

# SEG 2105: Mealer Project

Group 18:

Maria Stancu - 300243486

Bernadette Tona - 300151961

Trinity Bates - 300129927

Anoushka Jawale - 300233148

## TABLE OF CONTENTS

GitHub repository .....	1
Introduction .....	1
UML Diagram .....	2
Tasks distribution .....	2
Screenshots .....	4
Main Page .....	4
Administrator .....	4
Client .....	5
Cook .....	13
Conclusion and Lesson Learned .....	17

## GitHub repository

[https://github.com/mstan091/SEG2105\\_Project.git](https://github.com/mstan091/SEG2105_Project.git)

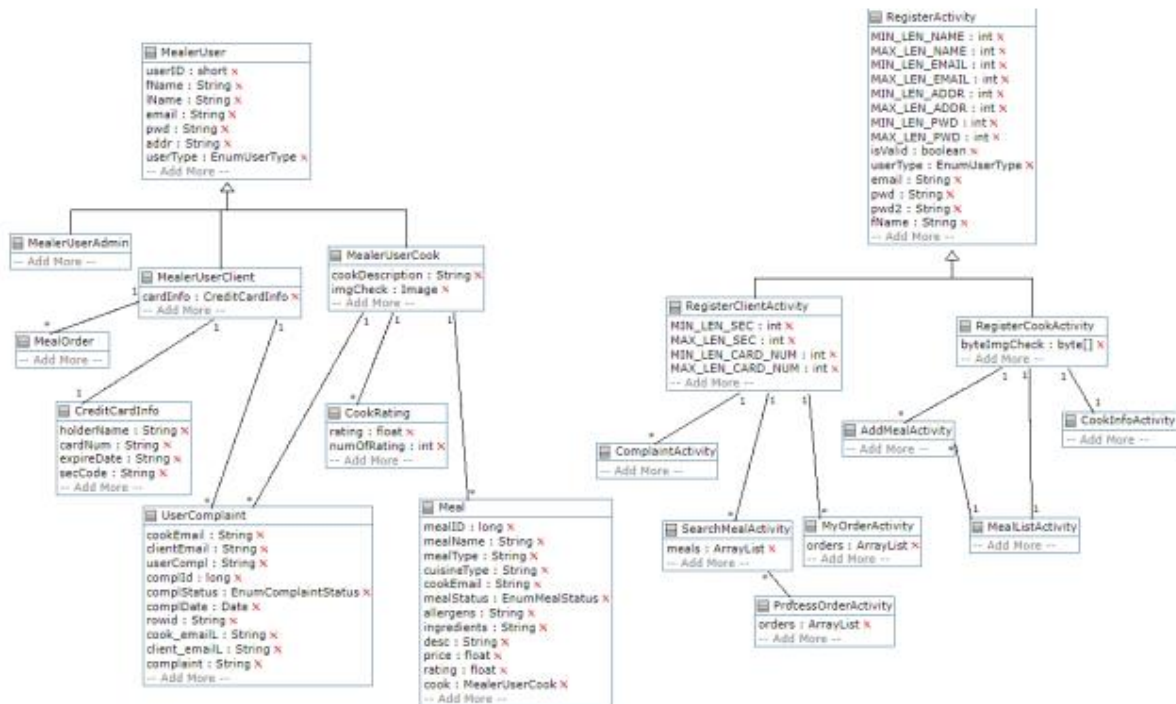
## Introduction

Group 18 was tasked with developing an android application that provides meals at home: “Mealer”, an Ottawa-based meal sharing application where local cooks can sell meals to clients from their home This app was built for three types of users:

- Cook: a user that makes meals at home and sells them to Clients.
- Client: a user that buys meals from Cooks. They order the meal through the application and pick it up from the Cook’s home.
- Administrator: a user that receives complaints about a Cooks from a Client and may suspend the Cook if necessary. The administrator is pre-registered.

This app could allow users to login or create an account as a cook or a client.

