**RESTAURANT ORDERING SYSTEM**

*A Project Based Learning Report Submitted in partial fulfilment of the requirements for the award of the degree*

*of*

**Bachelor of Technology**

**in The Department of CSE**

**Full Stack Application Development with 23SDCS12E**

Submitted by

**2310030130: Afraz**

**2310030108: Santhosh**

**2310030123: Siddarth**

**2310030292: Sashank**

Under the guidance of

**Prof. Chiranjeevi Sir**



Department of Electronics and Communication Engineering

Koneru Lakshmaiah Education Foundation, Aziz Nagar

Aziz Nagar – 500075 (Optional)

NOV - 2023.

**Table of Contents**

**Abstract**

**List of Figures**

**List of Tables**

**Table of Contents**

**Introduction**

**Methodology**

**Experiments**

**Results**

**Conclusion and Future Work**

**Abstract**

The Restaurant Ordering System is a comprehensive and modern full-stack web application that revolutionizes the food ordering process in restaurants. Traditional methods often lead to inefficiencies like delays, errors, and miscommunication, which negatively affect the customer experience. By introducing a digital platform, this system addresses these issues, providing customers with an intuitive, interactive interface to browse menus, customize orders, and place them online seamlessly. The system also enhances restaurant operations by providing staff and administrators with real-time tools to manage orders, monitor customer interactions, and update menu items efficiently.

The system is built using a modular architecture, where the frontend and backend are clearly separated to ensure scalability and maintainability. The frontend is developed with Vue.js, which ensures a fast, dynamic, and responsive user experience. Vue.js makes it easy to manage the state of the application, handle real-time updates, and deliver a smooth user interface. On the backend, Node.js with Express.js is used to handle the business logic, route HTTP requests, and facilitate communication with the database. Express.js provides a flexible framework to implement RESTful APIs for efficient communication between the frontend and backend.

For data storage, MySQL is used, providing reliable and structured data management. XAMPP serves as the local server environment, offering an easy-to-use platform for testing and managing the MySQL database during development. This local setup ensures smooth development and testing cycles, providing a straightforward way to manage the database without requiring complex server configurations.

The system adopts a **RESTful API architecture**, allowing for smooth communication between the frontend and backend, ensuring that both parts of the application can function independently and scale as needed. This architecture allows for modularity and flexibility, making it easier to maintain, upgrade, and extend the system in the future.

In conclusion, the Restaurant Ordering System is designed not only to improve the operational efficiency of restaurants but also to enhance customer experience by reducing wait times, minimizing human errors, and offering the convenience of online ordering. The project demonstrates the power of modern web technologies in solving real-world challenges and emphasizes the importance of full-stack development in building scalable, effective software solutions. By addressing both customer and operational needs, this system serves as a practical application of full-stack development, showcasing the critical role it plays in creating innovative, real-time software solutions.

**List of Figures**

**Figure 1: System Architecture Diagram**This diagram provides an overview of the system's modular architecture, showing the interaction and separation of concerns between the frontend, backend, and database components. It illustrates how the different parts of the application communicate with each other, with a focus on scalability, maintainability, and performance optimization.

**Figure 2: Customer Ordering Interface**This screenshot showcases the customer-facing interface of the system. It includes key functionalities such as browsing the menu, selecting items, customizing orders (e.g., adding special instructions), and submitting the order. The layout is designed to be intuitive, providing a smooth experience for customers while making the ordering process fast and simple.

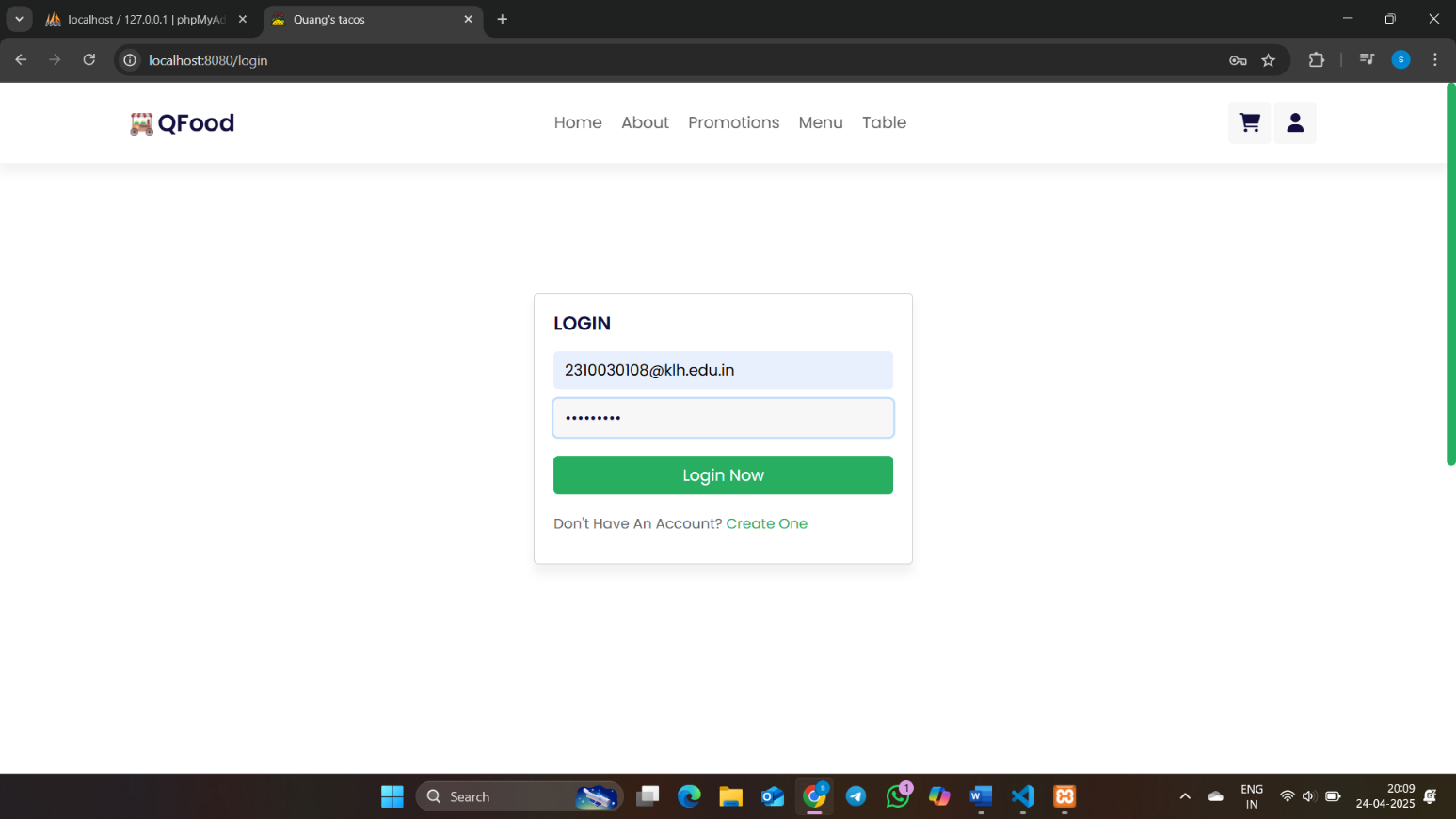
**Figure 3: User Authentication Flow**This flowchart illustrates the user authentication process in the application. It shows the steps involved in user registration, login, and role-based access control, where customers, staff, and admins are granted different levels of access based on their roles. This ensures a secure and controlled environment for users interacting with the system.

