Mini project report on

ZapFlavor: An Online Food Ordering System

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

Bachelor of Technology

in

Computer Science & Engineering

UE22CS351A – DBMS Project

Submitted by:

Mrunal R Anandache	**PES2UG22CS323**
N S Tushar	**PES2UG22CS327**

Under the guidance of
Dr. Saranya Rubini
Associate Professor
PES University

AUG - DEC 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

CERTIFICATE

This is to certify that the mini project entitled

ZapFlavor : An Online Food Ordering System

is a bonafide work carried out by

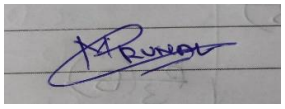
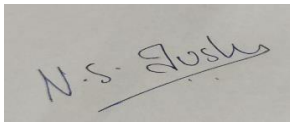
Mrunal R Anandache	**PES2UG22CS323**
N S Tushar	**PES2UG22CS327**

In partial fulfilment for the completion of fifth semester DBMS Project (UE22CS351A) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period AUG. 2024 – DEC. 2024. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The project has been approved as it satisfies the 5th semester academic requirements in respect of project work.

Signature
Dr. Saranya Rubini
Associate Professor

DECLARATION

We hereby declare that the DBMS Project entitled **ZapFlavor : An Online Food Ordering System** has been carried out by us under the guidance of **Dr. Saranya Rubini, Associate Professor** and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester AUG – DEC 2024.

Mrunal R Anandache	**PES2UG22CS323**	
N S Tushar	**PES2UG22CS327**	

GitHub Link for our project : GitHub Link

ABSTRACT

Objective

The *ZapFlavor* project aims to revolutionize the online food ordering experience by creating a modern, user-friendly platform tailored to meet the diverse needs of food lovers and restaurants alike. By employing a full-stack approach with technologies such as Node.js, Express, SQL, and EJS, the system ensures seamless order management, real-time updates, and a personalized journey for all users. The platform prioritizes dynamic interaction, secure data management, and visually appealing design, fostering convenience and engagement in every aspect of the food ordering process.

Scope

The platform serves both customers and restaurant administrators with role-specific functionalities. Customers can browse restaurants by cuisine, customize their orders, and view a detailed order summary before placing an order. Restaurants benefit from streamlined menu management, order tracking, and admin privileges for content updates. *ZapFlavor* integrates a robust SQL database for efficient data handling, ensuring secure storage of order details, customer preferences, and restaurant information.

Key Features

1. **Interactive Order Management:** Customers can easily add or delete items from the cart and view an updated order summary before proceeding to checkout.
2. **Cuisine-Based Navigation:** The homepage allows users to explore restaurants based on cuisine preferences, delivering a personalized browsing experience.
3. **Dynamic Delivery Assignment:** Once an order is placed, the system automatically assigns an available delivery partner and updates their status in the database for real-time tracking.
4. **Restaurant Admin Dashboard:** Verified restaurant administrators can edit menus, update restaurant information, and manage orders efficiently.
5. **Secure Data Storage and Authentication:** The SQL backend ensures reliable storage and secure handling of user and restaurant data, with scalable options for future enhancements.
6. **Responsive Design:** The platform is optimized for a seamless experience across devices, featuring visually engaging gradient backgrounds and intuitive layouts.

Conclusion

The *ZapFlavor* project offers a comprehensive solution to modern online food ordering needs by delivering real-time order updates, secure backend functionality, and tailored experiences for customers and restaurants. By bridging the gap between food lovers and service providers, *ZapFlavor* simplifies the ordering journey while maintaining flexibility and scalability. Future enhancements could include AI-driven menu recommendations, predictive delivery times, and integrated loyalty programs, ensuring continued user satisfaction and growth.

TABLE OF CONTENTS

1. INTRODUCTION

In today's fast-paced world, online food ordering has become a cornerstone of convenience and efficiency for both customers and restaurants. However, the increasing demand for diverse cuisines, coupled with the need for seamless order management, presents unique challenges. Traditional systems often fall short in providing personalized experiences, real-time updates, and streamlined operations, leading to inefficiencies in service delivery and customer satisfaction. Many existing platforms lack dynamic features such as customizable user interactions, responsive designs, and robust data management, which are essential for catering to modern user expectations.

In response to these challenges, *ZapFlavor: An Online Food Ordering System* offers a comprehensive solution that bridges the gap between customers and restaurants through a feature-rich, user-centric platform. The primary objective of this project is to enhance the online food ordering experience by providing a seamless interface for customers to browse cuisines, place and track orders, and manage preferences, while enabling restaurants to efficiently handle menus, orders, and delivery processes. The system is designed to deliver an intuitive, scalable, and secure digital framework that ensures convenience and satisfaction for all stakeholders.

Built using Node.js and Express for server-side logic, SQL for robust data management, and EJS for dynamic front-end rendering, this project integrates cutting-edge technologies to address the diverse needs of users. Key features include interactive cart management, cuisine-based navigation, dynamic delivery partner assignment, and responsive design for a consistent experience across devices. By combining a visually appealing user interface with secure data handling and role-specific functionalities, *ZapFlavor* aims to set a new standard in the online food ordering domain.

2. PROBLEM DEFINITION WITH USER REQUIREMENT SPECIFICATIONS

2.1. Problem Definition

Traditional food ordering systems face numerous challenges that create inefficiencies for both customers and restaurants, often leading to dissatisfaction and missed opportunities. Customers frequently encounter fragmented platforms, forcing them to browse multiple websites or apps to find their preferred restaurants or cuisines. These disjointed experiences make it difficult to discover relevant options, customize orders, or track deliveries effectively. The lack of a cohesive interface often results in frustration, incorrect orders, and delays in receiving updates, diminishing the overall experience.

For restaurants, traditional systems introduce operational bottlenecks that hinder efficient service delivery. Managing menus, tracking orders, and coordinating with delivery partners are labor-intensive tasks that consume valuable time and resources. These inefficiencies are compounded by outdated systems that lack real-time updates, dynamic features, or tools to optimize workflow. Restaurants often struggle to maintain consistency in handling high volumes of orders, which can lead to errors, miscommunication, and delays in fulfillment.

Additionally, the process of assigning delivery partners is often manual and prone to errors. Without automated systems to handle delivery logistics, restaurants face challenges in ensuring timely and accurate order delivery. This results in customer dissatisfaction, negative reviews, and a decline in repeat business.

These limitations affect both customers and restaurants, creating a gap in the online food ordering process that undermines the potential for convenience and efficiency. In a world where digital solutions are expected to be fast, user-friendly, and reliable, such outdated practices fail to meet modern expectations.

To address these challenges, there is a clear need for a comprehensive, user-centric platform that streamlines the online food ordering process. This solution must eliminate inefficiencies by offering features such as real-time updates, automated delivery assignment, and dynamic order management. By bridging the gap between customers and restaurants, the platform should enhance convenience, accuracy, and transparency, ensuring a satisfying experience for all stakeholders.

2.2. User Requirement Specifications

2.2.1. User Management

❖ **Requirement 1.1**: The system shall allow users to register as either customers or restaurant administrators, requiring basic details such as username, password, and email address.

❖ **Requirement 1.2**: Users shall be able to log in and log out securely with role-specific authentication (customer or administrator).

2.2.2. Restaurant and Menu Management

❖ **Requirement 2.1**: Restaurant administrators shall be able to create, edit, and delete menu items with details such as dish name, description, price, and availability.

❖ **Requirement 2.2**: Administrators shall have tools to update restaurant details, such as location, contact information, and operating hours.

2.2.3. Order Management

❖ **Requirement 3.1**: Customers shall be able to browse menus, customize orders, and add items to a virtual cart.

❖ **Requirement 3.2**: The system shall provide a detailed order summary, allowing customers to review and modify their selections before placing an order.

❖ **Requirement 3.3**: Administrators and delivery partners shall receive real-time notifications for new orders placed.

2.2.4. Delivery Assignment and Tracking

❖ **Requirement 4.1**: The system shall automatically assign available delivery partners to new orders and update their status to "on the way" in the database.

❖ **Requirement 4.2**: Delivery partners shall have access to order details and customer contact information for seamless delivery.

2.2.5. Advanced Search and Filtering

❖ **Requirement 5.1**: Customers shall have access to a search bar with filters such as cuisine type

❖ **Requirement 5.2**: The platform shall support advanced search for restaurant administrators to view specific order details or customer trends.

2.2.6. Security and Privacy

❖ **Requirement 6.1**: The platform shall use passport-based authentication to secure user sessions and data.

❖ **Requirement 6.2**: User passwords shall be encrypted using bcrypt module before being stored in the database.

2.2.7. Responsive Design and Accessibility

❖ **Requirement 7.1**: The platform shall be fully responsive, ensuring a seamless experience across desktops, tablets, and mobile devices.

❖ **Requirement 7.2**: The user interface shall adhere to accessibility standards to accommodate users with disabilities.

2.2.8. Performance and Scalability

❖ **Requirement 8.1**: Page loading times shall not exceed 3 seconds under normal traffic conditions.

❖ **Requirement 8.2**: The system shall handle concurrent activity for up to 1,000 active users without significant performance degradation.

2.2.9. Future Scope

❖ **Requirement 9.1**: The system shall be designed to support future enhancements, such as AI-based menu recommendations.

❖ **Requirement 9.2**: Integration with third-party APIs (e.g., food delivery platforms or payment gateways) shall be considered for expanded functionality.

By addressing these requirements, *ZapFlavor: An Online Food Ordering System* aims to enhance the food ordering experience, improve operational efficiency for restaurants, and provide a seamless, user-friendly platform for all stakeholders

3. LIST OF SOFTWARES/TOOLS/PROGRAMMING LANGUAGES USED

To develop a comprehensive and efficient *ZapFlavor: An Online Food Ordering System*, the following technologies were utilized:

3.1. Frontend Technologies:

❖ **EJS (Embedded JavaScript)**: A templating engine used to render dynamic web pages on the server side. EJS allows seamless integration of dynamic content into HTML, enhancing the user experience.

❖ **Bootstrap**: A CSS and JS framework used to create a responsive, mobile-first design. Bootstrap ensures seamless layout adjustments across various devices, providing an optimal viewing experience.

3.2. Backend Technologies:

❖ **Node.js**: A JavaScript runtime enabling server-side scripting, allowing the use of JavaScript on both the client and server sides. Node.js ensures high performance and scalability for handling user requests.

❖ **Express.js**: A robust web application framework for Node.js, used for building RESTful APIs and managing HTTP requests efficiently.

3.3. Database:

❖ **MySQL**: Used for managing structured data related to menu items, orders, user profiles, and delivery assignments. MySQL ensures data consistency, supports efficient querying, and handles relationships for seamless functionality.

3.4. Additional Tools and Libraries:

❖ **Git/GitHub**: For version control and collaboration, enabling team members to work on the project seamlessly and track changes effectively.

❖ **Visual Studio Code (VS Code)**: A versatile code editor supporting a wide range of extensions, making it ideal for full-stack development.

