A Project Report on

Orderista: An AI Based Food Ordering Application

Submitted in partial fulfillment of the requirements for the award
of the degree of

Bachelor of Engineering

in

Information Technology

by

Jahnavi Naik(17104046)
Ashwin Vishwakarma(17104009)
Tejas Raibagi(17104067)

Under the Guidance of

Mrs. Rujata Chaudhari
Mrs. Geetanjali Kalme

Department of Information Technology
A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W), Mumbai-400615
UNIVERSITY OF MUMBAI

Academic Year 2020-2021

Approval Sheet

This Project Report entitled ***"Orderista"*** Submitted by ***"Jahnavi Naik" (17104046), "Ashwin Vishwakarma"(17104009), "Tejas Raibagi" (17104067)***is approved for the partial fulfillment of the requirenment for the award of the degree of ***Bachelor of Engineering*** in ***Information Technology*** from ***University of Mumbai***.

Mrs. Geetanjali Kalme
Co-Guide

Mrs. Rujata Chaudhari
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Place:A.P.Shah Institute of Technology, Thane
Date:

CERTIFICATE

This is to certify that the project entitled ***"Orderista"*** submitted by ***"Jahnavi Naik" (17104046), "Ashwin Vishwakarma"(17104009), "Tejas Raibagi" (17104067)*** for the partial fulfillment of the requirement for award of a degree ***Bachelor of Engineering*** in ***Information Technology***,to the University of Mumbai,is a bonafide work carried out during academic year 2020-2021.

Mrs. Geetanjali Kalme
Co-Guide

Mrs. Rujata Chaudhari
Guide

Prof. Kiran Deshpande
Head Department of Information Technology

Dr. Uttam D.Kolekar
Principal

External Examiner(s)

1.

2.

Place:A.P.Shah Institute of Technology, Thane
Date:

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Jahnavi Naik (17104046)

Ashwin Vishwakarma (17104009)

Tejas Raibagi (17104067)

Date:

Abstract

In today's world, with people hustling for their jobs all the time, a huge count of people don't have the time to actually prepare their lunch and here canteen plays a major role. But then again people are in a rush and don't have much time to spend in the canteen as well to place an order or wait until their order is ready. This project is based on an AI-based cross-platform application that focuses on automating all the major canteen functionalities. It enables the user to register online, browse and choose from an E-menu card followed by placing the order and receiving confirmation after successful payment. With the help of this app, students and faculties can order food beforehand and can receive it during the break to ensure that the time spent in ordering and collecting the order is as low as possible. The app also docks in AI, which will help the users to get personal recommendations and food items which are popular amongst others. The objective is to reduce the manual paperwork as the app digitalizes every factor and provides a graphical representation of daily sales and allows comparison for weekly, monthly and annual sales. The feedback system allows the admin to monitor the app and make changes based on the user feedback.

Contents

1 Introduction **1**
- 1.1 Objectives . . . 2
- 1.2 Problem Definition . . . 2
- 1.3 Benefits for environment . . . 2
- 1.4 Scope . . . 3

2 Literature Review **4**

3 Proposed System **5**

4 Design **8**
- 4.1 Use Case Diagram . . . 8
- 4.2 Activity Diagram . . . 9
- 4.3 Class Diagram . . . 10

5 Implementation **11**

6 Testing **16**

7 Result **18**
- 7.0.1 Customer Portal . . . 19
- 7.0.2 Canteen Owner/ Manager Portal . . . 25
- 7.0.3 Admin Portal . . . 27

8 Conclusions and Future Scope **31**

Bibliography **32**

Publication **34**

List of Figures

3.1 Admin Portal 6
3.2 Canteen Manager Portal 6
3.3 Customer Portal 7

4.1 Use Case Diagram 8
4.2 Activity Diagram 9
4.3 Class Diagram 10

5.1 API Call Functions 11
5.2 Fetching Menu from Back-end 12
5.3 Building Homepage Items 13
5.4 Initializing Razorpay API 13
5.5 Razorpay Checkout 14
5.6 AI Implementation using KNN 15

7.1 Application Landing Page 19
7.2 Login Page 20
7.3 Home Page 21
7.4 Favourites Page 22
7.5 Cart Page 23
7.6 Profile Page 24
7.7 Canteen Owner/manager order Page 25
7.8 Canteen Owner/manager Analysis Page 26
7.9 Admin Analysis Page 27
7.10 Admin Feedback Page 28
7.11 Admin Messaging Page 29
7.12 Admin Rating Page 30

List of Tables

6.1 Test Case 1: Login 16
6.2 Test Case 2: Homepage 16
6.3 Test Case 3: Admin 17
6.4 Test Case 4: Canteen Owner 17
6.5 Test Case 5: Email and Passwords 17

Chapter 1

Introduction

In today's age of fast canteen automation, many canteens have chosen to focus on quick preparation and speedy delivery of orders. Until very recently, all of this delivery of orders were placed over the phone, but there are many disadvantages to this system, including the inconvenience of the customers needing to have a physical copy of the menu or the bill, lack of visual confirmation that the order was placed correctly, and the necessity for the canteen to have an employee answering the calls or taking the orders. This application is developed with the motive to optimize the entire canteen process within the college. It will help the college in numerous ways right from every student to the management at a broader end. This application helps automate the daily functionalities of the canteen in a faster way, accurate manner, and also to reduce the manual burden. During regular college working days, the main issue in our college canteen is not too spacious to accommodate enough students all together to have lunch at the same time. Since, this is specific to our college canteen, where many students visit the canteen only during their break so they have limited time to eat. So, this application helps them save time and order food beforehand, and have it wherever locations are already in existence but this project is unique and developed based on the technology and the way of implementing the application. The most unique part is the AI Recommender feature. By fetching real-time data from the placed orders, this feature enables the user to have their Personalized recommendations along with the Most enjoyed dishes or frequently ordered by others. The application mainly consists of three major domains with the targeted users namely the administration, canteen management, and users (students and faculties). The main advantage of an online ordering system is that it greatly simplifies the ordering process for both the customer and the canteen. Once an order is placed on the web page, the same data is stored in the database and later retrieved in much real-time by the desktop application of the canteen. This allows the canteen to quickly fulfill the orders with minimum delay and confusion. The main objective of our application includes making it convenient for those who have less time. It allows users to give their orders beforehand makes them save time and enables them to eat wherever they like. To avoid complications, the user interface is kept simple and uncluttered for users' easy understanding. To make it cost-effective as online transactions are far more secure. This application being cross-platform has the same base code because of which any modifications can be easily adjusted in near future as per the requirements.

1.1 Objectives

- To make it convenient for those who have less time.

- As there is a common lunch break for all and the canteen being not too spacious this issue of space will be overcomed.

- To allow users to give their orders beforehand makes them save time and enables them to eat wherever they like. No need to wait for placing an order or waiting for their order to be prepared.

- To minimize the need to look for space in the canteen to have a seat, one could add a pickup location for their delivery making it easier to go.

- As one places order before hand and order will be ready in time so it could be picked up and ate at specific locations in the college campus

1.2 Problem Definition

- Problem Identified

  The main challenge encountered by our college is that of space being a little less to accommodate all the students at the same time. The experience of ordering food is not quick and there are a lot of complications while receiving the order. Manual system involves paper work in the form of taking orders and maintaining cash which is full of risk and tedious processes.

- Solution

  So this application not only is helping for canteen automation but also solves all of these concerns and makes a much easier and feasible atmosphere for every student as well as the manager. Every student will be able to access via their moodle id and a user generated password, by using which they can log into the system.

1.3 Benefits for environment

- Reduced manual labour work which makes it faster
- Automated process making it more efficient
- No paperwork involved
- Major factors namely space and time won't be compromised

1.4 Scope

Due to high demand of canteen during breaks makes it almost impossible to get your food sufficient time before start of the next class. Using our application aims to tackle this issue by allowing students to order beforehand and directly collect food from the canteen once the break starts. This helps in saving a lot of time that could have been wasted by standing in a queue and waiting for your order to be cooked.

Chapter 2

Literature Review

Paper titled 'Canteen Automation System' was published by authors Sanil Sharma, Pranchal Jain, Rinshi Jain, Roshni Gupta. It was published in fulfillment of degree of BACHELOR OF ENGINEERING in COMPUTER SCIENCE ENGINEERING from Gyan Ganga Institute of Technology and sciences, Jabalpur(M.P) in Dec(2016). A few notable findings related to our project Canteen Automation system is the system where customers order their food and receive food in the canteen without any delay as they can directly go and collect what they ordered without waiting.The system requires very fewer time factors as compared to manual.The system will provide fast and efficient automated environment instead of slow and error prone manual system.The system will have GUI interface and very less training is required to learn it. The main key advantages from this system were related to Time efficiency, user friendly ui and flexibility. Apart from that a few Disadvantages found were Complexity for developing as it follows three tier architecture. The main objective behind creating this application is to target college students. Here, most of the food delivery applications target a variety of different audiences. The college students usually prefer to have meals from their canteen as it is budget-friendly, unlike other apps that provide food delivery from various price ranges. Having the ability to directly order from the canteen at a specific time or getting it delivered to the local delivery spot makes our app feasible among college students.

