**POST TEST**

**PEMROGRAMAN WEB 1**

**MODUL 9**



NAMA : Aef Nur Hidayah

NIM : 20230910054

KELAS : SINFC – 2023 - 03

FAKULTAS ILMU KOMPUTER

UNIVERSITAS KUNINGAN

TAHUN 2024

# **D. PRE-TEST**

1. Apa fungsi dari mysqli\_connect() dalam PHP?

Fungsi mysqli\_connect() dalam PHP digunakan untuk membuat koneksi ke database MySQL. Fungsi ini menghubungkan aplikasi PHP dengan database MySQL menggunakan ekstensi MySQLi (MySQL Improved). Fungsi ini menerima beberapa parameter untuk menentukan server database, nama pengguna, kata sandi, dan nama database yang ingin diakses.

2. Jelaskan perbedaan antara mysqli\_query() dan mysqli\_prepare().

mysqli\_query() digunakan untuk menjalankan query SQL langsung pada database. Fungsi ini mengirimkan query ke server MySQL dan mengeksekusinya. Kelebihannya lebih sederhana dan cepat, sedangkan kekurangannya rentan terhadap SQL Injection.

mysqli\_prepare() digunakan untuk mempersiapkan query SQL dengan menggunakan parameter yang akan diikat kemudian (binding). Ini lebih aman karena mencegah SQL Injection, karena data yang dimasukkan oleh pengguna tidak langsung digabungkan dalam query, melainkan diikat sebagai parameter. Kelebihannya keamanannya yang lebih baik dan penggunaan yang efisien sedangkan kekurangannya yaitu lebih kompleks saja.

3. Apa yang dimaksud dengan prepared statement dan sebutkan salah satu keuntungannya?

Prepared Statement adalah suatu teknik dalam pengolahan database yang memungkinkan kamu untuk menyiapkan query SQL terlebih dahulu, dan kemudian menjalankannya dengan memasukkan parameter-parameter secara terpisah dari query SQL itu sendiri. Teknik ini biasanya digunakan untuk meningkatkan keamanan dan efisiensi dalam menjalankan query yang berulang dengan parameter yang berbeda.

Pada prepared statement, query SQL dikirim ke server database terlebih dahulu tanpa nilai parameter (biasanya digantikan dengan tanda ? untuk parameter). Setelah itu, parameter yang akan digunakan dalam query diikat ke statement yang telah dipersiapkan tersebut, dan akhirnya query tersebut dieksekusi.

Keuntungan Prepared Statement:

Salah satu keuntungan utama dari prepared statement adalah keamanan terhadap SQL Injection. Karena nilai-nilai parameter tidak digabungkan langsung ke dalam query, maka tidak ada kesempatan bagi penyerang untuk memasukkan kode berbahaya (seperti SQL Injection) ke dalam query.

4. Sebutkan dua metode yang dapat digunakan untuk menangani kesalahan dalam MySQLi dan jelaskan cara kerjanya.

a. **Menggunakan mysqli\_error()**

Metode ini memungkinkan kita untuk mendapatkan pesan kesalahan yang terjadi pada koneksi atau eksekusi query yang gagal. Fungsi mysqli\_error() mengembalikan pesan kesalahan dari server MySQL jika ada masalah dalam menjalankan query atau koneksi.

**Cara Kerja:**

* Setelah memanggil fungsi mysqli\_query() atau fungsi MySQLi lainnya, jika query gagal, kita dapat menggunakan mysqli\_error($koneksi) untuk mendapatkan deskripsi kesalahan.
* Ini membantu kita untuk melihat alasan kegagalan query atau koneksi dan memperbaikinya.

b. **Menggunakan mysqli\_errno()**

Metode ini mengembalikan nomor kesalahan (error code) yang terkait dengan masalah yang terjadi pada query atau koneksi. Jika terjadi kesalahan, kita bisa mendapatkan kode kesalahan yang lebih spesifik, yang bisa digunakan untuk mendiagnosis masalah.

**Cara Kerja:**

* mysqli\_errno($koneksi) mengembalikan nomor kode kesalahan MySQL yang terkait dengan operasi terakhir. Setiap kesalahan memiliki kode unik yang bisa digunakan untuk mencari tahu lebih detail tentang masalah yang terjadi.
* Ini berguna untuk menangani kesalahan secara lebih terstruktur dan dapat memberikan informasi lebih lanjut selain pesan kesalahan biasa.

