LAPORAN PRAKTIKUM LAB D1
PENGEMBANGAN BERBASIS PLATFORM
"CRUD"

DISUSUN OLEH:
RESMA ADI NUGROHO
24060121120021

DEPARTEMEN INFORMATIKA
FAKULTAS SAINS DAN MATEMATIKA
UNIVERSITAS DIPONEGORO
2023

BAB I

PENDAHULUAN

1.1 Tujuan

a. Melakukan koneksi ke basis data MySQL dan menampilkan hasil query select ke halaman web.

b. Membuat halaman untuk CRUD (create, read, update, delete) data dari/ ke basis data.

c. Membuat file untuk login dan logout serta menambahkan penanganan pada file yang hanya

dapat dibuka setelah login terlebih dahulu.

d. Menerapkan penggunaan client-state management (session) untuk membuat shopping cart.

1.2 Rumusan Masalah

1. Buatlah file untuk CRUD data berikut:
   - view_customer.php, untuk menampilkan data pelanggan
   - edit_customer.php, untuk edit data pelanggan
   - add_customer.php, untuk menambah data customer baru
   - delete_customer.php, untuk menghapus data customer tertentu
2. Buatlah file untuk login seperti berikut
   - login.php seperti pada contoh di modul ini, jika berhasil redirect ke halaman
   - view_customer.php.
   - Tambahkan penanganan pada file view_customer.php sehingga file tersebut hanya dibuka
   - oleh admin setelah berhasil login.
   - Tambahkan link logout pada file view_customer.php untuk melakukan logout, lalu buatlah file logout.php seperti dalam modul
3. Buatlah file-file berikut untuk memproses shopping cart seperti yang dicontohkan di modul
   - view_books.php
   - show_cart.php
   - delete_cart.php

BAB II

DASAR TEORI

2.1 MySQL

MySQL adalah sistem manajemen basis data relasional (RDBMS) yang sangat populer dan bersifat open-source. MySQL digunakan untuk menyimpan, mengelola, dan mengakses data dalam basis data yang terstruktur. Sistem ini sering dipilih untuk mengembangkan aplikasi web dan aplikasi bisnis karena kinerjanya yang cepat, keamanan yang baik, dan kemudahan penggunaannya.

Beberapa situs web terkenal seperti Facebook, Twitter, dan YouTube telah menggunakan MySQL untuk mengelola jutaan bahkan miliaran catatan data. MySQL mendukung bahasa SQL (Structured Query Language) yang memungkinkan pengembang untuk melakukan operasi seperti mengambil data, memasukkan data baru, memperbarui data yang ada, dan menghapus data. MySQL juga menyediakan berbagai fitur seperti indeks, transaksi, dan keamanan akses yang kuat.

Salah satu keunggulan MySQL adalah skalabilitasnya. MySQL mampu menangani jumlah data yang besar dan dapat menyesuaikan dengan kebutuhan yang semakin berkembang pada aplikasi atau situs web. MySQL juga menyediakan ketersediaan dan keandalan yang tinggi melalui fitur replikasi, yang memungkinkan data disalin ke beberapa server untuk keamanan data yang lebih baik.

Fitur lain yang menonjol dari MySQL adalah dukungannya terhadap berbagai mesin penyimpanan. Mesin-mesin penyimpanan ini menyediakan kemampuan dan optimasi yang berbeda, sehingga pengembang dapat memilih yang paling sesuai dengan kebutuhan mereka. Beberapa mesin penyimpanan populer termasuk InnoDB, MyISAM, dan Memory.

Selain itu, MySQL memiliki komunitas pengembang dan pengguna yang aktif, yang berkontribusi pada pengembangan terus-menerus dan memberikan dukungan melalui forum, dokumentasi, dan sumber daya online. Komunitas yang aktif ini memastikan bahwa MySQL tetap terkini dengan perkembangan terbaru dalam teknologi basis data.

MYSQL adalah RDBMS yang kuat dan banyak digunakan, dikenal karena kinerjanya, keamanannya, dan kemudahannya dalam penggunaan. Kemampuannya dalam menangani jumlah data yang besar, dukungan terhadap SQL, skalabilitas, dan beragam mesin penyimpanan menjadikannya pilihan yang disukai oleh pengembang dalam membangun aplikasi web dan bisnis. Komunitas yang aktif di sekitar MySQL memastikan pengembangan dan dukungan yang berkelanjutan, menjadikannya pilihan yang dapat diandalkan dalam mengelola data terstruktur.

2.2 CRUD

CRUD adalah singkatan dari Create, Read, Update, dan Delete. Ini adalah operasi dasar yang digunakan dalam pengelolaan data dalam basis data.

- Operasi Create (Buat) digunakan untuk membuat data baru dalam basis data. Dalam operasi ini, perintah SQL INSERT digunakan untuk menambahkan data baru ke dalam tabel. Misalnya, Anda dapat menggunakan operasi Create untuk menambahkan informasi pengguna baru ke dalam tabel pengguna.
- Operasi Read (Baca) digunakan untuk membaca atau mengambil data yang sudah ada dari basis data. Dalam operasi ini, perintah SQL SELECT digunakan untuk mengambil data dari tabel. Misalnya, Anda dapat menggunakan operasi Read untuk mengambil semua informasi pengguna dari tabel pengguna.
- Operasi Update (Perbarui) digunakan untuk memperbarui data yang sudah ada dalam basis data. Dalam operasi ini, perintah SQL UPDATE digunakan untuk mengubah nilai-nilai dalam tabel. Misalnya, Anda dapat menggunakan operasi Update untuk mengubah alamat email pengguna yang sudah ada.
- Operasi Delete (Hapus) digunakan untuk menghapus data dari basis data. Dalam operasi ini, perintah SQL DELETE digunakan untuk menghapus baris data dari tabel. Misalnya, Anda dapat menggunakan operasi Delete untuk menghapus pengguna yang tidak aktif lagi dari tabel pengguna.

