

# ONLINE FOOD DELIVERY SYSTEM

## **Team Lead:**

Nellore Sri Lavanya

## **Frontend Lead:**

Pasikanti Akanksha

## **Backend Lead:**

Nikhil Ravindrabhau Botare

## **Team Members:**

- |     |                            |           |
|-----|----------------------------|-----------|
| 1.  | Nellore Sri Lavanya        | - 2487065 |
| 2.  | Nikhil Chaurasiya          | - 2486837 |
| 3.  | Nikhil Ravindrabhau Botare | - 2487734 |
| 4.  | Nikita Pandharinath Patole | - 2487699 |
| 5.  | Nithya K J                 | - 2487703 |
| 6.  | Panthagani Vikas Kumar     | - 2483838 |
| 7.  | Pallavi A Sarapure         | - 2487133 |
| 8.  | Pasikanti Akanksha         | - 2484032 |
| 9.  | Pokuru lokesh              | - 2485815 |
| 10. | Pooja X                    | - 2487194 |

<b>Project Code</b>	6
<b>Project Name</b>	Online Food Delivery System

## Table of Contents

<b>1. Introduction</b>	<b>3</b>
<b>2. System Overview</b>	<b>3</b>
<b>3. Sub-System Details</b>	<b>6</b>
<b>4. Data Organization</b>	<b>8</b>
<b>5. REST APIs to be Built</b>	<b>12</b>
<b>6. Implementation Details</b>	<b>17</b>
<b>7. Functional Testing</b>	<b>17</b>
<b>8. Conclusion</b>	<b>25</b>

## 1. Introduction

It is known globally that, in today's market, it is extremely difficult to start a new small-scale business and live-through the competition from the well-established and settled owners. In fast paced time of today, when everyone is squeezed for time, the majority of people are finicky when it comes to placing a food order. The customers of today are not only attracted because placing an order online is very convenient but also because they have visibility into the items offered, price and extremely simplified navigation for the order.

Online ordering system that I am proposing here, greatly simplifies the ordering process for both the customer and the restaurant. System presents an interactive and up-to-date menu with all available options in an easy to use manner. Customer can choose one or more items to place an order which will land in the Cart. Customer can view all the order details in the cart before checking out. At the end, customer gets order confirmation details. Once the order is placed it is entered in the database and retrieved in pretty much real time. This allows Restaurant Employees to quickly go through the orders as they are received and process all orders efficiently and effectively with minimal delays and confusion.

### 1.1. Scope and Overview:

The scope of the “**Online Food Delivery System**” will be to provide the functionality as described below. The system will be developed on a windows operating system Using Angular, Spring-Boot, Hibernate, MySQL.

## 2. System Overview

The “**Online Food Delivery System**” should support basic functionalities (explained in section 2.1) for all below listed users.

- Administrator (A)
- Restaurants (R)
- Customer (c)

### 2.1. Authentication & Authorization

#### 2.1.1 Authentication:

Any end-user should be authenticated using a unique user id and password.

#### 2.1.2 Authorization

List of operations are done by Admin, Restaurant and Customers.

**Admin:**

- Admin can add Restaurants details with user name and password.
- Admin can edit and delete Restaurants details.

**Restaurants:**

- Restaurant can add their food category details.
- Restaurant can add their available food items with price.
- Restaurant can see Customers orders.
- Proceed for preparing food.
- Sending order to customers.

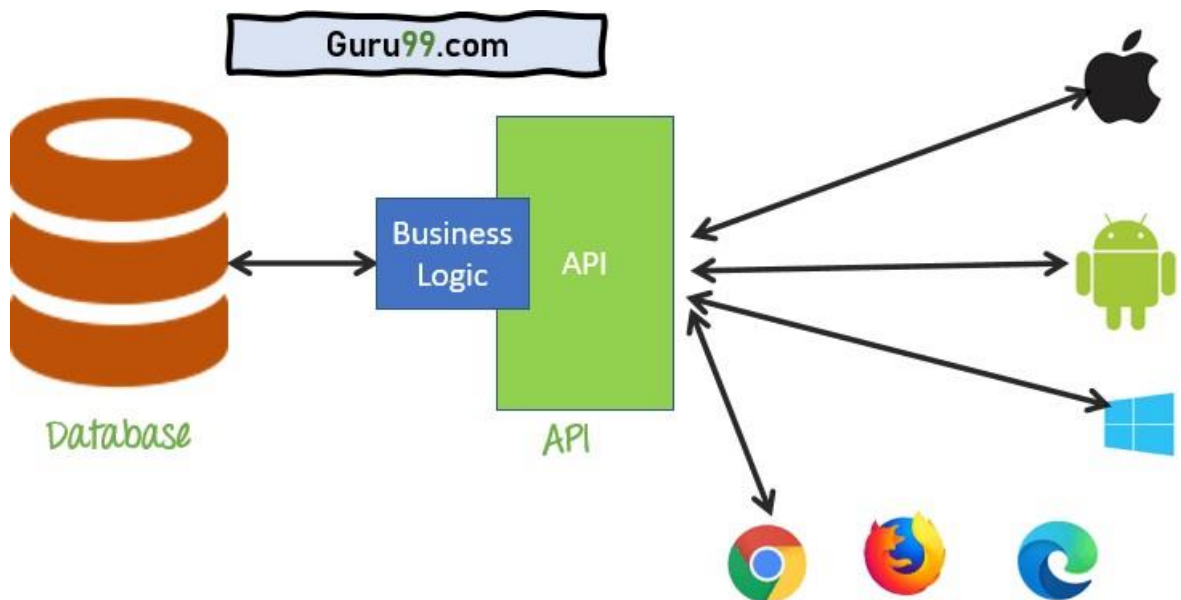
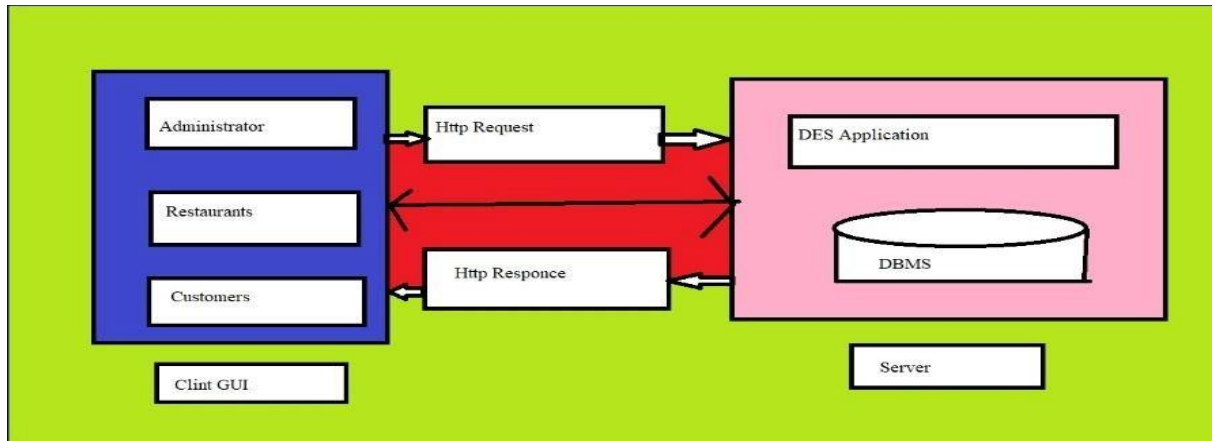
**Customers:**

- Register their accounts in signup.
- After register successfully user can login.
- After login they can find numbers of Restaurants and select best one.
- Customers can see food items present in Restaurant.
- Add to cart or Order by proceeding amount.
- Customers can check their status once order a food.