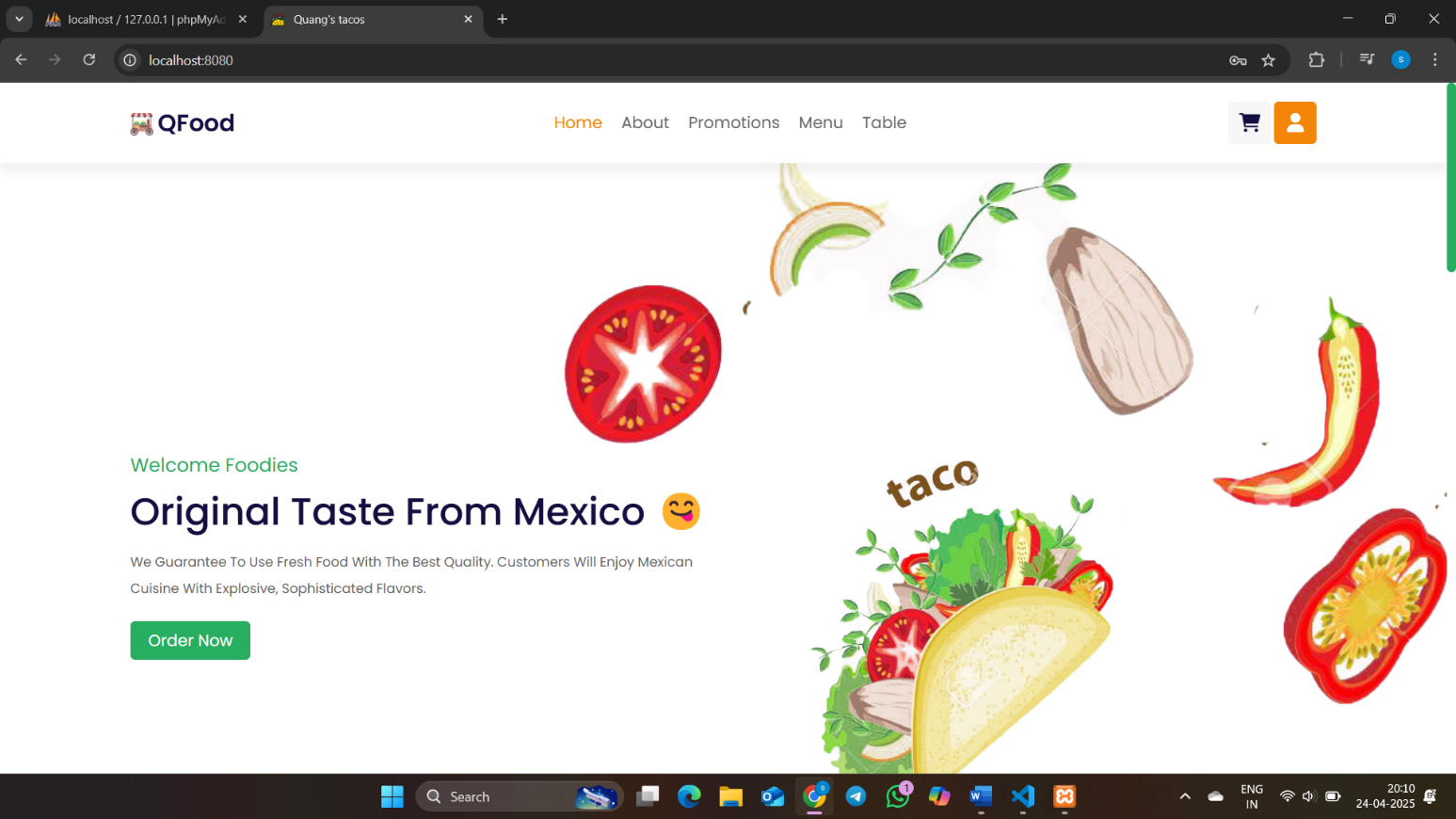
**Figure 4: Dynamic Menu Rendering**This screenshot demonstrates the dynamic rendering of menu items, where the system retrieves menu data from the backend and displays it to customers in real-time. The menu is highly customizable, with items updating automatically based on availability, price changes, or new additions, offering a dynamic user experience without requiring manual updates.

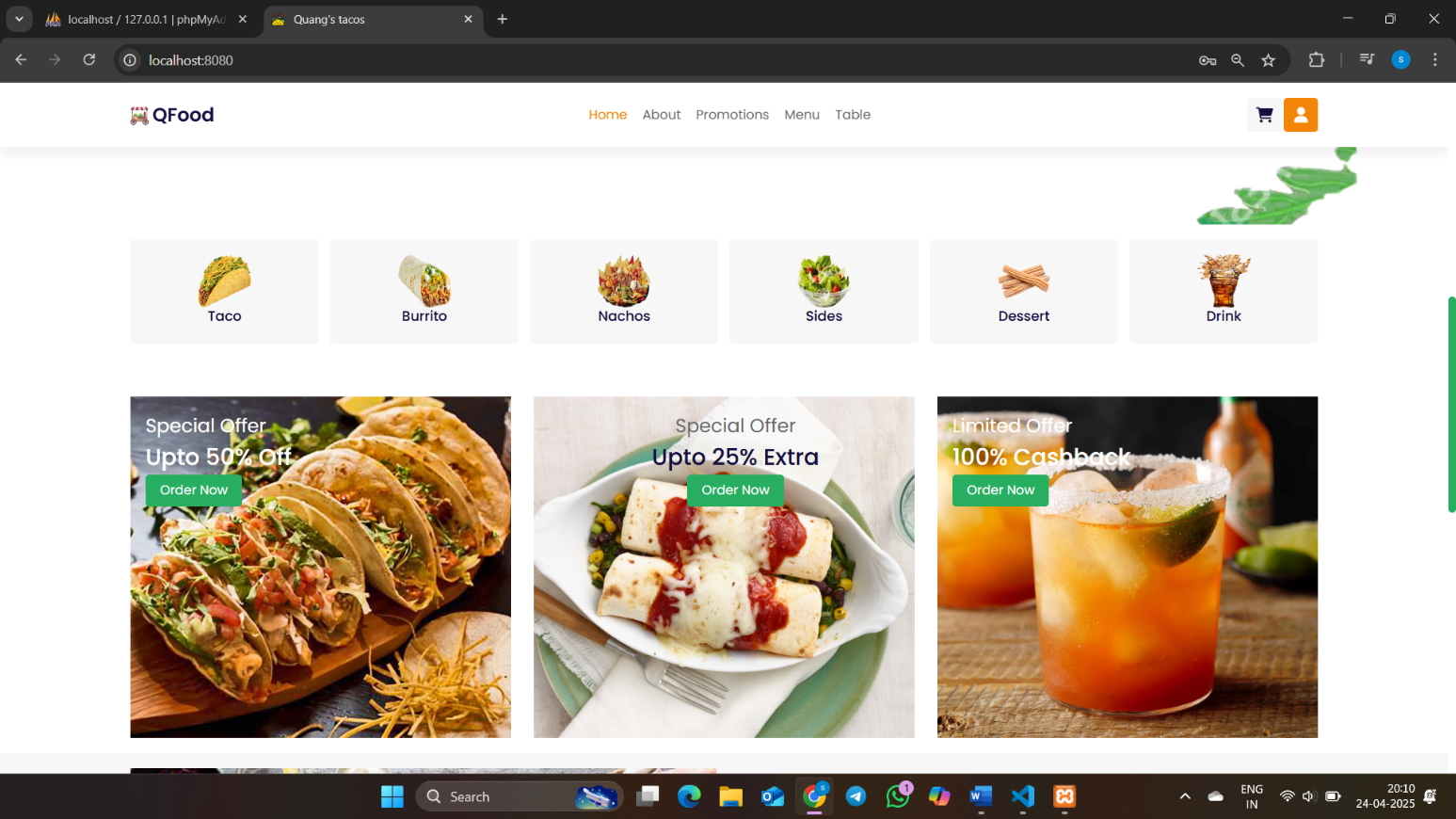
**Figure 5: Admin Dashboard for Order Tracking**This figure shows the admin dashboard, which provides a real-time overview of all orders placed in the system. Admins can monitor the progress of each order, update the status (e.g., preparing, completed, delivered), and track food deliveries. The interface is designed for easy navigation, allowing admins to manage the order flow effectively and ensure timely delivery.

