

Laporan Praktikum

Praktikum Pemrograman Web

Pertemuan 12

PHP CRUD



Nama Mahasiswa: Abdullah Afif
Habiburrohman(24/537611/SV/24441)
Kelas A1

Dosen Pengampu:
Dinar Nugroho Pratomo, S.Kom., M.IM., M.Cs.
Faza Maula Azif, S.Kom., M.Eng.

May 29, 2025

Contents

1	Pendahuluan	3
1.1	Tujuan	3
1.2	Dasar Teori	3
1.2.1	CRUD (Create, Read, Update, Delete)	3
1.2.2	CREATE	4
1.2.3	READ	4
1.2.4	UPDATE	4
1.2.5	DELETE	5
2	Hasil dan Pembahasan	6
2.1	db_config.php	6
2.2	index.php	6
2.3	tambah.php	8
2.4	edit.php	11
2.5	hapus.php	14
2.6	detail.php	14
2.7	style.css	16
2.8	Hasil Praktikum	23
2.9	GitHub	25
3	Kesimpulan	26
3.1	Kesimpulan	26
3.2	Referensi	26

List of Figures

2.1	Tampilan Awal	23
2.2	Detail Transaksi	23
2.3	Edit Transaksi	23
2.4	Hapus Transaksi	24
2.5	Hasil Setelah Hapus Transaksi	24
2.6	Tambah Transaksi	24
2.7	Hasil Setelah Tambah Transaksi	25

Chapter 1

Pendahuluan

1.1 Tujuan

1. Mahasiswa mampu memahami konsep Create, Read, Update, dan Delete (CRUD) dalam pemrograman web.
2. Mahasiswa mampu mengimplementasikan operasi CRUD menggunakan PHP dan MySQL.

1.2 Dasar Teori

1.2.1 CRUD (Create, Read, Update, Delete)

CRUD adalah akronim dari Create, Read, Update, dan Delete, yang merupakan empat operasi dasar yang harus dimiliki oleh aplikasi perangkat lunak untuk mengelola data secara persisten. Operasi ini sering digunakan dalam aplikasi berbasis database, baik itu Relational Database Management Systems (RDBMS) seperti MySQL, PostgreSQL, maupun database NoSQL seperti MongoDB dan DynamoDB.

1. **Create (Membuat):** Operasi ini digunakan untuk menambahkan data baru ke dalam sistem. Contohnya, dalam sistem perpustakaan, fungsi Create digunakan untuk menambahkan buku baru ke dalam katalog dengan informasi seperti judul, penulis, dan ISBN.
2. **Read (Membaca):** Operasi ini digunakan untuk membaca atau mengambil data yang sudah ada di dalam sistem tanpa mengubahnya. Contohnya, pengguna dapat melihat daftar semua buku yang tersedia di katalog atau mencari buku tertentu berdasarkan judul atau ISBN.
3. **Update (Memperbarui):** Operasi ini digunakan untuk memperbarui atau mengedit data yang sudah ada. Contohnya, jika ada kesalahan pada informasi buku, pengguna dapat memperbarui data tersebut dengan informasi yang benar.
4. **Delete (Menghapus):** Operasi ini digunakan untuk menghapus data dari sistem. Contohnya, jika sebuah buku sudah tidak tersedia, pengguna dapat menghapusnya dari katalog.

CRUD juga memiliki hubungan dengan metode HTTP dalam pengembangan aplikasi web:

- **Create:** Menggunakan metode HTTP POST.

- **Read:** Menggunakan metode HTTP GET.
- **Update:** Menggunakan metode HTTP PUT atau PATCH.
- **Delete:** Menggunakan metode HTTP DELETE.

Paradigma CRUD sangat umum digunakan dalam pengembangan aplikasi web karena memberikan kerangka kerja yang mudah diingat untuk membangun model yang lengkap dan fungsional.

1.2.2 CREATE

Operasi CREATE digunakan untuk menambahkan data baru ke dalam sistem. Dalam konteks REST, CREATE biasanya diimplementasikan menggunakan metode HTTP POST. Contohnya, untuk menambahkan item baru ke dalam database, kita dapat menggunakan query SQL INSERT atau metode insert() dalam database NoSQL seperti MongoDB.