5. Bagaimana cara menutup koneksi ke database MySQL dalam PHP setelah selesai digunakan?

Untuk menutup koneksi ke database MySQL dalam PHP setelah selesai digunakan, kamu dapat menggunakan fungsi mysqli\_close(). Fungsi ini digunakan untuk menutup koneksi yang telah dibuat menggunakan mysqli\_connect().

|  |
| --- |
| mysqli\_close($koneksi); |

6. Apa perbedaan antara fungsi mysqli\_fetch\_assot() dan mysqli\_fetch\_row() dalam PHP?

Perbedaan antara mysqli\_fetch\_assoc() dan mysqli\_fetch\_row() terletak pada cara mereka mengembalikan hasil query dari database MySQL dalam bentuk array.

**a. mysqli\_fetch\_assoc()**

Fungsi ini mengembalikan hasil query sebagai **array asosiatif**, di mana setiap elemen array berisi pasangan **kolom-nama** dan **nilai**. Nama kolom dari hasil query akan menjadi **kunci** array, dan nilai pada baris tersebut akan menjadi **nilai** array.

|  |
| --- |
| $baris = mysqli\_fetch\_assoc($result); |

* **$result**: Hasil dari eksekusi query (mysqli\_query() atau mysqli\_store\_result()).
* **$baris**: Array asosiatif yang berisi nama kolom dan nilai data dari baris hasil query.

b. **mysqli\_fetch\_row()**

Fungsi ini mengembalikan hasil query sebagai **array numerik**, di mana setiap elemen array berisi nilai dari kolom yang sesuai berdasarkan urutan kolom dalam query. Indeks array dimulai dari 0 dan berurutan sesuai dengan posisi kolom dalam hasil query.

|  |
| --- |
| $baris = mysqli\_fetch\_row($result); |

* **$result**: Hasil dari eksekusi query (mysqli\_query() atau mysqli\_store\_result()).
* **$baris**: Array numerik yang berisi nilai data dari baris hasil query.

7. Tuliskan sintaks dasar untuk melakukan query SQL UPDATE menggunakan MySQLi.

|  |
| --- |
| $query = "UPDATE nama\_tabel SET kolom1 = 'nilai1', kolom2 = 'nilai2' WHERE kondisi";  $result = mysqli\_query($koneksi, $query); |

* **nama\_tabel**: Nama tabel yang akan diperbarui.
* **kolom1, kolom2**: Nama kolom yang ingin diperbarui.
* **nilai1, nilai2**: Nilai baru yang akan dimasukkan ke dalam kolom tersebut.
* **kondisi**: Syarat yang menentukan baris mana yang akan diperbarui (biasanya menggunakan WHERE).

8. Apa yang dilakukan fungsi mysqli\_stmt\_bind\_param()?

Fungsi **mysqli\_stmt\_bind\_param()** digunakan untuk mengikat parameter ke **prepared statement** di MySQLi. Fungsi ini memungkinkan kita untuk mengirimkan nilai-nilai variabel sebagai parameter ke query SQL yang telah dipersiapkan menggunakan **mysqli\_prepare()**.

Tujuan utama dari fungsi ini adalah untuk mencegah **SQL Injection** dan memungkinkan eksekusi query dengan nilai-nilai yang dinamis atau input dari pengguna, dengan cara yang lebih aman.

**Cara Kerja:**

1. Query SQL dipersiapkan terlebih dahulu menggunakan mysqli\_prepare().
2. Parameter-parameter dalam query digantikan dengan tanda ?.
3. Fungsi mysqli\_stmt\_bind\_param() digunakan untuk mengikat nilai ke tanda ? tersebut, memastikan bahwa nilai tersebut diproses dengan cara yang aman.

9. Apa itu mysqli\_stmt\_get\_result() dan kapan anda menggunakannya?

Fungsi **mysqli\_stmt\_get\_result()** digunakan untuk mengambil hasil dari sebuah **prepared statement** yang telah dieksekusi menggunakan **MySQLi**. Fungsi ini mengembalikan objek hasil query, yang dapat digunakan untuk mengambil baris data dari query menggunakan fungsi-fungsi seperti mysqli\_fetch\_assoc(), mysqli\_fetch\_row(), dll.

Fungsi ini hanya dapat digunakan jika prepared statement yang dieksekusi menghasilkan **result set** (seperti query SELECT). Jika query yang dijalankan bukan query yang menghasilkan hasil (seperti INSERT, UPDATE, atau DELETE), maka fungsi ini tidak dapat digunakan.