## UML Diagram



## Tasks distribution

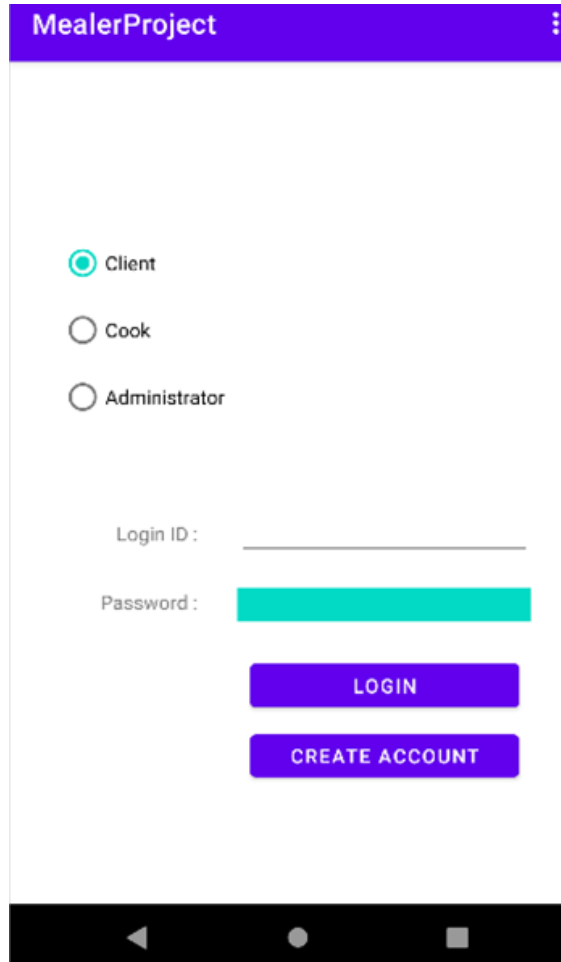
	Deliverable 1	Deliverable 2	Deliverable 3
<b>Maria</b>	<ul style="list-style-type: none"> <li>- Allow a user to create an account as a cook</li> </ul>	<ul style="list-style-type: none"> <li>- The Administrator can action the complaint (dismiss complaint or suspend cook)</li> </ul>	<ul style="list-style-type: none"> <li>- The Cook can add a meal to the menu</li> <li>- The Cook can add a meal to the offered meals list</li> </ul>

<b>Anoushka</b>	<ul style="list-style-type: none"> <li>- Allow a user to create an account as a client</li> </ul>	<ul style="list-style-type: none"> <li>- The complaints list is stored in the database</li> <li>- 2-unit test</li> </ul>	<ul style="list-style-type: none"> <li>- The Cook cannot delete a meal from the menu if it is currently in the offered meals list</li> <li>- UML</li> </ul>
<b>Trinity</b>	<ul style="list-style-type: none"> <li>- The Administrator, Cook, or Client user can see the “welcome screen” after successful authentication. The welcome screen specifies the user role.</li> </ul>	<ul style="list-style-type: none"> <li>- The Cook can see a message informing them that they have been suspended</li> <li>- UML</li> </ul>	<ul style="list-style-type: none"> <li>- The Cook can add a meal to the offered meals list.</li> <li>- 4-unit test case</li> </ul>
<b>Bernadette</b>	<ul style="list-style-type: none"> <li>- The administrator registers</li> <li>- The user can logout</li> <li>- UML</li> </ul>	<ul style="list-style-type: none"> <li>- When a user registers, their account information is stored in the database</li> <li>- 2-unit test case</li> </ul>	<ul style="list-style-type: none"> <li>- When a suspended Cook logs on, they can only see the suspension message and log off. They cannot perform any other action.</li> </ul>