1. **HTTP POST:** Digunakan untuk membuat resource baru.
2. **SQL INSERT:** Menambahkan baris baru ke tabel database.
3. **NoSQL insert():** Menambahkan dokumen baru ke koleksi.

1.2.3 READ

Operasi READ digunakan untuk mengambil data dari sistem tanpa mengubahnya. Dalam konteks REST, READ biasanya diimplementasikan menggunakan metode HTTP GET. Contohnya, untuk membaca data dari database, kita dapat menggunakan query SQL SELECT atau metode find() dalam database NoSQL seperti MongoDB.

1. **HTTP GET:** Digunakan untuk mengambil resource.
2. **SQL SELECT:** Mengambil data dari tabel database.
3. **NoSQL find():** Mengambil dokumen dari koleksi.

1.2.4 UPDATE

Operasi UPDATE digunakan untuk memperbarui data yang sudah ada di dalam sistem. Dalam konteks REST, UPDATE biasanya diimplementasikan menggunakan metode HTTP PUT atau PATCH. Contohnya, untuk memperbarui data di database, kita dapat menggunakan query SQL UPDATE atau metode findByIdAndUpdate() dalam database NoSQL seperti MongoDB.

1. **HTTP PUT/PATCH:** Digunakan untuk memperbarui resource.
2. **SQL UPDATE:** Memperbarui data di tabel database.
3. **NoSQL findByIdAndUpdate():** Memperbarui dokumen di koleksi.

1.2.5 DELETE

Operasi DELETE digunakan untuk menghapus data dari sistem. Dalam konteks REST, DELETE biasanya diimplementasikan menggunakan metode HTTP DELETE. Contohnya, untuk menghapus data dari database, kita dapat menggunakan query SQL DELETE atau metode `findByIdAndDelete()` dalam database NoSQL seperti MongoDB.

1. **HTTP DELETE:** Digunakan untuk menghapus resource.
2. **SQL DELETE:** Menghapus data dari tabel database.
3. **NoSQL `findByIdAndDelete()`:** Menghapus dokumen dari koleksi.

Chapter 2

Hasil dan Pembahasan

Pada praktikum kali ini, mahasiswa menggunakan kode PHP yang sudah dibuat dan disediakan pada eLok. Kode pada eLok dimodifikasi sedemikian rupa sehingga dapat digunakan untuk menampilkan data-data menggunakan database MySQL. Pengembangan lain yang dilakukan adalah dengan merubah tampilan dari kode eLok yang cenderung netral menjadi menyesuaikan dengan desain yang telah mahasiswa buat sebelumnya. Kode akan disertakan dengan penjelasan secara singkat ke beberapa bagian penting dari kode tersebut.

2.1 db_config.php

```
1 <?php
2 $host = "localhost";
3 $username = "root";
4 $password = "";
5 $database = "lapkeu_dummy";
6
7 $conn = mysqli_connect($host, $username, $password, $database);
8
9 if (!$conn) {
10     die("Koneksi_gagal: " . mysqli_connect_error());
11 }
12 ?>
```

1. Bagian kode ini berfungsi untuk menghubungkan PHP dengan database MySQL.
2. Variabel `$host`, `$username`, `$password`, dan `$database` digunakan untuk menyimpan informasi koneksi ke database.
3. Fungsi `mysqli_connect` digunakan untuk membuat koneksi ke database.
4. Jika koneksi gagal, akan menampilkan pesan error menggunakan `die()`.
5. Jika koneksi berhasil, variabel `$conn` akan digunakan untuk melakukan query ke database.