## 2.2 Functional Flow

The functional flow of the messages across different application components is shown below.

Ex. - Web Application.



## 2.3 Environment:

The system will be developed on any Windows OS machine using J2EE, Hibernate and Spring-Boot, MySQL.

- Intel hardware machine (PC P4-2.26 GHz, 512 MB RAM, 40 GB HDD)
- Server – Apache Tomcat 8 or higher
- Database – MySQL
- JRE 8
- Eclipse IDE or Spring Tool or IntelliJ IDEA.

GITHUB LINK:<https://github.com/NelloreLavanya/AdvancedTraining-java.git>

## FUNCTIONAL SPECIFICATION:

### 3. Sub-system Details:

The “**Online Food Delivery System**” should be defined, where in all users need to login successfully before performing any of their respective operations. Find below tables that provide functionality descriptions for each type of user / sub-system. Against each requirement, indicative data is listed in column ‘Data to include’. Further, suggested to add/modify more details wherever required with an approval from customer/faculty.

#### 3.1 Administrator

The administrator as a user is defined to perform below listed operations after successful login.

ID	Objects	Operation	Data to include	Remarks
1	Manage Restaurant	Add, View, Delete, Modify	Restaurant name, Contact number, Email, Address. GST Number	

#### 3.2 Restaurants:

The Restaurants as a user is defined to perform below listed operations after successful login.

ID	Objects	Operations	Data to include	Remarks
1	Manage Category	Add, view, Modify	Adding food Category	

2	Manage Product	Add, view, delete, Modify	Adding number of food items present in restaurants.	
3	New Orders	Showing Orders from customers	Orders from customers	
4	Preparing Orders	Prepare food	Proceed to prepare	
5	Completed Orders	Return to customers	Return to customer	

### 3.3 Customer:

The customer as a user is defined to perform below listed operations after successful login.

ID	Objects	Operation	Data to include	Remarks
1	User	Register	Name, Contact Number, Email	
2	User	Login	Email, Password	
3	Restaurants	Choose Restaurant	View Product	
4	Product	Add to cart, Delete from cart, Delete all the products from cart, Proceed to pay	Product id, Product Name, Price, Quantity, Total Price.	
5	Your Order	Check status of your food	Status	

## FUNCTIONAL SPECIFICATION

### 3.3 Login| Logout:

{Web Application – Angular, Spring-Boot, Hibernate, MySQL}

- Go to Registration screen when you click on Register link.
- Go to Success screen when you login successfully after entering valid username & password fetched from the database.
- Redirect back to same login screen if username & password are not matching.

## 4. Data Organization

This section explains the data storage requirements of the Online Food Delivery System and indicative data description along with suggested table (database) structure. The following section explains few of the tables (fields) with description. However, in similar approach need to be considered for all other tables.

### 4.1 Table: customer details ( database name: customer):

In this table we can find all users, Restaurant id's, Admin details we can find here.

Authentication, and authorization / privileges should be kept in one or more tables, as necessary and applicable.

Field Name	Description
Email(Primary Key)	Customer user name
Address	Customer address
Contact	Customer contact number
Name	Customer name
Password	Password given by user

### 4.2 Table: Admin Details (Database name: admin):

In this table contains all restaurant details.

Field Name	Description
Username(Primary Key)	Admin username
Password	Password for admin login

### 4.3 Table: Product Details (Database name: product):

Field Name	Description
Id(Primary Key)	Automatically generate by system
Actualprice	Enter price of product



avail	Say yes or no for availability of product
Category	Category of product adding
Des	Write description of product
Discount	Enter the discount amount
Imagepath	Enter the path of image adding
Name	Enter the name of the product
Price	Enter the price of product

#### 4.4 Table: Cart (database name: cart)

Here we can add food to cart

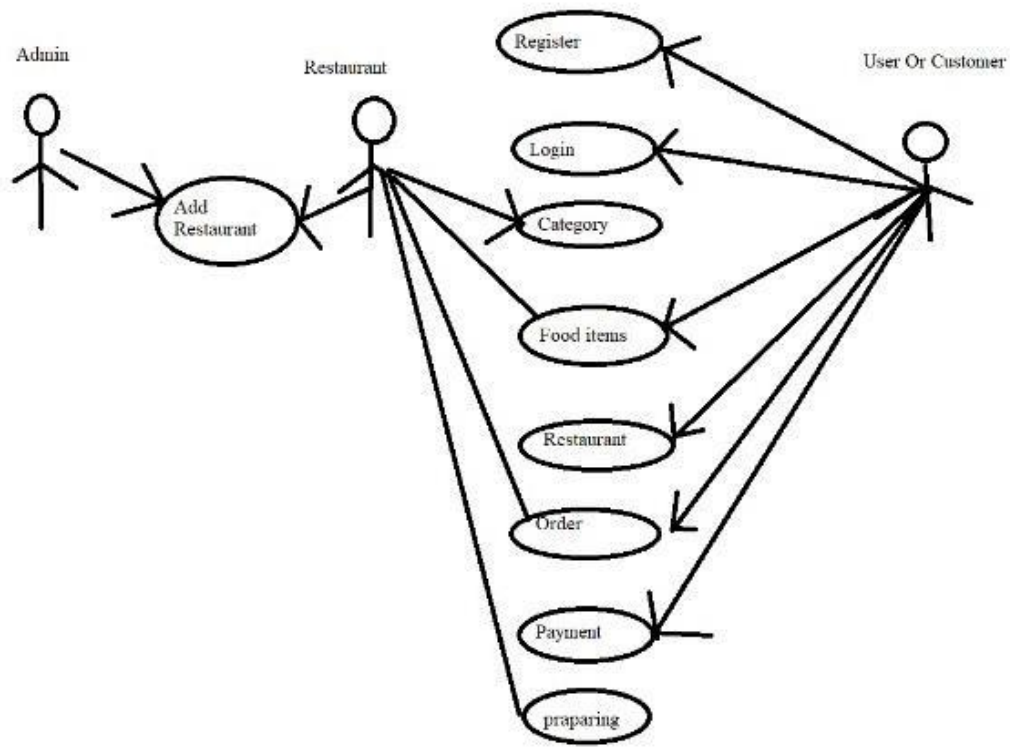
Field Name	Description
Id(Primary key)	System automatically generate
Price	Item price
Quantity	Quantity of order
ProductID	ID of product

#### 4.5 Table: User Purchase (Database name: purchase):

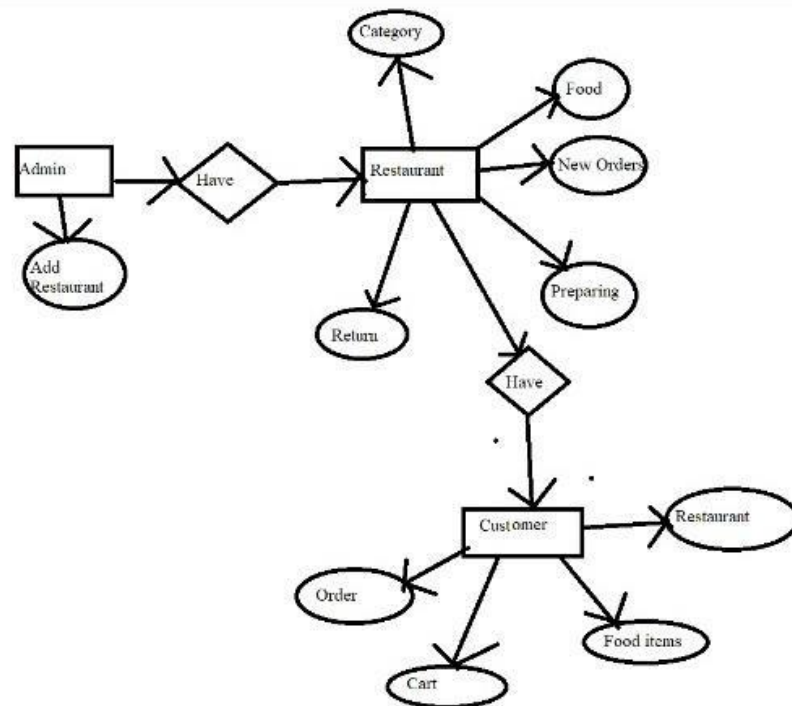
We can get in below table users transaction details.