Chapter 3

Proposed System

Orderista provides a complete digital solution for ordering food online across campus, improving ordering and service time drastically. It provides systems to help Admins and Canteen owners to monitor their progress and sales and improve their service with the help of user feedback. The app also docks in AI services to provide curated menus based on user preference to help order food. Alongside personal recommendations, AI can also provide a weekly accurate menu that a client can follow for a variety of choices to enjoy over the week. The technology stack of choice is Flutter for front-end development, NodeJS, and PostgreSQL for the back-end. The stack was selected due to its efficiency in development and performance quality. Flutter helps to create stunning UIs for both iOS and Android via a single code base hence making the overall process simpler. NodeJS and PostgreSQL provide faster communication and help retrieve data quicker and reduce the load on the server. Security is one of the aspects that was given utmost importance while creating the application. Upon Login JWT token is generated and is stored on the device. The token is useful to authenticate the user asynchronously when the user opens the app for the second time. This provides an immersive experience for the user. The JWT token can be only decrypted by the server to retrieve the data hence making the contents of the token secure. The flow of various entities is explained below.
User enters the credentials which are sent to the back-end where they are validated, and only upon their confirmation the user is allowed to use the features of the application.
Once the user is logged in, the user can order food, add to favourites, manage order history. On clicking the heart button on each food item card an api call is initiated which adds the user favourites into the backend.
Food items added to cart are saved locally using provider for the current session and when the payment is approved another api is called and the order is saved in the database which can be later used by the canteen owner to confirm the order once the order is picked up.

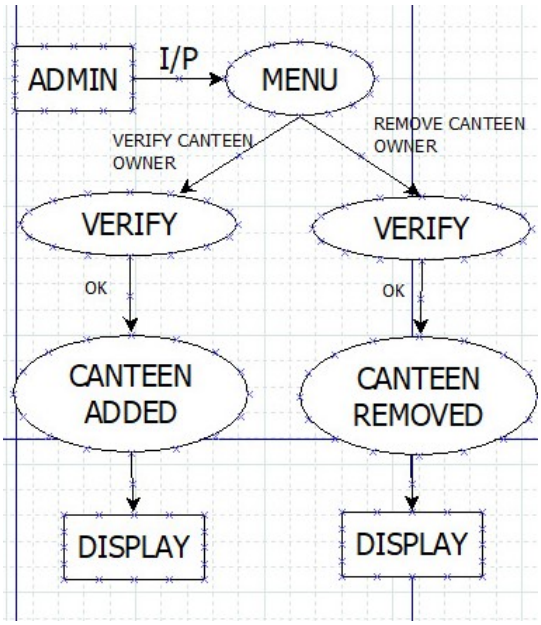

Figure 3.1: Admin Portal
When a user logs in with the role of Admin then he or she can add users or canteen manager and can have all the rights over the application.

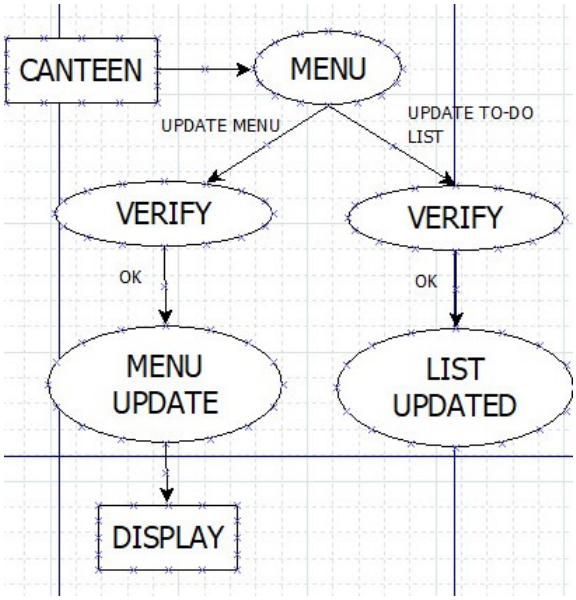

Figure 3.2: Canteen Manager Portal
When a user logs in with the rights of Canteen managers end then he can receive the users order and can change the status as per the state order is in and can add the menu or change them on a daily basis.

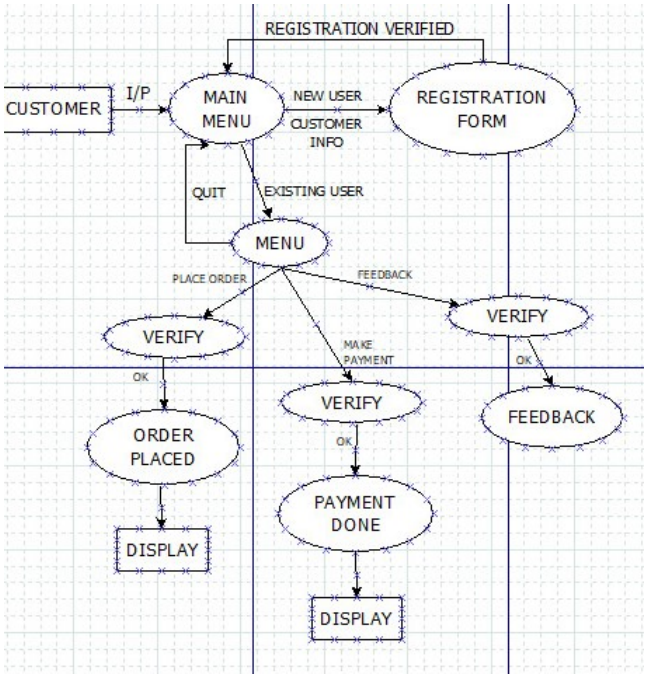

Figure 3.3: Customer Portal

If a User logs in Student/Faculty he will be first viewed on the home page where in all possible dishes will be available where in the selected dishes can be added to cart to place an order

Chapter 4

Design

4.1 Use Case Diagram

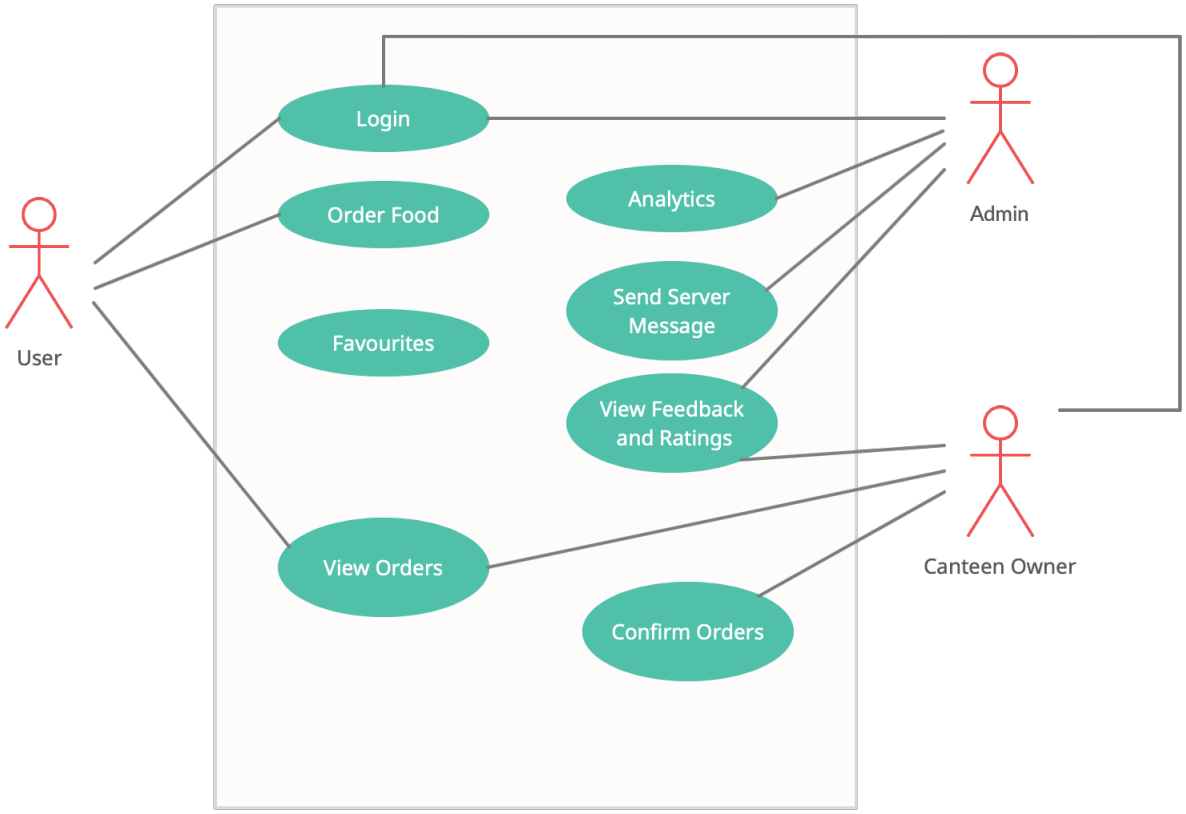

Figure 4.1: Use Case Diagram
User can login, order food, view their orders, Admin and canteen owners can check analytics, send messages, view feedback and rating

