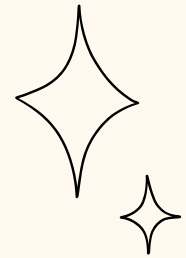
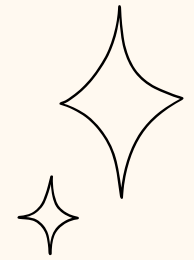
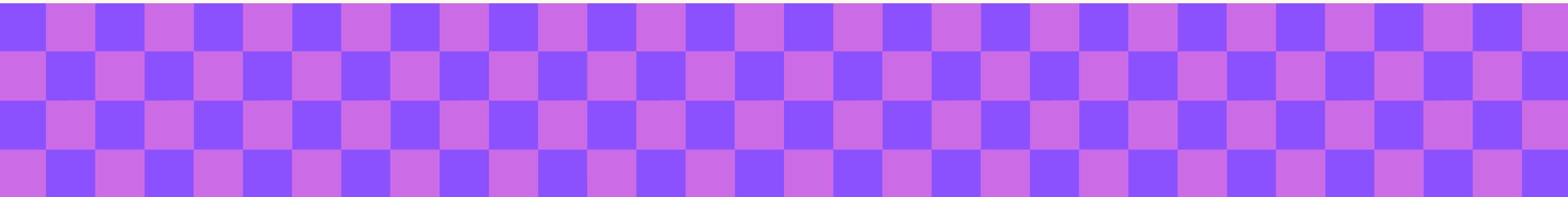


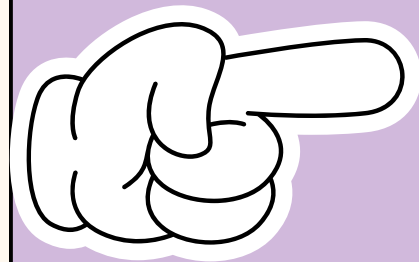
# ***PHP CONNECT TO DATABASE***



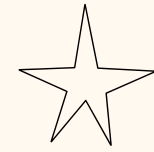
DIAN X BADONG



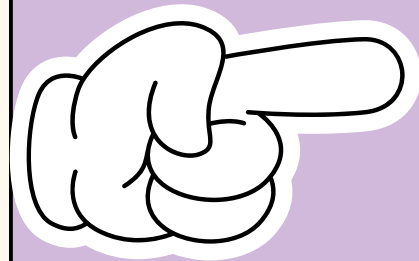
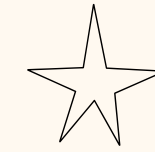
# ☆ *Introduction* ☆



**PHP is a widely used server-side scripting language that powers many dynamic websites and applications. One of its core functionalities is the ability to interact with databases, particularly MySQL. This capability allows web applications to manage data dynamically, such as handling user accounts, storing content, or processing transactions.**



# ***Discussion***



**When developing dynamic web applications, databases play an essential role in storing user information, product details, or any other data that needs to be accessed, modified, or updated. PHP offers multiple methods for connecting to databases, with MySQL being the most commonly used.**

# ☆ *Key elements* ☆

- 1.Database Server:** The database management system (DBMS) where the database is hosted, such as MySQL, MariaDB, or others.
- 2.Hostname:** The address of the database server (e.g., "localhost" for a local server or an IP address/domain for remote servers).
- 3.Username:** The login credential used to access the database.
- 4.Password:** The password associated with the username.
- 5.Database Name:** The specific database to which you want to connect.

# *Methods to Connect PHP to a MySQL Database*

PHP offers two modern approaches to establish a connection with a MySQL database: MySQLi and PDO.

