

*22<sup>nd</sup> May 2024*

## Online Food Ordering System

*Version 1.0*

***by NIKUL RAM***

*nikulram007@gmail.com - [www.nikulram.com](http://www.nikulram.com)*

*[www.github.com/nikulram](http://www.github.com/nikulram)*

*LinkedIn: [www.linkedin.com/in/nikul-ram/](http://www.linkedin.com/in/nikul-ram/)*

# Contents

<b>1</b>	<b>Part 1: Project Introduction and Rationale.....</b>	<b>4</b>
1.1	Overview .....	4
1.2	Objective .....	4
1.3	Technologies Used .....	4
1.4	Inspiration .....	4
1.5	Project Scope.....	4
<b>2</b>	<b>Part 2: Technical Architecture and System Components .....</b>	<b>5</b>
2.1	Overview .....	5
2.2	System Components.....	5
2.3	Data Flow Diagram .....	5
<b>3</b>	<b>Part 3: Development Phases, Challenges, and Solutions .....</b>	<b>6</b>
3.1	Project Phases.....	6
3.2	Challenges and Solutions .....	7
3.3	Development Tools and Technologies .....	7
<b>4</b>	<b>Part 4: Features and Functionalities .....</b>	<b>8</b>
4.1	User Features .....	8
4.2	Admin Features.....	8
4.3	Security Features.....	9
<b>5</b>	<b>Part 5: Challenges and Lessons Learned .....</b>	<b>10</b>
5.1	Integration of Components.....	10
5.2	Real-Time Updates .....	10
5.3	Cross-Browser Compatibility .....	10
5.4	User Feedback.....	10
5.5	Documentation .....	10
<b>6</b>	<b>Part 6: Detailed Explanation of Project Files .....</b>	<b>10</b>
6.1	Project Structure: .....	10
6.2	HTML Files.....	12
6.3	CSS File.....	13
6.4	JavaScript Files.....	13
6.5	PHP Files .....	14
6.6	SQL Schema.....	16
<b>7</b>	<b>Part 7: Visual Documentation .....</b>	<b>17</b>
7.1	Early Development Stages Visuals .....	17

7.2 Final User Interface.....	23
7.3 Admin Interface.....	29
7.4 Database Shema .....	32
<b>8 Part 8: Reflection and Future Prospects.....</b>	<b>32</b>
8.1 Future Enhancements/Ideas.....	32
8.2 Concluding Remarks.....	33
<b>9 Part 9: References.....</b>	<b>33</b>

# 1. Part 1: Project Introduction and Rationale

---

## 1.1 Overview

The Online Food Ordering System is designed to facilitate seamless interaction between customers and food vendors. This system allows customers to browse menus, add items to their cart, and place orders, while vendors can manage their menu and track orders efficiently. The system is built with a focus on user experience, ensuring ease of use and reliability (Chavan, Jadhav, Korade, & Teli, 2015).

## 1.2 Objective

The primary objective of the Online Food Ordering System is to provide a comprehensive platform that simplifies the process of ordering food online. Key objectives include:

- Enabling users to browse and order from a digital menu.
- Allowing administrators to manage menu items, user information, and orders.
- Ensuring secure and efficient transactions.

## 1.3 Technologies Used

- **Frontend:** HTML, CSS, JavaScript
- **Backend:** PHP
- **Database:** MySQL
- **Server Environment:** XAMPP
- **Version Control:** Git and GitHub

## 1.4 Inspiration

The inspiration behind this project is to address the growing demand for online food ordering services (Chavan, Jadhav, Korade, & Teli, 2015). By digitizing the ordering process, this system aims to enhance customer convenience, streamline vendor operations, and reduce the errors associated with traditional ordering methods.

## 1.5 Project Scope

The scope of this project includes:

- **User Interface:** Development of intuitive interfaces for customers and administrators.
- **Database Management:** Efficient handling of data related to users, orders, and menu items.
- **Security:** Implementing secure practices for data handling and transactions.
- **Order Processing:** Automating the order process from cart management to checkout and order history.

## 2 . Part 2: Technical Architecture and System Components

---

### 2.1 Overview

The Online Food Ordering System is structured around a modular architecture, which allows for scalability and ease of maintenance. The system comprises several key components that work together seamlessly to provide a cohesive user experience.

### 2.2 System Components

#### 1. User Interface:

- **Purpose:** To provide an interactive platform for users to navigate through the system.
- **Technologies Used:** HTML, CSS, JavaScript.

#### 2. Backend Logic:

- **Purpose:** To handle data processing, business logic, and server-side operations.
- **Technologies Used:** PHP.

#### 3. Database:

- **Purpose:** To store and manage data related to users, orders, and menu items.
- **Technologies Used:** MySQL.

#### 4. Admin Panel:

- **Purpose:** To allow administrators to manage menu items, track orders, and oversee user activity.
- **Technologies Used:** HTML, CSS, JavaScript, PHP.

#### 5. Order Processing Module:

- **Purpose:** To manage the lifecycle of an order from creation to completion.
- **Technologies Used:** PHP, MySQL.

### 2.3 Data Flow Diagram

Here is an overview of how data flows through the system:

- **User Interaction:** Users interact with the frontend to browse menus and place orders.
- **Backend Processing:** User actions trigger backend scripts that process data and interact with the database.
- **Database Operations:** Data is retrieved from or stored in the MySQL database (*Stobart & Vassileiou, 2004*).

- **Admin Management:** Administrators use the admin panel to update menu items, track orders, and view users.

## 3 . Part 3: Development Phases, Challenges, and Solutions

---

### 3.1 Project Phases

#### 1. Initial Planning and Feasibility Study:

- **Goal:** Define project scope and assess technical feasibility.
- **Activities:** Research, technology evaluation, initial setup of development environment.
- **Outcome:** Project plan and setup of XAMPP for local development (*Kumari & Nandal, 2017*).

#### 2. System Design and Architecture Planning:

- **Goal:** Create a detailed system design.
- **Activities:** Designing database schema, UI/UX design.
- **Outcome:** A system design ready for implementation.

#### 3. Implementation:

- **Goal:** Develop the system components.
- **Activities:** Coding frontend and backend, integrating database operations.
- **Outcome:** A functional prototype with essential features implemented.

#### 4. Testing and Optimization:

- **Goal:** Ensure system meets requirements.
- **Activities:** Testing individual components, performance tuning.
- **Outcome:** A refined system with optimized performance.

#### 5. Deployment:

- **Goal:** Prepare the system for deployment.
- **Activities:** Final testing, preparing documentation.
- **Outcome:** A ready-to-deploy system with comprehensive documentation.

## 3.2 Challenges and Solutions

### 1. Responsive Design:

- **Challenge:** Ensuring the UI is responsive across various devices.
- **Solution:** Implemented CSS media queries to handle different screen sizes (CSS, 2022).

### 2. Database Optimization:

- **Challenge:** Managing database operations efficiently.
- **Solution:** Normalized the database schema and optimized queries.

### 3. Secure Transactions:

- **Challenge:** Ensuring secure data handling and transactions.
- **Solution:** Used prepared statements to prevent SQL injection and implemented HTTPS.

### 4. User Authentication:

- **Challenge:** Implementing secure user authentication (*Bhanderi, 2021*).
- **Solution:** Used PHP sessions for managing user authentication and hashed passwords for security (*Bhanderi, 2021*).

### 5. Image Handling:

- **Challenge:** Handling dynamic image uploads and displays.
- **Solution:** Ensured proper validation of image URLs and provided a default placeholder image (Implemented in Menu and Admin-Menu).

## 3.3 Development Tools and Technologies

- **IDE:** Visual Studio Code.
- **Version Control:** Git.
- **Development Environment:** XAMPP.
- **Libraries:** jQuery for simplified JavaScript operations.
- **Testing Tools:** PHPUnit for backend testing, manual testing for frontend (*Zandstra, 2016*).

## 4 . Part 4: Features and Functionalities

---

### 4.1 User Features

#### 1. Browse Menu:

- **Description:** Users can browse through various menu items categorized neatly.
- **Implementation:** Menu items are fetched from the database and displayed dynamically.

#### 2. Add to Cart:

- **Description:** Users can add menu items to their cart for review before placing an order.
- **Implementation:** JavaScript functions handle cart operations and store items in local storage (*Flanagan, 2011*).

#### 3. Checkout:

- **Description:** Users can review their cart and proceed to checkout.
- **Implementation:** Checkout form validates user input and processes payment information securely.

#### 4. Order History:

- **Description:** Users can view their past orders.
- **Implementation:** Order history is fetched from the database and displayed on the user's profile.

### 4.2 Admin Features

#### 2. Manage Menu Items:

- **Description:** Admins can add, edit, and delete menu items.
- **Implementation:** Admin panel provides forms for CRUD operations on menu items.

#### 3. View Orders:

- **Description:** Admins can view all orders placed by users.
- **Implementation:** Orders are displayed in a tabular format with details like order ID, user, and status.



#### 4. Manage Users:

- **Description:** Admins can view user information and permissions.
- **Implementation:** Admin panel allows viewing user information.

#### 5. Analytics and Reports:

- **Description:** Admins can view analytics on sales and user activity.
- **Implementation:** Data is visualized using charts and tables for easy understanding.