4.2 Activity Diagram

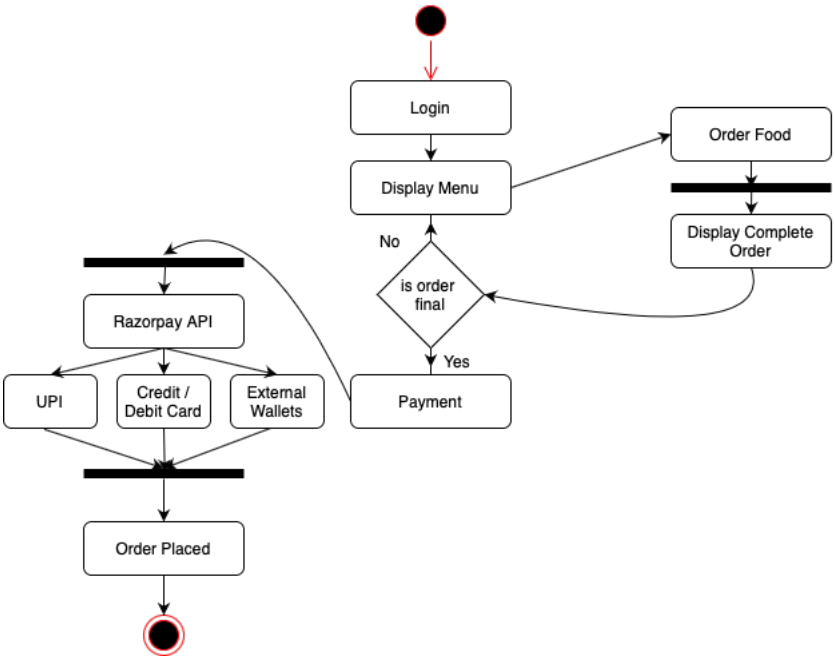

Figure 4.2: Activity Diagram

According to the activity diagram, user must first login to the app, a list of menu will be displayed and the user can add items to the cart as per their convenience. Once they have added items to the cart the entire order is displayed. If the user is not satisfied with the order, they can go back and alter it. If satisfied the user can initiate the payment. Upon approval of the payment the user can visit the canteen to pick up the order.

4.3 Class Diagram

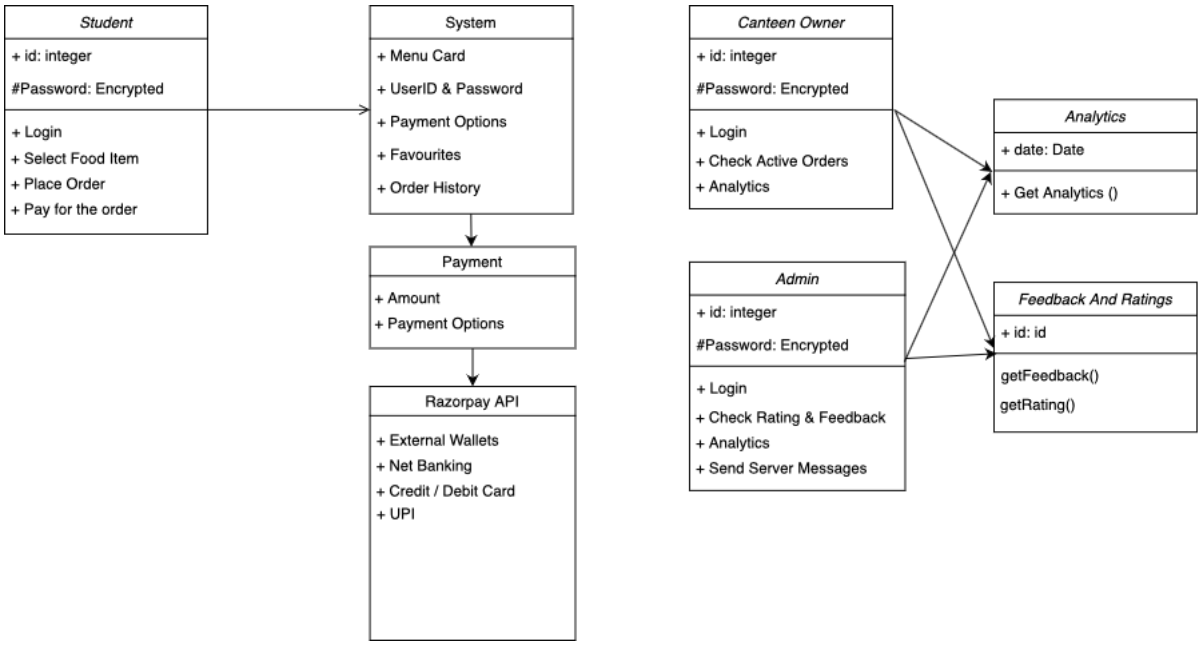

Figure 4.3: Class Diagram

This above diagram depicts the class diagram of the project which consists of the different user types, modules and the attributes and how they are linked.

Chapter 5

Implementation

```
import 'dart:convert';
import 'package:flutter_dotenv/flutter_dotenv.dart';
import 'package:http/http.dart' as http;

String domain = env['API_URL'];

Future httpPost(String route, [dynamic data]) async {
  var dataStr = jsonEncode(data);
  var url = "$domain/$route";
  var result = await http
      .post(url, body: dataStr, headers: {"Content-Type": "application/json"});
  return jsonDecode(result.body);
}

Future httpGet(String route, [dynamic data]) async {
  var url = "$domain/$route";
  var result =
      await http.get(url, headers: {"Content-Type": "application/json"});
  return jsonDecode(result.body);
}
```

Figure 5.1: API Call Functions

```
void retrieveMenu() async {
  if (this.mounted) {
    setState(() {
      _isFetching = true;
    });
  }
  var menu = await httpGet("api/v1/menu/food");
  _menuList = menu["menu"];
  print(_menuList);
  for (var u in _menuList) {
    Food food = Food(
        id: u["dishid"],
        category: u["dishtag"],
        name: u["dishname"],
        price: u["dishcost"],
        imagePath: u["dishimage"]);

    Provider.of<Menu>(context, listen: false).addFoodItemsToProvider(food);
  }

  if (this.mounted) {
    setState(() {
      _isFetching = false;
    });
  }

  prefs.setBool("shouldFetchMenu", false);
}
```

Figure 5.2: Fetching Menu from Back-end

```
Padding(
  padding: const EdgeInsets.all(16.0),
  child: query == ""
      ? Consumer<Menu>(
          builder: (context, menu, child) {
            return Column(
                children: menu.foodItems
                    .map(_buildList)
                    .toList()); // Column
          },
        ) // Consumer
```

```
Widget _buildList(Food food) {
  return FoodItem(
    name: food.name,
    id: food.id,
    cost: food.price,
    tag: food.category,
    imagePath: food.imagePath,
  );
}
```

Figure 5.3: Building Homepage Items

```
//Razorpay API
_razorpay = Razorpay();
_razorpay.on(Razorpay.EVENT_PAYMENT_SUCCESS, _handlePaymentSuccess);
_razorpay.on(Razorpay.EVENT_PAYMENT_ERROR, _handlePaymentError);
_razorpay.on(Razorpay.EVENT_EXTERNAL_WALLET, _handleExternalWallet);
```

Figure 5.4: Initializing Razorpay API

```
void openCheckout() async {
  int amount = Provider.of<Cart>(context, listen: false).amount;
  print(amount);
  var options = {
    'key': 'rzp_test_i3aeiM0nABKvDz',
    'amount': amount * 100,
    'name': 'A.P Shah Institute of Technology',
    'description': 'Payment for Food',
    'prefill': {'contact': '1234567890', 'email': 'test@razorpay.com'},
    'external': {
      'wallets': ['paytm']
    }
  };

  try {
    _razorpay.open(options);
  } catch (e) {
    print(e.toString());
  }
}
```

Figure 5.5: Razorpay Checkout

```python
@app.route("/recommend", methods=["POST"])
def recommendation():
    try:
        menu_df = pd.read_csv('AI_Data.csv')
        rating_df = pd.read_csv('FoodRating.csv')
        df = pd.merge(rating_df, menu_df, on='foodid')
        rating_features_df = df.pivot_table(
            index='name', columns='userid', values='rating').fillna(0)
        rating_features_df_matrix = csr_matrix(rating_features_df.values)
        model_knn = NearestNeighbors(metric='cosine', algorithm='brute')
        model_knn.fit(rating_features_df_matrix)
        recos = []

        def recommend(index):
            query_index = index
            print(query_index)
            distances, indices = model_knn.kneighbors(
                rating_features_df.iloc[query_index, :].values.reshape(1, -1), n_neighbors=4)
            rating_features_df.head()
            indices.flatten()
            distances.flatten()
            for i in range(0, len(distances.flatten())):
                if i == 0:
                    print('Recommendations for {0}:\n'.format(
                        rating_features_df.index[query_index]))
                else:
                    recos.append(
                        rating_features_df.index[indices.flatten()[i]])

            return recos

        reco = recommend(10)

        return jsonify({
            "recommendation": reco,
            "msg": "Recommendation"
        })
    except Exception as e:
        return jsonify({
            "error": e
        })
```