**Kapan Menggunakan mysqli\_stmt\_get\_result():**

* Gunakan **mysqli\_stmt\_get\_result()** ketika kamu ingin mendapatkan hasil dari query SELECT yang telah dieksekusi menggunakan prepared statement.
* Fungsi ini memudahkan dalam mengambil dan memproses data hasil query, terutama ketika query menghasilkan banyak baris.

10. Sebutkan Langkah – Langkah dasar utnuk menangani hasil query menggunakan myqli\_query().

**Langkah 1: Membuat Koneksi ke Database**

Langkah pertama adalah membuat koneksi ke database menggunakan **mysqli\_connect()**. Pastikan kamu sudah memeriksa apakah koneksi berhasil atau tidak.

|  |
| --- |
| $koneksi = mysqli\_connect("localhost", "root", "", "my\_database");  if (!$koneksi) {  die("Koneksi gagal: " . mysqli\_connect\_error());  } |

**Langkah 2: Menulis Query SQL**

Tulis query SQL yang akan dijalankan. Misalnya, query SELECT untuk mengambil data dari tabel.

|  |
| --- |
| $query = "SELECT id, username, email FROM users"; |

**Langkah 3: Menjalankan Query menggunakan mysqli\_query()**

Gunakan **mysqli\_query()** untuk menjalankan query SQL pada koneksi yang telah dibuat. Fungsi ini mengembalikan objek hasil jika query berhasil (untuk query SELECT), atau TRUE/FALSE jika itu adalah query yang tidak menghasilkan data (seperti INSERT, UPDATE, DELETE).

|  |
| --- |
| $result = mysqli\_query($koneksi, $query); |

**Langkah 4: Memeriksa Apakah Query Berhasil**

Periksa apakah query berhasil dieksekusi. Jika query gagal, kamu bisa menangani kesalahan menggunakan **mysqli\_error()**.

|  |
| --- |
| if (!$result) {  die("Query gagal: " . mysqli\_error($koneksi));  } |

**Langkah 5: Mengambil dan Memproses Hasil Query**

Jika query adalah SELECT, gunakan fungsi seperti **mysqli\_fetch\_assoc()** atau **mysqli\_fetch\_row()** untuk mengambil hasil query satu per satu. Ini akan mengembalikan data dari hasil query.

Menggunakan **mysqli\_fetch\_assoc()** (Array Asosiatif):

|  |
| --- |
| while ($row = mysqli\_fetch\_assoc($result)) {  echo "ID: " . $row['id'] . "<br>";  echo "Username: " . $row['username'] . "<br>";  echo "Email: " . $row['email'] . "<br>";  } |

Menggunakan **mysqli\_fetch\_row()** (Array Numerik):

|  |
| --- |
| while ($row = mysqli\_fetch\_row($result)) {  echo "ID: " . $row[0] . "<br>";  echo "Username: " . $row[1] . "<br>";  echo "Email: " . $row[2] . "<br>";  } |

**Langkah 6: Menutup Koneksi**

Setelah selesai mengambil dan memproses hasil query, jangan lupa untuk menutup koneksi ke database menggunakan **mysqli\_close()**.

|  |
| --- |
| mysqli\_close($koneksi); |

Penjelasan:

1. Membuat koneksi: Menggunakan mysqli\_connect() untuk membuat koneksi ke database.
2. Menulis query: Query SQL disusun untuk mengambil data yang diinginkan.
3. Menjalankan query: mysqli\_query() digunakan untuk menjalankan query.
4. Memeriksa hasil: Memastikan query berhasil dijalankan dengan memeriksa objek hasil.
5. Mengambil data: Menggunakan mysqli\_fetch\_assoc() atau mysqli\_fetch\_row() untuk memproses hasil query baris demi baris.
6. Menutup koneksi: Koneksi ditutup dengan mysqli\_close() untuk melepaskan sumber daya.

# **E. POST-TEST**

1. Tambahkan Halaman Buku Tamu

Untuk menambahkan halaman buku tamu pada proyek ini, kita bisa membuat file baru bernama Latihan\_09\_bukutamu.php. Halaman ini akan memungkinkan pengguna untuk**Mengisi buku tamu** dengan nama, email, dan pesan.**Menampilkan daftar buku tamu** yang telah diisi oleh pengguna sebelumnya.