❖ **Online Gantt Chart**: Used for project scheduling and timeline management, aiding in planning and tracking progress across development phases.

❖ **DrawSQL**: A schema design tool utilized to model and visualize the database structure, ensuring optimal data organization.

❖ **Draw.io**: Used for creating the ER diagram to represent entities and relationships within the system's database.

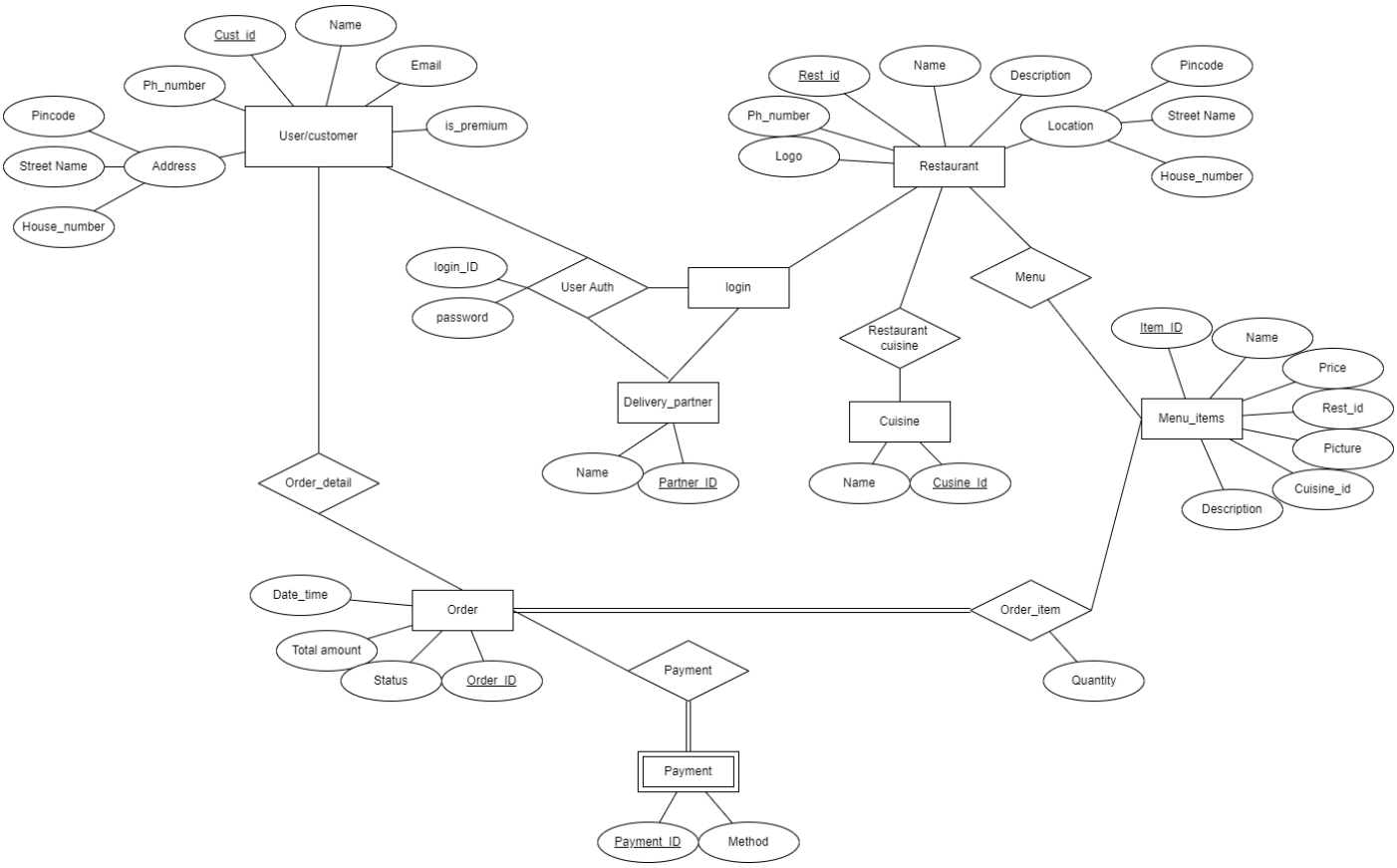
❖ **Hopscotch**: Utilized for testing API endpoints, ensuring reliability and accurate communication between the frontend and backend.

❖ **Chrome Browser**: Employed for testing and debugging the website to ensure compatibility and seamless performance across devices.

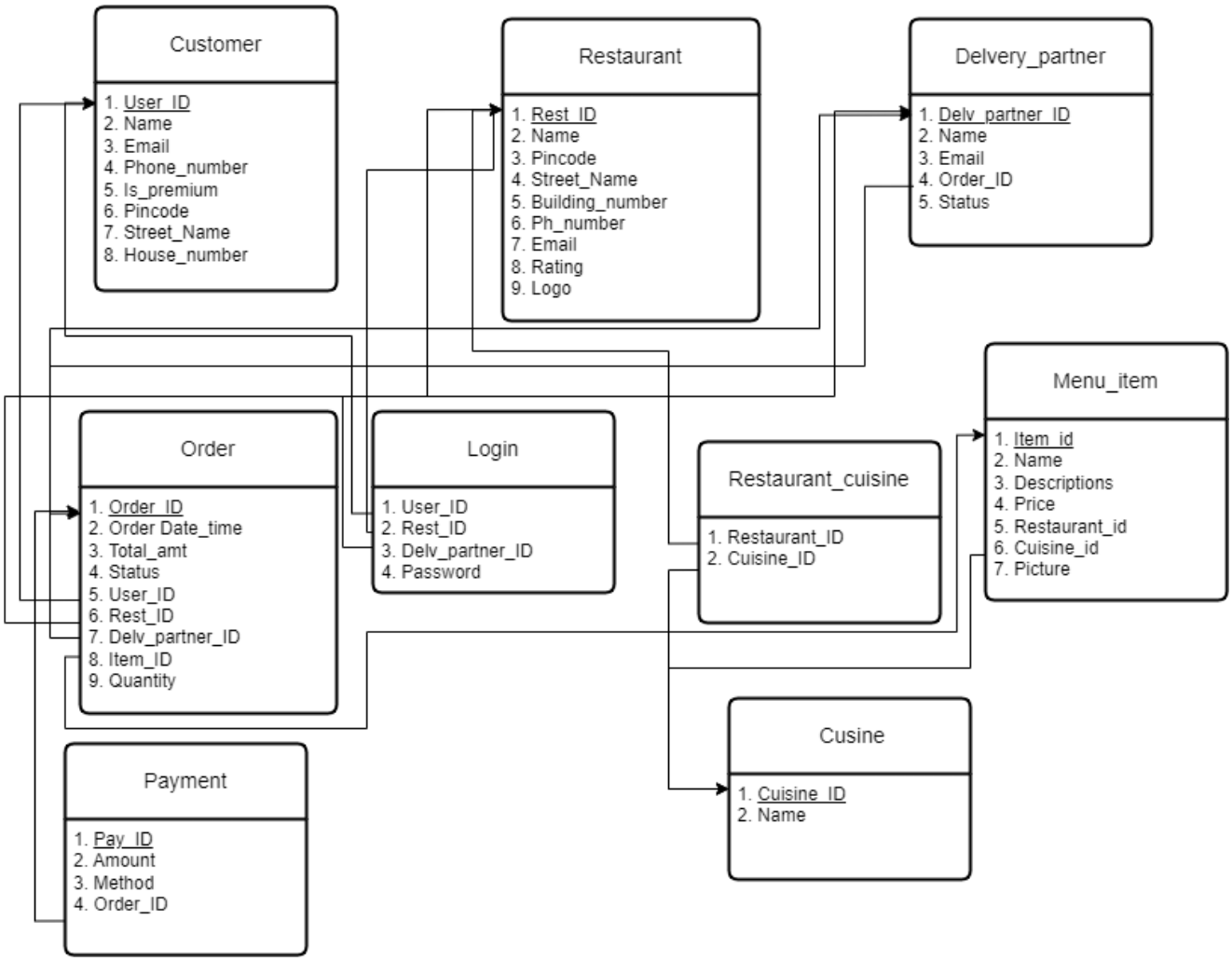
This comprehensive combination of tools, frameworks, and programming languages enabled the development of a robust, user-friendly, and scalable platform that enhances the food ordering experience for customers and streamlines operations for restaurants.

4. ENTITY-RELATIONSHIP (ER) MODEL

-



5. ER TO RELATIONAL MAPPING



6. DDL STATEMENTS

-Table for Restaurants

```
CREATE TABLE Restaurants (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description VARCHAR(200),
    image VARCHAR(1000) DEFAULT 'https://images.unsplash.com/photo-
1517248135467-4c7edcad34c4?q=80&w=2070&auto=format&fit=crop&ixlib=rb-
4.0.3&ixid=M3wxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D
',
    location VARCHAR(255) NOT NULL,
    contact_no VARCHAR(20),
    cuisine_id INT,
    FOREIGN KEY (cuisine_id) REFERENCES Cuisine(cuisine_id)
);
```

-Table for users

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password VARCHAR(255) NOT NULL
);
```

-Table for Orders

```
CREATE TABLE Orders (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id BIGINT,
    restaurant_id INT NOT NULL,
    menu_item_id INT NOT NULL,
    quantity INT NOT NULL,
    total_price DECIMAL(10,2) NOT NULL,
    order_id BIGINT,
    status ENUM('in_cart', 'ordered', 'deleted', 'received') DEFAULT 'in_cart',
    FOREIGN KEY (restaurant_id) REFERENCES Restaurants(id),
    FOREIGN KEY (menu_item_id) REFERENCES MenuItems(id),
    FOREIGN KEY (order_id) REFERENCES OrderSummary(id)
```

```
);
```

-Table for Menu

```
CREATE TABLE Menu (
    id INT AUTO_INCREMENT PRIMARY KEY,
    restaurant_id INT NOT NULL,
    name VARCHAR(255) NOT NULL,
    description VARCHAR(300),
    image VARCHAR(1000),
    price DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (restaurant_id) REFERENCES Restaurants(id)
);
```

-Table for Delivery Partners

```
CREATE TABLE Delivery_Part (
    delv_part_id INT PRIMARY KEY,
    name CHAR(20) NOT NULL,
    email VARCHAR(50),
    ph_num BIGINT,
    order_id BIGINT,
    status ENUM('available', 'received', 'delivered', 'on the way') NOT NULL,
    user_id BIGINT,
    FOREIGN KEY (order_id) REFERENCES Orders(order_id) -- Assuming Orders
table exists
);
```

-Table for Cuisines

```
CREATE TABLE cuisines (
    cuisine_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL
);
```

7. DML STATEMENTS – CRUD OPERATIONS

7.1 Registration of a new user:

```
INSERT INTO users (name, email, password) VALUES (?, ?, ?)
```

7.2 Fetching all restaurants:

```
SELECT * FROM restaurants
```

7.3 Fetching restaurants by cuisine:

