

Final Project Report

Android Mealer App

2022 Fall SEG2105[B]

Course Coordinator: Wassim El Almar

Members:

Patrick Meyer - 300220498

Allen Mei - 300238743

Ashvin Ramanathan - 300242541

Oscar Li - 300248450

Table of Contents

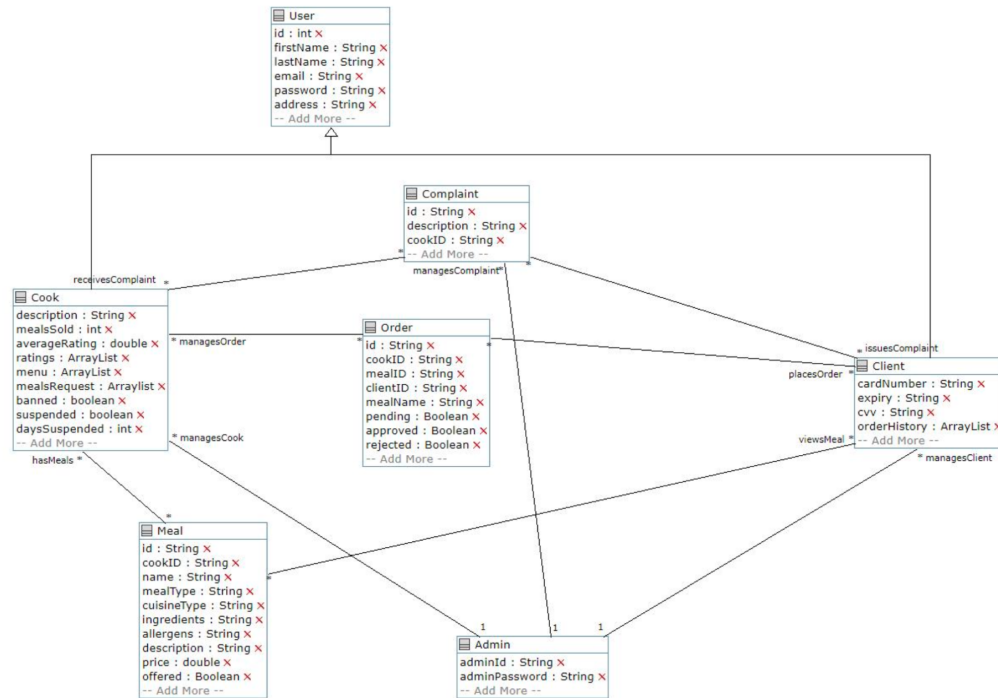
Table of Contents	2
Introduction	2
Final UML Diagram	3
Contributions	3
Screenshots	4
Lessons learned	6

Introduction

Mealer is a simple Android application that allows the purchase of meals by clients from cooks. Mealer has an admin who manages a list of complaints filed by clients and they can administer suspensions to cooks they deem problematic. The application was programmed entirely in Java and user information was stored into a Firebase Database.

This report aims to provide an overview of the functions of the Mealer app and contributions of group members. Included is an updated UML diagram, each group member's contribution, screenshots, and lessons learned.

Final UML Diagram

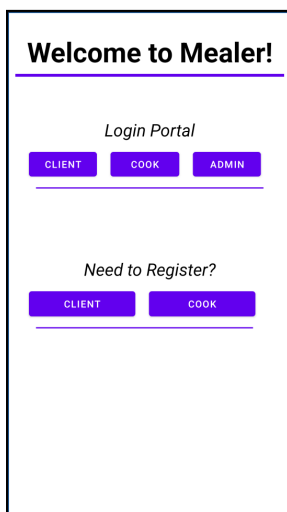


Contributions

Member	Dev. 1	Dev. 2	Dev. 3	Dev. 4
Patrick	Firebase + Backend coding	Backend	Backend	Backend
Allen	Firebase + Backend coding	Backend coding	Backend coding	Backend coding
Ashvin	Separated Cook+Clients w/ LogOuts	JUnit (tried android testing)	UI + JUnit Testing	UI + JUnit Testing
Oscar	UML Diagram + Deliverable code outline (functions and classes needed)	UML Diagram + Deliverable code outline (functions and classes needed)	UML Diagram + Deliverable code outline (functions and classes needed)	Report + UML Diagram + Deliverable code outline (functions and classes needed)

Further elaborating on what everyone worked on. Patrick and Allen strictly worked on the backend side of the application. From setting up the Firebase and getting it to work, to implementing required functionalities. On the other hand Ashvin worked primarily on the unit testing and on the user interface of the application. Great applications require great UI's! Lastly, Oscar was in charge of designing the UML diagrams and making outlines of the functions required to fulfill the given requirements. Without his help we would not have been able to code as effectively.