### 4.3 Security Features

#### 1. Password Hashing:

- **Description:** Ensures user passwords are stored securely.
- **Implementation:** PHP's `password_hash()` function is used for hashing passwords before storing them in the database.

#### 2. Prepared Statements:

- **Description:** Prevents SQL injection attacks.
- **Implementation:** Used in all database queries involving user inputs.

#### 3. Session Management:

- **Description:** Securely manages user sessions.
- **Implementation:** PHP sessions are used to maintain user login states.

#### 4. Image Validation:

- **Description:** Validates image URLs to prevent broken images.
- **Implementation:** JavaScript functions handle image URL validation and provide a default placeholder if validation fails.

## 5 . Part 5: Challenges and Lessons Learned

---

### 5.1 Integration of Components

Integrating various components like the user interface, backend logic, and database operations was a complex task. Ensuring ideal data flow and interaction between these components required meticulous planning and testing.

### 5.2 Real-Time Updates

Implementing real-time updates for cart items and order status was challenging. JavaScript and AJAX were used extensively to provide a smooth user experience without page reloads (*Flanagan, 2011*).

### 5.3 Cross-Browser Compatibility

Ensuring the system worked consistently across different browsers was a significant challenge. Extensive testing and adjustments in CSS and JavaScript were made to achieve compatibility.

### 5.4 User Feedback

Incorporating user feedback during development helped identify usability issues and areas for improvement. Iterative testing and refinement based on feedback were crucial for enhancing user experience.

### 5.5 Documentation

Maintaining thorough documentation throughout the project was essential for clarity and future maintenance. This included in-line code comments, system architecture visuals, and user manuals.

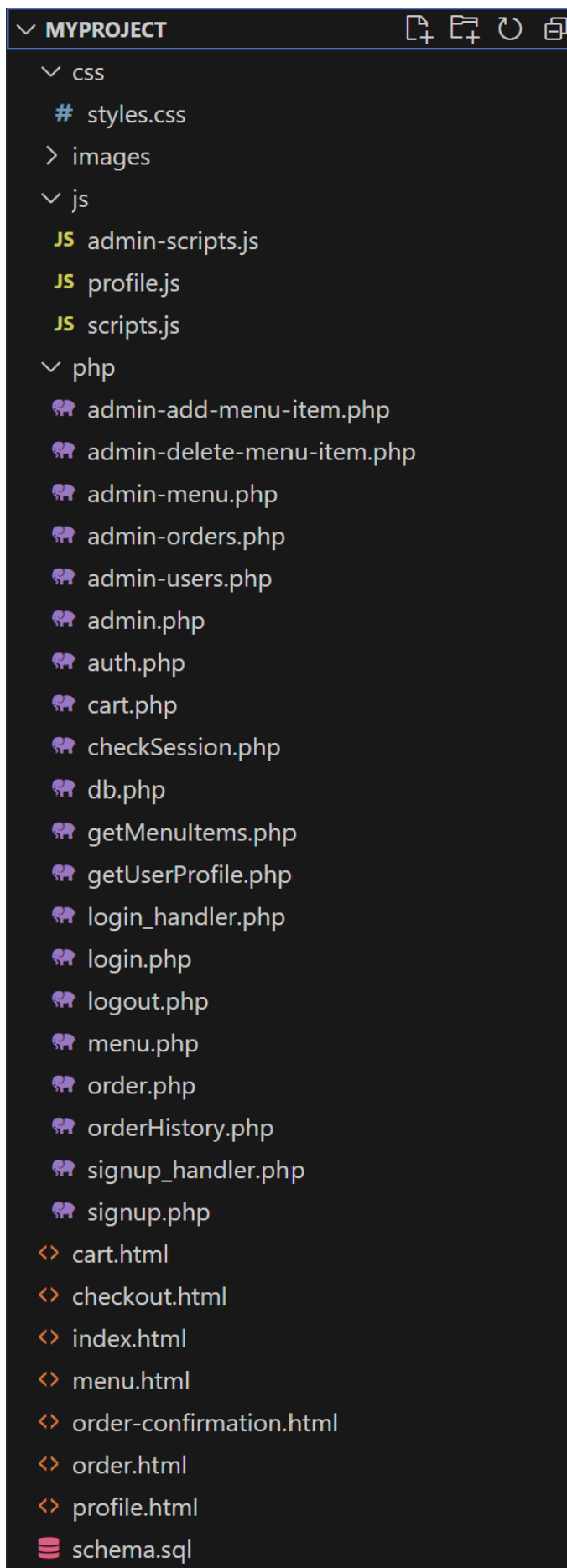
## 6 . Part 6: Detailed Explanation of Project Files

---

This section provides a comprehensive breakdown of each file and folder within the project. It aims to give an in-depth understanding of the structure and functionality of the codebase.

### 6.1 Project Structure:

The project is organized into various folders and files, each serving a specific purpose. Here is the overall structure:



**Figure1.0** – The Screenshot is displaying the final project structure/files that was made for the Online Food Ordering System v1.0. It is possible to see all the ordered files in consideration.

## 6.2 HTML Files

### index.html

- **Purpose:** The main landing page of the application.
- **Features:** Displays a welcome message and navigation links to the menu and cart. The page includes a welcoming message and essential links to guide users to the menu and their cart (*Hickson, 2014*).

### menu.html

- **Purpose:** Displays the menu items grouped by categories (*Hickson, 2014*).
- **Features:** Dynamically loads menu items using JavaScript (*Flanagan, 2011*). Users can view different food categories and their respective items, which are fetched from the database and displayed dynamically.

### cart.html

- **Purpose:** Shows the user's cart with items they intend to purchase.
- **Features:** Allows users to update item quantities or remove items from the cart. The page provides real-time updates of the cart contents and total price. It also ensures users cannot proceed to checkout without being logged in.

### checkout.html

- **Purpose:** The checkout page where users enter payment details.
- **Features:** Collects card information, CVV, and other details for processing orders. The form includes client-side validation to ensure the card details are correct before submission. If the user is not logged in, they are prompted to log in before they can complete the checkout process.

### order-confirmation.html

- **Purpose:** Displays a confirmation message after an order is placed.
- **Features:** Simple page showing order success and a link to order history. It confirms that the order was placed successfully and provides a link to view past orders.

### order.html

- **Purpose:** Displays the user's order history.
- **Features:** Dynamically loads past orders for the logged-in user. Users can see details of their previous orders, including order IDs, statuses, and dates.

### profile.html

- **Purpose:** Shows the user's profile information.
- **Features:** Displays username and email. It fetches and shows the current logged-in user's profile details.

## 6.3 CSS File

### styles.css

- **Purpose:** Provides styling for the entire application (*Peterson, 2014*).
- **Features:** Includes general styles, form styles, table styles, and responsive design adjustments. The file ensures a consistent look and feel across the application, with a dark theme and responsive design for different screen sizes (*CSS, 2022*).

## 6.4 JavaScript Files

### admin-scripts.js

- **Purpose:** Manages admin functionalities like adding and deleting menu items.
- **Features:** Handles form submissions and dynamically updates the admin menu table. It fetches menu items from the server and updates the admin interface accordingly. It also provides validation for adding new items and ensures smooth CRUD operations.

### profile.js

- **Purpose:** Loads user profile data.
- **Features:** Fetches and displays the logged-in user's profile information. It ensures that the profile information is displayed correctly and is updated in real-time if changes occur.

### scripts.js

- **Purpose:** Manages core functionalities like loading menu items, cart operations, and handling order placements.
- **Features:** Handles adding to cart, removing from cart, updating cart quantities, and placing orders. It includes client-side validation for cart and checkout processes, ensures secure transactions, and provides a seamless user experience by dynamically updating the UI without page reloads.

## 6.5 PHP Files

### admin-add-menu-item.php

- **Purpose:** Adds new menu items to the database.
- **Features:** Validates and inserts data into the menu\_items table. It ensures that only authenticated and authorized admin users can add new items and that the data is sanitized before insertion (McLaughlin, 2012).

### admin-delete-menu-item.php

- **Purpose:** Deletes menu items from the database.
- **Features:** Ensures associated records in order\_items are also deleted to maintain data integrity. It uses transactions to ensure that both the menu item and related order items are deleted atomically (McLaughlin, 2012).

### admin-menu.php

- **Purpose:** Displays the admin menu page.
- **Features:** Fetches and displays all menu items, grouped by category, in a table format. It allows the admin to view, add, and delete menu items (McLaughlin, 2012).

### admin-orders.php

- **Purpose:** Displays all orders for admin review.
- **Features:** Fetches and displays orders including details like username, total price, status, and order date. It helps admins track and manage all orders placed by users.

### admin-users.php

- **Purpose:** Displays all users for admin management.
- **Features:** Fetches and displays user details including username, email, admin status, and creation date. Admins can view and manage user accounts, including setting admin privileges (McLaughlin, 2012).

### admin.php

- **Purpose:** Provides an entry point for the admin panel.
- **Features:** Contains navigation to various admin functionalities. It checks if the user is an admin before granting access to the admin panel (McLaughlin, 2012).

### auth.php

- **Purpose:** Handles user authentication for login and signup.
- **Features:** Validates and processes user credentials. It manages session creation and ensures secure handling of user credentials.

### cart.php

- **Purpose:** Manages cart operations including creating orders.
- **Features:** Processes cart data and inserts order and order items into the database. It validates cart contents and user sessions before processing orders.