```
SELECT restaurants.*, cuisines.name AS cuisine_name
FROM restaurants
INNER JOIN cuisines ON restaurants.cuisine_id = cuisines.cuisine_id
WHERE restaurants.cuisine_id = ?
```

7.4 Inserting a new restaurant:

```
SELECT add_new_restaurant(?, ?, ?, ?) AS message
```

7.5 Updating a restaurant's details:

```
CALL update_restaurant(?, ?, ?, ?, ?)
```

7.6 Deleting a restaurant:

```
DELETE FROM restaurants WHERE id = ?
```

7.7 Fetching the menu of a restaurant:

```
SELECT * FROM menu WHERE restaurant_id = ?
```

7.8 Adding a menu item:

```
INSERT INTO menu (restaurant_id, name, description, price) VALUES (?, ?, ?, ?)
```

7.9 : Fetching a specific menu item:

SELECT * FROM menu WHERE id = ?

7.10: Updating a menu item:

UPDATE menu SET name = ?, description = ?, price = ? WHERE id = ?

7.11: Deleting a menu item:

DELETE FROM menu WHERE id = ?

7.12: Fetching orders that are in the cart:

SELECT * FROM orders WHERE user_id = ? AND status = 'in_cart'

7.13: Inserting a new order:

INSERT INTO orders (user_id, restaurant_id, menu_item_id, quantity, total_price, order_id, status)
VALUES (?, ?, ?, ?, ?, ?, 'in_cart')

7.14: Deleting an item from the cart:

UPDATE orders SET status = 'deleted' WHERE id = ? AND status = 'in_cart'

7.15: Fetching a random delivery partner who is available:

SELECT * FROM Delivery_part WHERE status = 'available' ORDER BY RAND() LIMIT 1

7.16: Assigning a delivery partner to an order:

UPDATE Delivery_part
SET order_id = ?, user_id = ?, status = 'on the way'
WHERE delv_part_id = ?

7.17: Updating the status of a delivery partner to 'available':

UPDATE Delivery_Part SET status = 'available' WHERE delv_part_id = ?

8. QUERIES (JOIN QUERY)

```
app.get("/home/:id", checkAuthenticated, (req, res) => {
  let { id } = req.params;
  const sel_cuisine = `
    SELECT restaurants.*, cuisines.name AS cuisine_name
    FROM restaurants
    INNER JOIN cuisines ON restaurants.cuisine_id = cuisines.cuisine_id
    WHERE restaurants.cuisine_id = ?;
  `;

  db.query(sel_cuisine, [id], (err, results) => {
    if (err) {
      console.log(err);
    } else {
      res.render('pages/index', { restaurants: results });
    }
  });
});
```

9. STORED PROCEDURE, FUNCTIONS AND TRIGGERS

STORED PROCEDURE:

```
137     -- ############### Procedure ####################
138     DELIMITER //
139
140 •   CREATE PROCEDURE update_restaurant(
141       IN restaurant_id INT,
142       IN restaurant_name VARCHAR(255),
143       IN restaurant_description VARCHAR(200),
144       IN restaurant_location VARCHAR(255),
145       IN restaurant_contact VARCHAR(20)
146     )
147     BEGIN
148       UPDATE restaurants
149       SET name = restaurant_name,
150           description = restaurant_description,
151           location = restaurant_location,
152           contact_no = restaurant_contact
153       WHERE id = restaurant_id;
154     END //
155
156     DELIMITER ;
```

STORED FUNCTION:

```
112     -- ############### Function ####################
113     DELIMITER //
114
115 •   CREATE FUNCTION add_new_restaurant(
116       restaurant_name VARCHAR(255),
117       restaurant_description VARCHAR(200),
118       restaurant_location VARCHAR(255),
119       restaurant_contact VARCHAR(20)
120     )
121     RETURNS VARCHAR(255)
122     DETERMINISTIC
123     BEGIN
124       DECLARE msg VARCHAR(255);
125
126       -- Insert into the restaurants table
127       INSERT INTO restaurants (name, description, location, contact_no)
128       VALUES (restaurant_name, restaurant_description, restaurant_location, restaurant_contact);
129
130       -- Return a success message
131       SET msg = CONCAT('New restaurant "', restaurant_name, '" added successfully.');
132       RETURN msg;
133     END //
134
135     DELIMITER ;
```

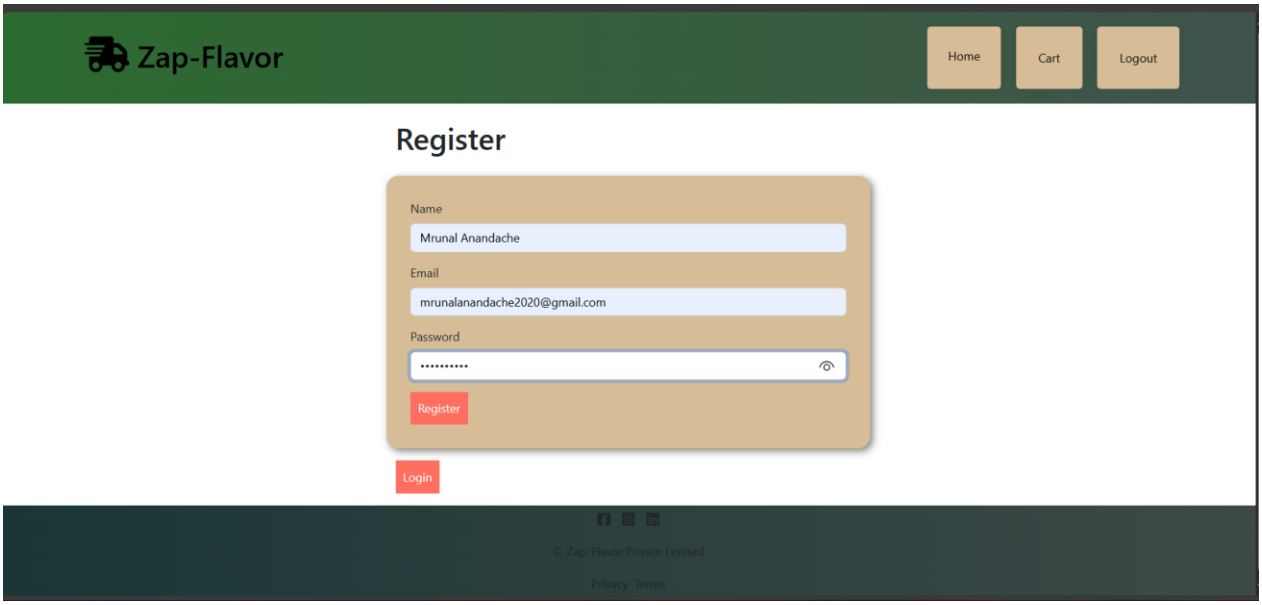
TRIGGERS:

```
 97     -- ############### TRIGGER ####################
 98     DELIMITER $$
 99
100 •   CREATE TRIGGER update_delivery_partner_status
101     BEFORE UPDATE ON Delivery_Part
102     FOR EACH ROW
103     BEGIN
104         -- Check if the new status is not 'available', and change it to 'available'
105         IF NEW.status != OLD.status THEN
106             SET NEW.status = 'available';
107         END IF;
108     END $$
109
110     DELIMITER ;
```

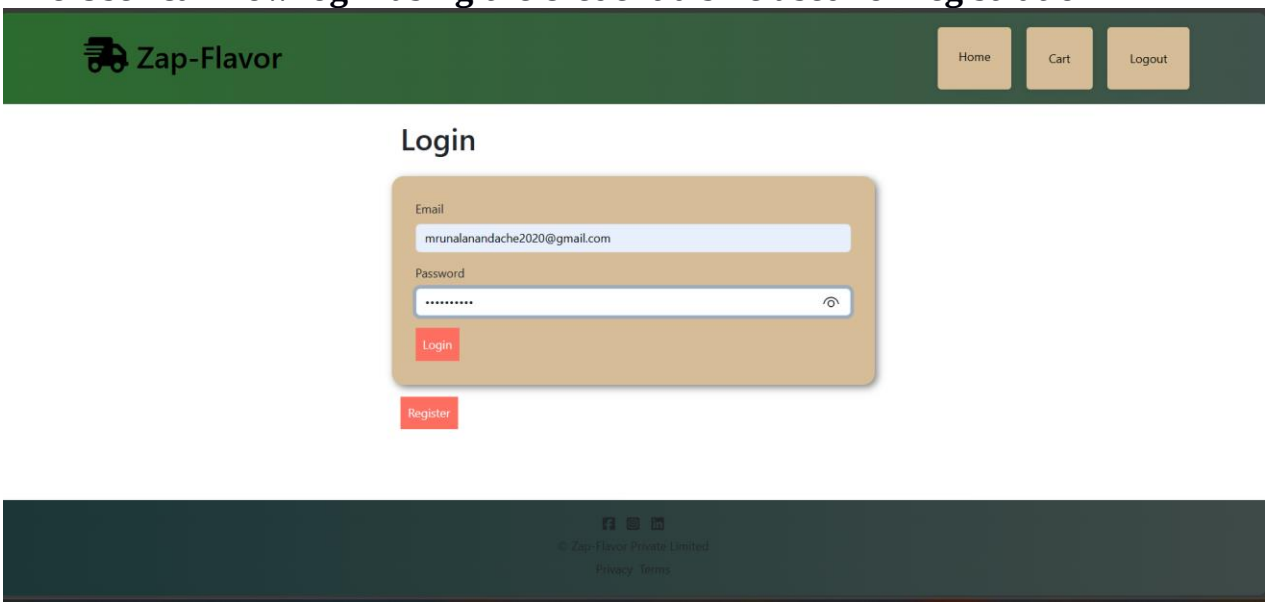
10. FRONT END DEVELOPMENT (FUNCTIONALITIES/FEATURES OF THE APPLICATION)

10.1 From Users Point Of View(FRONTEND):

10.1.1 : Registration Page:
The User can Register using his/her credentials

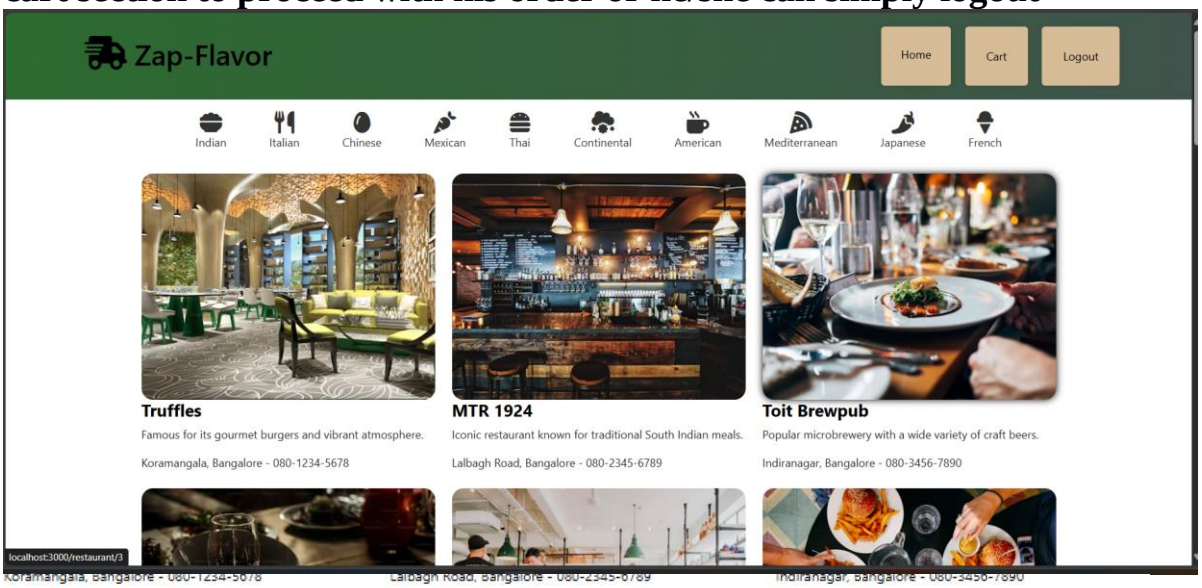


10.1.2 : Login Page:
The User can now login using the credentials he used for registration



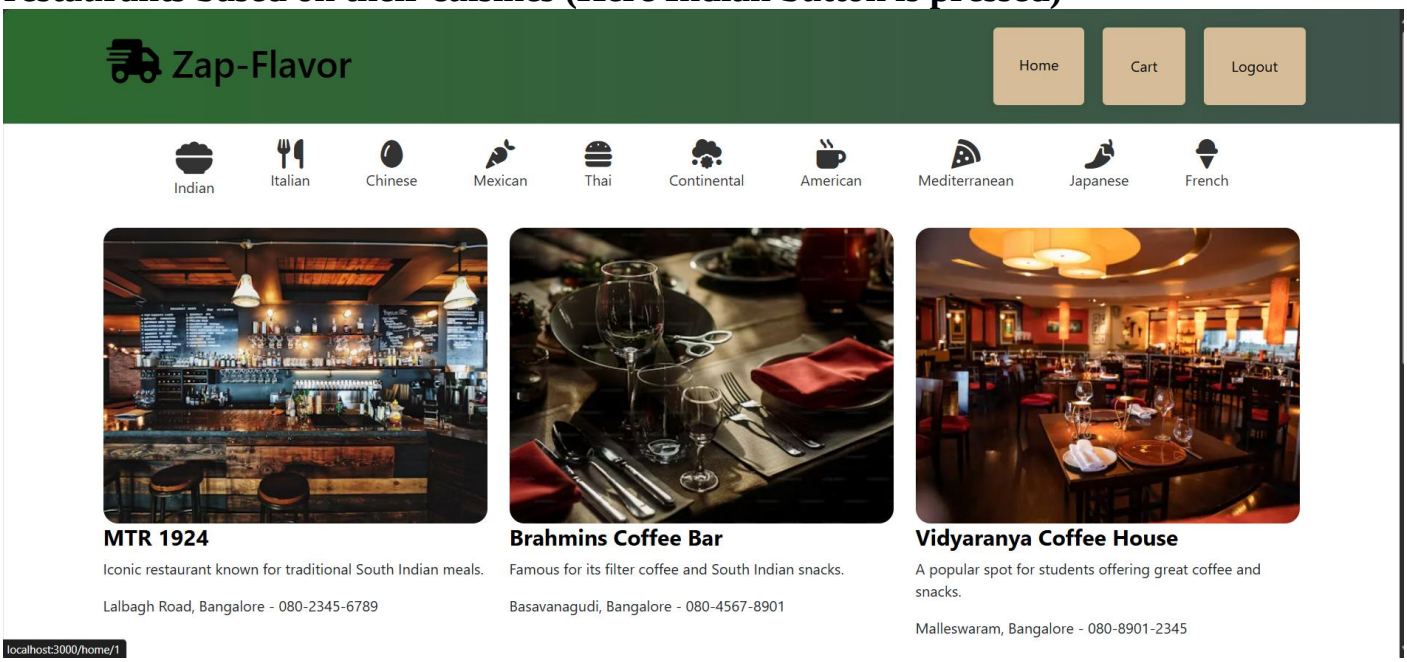
10.1.3 : Home Page:

Once Logged In the user can view the home page with all the listings of restaurants, the user can click on any restaurant and can go the page where he/she can view the menu items available , also on the navbar he can directly jump to the cart section to proceed with his order or he/she can simply logout



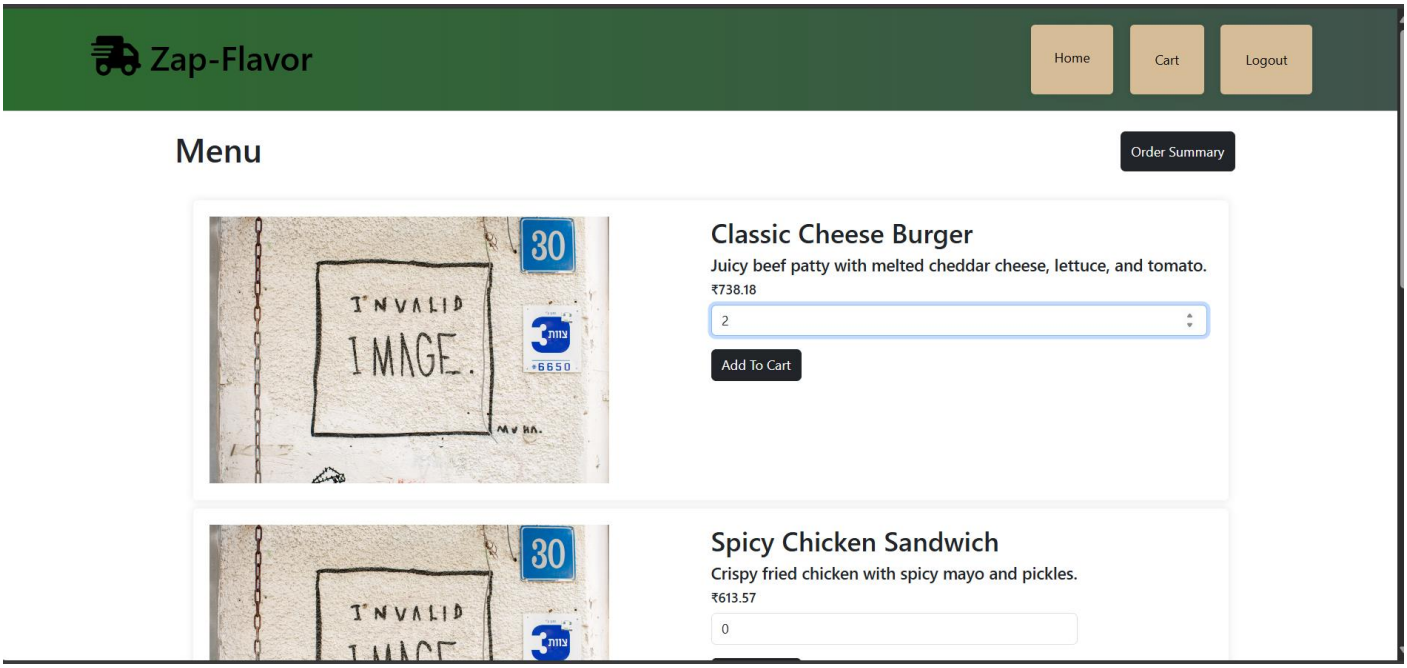
10.1.4 : Cuisines:

The User can choose the cuisines shown just below the navbar to filter out restaurants based on their cuisines (Here Indian button is pressed)



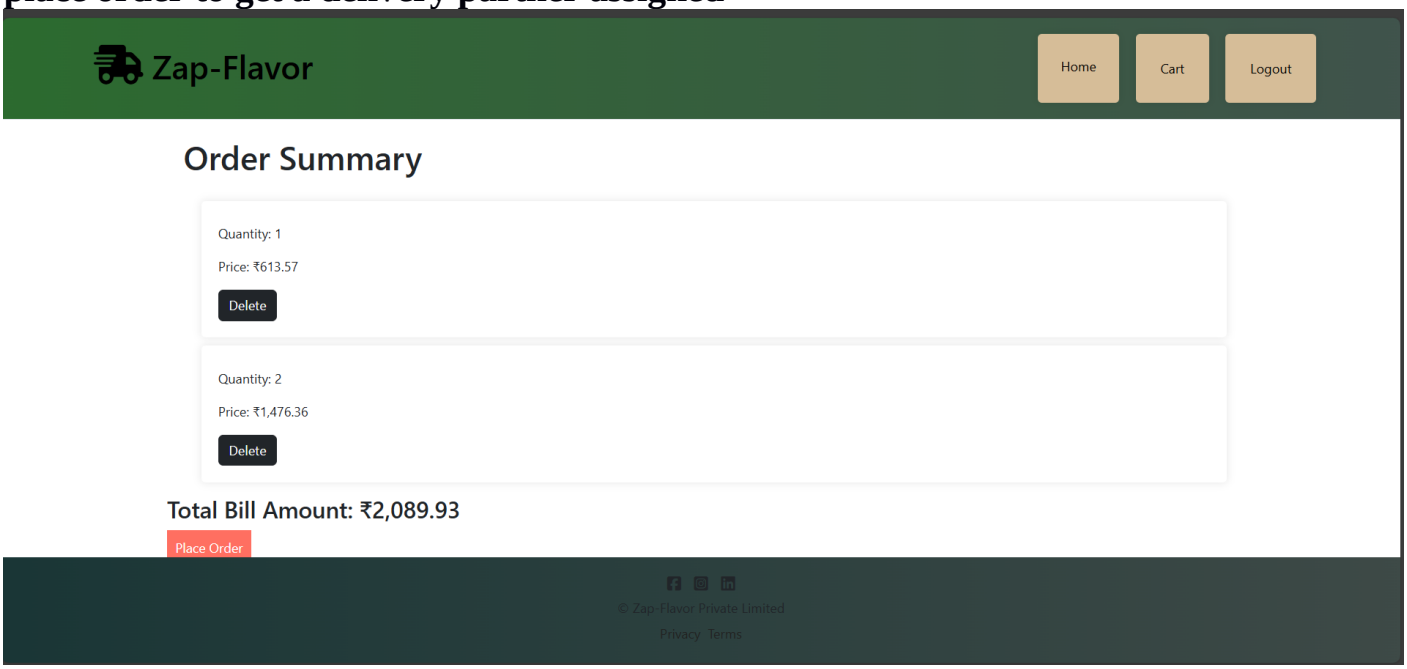
10.1.5 : Menu Page:

On clicking on any of the Restaurants images , the user will be redirected to the menu page from where he/she can proceed with choosing the food items and then pushing them to cart