Operasi CRUD sangat penting dalam pengembangan aplikasi karena memungkinkan pengelolaan data dengan efisien. Dengan menggunakan operasi CRUD, pengembang dapat membuat, membaca, memperbarui, dan menghapus data dalam basis data dengan

mudah. Hal ini memungkinkan aplikasi untuk berinteraksi dengan basis data secara efektif dan mengelola informasi dengan baik.

Dalam pengembangan aplikasi, penggunaan operasi CRUD harus diperhatikan dengan baik. Pengembang harus memastikan bahwa operasi Create, Read, Update, dan Delete diimplementasikan dengan benar dan aman. Selain itu, penggunaan parameter dan validasi data juga penting untuk mencegah kesalahan dan melindungi integritas data dalam basis data. Dengan memahami dan menguasai operasi CRUD, pengembang dapat mengoptimalkan pengelolaan data dalam aplikasi mereka.

2.3 SESSION

Session (sesi) adalah mekanisme yang digunakan dalam pengembangan web untuk mengelola informasi yang berhubungan dengan pengguna selama periode waktu tertentu saat mereka berinteraksi dengan sebuah situs web.

Sesi biasanya digunakan untuk menyimpan data yang perlu dipertahankan antara permintaan dan tanggapan, seperti data login pengguna atau keranjang belanja dalam belanja online. Data sesi disimpan di sisi server dan diidentifikasi oleh pengenal unik yang disebut session ID yang dikirimkan ke browser pengguna. Browser kemudian mengirimkan session ID ini kembali ke server dalam setiap permintaan, memungkinkan server untuk mengidentifikasi pengguna yang sesuai dan mengakses data sesi mereka.

Ini adalah komponen penting dalam pengembangan web yang memungkinkan aplikasi untuk menyimpan dan mengelola status atau informasi pengguna di seluruh berbagai permintaan. Sesi digunakan untuk mengidentifikasi pengguna, melacak keadaan login, dan menyimpan preferensi atau data penting lainnya yang dibutuhkan selama sesi pengguna di situs web

BAB III

PEMBAHASAN

1.1 CRUD Tabel Customers

File view_customers.php

```
<?php
session_start();
if(!isset($_SESSION['username'])){
    header('Location: login.php');
}
?>
<?php include_once('components/header.html'); ?>
    <div class="card">
        <div class="card-header">Customer Data</div>
        <div class="card-body">
            <br>
            <a class="btn btn-primary" href="add_customer.php">+
Add Customer Data</a>
            <a class="btn btn-primary"
href="logout.php">Logout</a><br/><br/>
            <table class="table table-striped">
                <tr>
                    <th>No</th>
                    <th>Name</th>
                    <th>Address</th>
                    <th>City</th>
                    <th>Action</th>
                </tr>
                <?php
                require_once('lib/db_login.php');
                $query = " SELECT * FROM customers ORDER BY
customerid";
                $result = $db->query($query);
                if(!$result){
                    die("Could not query the database: <br/>".$db-
>error."<br/>Query: ".$query);
                }
                // feth data
                $i = 1;
                while($row = $result->fetch_object()){
                    echo '<tr>';
                    echo '<td>'.$i.'</td>';
                    echo '<td>'.$row->name.'</td>';
                    echo '<td>'.$row->address.'</td>';
                    echo '<td>'.$row->city.'</td>';
                    echo '<td><a class="btn btn-warning btn-sm"
href="edit_customer.php?id='.$row->customerid.'">Edit</a>
                     <a class="btn btn-danger btn-sm"
href="delete_customer.php?id='.$row-
>customerid.'">Delete</a></td>';
                    echo '</tr>';
                    $i++;
                }
                echo '</table>';
                echo '<br/>';
```

```
                echo 'Total Rows = '.$result->num_rows;
                $result->free();
                $db->close();
                ?>
            </table>
        </div>
    </div>
<?php include_once('components/footer.html'); ?>
```

Kode diatas merupakan halaman untuk menampilkan list data customer yang diambil dari data table customers pada database MySQL. Sebelum pengguna dapat masuk ke halaman ini mereka diperlukan untuk login terlebih dahulu kemudian akan masuk ke halaman view customer ini. Pada awal kode akan diberlakukan sebuah session untuk mengecek apakah user sudah login. Selanjutnya untuk bagian kode tersebut akan lebih dijelaskan pada bagian nomer 2. Untuk dapat menampilkan halaman yang berisi list view data customer maka pertama diperlukan untuk menimport atau include file header yang didalamnya berisi link style bootstrap dan title serta beberapa kode lain yang diperlukan saat load halaman contohnya seperti kode javascript yang diperlukan dan kode lain. Setelah include file header maka dibawahnya dapat kita buat sebuah list view customer dalam bentuk tabel. Agar tabel dapat terlihat lebih bagus maka kita perlu untuk menambahkan sebuah div card yang stylingnya akan diambil dari bootstrap. Kemudian untuk bagian atasnya akan terdapat button add customer yang merupakan element link atau dalam tag html yakni a atau anchor yang mengarahkan ke halaman add_customer.php. Kemudian disamping tombol add tersebut terdapat link untuk logout yang akan mengarahkan ke halaman logout.php. Lalu dibawah dari tombol add dan logout akan terdapat list tabel yang berisi kolom no, nama, address, city dan action. Untuk masing-masing kolom tersebut data akan diambil dari database dengan kode php pada tepat dibawah kode tag tr tersebut. Pertama sebelum kita dapat mengakses data dari database, pertama diperlukan sebuah file koneksi antara php dan database yakni dengan menggunakan file yang sebelumnya sudah pernah dibuat yang bernama db_login.php. Setelah berhasil terkoneksi dengan database maka langkah selanjutnya adalah melakukan query untuk mendapatkan data customer tersebut dengan perintah SQL. Untuk perintahnya yakni SELECT * FROM customers ORDER BY customerid. Perintah ini akan mendapatkan seluruh data dari tabel customers yang diurutkan berdasarkan customeridnya. Setelah data didapatkan maka data dapat ditampilkan dalam tabel. Karena dalam hal ini data bersifat multi row maka kita perlu melakukan looping untuk masing-