### checkSession.php

- **Purpose:** Checks if a user is logged in.
- **Features:** Returns session status in JSON format. It helps in managing frontend behaviors based on user login status.

### db.php

- **Purpose:** Establishes a database connection.
- **Features:** Connects to MySQL database using **mysqli**. It handles connection errors and sets the character set for the connection (McLaughlin, 2012).

### getMenuItems.php

- **Purpose:** Fetches all menu items from the database.
- **Features:** Returns menu items in JSON format. It groups items by category and provides them to the frontend for dynamic display.

### getUserProfile.php

- **Purpose:** Fetches the logged-in user's profile information.
- **Features:** Returns username and email in JSON format. It ensures secure access to user profile data.

### login\_handler.php

- **Purpose:** Processes user login.
- **Features:** Validates user credentials and sets session variables. It checks for correct username and password, sets session variables, and redirects users based on their role (admin or regular user).

### login.php

- **Purpose:** Provides the login interface.
- **Features:** Displays a form for username and password input. It includes feedback for incorrect login attempts.

### logout.php

- **Purpose:** Logs out the user.
- **Features:** Destroys the session and redirects to the homepage. It ensures a secure logout by clearing session data.

### menu.php

- **Purpose:** Manages menu operations.
- **Features:** Handles fetching and adding menu items via AJAX. It validates input data and manages menu item operations securely.

### order.php

- **Purpose:** Processes orders.

- **Features:** Handles order creation and order item insertion. It validates order data, processes payments, and updates the database with new order information.

#### orderHistory.php

- **Purpose:** Fetches order history for the logged-in user.
- **Features:** Returns order history in JSON format. It ensures secure access to user-specific order history.

#### signup\_handler.php

- **Purpose:** Processes user signup.
- **Features:** Validates and inserts new user data into the database. It checks for existing usernames/emails and securely hashes passwords before storing them.

#### signup.php

- **Purpose:** Provides the signup interface.
- **Features:** Displays a form for username, email, and password input. It includes feedback for signup errors and ensures all required fields are filled.

## 6.6 SQL Schema

#### schema.sql

- **Purpose:** Defines the database schema for the project.
- **Features:** Contains SQL commands to create tables and insert initial data. It sets up the structure for **users**, **menu\_items**, **orders**, and **order\_items tables**, including relationships and constraints.

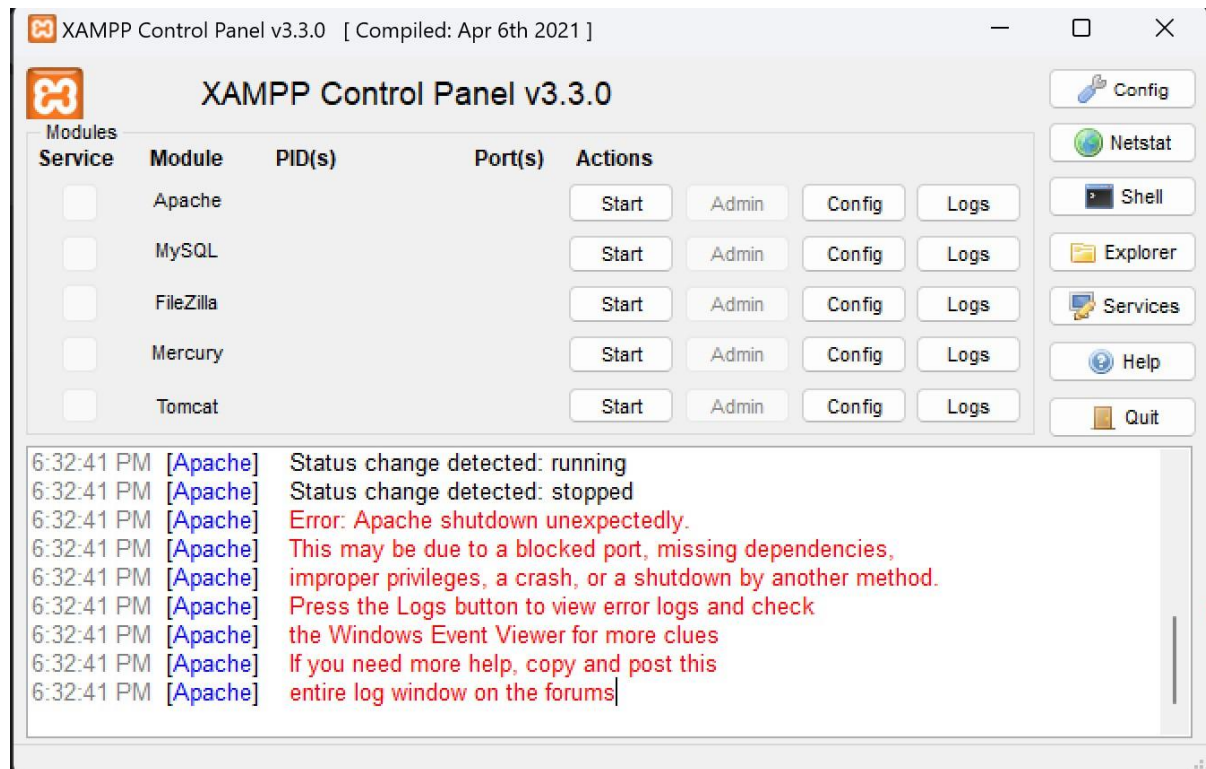


## 7. Part 7: Visual Documentation

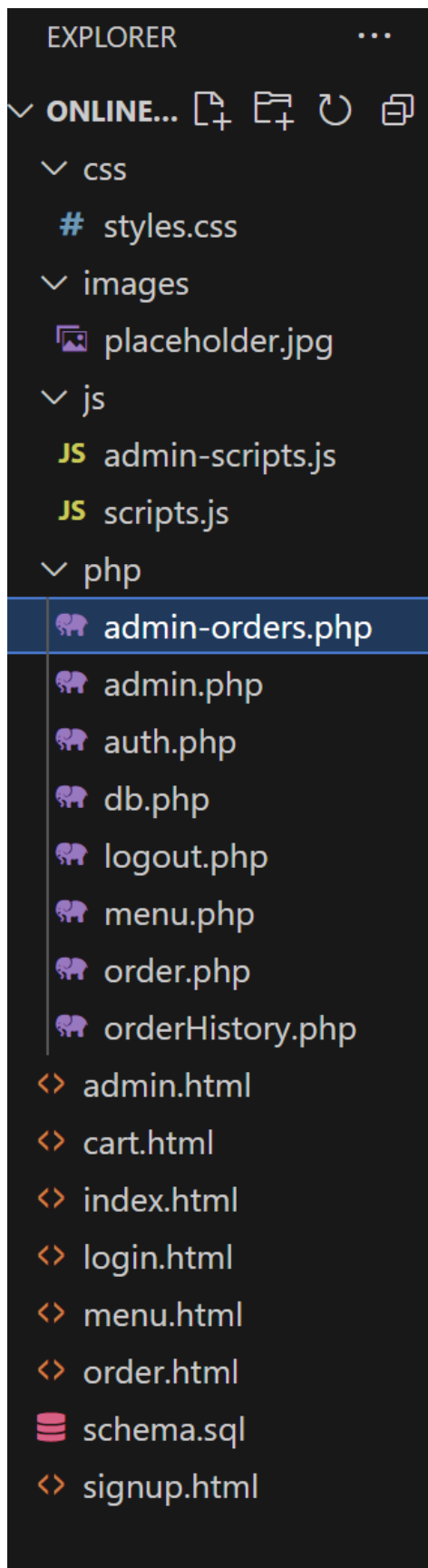
This section includes visual representations of various stages and components of the project. Screenshots help in understanding the structure, interface, and functionality of the system.

### 7.1 Early Development Stages Visuals

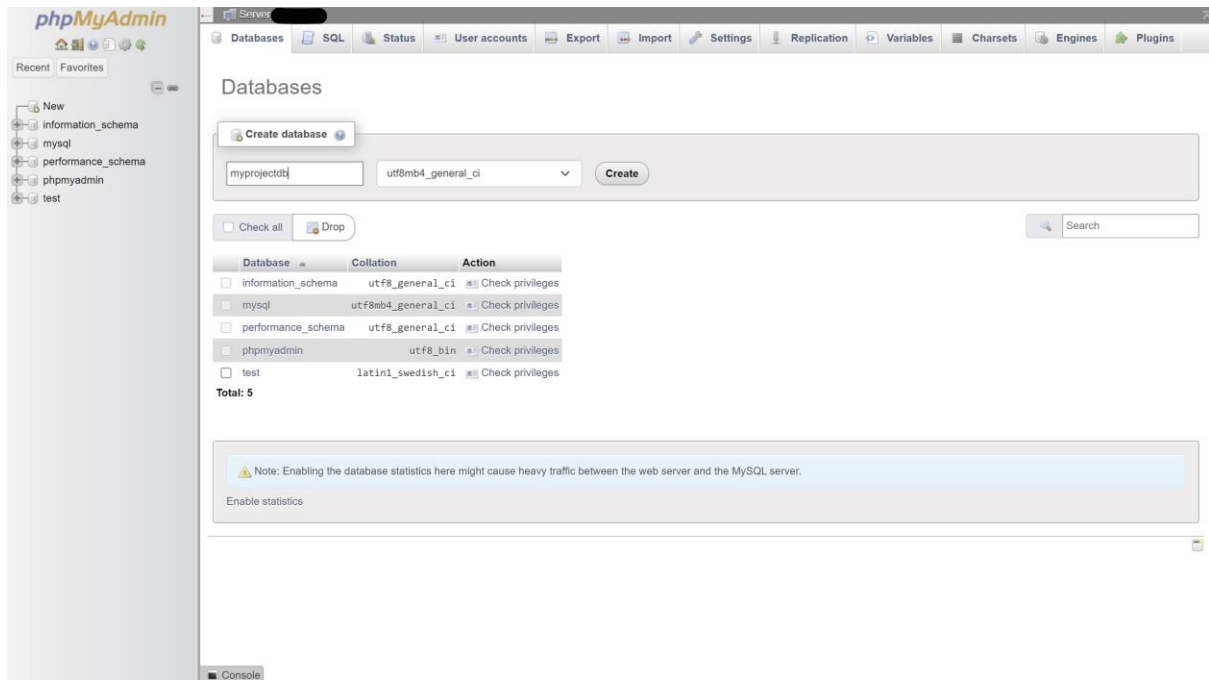
- *Screenshots of Early Development phase*




























































