**# Restaurant Reservation Report**

**## 1. Overview**

The restaurant reservation system is a PHP-based web application that allows restaurant staff to manage customers, reservations, and customer preferences. The system consists of two main components:

1. `RestaurantDatabase.php`: Handles database operations

2. `index.php`: Manages the user interface and request handling

The system uses MySQL as its database backend and follows a simple MVC (Model-View-Controller) pattern, where the RestaurantDatabase class serves as the Model, the index.php file contains both the Controller and View components.

**## 2. RestaurantDatabase.php**

**### 2.1 Purpose**

The RestaurantDatabase class encapsulates all database operations, providing a clean interface for interacting with the MySQL database.

**#### 2.2.1 addSpecialRequest($reservationId, $requests)**

- Functionality: Adds a new special request to an existing reservation.

- Implementation: Uses an UPDATE query to append the new request to the existing specialRequests field.

- Usage: Called when a user wants to add a special request to a reservation.

**#### 2.2.2 findReservations($customerId)**

- Functionality: Retrieves all reservations for a specific customer.

- Implementation: Uses a SELECT query to fetch reservations associated with the given customerId.

- Usage: Used to display a customer's reservation history or to check for existing reservations.

**#### 2.2.3 deleteReservation($reservationId)**

- Functionality: Deletes a specific reservation from the database.

- Implementation: Uses a DELETE query to remove the reservation with the given reservationId.

- Usage: Called when a user wants to cancel a reservation.

**#### 2.2.4 searchPreferences($customerId)**

- Functionality: Retrieves the preferences for a specific customer.

- Implementation: Uses a SELECT query to fetch preferences associated with the given customerId.

- Usage: Used to display or consider a customer's preferences when making reservations.

**### 2.3 Additional Methods**

- `addCustomer`: Adds a new customer to the database.

- `addReservation`: Creates a new reservation in the database.

- `getAllReservationsWithCustomers`: Retrieves all reservations with associated customer information.

- `getAllCustomers`: Fetches all customers from the database.

- `getCustomerById`: Retrieves a specific customer's information.

**## 3. index.php**

**### 3.1 Purpose**

The index.php file serves as both the controller and view for the web application. It handles user requests, interacts with the RestaurantDatabase class, and generates HTML output.

**### 3.2 Key Components**

**#### 3.2.1 RestaurantPortal Class**

- Manages the overall functionality of the web application.

- Handles different actions based on user requests (e.g., viewing reservations, adding customers, etc.).

**#### 3.2.2 handleRequest() Method**

- Acts as the main entry point for processing user requests.

- Uses a switch statement to route requests to appropriate methods based on the 'action' parameter.

**#### 3.2.3 View Methods**

- `reservationsPage()`: Displays the reservations list and provides options to add new reservations.

- `customersPage()`: Shows the customer list and allows adding new customers.

- `addReservationPage()`: Presents a form for adding new reservations.

- `addCustomerPage()`: Displays a form for adding new customers.

- `viewCustomerReservations()`: Shows reservations for a specific customer.

**#### 3.2.4 Action Methods**

- `addReservation()`: Processes the form submission for adding a new reservation.

- `addCustomer()`: Handles the addition of a new customer to the system.

- `deleteCustomer()`: Manages the deletion of a customer and their associated reservations.

- `addSpecialRequest()`: Processes the addition of special requests to existing reservations.

**## 4. System Capabilities**

The restaurant reservation system provides the following key functionalities:

1. Customer Management:

1. Add new customers

2. View customer list

3. Delete customers (including their reservations)

2. Reservation Management:

1. Create new reservations

2. View all reservations

3. View reservations for a specific customer

4. Delete reservations

5. Add special requests to existing reservations

3. Customer Preferences:

1. Store and retrieve customer preferences (e.g., favorite table, dietary restrictions)

4. User Interface:

1. Simple and intuitive web interface for staff to manage reservations and customers

2. Responsive design for use on various devices

**## 5. Conclusion**

The restaurant reservation system provides a comprehensive solution for managing customers, reservations, and preferences. By separating database operations (RestaurantDatabase.php) from the user interface and request handling (index.php), the system maintains a clean and modular structure.

The system can be further improved by adding features such as:

- User authentication for staff members

- Advanced search and filtering options for reservations and customers

- Reporting and analytics capabilities

- Integration with other restaurant management systems (e.g., table management, point of sale)

Overall, the current implementation provides a solid foundation for a restaurant reservation system that can be easily extended and maintained.