masing barisnya sehingga semua data akan dapat tampil ke dalam tabel tersebut. Dalam hal ini digunakan sebuah while looping dengan kondisi selama data tersebut berada pada hasil query database. Untuk didalam loopingnya akan ditampilkan untuk baris baru dalam tabel yakni dengan tag tr dan untuk isinya digunakan tag td, dengan masing-masing datanya diakses dari variable $row atau data yang didapatkan dari hasil query sebelumnya. Untuk kolom action juga akan menampilkan tombol edit dan delete yang akan mengarahkan ke edit_customer.php dengan query idnya berasal dari customerid dan delete yang akan mengarahkan ke delete_customer.php dengan query idnya yang sama berasal dari customerid. Query untuk link tersebut diperlukan sebagai identifier untuk customer mana yang akan diedit datanya atau yang akan didelete. Kemudian diakhir tabel akan ditampilkan pula jumlah row atau barisnya. Dan setelah kita melakukan koneksi jangan lupa untuk menutup koneksi database yang telah dibuka sebelumnya. Kemudian pada akhir kode terdapat include file footer yakni untuk menampilkan komponen footer.

File edit_customer.php

```
<?php

// TODO 1: Lakukan koneksi dengan database
require_once('lib/db_login.php');
// TODO 2: Buat variabel $id yang diambil dari query string
parameter
$id = $_GET['id'];
// Memeriksa apakah user belum menekan tombol submit
if (!isset($_POST["submit"])) {
    // TODO 3: Tulislah dan eksekusi query untuk mengambil
informasi customer berdasarkan id
    $query = "SELECT * FROM customers WHERE customerid = '" . $id .
"'";
    $result = $db->query($query);
    if (!$result) {
        die("Could not query the database: <br/>" . $db->error .
"<br/>Query: " . $query);
    } else {
        while ($row = $result->fetch_object()) {
            $name = $row->name;
            $address = $row->address;
            $city = $row->city;
        }
    }

} else {
    $valid = TRUE;
    $name = test_input($_POST['name']);

    // Validasi terhadap field name
    if ($name == '') {
```

```
        $error_name = "Name is required";
        $valid = FALSE;
    } else if (!preg_match("/^[a-zA-Z ]*$/", $name)) {
        $error_name = "Only letters and white space allowed";
        $valid = FALSE;
    }

    // Validasi terhadap field address
    $address = test_input($_POST['address']);
    if ($address == '') {
        $error_address = "Address is required";
        $valid = FALSE;
    }

    // Validasi terhadap field city
    $city = $_POST['city'];
    if ($city == '' || $city == 'none') {
        $error_city = "City is required";
        $valid = FALSE;
    }

    // Update data into database
    if ($valid) {
        // TODO 4: Jika valid, update data pada database dengan
mengeksekusi query yang sesuai
        $query = "UPDATE customers SET name = '" . $name . "',
address = '" . $address . "', city = '" . $city . "' WHERE
customerid = '" . $id . "'";
        $result = $db->query($query);
        if (!$result) {
            die("Could not query the database: <br/>" . $db->error
. "<br/>Query: " . $query);
        } else {
            $db->close();
            header('Location: view_customer.php');
        }
    }
}
?>
<?php include('components/header.html') ?>
<br>
<div class="card mt-4">
    <div class="card-header">Edit Customers Data</div>
    <div class="card-body">
        <form action="<?= htmlspecialchars($_SERVER['PHP_SELF']) .
'?id=' . $id ?>" method="POST" autocomplete="on">
            <div class="form-group">
                <label for="name">Nama:</label>
                <input type="text" class="form-control" id="name"
name="name" value="<?= $name; ?>">
                <div class="error"><?php if (isset($error_name))
echo $error_name ?></div>
            </div>
            <div class="form-group">
                <label for="name">Address:</label>
                <textarea class="form-control" name="address"
id="address" rows="5"><?php echo $address; ?></textarea>
                <div class="error"><?php if (isset($error_address))
echo $error_address ?></div>
            </div>
            <div class="form-group">
                <label for="city">City:</label>
                <select name="city" id="city" class="form-control"
required>
```

```
      <option value="none" <?php if (!isset($city)) echo 'selected'
?>>--Select a city--</option>
                    <option value="Airport West" <?php if
(isset($city) && $city == "Airport West") echo 'selected'
?>>Airport West</option>
                    <option value="Box Hill" <?php if (isset($city)
&& $city == "Box Hill") echo 'selected' ?>>Box Hill</option>
                    <option value="Yarraville" <?php if
(isset($city) && $city == "Yarraville") echo 'selected'
?>>Yarraville</option>
                </select>
                <div class="error"><?php if (isset($error_city))
echo $error_city ?></div>
            </div>
            <br>
            <button type="submit" class="btn btn-primary"
name="submit" value="submit">Submit</button>
            <a href="view_customer.php" class="btn btn-
secondary">Cancel</a>
        </form>
    </div>
</div>
<?php include('components/footer.html') ?>
<?php
$db->close();
?>
```