Screenshots



Welcome to Mealer!

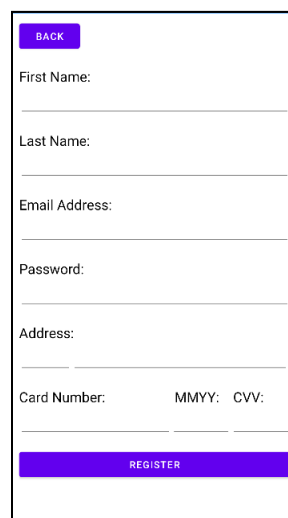
Login Portal

CLIENT COOK ADMIN

Need to Register?

CLIENT COOK

Main menu



BACK

First Name:

Last Name:

Email Address:

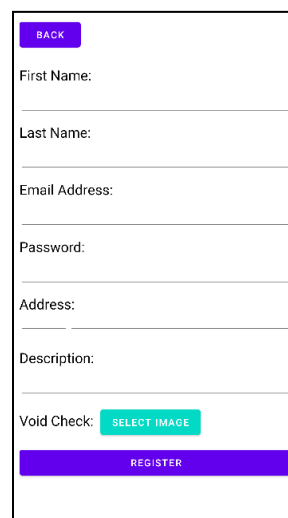
Password:

Address:

Card Number: MMY: CVV:

REGISTER

Client registration



BACK

First Name:

Last Name:

Email Address:

Password:

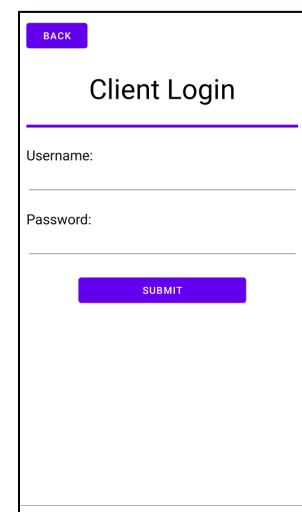
Address:

Description:

Void Check: SELECT IMAGE

REGISTER

Cook registration



BACK

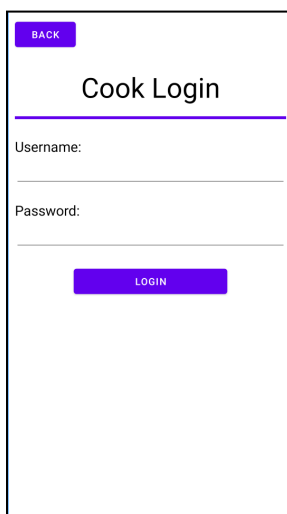
Client Login

Username:

Password:

SUBMIT

Client login



BACK

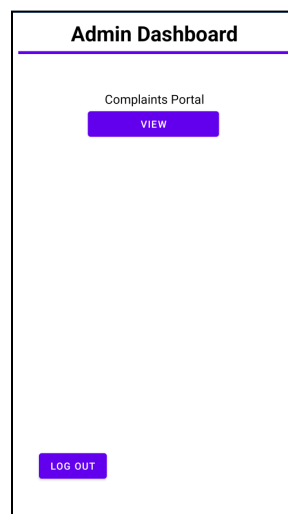
Cook Login

Username:

Password:

LOGIN

Cook login



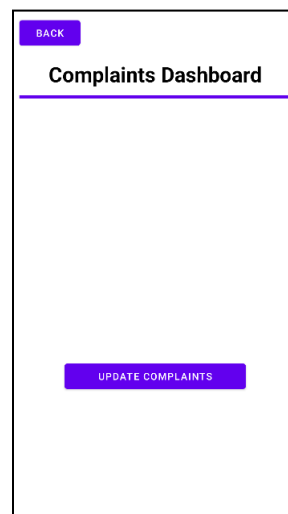
Admin Dashboard

Complaints Portal

VIEW

LOG OUT

Admin dashboard
(homepage)