Field Name	Description
Id(Primary Key)	System automatically generate
Dop	Date of purchase
Productname	Name of the product
Quantity	Quantity of product
Totalcost	Total cost of order
Transactionid	Id generated for transaction
customeremail	Email id of customer placed an order

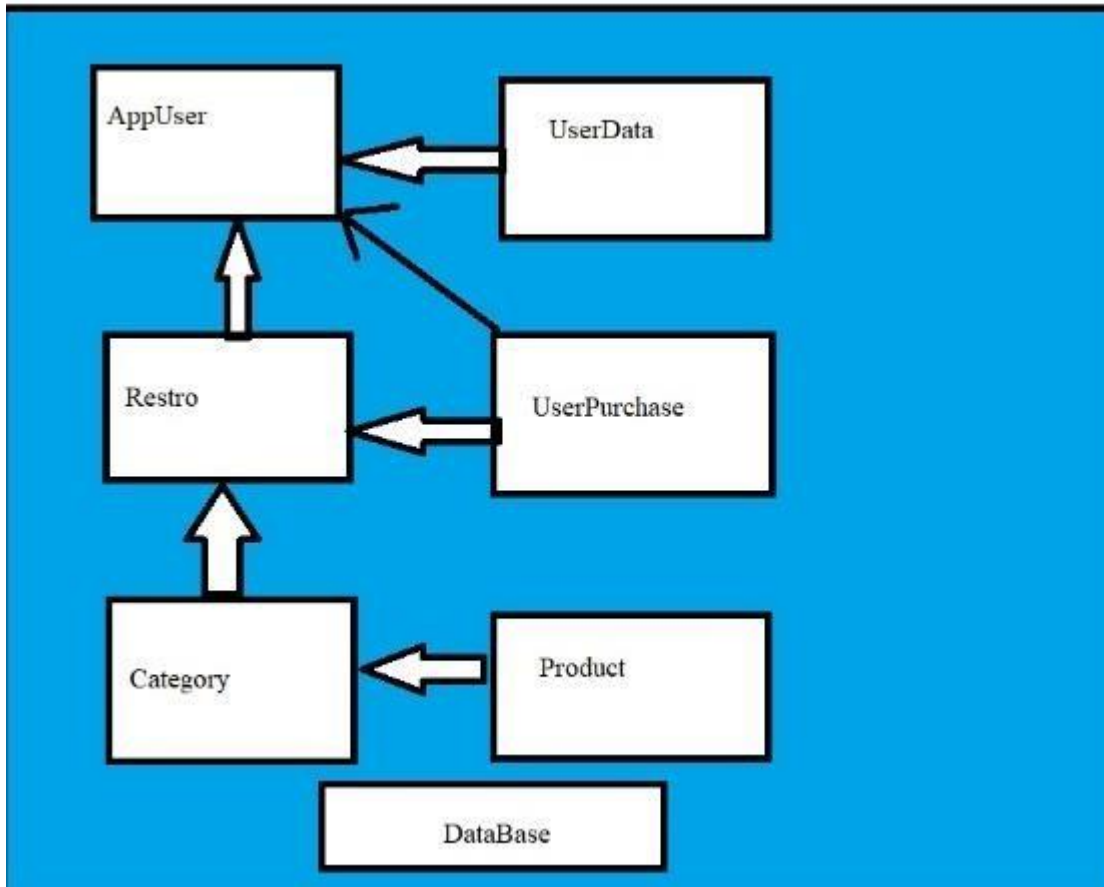
#### 4.6 UML Diagram:



#### 4.5 Entity Relationship Diagram:



#### 4.6 Database Schema:



## FUNCTIONAL SPECIFICATION

### 5 REST APIs to be Built.

Create following REST resources which are required in the application,

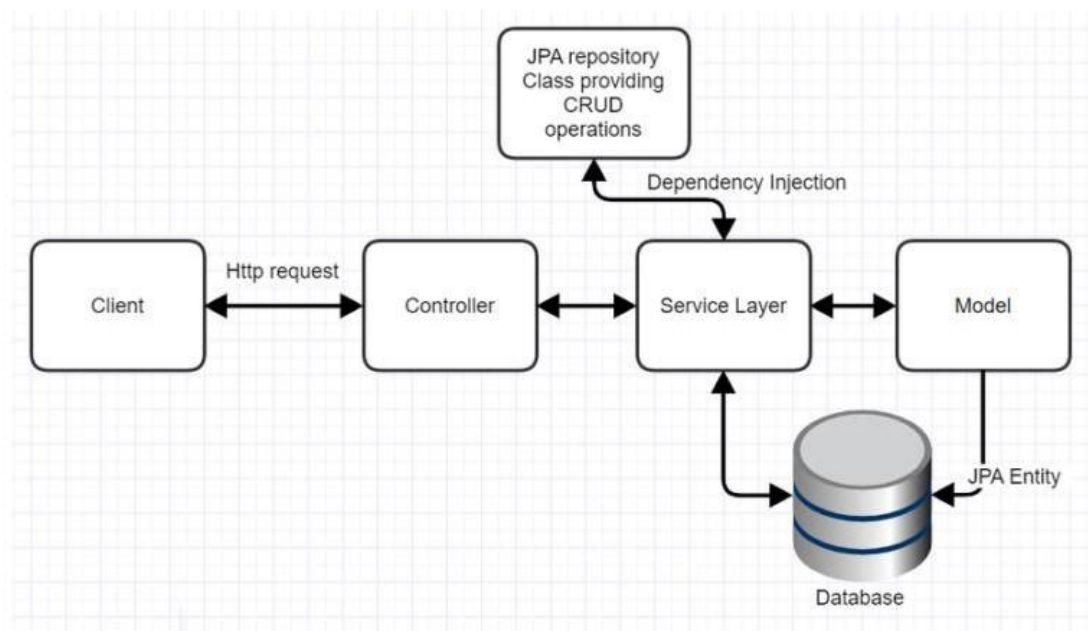
1. Creating **User** Entity: Create Spring Boot with Microservices Application with Spring Data JPA

### Technology stack:

- Spring-Boot
- Spring REST

#### 5.1. Steps for creating a project in Spring Boot:

- In Eclipse IDE, we have to create a new Maven Project using required Group Id , Artifact Id ,Packaging and version.
- In POM.xml we need dependencies such as Spring Data JPA, Spring Boot Devtools, MySQL Driver and Spring web to support Spring application.
- Later we have to configure application. properties to connect with MySQL database.
- By creating Model, Service, Repository, Controller Packages we can perform the required Business Logics.