Dari kode diatas dapat dirincikan atau dijelaskan untuk masing-masing komponen kodenya yakni:

- Koneksi Database: Kode pertama adalah untuk menghubungkan ke database MySQL. Ini dilakukan dengan mengimpor file db_login.php, yang mungkin berisi informasi seperti nama pengguna, kata sandi, dan nama database yang diperlukan untuk koneksi.
- Mengambil Parameter ID: Kode selanjutnya mengambil nilai id dari parameter query string. Nilai ini digunakan untuk mengidentifikasi pelanggan yang akan diedit.
- Pemeriksaan Aksi: Kode kemudian memeriksa apakah pengguna telah mengirimkan formulir (melalui tombol submit) atau belum. Jika tidak, maka ia akan mengeksekusi kode untuk mengambil informasi pelanggan dari database berdasarkan id yang diberikan.
- Validasi Data: Jika pengguna telah mengirimkan formulir, maka data yang dimasukkan akan divalidasi. Ini termasuk validasi nama (hanya huruf dan spasi

diperbolehkan), alamat, dan kota yang dipilih. Setiap kesalahan validasi akan menyebabkan pesan kesalahan yang sesuai ditampilkan.

- Pembaruan Data: Jika semua data yang dimasukkan valid, maka kode akan melakukan pembaruan data ke dalam database. Ini dilakukan dengan menjalankan query SQL UPDATE yang sesuai dengan data yang dimasukkan.
- Tampilan Formulir: Kode juga menghasilkan sebuah formulir yang memungkinkan pengguna untuk mengedit data pelanggan. Data pelanggan yang saat ini ada dalam database ditampilkan dalam formulir ini. Pengguna dapat mengubah data tersebut dan mengirimkan perubahan dengan menekan tombol "Submit".
- Penanganan Error: Jika ada kesalahan dalam validasi data atau dalam eksekusi query database, pesan kesalahan yang relevan akan ditampilkan di sebelah bidang input yang bermasalah.
- Tombol Cancel: Pengguna memiliki opsi untuk membatalkan proses pengeditan dengan menekan tombol "Cancel". Ini akan mengarahkannya kembali ke halaman yang menampilkan daftar pelanggan.
- Penutup: Kode di bawahnya adalah kode untuk menutup koneksi ke database setelah semua operasi selesai.

File add_customer.php

```
<?php
require_once('lib/db_login.php');
if (isset($_POST["submit"])) {
    $valid = TRUE;
    $name = test_input($_POST['name']) ?? "";

    // Validasi terhadap field name
    if ($name == '') {
        $error_name = "Name is required";
        $valid = FALSE;
    } else if (!preg_match("/^[a-zA-Z ]*$/", $name)) {
        $error_name = "Only letters and white space allowed";
        $valid = FALSE;
    }

    // Validasi terhadap field address
    $address = test_input($_POST['address']);
    if ($address == '') {
        $error_address = "Address is required";
        $valid = FALSE;
    }

    // Validasi terhadap field city
    $city = $_POST['city'];
    if ($city == '' || $city == 'none') {
```

```
        $error_city = "City is required";
        $valid = FALSE;
    }

    // Update data into database
    if ($valid) {
        // TODO 4: Jika valid, update data pada database dengan
mengeksekusi query yang sesuai
        $query = "INSERT INTO customers (name, address, city)
VALUES (?, ?, ?)";

        // Persiapkan statement SQL
        $stmt = $db->prepare($query);

        // Periksa apakah persiapan statement berhasil
        if (!$stmt) {
            die("Gagal mempersiapkan statement: " . $db->error);
        }

        // Bind parameter ke placeholder
        $stmt->bind_param("sss", $name, $address, $city);

        // Eksekusi statement
        if ($stmt->execute()) {
            echo "Data berhasil dimasukkan.";
            $db->close();
            header('Location: view_customer.php');
        } else {
            echo "Gagal memasukkan data: " . $stmt->error;
        }
    }
}
?>
<?php include('components/header.html') ?>
<br>
<div class="card mt-4">
    <div class="card-header">Add Customers Data</div>
    <div class="card-body">
        <form method="POST" autocomplete="on" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]); ?>">
            <div class="form-group">
                <label for="name">Nama:</label>
                <input type="text" class="form-control" id="name"
name="name" value="">
                <div class="error"><?php if (isset($error_name))
echo $error_name ?></div>
            </div>
            <div class="form-group">
                <label for="name">Address:</label>
                <textarea class="form-control" name="address"
id="address" rows="5"></textarea>
                <div class="error"><?php if (isset($error_address))
echo $error_address ?></div>
            </div>
            <div class="form-group">
                <label for="city">City:</label>
                <select name="city" id="city" class="form-control"
required>
                    <option value="none" <?php if (!isset($city))
echo 'selected' ?>>--Select a city--</option>
```

```
      <option value="Airport West" <?php if (isset($city) && $city
== "Airport West") echo 'selected' ?>>Airport West</option>
                    <option value="Box Hill" <?php if (isset($city)
&& $city == "Box Hill") echo 'selected' ?>>Box Hill</option>
                    <option value="Yarraville" <?php if
(isset($city) && $city == "Yarraville") echo 'selected'
?>>Yarraville</option>
                </select>
                <div class="error"><?php if (isset($error_city))
echo $error_city ?></div>
            </div>
            <br>
            <button type="submit" class="btn btn-primary"
name="submit" value="submit">Submit</button>
            <a href="view_customer.php" class="btn btn-
secondary">Cancel</a>
        </form>
    </div>
</div>
<?php include('components/footer.html') ?>
<?php
$db->close();
?>
```

