

Name: Edmond Agyemang

Date: Dec 10, 2024

CIS 344 Final Project Report: restaurant_reservations

localhost / localhost / restaurant_reservations | phpMyAdmin 5.2.1

Server: localhost » Database: restaurant_reservations

Structure SQL Search Query Export Import Operations Privileges Routines Events More

Filters

Containing the word:

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> customers	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	–
<input type="checkbox"/> diningPreferences	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	–
<input type="checkbox"/> reservations	★ Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	32.0 KiB	–
3 tables	Sum	15	InnoDB	utf8mb4_general_ci	80.0 KiB	0 B

Check all With selected:

Print Data dictionary

Create new table

Table name Number of columns Create

Beginning with the MySQL component of the project, I established the ‘restaurant_reservations’ database and constructed three tables named ‘customers’, ‘diningpreferences’, and ‘reservations’. Each table was populated with essential attributes and corresponding foreign keys. Subsequently, I inserted test entries into the all the tables to verify its initialization and ensure its visibility, as demonstrated here.

SELECT * From ‘Customers’

✓ Showing rows 0 - 6 (7 total, Query took 0.0001 seconds.)

```
SELECT * FROM `customers`
```

☐ Profiling [[Edit inline](#)] [[Edit](#)] [[Explain SQL](#)] [[Create PHP code](#)] [[Refresh](#)]

☐ Show all | Number of rows: 25 | Filter rows: Sort by key: None

Extra options

	customerId	customerName	contactInfo
<input type="checkbox"/> Edit Copy Delete	1	Baa kofi	baakofi@example.com
<input type="checkbox"/> Edit Copy Delete	2	Janet Amoa	janetamoa@example.com
<input type="checkbox"/> Edit Copy Delete	3	Alke Johan	alkej@example.com
<input type="checkbox"/> Edit Copy Delete	4	Bobby Brown	bobbybrown@example.com
<input type="checkbox"/> Edit Copy Delete	5	Charlie Jakes	charlied@example.com
<input type="checkbox"/> Edit Copy Delete	6	Yaa Frimpong	yaa@gmail.com
<input type="checkbox"/> Edit Copy Delete	7	Edmond Agyemang	Edmond@gmail.com

SELECT * From 'diningPreferences'

✓ Showing rows 0 - 4 (5 total, Query took 0.0001 seconds.)

```
SELECT * FROM `diningPreferences`
```

☐ Profiling [[Edit inline](#)] [[Edit](#)] [[Explain SQL](#)] [[Create PHP code](#)] [[Refresh](#)]

☐ Show all | Number of rows: 25 | Filter rows: Sort by key: None

Extra options

	preferencId	customerId	favoriteTable	dietaryRestrictions
<input type="checkbox"/> Edit Copy Delete	1	1	Table 5	None
<input type="checkbox"/> Edit Copy Delete	2	2	Table 9	Gluten-free
<input type="checkbox"/> Edit Copy Delete	3	3	Table 20	Vegetarian
<input type="checkbox"/> Edit Copy Delete	4	4	Table 15	No dairy
<input type="checkbox"/> Edit Copy Delete	5	5	Table 1	Vegan

☐ Check all With selected: [Edit](#) [Copy](#) [Delete](#) [Export](#)

☐ Show all | Number of rows: 25 | Filter rows: Sort by key: None

SELECT * From 'reservations'

Showing rows 0 - 4 (5 total, Query took 0.0001 seconds.)