**Figure2.0** – This image shows that the XAMPP Control Panel was not starting Apache correctly, even after changing CONF Files and other methods to fix. The error message was resolved with reinstalling and making sure no other services was using the same port as of Apache and had to run as an administrator which fixed the issue (Kumari & Nandal, 2017).



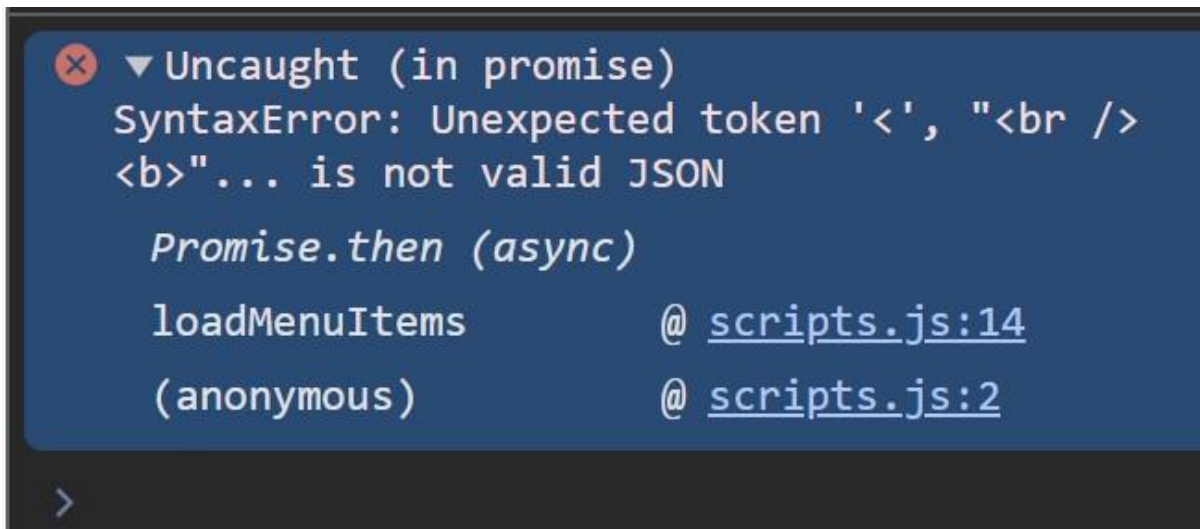
**Figure2.1** – Early stage of files and project structure when it was first started out with the project.



**Figure2.2** – Creation of database for the project. It is possible to see in the above visual the database name that was selected for the project known as myprojectdb.

←T→				id	user_id	total_price	status
<input type="checkbox"/>	 Edit	 Copy	 Delete	1	1	15.99	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	2	1	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	3	2	30.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	4	2	0.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	5	2	20.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	6	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	7	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	8	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	9	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	10	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	11	2	200.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	12	2	0.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	13	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	14	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	15	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	16	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	17	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	18	2	10.00	Pending
<input type="checkbox"/>	 Edit	 Copy	 Delete	19	2	10.00	Pending

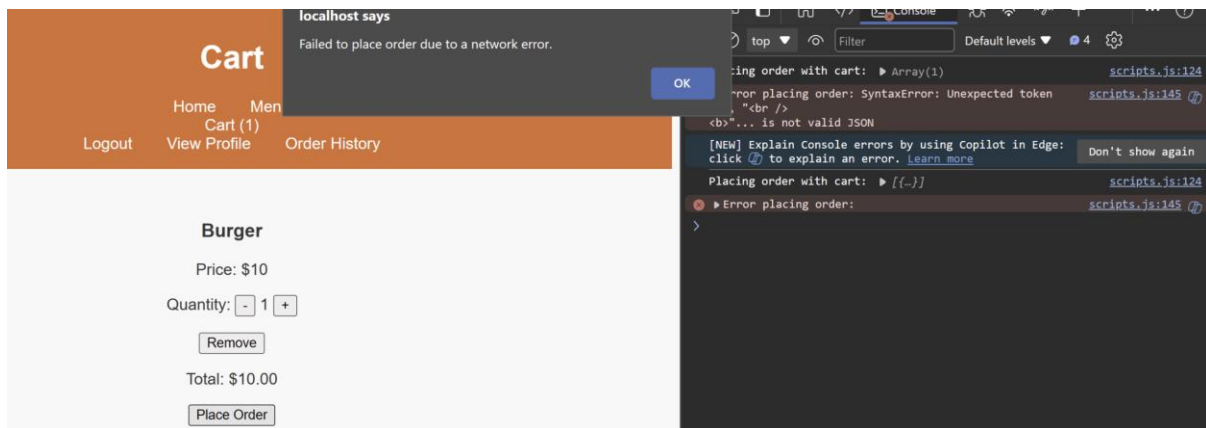
**Figure2.3** – Image showing the early stage of testing the orders and checkout section of the project in the database. Few errors came across and the issues were resolved: where the orders were being processed and sent to the database, but the project system was not registering the transaction which resulted in showing the status being “pending orders” rather than “completed orders”. This was then successfully resolved and made the final version of the project work.






**Figure3.0** – Image shows, error in scripts.js while testing, which is showed as a Syntax Error.



**Figure3.1** – Errors were encountered when trying to import the schema.sql file into the database. This happened because the schema.sql file was updated as the project progressed. As a result, MySQL threw a duplication error when it was attempted to import the updated schema.sql file (Steinhoff, n.d.).



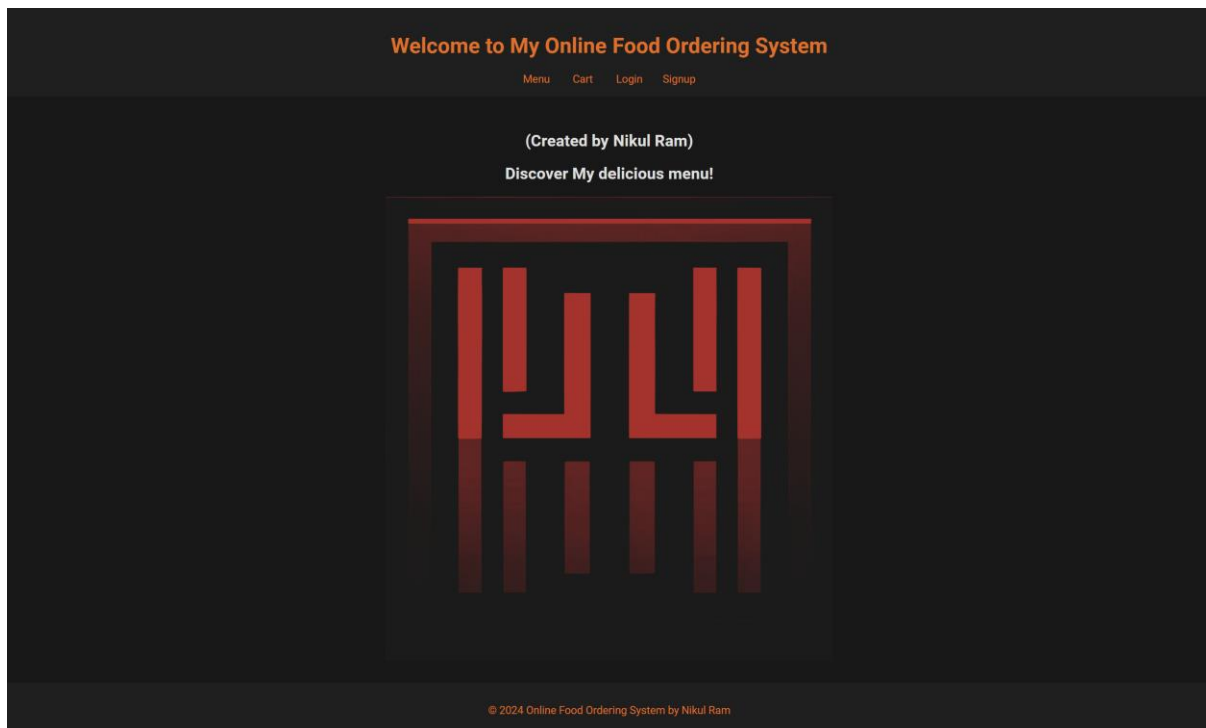
**Figure3.2** – Orders not going through error which led to update cart, checkout and order-confirmation which fixed the issue. It is possible to see as said in Figure2.3 which is the part of the error that was encountered.

