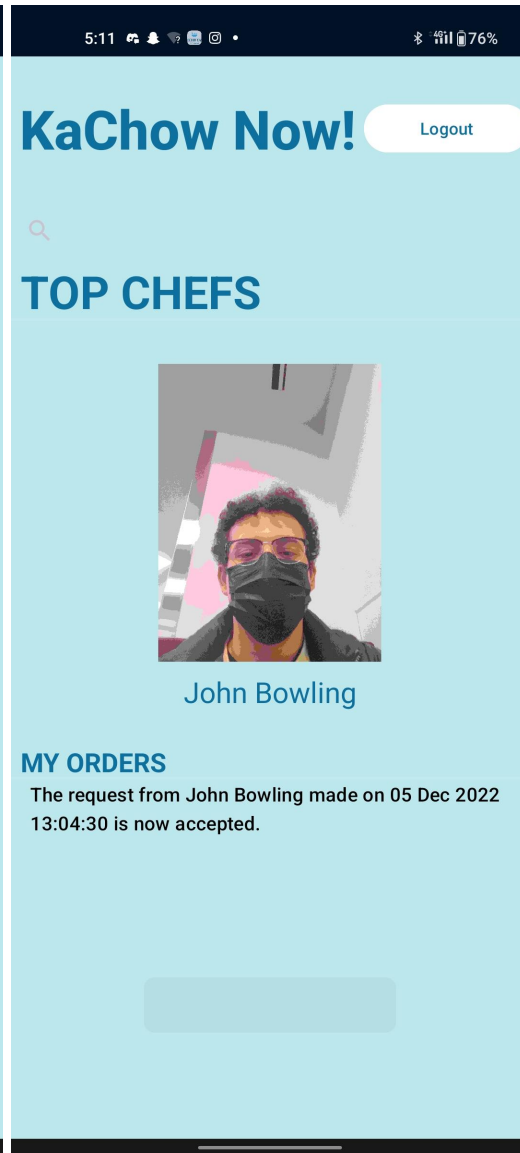
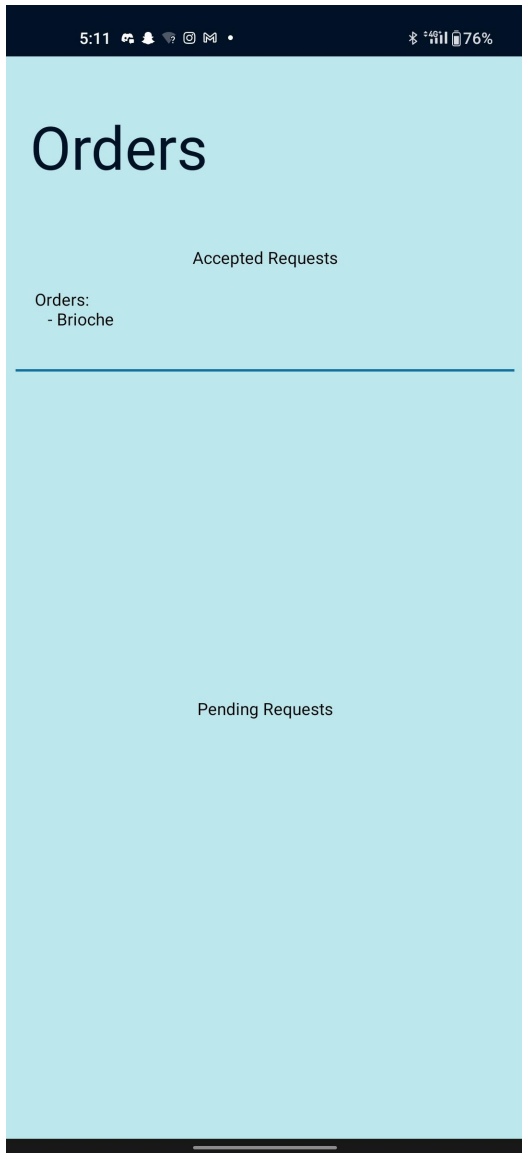
Delete Meal