SELECT * FROM `reservations`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

Edit

Copy

Delete

11

2024-12-10 19:00:00

2 Window seat preferred

Edit

Copy

Delete

22

2024-12-10 20:00:00

4 Allergic to peanuts

Edit

Copy

Delete

33

2024-12-10 18:30:00

3 Celebrating anniversary

Edit

Copy

Delete

44

2024-12-10 19:45:00

5 Need high chair for a toddler

Edit

Copy

Delete

55

2024-12-10 20:30:00

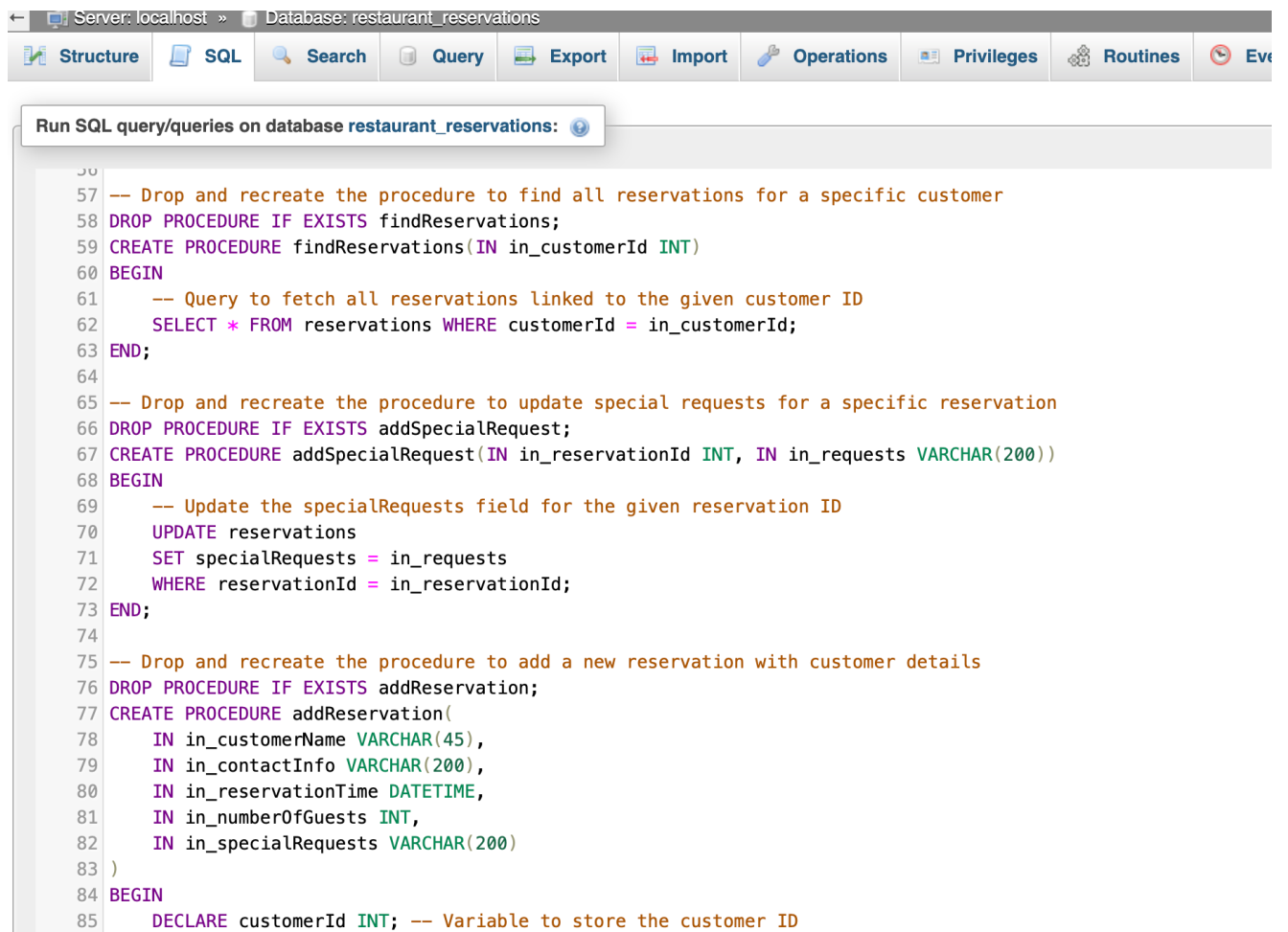
2 Vegetarian meal options

Check all

With selected: Edit Copy Delete Export

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Next, I have created two stored procedures to 'find the reservation' and 'add a special customer request'. findReservations procedure uses the existing customer id to find the reservation and addSpecialRequest procedure uses the existing reservation for the same.