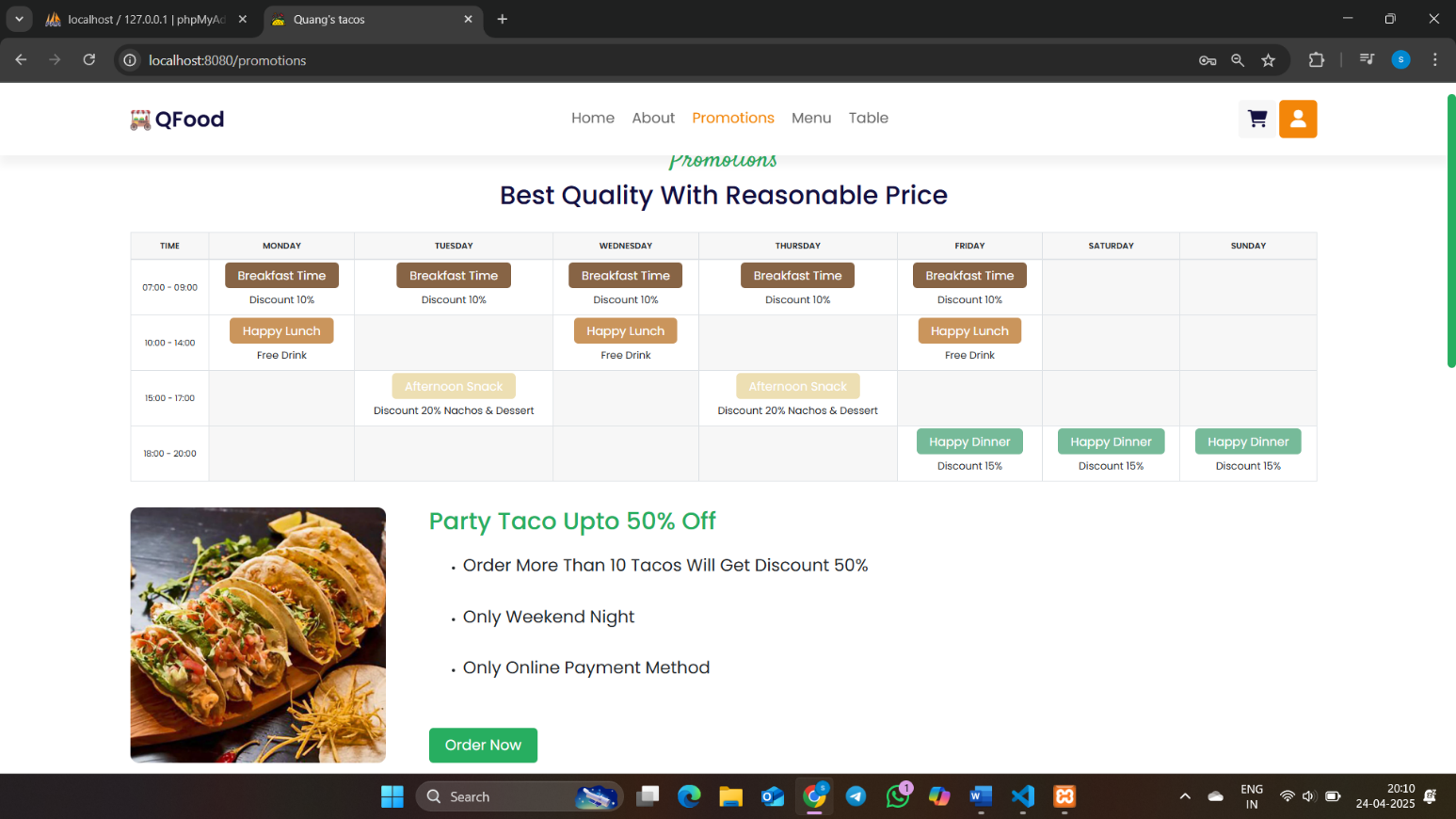
**Figure 6: Real-Time Order Updates**This diagram illustrates the system's real-time order update functionality, where both customers and restaurant staff receive instant notifications about the status of their orders. The feature ensures that users are kept informed throughout the entire order process, from placement to delivery, improving customer satisfaction and communication within the restaurant.