	<b>Deliverable 4</b>
<b>Maria</b>	<ul style="list-style-type: none"> <li>- UML</li> <li>- The Client can submit a purchase request The Cook can receive the purchase request submitted by the Client</li> <li>- The Client can view the status of their purchase (pending, approved, or rejected)</li> <li>- The Client can rate the Cook from which they have purchased a meal 5 The Client can submit a complaint about a Cook to the administrator</li> <li>- The Cook can view, and approve/reject purchase requests received from Clients.</li> </ul>
<b>Anoushka</b>	<ul style="list-style-type: none"> <li>- The Cook can view their profile and rating. All fields should be validated.</li> <li>- There should be appropriate error messages for incorrect input.</li> </ul>
<b>Trinity</b>	<ul style="list-style-type: none"> <li>- The Client can search for a meal</li> <li>- The Client can see search results for meals offered by non-suspended Cooks</li> </ul>
<b>Bernadette</b>	<ul style="list-style-type: none"> <li>- The Client can view the Cook’s information and rating for each meal in the search result</li> <li>- The Client can view the meal’s information for each meal</li> </ul>

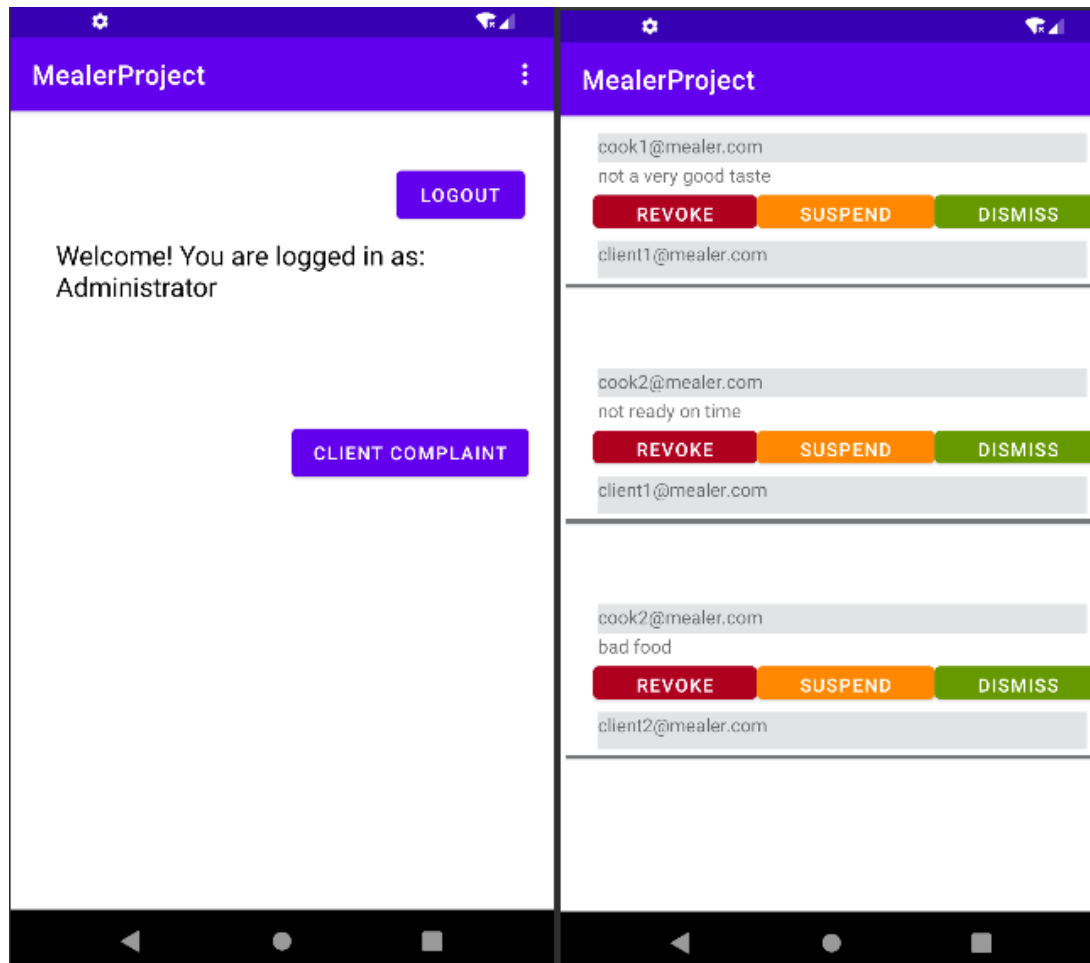
## Screenshots

Main Page



The screenshot shows the main page of the MealerProject application. At the top, there is a purple header bar with the text "MealerProject" and a three-dot menu icon. Below the header, there are three radio buttons for user roles: "Client" (selected), "Cook", and "Administrator". Below the radio buttons, there are two