**MySQL** is a database management system that helps store, organize, and retrieve data efficiently.

**PDO** is a tool in PHP that lets your PHP scripts connect and interact with a database like MySQL.

# ***Purpose of Connecting PHP to a MySQL Database***

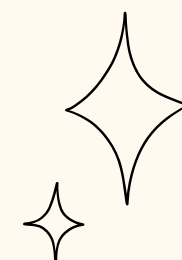
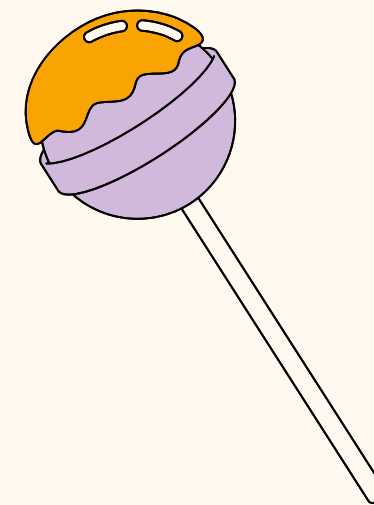
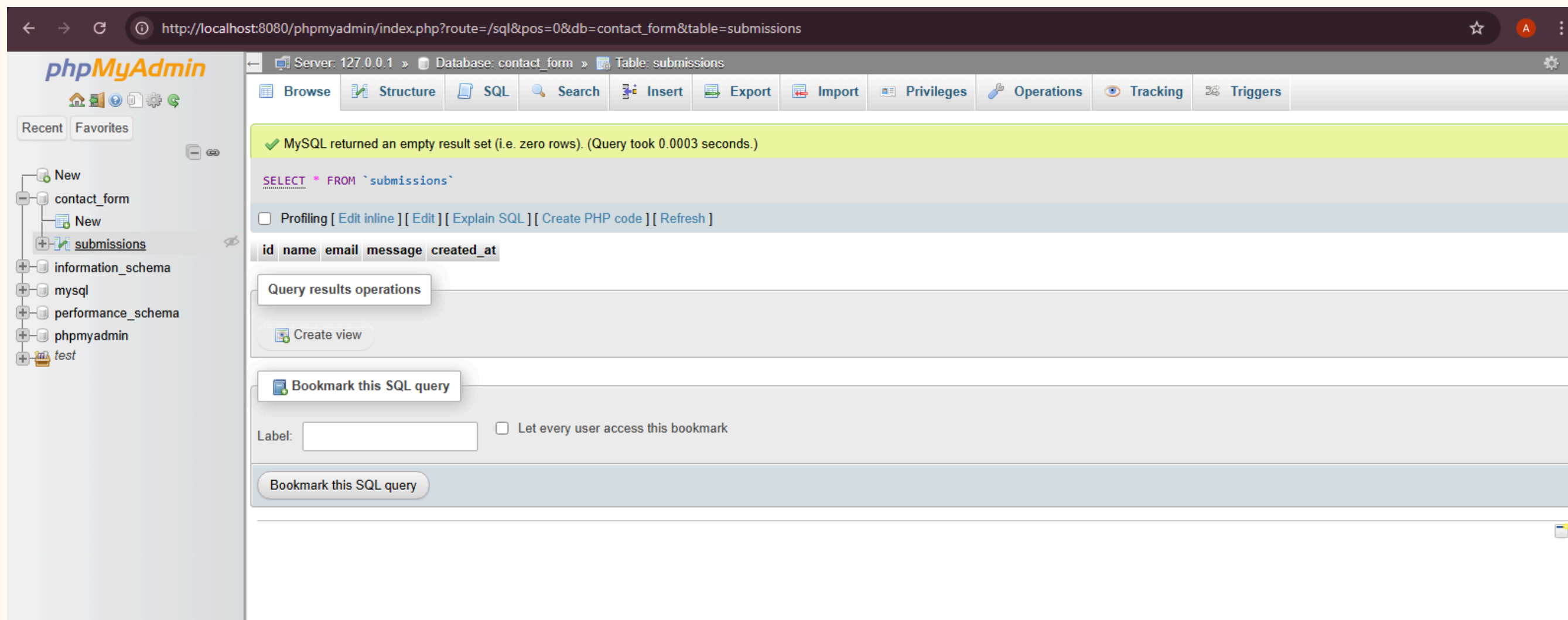
1. **Dynamic Data Management:** To fetch, store, or update data for applications like e-commerce platforms, blogs, or user dashboards.
2. **Efficient Data Handling:** Databases provide structured storage for large datasets.
3. **Interactive Features:** Enables features like login systems, product searches, or user profiles.