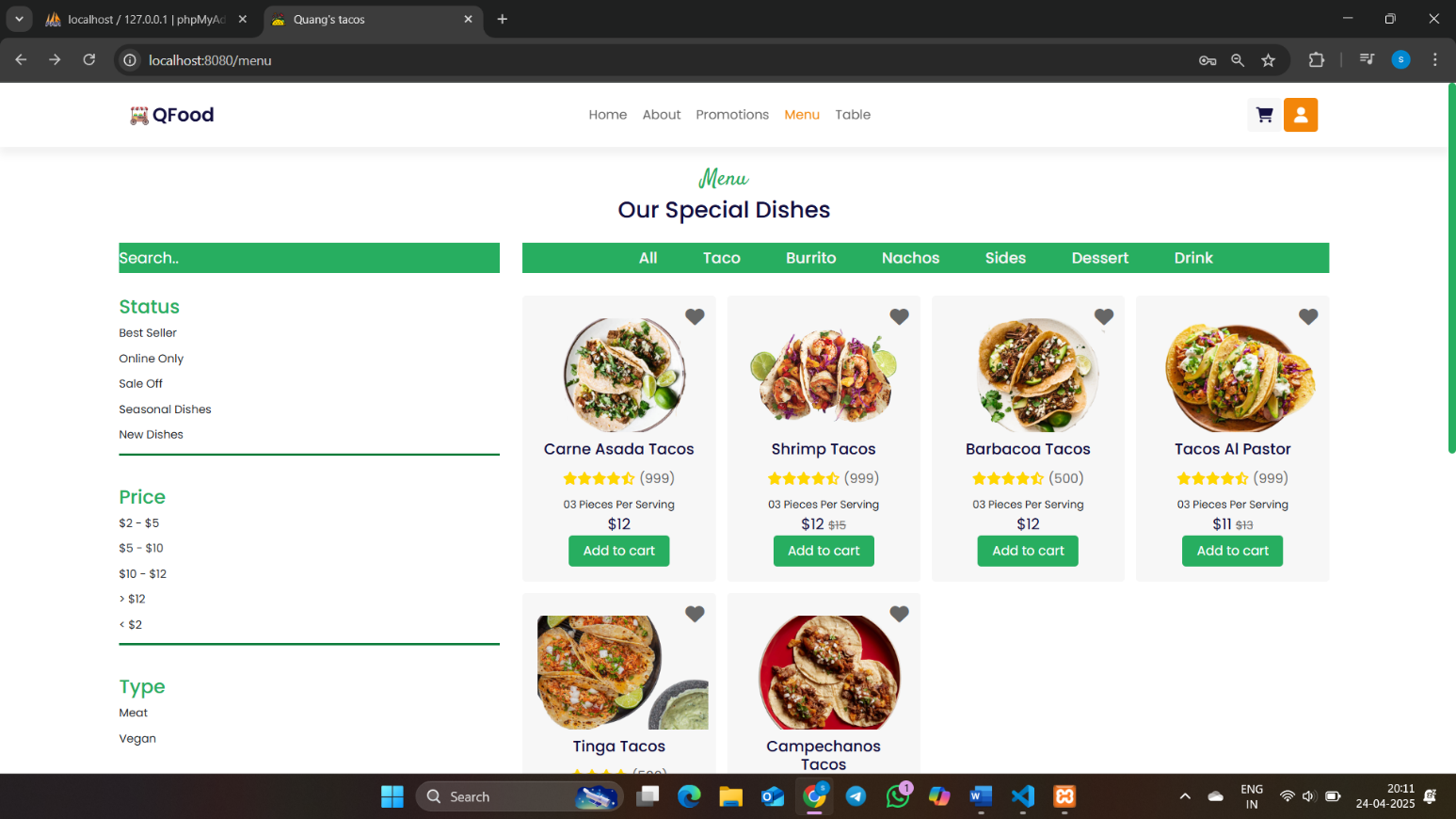
**List of Figures(ScreenShots)**

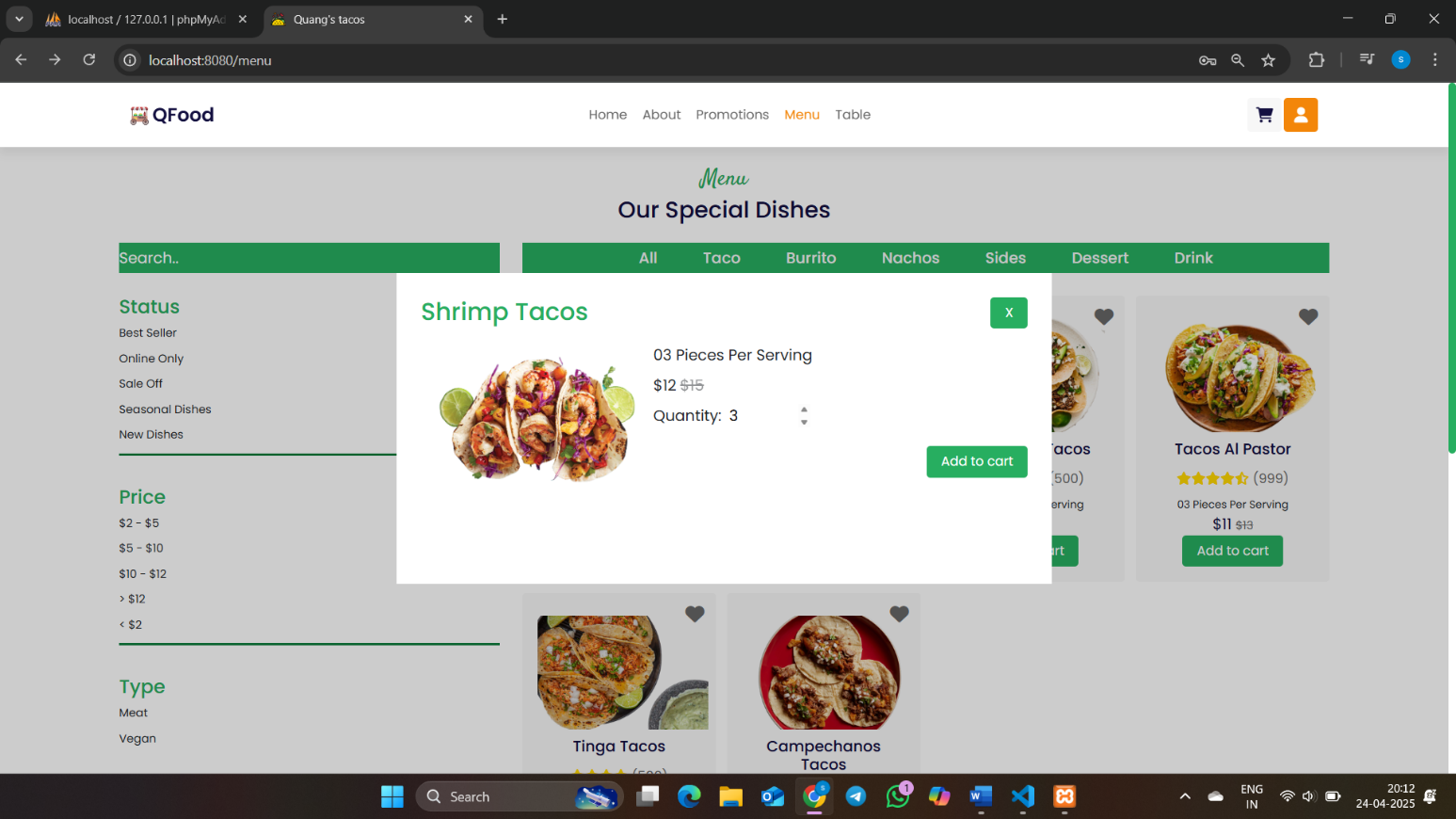
****

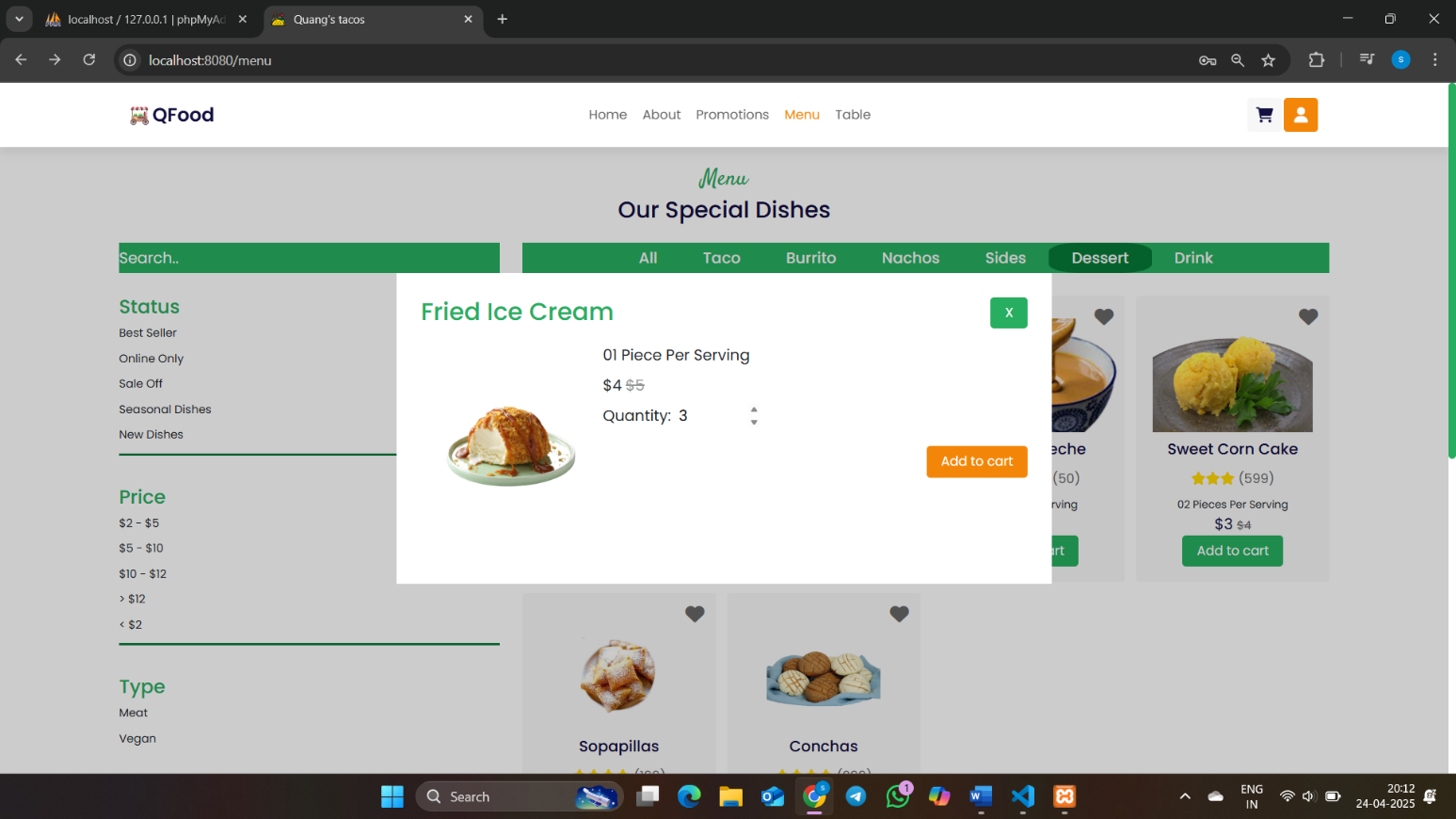
****

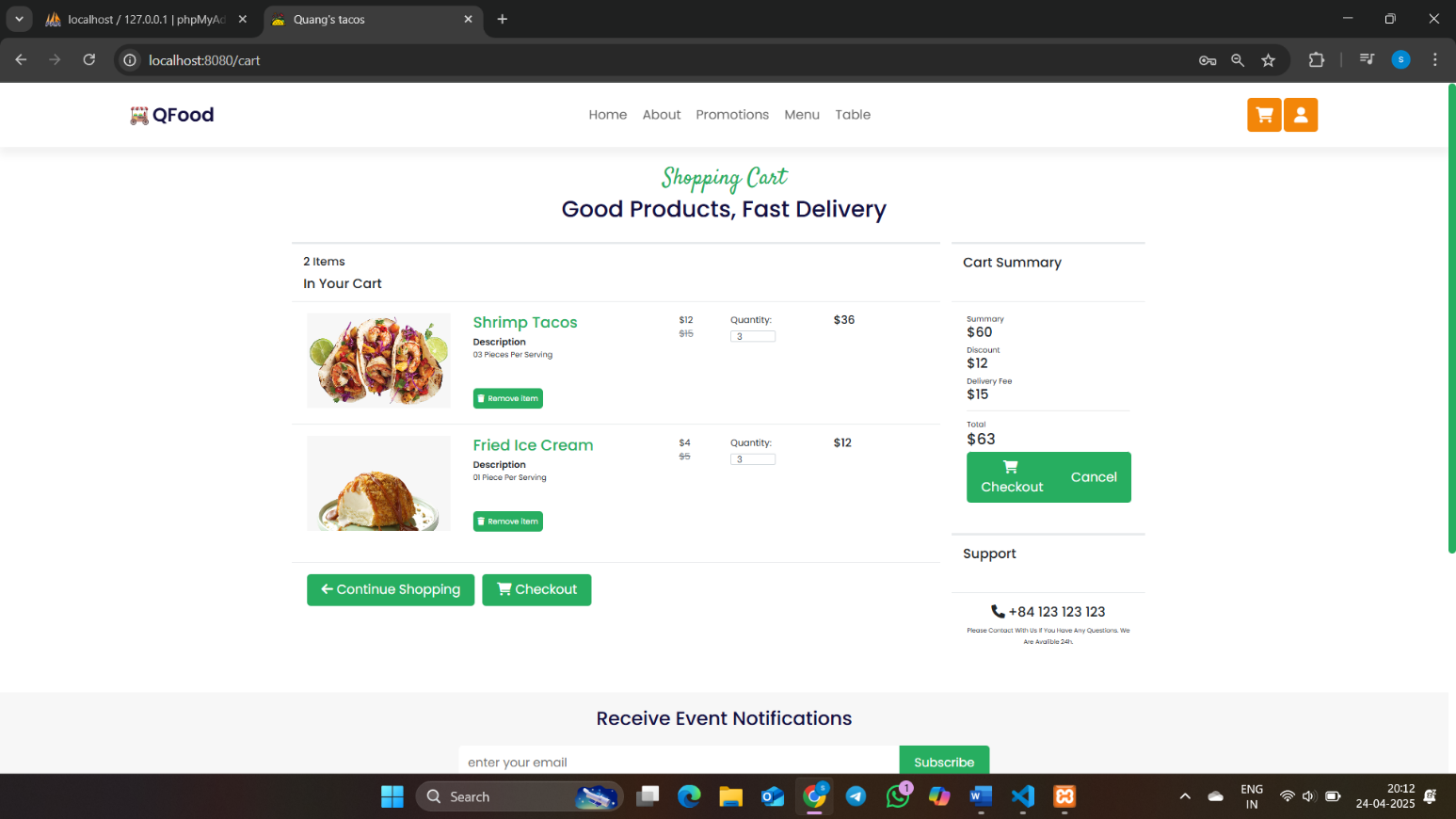
****

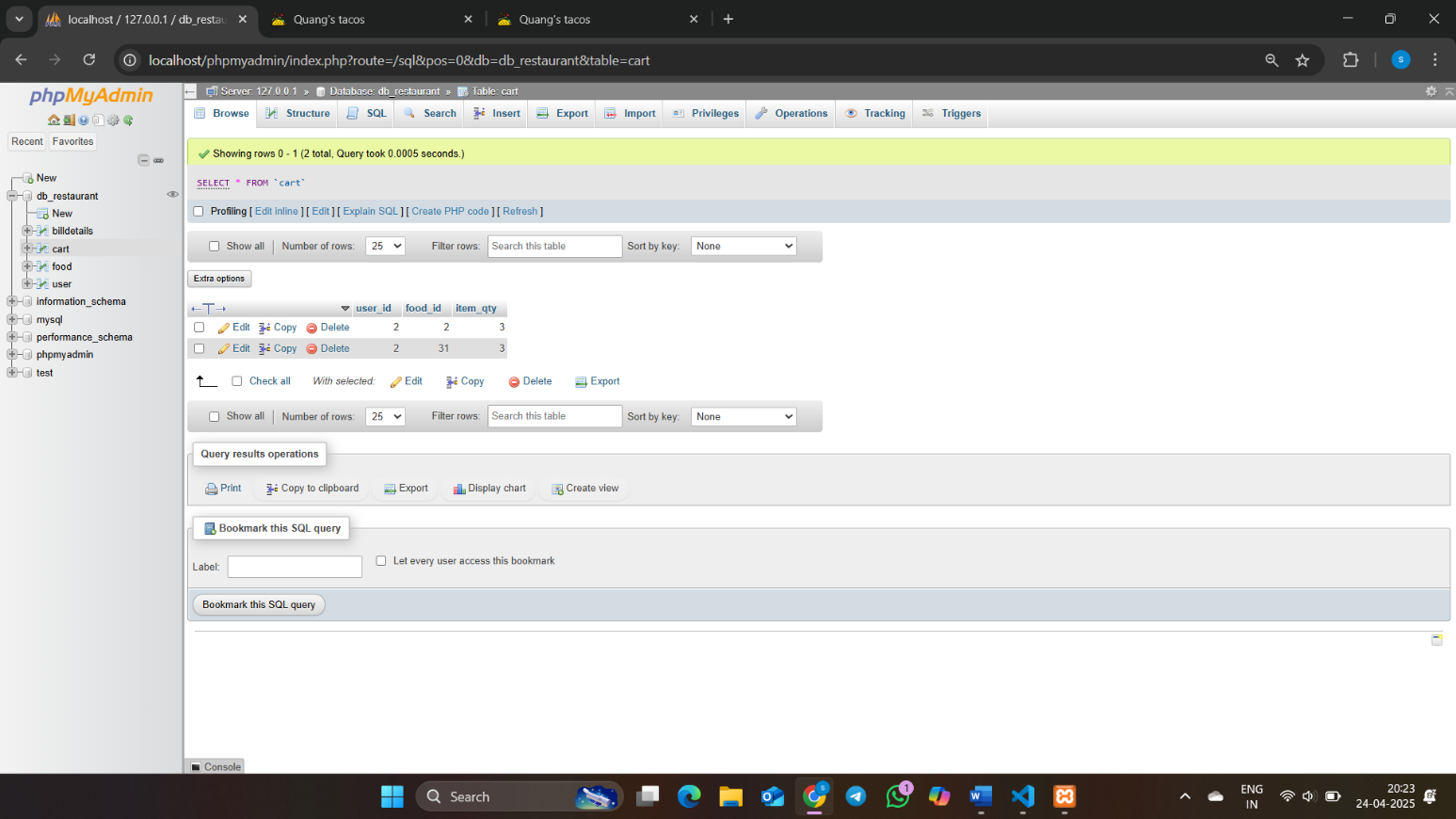
****

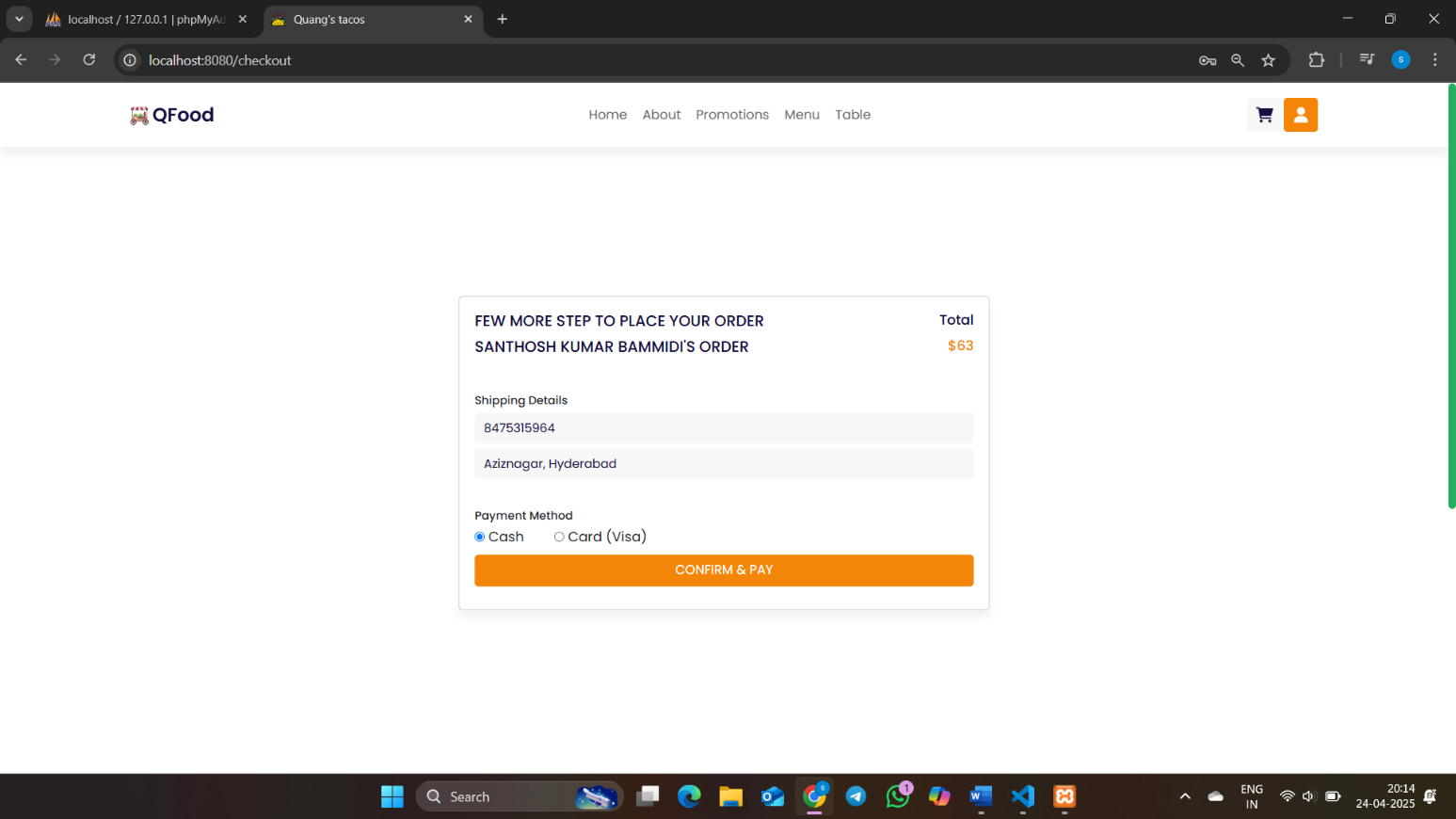
****

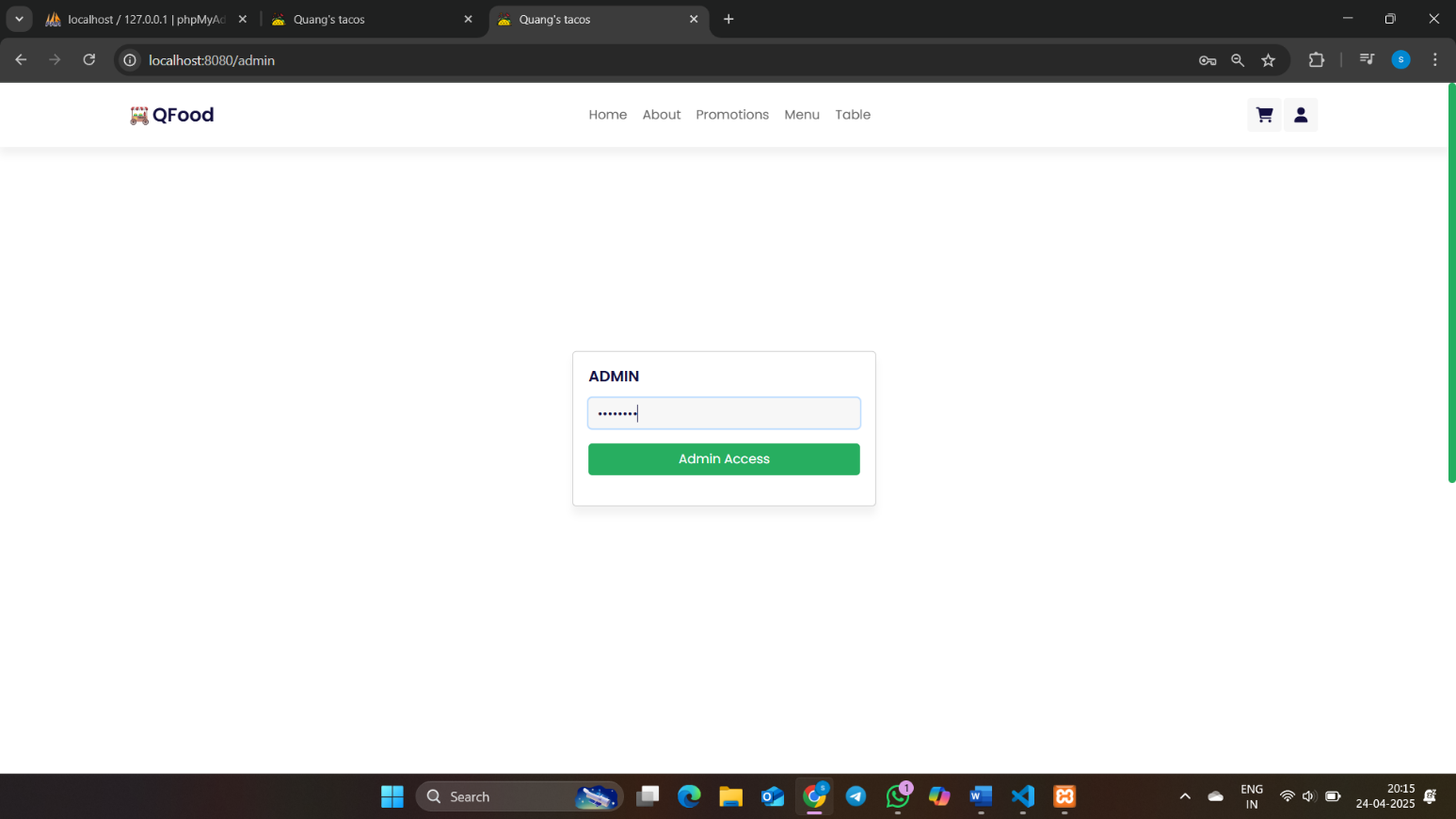
****

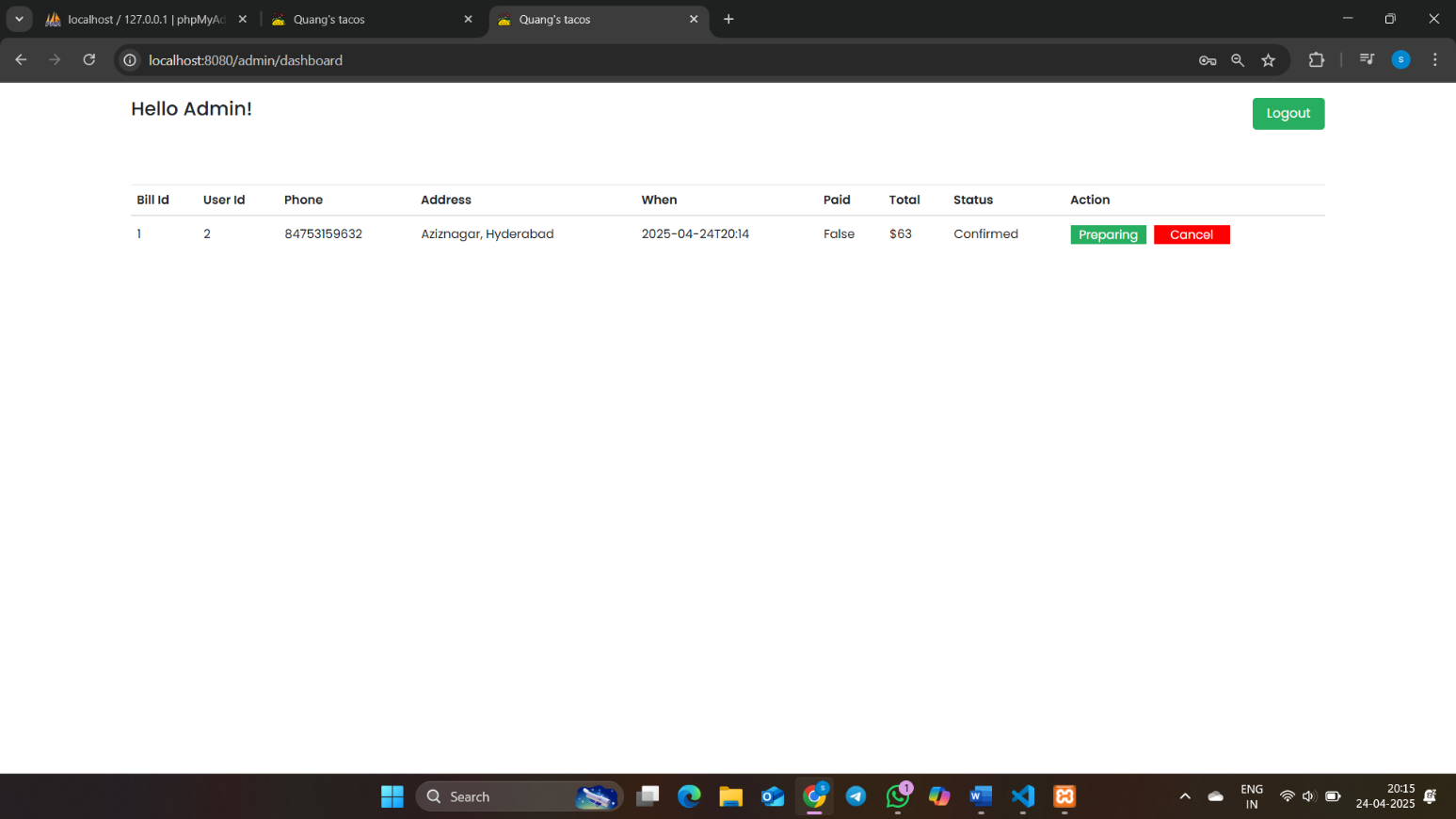
****

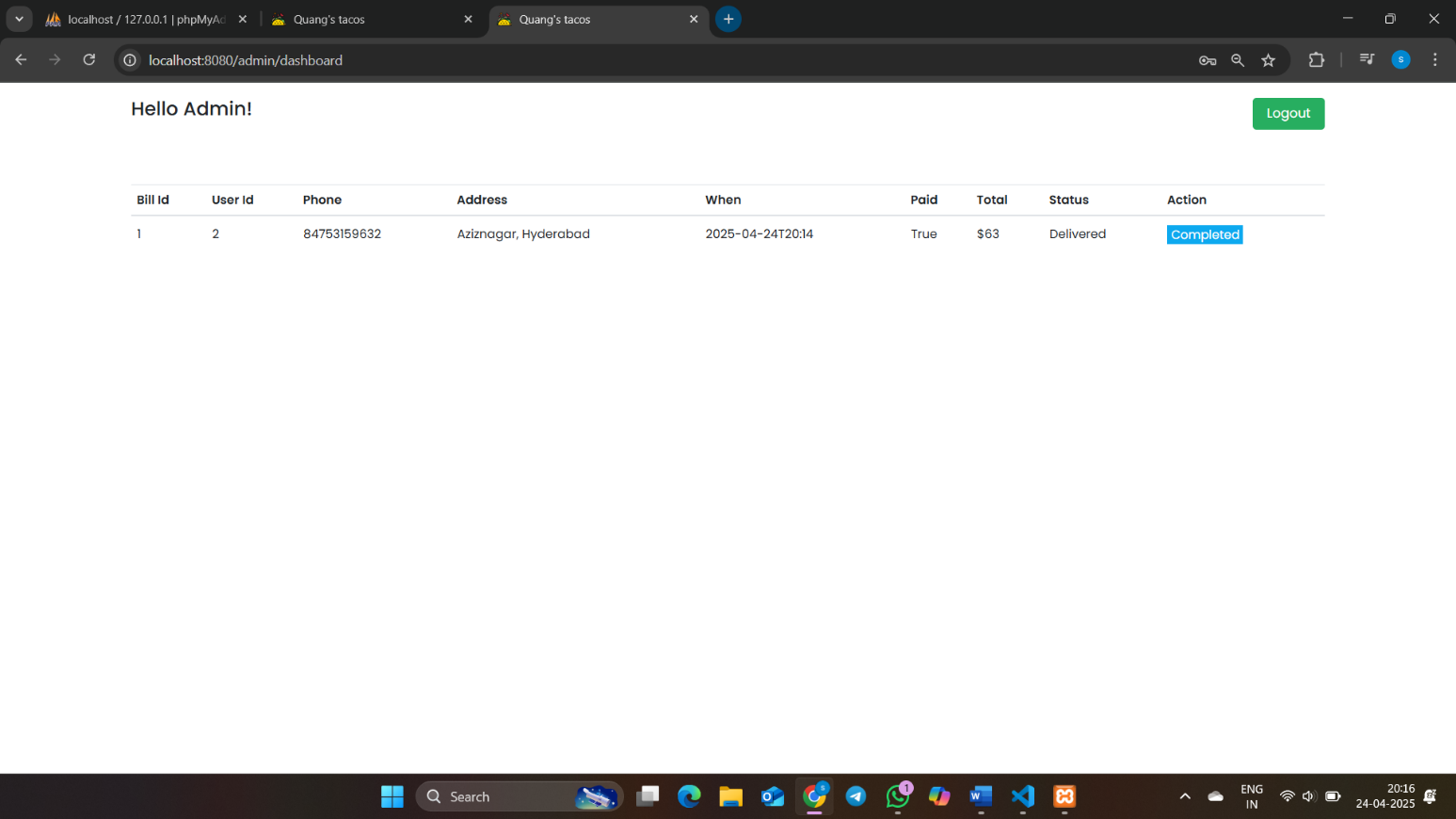
****

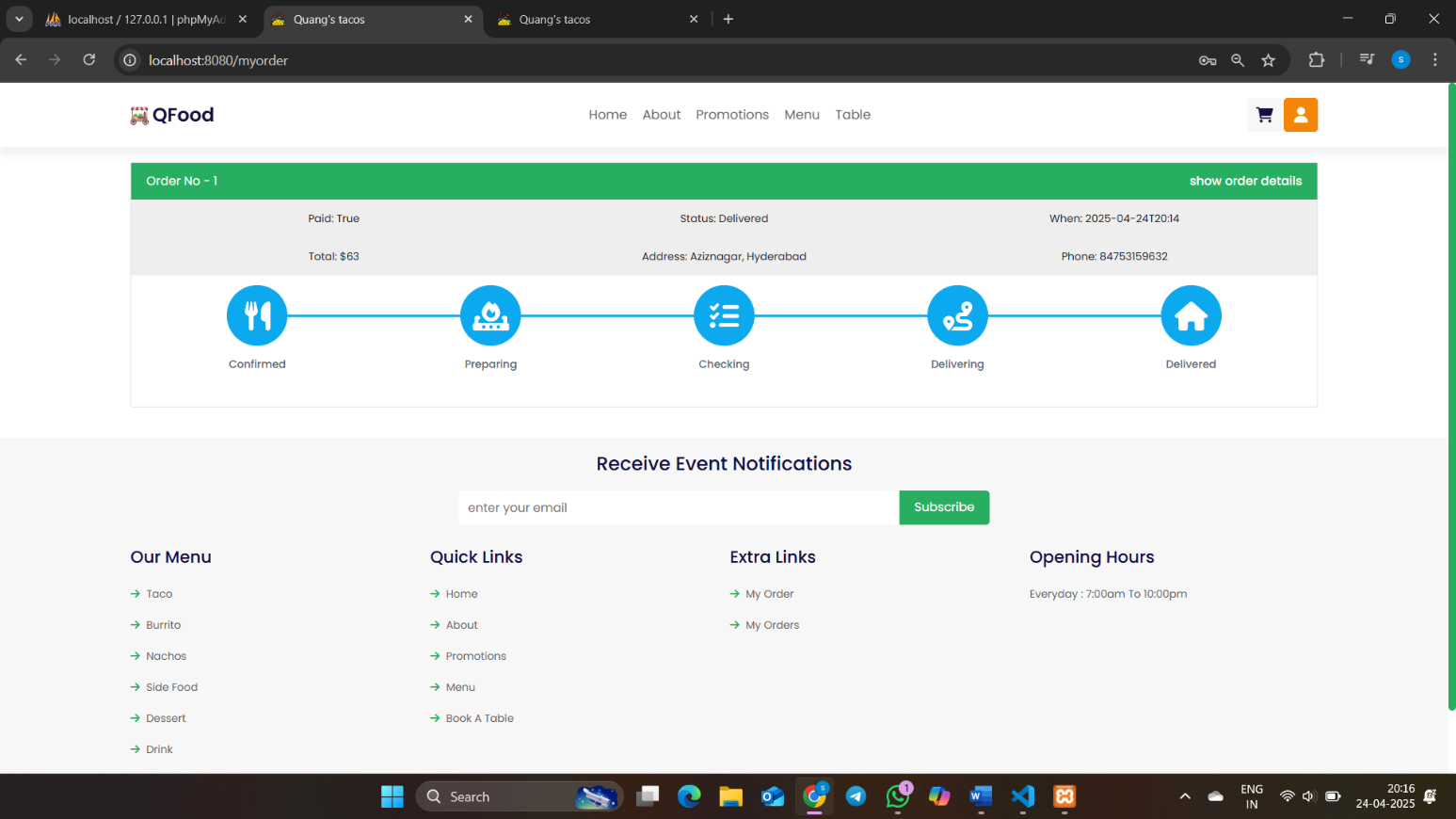
****

****

****

****

****

****

**List of Tables**

**Table 1: Customers**This table holds customer information, including Customer ID (primary key), name, email, phone number, and delivery address. It also stores encrypted passwords for user authentication and a history of past orders. The Customer ID acts as a unique identifier for each user, helping link their orders to their profile. This table is integral to customer management, personalized services, and order tracking.

**Table 2: Orders**The Orders table captures information about each order placed by a customer. Key fields include Order ID (primary key), Customer ID (foreign key), total order amount, order status (e.g., pending, preparing, delivered), and timestamps (order placed and last updated). It allows the system to track and update the status of each order. This table connects to the Customers table via Customer ID, enabling detailed order history for individual customers.

**Table 3: MenuItems**This table stores details of the items available on the restaurant’s menu. Each record includes Item ID (primary key), item name, description, price, and availability status. This table is critical for rendering the menu dynamically on the customer-facing interface and managing menu updates (e.g., new items, price changes, or seasonal availability). It also supports categorizing items, such as appetizers, mains, and desserts.