10.1.6 : Order Summary Page:

After adding the food items to the cart , the user can jump to the order summary section by using the button from where he can view all his cart item and then press place order to get a delivery partner assigned



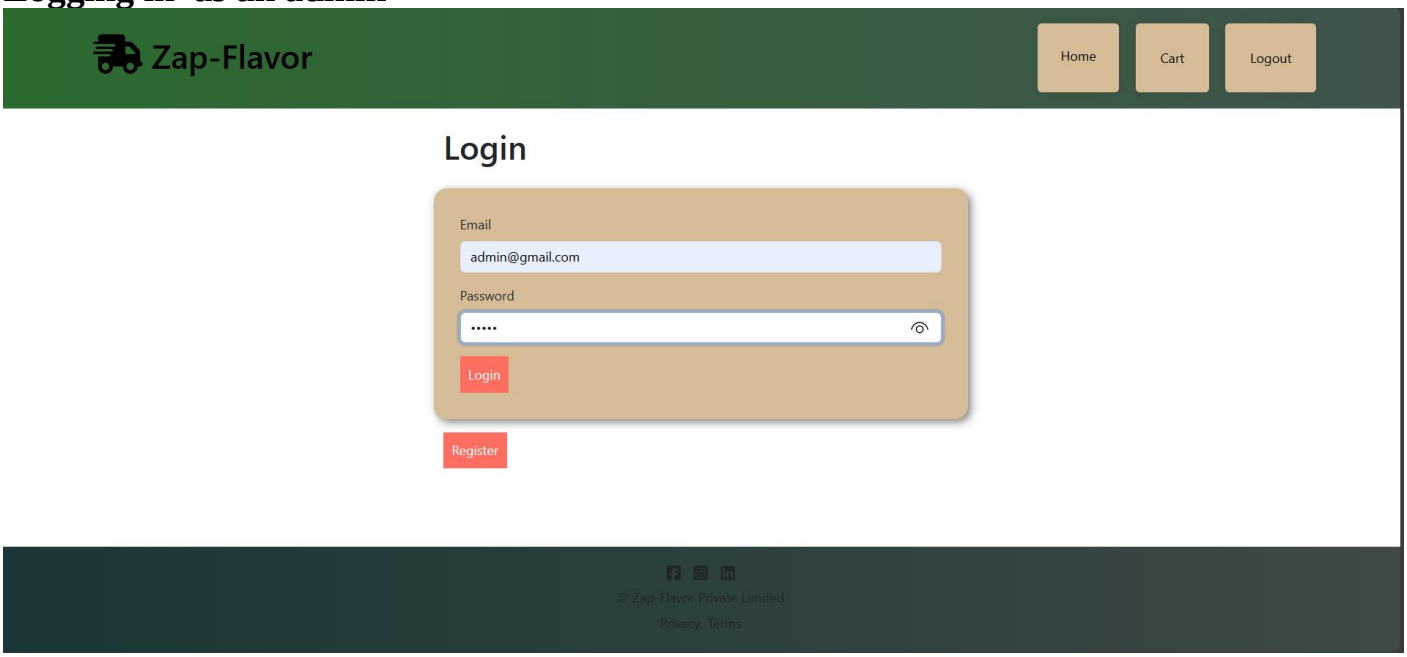
10.1.7 : Order Confirmation Page:

Once placed order a delivery partner will be assigned with all his details displayed

10.2:From Admin's Point Of View(FRONTEND):

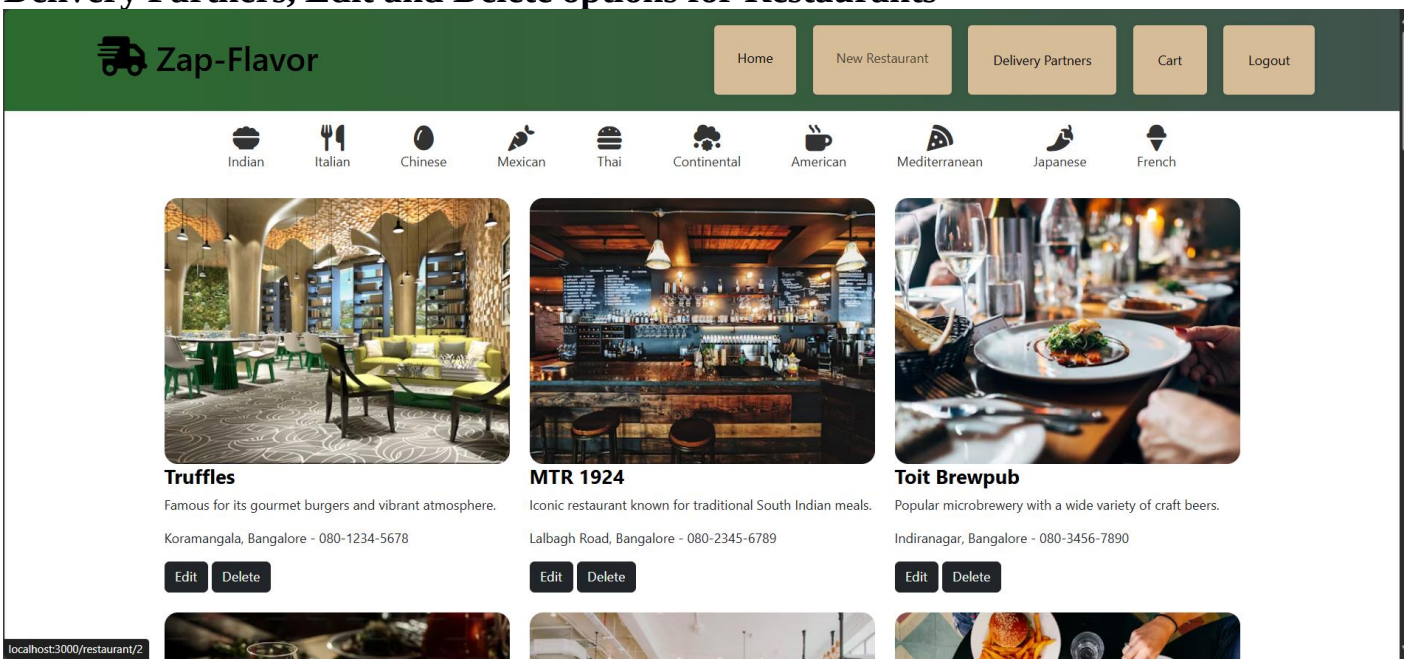
10.2.1: Login Page:

Logging in as an admin



10.2.2: Home Page:

The Admin can view additional buttons on the home page like New Restaurant, Delivery Partners, Edit and Delete options for Restaurants



Edit Delete

Edit Delete

Edit Delete

Brahmins Coffee Bar

Famous for its filter coffee and South Indian snacks.

Basavanagudi, Bangalore - 080-4567-8901

Edit Delete

Smoke House Deli

Offers a mix of continental and Asian cuisine in a cozy setting.

Lavelle Road, Bangalore - 080-5678-9012

Edit Delete

The Fatty Bao

Trendy Asian eatery known for its baos and sushi.

Indiranagar, Bangalore - 080-6789-0123

Edit Delete

Koshys

Legendary cafe famous for its breakfast and traditional dishes.

St. Marks Road, Bangalore - 080-7890-1234

Edit Delete

Vidyaranya Coffee House

A popular spot for students offering great coffee and snacks.

Malleswaram, Bangalore - 080-8901-2345

Edit Delete

The 13th Floor

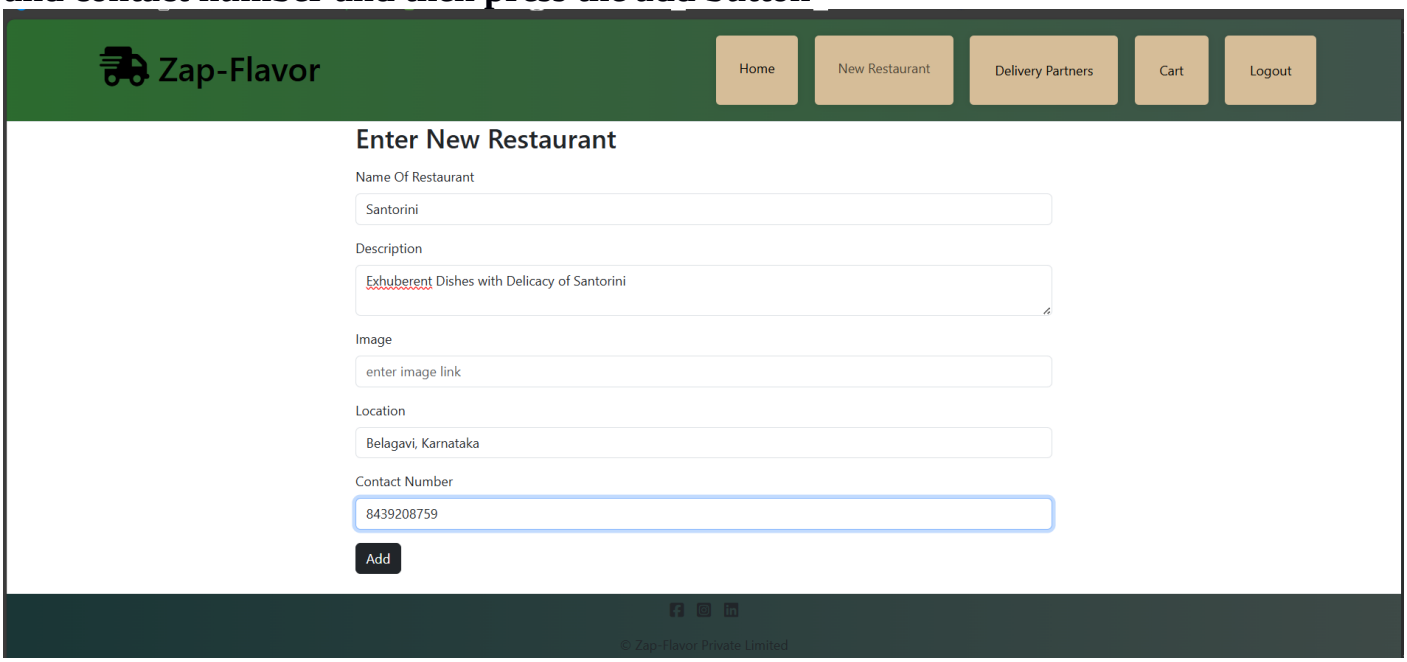
Rooftop restaurant with stunning views and gourmet dishes.

MG Road, Bangalore - 080-9012-3456

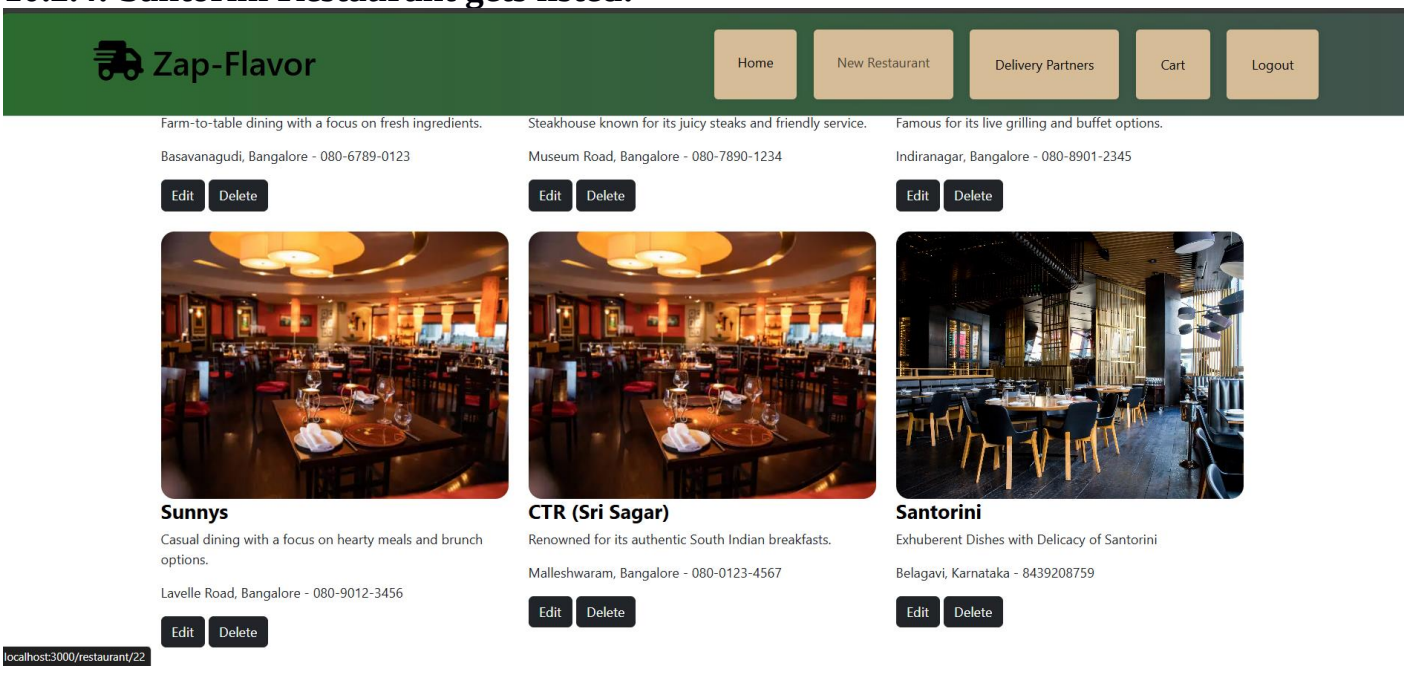
Edit Delete