Figure 5.6: AI Implementation using KNN

Chapter 6

Testing

Test No.	Test Name	Expected Result	Actual Result	Pass or Fail
1.	Empty MoodleID Field	Enter MoodleID	Please Enter Moodle ID	Pass
2.	Empty Password Field	Enter Password	Please Enter Password	Pass
3.	Incorrect Moodle ID	Incorrect Moodle ID	No user found	Pass
4.	Incorrect Password ID	Incorrect Password	Invalid Password	Pass

Table 6.1: Test Case 1: Login

Test No.	Test Name	Expected Result	Actual Result
1.	Menu Fetching	Json Object with menu	Json object with menu
2.	Retrieve Feedback Status	Display pop up if last order not rated	Display pop up if last order not rated
3.	Server Messages	Display when message available	Display when message available
4.	Add to Cart	Add item to cart	Add item to cart

Table 6.2: Test Case 2: Homepage

Test No.	Test Name	Expected Result	Actual Result
1.	Feedback	Json object with feedbacks	Json object with feedbacks
2.	Ratings	Json object with ratings	Json object with ratings
3.	Create Server Message	Message sent successfully	Message sent successfully
4.	Analytics	Data visualization based on date range	Data visualization based on date range

Table 6.3: Test Case 3: Admin

Test No.	Test Name	Expected Result	Actual Result
1.	Orders	List of all active orders	List of all active orders
2.	Scan Order	If correct, completed order button else incorrect order message	If correct, complete order button else incorrect order message
3.	Analytics	Data visualization based on date range	Data visualization based on date range

Table 6.4: Test Case 4: Canteen Owner

Test No.	Test Name	Expected Result	Actual Result
1.	Email Verification	OTP sent for verification	OTP sent
2.	Change Password	OTP verification, change password	OTP verification, change password