**Fig A : Representation of Spring MVC pattern**

## 5.2 Steps to create entities to perform Business logics:

### 1. create AppUser Entity:

1. In IntelliJ IDEA We need to create an Entity:(customer).
2. By Creating a customer Repository interface and will make use of Spring Data JPA.
  - a. Will have a query to validate user.
  - b. Add the User details by extending JPA Repository.
3. By Creating a UserServiceImpl class we can perform the required Business logics and exposes all Services.
4. Finally, by creating a UserController class will handle all http requests and also will have following URL's

URL	Methods	Description	Format
localhost:8084/customer	POST	Give a single user description searched based on username	JSON
localhost:8084/customers/search/{keyword}	GET	Give a keyword	JSON
localhost:8084/customer/{email}	DELETE	Give a email of customer	JSON

### 2. Create Cart Entity:

Creating **Cart** Entity:

Build a RESTful resource for **Cart** manipulations, where CRUD operations to be carried out. Here will have multiple layers into the application:

1. Create an Entity: category
2. Create a CategoryRepository interface and will make use of Spring Data JPA
  - a. addcart method used to add food item type.
  - b. getCart method is used to get category types.
  - c. updateCart method used to update food item type.
3. Create a cartServiceImpl class and will expose all these services.
4. Finally, create a cartController will have the following Uri's:

URL	Methods	Description	Format
localhost:8084/carts	GET	Getting the details of product in cart	JSON
localhost:8084/carts/add/{id}	PUT	Add product to cart by product id	JSON
localhost: 8084/carts/minus/{id}	PUT	Remove product from cart	JSON
localhost: 8084/carts/{id}	DELETE	Delete product present in cart by id	URL

### 3. Create Product Entity:

Creating **Product** Entity:

Build a RESTful resource for **Product** manipulations, where CRUD operations to be carried out. Here will have multiple layers into the application:

2. Create an Entity: Product
3. Create a ProductRepository interface and will make use of Spring Data JPA
  - b. UpdateProductStatus method used to update status.
  - b. getProductsByCategory used to select the category
  - c. getProductById method used to getting food item.
  - d. getProductCountByName method is used to get name of food item
4. Create a ProductServiceImpl class and will expose all these services.
5. Finally, create a ProductController will have the following Uri's:

URL	Methods	Description	Format
localhost:8084/products/Admin	GET	Getting admin products	JSON
localhost:8084/products	POST	Product price	JSON
localhost:8084/products/{id}	DELETE	Delete product by id	URL
localhost:8084/products/chinese	GET	Get chinese food	URL

### 5. Create purchase Entity:

Creating **purchase** Entity:

Build a RESTful resource for **purchase** manipulations, where CRUD operations to be carried out. Here will have multiple layers into the application

1. Create an Entity: Product

2. Create a purchaseRepository interface and will make use of Spring

Data JPA .

a. Purchase method used to purchase the food .

b. getrecord used to get all details.

c. updatestatus method used to update ordered status.

3. Create a userpurchaseServiceImpl class and will expose all these services.

4. Finally, create a UserpurchaseController will have the following Uri's:

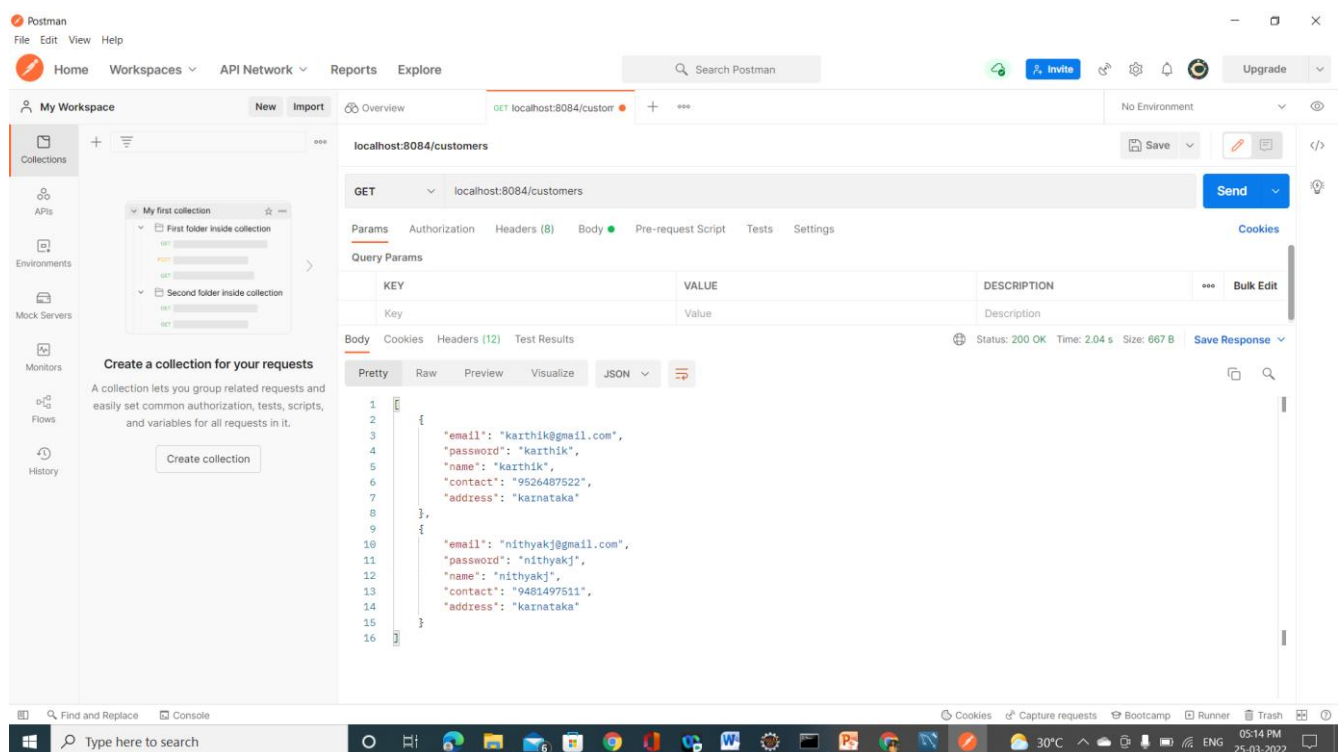
URL	Method s	Descriptio n	JSO N
localhost:8084/purchase/byEmail/{email}	GET	Get order purchase by email	URL
localhost:8084/ purchase	POST	Getting order placed details	URL