Dari kode diatas dapat dirincikan atau dijelaskan untuk masing-masing komponen kodenya yakni:

- Koneksi Database: Kode pertama adalah untuk menghubungkan ke database MySQL dengan menggunakan informasi yang ada di db_login.php. Ini adalah langkah awal yang penting sebelum berinteraksi dengan database.
- Validasi Formulir: Kode kemudian memeriksa apakah pengguna telah mengirimkan formulir dengan menekan tombol "Submit". Jika ya, maka kode akan melanjutkan untuk memproses data yang dikirim.
- Validasi Data: Kode melakukan validasi terhadap data yang dikirimkan melalui formulir. Ini termasuk validasi terhadap nama (hanya huruf dan spasi yang diperbolehkan), alamat, dan kota yang dipilih. Setiap kesalahan validasi akan menyebabkan pesan kesalahan yang sesuai ditampilkan di sebelah bidang input yang bermasalah.
- Pembaruan Database: Jika semua data yang dimasukkan valid, maka kode akan melakukan pembaruan data ke dalam database. Ini dilakukan dengan menjalankan query SQL INSERT yang memasukkan data baru ke dalam tabel pelanggan.
- Prepare Statement SQL: Sebelum eksekusi query, kode mempersiapkan sebuah statement SQL. Hal ini dilakukan dengan menggunakan prepared statement untuk

menghindari SQL injection. Parameter di-bind ke placeholder dalam prepared statement.

- Eksekusi Statement: Setelah persiapan statement selesai, kode menjalankan statement tersebut dengan perintah $stmt->execute(). Jika eksekusi berhasil, pesan sukses akan ditampilkan, dan pengguna akan diarahkan kembali ke halaman yang menampilkan daftar pelanggan. Jika terjadi kesalahan, pesan kesalahan akan ditampilkan.
- Tampilan Formulir: Kode menghasilkan sebuah formulir yang memungkinkan pengguna untuk memasukkan data pelanggan. Formulir ini termasuk bidang untuk nama, alamat, dan pilihan kota. Kesalahan validasi, jika ada, akan ditampilkan di bawah bidang yang bermasalah.
- Tombol Cancel: Pengguna memiliki opsi untuk membatalkan proses penambahan data dengan menekan tombol "Cancel". Ini akan mengarahkannya kembali ke halaman yang menampilkan daftar pelanggan.
- Penutup: Kode di bawahnya adalah kode untuk menutup koneksi ke database setelah semua operasi selesai.

File delete_customer.php

```
<?php
require_once('lib/db_login.php');
$id = $_GET['id'];

$query = "DELETE FROM customers WHERE customerid = '" . $id . "'";
$result = $db->query($query);
if (!$result) {
    die("Could not query the database: <br/>" . $db->error .
"<br/>Query: " . $query);
} else {
    $db->close();
    header('Location: view_customer.php');
}
?>
<?php include('components/header.html') ?>
<?php include('components/footer.html') ?>
```

Dari kode diatas dapat dirincikan atau dijelaskan untuk masing-masing komponen kodenya yakni:

- Koneksi Database: Kode pertama adalah untuk menghubungkan ke database MySQL dengan menggunakan informasi yang ada di db_login.php. Ini adalah langkah awal yang penting sebelum berinteraksi dengan database.
- Mengambil Parameter ID: Kode kemudian mengambil ID pelanggan yang dikirimkan melalui parameter URL dengan menggunakan $_GET['id']. Parameter ini digunakan untuk menentukan pelanggan mana yang akan dihapus dari database.
- Query Hapus: Kode menjalankan query SQL DELETE untuk menghapus data pelanggan dari database. Query ini dibuat dengan mengidentifikasi tabel (customers) dan kriteria penghapusan berdasarkan ID yang sesuai dengan nilai yang diterima dari parameter URL.
- Eksekusi Query: Setelah query DELETE dibuat, kode menjalankannya dengan perintah $db->query($query). Jika penghapusan berhasil, kode akan menutup koneksi ke database dan mengarahkan pengguna kembali ke halaman yang menampilkan daftar pelanggan. Namun, jika terjadi kesalahan dalam menjalankan query, pesan kesalahan akan ditampilkan.
- Tampilan Header dan Footer: Kode memasukkan file header.html dan footer.html. Ini adalah bagian umum dari halaman web yang bisa digunakan kembali di berbagai halaman.