2.2 index.php

```
1 <!DOCTYPE html>
2 <html lang="id">
```

```

3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale
    ↳ =1.0">
6   <title>Transaksi</title>
7   <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10  <div class="containeridx">
11    <h1>Data Transaksi</h1>
12
13    <div class="header-action">
14      <a href="tambah.php" class="btn">Tambah Transaksi</a>
15    </div>
16
17    <table>
18      <thead>
19        <tr>
20          <th>No</th>
21          <th>ID Transaksi</th>
22          <th>ID Bisnis</th>
23          <th>Tanggal</th>
24          <th>Deskripsi</th>
25          <th>Aksi</th>
26        </tr>
27      </thead>
28      <tbody>

```

1. Bagian kode ini adalah dasar dari website.
2. Terdapat head untuk menyimpan metadata dari website
3. Disusul dengan body yang berisi judul dan tabel yang akan digunakan untuk menampilkan data transaksi.

```

1 <?php
2 include_once("db_config.php"); //
3   ↳ Query untuk mengambil data transaksi
  $result = mysqli_query($conn, "SELECT * FROM
    ↳ lapkeu_transaction ORDER BY transaction_id DESC"
    ↳ );
4
5 // Cek apakah ada data
6 if (mysqli_num_rows($result) > 0) {
7   $no = 1;
8   // Looping untuk menampilkan data
9   while($row = mysqli_fetch_assoc($result)) {
10     echo "<tr>";
11     echo "<td>". $no++. "</td>";
12     echo "<td>". $row['transaction_id']. "</td>";
13     echo "<td>". $row['business_id']. "</td>";
14     echo "<td>". $row['date']. "</td>";
15     echo "<td>". $row['description']. "</td>";
    ↳ echo "<td>";

```



```

16         echo "<a_href='detail.php?id=".$row['
           ↳ transaction_id']."'>Detail </
           ↳ a>";
17         echo "<a_href='edit.php?id=".$row['
           ↳ transaction_id']."'>Edit </a>";
18         echo "<a_href='hapus.php?id=".$row['
           ↳ transaction_id']."'>Hapus </a>";
19         echo "</td>";
20         echo "</tr>";
21     } else {
22         echo "<tr><td_colspan='6'>Tidak ada data</td></tr>";
23     }
24
25     // Tutup koneksi
26     mysqli_close($conn);
27     ?>
28     </tbody>
29     </table>
30 </div>
31 </body>
32 </html>

```

1. Bagian kode ini berfungsi untuk mengambil data dari database MySQL yang telah dibuat sebelumnya.
2. Menggunakan fungsi `mysqli_query` untuk mengambil data dari tabel `lapkeu_transaction`.
3. Jika data ditemukan, maka akan ditampilkan dalam bentuk tabel dengan kolom No, ID Transaksi, ID Bisnis, Tanggal, Deskripsi, dan Aksi.
4. Setiap baris data akan memiliki tombol Detail, Edit, dan Hapus yang mengarah ke halaman masing-masing.
5. Jika tidak ada data, akan ditampilkan pesan "Tidak ada data".
6. Terakhir, koneksi ke database ditutup dengan `mysqli_close`.

2.3 tambah.php

```

1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale
           ↳ =1.0">
6     <title>Tambah Transaksi</title>
7     <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10     <div class="container">
11         <h1>Tambah Transaksi</h1>

```

1. Bagian kode ini adalah dasar dari halaman untuk menambahkan transaksi baru.
2. Terdapat head untuk menyimpan metadata dari website
3. Disusul dengan body yang berisi judul dan form untuk menambahkan data transaksi.

```

1      <?php
2      // Include file koneksi database
3      include_once("db_config.php");
4      // Cek apakah form telah di-submit
5      if(isset($_POST['submit'])) {
6          $business_id = $_POST['business_id'];
7          $date = $_POST['date'];
8          $description = $_POST['description'];
9
10         // Validasi form
11         $errors = array();
12
13         if(empty($business_id)) {
14             $errors[] = "ID_Bisnis_tidak_boleh_kosong";
15         }
16
17         if(empty($date)) {
18             $errors[] = "Tanggal_tidak_boleh_kosong";
19         }
20
21         if(empty($description)) {
22             $errors[] = "Deskripsi_tidak_boleh_kosong";
23         }
24         // Jika tidak ada error, simpan data
25         if(empty($errors)) {
26             $result = mysqli_query($conn, "INSERT INTO
27             ↳ lapkeu_transaction(business_id,date,
28             ↳ description)
29             ↳ VALUES('$business_id',
30             ↳ '$date','$description')");
31             if($result) {
32                 echo "<div class='alert alert-success'>";
33                 echo "Data transaksi berhasil ditambahkan.<a href
34                 ↳ ='index.php' style='color:var(--dark-purple
35                 ↳ );font-weight:600;'>Lihat Data</a>";
36                 echo "</div>";
37             } else {
38                 echo "<div class='alert alert-danger'>";
39                 echo "Error: " . mysqli_error($conn);
40                 echo "</div>";
41             }
42         } else {
43             echo "<div class='alert alert-danger'>";
44             echo "<ul>";
45             foreach($errors as $error) {
46                 echo "<li>$error</li>";
47             }
48             echo "</ul>";
49             echo "</div>";
50         }
51     }