input fields: "Login ID :" and "Password :". The "Password :" field is filled with a red rectangle. Below the input fields, there are two red buttons: "LOGIN" and "CREATE ACCOUNT". At the bottom of the screen, there is a black navigation bar with three icons: a back arrow, a home circle, and a recent apps square.

Administrator



Client

The screenshot displays the MealerProject mobile application interface, split into two panels. The left panel contains a registration form with the following fields: First Name, Last Name, Email (pre-filled with 'your.email@uOttawa.ca'), Password, Retype Password, Address, Credit Card Number, Card Holder Name, Security Code (pre-filled with 'SVC'), and Expire Date (pre-filled with 'MM/YY'). A blue 'REGIST' button is at the bottom right of this panel. The right panel shows a logged-in client dashboard with a blue 'LOGOUT' button at the top right, a welcome message 'Welcome! You are logged in as: Client', a 'Search Food' section, and three input fields: Meal Name, Meal Type, and Cuisine Type. Below these fields are two blue buttons: 'SEARCH MEAL' and 'MY ORDERS'. The app has a blue header bar with the title 'MealerProject' and a mobile status bar at the top.

**MealerProject**

First Name :

Last Name :

Email :

Password :

Retype Password :

Address :

Credit Card Number :

Card Holder Name :

Security Code :

Expire Date :

**REGIST**

**MealerProject**

**LOGOUT**

Welcome! You are logged in as: Client

Search Food

Meal Name:

Meal Type:

Cuisine Type:

**SEARCH MEAL**

**MY ORDERS**

Search Meal:

## MealerProject

Meal1

Main

Indian

\$: 10.0

desc 1

tofu,onion,pepper

allergens

1 cook street mealer city

Pickup Time : Pickup time

★★★★★

Time

Order

Meal2

Apetizer

Meditranean

\$: 5.5

desc 2

olive,tomatoes,pepper

allergens

2 cook street mealer city

Pickup Time : Pickup time

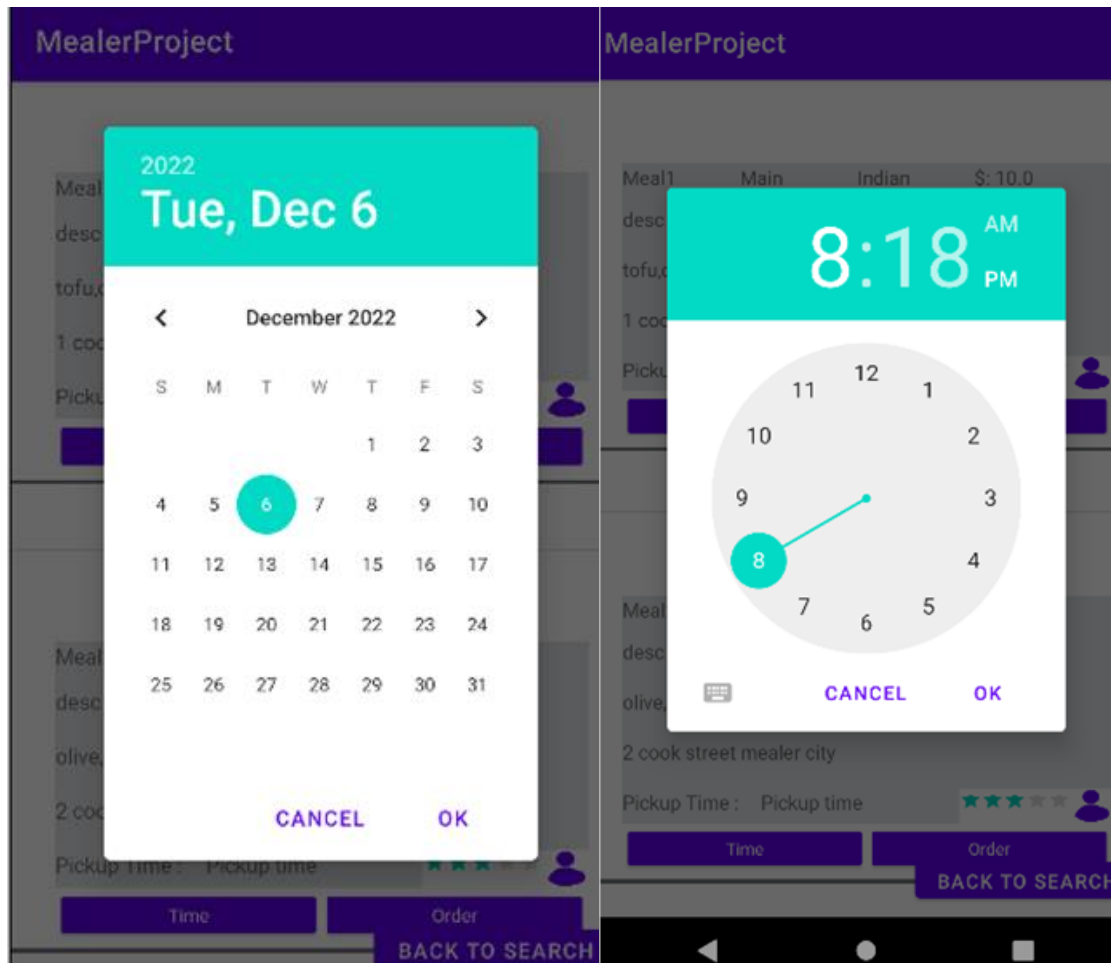
★★★★★

Time

Order

BACK TO SEARCH

Pick date and time and click order:



See the cook profile by clicking the person icon:



## MealerProject

Meal1

Main

Indian

\$: 10.0

desc 1


tofu,onion,pepper

allergens

1 cook street mealer city

Pickup Time : Pickup time

★★★★★



Time

Order

Meal2

Apetizer

Meditranean

\$: 5.5

desc 2


olive,tomatoes,pepper

allergens

2 cook street mealer city

Pickup Time : Pickup time

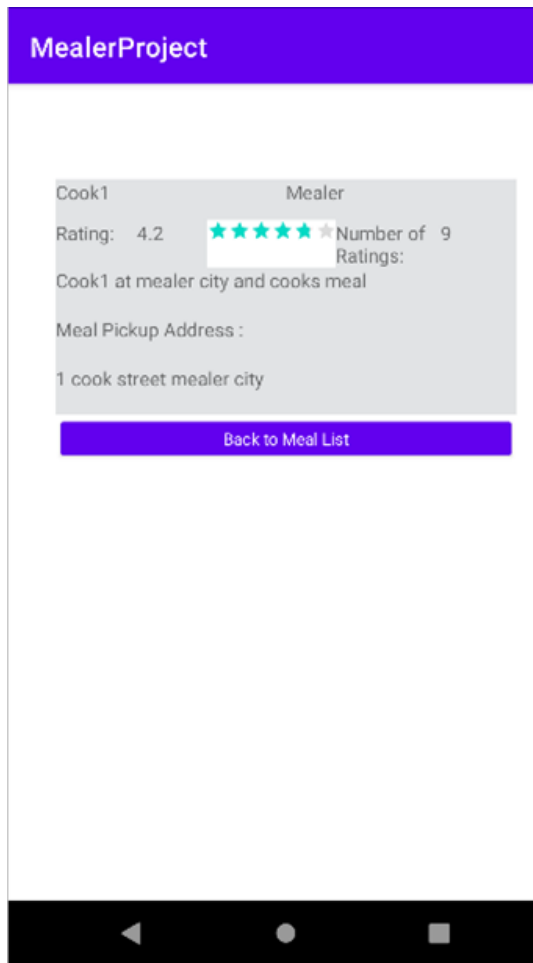
★★★★★



Time

Order

BACK TO SEARCH



View my orders:

**MealerProject**

Meal1	Main	Indian
Pending	cook1@mealer.com	Pickup time

Type complaint...