10.2.3: New Restaurant Creation Page:

The Admin has to Fill the form with all the necessary details like name of the restaurant, description, image link(if not then default image will be used), location and contact number and then press the add button

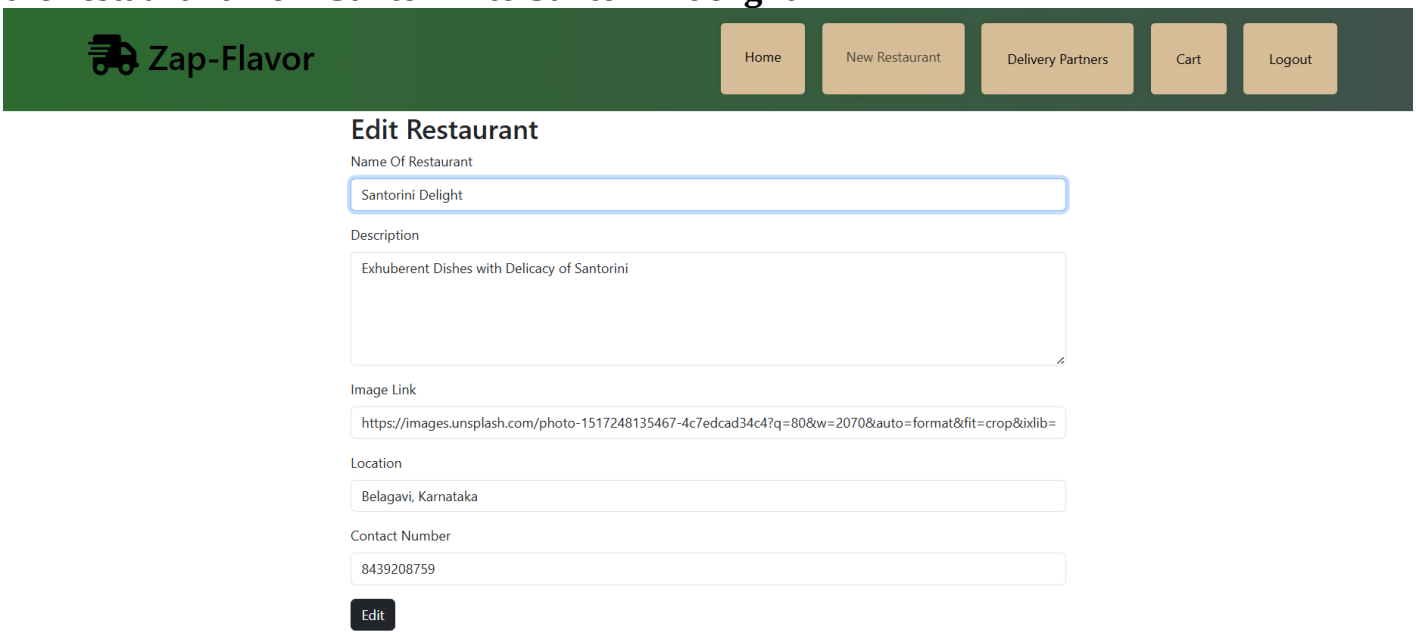


10.2.4: Santorini Restaurant gets listed:

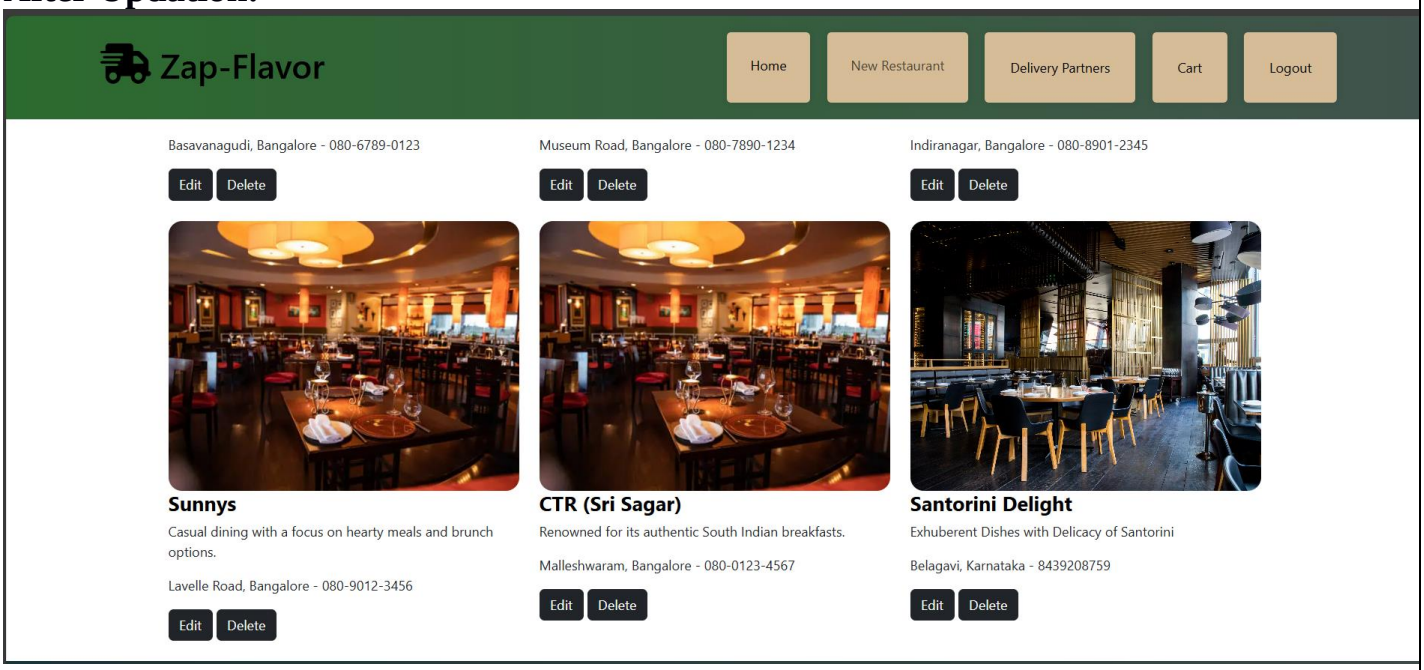


10.2.5: Edit Existing Restaurants Details:

The admin can click the edit button to edit the details lets say he edits the name of the restaurant from Santorini to Santorini delight

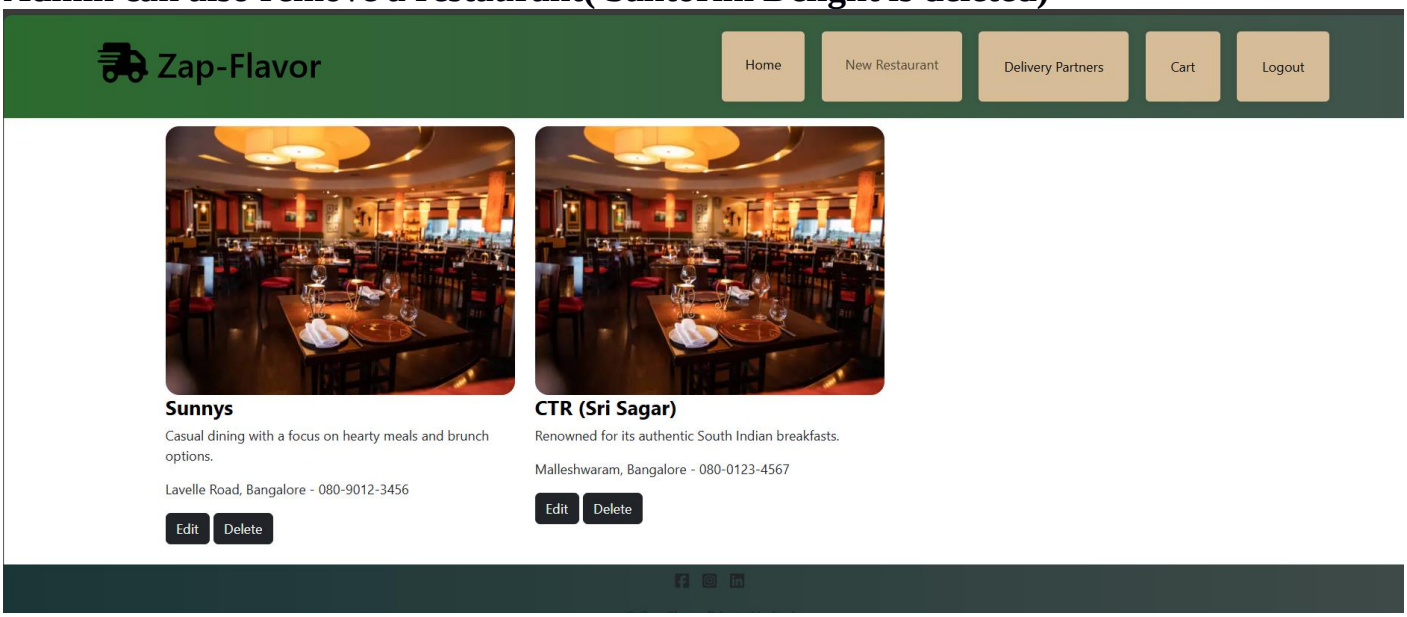


After Updation:



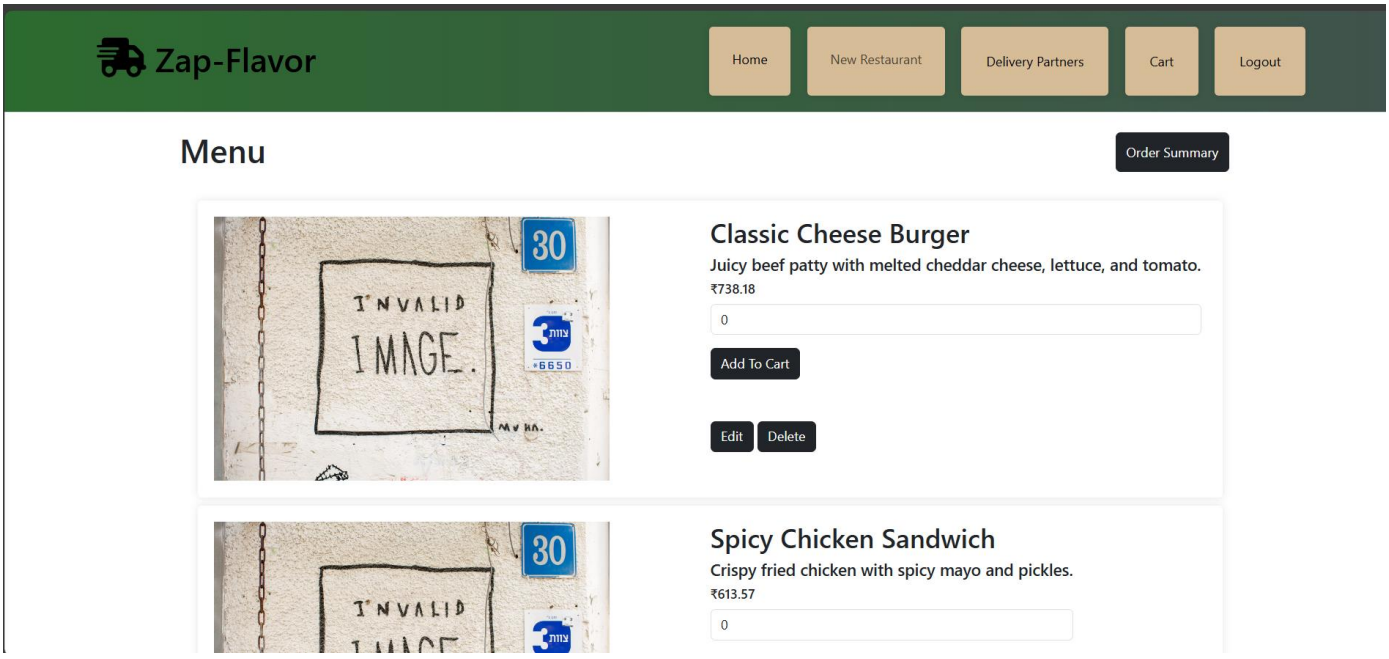
10.2.6: Delete Existing Restaurant:

Admin can also remove a restaurant(Santorini Delight is deleted)



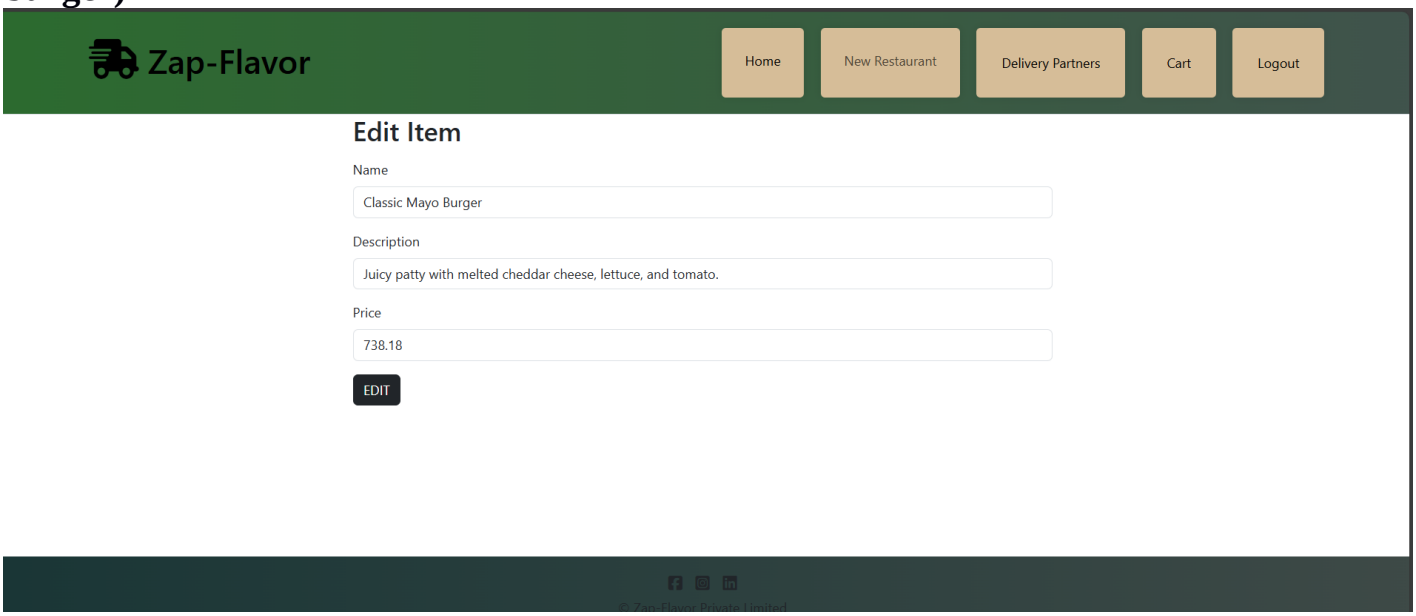
10.2.7: Menu Page:

There are two more buttons available to the admin to add and to delete a menu item

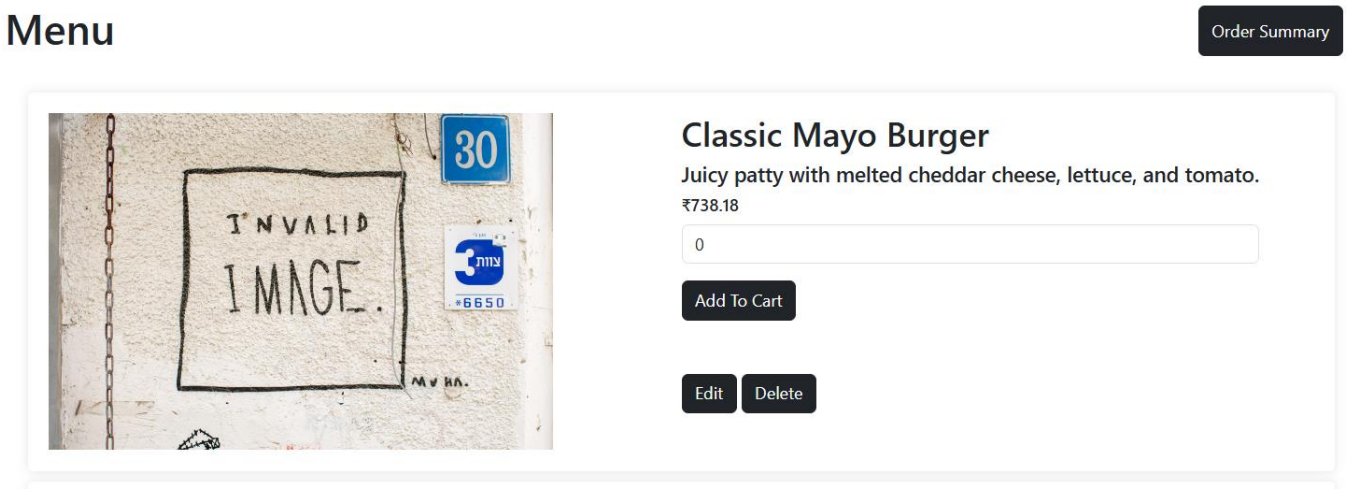


10.2.8: Menu item Edit page:

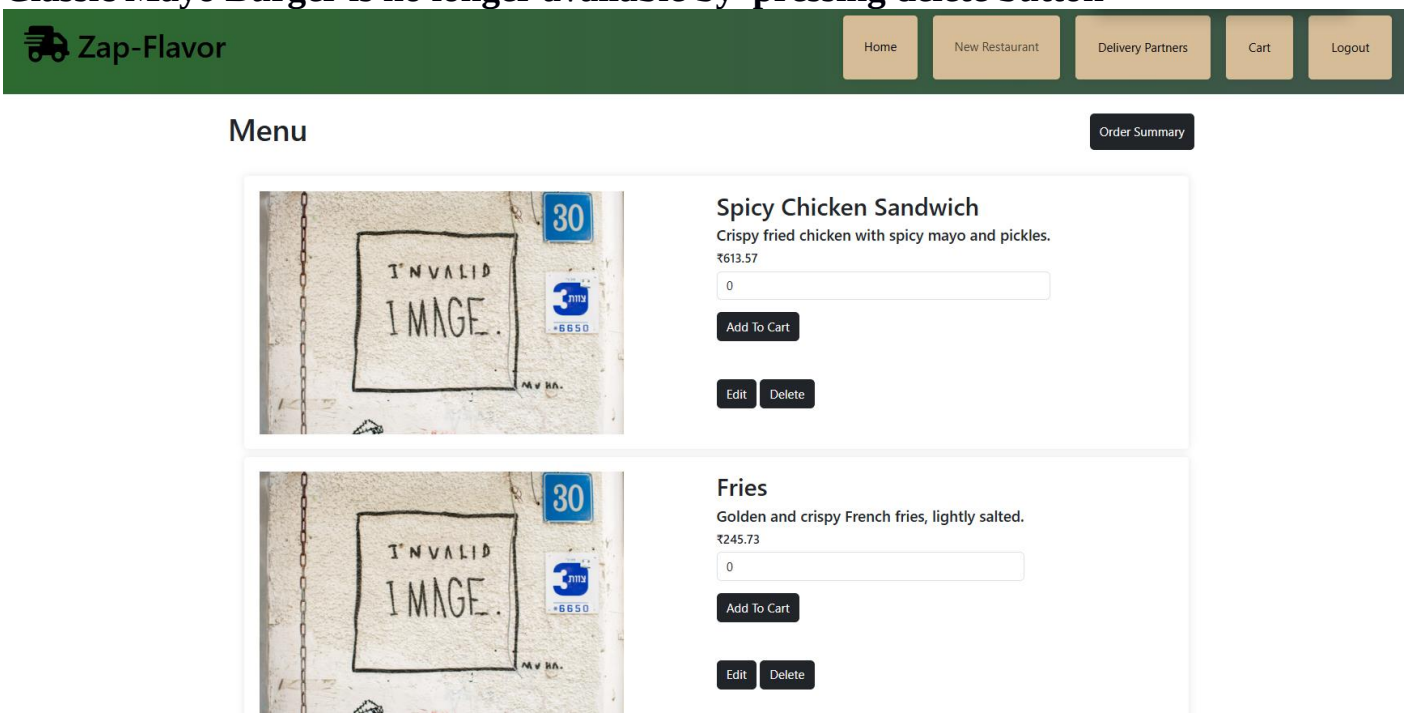
Edit the details of the food item listed here (classic cheese burger to classic mayo burger)



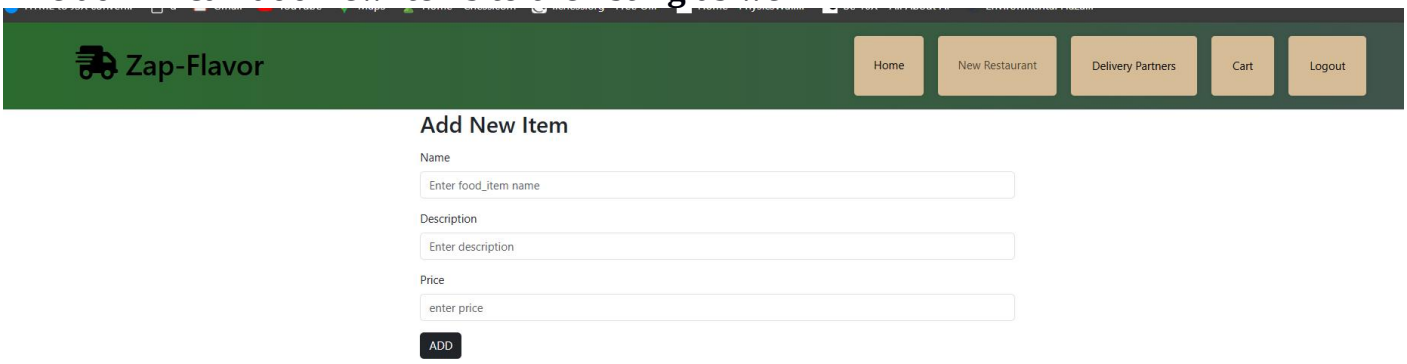
After Updation:



10.2.9: Deletion of Food item:
Classic Mayo Burger is no longer available by pressing delete button

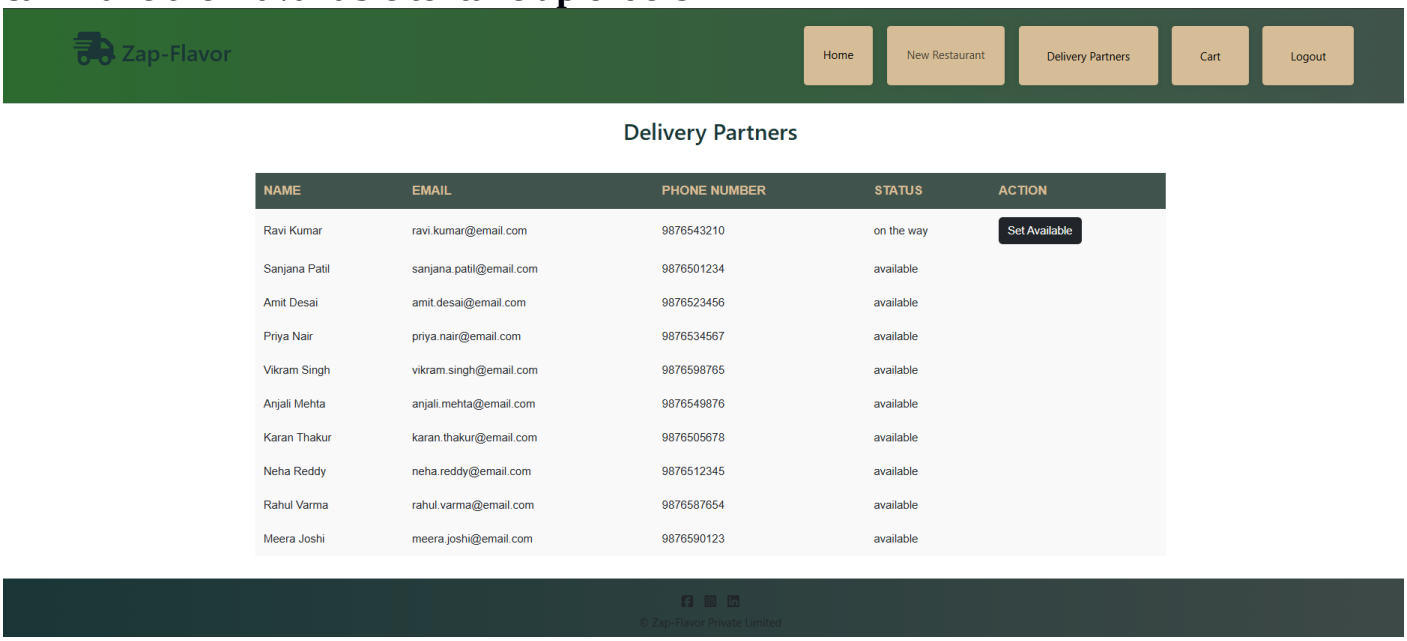


10.2.10: Add New Food item page:
The admin can add new items to the listing as well

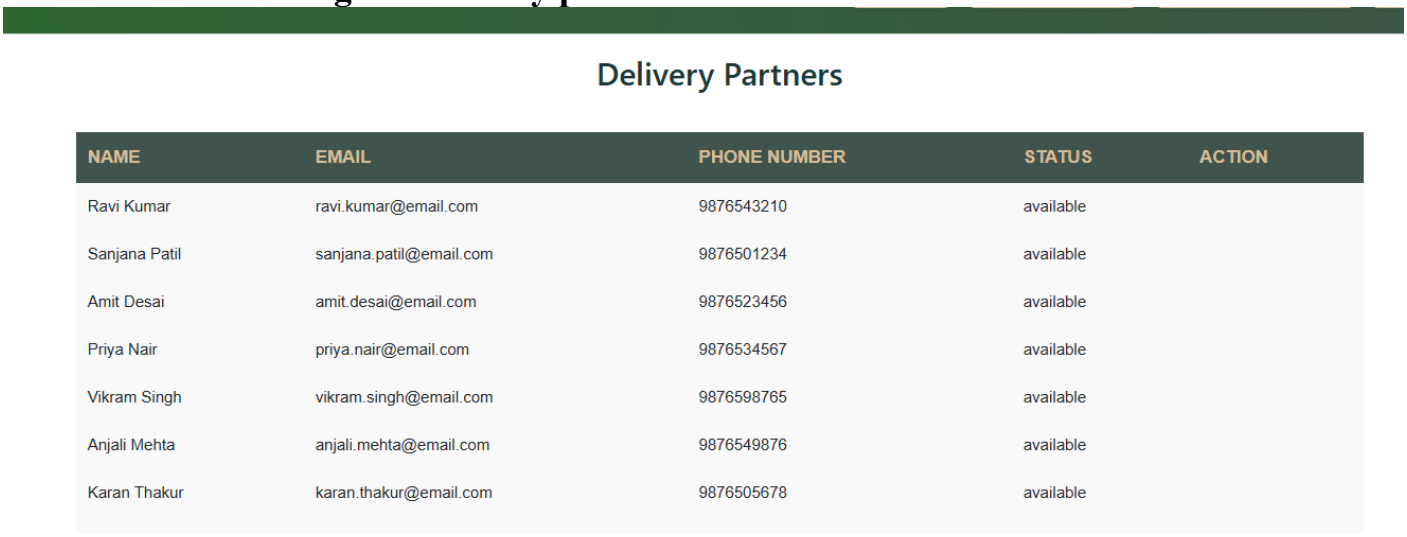


10.2.11: Delivery Partners Page:

By clicking the button on the navbar the delivery partners are listed and the admin can make them available to take up orders