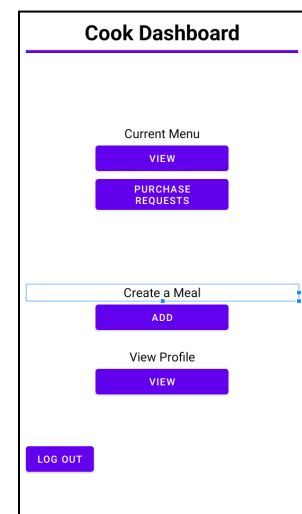


BACK

Complaints Dashboard

UPDATE COMPLAINTS

Complaints
dashboard



Cook Dashboard

Current Menu

VIEW

PURCHASE REQUESTS

Create a Meal

ADD

View Profile

VIEW

LOG OUT

Cook dashboard
(homepage)

BACK

MENU

Item 1

Sub Item 1

Item 2

Sub Item 2

Item 3

Sub Item 3

Item 4

Sub Item 4

Item 5

Sub Item 5

REFRESH

OFFERED MEALS

Cook menu page

Offered Meals

Item 1

Sub Item 1

Item 2

Sub Item 2

Item 3

Sub Item 3

Item 4

Sub Item 4

Item 5

Sub Item 5

REFRESH PAGE

BACK

Cook offered meals page

Meal Name:

Meal Type:

Cuisine Type:

Ingredients:

Allergens:

Price:

Description:

BACK

ADD MEAL

Cook create a meal

Your profile

Your information:

Description:

Number of meals sold:

Rating:

Email:

Name:

Address:

BACK

Cook profile

Address:

Email:

Cook description:

Cook rating (x/5):

Meal information:

Meal name:

Description:

Meal type:

Cuisine type:

Price:

ORDER

DISMISS

Meal information pop-up

ADD TO OFFERED MEALS

REMOVE FROM MENU

DISMISS

Add/fremove meal pop-up

Rate meal (x/5):

RATE

Issue complaint:

COMPLAIN

Rate/complain about Cook pop-up

enter suspension length (if blank set to 1)

SUSPEND

BAN

DISMISS

Admin complaint pop-up

Client Dashboard

SEARCH MEALS

STATUS OF PURCHASES

LOG OUT

Client dashboard
(homepage)

BACK

UPDATE

Cook/Client requests
page

Search:

Filter:
☐ Meal Name ☐ Meal Type ☐ Cuisine

SEARCH

BACK

Client meal search

Client Dashboard

SEARCH MEALS

STATUS OF PURCHASES

LOG OUT

Client dashboard
(homepage)

Cook Dashboard

LOG OUT

Banned Cook
dashboard
(homepage)

Lessons learned

Throughout the process of creating Mealer we encountered multiple problems. The following is a list of some issues that were fixed and the lessons learned from them:

- Most members on our team had never worked with a real-time database so hearing that we were gonna need to implement one for our project initially scared us. The initial setup of the Firebase database was, to be honest, pretty challenging and required a lot of searching on the web. Over the course of the term we got increasingly comfortable with Firebase and discovered its true potential. We now know the importance of databases and how they can effectively store data and make our lives easier.
- Code DRY. The implementation of every deliverable required us to code similar functionalities for different classes. Coding over and over the same functions/pieces of code would take longer and reduce the maintainability of our application. Reusing pieces of code helped us maintain and limit issues caused by rewriting them.
- Plan ahead. For the first deliverable we waited until the last possible day to submit our work. Thankfully, we were able to implement everything required and it functioned, but it would have been equally possible that it would not have functioned and we would have been stuck on a small issue that we could have fixed with an extra day. For the next three deliverables we made sure to work and finish the deliverables at least three days before they were due. This ensured a lengthy amount of time to fix problems.
- Use Github. Many members on our team hadn't used Github to manage files/projects. Since we were forced to use Github for our project, we learned very rapidly the impressive functionalities provided. With Github we can very easily receive and upload files that others can then pull. This made it extremely easy to collaborate on the project remotely.