Category	Name	Description	Price	Image	Actions
<b>Beverages</b>					
Beverages	Summer Drink	A flavorful mix fruit drink	1.99	 <a href="http://yourwebsite.com/myproject/images/mixdrink.jpg">http://yourwebsite.com/myproject/images/mixdrink.jpg</a>	<a href="#">Delete</a>
<b>Main Course</b>					
Main Course	Burger	A Delicious Burger	5.99	 <a href="http://yourwebsite.com/myproject/images/burger.jpg">http://yourwebsite.com/myproject/images/burger.jpg</a>	<a href="#">Delete</a>
<b>Vegan</b>					
Vegan	Vegan Burger	A delicious veggie burger	3.99	 <a href="http://yourwebsite.com/myproject/images/veggieburger.jpg">http://yourwebsite.com/myproject/images/veggieburger.jpg</a>	<a href="#">Delete</a>

**Figure3.3** – The above screenshot displays the Admin-Menu, where the images were not showing, and the relative process in order to debug to fix the issue.

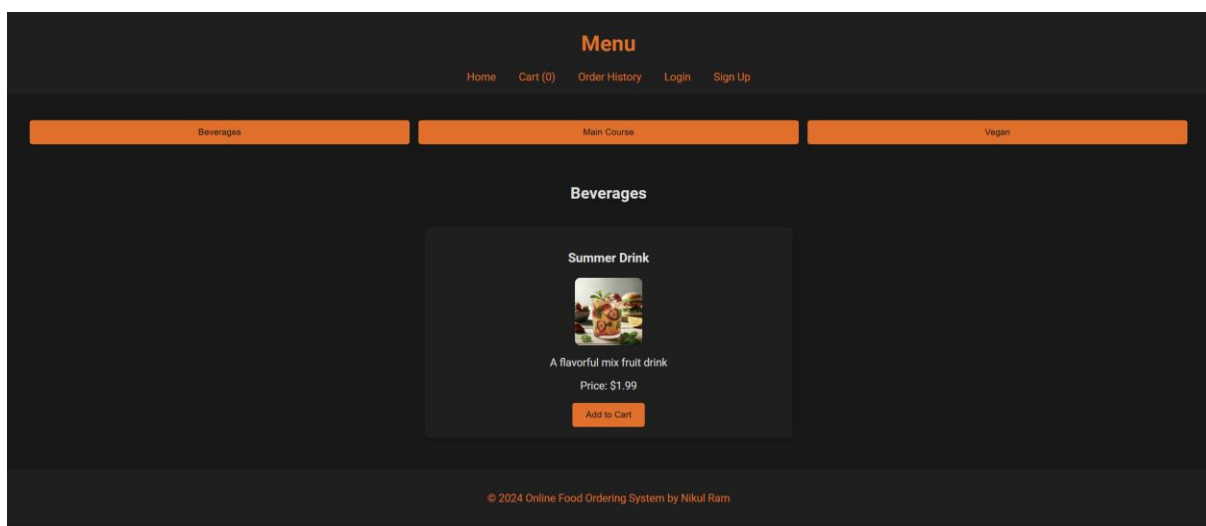
## 7.2 Final User Interface

- **Landing Page:** The main page with a welcome message and navigation links.



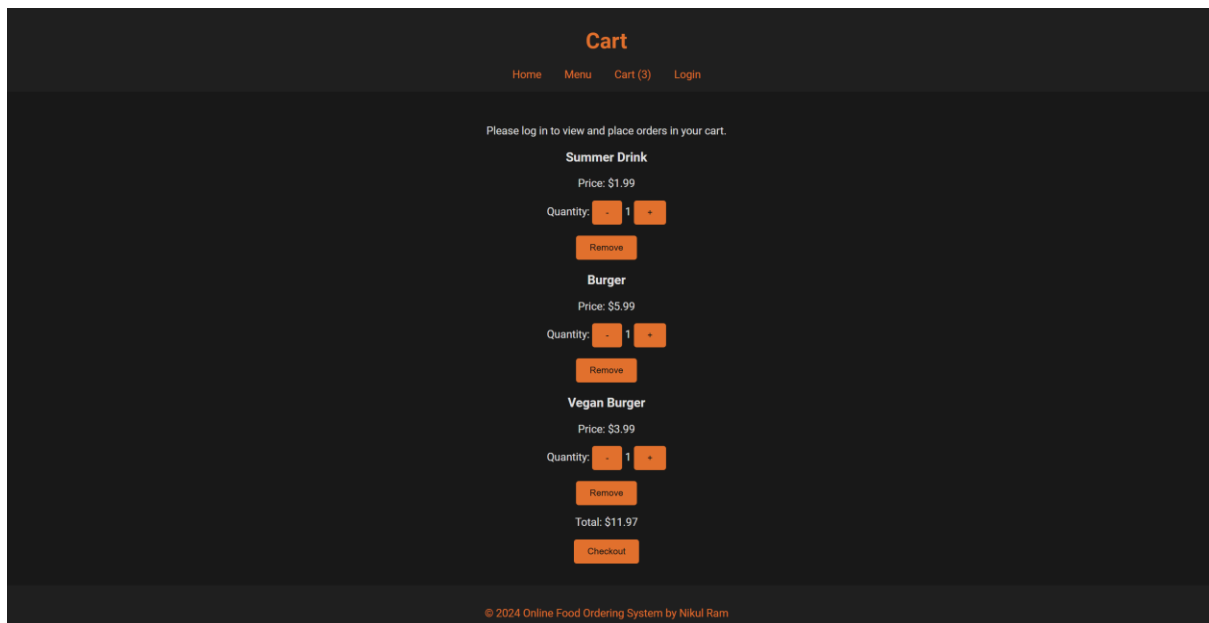
**Figure4.0** – Index/Main Page.

- **Menu Page:** Displays menu items categorized into sections.



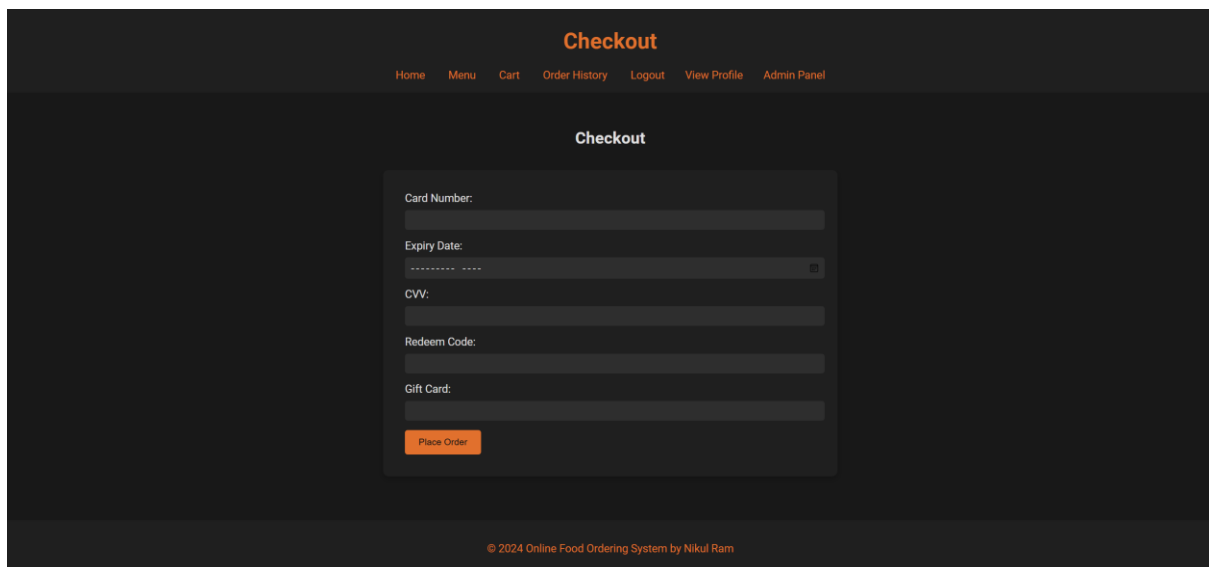
**Figure4.1** – Menu Page.

- **Cart Page:** Shows items added to the cart with options to update or remove items.