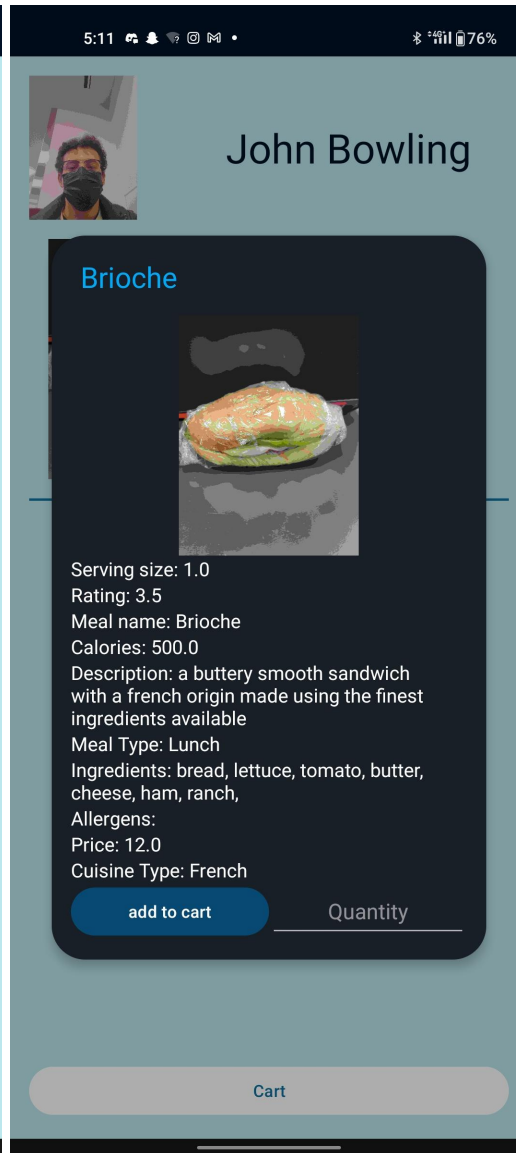
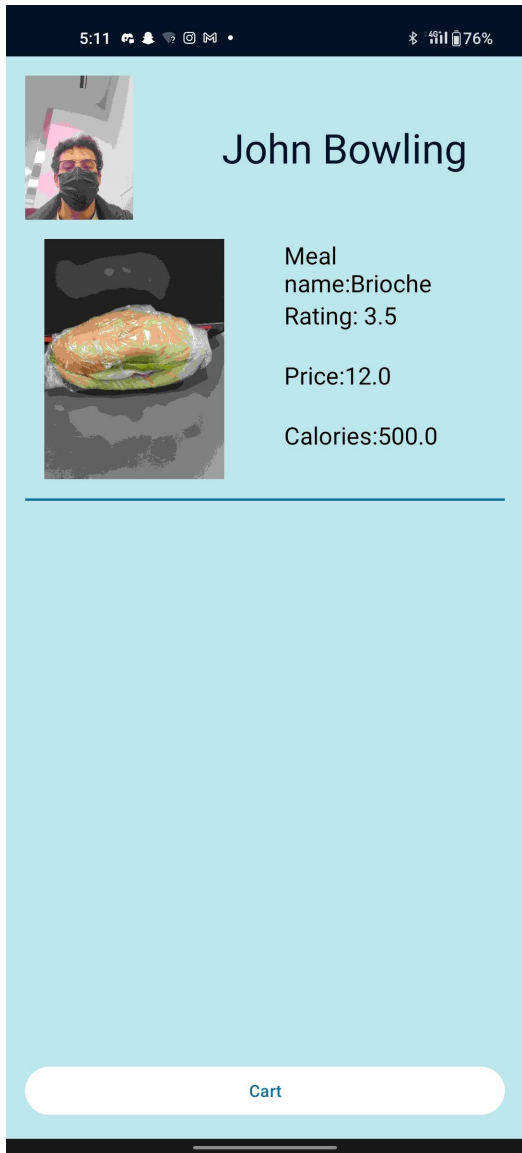
Price:12.0