Submit Complaint

★ ★ ★ ★ ★

Rate cook

Enter a complaint and  
or rate the meal

BACK

If the order status is pending, you cannot rate the cook. It must be approved or rejected by the cook.

When a complaint is entered or you rate cook the status change to completed

Order Status:

## MealerProject

Meal1	Main	Indian
Completed	cook1@mealer.com	Pickup time

complaints

Submit Complaint

★

★

★

★

★

Rate cook

Meal1	Main	Indian
Approved	cook1@mealer.com	2022/11/6 20:14

Type complaint...

Submit Complaint

★

★

★

★

★

Rate cook

BACK

Cook

**MealerProject**

First Name :

Last Name :

Email:

Password :

Retype Password :

Address :

Cook Description :

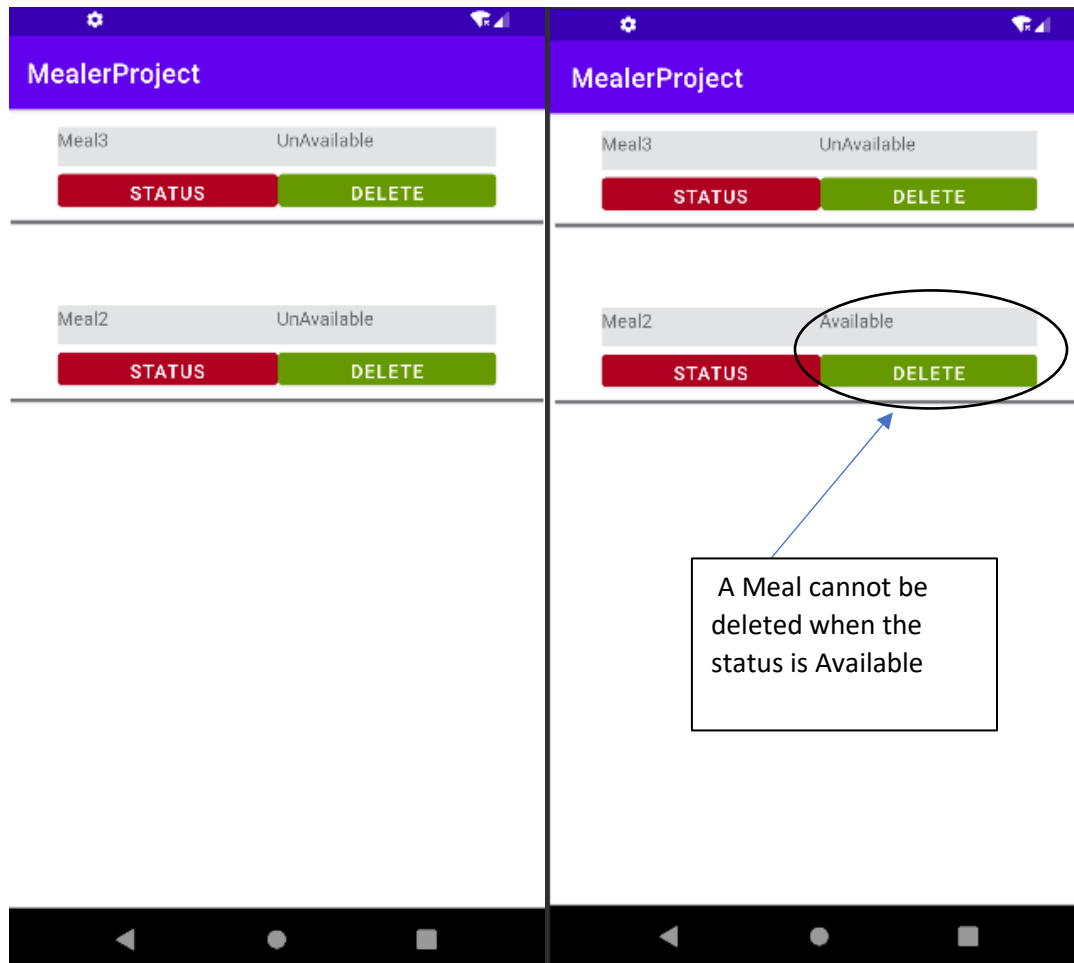
Load a Void Check Here :