1. **Buat Tabel Buku Tamu di Database**

Tambahkan tabel baru ke dalam database db\_alumni. Gunakan SQL berikut untuk membuat tabel:

|  |
| --- |
| CREATE TABLE bukutamu (  id INT AUTO\_INCREMENT PRIMARY KEY,  nama VARCHAR(100) NOT NULL,  email VARCHAR(100),  pesan TEXT NOT NULL,  tanggal TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  ); |

1. **Tambahkan Menu Buku Tamu**

Edit file Latihan\_09\_menu.php dan tambahkan tautan ke halaman buku tamu:

|  |
| --- |
| <aside class="col-md-2 p-0">  <nav class="nav flex-column bg-light p-3 m-0">  <a class="nav-link" href="?menu=home">Home</a>  <a class="nav-link" href="?menu=alumni">Alumni</a>  <a class="nav-link" href="?menu=bukutamu">Buku Tamu</a>  </nav>  </aside> |

1. Buat File Latihan\_09\_bukuTamu.php

|  |
| --- |
| <?php  include 'Latihan\_09\_config.php'; // Koneksi database  // Proses form jika ada data yang dikirim  if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {  $nama = $\_POST['nama'];  $email = $\_POST['email'];  $pesan = $\_POST['pesan'];  // Validasi input  if (!empty($nama) && !empty($pesan)) {  $sql = "INSERT INTO bukutamu (nama, email, pesan) VALUES ('$nama', '$email', '$pesan')";  if ($conn->query($sql) === TRUE) {  echo "<div class='alert alert-success'>Pesan berhasil disimpan.</div>";  } else {  echo "<div class='alert alert-danger'>Error: " . $conn->error . "</div>";  }  } else {  echo "<div class='alert alert-danger'>Nama dan Pesan wajib diisi.</div>";  }  }  ?>  <div class="container mt-5">  <h2>Buku Tamu</h2>  <form method="POST" action="">  <div class="mb-3">  <label for="nama" class="form-label">Nama:</label>  <input type="text" class="form-control" id="nama" name="nama" required>  </div>  <div class="mb-3">  <label for="email" class="form-label">Email (opsional):</label>  <input type="email" class="form-control" id="email" name="email">  </div>  <div class="mb-3">  <label for="pesan" class="form-label">Pesan:</label>  <textarea class="form-control" id="pesan" name="pesan" rows="4" required></textarea>  </div>  <button type="submit" class="btn btn-primary">Kirim</button>  </form>  <hr>  <h3>Daftar Buku Tamu</h3>  <?php  // Menampilkan daftar buku tamu  $sql = "SELECT \* FROM bukutamu ORDER BY tanggal DESC";  $result = $conn->query($sql);  if ($result->num\_rows > 0) {  echo "<ul class='list-group'>";  while ($row = $result->fetch\_assoc()) {  echo "<li class='list-group-item'>  <strong>" . htmlspecialchars($row['nama']) . "</strong> (" . htmlspecialchars($row['email']) . ")<br>  <small>" . $row['tanggal'] . "</small><br>  " . nl2br(htmlspecialchars($row['pesan'])) . "  </li>";  }  echo "</ul>";  } else {  echo "<p class='alert alert-warning'>Belum ada pesan.</p>";  }  $conn->close();  ?>  </div> |

1. Update Logika di Latihan\_09\_index.php

Tambahkan kondisi untuk membuat halaman buku tamu di dalam file Latihan\_09\_index.php

|  |
| --- |
| else if($menu == "bukutamu") {  @include "Latihan\_09\_bukutamu.php";  } |

Analisis :

Fitur Buku Tamu yang Ditambahkan :

1. Form Input Buku Tamu:  
   Pengguna dapat mengisi nama, email (opsional), dan pesan.
2. Validasi Input:  
   Nama dan pesan wajib diisi untuk mencegah data kosong.
3. Menampilkan Pesan:  
   Semua pesan yang telah dikirimkan akan ditampilkan di halaman buku tamu dengan format rapi.
4. Keamanan:  
   Data yang ditampilkan difilter dengan htmlspecialchars untuk mencegah serangan XSS.

# **F. LATIHAN / TUGAS**

1. Lengkapi portal alumni dengan menambahkan fitur bursa kerja dan penelusuran alumni.

|  |
| --- |
| <script>          document.getElementById('searchAlumni').addEventListener('input', function(event) {              const searchValue = event.target.value.toLowerCase();              console.log('Searching for:', searchValue);          });      </script> |

Fungsi penelusuran masih sangat dasar, hanya mencatat input pencarian di konsol. Fitur penelusuran lebih lanjut dapat dikembangkan, misalnya dengan mengambil data alumni dari sebuah API atau database.

2. Simpan seluruh praktikum pada modul sebelumnya kedalam repository git yang sudah anda buat.