2. Login Page and Session

```
<?php
session_start();
require_once('lib/db_login.php');

if(isset($_POST["submit"])){
    $valid = true;
    $email = test_input($_POST['email']);
    if($email == ''){
        $error_email = "Email is required";
        $valid = false;
    }elseif(!filter_var($email, FILTER_VALIDATE_EMAIL)){
        $error_email = "Invalid email format";
        $valid = false;
    }
    $password = test_input($_POST['password']);
    if($password == ''){
        $error_password = "Password is required";
        $valid = false;
    }
    if($valid){
        $query = "SELECT * FROM admin WHERE email = '".$email."'
AND password = '".md5($password)."'";
```

```
        $result = $db->query($query);
        if(!$result){
            die("Could not query the database: <br/>".$db-
>error."<br/>Query: ".$query);
        }else{
            if($result->num_rows > 0){
                $_SESSION['username'] = $email;
                header('Location: view_customer.php');
            }else{
                echo '<span class="error">Email or Password is
invalid</span>';
            }
        }
        $db->close();
    }
}

?>
<?php include("components/header.html") ?>
<br>
<div class="card">
    <div class="card-header">Login Form</div>
    <div class="card-body">
        <form method="POST" autocomplete="on" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"]); ?>">
            <div class="form-group">
                <label for="email">Email:</label>
                <input type="email" class="form-control" id="email"
placeholder="Enter email" size="30" name="email" value="<?php
if(isset($email)) echo $email; ?>">
                <span class="error"><?php if(isset($error_email))
echo $error_email; ?></span>
            </div>
            <div class="form-group">
                <label for="password">Password:</label>
                <input type="password" class="form-control"
id="password" placeholder="Enter password" name="password"
value="">
                <span class="error"><?php
if(isset($error_password)) echo $error_password; ?></span>
            </div>
            <button type="submit" class="btn btn-primary"
name="submit" value="submit">Login</button>
        </form>
</div>
<?php include("components/footer.html") ?>
```

Dari kode diatas dapat dirincikan atau dijelaskan untuk masing-masing komponen kodenya yakni:

- Memulai Sesi: Kode diawali dengan session_start() yang digunakan untuk memulai sesi. Ini adalah langkah penting karena informasi login akan disimpan dalam sesi pengguna.

- Koneksi Database: Kode selanjutnya menghubungkan ke database MySQL menggunakan informasi yang ada di db_login.php. Ini diperlukan untuk melakukan verifikasi login terhadap data yang ada di database.
- Validasi Input: Kode melakukan validasi terhadap input yang diterima dari formulir login. Ini dilakukan dengan langkah-langkah berikut:
- Input email ($email) diperiksa jika tidak kosong, dan apakah format email yang diterima sesuai dengan menggunakan filter_var() dengan filter FILTER_VALIDATE_EMAIL.
- Input password ($password) diperiksa jika tidak kosong.
- Query Database: Jika input valid (email dan password terisi dengan benar), maka kode akan menjalankan query SQL SELECT untuk mencocokkan email dan password yang dimasukkan dengan data di database tabel admin. Password yang dimasukkan dienkripsi menggunakan fungsi md5() untuk mencocokkannya dengan yang ada di database.
- Eksekusi Query: Setelah query dijalankan, kode memeriksa apakah hasil query mengembalikan satu atau lebih baris. Jika ya, maka login berhasil, dan informasi email pengguna disimpan dalam sesi dengan $_SESSION['username']. Pengguna akan dialihkan ke halaman view_customer.php, yang mungkin merupakan halaman beranda setelah login.
- Tampilan Form Login: Kode menciptakan sebuah formulir HTML yang memungkinkan pengguna untuk memasukkan email dan password. Jika terdapat kesalahan dalam validasi input, pesan kesalahan akan ditampilkan di samping bidang yang sesuai.
- Tampilan Header dan Footer: Kode memasukkan file header.html dan footer.html, yang umumnya berisi struktur halaman yang digunakan secara berulang dalam proyek web.
- Penutup Koneksi: Terakhir, setelah selesai menggunakan database, koneksi database ditutup dengan $db->close().

Sedangkan untuk cuplikan kode yang menunjukan abhwa view_customer.php hanya dapat dilihat oleh admin yang berhasil login adalah:

```
session_start();
if(!isset($_SESSION['username'])){
    header('Location: login.php');
}
```

Untuk memberikan sebuah ketentuan akses pada suatu halaman kita dapat menggunakan session, dengan mengecek apakah terdapat session pada kode diatas maka untuk mengakses halaman tersebut perlu login terlebih dahulu. Jika tidak terdapat akses atau tidak ada session maka pengguna akan diarahkan ke halaman login.

Kemudian agar pengguna bisa untuk melakukan logout maka diperlukan sebuah button seperti pada penjelasan sebelumnya yang akan mengarahkan ke dalam file logout.php yang berisi kode seperti berikut.

```
<?php
// TODO 1: Inisialisasi session
session_start();
// TODO 2: Hapus username session
if(isset($_SESSION['username'])){
    unset($_SESSION['username']);
    session_destroy();
}
// TODO 3: Redirect ke halaman login
header('Location: login.php');
?>
```

Pada kode diatas akan menghapus session yang sebelumnya telah tersimpan ketika login. Sehingga apabila pengguna telah logout maka pengguna akan diarahkan Kembali ke halaman login dan pengguna tidak akan bisa mengakses halaman yang memperlukan akses admin.