# ***Advantages of PHP-MySQL Integration***

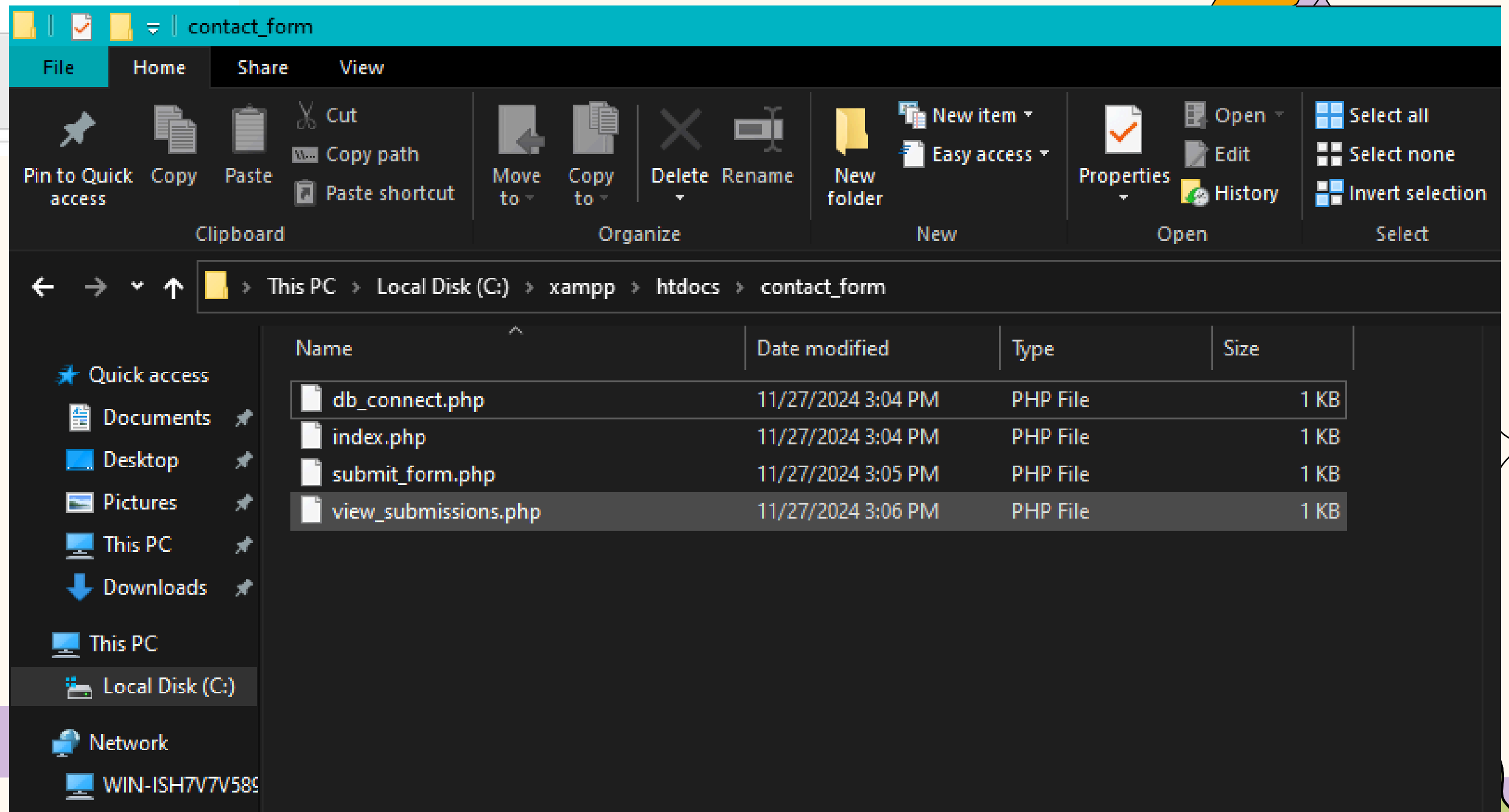
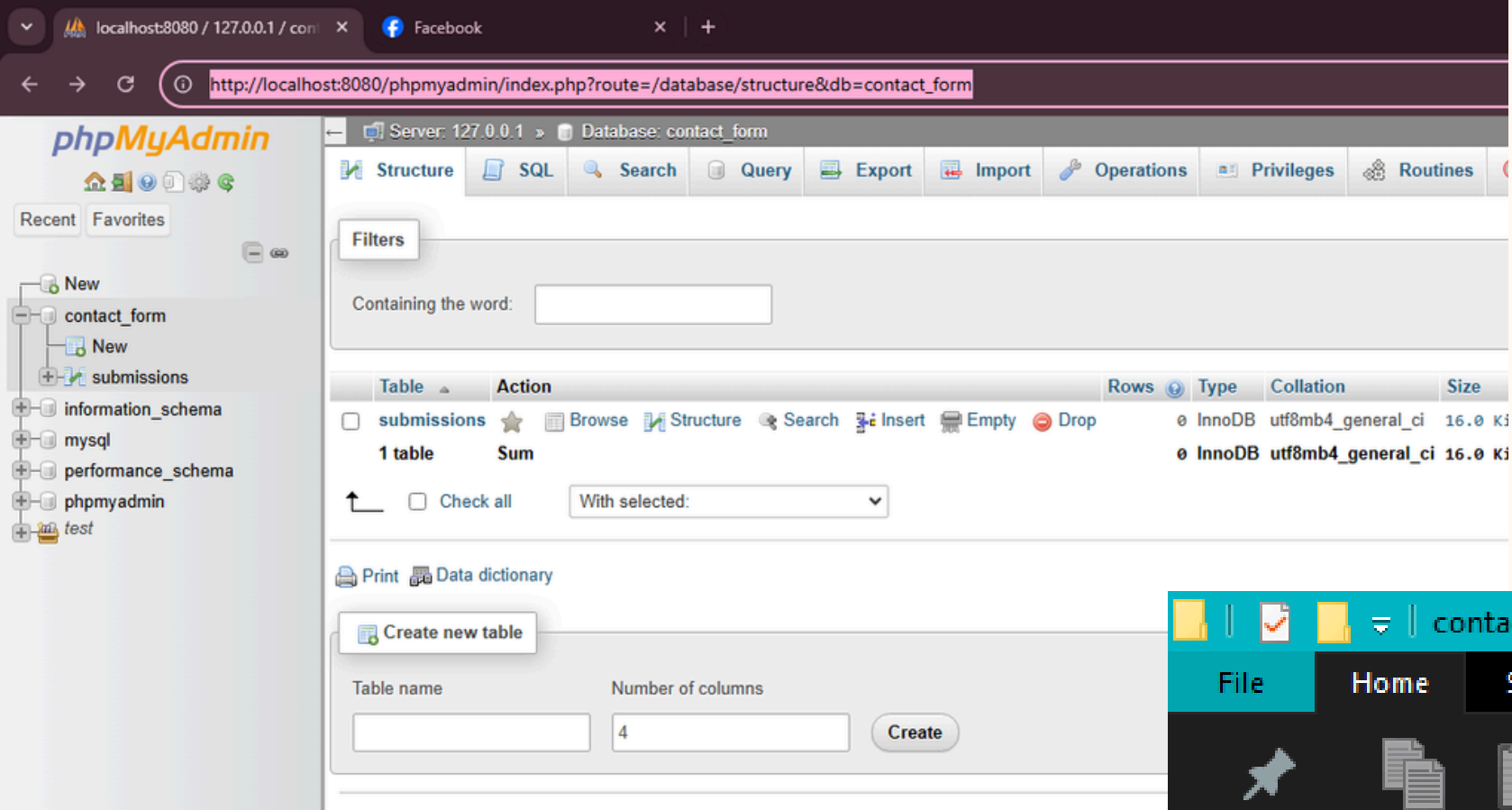
1. **Dynamic Content**: Enables real-time updates and personalized user experiences.
2. **Security**: Supports features like prepared statements to protect against SQL injection.
3. **Scalability**: Efficiently handles growing data and user demands.
4. **Flexibility**: Suitable for diverse applications, from small blogs to large e-commerce sites.