10.2.12: After making the delivery partner 'Available':



11. REFERENCES/BIBLIOGRAPHY

- **Bootstrap Documentation: https://getbootstrap.com/docs Used for responsive design and frontend styling in the project.**
- **EJS Documentation:** EJS -- Embedded JavaScript templates **Official guide for implementing EJS components and state management.**
- **Node.js Documentation: https://nodejs.org/en/docs Reference for server-side JavaScript programming.**
- Express.js Documentation: **https://expressjs.com/ Used for backend routing and middleware integration.**
- **SQL Tutorial: PES University – pesuacademy.com Resource for database query language implementation.**
- **Passport Guide: https://www.passportjs.org/docs/ Passport is middleware for Node.js that makes it easy to implement authentication and authorization..**
- Online Gantt Chart Tool: **https://teamgantt.com/ Tool for managing and visualizing project timelines.**
- Draw.io Documentation: **https://drawio-app.com/ Used for creating the ER diagram for the database.**

APPENDIX A DEFINITIONS, ACRONYMS AND ABBREVIATIONS

A.a. Definitions:

- **ZapFlavor**: A platform designed to allow users to order food from various restaurants, view restaurant menus, manage carts, and track their orders.
- **Order Summary Page**: A page where users can review their cart, see added items, and proceed with placing an order.
- **Cuisine-Based Restaurant Navigation**: A feature allowing users to browse restaurants based on the type of cuisine they prefer.
- **Admin User**: A user role that has the authority to edit or delete restaurant information within the system.