The screenshot shows a MySQL IDE window with the title bar 'Server: localhost » Database: restaurant_reservations'. The interface includes tabs for Structure, SQL, Search, Query, Export, Import, Operations, Privileges, Routines, and Events. A toolbar at the top contains icons for these functions. Below the toolbar, a text box indicates 'Run SQL query/queries on database restaurant_reservations:'. The main area displays SQL code for creating and dropping stored procedures. The code is as follows:

```

50
51 -- Drop and recreate the procedure to find all reservations for a specific customer
52 DROP PROCEDURE IF EXISTS findReservations;
53 CREATE PROCEDURE findReservations(IN in_customerId INT)
54 BEGIN
55     -- Query to fetch all reservations linked to the given customer ID
56     SELECT * FROM reservations WHERE customerId = in_customerId;
57 END;
58
59 -- Drop and recreate the procedure to update special requests for a specific reservation
60 DROP PROCEDURE IF EXISTS addSpecialRequest;
61 CREATE PROCEDURE addSpecialRequest(IN in_reservationId INT, IN in_requests VARCHAR(200))
62 BEGIN
63     -- Update the specialRequests field for the given reservation ID
64     UPDATE reservations
65     SET specialRequests = in_requests
66     WHERE reservationId = in_reservationId;
67 END;
68
69 -- Drop and recreate the procedure to add a new reservation with customer details
70 DROP PROCEDURE IF EXISTS addReservation;
71 CREATE PROCEDURE addReservation(
72     IN in_customerName VARCHAR(45),
73     IN in_contactInfo VARCHAR(200),
74     IN in_reservationTime DATETIME,
75     IN in_numberOfGuests INT,
76     IN in_specialRequests VARCHAR(200)
77 )
78 BEGIN
79     DECLARE customerId INT; -- Variable to store the customer ID

```

The next stored procedure is addReservation procedure. This procedure checks if the customer id exists or not. If the customer id is not present, the new customer id created and then the reservation is made.

```
84 BEGIN
85     DECLARE customerId INT; -- Variable to store the customer ID
86
87     -- Check if the customer already exists in the Customers table
88     SELECT customerId INTO customerId
89     FROM customers
90     WHERE customerName = in_customerName AND contactInfo = in_contactInfo;
91
92     -- If customer does not exist, insert a new customer record
93     IF customerId IS NULL THEN
94         INSERT INTO customers (customerName, contactInfo)
95         VALUES (in_customerName, in_contactInfo);
96         SET customerId = LAST_INSERT_ID(); -- Retrieve the ID of the newly inserted customer
97     END IF;
98
99     -- Insert the reservation details into the Reservations table
100    INSERT INTO reservations (customerId, reservationTime, numberOfGuests, specialRequests)
101    VALUES (customerId, in_reservationTime, in_numberOfGuests, in_specialRequests);
102 END;
103
```

Clear

Format

Get auto-saved query

PHP: I started off by modifying the restaurant_reservations.php file with the appropriate MySQL

Credentials: To first communicate with MySQL, I used the same port number database that's in

MySQL so PHP can retrieve the information.

```
viewReservations.php  ReataurantDatabase.php ×  addReservatio
ReataurantDatabase.php > RestaurantDatabase > $port
1  <?php
2  class RestaurantDatabase {
3      private $host = "localhost";
4      private $port = "3306";
5      private $database = "restaurant_reservations";
6      private $user = "root";
7      private $password = "";
8      private $connection;
9
10     public function __construct() {
11         $this->connect();
12     }
13
```