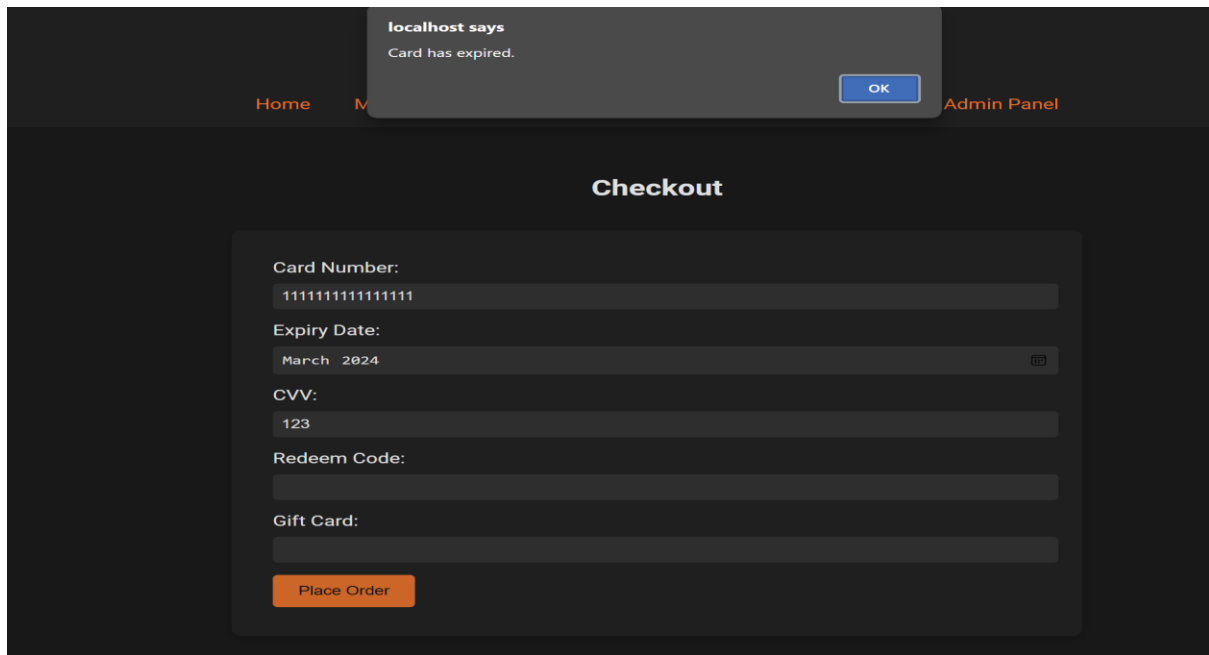
**Figure4.2** – Cart page without logging in.

- **Checkout Page:** Form to enter payment details and place an order.

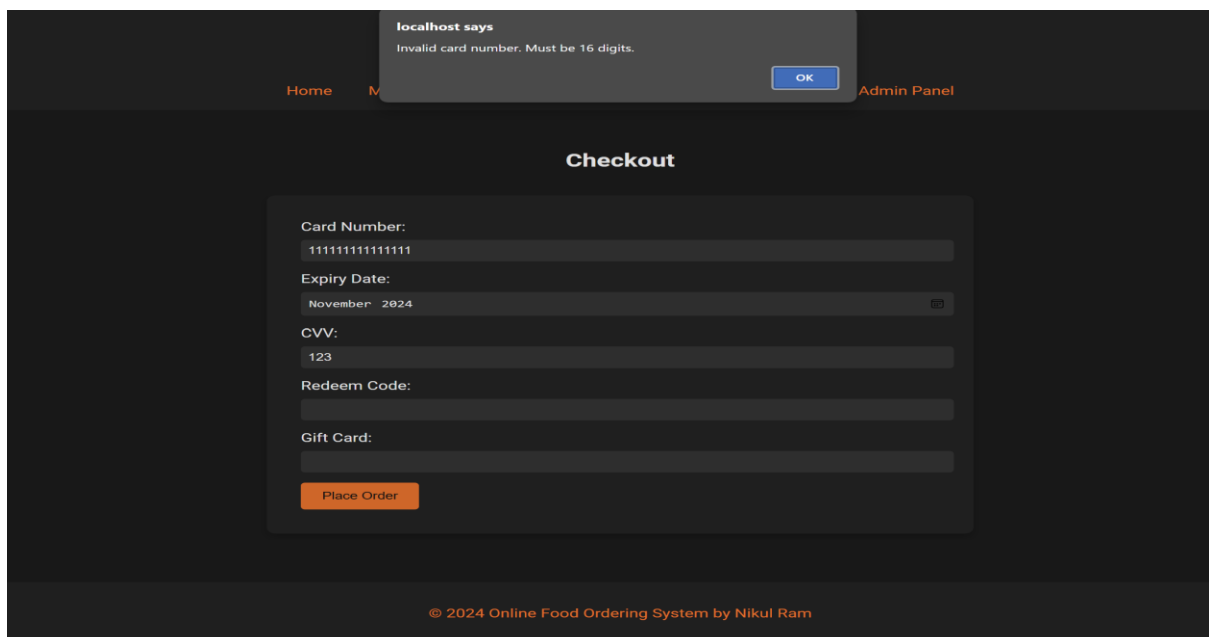


**Figure4.3** – Checkout Page.





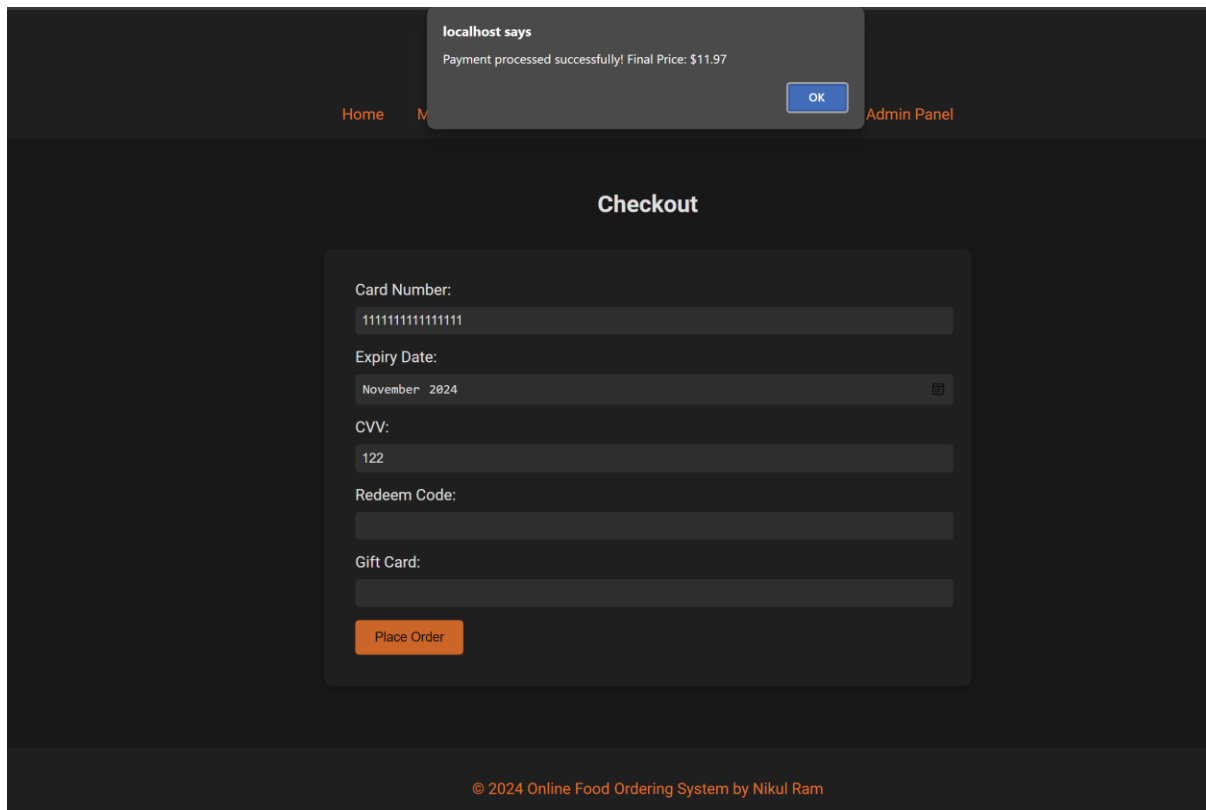
**Figure4.4** – Checkout Page with card expired message.



**Figure4.5** – Checkout Page with invalid card/card number message.

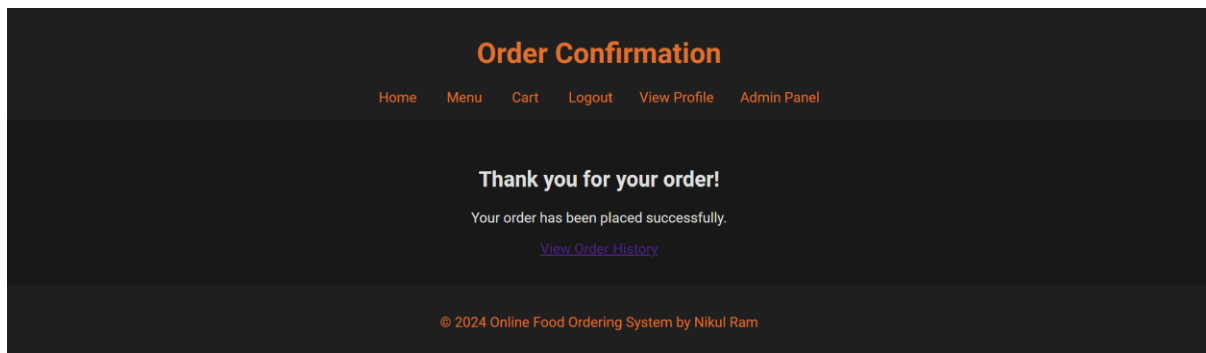
The screenshot shows a web application interface for a checkout process. At the top, there is a navigation bar with links for 'Home', 'Menu', and 'Admin Panel'. A modal message box is displayed, stating 'localhost says Invalid CVV. Must be 3 digits.' with an 'OK' button. The main content area is titled 'Checkout' and contains a form with the following fields: 'Card Number' (filled with '1111111111111111'), 'Expiry Date' (set to 'November 2024'), 'CVV' (filled with '12'), 'Redeem Code', and 'Gift Card'. A 'Place Order' button is located at the bottom of the form. The footer of the page reads '© 2024 Online Food Ordering System by Nikul Ram'.