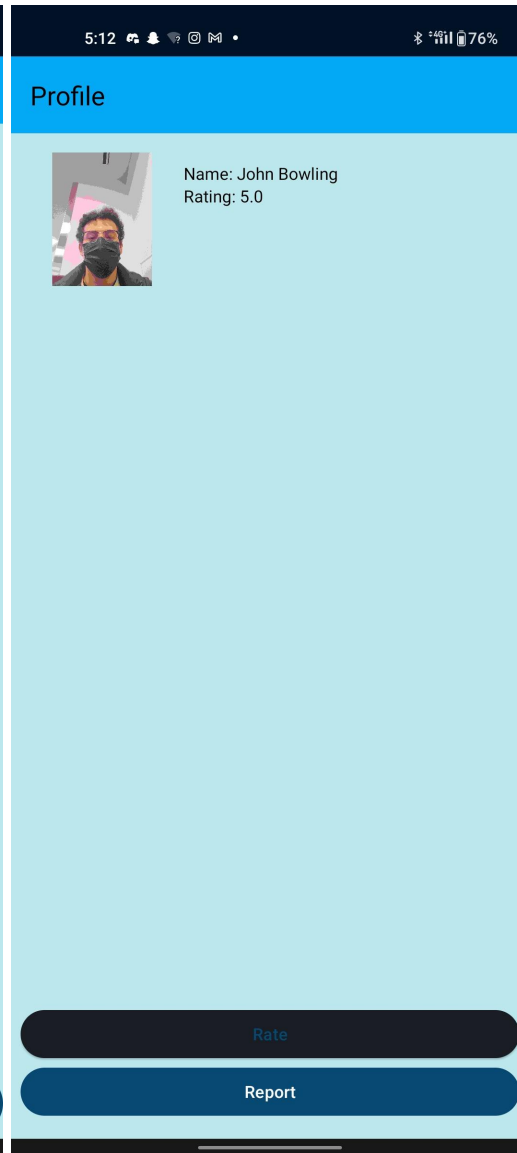
Allergens:

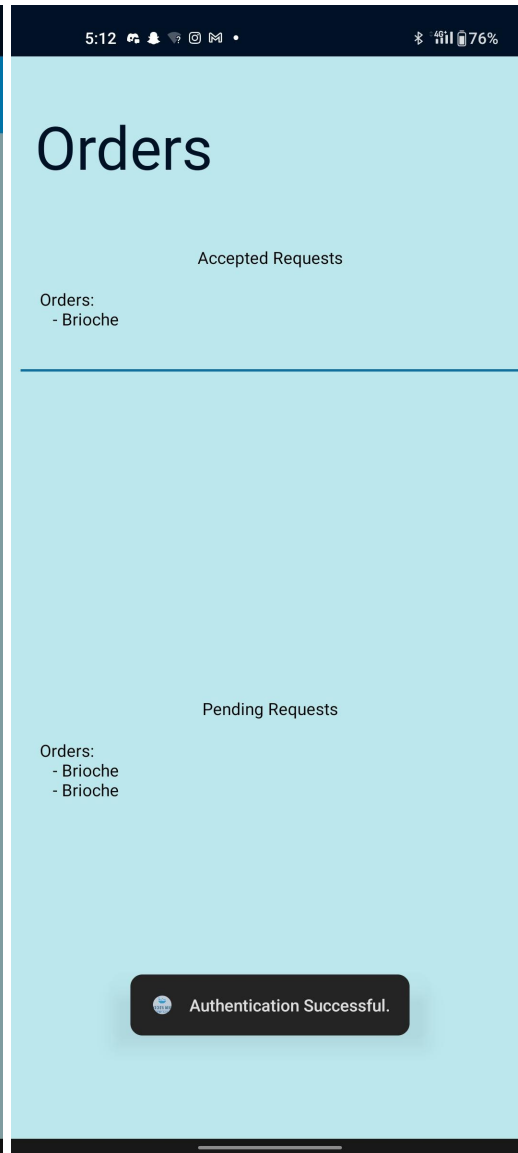
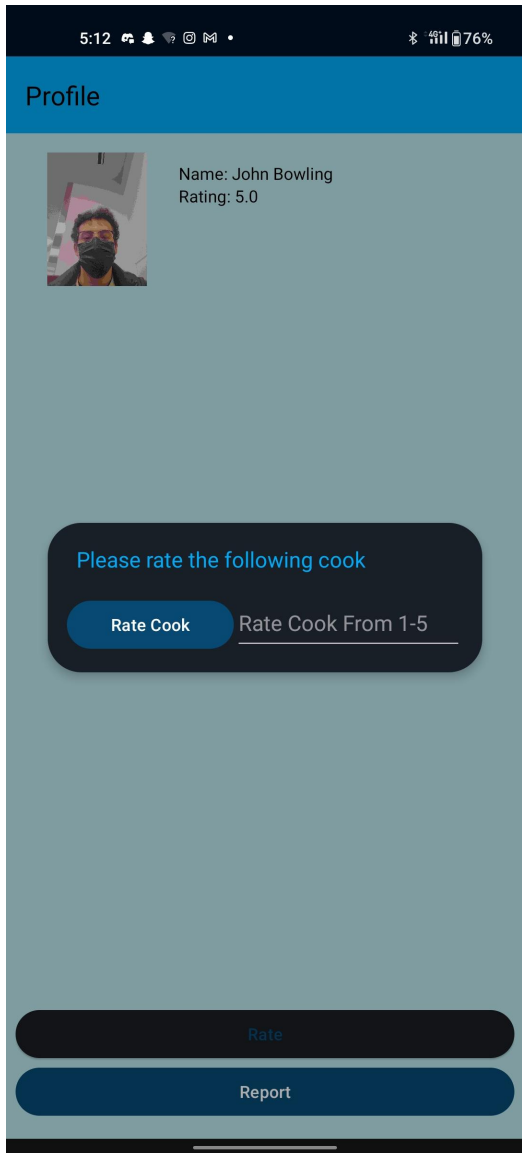
Cuisine Type:French

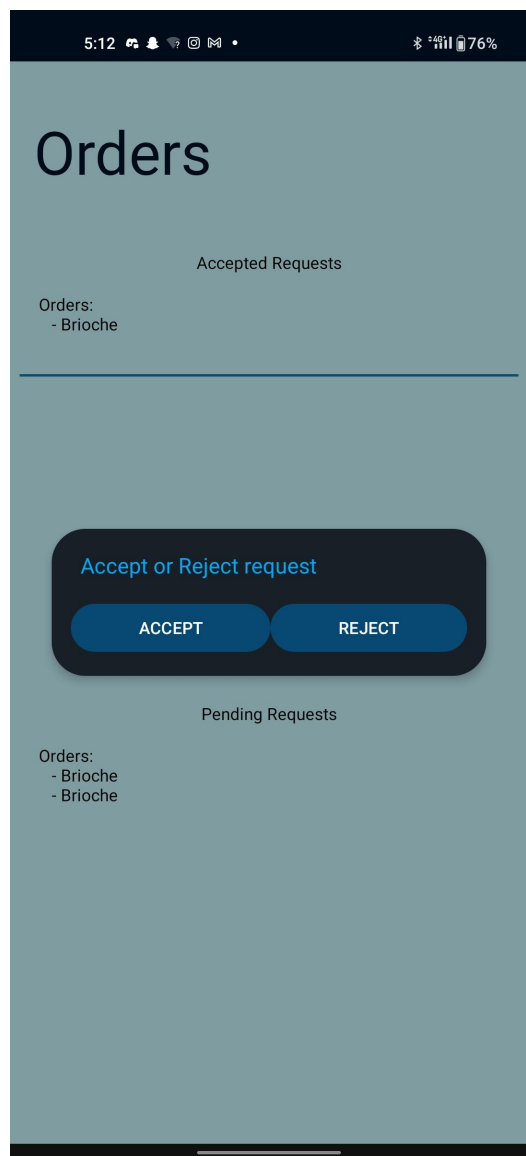
Add a Meal















5:15



   75%

Submit Report

Meal Ordered:

Date:

Day


Month




Year

Describe the situation.