**Table 4: Order\_Items**The Order\_Items table bridges the Orders and Menu Items tables, detailing the specific items within each order. Fields include Order Item ID (primary key), Order ID (foreign key), Item ID (foreign key), quantity, and any special instructions (e.g., dietary restrictions). This table provides a breakdown of each customer's order, facilitating accurate processing and ensuring that the correct items are prepared and delivered.

**Table 5: Staff**This table contains details about restaurant staff, including Staff ID (primary key), name, role (e.g., chef, server, admin), username, password, and work schedule. The Staff table supports role-based access control, ensuring that users have permissions based on their job roles. It also helps manage staff shifts, track their login activities, and assign responsibilities, such as order management or updating statuses.

**RESTAURANT ORDERING SYSTEM**

# **Introduction**

The Restaurant Ordering System is a web application designed to streamline the food ordering process in restaurants. It enhances the customer experience by offering an efficient platform for browsing the menu, placing orders, and tracking delivery statuses in real-time. It also empowers restaurant staff with tools to manage orders, monitor food preparation, and ensure timely delivery, using modern technologies to provide a scalable solution for the restaurant industry.

With the growing trend of digital transformation in the foodservice sector, restaurants are increasingly adopting technology to improve operational efficiency and customer satisfaction. The Restaurant Ordering System addresses these needs by providing an easy-to-use interface for customers and an intuitive dashboard for staff. It simplifies the ordering process, reduces human error, and provides both customers and staff with instant updates about order status.