Table 6.5: Test Case 5: Email and Passwords

Chapter 7

Result

7.0.1 Customer Portal

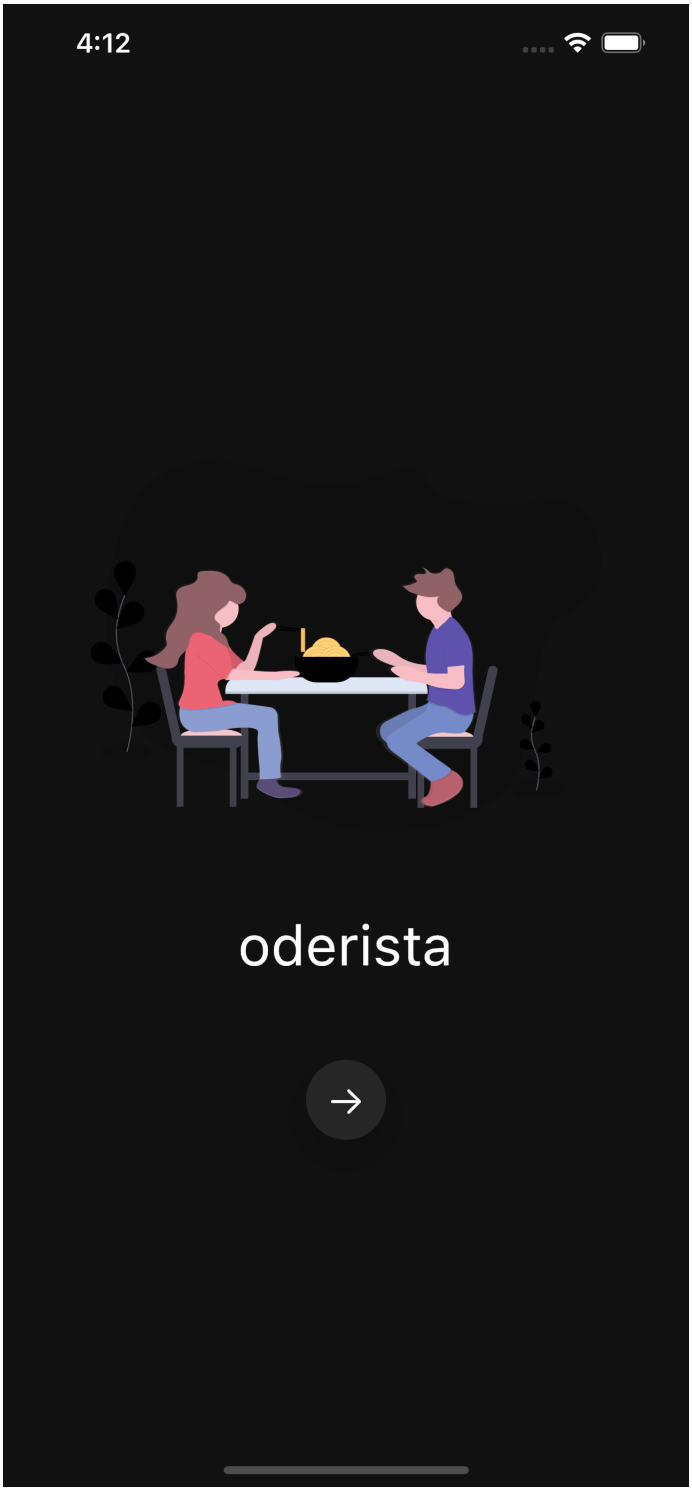

Figure 7.1: Application Landing Page

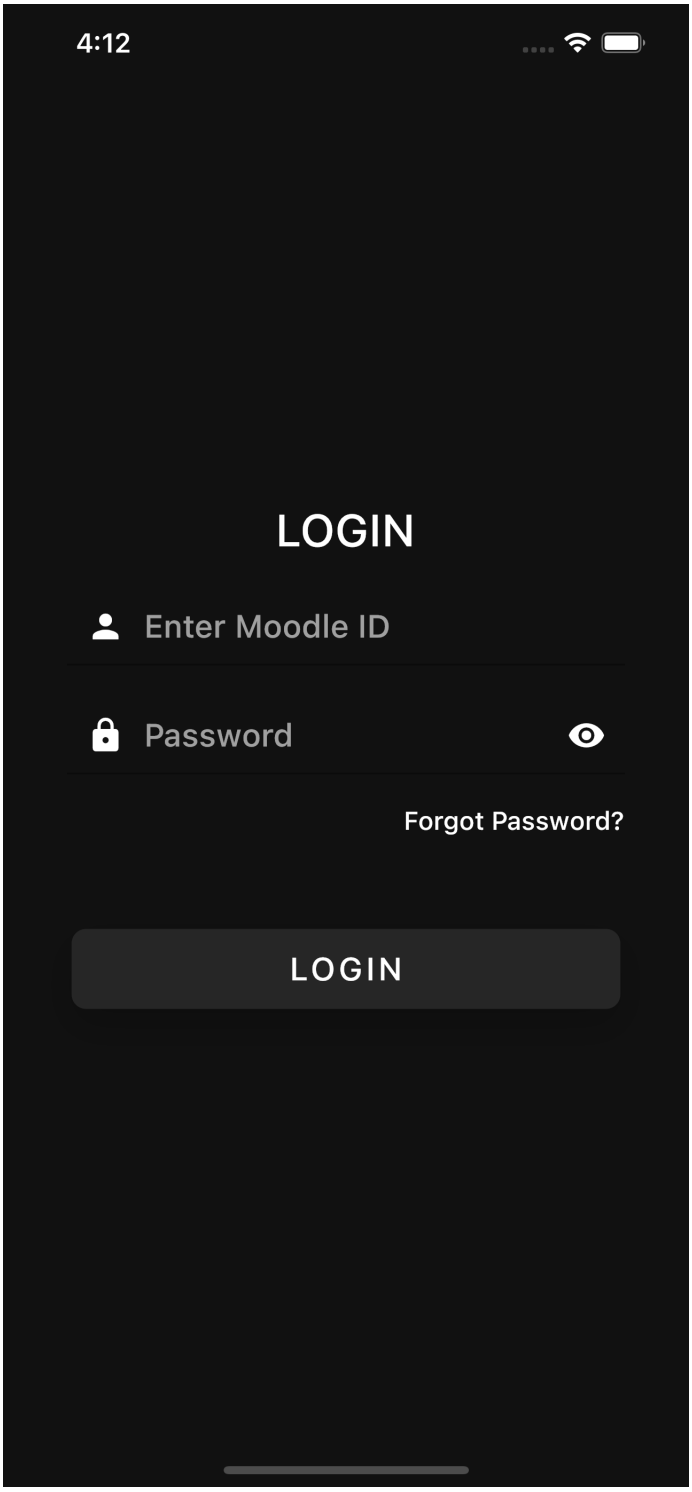

Figure 7.2: Login Page

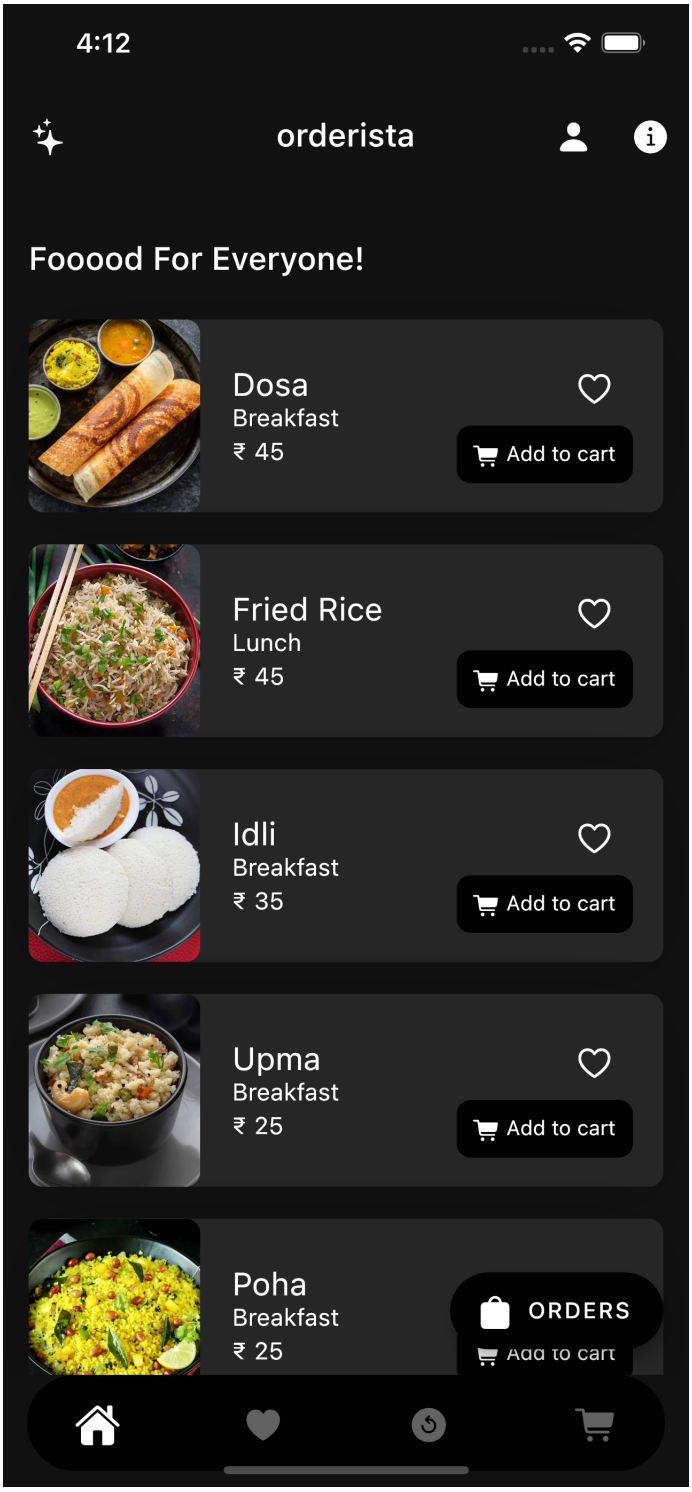

Figure 7.3: Home Page

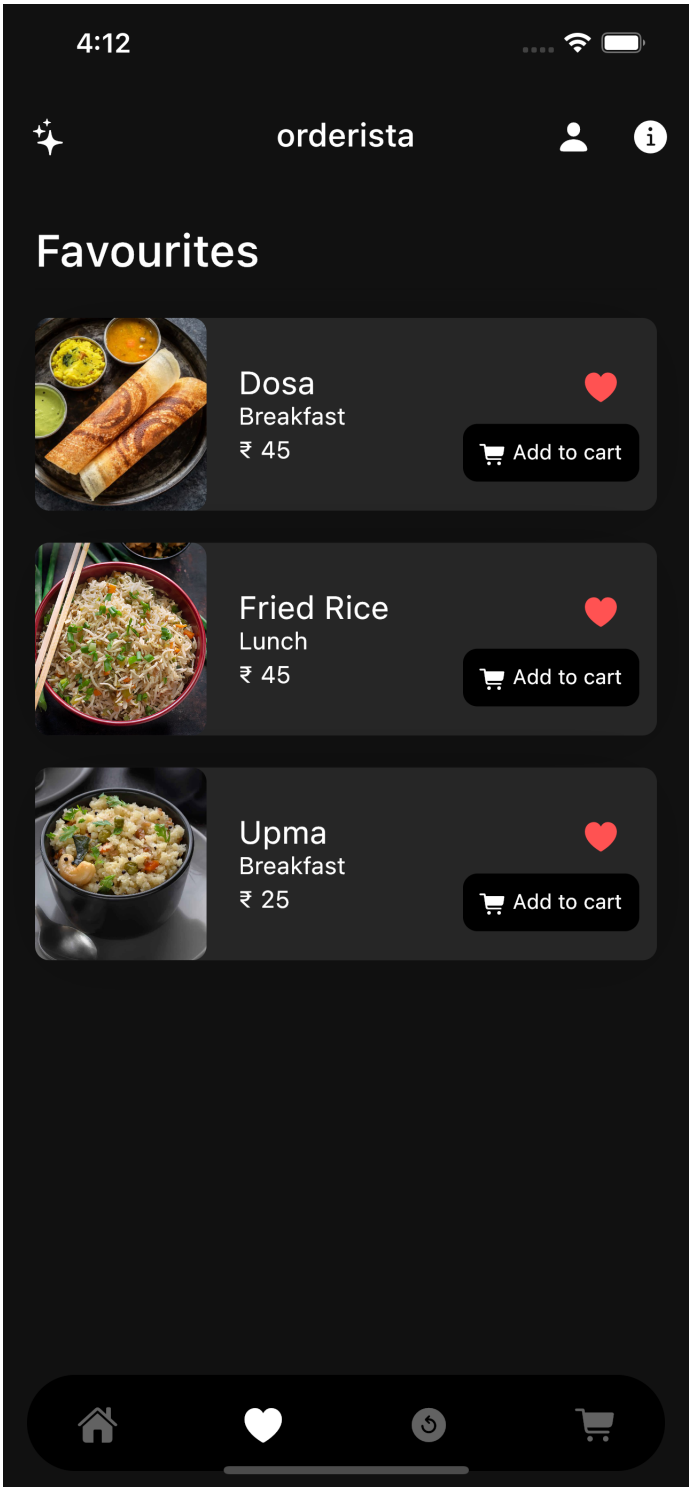

Figure 7.4: Favourites Page

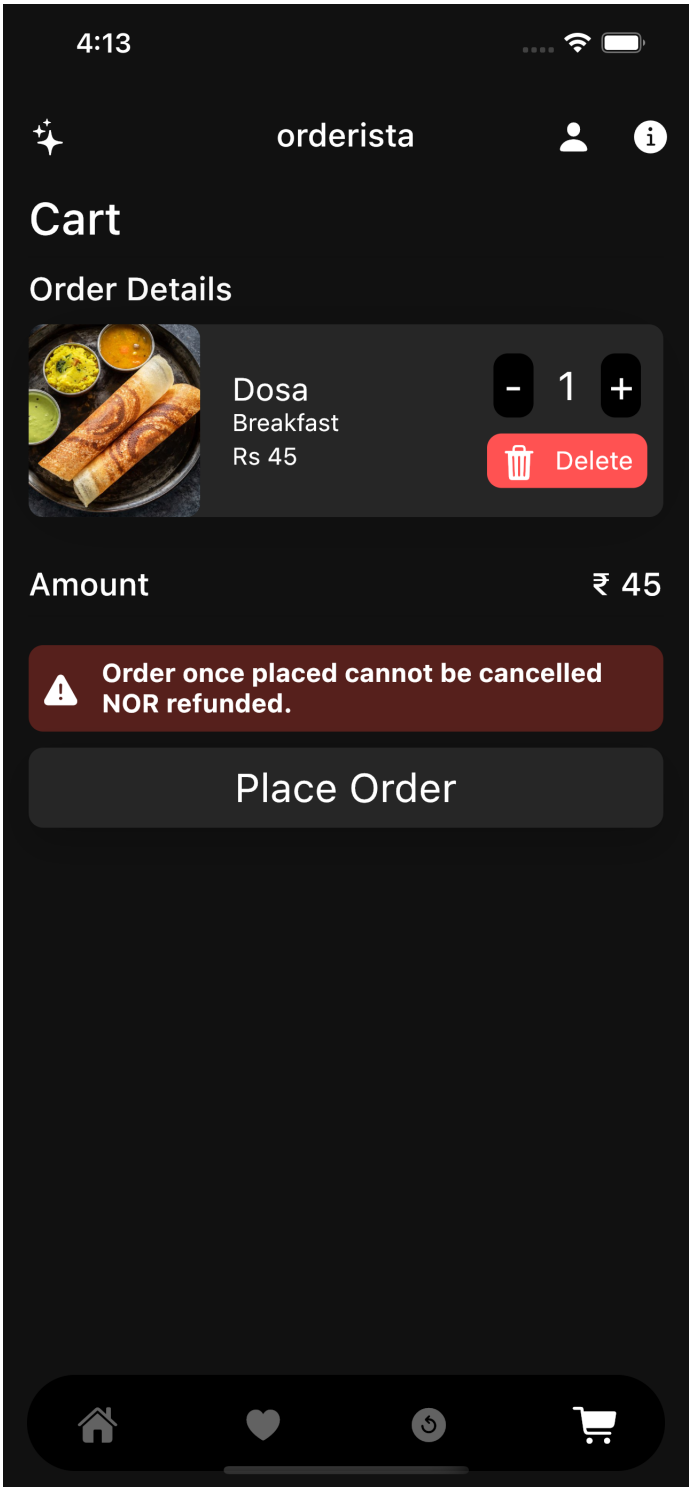

Figure 7.5: Cart Page

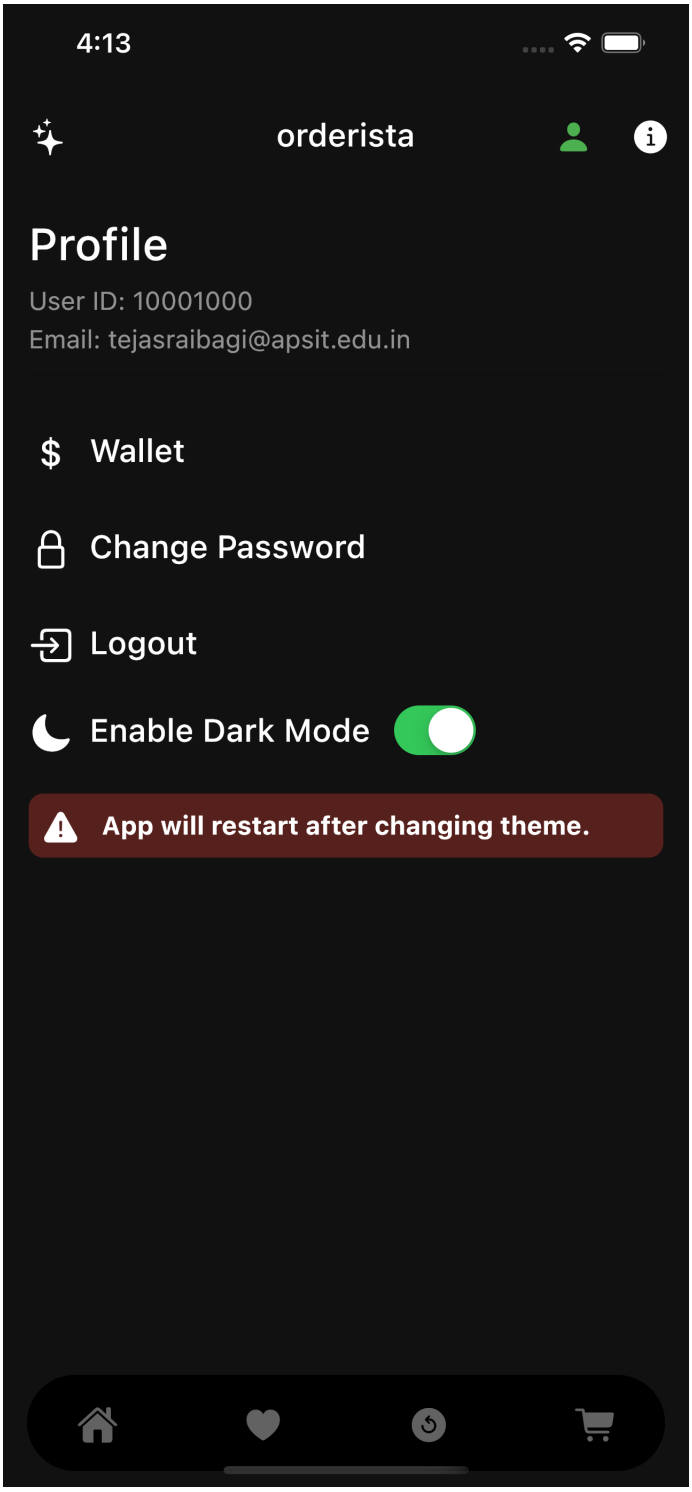

Figure 7.6: Profile Page

7.0.2 Canteen Owner/ Manager Portal

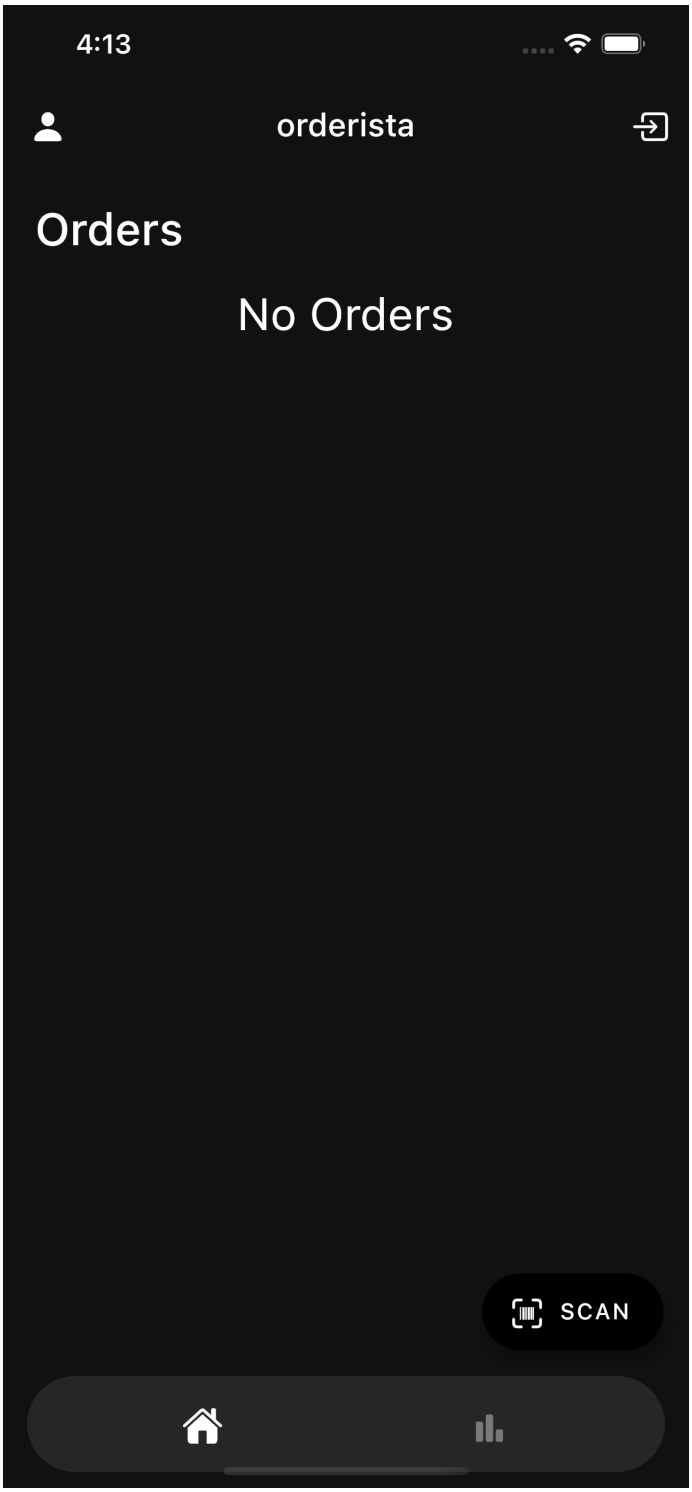

Figure 7.7: Canteen Owner/manager order Page

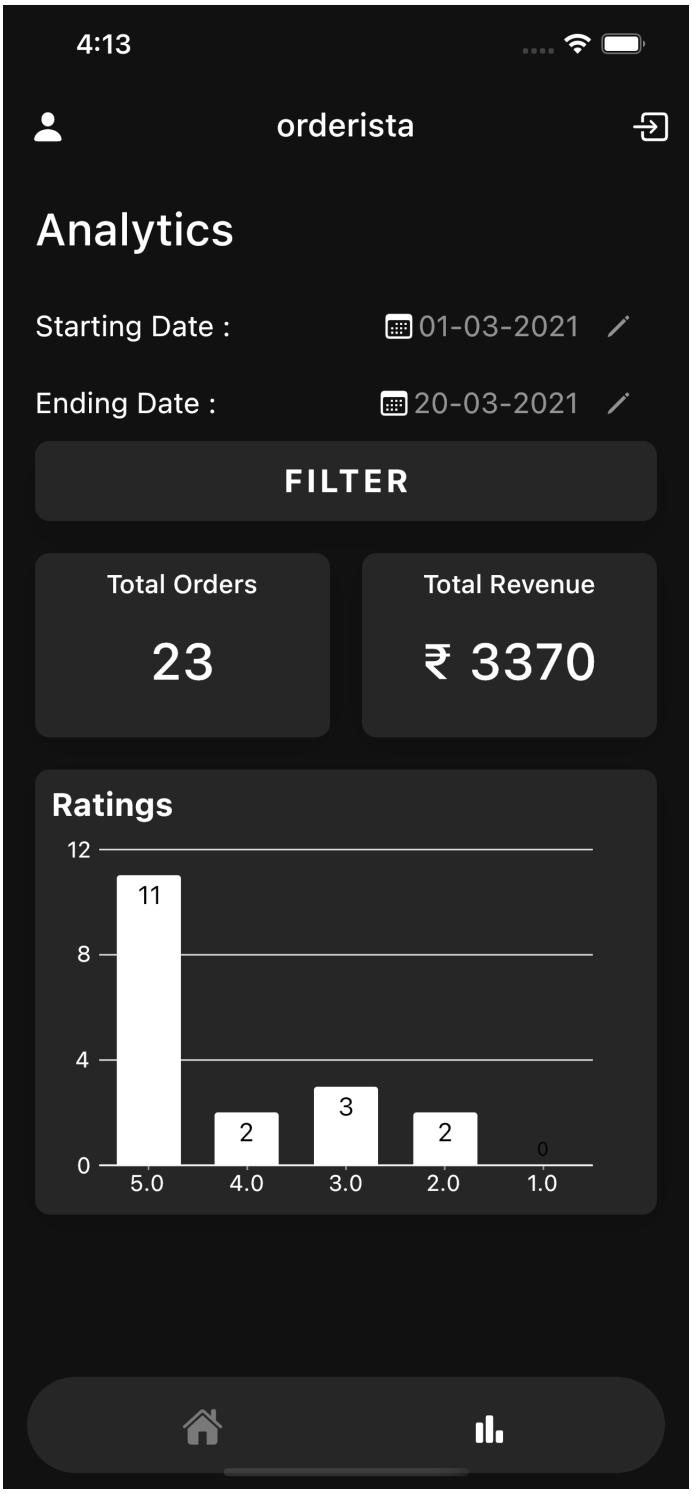

Figure 7.8: Canteen Owner/manager Analysis Page

7.0.3 Admin Portal

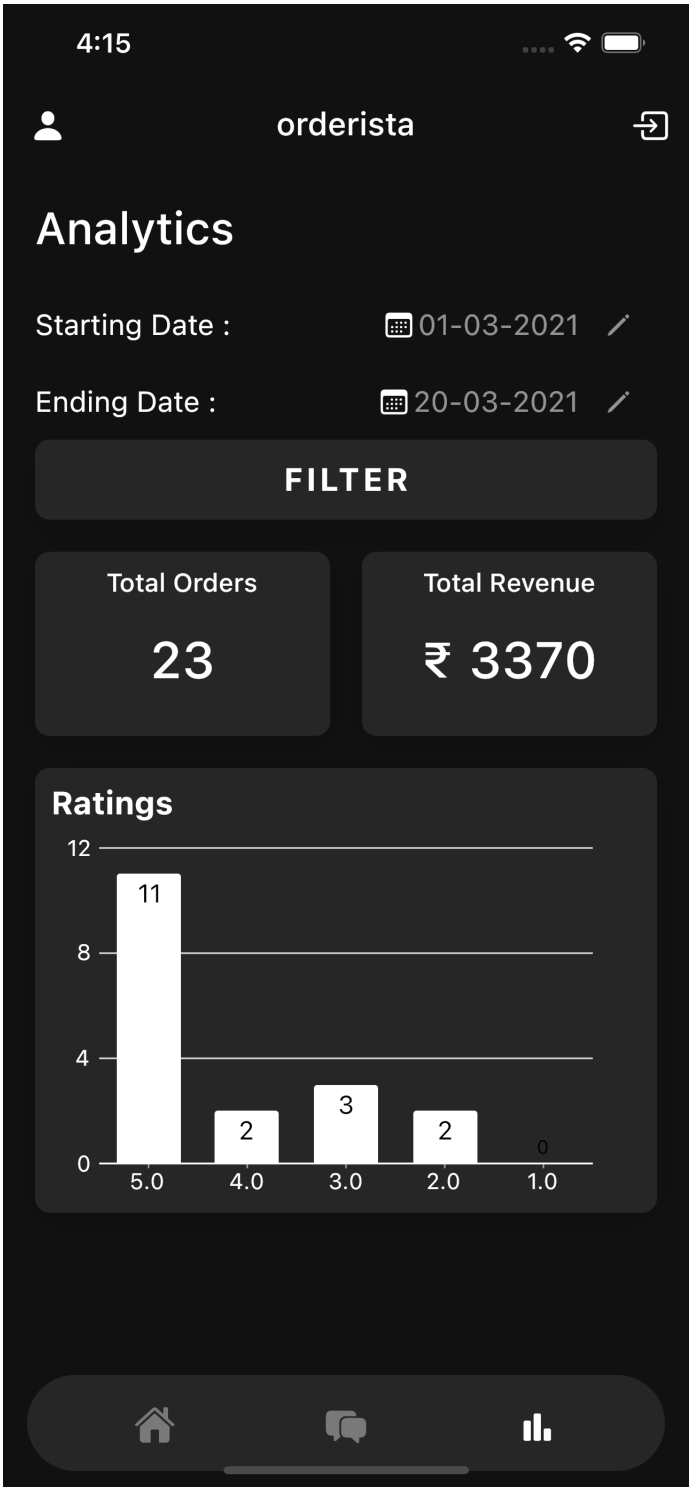

Figure 7.9: Admin Analysis Page

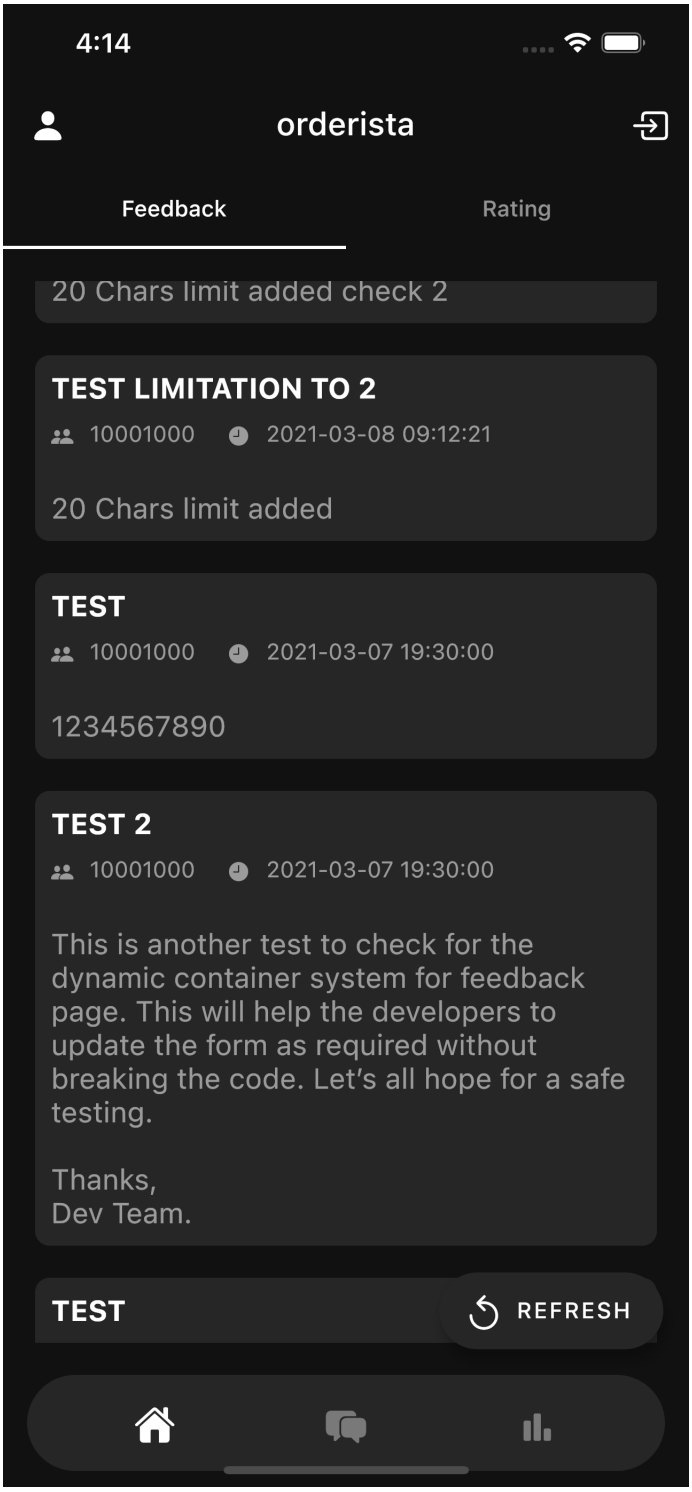

Figure 7.10: Admin Feedback Page

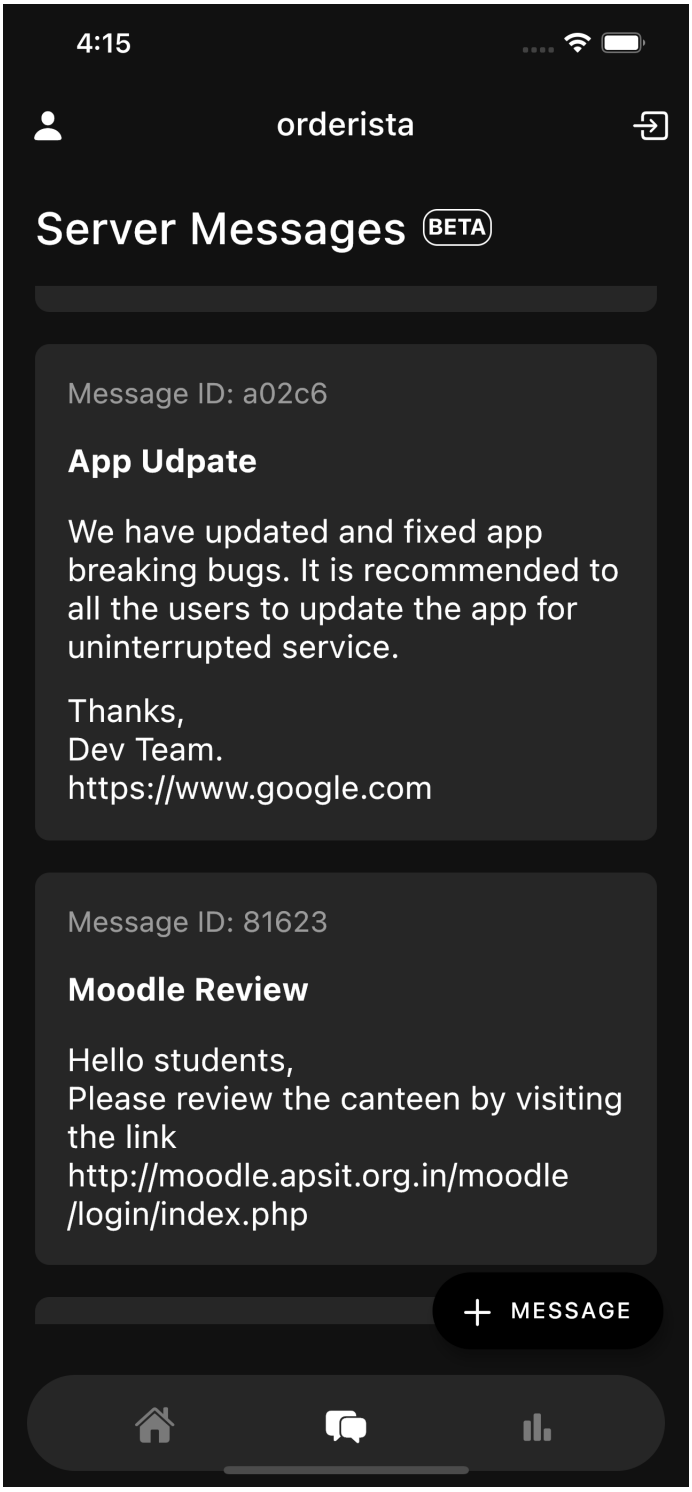