```

```

46     }
47     ?>

```

1. Bagian kode ini berfungsi untuk menangani pengiriman form.
2. Jika form disubmit, data akan diambil dari inputan dan disimpan dalam variabel.
3. Validasi dilakukan untuk memastikan bahwa semua field tidak kosong.
4. Jika validasi berhasil, data akan disimpan ke dalam tabel `lapkeu_transaction` menggunakan query `INSERT`.
5. Jika berhasil, akan ditampilkan pesan sukses dengan link ke halaman utama. Jika gagal, akan ditampilkan pesan error.
6. Jika ada error pada validasi, akan ditampilkan pesan error yang sesuai.
7. Jika form belum disubmit, halaman akan menampilkan form kosong untuk diisi.

```

1      <form action="tambah.php" method="post">
2          <div class="form-group">
3              <label for="business_id">ID Bisnis</label>
4              <input type="number" name="business_id" id="
5                  ↪ business_id" required>
6          </div>
7
8          <div class="form-group">
9              <label for="date">Tanggal</label>
10             <input type="date" name="date" id="date" required>
11         </div>
12
13         <div class="form-group">
14             <label for="description">Deskripsi</label>
15             <textarea name="description" id="description" required
16                 ↪ ></textarea>
17         </div>
18
19         <div style="margin-top: 20px;">
20             <input type="submit" name="submit" value="Simpan"
21                 ↪ class="btn">
22             <a href="index.php" class="btn btn-secondary">Batal</a
23                 ↪ >
24         </div>
25     </form>
26 </div>
27 </body>
28 </html>

```

1. Bagian kode ini adalah form untuk menambahkan transaksi baru.
2. Terdapat input untuk ID Bisnis, Tanggal, dan Deskripsi.
3. Setiap input memiliki label yang sesuai dan atribut `required` untuk memastikan field tidak kosong.

4. Terdapat tombol **Simpan** untuk mengirimkan data ke server dan tombol **Batal** untuk kembali ke halaman utama.
5. Form ini akan mengirimkan data ke halaman `tambah.php` dengan metode POST.

2.4 edit.php

```

1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale
      ↪ =1.0">
6     <title>Edit Transaksi</title>
7     <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10     <div class="container">
11         <h1>Edit Transaksi</h1>

```

1. Bagian kode ini adalah dasar dari halaman untuk mengedit transaksi yang sudah ada.
2. Terdapat head untuk menyimpan metadata dari website
3. Disusul dengan body yang berisi judul dan form untuk mengedit data transaksi.

```

1 <?php
2 // Include file koneksi database
3 include_once("db_config.php");
4
5 // Cek apakah ada ID yang dikirimkan
6 if(!isset($_GET['id'])) {
7     header("Location: index.php");
8     exit();
9 }
10
11 $id = $_GET['id'];
12 // Cek apakah form telah di-submit
13 if(isset($_POST['update'])) {
14     $business_id = $_POST['business_id'];
15     $date = $_POST['date'];
16     $description = $_POST['description'];
17
18     // Validasi form
19     $errors = array();
20
21     if(empty($business_id)) {
22         $errors[] = "ID_Bisnis tidak boleh kosong";
23     }
24
25     if(empty($date)) {
26         $errors[] = "Tanggal tidak boleh kosong";
27     }
28