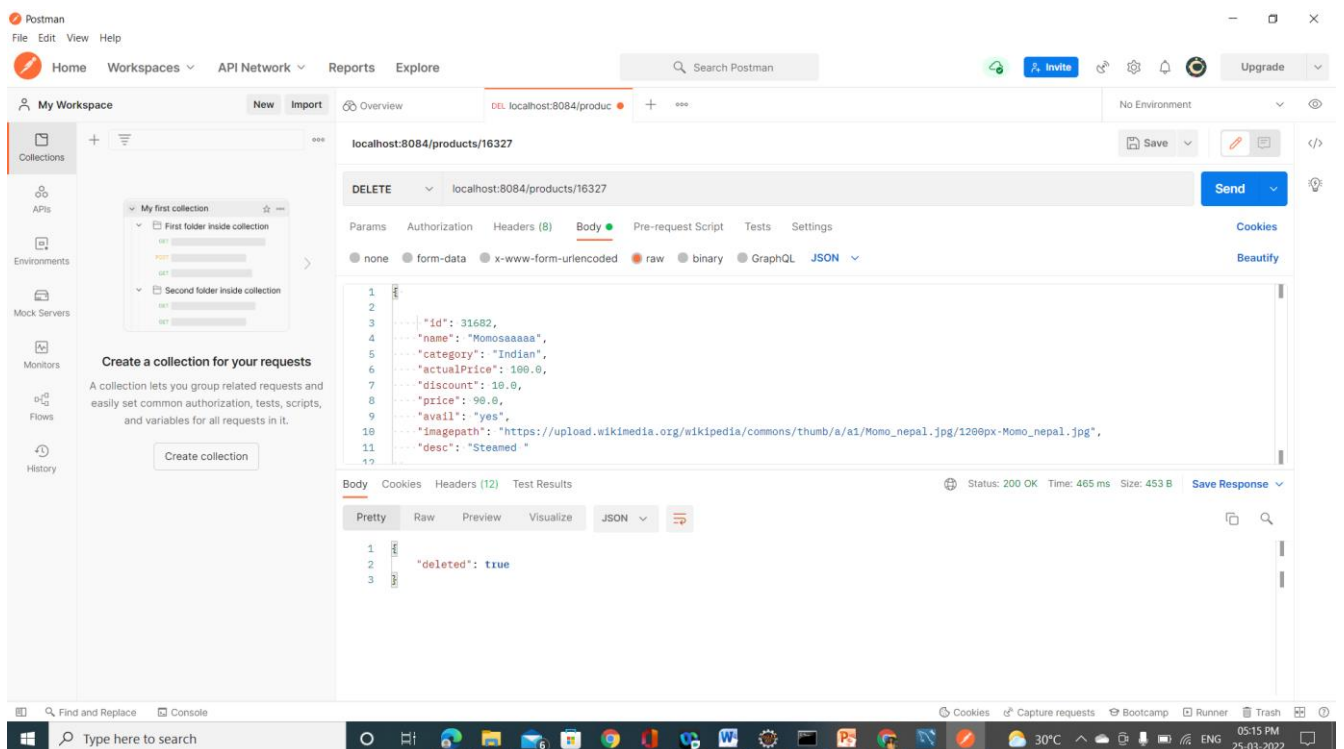
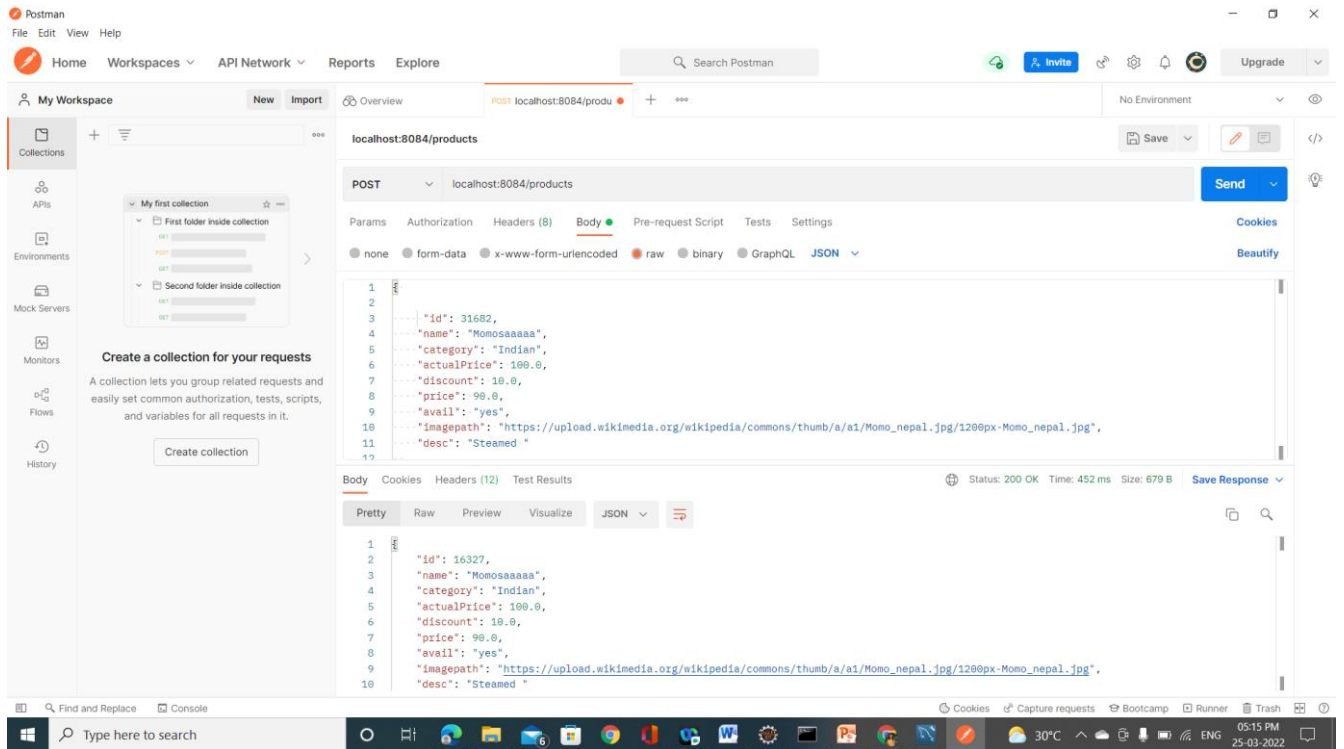


## 6. Implementation Details:

After deciding the technologies, We started to implement our system. First, we developed the backend for our system by creating a REST API for the server side application using java-based spring boot in IntelliJ IDEA. Then, we configured the backend API to connect to MySQL server @localhost:3306 to access the food ordering database created in the MySQL server which contains the tables for admin, cart, user, product and purchase. Then, we initially tested the API using postman by sending data in JSON format through HTTP GET, POST requests using the relevant URL endpoints.

## 7. Functional Postman Testing:





Postman  
File Edit View Help

Home Workspaces API Network Reports Explore

Search Postman

My Workspace New Import Overview POST localhost:8084/purchase

Save

POST localhost:8084/purchase

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1
2
3
4
5
6
7
8
9
10
{
  "id": 31682,
  "name": "MomoSaaaaa",
  "category": "Indian",
  "actualPrice": 100.0,
  "discount": 10.0,
  "price": 90.0,
  "createdAt": "2022-03-25T05:16:00.000Z"
}

```

Body Cookies Headers (12) Test Results

Status: 200 OK Time: 1233 ms Size: 453 B Save Response

Pretty Raw Preview Visualize JSON

```

1
2
3
{
  "created": true
}

```

Find and Replace Console

Type here to search

Windows Taskbar: 30°C, 05:16 PM, 25-03-2022

Postman  
File Edit View Help

Home Workspaces API Network Reports Explore

Search Postman

My Workspace New Import Overview GET localhost:8084/customers

Save

GET localhost:8084/customers

Params Authorization Headers (7) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL

This request does not have a body

Body Cookies (1) Headers (12) Test Results

Status: 200 OK Time: 601 ms Size: 667 B Save Response

Pretty Raw Preview Visualize JSON

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
{
  "email": "karthik@gmail.com",
  "password": "karthik",
  "name": "karthik",
  "contact": "9526487522",
  "address": "kainataka"
},
{
  "email": "nithyakj@gmail.com",
  "password": "nithyakj",
  "name": "nithyakj",
  "contact": "9481497511",
  "address": "kainataka"
}