A key feature is role-based access control (RBAC), ensuring that users such as customers, staff, and administrators can access only relevant features. Customers can browse the menu, place orders, and track deliveries, while staff can update order statuses and manage queues. Admins have advanced functionalities for monitoring performance, managing users, and updating the menu.

The system also includes dynamic menu rendering, which updates in real-time based on the restaurant's inventory, ensuring customers only see items in stock. It allows order customization, such as adding special instructions or adjusting quantities, catering to individual preferences.

The backend uses a MySQL database to store and manage data, with communication between the frontend and backend facilitated by a RESTful API. The application is built using Vue.js for the frontend, Node.js with Express.js for the backend, and MySQL for database management, ensuring reliability and scalability.

Overall, the Restaurant Ordering System is an integrated solution that improves restaurant operations and enhances the customer dining experience. It leverages modern web technologies, setting a foundation for future scalability and growth.

# **METHODOLOGY**

The development of the Restaurant Ordering System followed an Agile approach, incorporating key phases: requirement gathering, design, development, testing, and deployment.

**1. Requirement Gathering**  
Requirements were gathered from stakeholders, including restaurant owners and customers, to identify key features like order management, real-time updates, and role-based access control.

**2.System Design**  
The system was designed with a modular architecture, using Vue.js for the frontend, Node.js with Express.js for the backend, and MySQL for the database. A RESTful API was implemented for frontend-backend communication.

**3.Development**  
Development was carried out in sprints, focusing on features such as dynamic menu rendering and order tracking. Real-time updates were implemented using web sockets.

**4.Testing**  
Unit tests, integration tests, and user acceptance testing (UAT) were conducted to ensure the system functioned correctly and met user expectations.

**5. Deployment and Maintenance**  
After testing, the system was deployed and is continuously monitored for performance. Regular updates are made based on user feedback.

# **EXPERIMENTS**

Several experiments were conducted to test the core features and performance of the Restaurant Ordering System.

**1. User Interaction Testing**  
Users tested the interface by browsing the menu, placing orders, and tracking deliveries. This experiment aimed to ensure the system was intuitive and user-friendly.

**2. Order Management and Real-Time Updates**  
Customers placed orders, and staff updated statuses to check real-time updates. This tested the efficiency of web sockets in delivering instant updates to users.

**3. Performance and Load Testing**  
Simulated high traffic scenarios were used to evaluate the system’s ability to handle multiple users and orders simultaneously, ensuring scalability.

**4. Security Testing**  
Authentication and role-based access control (RBAC) were tested to confirm that users could only access features relevant to their roles. Security vulnerabilities were also checked.

# **RESULTS**

* The system allowed customers to browse menus, place orders, and track deliveries efficiently through a clean and responsive interface.