**Figure4.6** – Checkout Page with Invalid CVV message.



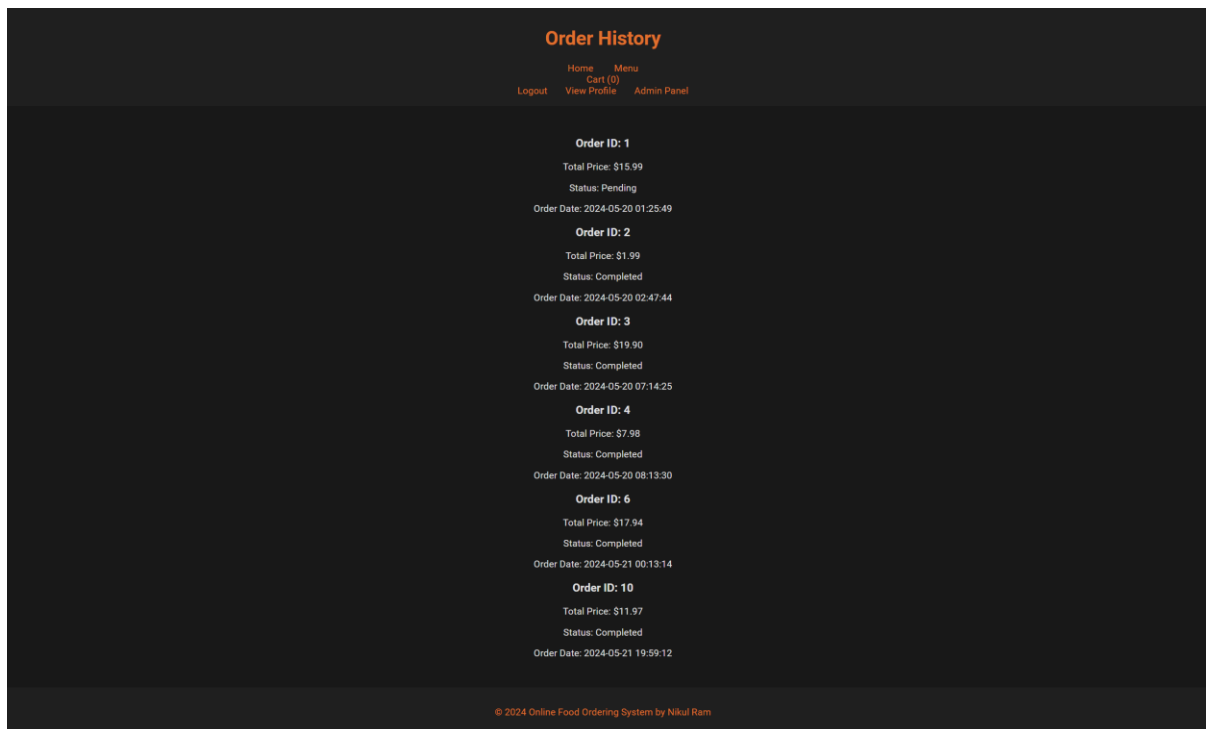
**Figure4.7** – Checkout Page with payment successful message.

- **Order Confirmation Page:** Confirms the order has been placed successfully.



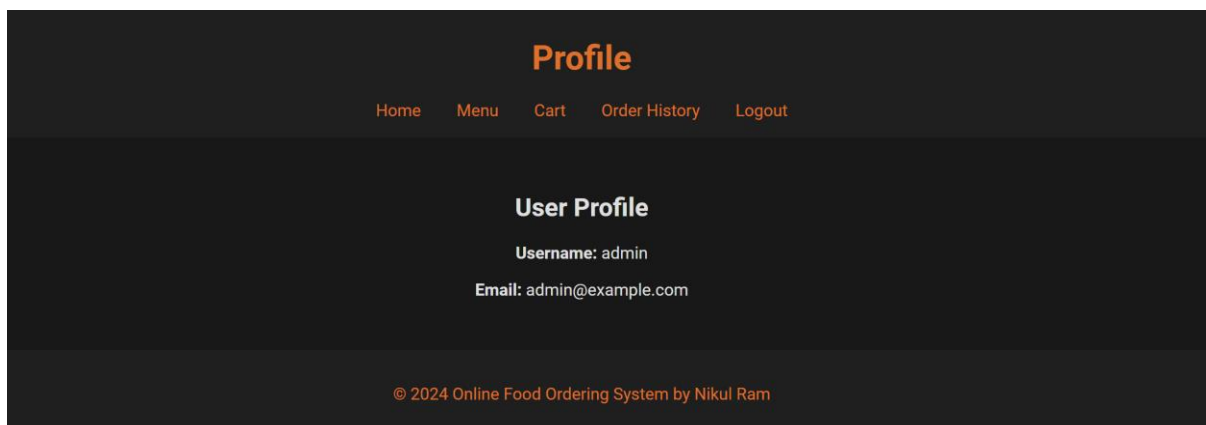
**Figure4.8** – Order Confirmation page which comes only after successful checkout.

- **Order History Page:** Lists past orders made by the user.



**Figure4.9** – Order History Page of the user.

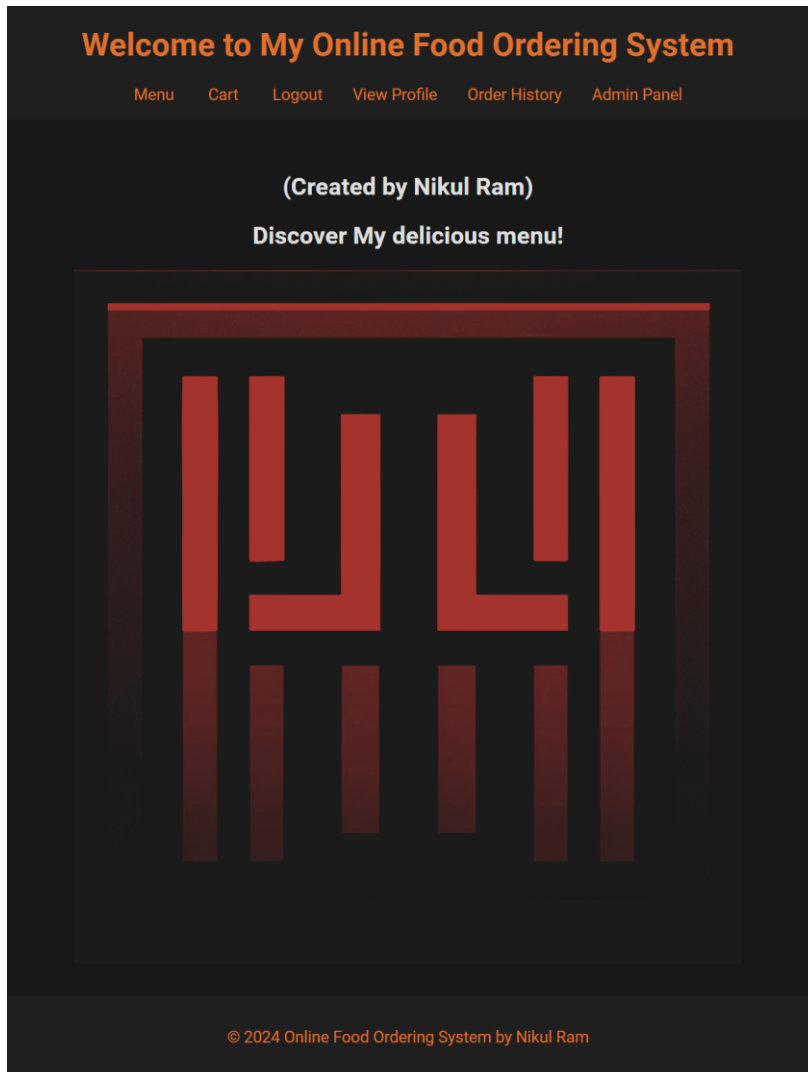
- **Profile Page:** Displays user profile information.