SUBMIT REPORT

5:16



   75%

LOGOUT


INBOX

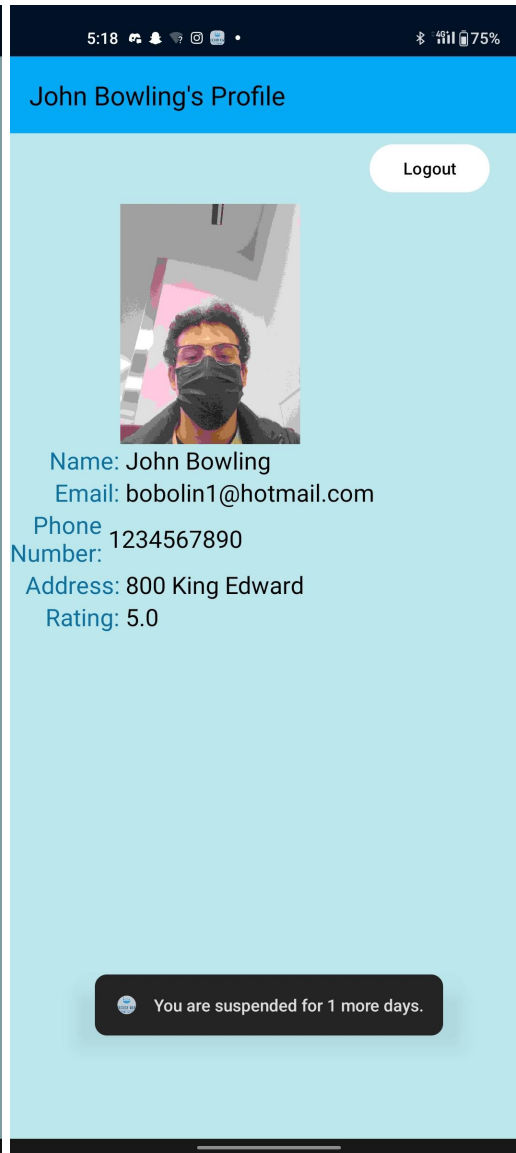
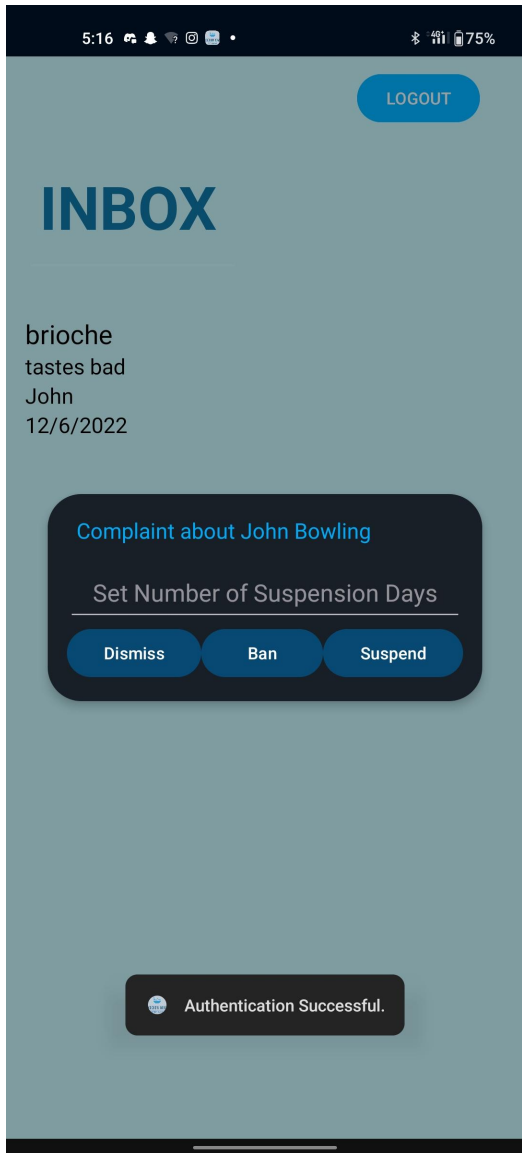
brioche