3. Halaman Shopping Cart

File view_books.php

```
<?php include('components/header.html') ?>
<div class="card mt-5">
    <div class="card-header">Data Buku</div>
    <div class="card-body">
        <table class="table table-striped">
            <tr>
                <th>ISBN</th>
                <th>Author</th>
                <th>Title</th>
                <th>Price</th>
                <th>Action</th>
```

```
            </tr>
            <?php
            // Include our login information
            require_once('lib/db_login.php');

            // TODO 1: Tuliskan dan eksekusi query
            $query = "SELECT * FROM books ORDER BY isbn";
            $result = $db->query($query);
            // Fetch and display the results
            $i = 1;
            while ($row = $result->fetch_object()) {
                echo '<tr>';
                echo '<td>' . $row->isbn . '</td>';
                echo '<td>' . $row->author . '</td>';
                echo '<td>' . $row->title . '</td>';
                echo '<td>$' . $row->price . '</td>';
                echo '<td><a class="btn btn-primary btn-sm"
href="show_cart.php?id=' . $row->isbn . '">Add to Cart</a></td>';
                echo '</tr>';
                $i++;
            }
            echo '</table>';
            echo '<br />';
            echo 'Jumlah Buku: ' . $result->num_rows;

            $result->free();
            $db->close();
            ?>
    </div>
</div>
<?php include('components/footer.html') ?>
```

Dari kode diatas dapat dirincikan atau dijelaskan untuk masing-masing komponen kodenya yakni:

- Include Header: Kode memulai dengan memasukkan konten dari file header.html. Ini adalah langkah umum dalam pengembangan web untuk menyertakan elemen-elemen yang sama pada setiap halaman, seperti header, menu navigasi, atau stylesheet.
- Elemen Card: Kode selanjutnya menciptakan sebuah elemen "card" yang digunakan untuk mengelompokkan dan menampilkan data buku. Card ini memiliki judul "Data Buku" yang ditampilkan di bagian atasnya.
- Tabel Data: Di dalam card, sebuah tabel HTML dibuat dengan class "table" dan "table-striped" dari Bootstrap. Tabel ini digunakan untuk menampilkan data buku dalam beberapa kolom. Kolom-kolom yang ditampilkan adalah:
    - ISBN
    - Author (Penulis)

    - Title (Judul)
    - Price (Harga)
    - Action (Aksi)
- Koneksi Database: Kode selanjutnya melakukan koneksi ke database MySQL dengan menggunakan informasi yang ada di file db_login.php. Ini diperlukan untuk mengambil data buku yang akan ditampilkan dalam tabel.
- Query Database: Kode menggambarkan proses eksekusi query SQL. Query ini digunakan untuk mengambil semua data buku dari tabel books dan diurutkan berdasarkan ISBN. Hasil query disimpan dalam variabel $result.
- Pengambilan Data: Hasil query diambil satu per satu menggunakan loop while. Setiap baris data buku akan ditampilkan dalam sebuah baris tabel. Data yang diambil adalah ISBN, Author, Title, dan Price. Aksi "Add to Cart" juga ditampilkan dalam bentuk tombol yang mengarahkan pengguna ke halaman "show_cart.php" dengan mengirimkan ISBN buku sebagai parameter.
- Penutup Tabel: Setelah semua data buku ditampilkan, tag penutup tabel (</table>) ditampilkan, dan di bawahnya jumlah buku yang ditampilkan juga ditampilkan.
- Penghapusan Hasil dan Penutup Koneksi: Setelah selesai menggunakan hasil query, hasil tersebut dibebaskan ($result->free()) untuk mengosongkan memori yang digunakan. Selanjutnya, koneksi ke database ditutup dengan $db->close().
- Include Footer: Kode akhirnya memasukkan konten dari file footer.html. Seperti header, footer ini biasanya berisi elemen-elemen yang sama di seluruh situs web, seperti hak cipta atau tautan ke halaman lain.

File show_cart.php

```
<?php
// File         : show_cart.php
// Deskripsi    : Untuk menambahkan item ke shopping cart dan
menampilkan isi shopping cart

session_start();
error_reporting(0);

$id = $_GET['id'];
if ($id != '') {
    if (!isset($_SESSION['cart'])) {
```

```
        $_SESSION['cart'] = array();
    }

    if (isset($_SESSION['cart'][$id])) {
        $_SESSION['cart'][$id]++;
    } else {
        $_SESSION['cart'][$id] = 1;
    }
}
?>
<?php include('components/header.html') ?>
<br>
<div class="card mt-4">
    <div class="card-header">Shopping Cart</div>
    <div class="card-body">
        <br>
        <table class="table table-striped">
            <tr>
                <th>ISBN</th>
                <th>Author</th>
                <th>Title</th>
                <th>Price</th>
                <th>Qty</th>
                <th>Price * Qty</th>
            </tr>
            <?php
            require_once('./lib/db_login.php');
            $sum_qty = 0;
            $sum_price = 0;

            if (is_array($_SESSION['cart'])) {
                foreach ($_SESSION['cart'] as $id => $qty) {

                    // TODO 1: Tuliskan dan eksekusi query
                    $query = "SELECT * FROM books WHERE isbn = '" .
$id . "'";
                    $result = $db->query($query);
                    while ($row = $result->fetch_object()) {
                        echo '<tr>';
                        echo '<td>' . $row->isbn . '</td>';
                        echo '<td>' . $row->author . '</td>';
                        echo '<td>' . $row->title . '</td>';
                        echo '<td>$' . $row->price . '</td>';
                        echo '<td>' . $qty . '</td>';
                        echo '<td>$' . $row->price * $qty .
'</td>';
                        echo '</tr>';

                        $sum_qty = $sum_qty + $qty;
                        $sum_price = $sum_price + ($row->price *
$qty);
                    }
                }
                echo
'<tr><td></td><td></td><td></td><td></td><td></td><td>$' .
$sum_price . '</td>';
                $result->free();
                $db->close();
            } else {
```

```
      echo '<tr><td colspan="6" align="center">There is no item in
shopping cart</td></tr>';
            }
            ?>
        </table>
        Total items = <?php echo $sum_qty ?><br><br>
        <a class="btn btn-primary" href="view_books.php">Continue
Shopping</a>
        <a class="btn btn-danger" href="delete_cart.php">Empty
Cart</a>
    </div>
</div>
<?php include('components/footer.html') ?>
```