- **User Authentication**: The process of verifying the identity of a user to ensure secure access to the platform.
- **Responsive Design**: A design approach ensuring the platform adapts seamlessly to various devices like desktops, tablets, and smartphones.
- **SQL Database**: A relational database used to store order information, restaurant details, and customer profiles.

- **Random Delivery Partner Assignment**: The process by which an order is assigned to a randomly available delivery partner when the user presses the 'Place Order' button.
- **Delivery Partner Status Update**: A feature ensuring the delivery partner's status updates to 'on the way' once an order has been assigned to them.
- **EJS**: Embedded JavaScript, used for dynamically rendering HTML pages within the application.

A.b. Acronyms and Abbreviations:

- **SQL**: Structured Query Language, used for managing and querying data in relational databases.
- **PASSPORT**: middleware for Node.js that makes it easy to implement authentication and authorization
- **CRUD**: Create, Read, Update, Delete – basic operations performed on database records.
- **UI**: User Interface, the part of the application with which users interact.
- **UX**: User Experience, the overall experience of a user while interacting with the platform.
- **API**: Application Programming Interface, enabling communication between the frontend and backend of the system.
- **HTTP**: Hypertext Transfer Protocol, the foundation for data communication on the web.
- **Restaurant Permissions**: Admin privileges to manage restaurant details, including edit and delete capabilities.