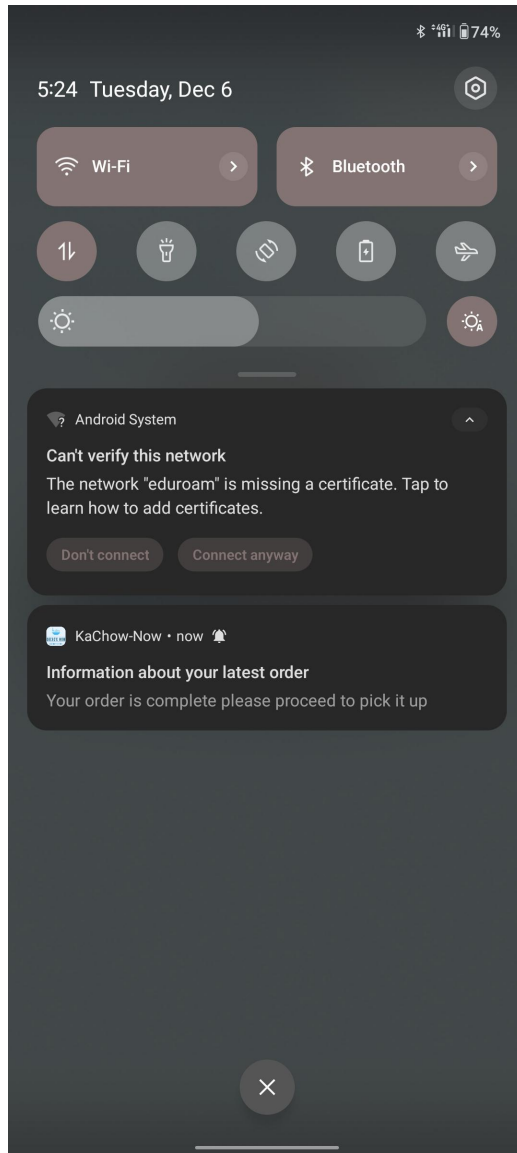
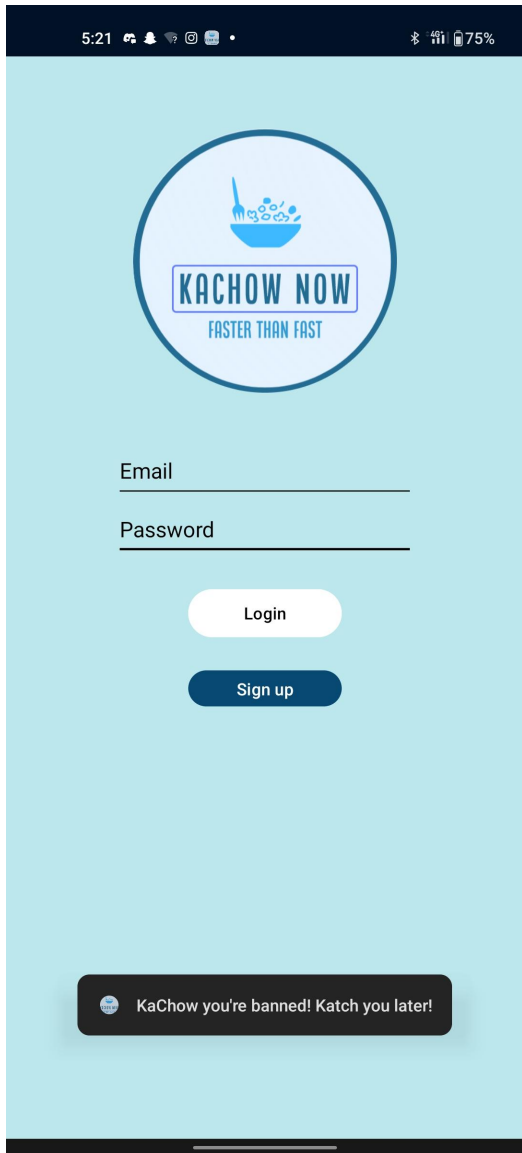
tastes bad

John

12/6/2022

 Authentication Successful.