```

```

29         if(empty($description)) {
30             $errors[] = "Deskripsi_tidak_boleh_kosong";
31         }
32         // Jika tidak ada error, update data
33         if(empty($errors)) {
34             $result = mysqli_query($conn, "UPDATE_
35                 ↳ lapkeu_transaction SET
36                 ↳ business_id='
37                 ↳ $business_id',
38                 ↳ date='$date',
39                 ↳ description='
40                 ↳ $description'
41                 ↳ WHERE transaction_id=
42                 ↳ $id");
43             if($result) {
44                 echo "<div class='alert alert-success'>";
45                 echo "Data_transaksi_berhasil_diperbarui.<a href
46                 ↳ ='index.php' style='color:var(--dark-purple
47                 ↳ );font-weight:600;'>Lihat Data</a>";
48                 echo "</div>";
49             } else {
50                 echo "<div class='alert alert-danger'>";
51                 echo "Error: " . mysqli_error($conn);
52                 echo "</div>";
53             }
54         } else {
55             echo "<div class='alert alert-danger'>";
56             echo "<ul>";
57             foreach($errors as $error) {
58                 echo "<li>$error</li>";
59             }
60             echo "</ul>";
61             echo "</div>";
62         }
63     }
64     // Ambil data transaksi berdasarkan ID
65     $result = mysqli_query($conn, "SELECT * FROM_
66         ↳ lapkeu_transaction WHERE transaction_id=$id");
67
68     // Jika data tidak ditemukan, kembali ke halaman utama
69     if(mysqli_num_rows($result) == 0) {
70         header("Location: index.php");
71         exit();
72     }
73
74     // Ambil data untuk ditampilkan di form
75     $row = mysqli_fetch_assoc($result);
76     $business_id = $row['business_id'];
77     $date = $row['date'];
78     $description = $row['description'];
79     ?>

```

1. Bagian kode ini berfungsi untuk menangani pengambilan data transaksi yang akan diedit.
2. Jika ID transaksi tidak ditemukan, pengguna akan diarahkan kembali ke halaman utama.

3. Jika form disubmit, data akan diambil dari inputan dan disimpan dalam variabel.
4. Validasi dilakukan untuk memastikan bahwa semua field tidak kosong.
5. Jika validasi berhasil, data akan diperbarui ke dalam tabel `lapkeu_transaction` menggunakan query `UPDATE`.
6. Jika berhasil, akan ditampilkan pesan sukses dengan link ke halaman utama. Jika gagal, akan ditampilkan pesan error.
7. Jika ada error pada validasi, akan ditampilkan pesan error yang sesuai.
8. Data transaksi yang akan diedit diambil dari database berdasarkan ID yang diberikan.
9. Jika data ditemukan, akan ditampilkan dalam form untuk diedit.
10. Jika data tidak ditemukan, pengguna akan diarahkan kembali ke halaman utama.

```

1      <form action="edit.php?id=?php_echo_$id;" method="post">
2          <div class="form-group">
3              <label for="business_id">ID Bisnis</label>
4              <input type="number" name="business_id" id="
                    ↪ business_id" value="<?php_echo_$business_id;"
                    ↪ required>
5          </div>
6
7          <div class="form-group">
8              <label for="date">Tanggal</label>
9              <input type="date" name="date" id="date" value="<?php_
                    ↪ echo_$date;" required>
10         </div>
11
12         <div class="form-group">
13             <label for="description">Deskripsi</label>
14             <textarea name="description" id="description" required
                    ↪ ><?php echo $description; ?></textarea>
15         </div>
16
17         <div style="margin-top: 20px;">
18             <input type="submit" name="update" value="Update"
                    ↪ class="btn">
19             <a href="index.php" class="btn btn-secondary">Batal</a>
                    ↪ >
20         </div>
21     </form>
22 </div>
23 </body>
24 </html>

```

1. Bagian kode ini adalah form untuk mengedit transaksi yang sudah ada.
2. Terdapat input untuk ID Bisnis, Tanggal, dan Deskripsi yang sudah diisi dengan data yang ada.
3. Setiap input memiliki label yang sesuai dan atribut `required` untuk memastikan field tidak kosong.

4. Terdapat tombol **Update** untuk mengirimkan data ke server dan tombol **Batal** untuk kembali ke halaman utama.
5. Form ini akan mengirimkan data ke halaman `edit.php` dengan metode **POST**.
6. Data yang sudah ada akan ditampilkan pada inputan sehingga pengguna dapat mengeditnya.