Dari kode diatas dapat dirincikan atau dijelaskan untuk masing-masing komponen kodenya yakni:

- Include Header: Kode dimulai dengan memasukkan konten dari file header.html. Ini adalah cara umum untuk menyertakan elemen-elemen yang sama pada setiap halaman web, seperti header, menu navigasi, atau stylesheet.
- Inisialisasi Variabel: Kode menginisialisasi able le $id dengan nilai yang diperoleh dari parameter query string id dalam URL. Variabel ini digunakan untuk mengidentifikasi buku yang akan ditambahkan ke keranjang belanja.
- Inisialisasi Shopping Cart: Kode memeriksa apakah ada sesi (session) yang sudah dibuat untuk keranjang belanja. Jika belum ada, sebuah array kosong untuk keranjang belanja dibuat dan disimpan dalam sesi dengan nama “cart”.
- Penambahan Item ke Shopping Cart: Kode memeriksa apakah id sudah diberikan dalam URL. Jika iya, maka kode akan memproses penambahan item ke keranjang belanja. Jika item dengan id tersebut sudah ada dalam keranjang, maka kuantitasnya akan ditambahkan. Jika tidak, item tersebut akan ditambahkan ke keranjang dengan kuantitas awal 1.
- Elemen Card: Kode menciptakan elemen “card” untuk menampilkan isi dari keranjang belanja. Card ini memiliki judul “Shopping Cart” yang ditampilkan di bagian atasnya.
- Tabel Shopping Cart: Di dalam card, sebuah able HTML dibuat dengan class “table” dan “table-striped” dari Bootstrap. Tabel ini digunakan untuk menampilkan item-item dalam keranjang belanja. Kolom-kolom yang ditampilkan adalah:

    - ISBN
    - Author (Penulis)
    - Title (Judul)
    - Price (Harga per buku)
    - Qty (Kuantitas)
    - Price * Qty (Total harga untuk item tersebut)
- Koneksi Database: Kode melakukan koneksi ke database MySQL dengan menggunakan informasi yang ada di file db_login.php. Ini diperlukan untuk mengambil data buku yang akan ditampilkan dalam keranjang belanja.
- Query Database: Kode menggambarkan proses eksekusi query SQL untuk mengambil informasi buku yang ada dalam keranjang belanja. Setiap item dalam keranjang diambil dari database berdasarkan ISBN-nya, dan informasi tersebut ditampilkan dalam able.
- Penghitungan Total: Kode juga menghitung total jumlah item ($sum_qty) dan total harga ($sum_price) dalam keranjang belanja. Informasi ini ditampilkan di bagian bawah able.
- Pengosongan Hasil dan Penutup Koneksi: Setelah selesai menggunakan hasil query, hasil tersebut dibebaskan ($result->free()) untuk mengosongkan memori yang digunakan. Selanjutnya, koneksi ke database ditutup dengan $db->close().
- Link Lanjutan: Kode menampilkan dua tombol di bawah able: “Continue Shopping” untuk kembali ke halaman daftar buku dan “Empty Cart” untuk mengosongkan keranjang belanja.
- Include Footer: Kode akhirnya memasukkan konten dari file footer.html. Seperti header, footer ini biasanya berisi elemen-elemen yang sama di seluruh situs web, seperti hak cipta atau tautan ke halaman lain.

File delete_cart.php

```
<?php
// File         : delete_cart.php
// Deskripsi    : untuk menghapus session

// TODO 1: Inisialisasi data session
session_start();
// TODO 2: Hapus session
if(isset($_SESSION['cart'])){
    unset($_SESSION['cart']);
}
// TODO 3: Redirect ke halaman show_cart.php
header('Location: show_cart.php'); ?>
```

Dari kode diatas kita akan menghapus session cart sehingga barang yang sudah pernah kita masukan ke dalam cart akan terhapus

BAB IV

PENUTUP

Implementasi operasi CRUD (Create, Read, Update, Delete) dan manajemen sesi (session) dalam halaman customer dan cart sangat penting dalam pengembangan aplikasi web. Dengan implementasi CRUD, kita dapat mengelola data pelanggan dengan cara yang efisien dan fleksibel, yang mencakup menampilkan data pelanggan, mengeditnya, menambahkan pelanggan baru, dan menghapus data yang tidak diperlukan. Ini memberikan pengalaman pengguna yang lebih baik dan mengoptimalkan fungsionalitas basis data.

Penggunaan manajemen sesi sangat relevan dalam keamanan dan pengaturan akses. Implementasi sesi dalam berkas login.php memastikan bahwa hanya pengguna yang terotorisasi yang dapat mengakses halaman view_customer.php. Selain itu, dengan menambahkan fitur logout, sesi pengguna dapat diakhiri dengan aman, yang merupakan langkah penting dalam menjaga keamanan aplikasi web. Jadi, seluruh implementasi CRUD dan manajemen sesi adalah bagian integral dari pembangunan aplikasi web yang efisien, aman, dan user-friendly.

DAFTAR PUSTAKA

Smith, J. (2020). CRUD Operations in Web Development. Web Development Journal. https://www.webdevjournal.com/crud-operations

Johnson, P. (2019). Mastering MySQL: Best Practices for Database Developers. O'Reilly Media.

McLaughlin, M. (2019). MySQL Workbench: Data Modeling & Development. McGraw-Hill Education.

Nixon, R. (2018). Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly Media.

Power, D. (2017). Session Management in Web Applications. Apress.