Next, I added methods in php to do various tasks such as adding and viewing reservations, Adding customer. Each method in this portion of the python code is similar except for changing the “query” portion to each method's respective statements that correspond to the statements in MySQL.

```

29
30     private function addReservation() {
31         if ($_SERVER['REQUEST_METHOD'] === 'POST') {
32             $customerName = $_POST['customer_name'];
33             $contactInfo = $_POST['contact_info'] ?? null;
34             $reservationTime = $_POST['reservation_time'];
35             $numberOfGuests = $_POST['number_of_guests'];
36             $specialRequests = $_POST['special_requests'] ?? '';
37
38             $this->db->addReservation($customerName, $contactInfo, $reservationTime, $numberOfGuests, $specialRequests);
39             header("Location: index.php?action=viewReservations&message=Reservation+Added");
40         } else {
41             include 'templates/addReservation.php';
42         }
43     }
44
45     private function viewReservations() {
46         $reservations = $this->db->getAllReservations();
47         include 'templates/viewReservations.php';
48     }
49 }
50
51 $portal = new RestaurantPortal();
52 $portal->handleRequest();
53 ?>

```

```

38
39     public function getAllReservations() {
40         $sql = "SELECT * FROM reservations";
41         $stmt = $this->pdo->query($sql);
42         return $stmt->fetchAll(PDO::FETCH_ASSOC);
43     }
44
45     public function deleteReservation($reservationId) {
46         $sql = "DELETE FROM reservations WHERE reservationId = :reservationId";
47         $stmt = $this->pdo->prepare($sql);
48         $stmt->execute([':reservationId' => $reservationId]);
49     }
50
51     // Add a special request to an existing reservation
52     public function addSpecialRequest($reservationId, $requests) {
53         $sql = "UPDATE reservations SET specialRequests = :specialRequests WHERE reservationId = :reservationId";
54         $stmt = $this->pdo->prepare($sql);
55         $stmt->execute([
56             ':specialRequests' => $requests,
57             ':reservationId' => $reservationId,
58         ]);
59     }
60
61     // Find all reservations for a specific customer
62     public function findReservations($customerId) {
63         $sql = "SELECT * FROM reservations WHERE customerId = :customerId";
64         $stmt = $this->pdo->prepare($sql);
65         $stmt->execute([':customerId' => $customerId]);
66         return $stmt->fetchAll(PDO::FETCH_ASSOC);
67     }
68
69     // Search dining preferences for a specific customer
70     public function searchPreferences($customerId) {
71         $sql = "SELECT favoriteTable, dietaryRestrictions FROM diningPreferences WHERE customerId = :customerId";
72         $stmt = $this->pdo->prepare($sql);
73         $stmt->execute([':customerId' => $customerId]);
74         return $stmt->fetch(PDO::FETCH_ASSOC); // Single row for preferences
75     }
76 }
77 ?>

```

Some of the additional functions I implemented are `getCustomerPreference`, `addSpecialRequest`, `findReservation`, and `deleteReservation`.

Now in the `restaurantServer.php` file

Mainly I called the methods from the `restaurantDatabase.php` and made sure it was visible on the local host web page. For adding reservation, in `do_POST` I initialize the variables from the reservation table and then call the method to add a new reservation from the database file.

```
51
52     // Add the reservation
53     $stmt = $this->connection->prepare(
54         "INSERT INTO Reservations (customerId, reservationTime, numberOfGuests, specialRequests) VALUES (?, ?, ?, ?"
55     );
56     $stmt->bind_param("isis", $customerId, $reservationTime, $numberOfGuests, $specialRequests);
57     $stmt->execute();
58     $stmt->close();
59 }
60
61 public function getAllReservations() {
62     $result = $this->connection->query("SELECT * FROM Reservations");
63     return $result->fetch_all(MYSQLI_ASSOC);
64 }
65
66 public function getCustomerPreferences($customerId) {
67     $stmt = $this->connection->prepare(
68         "SELECT favoriteTable, dietaryRestrictions FROM DiningPreferences WHERE customerId = ?"
69     );
70     $stmt->bind_param("i", $customerId);
71     $stmt->execute();
72     $result = $stmt->get_result();
73     $preferences = $result->fetch_assoc();
74     $stmt->close();
75     return $preferences;
76 }
77
```

Similarly, I have called other functions as well in the server file as per the requirements of the project.