* Restaurant staff were able to manage orders in real-time, improving speed and accuracy in food preparation and delivery.
* The dynamic menu rendering ensured that only available items were shown, reducing order errors.
* Role-based access control worked effectively, with each user type accessing only relevant features.
* The system demonstrated stable performance under load, handling multiple users and orders without delays.
* Real-time communication between frontend and backend was successfully implemented using web sockets, providing instant order status updates.

# **CONCLUSION and FUTURE WORK**

* **Conclusion:**
* The system simplifies food ordering with an intuitive customer interface and efficient admin tools.
* Real-time updates and role-based access improve service speed and security.
* Modular architecture ensures maintainability and scalability.
* Technologies like Vue.js, Node.js, and MySQL provide a robust backend and responsive frontend.
* **Future Work:**
* Add **online payment gateway** for smoother transactions.
* Implement **customer feedback and ratings** for continuous service improvement.
* Integrate **data analytics** to monitor sales and popular items.
* Develop **mobile applications** for Android and iOS for wider accessibility.
* Use **AI-based suggestions** and **chatbots** to enhance user engagement and support.
* **Voice-based Quiz and Accessibility Options:** Enhance usability for visually impaired users with audio support and keyboard navigation.

##### **References**

1. Vue.js – Official Guide. <https://vuejs.org>
2. Node.js – Documentation. <https://nodejs.org>
3. Express.js – Web Framework. <https://expressjs.com>
4. MySQL – Reference Manual. <https://dev.mysql.com>
5. Sharma, R., & Singh, P. (2020). *Online Food Ordering System Using Web Technologies*. IJCA, 176(38), 1-5.