```

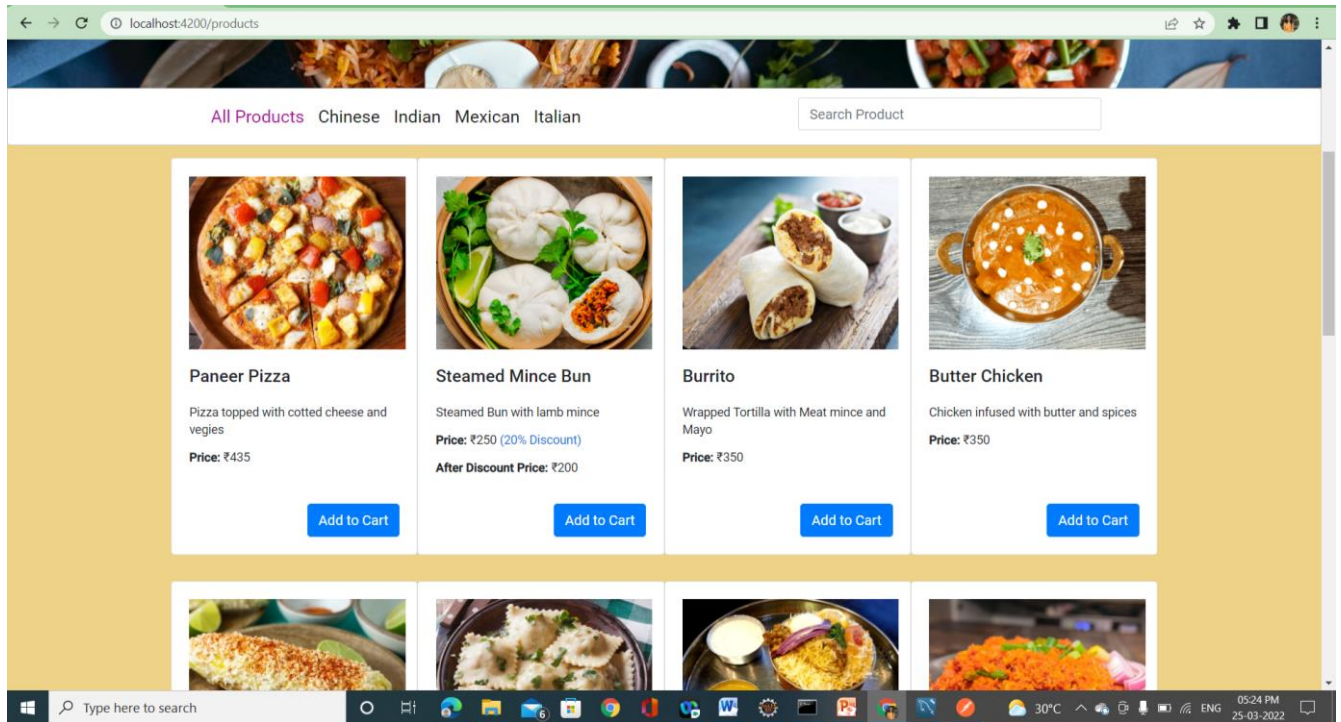
Find and Replace Console

Type here to search

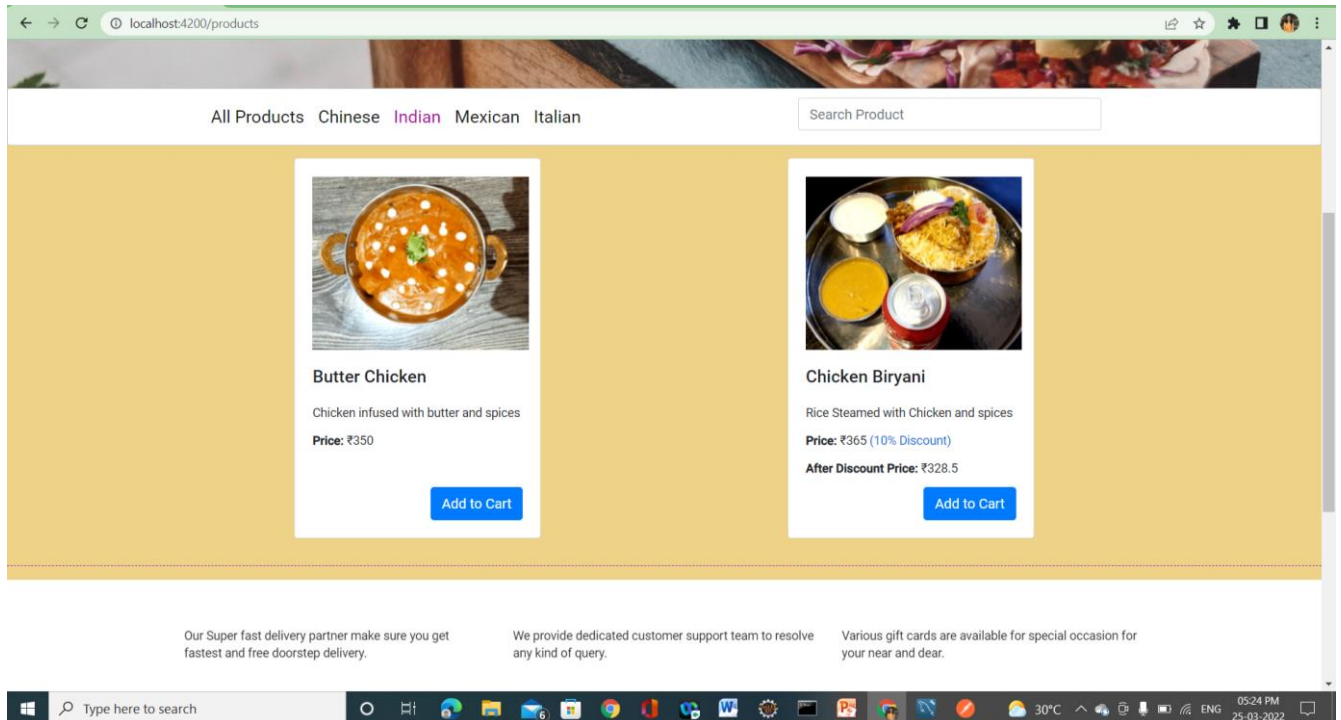
Windows Taskbar: 30°C, 05:20 PM, 25-03-2022

## Home Page:

### ALL PRODUCTS:



## Indian Product:





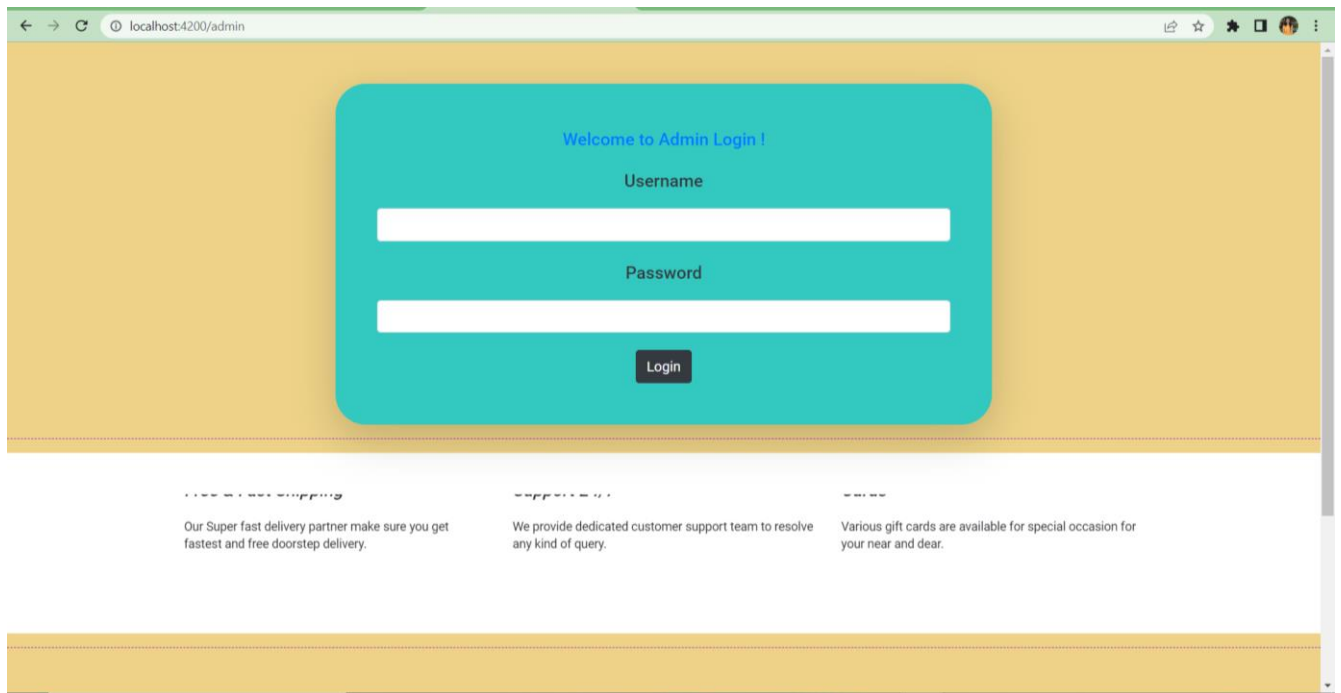
## Italian Product:

The screenshot shows a web application interface for Italian products. The browser address bar displays 'localhost:4200/products'. The navigation bar includes links for 'All Products', 'Chinese', 'Indian', 'Mexican', and 'Italian' (which is highlighted). A search bar labeled 'Search Product' is positioned to the right. The main content area features three product cards: 1. 'Paneer Pizza' with a description 'Pizza topped with cotted cheese and vegies' and a price of ₹435. 2. 'Ravioli' with a description 'Ravioli pasta filled with veg mince', a price of ₹200 (18% Discount), and an after-discount price of ₹164. 3. 'Red Sause Pasta' with a description 'Pasta with Tomato and oregano' and a price of ₹435. Each card includes an 'Add to Cart' button. Below the product cards, there are three promotional banners: 'Free & Fast Shipping' with the text 'Our Super fast delivery partner make sure you get', 'Support 24/7' with 'We provide dedicated customer support team to resolve', and 'Cards' with 'Various gift cards are available for special occasion for'. The Windows taskbar at the bottom shows the search bar, taskbar icons, system tray with a temperature of 30°C, and the date/time '05:24 PM 25-03-2022'.

## Mexican Product:

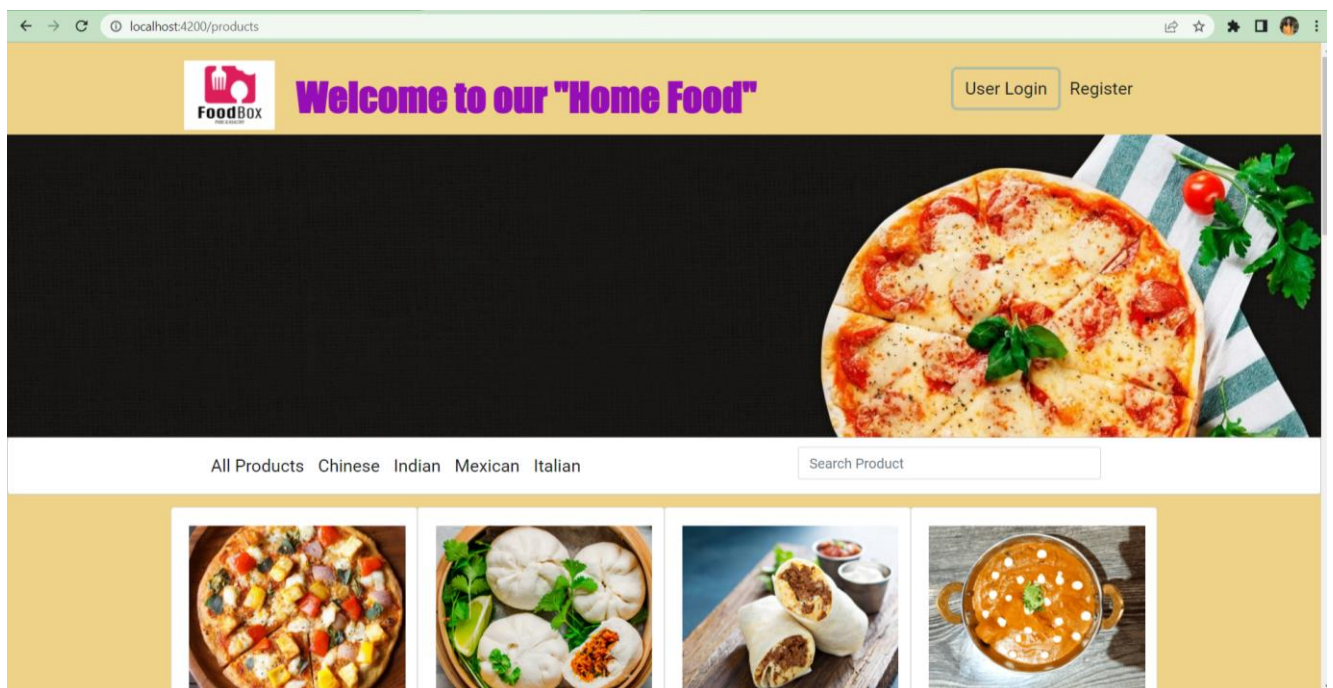
The screenshot shows a web application interface for Mexican products. The browser address bar displays 'localhost:4200/products'. The navigation bar includes links for 'All Products', 'Chinese', 'Indian', 'Mexican' (which is highlighted), and 'Italian'. A search bar labeled 'Search Product' is positioned to the right. The main content area features two product cards: 1. 'Burrito' with a description 'Wrapped Tortilla with Meat mince and Mayo' and a price of ₹350. 2. 'Elote de Corn' with a description 'Corn topped with cream cheese and spice', a price of ₹180 (7% Discount), and an after-discount price of ₹167.4. Each card includes an 'Add to Cart' button. The Windows taskbar at the bottom shows the search bar, taskbar icons, system tray with a temperature of 30°C, and the date/time '05:24 PM 25-03-2022'.