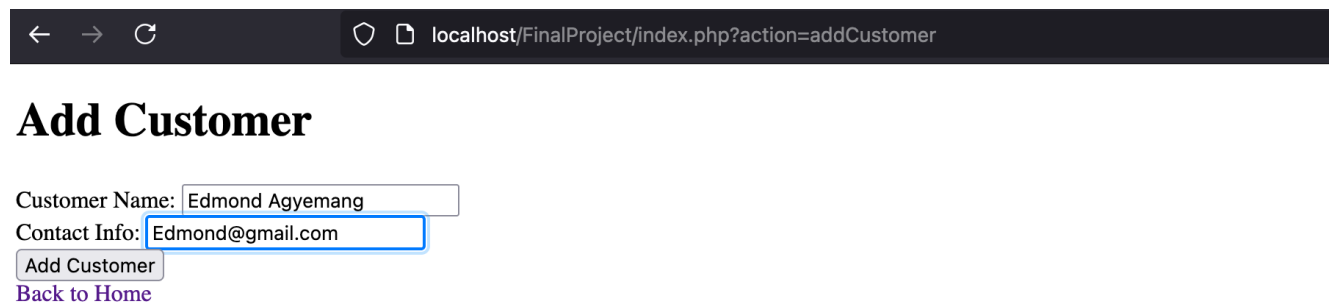
Other things I added were some validation inputs to make sure that a user actually inserts data into the forms when adding a reservation and deleting a reservation.

If they do not enter information they are prompted to do it.

I also made sure to place the menu navigation links on all the pages throughout the website to make the view uniform and pleasant.

I also added some comments to guide the user on how to type in their reservation.

See Screenshots of the website in full function below.



The screenshot shows a web browser window with the address bar displaying 'localhost/FinalProject/index.php?action=addCustomer'. The page title is 'Add Customer'. Below the title, there are two input fields: 'Customer Name:' with the value 'Edmond Agyemang' and 'Contact Info:' with the value 'Edmond@gmail.com'. Below these fields is a button labeled 'Add Customer' and a link labeled 'Back to Home'.

← → ↻ localhost/FinalProject/index.php?action=addCustomer

Add Customer

Customer Name: Edmond Agyemang

Contact Info: Edmond@gmail.com


Add Customer

[Back to Home](#)

ADD RESERVATIONS

Add Reservation

Customer Name:

Reservation Time: 

Number of Guests:

Special Requests:

[Back to Home](#)

Reservation Added

Reservation has been successfully added.

[Add Another Reservation](#)
[View Reservations](#)

VIEW TO SEE IF RESERVATION ADDED

All Reservations

Reservation ID	Customer ID	Reservation Time	Number of Guests	Special Requests
1	1	2024-12-10 19:00:00	2	Window seat preferred
2	2	2024-12-10 20:00:00	4	Allergic to peanuts
3	3	2024-12-10 18:30:00	3	Celebrating anniversary
4	4	2024-12-10 19:45:00	5	Need high chair for a toddler
5	5	2024-12-10 20:30:00	2	Vegetarian meal options
6	7	2024-12-14 11:40:00	2	Jollof rice , very spicy

[Back to Home](#)

DELETE RESERVATION BY ID

Delete Reservations

[Home](#) | [Add Reservation](#) | [Delete Reservation](#) | [View Reservations](#)

Delete Reservation

Reservation ID to delete: (Please input the ID you wish to delete)

Delete Reservations

[Home](#) | [Add Reservation](#) | [Delete Reservation](#) | [View Reservations](#)

Delete Reservation

Reservation ID to delete: (Please input the ID you wish to delete)

Reservation Deleted

Reservation has been successfully deleted.

[Delete Another Reservation](#)
[View Reservations](#)

VIEW ALL RESERVATIONS TO SEE IF RESERVATION DELETED