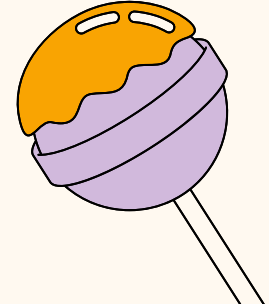
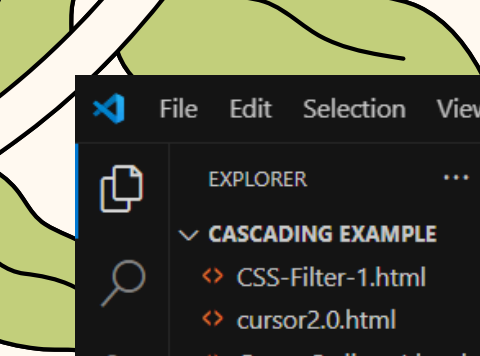
# Connecting the PHP to Database example

Create the Database in phpMyAdmin or using MySQL CLI, create a database for storing the information:










Visual Studio Code interface showing the Explorer sidebar with a project named "CASCADING EXAMPLE". The Explorer sidebar lists files: CSS-Filter-1.html, cursor2.0.html, CursorStyling-1.html, Damn-0.html, DANDADAN.html, db\_connect.php, EXAMP1.CSS, example 3.html, example.php, example1.html, example1.txt, EXAMPLE2.html, hands-on.css, hands-on.html, and HTML.code-workspace. The main editor area displays the code for db\_connect.php:

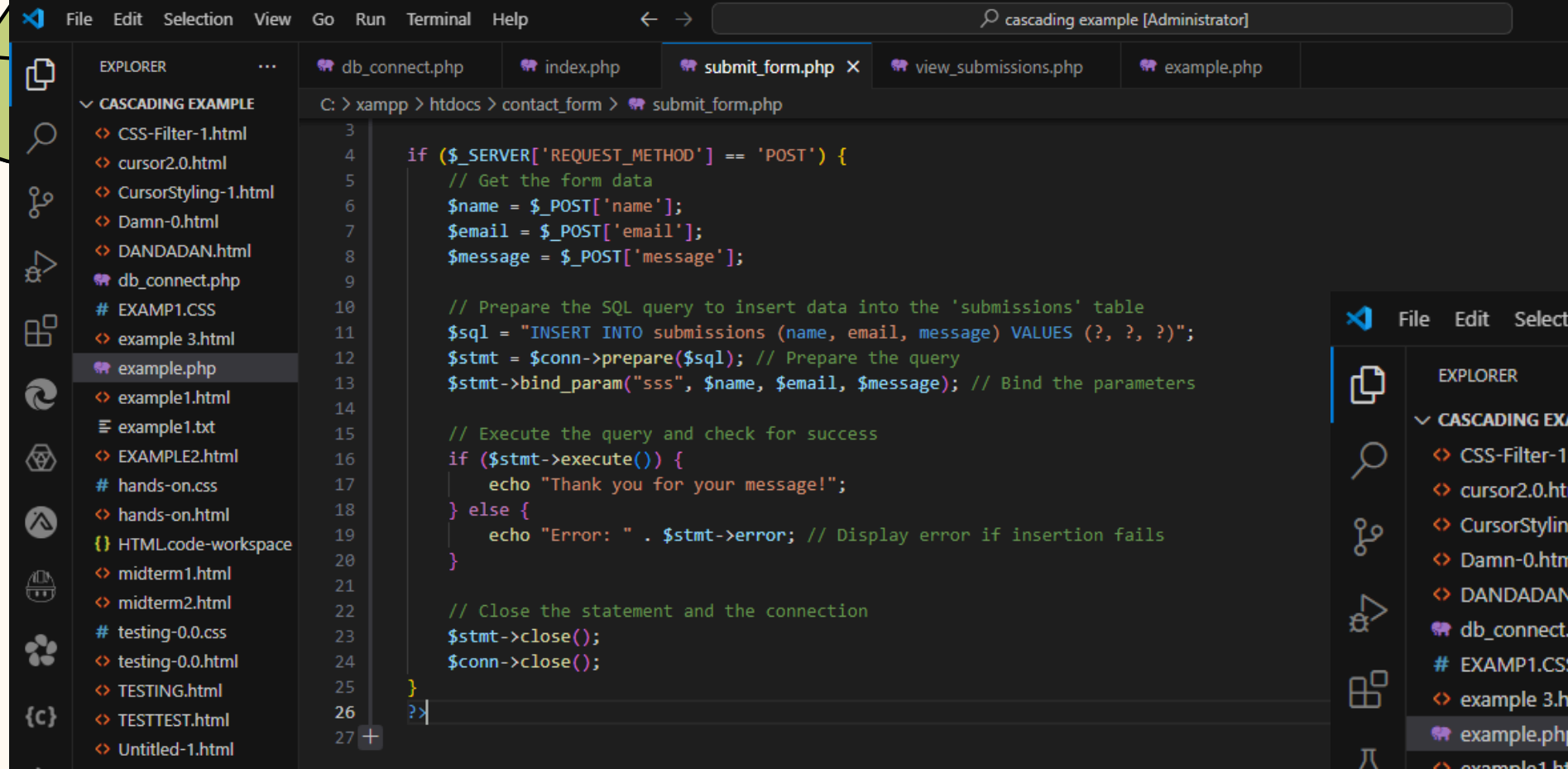
```
C: > xampp > htdocs > contact_form > db_connect.php
1  <?php
2  $host = "localhost"; // MySQL server
3  $username = "root"; // Default username for XAMPP or local server
4  $password = ""; // Default password for XAMPP (empty)
5  $dbname = "contact_form"; // The database we created
6
7  // Create connection
8  $conn = new mysqli($host, $username, $password, $dbname);
9
10 // Check connection
11 if ($conn->connect_error) {
12     die("Connection failed: " . $conn->connect_error);
13 }
14 ?>
15
```



Visual Studio Code interface showing the Explorer sidebar with a project named "CASCADING EXAMPLE". The Explorer sidebar lists files: CSS-Filter-1.html, cursor2.0.html, CursorStyling-1.html, Damn-0.html, DANDADAN.html, db\_connect.php, EXAMP1.CSS, example 3.html, example.php, example1.html, example1.txt, EXAMPLE2.html, hands-on.css, hands-on.html, HTML.code-workspace, midterm1.html, midterm2.html, testing-0.0.css, testing-0.0.html, TESTING.html, TESTTEST.html, Untitled-1.html, WeatherApp.html, and WeatherGlowingBtn.h... The main editor area displays the code for index.php:

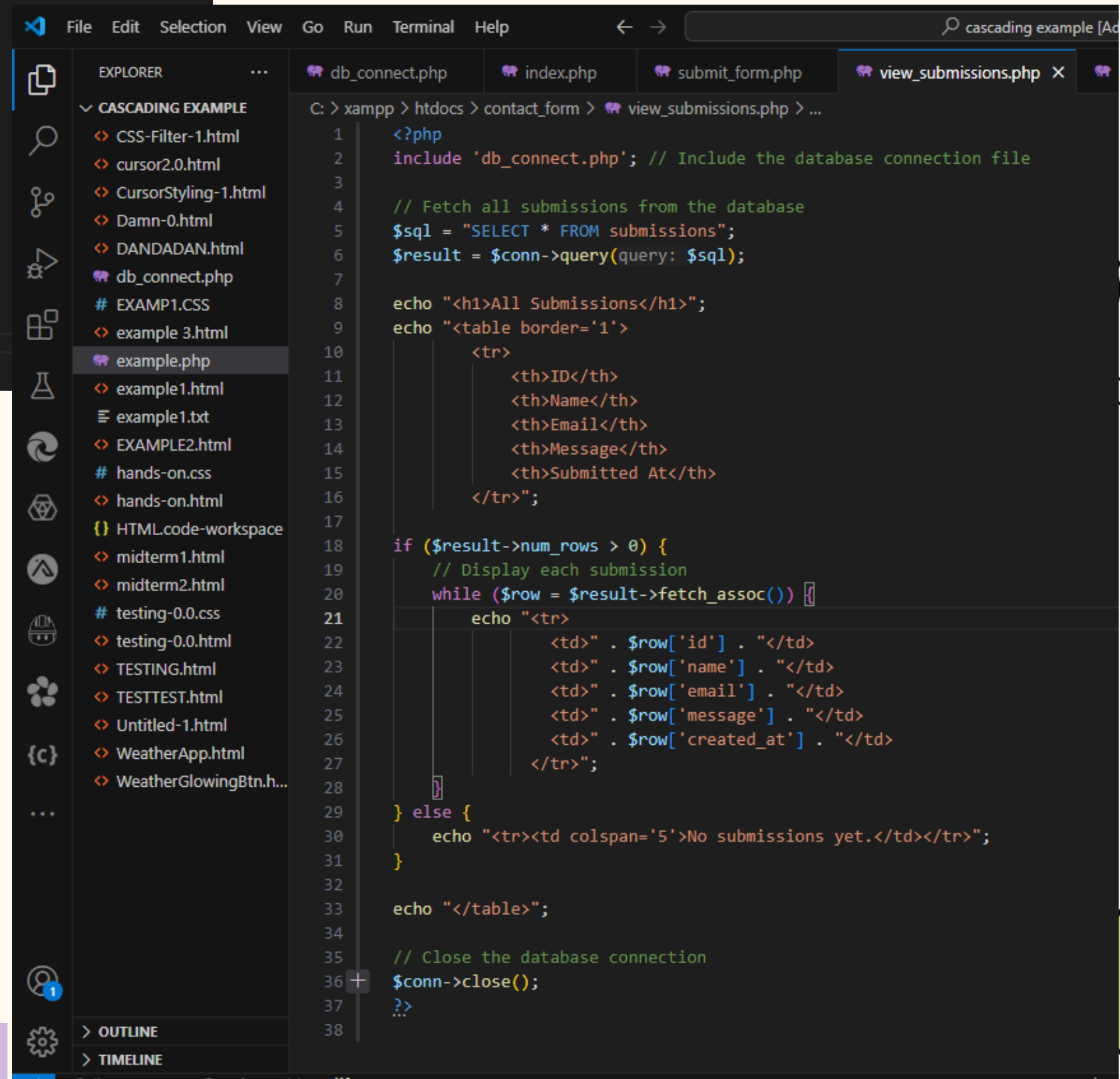
```
C: > xampp > htdocs > contact_form > index.php
1  <?php include 'db_connect.php'; ?>
2
3  <!DOCTYPE html>
4  <html lang="en">
5  <head>
6      <meta charset="UTF-8">
7      <meta name="viewport" content="width=device-width, initial-scale=1.0">
8      <title>Contact Us</title>
9  </head>
10 <body>
11     <h1>Contact Us</h1>
12     <form action="submit_form.php" method="POST">
13         <label for="name">Name:</label><br>
14         <input type="text" id="name" name="name" required><br><br>
15
16         <label for="email">Email:</label><br>
17         <input type="email" id="email" name="email" required><br><br>
18
19         <label for="message">Message:</label><br>
20         <textarea id="message" name="message" rows="4" required></textarea><br><br>
21
22         <button type="submit">Submit</button>
23     </form>
24 </body>
25 </html>
```

The bottom status bar shows the terminal output: PS C:\Users\Administrator\Documents\cascading example>