## Admin Page:

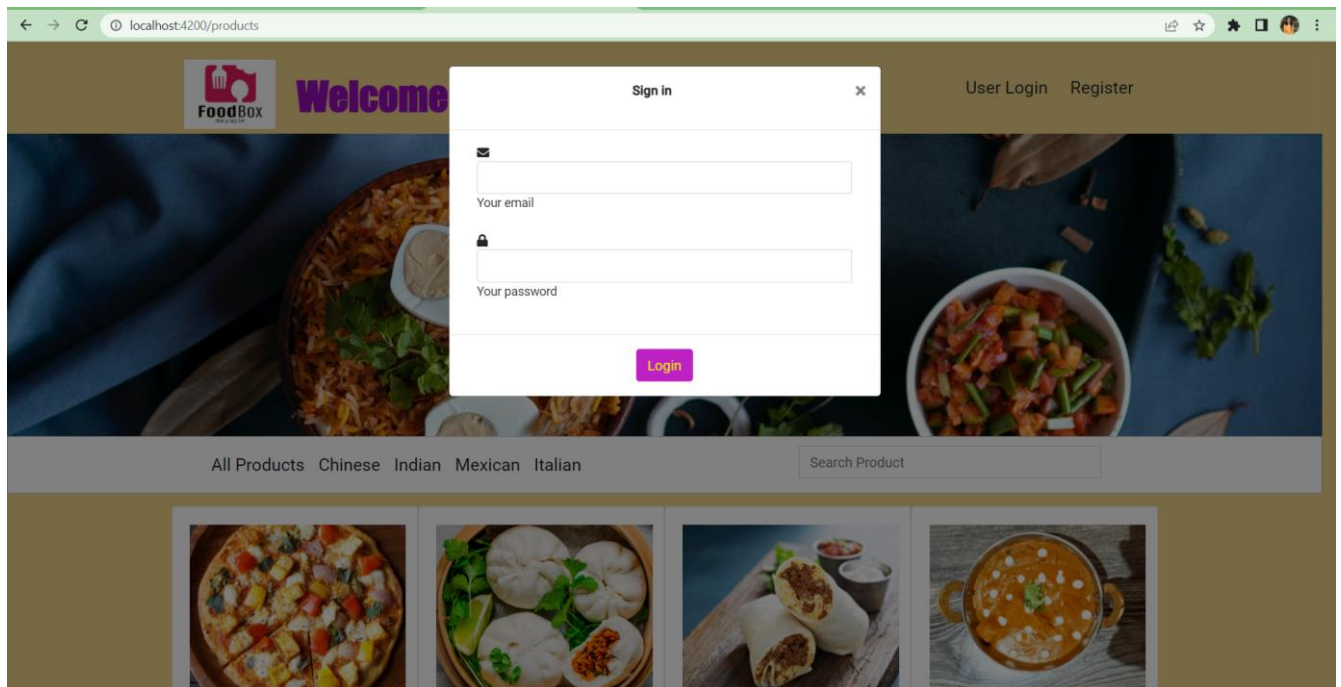


The screenshot shows a web browser window with the address bar displaying 'localhost:4200/admin'. The page has a yellow background. In the center, there is a teal-colored rounded rectangle containing the text 'Welcome to Admin Login !' in blue. Below this text are two white input fields: the first is labeled 'Username' and the second is labeled 'Password'. A dark grey 'Login' button is positioned below the password field. At the bottom of the page, there are three columns of text separated by dashed lines. The first column is titled 'Fast Delivery' and states: 'Our Super fast delivery partner make sure you get fastest and free doorstep delivery.' The second column is titled 'Customer Support' and states: 'We provide dedicated customer support team to resolve any kind of query.' The third column is titled 'Gift Cards' and states: 'Various gift cards are available for special occasion for your near and dear.'

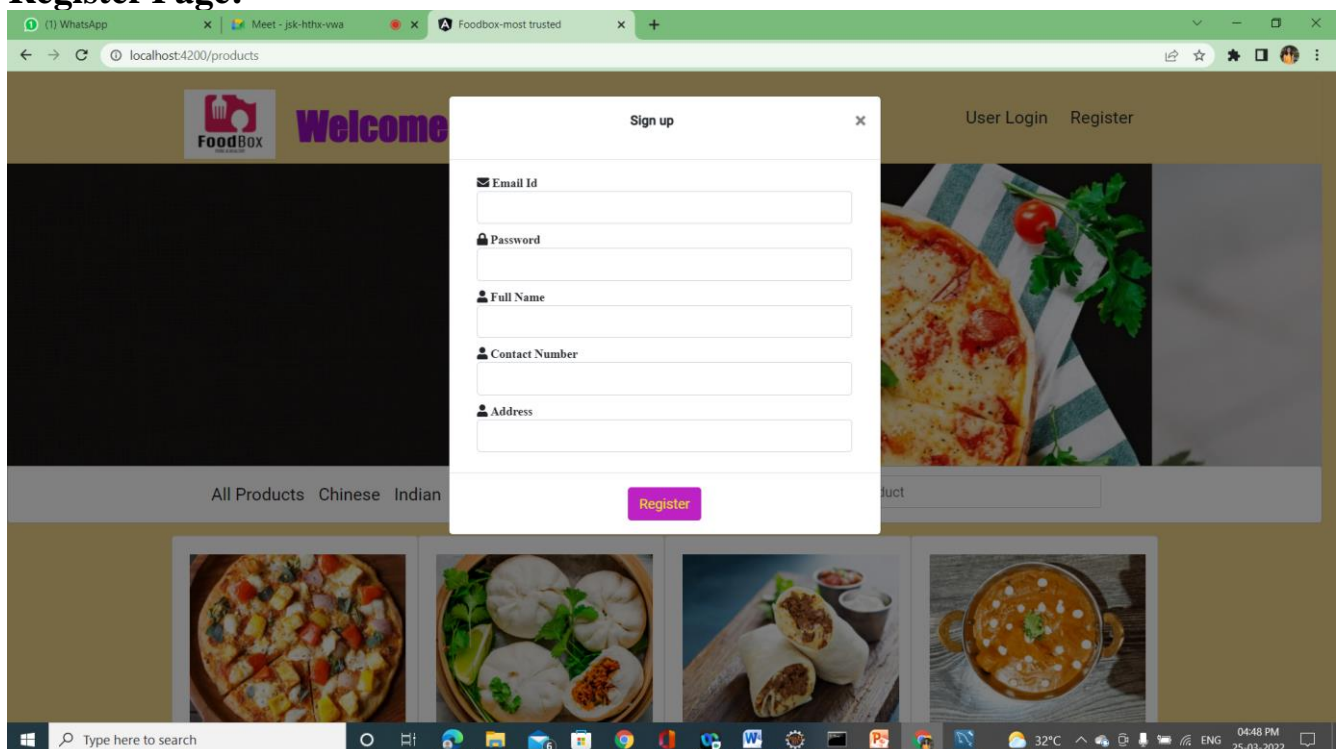
## Restaurant Page:



## User Page:



## Register Page:



## Adding Product By admin Page:

The screenshot shows the 'The Foodbox' Admin Dashboard. The main section is 'Product List' with a table of products. A 'Product Details' modal is open, allowing an admin to add or edit a product. The modal includes fields for Name, Description, Category, Price, Discount, Availability (Yes/No), and Image Link. The background table lists products like Paneer Pizza, Steamed Mince Bun, Burrito, etc., with 'Update' and 'Delete' buttons for each.

Name	Description
Paneer Pizza	Pizza topped with cotted cheese and vegies
Steamed Mince Bun	Steamed Bun with lamb mince
Burrito	Wrapped Tortilla with Meat mince and Mayo
Butter Chicken	Chicken infused with butter and spices
Elote de Corn	Corn topped with cream cheese and sp
Ravioli	Ravioli pasta filled with veg mince
Chicken Biryani	Rice Steamed with Chicken and spices
Egg Fried Rice	Rice with Egg and Chinese sauses
Red Sause Pasta	Pasta with Tomato and oregano

ImagePath	Actions
/assets/images/paneerpizza.jpg	<button>Update</button> <button>Delete</button>
/assets/images/buns.jpg	<button>Update</button> <button>Delete</button>
/assets/images/Burrito.jpg	<button>Update</button> <button>Delete</button>
/assets/images/ButterChicken.png	<button>Update</button> <button>Delete</button>
/assets/images/elote.jpg	<button>Update</button> <button>Delete</button>
/assets/images/ravioli.jpg	<button>Update</button> <button>Delete</button>
/assets/images/biryani.jpg	<button>Update</button> <button>Delete</button>
/assets/images/EggfriedRice.jpg	<button>Update</button> <button>Delete</button>
/assets/images/redPasta.jpg	<button>Update</button> <button>Delete</button>

## Admin Page after Login:

The screenshot shows the 'The Foodbox' Admin Dashboard after login. The main section is 'Users List' with a table of transactions. Below the table is a 'Contact us' section with contact information.

Transaction Id	Date of Purchase	Product Name	Quantity	Total Cost	Customer Email	Action
TFIB629561791225		Paneer Pizza	1	435	karthik@gmail.com	<button>Delete</button>
TFIB815920340301		Momos	1	90	nithyakj@gmail.com	<button>Delete</button>

**Contact us**

Phone: +91 8805364132

Email: FoodBox@gmail.com

Mon-Fri: 9.00 am - 8.00 pm



## **8. Conclusion:**

- With online ordering on board you will enrichen your customer experience by making the process of ‘placing orders’ a lot easier. It will show that you value your customer’s time.
- Online ordering will guarantee a ‘level up’ to your web presence. And a good web presence will make you stand out in the search engine rankings and bring more customers to you.
- Online ordering will boost your productivity by eliminating the inefficient process of taking orders. It will help you to plan and implement an adaptive marketing campaign.
- Utilising the latest online ordering technology for your restaurant will also help you to tap into a massive customer base which is tech-savvy and believes in ‘online way’.