Figure 7.11: Admin Messaging Page

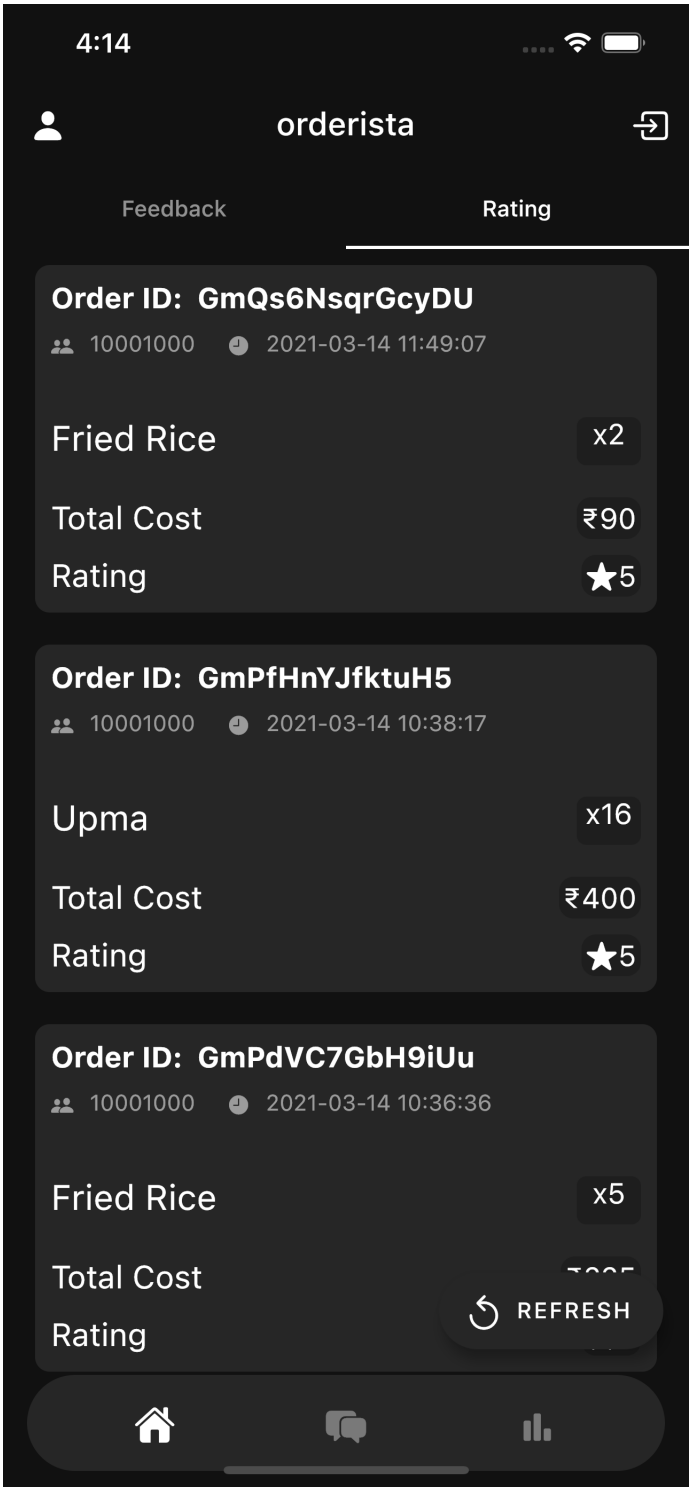

Figure 7.12: Admin Rating Page

Chapter 8

Conclusions and Future Scope

In the proposed system, an AI-based cross-platform application is developed to automate the daily canteen functionalities. This ensures to enable the end-user to register online, select the food they plan to have for their lunch from the e-menu card, and place the order online by selecting the food that aims to reduce the load in the canteen. Here, the entire process of taking the order and serving the food, which is automated utmost time in our day-to-day life. The system mainly will be aiming to reduce the load on the canteens end, as the entire process of taking orders and serving is automated. It also aims at contributing for a better environment as it Digitalizes every aspect of the data for better analytics.

Bibliography