2.5 hapus.php

```
1 <?php
2 // Include file koneksi database
3 include_once("db_config.php");
4
5 // Cek apakah ada ID yang dikirimkan
6 if(isset($_GET['id'])) {
7     $id = $_GET['id'];
8
9     // Query untuk menghapus data
10    $result = mysqli_query($conn, "DELETE FROM lapkeu_transaction
        ↳ WHERE transaction_id=$id");
11
12    // Redirect ke halaman utama
13    header("Location: index.php");
14 } else {
15     // Jika tidak ada ID yang dikirimkan, kembali ke halaman utama
16     header("Location: index.php");
17 }
18 ?>
```

1. Bagian kode ini berfungsi untuk menghapus data transaksi berdasarkan ID yang diberikan.
2. Jika ID transaksi ditemukan, maka akan dilakukan query **DELETE** untuk menghapus data dari tabel `lapkeu_transaction`.
3. Setelah berhasil menghapus, pengguna akan diarahkan kembali ke halaman utama.
4. Jika tidak ada ID yang diberikan, pengguna juga akan diarahkan kembali ke halaman utama.

2.6 detail.php

```
1 <?php
2 // Include file koneksi database
3 include_once("db_config.php");
4
5 // Cek apakah ada ID yang dikirimkan
6 if(!isset($_GET['id'])) {
7     header("Location: index.php");
8     exit();
9 }
10
11 $id = $_GET['id'];
```

```

12
13 // Ambil data transaksi berdasarkan ID
14 $result = mysqli_query($conn, "SELECT * FROM lapkeu_transaction WHERE
    ↳ transaction_id=$id");
15
16 // Jika data tidak ditemukan, kembali ke halaman utama
17 if(mysqli_num_rows($result) == 0) {
18     header("Location: index.php");
19     exit();
20 }
21
22 // Ambil data untuk ditampilkan
23 $data = mysqli_fetch_assoc($result);
24 $transaction_id = $data['transaction_id'];
25 $business_id = $data['business_id'];
26 $date = $data['date'];
27 $description = $data['description'];
28 ?>

```

1. Bagian kode ini berfungsi untuk menampilkan detail dari transaksi yang dipilih.
2. Jika ID transaksi tidak ditemukan, pengguna akan diarahkan kembali ke halaman utama.
3. Data transaksi diambil dari database berdasarkan ID yang diberikan.
4. Jika data ditemukan, akan ditampilkan detail transaksi seperti ID Transaksi, ID Bisnis, Tanggal, dan Deskripsi.
5. Jika data tidak ditemukan, pengguna akan diarahkan kembali ke halaman utama.

```

1 <!DOCTYPE html>
2 <html lang="id">
3 <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale
    ↳ =1.0">
6     <title>Detail Transaksi</title>
7     <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10     <div class="container">
11         <h1>Detail Transaksi</h1>
12
13         <div class="detail-item">
14             <span class="detail-label">ID Transaksi:</span>
15             <span><?php echo $transaction_id; ?></span>
16         </div>
17
18         <div class="detail-item">
19             <span class="detail-label">ID Bisnis:</span>
20             <span><?php echo $business_id; ?></span>
21         </div>
22
23         <div class="detail-item">

```



```

24         <span class="detail-label">Tanggal:</span>
25         <span><?php echo $date; ?></span>
26     </div>
27
28     <div class="detail-item">
29         <span class="detail-label">Deskripsi:</span>
30         <span><?php echo $description; ?></span>
31     </div>
32
33     <div class="actions">
34         <a href="index.php" class="btn">Kembali</a>
35         <a href="edit.php?id=<?php echo $transaction_id; ?>" class
36             ↪ ="btn btn-edit">Edit</a>
37         <a href="hapus.php?id=<?php echo $transaction_id; ?>"
38             ↪ class="btn btn-delete" onclick="return confirm('
39             ↪ Yakin ingin menghapus data?')">Hapus</a>
40     </div>
</body>
</html>