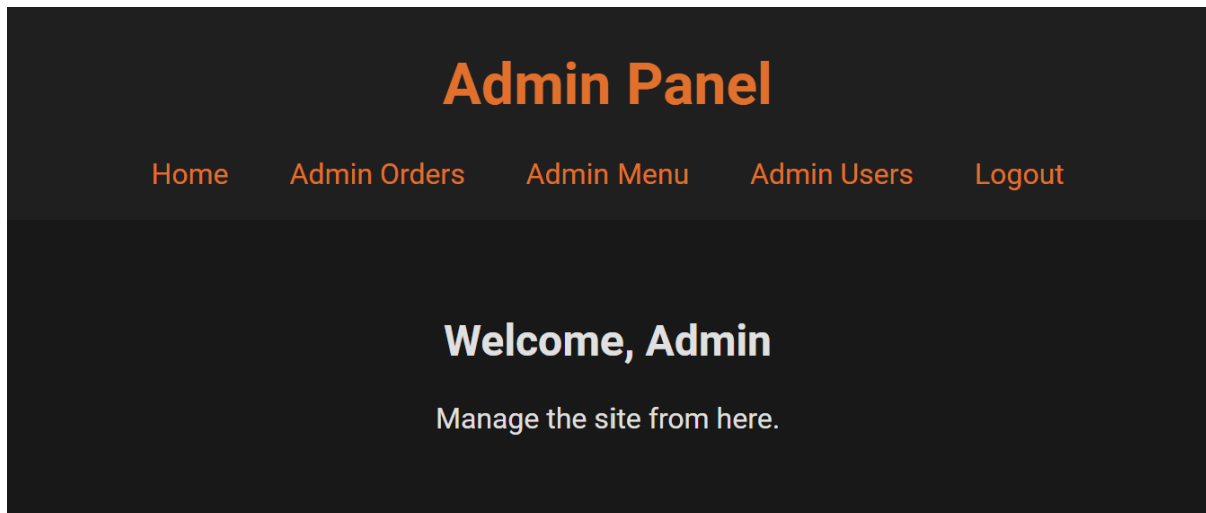
**Figure5.0** – Profile page of the user.

## 7.3 Admin Interface

- **Admin Panel:** Main dashboard for the admin.

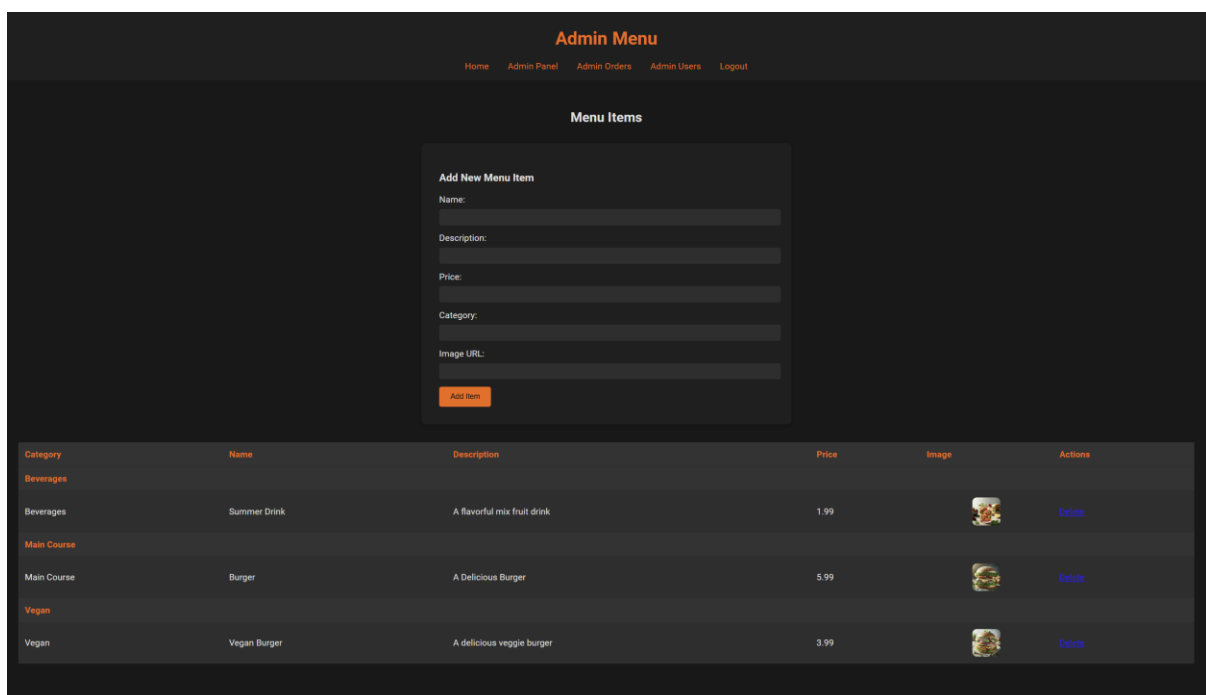


**Figure6.0** – Admin Index Page.



**Figure6.1** – Admin Panel Page.

- **Manage Menu:** Interface for adding, updating, and deleting menu items.



**Figure6.2** – Admin-Menu Page.

- **Manage Orders:** View all orders placed by users.

Admin Orders				
Home	Admin Panel	Admin Menu	Admin Users	Logout
Orders List				
Order ID	Username	Total Price	Status	Order Date
1	admin	15.99	Pending	2024-05-20 01:25:49
2	admin	1.99	Completed	2024-05-20 02:47:44
3	admin	19.90	Completed	2024-05-20 07:14:25
4	admin	7.98	Completed	2024-05-20 08:13:30
5	123	2.00	Completed	2024-05-20 23:36:17
6	admin	17.94	Completed	2024-05-21 00:13:14
7	123	17.96	Completed	2024-05-21 03:10:42
8	123	1.99	Completed	2024-05-21 03:30:59
9	jalpa11	17.94	Completed	2024-05-21 07:16:29
10	admin	11.97	Completed	2024-05-21 19:59:12

**Figure6.3** – Admin-Orders Page.

- **Manage Users:** View user details and permissions.

Admin Users				
Home	Admin Panel	Admin Orders	Admin Menu	Logout
Users List				
User ID	Username	Email	Admin Status	Created At
1	admin	admin@example.com	Yes	2024-05-20 01:25:49
2	123	hhtest@example.com	No	2024-05-20 02:58:23

**Figure6.4** – Admin-Users Page.

## 7.4 Database Shema

- *Visual representation of the database schema, showing tables, relationships, and key fields.*

Table	Action	Rows	Type	Collation	Size	Overhead
menu_items	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 K B	-
orders	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_general_ci	32.0 K B	-
order_items	Browse Structure Search Insert Empty Drop	17	InnoDB	utf8mb4_general_ci	48.0 K B	-
users	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 K B	-
4 tables	Sum	33	InnoDB	utf8mb4_general_ci	112.0 K B	0 B

**Figure 7.0** – Final database schema(schema.sql) information in order to monitor the food ordering system.

## 8. Part 8: Reflection and Future Prospects

### 8.1 Future Enhancements/Ideas

#### 1. Mobile App Integration:

- **Description:** Developing a mobile app for the system.
- **Benefits:** Increases accessibility and convenience for users.

#### 2. Automated Email Notifications:

- **Description:** Sending automated emails for order confirmations and updates.
- **Benefits:** Enhances user engagement and communication.

#### 3. Enhanced Security Measures:

- **Description:** Implementing two-factor authentication and other security features.
- **Benefits:** Increases user data protection and system security.



## 8.2 Concluding Remarks

The Online Food Ordering System project is an amazing project that exemplifies how different technologies may be seamlessly integrated to provide a smooth user experience (*Chavan, Jadhav, Korade, & Teli, 2015*). Every stage of the process, from early planning to implementation and testing, has helped to create a solid and trustworthy system. The difficulties encountered throughout development offered priceless teaching moments, and the answers put in place made sure the project would succeed. For example, I was solving this issue that admin or other links if pasted should not lead to open the specific page which users are not supposed to see by pasting the link or accessing somehow. Like the Admin Panel during development phase could be visited with link access without logging in as an Admin which was later fixed that people can't access without admin log in. In conclusion, this material provides a thorough overview of the project's progress, emphasizing its accomplishments and untapped potential.

## 9. Part 9: References

---

Bhanderi, Dixita Dinesh. "Enhanced Two Factor Authentication with MD5." PhD diss., California State University, Sacramento, 2021.

Chavan, Varsha, Priya Jadhav, Snehal Korade, and Priyanka Teli. "Implementing Customizable Online Food Ordering System Using Web Based Application." *International Journal of Innovative Science, Engineering & Technology* 2, no. 4 (2015): 722-727.

CSS, Cascading Style Sheets. "Cascading Style Sheets." Accessed April 27, 2022. <https://developer.mozilla.org/en-US/docs/Web/CSS>.

Flanagan, David. *JavaScript: The Definitive Guide: Activate Your Web Pages*. Sebastopol, CA: O'Reilly Media, Inc., 2011.

Hickson, Anthony. *HTML, The Living Standard*. Raleigh: Lulu.com, 2014.

Kumari, Punam, and Rainu Nandal. "A Research Paper on Website Development Optimization Using Xampp/PHP." *International Journal of Advanced Research in Computer Science* 8, no. 5 (2017).

McLaughlin, Brett. *PHP & MySQL: The Missing Manual*. Sebastopol, CA: O'Reilly Media, Inc., 2012.

Peterson, Clarissa. *Learning Responsive Web Design: A Beginner's Guide*. Sebastopol, CA: O'Reilly Media, Inc., 2014.

Stobart, Simon, and Mike Vassileiou. *PHP and MySQL Manual: Simple, Yet Powerful Web Programming*. New York: Springer Science & Business Media, 2004.

Steinhoff, Sascha. "Evaluation of MySQL 8.0 Spatial Data Features."

Zandstra, Matt. "Testing with PHPUnit." In *PHP Objects, Patterns, and Practice*, 435-464. 2016.