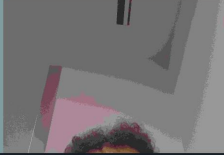
5:30

73%

KaChow Now!

[Logout](#)

TOP CHEFS



Rate your last meal

Rate Meal

Enter a rating from 1 to 5

John Bowling



Authentication Successful.

Lessons Learned:

Throughout the development of our app, we ran into many problems, but we eventually overcame them. This allowed us to learn many valuable lessons, such as the importance of time management, effective communication, and the importance of organization.

One of the biggest lessons we learned during the development of our application was how to manage our time. Since all of us are very busy, it was extremely hard to find the time to work on the project. For example, during deliverable 1, everyone in our group had either midterms or assignments coming up so we could not find time to work on the project which meant we ended up working on the project right up to the deadline. However, our time management got even worse during deliverable 2 when all of our group members had the same midterms on the same day, so it was very difficult to coordinate a time to meet up and work on the project when everyone wanted to study for their exams. Our immense amount of work in other classes caused us to work on the deliverable until the very last minute, but luckily we managed to finish just on time. After that close call, we decided to plan our meetings properly so that the same problem with deliverable 2 will not repeat itself. This led to a drastic improvement in our time management as our group managed to finish deliverable 3 a few days before the deadline. To further improve our time management, we decided to use the SEG tutorial and lab sessions that were now over as more time slots to furthermore work on our project. Combining the time we managed to find through weekdays and weekends, we once again finish deliverable 4 a few days ahead of the deadline. Throughout all the deliverables, we found that we excelled at assigning tasks to one another based on everyone's strengths and weaknesses, this led to everyone having something to work on.

Another lesson we learned during the development of our application was how to effectively communicate with each other. As mentioned above, we were great at assigning tasks to each other, but sometimes tasks get lost in communication which resulted in multiple group members doing the same thing twice. Luckily, this only happened a few times, but it was still frustrating as one of the two members could have been working on something else instead. When we tried to solve this issue, it led to us unintentionally removing both versions of the duplicated work, so we had to go back and redo the same work once again. We noticed that this usually appears when some of our group members did stuff on their own time, which in itself is a good thing, but then someone else who did not know what they did would unintentionally do the same work again. However, we eventually began communicating more effectively as we would tell one another at the beginning of our meetings if we made any changes and if we did, then we would describe what changes were made so everyone would be informed.

The last lesson we learned throughout this project was how valuable organization truly is. During our meetings for deliverable 1, we did not set any goals, so it was difficult to tell whether or not we were making steady progress. This issue led us to begin working on too many things at once without actually finishing them and so, oftentimes the amount of work felt extremely overwhelming. We never planned anything out, our goal was to just jump right into coding and figure out everything along the way. Unfortunately, we did not improve on our organization strategy until deliverable 3 when we decided to begin setting goals for each time we met up so there would be a proper objective. Another thing we decided to do was use a checklist system. Instead of checking the application requirements before and after the deliverable, we would check them off along the way so it was easier to determine what our next goals were. This exceptionally improved our organization because we finally had a plan on how to approach future deliverables. Also, the satisfaction of accomplishing our goals gave us the motivation to continue working on the project despite the amount of other work we had in other classes.

In conclusion, all of us learned many valuable lessons throughout this project such as improving our time management, using effective communication, and the importance of organization. All the lessons we learned will surely help us in our future careers where we will inevitably be working with others.