The image shows a VS Code editor window with the file explorer on the left. The active file is `submit_form.php` located at `C:\xampp\htdocs\contact_form\submit_form.php`. The code is a PHP script that handles form submissions by connecting to a database and inserting data into a table named `submissions`.

```
3
4 if ($_SERVER['REQUEST_METHOD'] == 'POST') {
5     // Get the form data
6     $name = $_POST['name'];
7     $email = $_POST['email'];
8     $message = $_POST['message'];
9
10    // Prepare the SQL query to insert data into the 'submissions' table
11    $sql = "INSERT INTO submissions (name, email, message) VALUES (?, ?, ?)";
12    $stmt = $conn->prepare($sql); // Prepare the query
13    $stmt->bind_param("sss", $name, $email, $message); // Bind the parameters
14
15    // Execute the query and check for success
16    if ($stmt->execute()) {
17        echo "Thank you for your message!";
18    } else {
19        echo "Error: " . $stmt->error; // Display error if insertion fails
20    }
21
22    // Close the statement and the connection
23    $stmt->close();
24    $conn->close();
25 }
26 ?>
```



The image shows a VS Code editor window with the file explorer on the left. The active file is `view_submissions.php` located at `C:\xampp\htdocs\contact_form\view_submissions.php`. The code is a PHP script that fetches all submissions from the database and displays them in an HTML table.

```
1 <?php
2 include 'db_connect.php'; // Include the database connection file
3
4 // Fetch all submissions from the database
5 $sql = "SELECT * FROM submissions";
6 $result = $conn->query(query: $sql);
7
8 echo "<h1>All Submissions</h1>";
9 echo "<table border='1'>
10     <tr>
11         <th>ID</th>
12         <th>Name</th>
13         <th>Email</th>
14         <th>Message</th>
15         <th>Submitted At</th>
16     </tr>";
17
18 if ($result->num_rows > 0) {
19     // Display each submission
20     while ($row = $result->fetch_assoc()) {
21         echo "<tr>
22             <td>" . $row['id'] . "</td>
23             <td>" . $row['name'] . "</td>
24             <td>" . $row['email'] . "</td>
25             <td>" . $row['message'] . "</td>
26             <td>" . $row['created_at'] . "</td>
27         </tr>";
28     }
29 } else {
30     echo "<tr><td colspan='5'>No submissions yet.</td></tr>";
31 }
32
33 echo "</table>";
34
35 // Close the database connection
36 $conn->close();
37 ?>
```



# OUTCOME

← → ↻ ⓘ http://localhost:8080/contact\_form/

## Contact Us




Name:

Email:

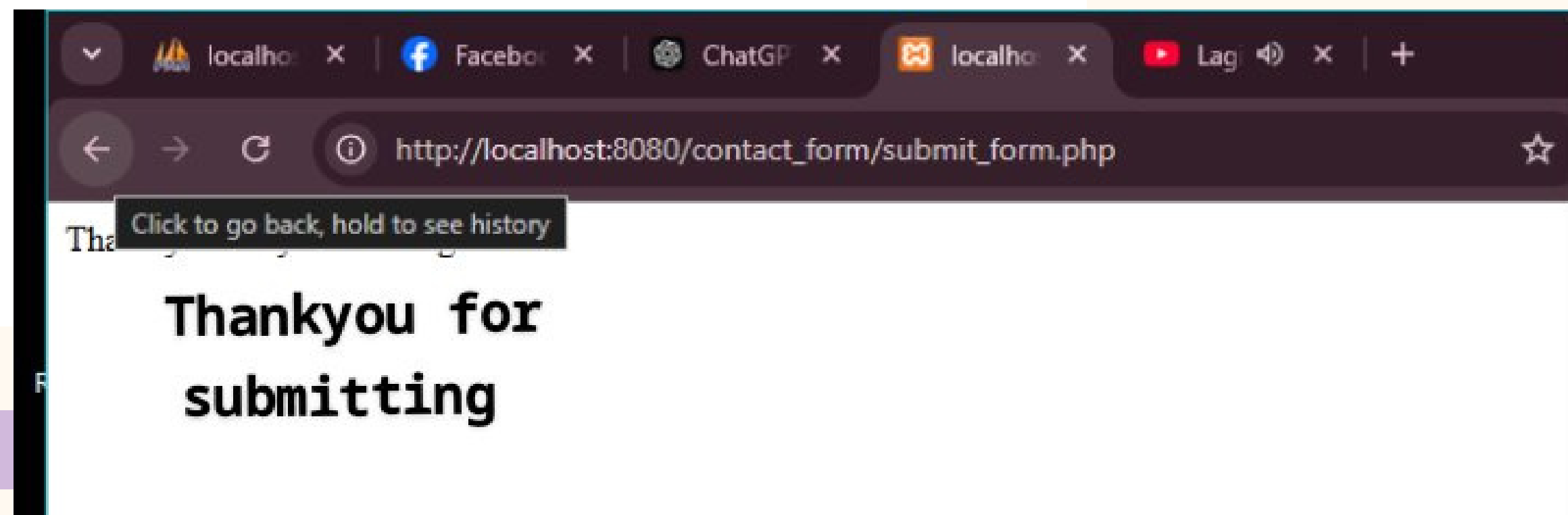
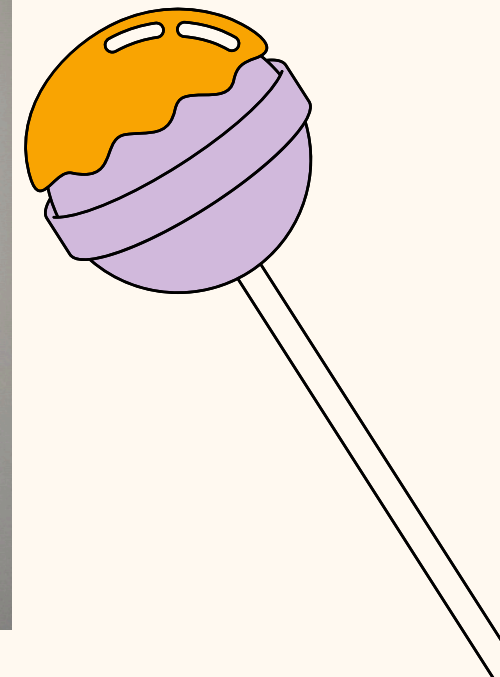
Message:

Submit

Index of /quizform

<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
 <a href="#">Parent Directory</a>			
 <a href="#">form.php</a>	2024-12-03 05:44	725	
 <a href="#">process.php</a>	2024-12-03 05:45	593	

Apache/2.4.58 (Win64) OpenSSL/3.1.3 PHP/8.0.30 Server at localhost Port 80

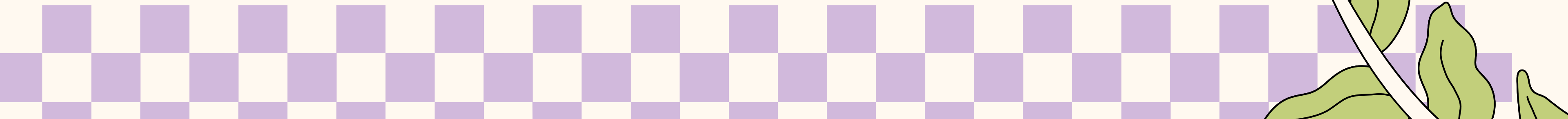




## Step 4: Test the Application

1. Place all files (db\_connect.php, index.php, submit\_rsvp.php, guest\_list.php) in the root directory of your server (e.g., htdocs for XAMPP).
2. Open a browser and navigate to:
  - <http://localhost/index.php> to access the RSVP form.
  - [http://localhost/contact\\_form.php](http://localhost/contact_form.php) to .

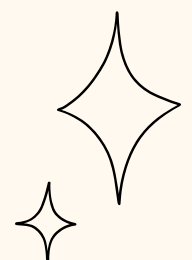
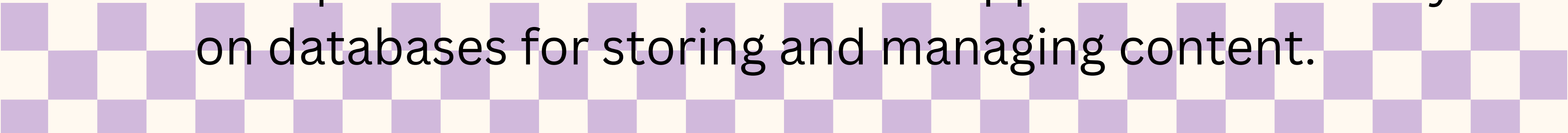
Test the form .

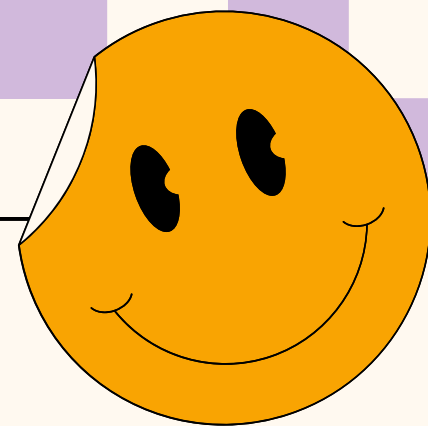




# *CONCLUSION*

Connecting PHP to a database is an essential step in building dynamic, data-driven websites. The process is straightforward and can be achieved using MySQLi or PDO, both of which offer secure ways to manage database interactions. In this report, we discussed the basic concepts of database connection, provided an example of how to connect PHP to a MySQL database, and demonstrated how to retrieve and display data. With this knowledge, developers can create powerful and scalable web applications that rely on databases for storing and managing content.





***Yun lang tapos  
na***