**MealerProject**

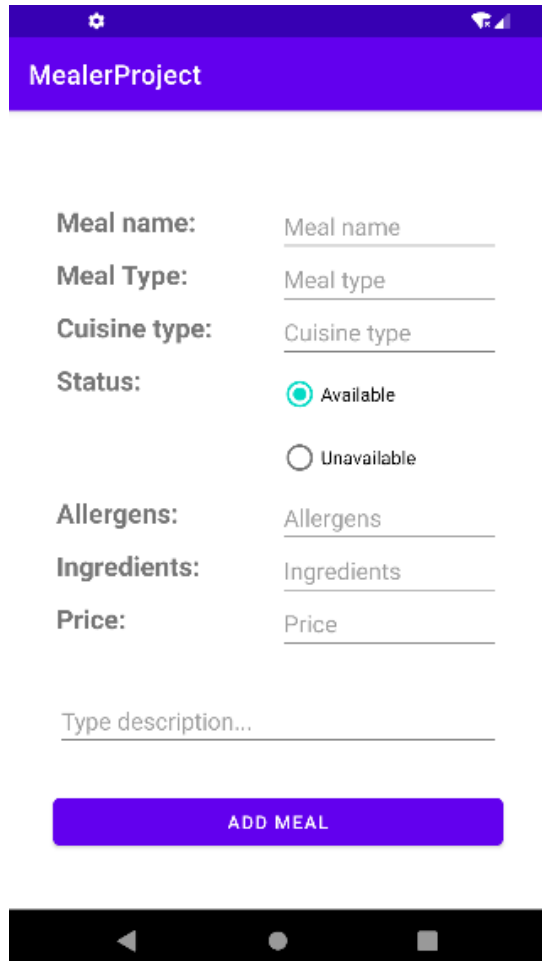
Welcome! You are logged in as: Cook

Your account status is: Active

Edit your menu:



Add meal:



The screenshot shows a mobile application interface for adding a meal. At the top, there is a blue header bar with a settings icon on the left and a Wi-Fi and battery status icon on the right. Below the header, the title "MealerProject" is displayed in white text. The main form area has a light gray background and contains several input fields and a status selection section. The fields are labeled "Meal name:", "Meal Type:", "Cuisine type:", "Allergens:", "Ingredients:", and "Price:", each followed by a text input field. The "Status:" label is followed by two radio button options: "Available" (which is selected) and "Unavailable". Below these fields is a "Type description..." label followed by a text input field. At the bottom of the form is a blue button with the text "ADD MEAL" in white. The very bottom of the screen shows a black Android navigation bar with the back, home, and recents icons.

Meal name:

Meal Type:

Cuisine type:

Status: ☒ Available ☐ Unavailable

Allergens:

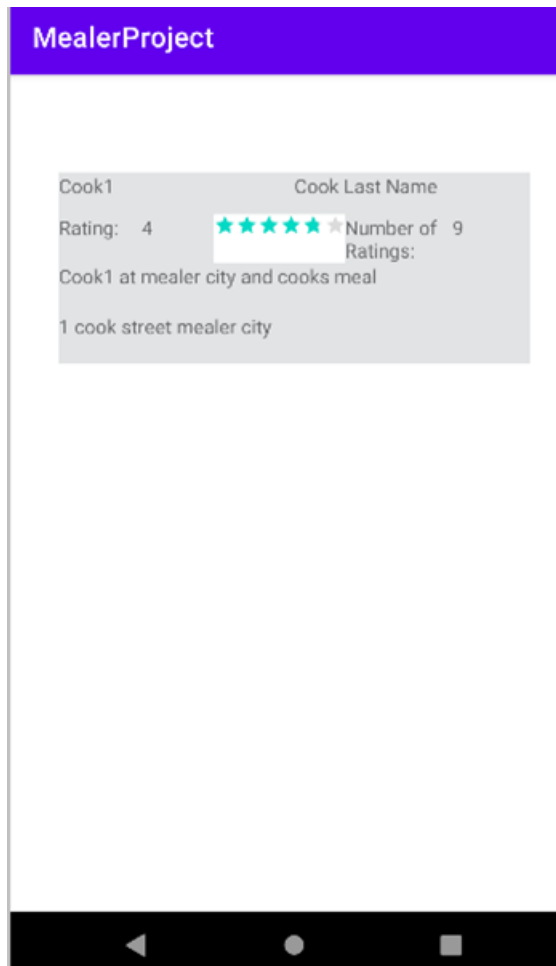
Ingredients:

Price:

Type description...

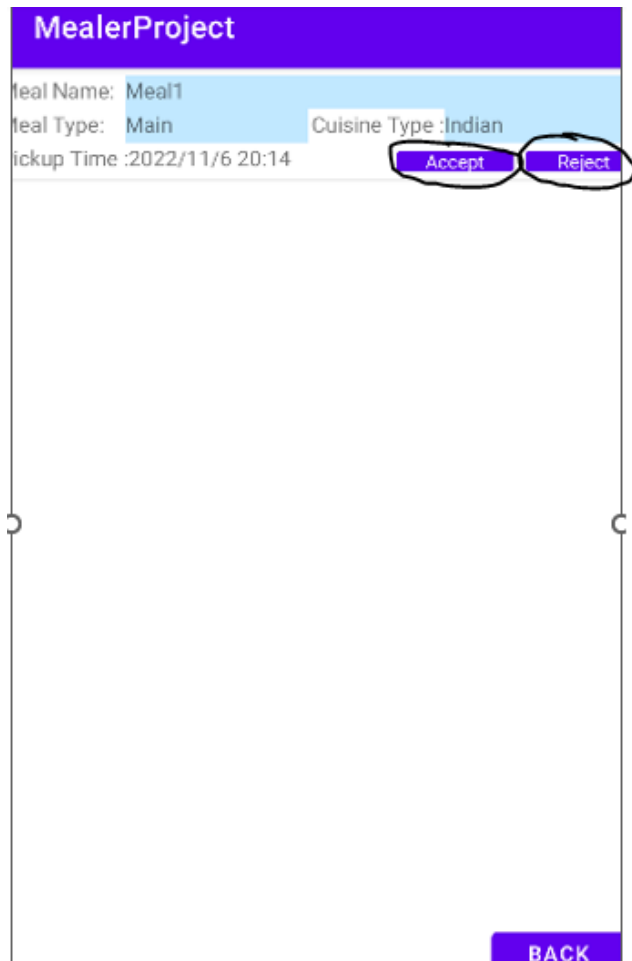
ADD MEAL

View Profile:



View Orders and approved or reject order :





The screenshot displays the 'MealerProject' app interface. At the top, a purple header bar contains the title 'MealerProject'. Below this, a light blue header bar displays meal details: 'Meal Name: Meal1', 'Meal Type: Main', 'Cuisine Type: Indian', and 'Pickup Time: 2022/11/6 20:14'. Below the header, there are two purple buttons labeled 'Accept' and 'Reject', which are circled in black. At the bottom right of the screen, there is a purple button labeled 'BACK'.

## Conclusion and Lesson Learned

Working together on developing this app the members of group 18 improved their organization skill, communication skill, and logic. The team was working very hard to implement all the requirements in time for each deliverable. We found the first derivable to be the most challenging as the team members needed to organize the tasks, learn to work together, design the application, and plan the implementation to meet the due dates. Also, the team needed to acquire the knowledge about how to develop apps in Android Studio as it was something they did not used before. We also needed to learn more about databases and how to integrate SQLite database and Android Studio. We learned several lessons throughout the project:

- How to better separate the tasks
- How to choose naming convention for the .java and .xml files.
- The difference between API 28 and API 33
- Design the database structure at the beginning of the project