[1] www.irjet.net

[2] http://ijirt.org/Article?manuscript=147528

[3] http://www.ijsrd.com/articles/IJSRDV7I20481 .pdf

[4] https://nevonprojects.com/canteen-automation-system

[5] https://www.slideshare.net/100008381806318/cantee n-automation-system-updated-revised

[6] https://phpgurukul.com/food-ordering-system-using-p hp-and-mysql

[7] http://ijarcet.org/wp-content/uploads/IJARCET-VOL 9-ISSUE-3-91-97.pdf

Acknowledgement

We have great pleasure in presenting the report on **Orderista.** We take this opportunity to express our sincere thanks towards our guide **Mrs. Rujata Chaudhari** & Co-Guide **Mrs. Geetanjali Kalme** Department of IT, APSIT thane for providing the technical guidelines and suggestions regarding line of work. We would like to express our gratitude towards his constant encouragement, support and guidance through the development of project.

We thank **Prof. Kiran B. Deshpande** Head of Department,IT, APSIT for his encouragement during progress meeting and providing guidelines to write this report.

We thank **Prof. Vishal S. Badgujar** BE project co-ordinator, Department of IT, APSIT for being encouraging throughout the course and for guidance.

We also thank the entire staff of APSIT for their invaluable help rendered during the course of this work. We wish to express our deep gratitude towards all our colleagues of APSIT for their encouragement.

Student Name1: Jahnavi Naik
Student ID1: 17104046

Student Name2: Ashwin Vishwakarma
Student ID2: 17104009

Student Name3: Tejas Raibagi
Student ID3: 17104067

Publication

Paper entitled **"Orderista - An AI based food ordering system"** is presented at **"IEEE International Conference on Artificial Intelligence and Smart Systems (ICAIS 2021)"** by **"Jahanvi Naik, Ashwin Vishwakarma, Tejas Raibagi, Rujata Chaudhari, Geetanjali Kalme".**