```

1. Bagian kode ini adalah tampilan detail dari transaksi yang dipilih.
2. Terdapat head untuk menyimpan metadata dari website.
3. Disusul dengan body yang berisi judul dan detail transaksi.
4. Setiap detail transaksi ditampilkan dalam elemen `div` dengan kelas `detail-item`.
5. Setiap detail memiliki label yang ditampilkan dengan kelas `detail-label` dan nilai yang diambil dari variabel PHP.
6. Terdapat tombol `Kembali` untuk kembali ke halaman utama, tombol `Edit` untuk mengedit transaksi, dan tombol `Hapus` untuk menghapus transaksi.
7. Tombol `Hapus` akan meminta konfirmasi sebelum menghapus data.
8. Form ini akan mengirimkan data ke halaman `detail.php` dengan metode `GET` untuk menampilkan detail transaksi berdasarkan ID yang diberikan.

2.7 style.css

```

1  /* Color scheme */
2  :root {
3      --pale-white: #F4F9F9;
4      --light-blue: #CCF2F4;
5      --medium-blue: #A4EBF3;
6      --light-purple: #A9B5DF;
7      --mid-purple: #7886C7;
8      --dark-purple: #2D336B;
9  }

```

1. Bagian kode ini mendefinisikan skema warna yang akan digunakan di seluruh halaman.

2. Variabel CSS didefinisikan dengan nama yang mudah diingat untuk digunakan dalam styling elemen HTML.
3. Warna-warna ini akan digunakan untuk memberikan konsistensi visual pada tampilan website.
4. Penggunaan skema warna ini sama dengan yang terdapat pada desain pada Figma.

```

1  /* Global styles */
2  body {
3    font-family: "Segoe UI", Tahoma, Geneva, Verdana, sans-serif;
4    color: var(--dark-purple);
5    background-color: var(--pale-white);
6    position: relative;
7    margin: 0;
8    padding: 0;
9  }
10
11  body::before {
12    content: "";
13    position: fixed;
14    top: 0;
15    left: 0;
16    right: 0;
17    bottom: 0;
18    background-image: radial-gradient(var(--light-purple) 1px,
19      ↪ transparent 1px);
20    background-size: 30px 30px;
21    opacity: 0.5;
22    pointer-events: none;
23    z-index: -1;
24  }

```

1. Bagian kode ini mengatur gaya global untuk elemen `body`.
2. Font yang digunakan adalah "Segoe UI", yang merupakan font sans-serif yang modern dan mudah dibaca.
3. Warna teks diatur menggunakan variabel `--dark-purple` dan latar belakang menggunakan `--pale-white`.
4. Efek latar belakang tambahan ditambahkan dengan menggunakan pseudo-elemen `::before` untuk memberikan tampilan yang lebih menarik.
5. Latar belakang menggunakan gradien radial dengan warna `--light-purple` yang memberikan efek visual yang halus.

```

1  .container {
2    max-width: 600px;
3    margin: 20px auto;
4    background-color: white;
5    padding: 25px;
6    border-radius: 8px;
7    box-shadow: 0 4px 15px rgba(45, 51, 107, 0.1);

```

```

8  }
9
10 .containeridx {
11     max-width: 1000px;
12     margin: 20px auto;
13     background-color: white;
14     padding: 25px;
15     border-radius: 8px;
16     box-shadow: 0 4px 15px rgba(45, 51, 107, 0.1);
17 }

```

1. Bagian kode ini mengatur gaya untuk elemen dengan kelas `container` dan `containeridx`.
2. Keduanya memiliki lebar maksimum yang ditentukan, margin otomatis untuk pusatkan konten, dan latar belakang putih.
3. Padding digunakan untuk memberikan ruang di dalam kontainer, dan border-radius untuk membuat sudut yang halus.
4. Efek bayangan ditambahkan untuk memberikan kedalaman visual pada kontainer.
5. Kelas `container` digunakan untuk halaman tambah dan edit, sedangkan `containeridx` digunakan untuk halaman index.
6. Container ini digunakan juga pada website yang dibuat mahasiswa nantinya. Work in progress.

```

1  h1 {
2      text-align: center;
3      margin-bottom: 20px;
4      color: var(--dark-purple);
5      font-weight: 600;
6  }
7
8  /* Table styles */
9  table {
10     width: 100%;
11     border-collapse: collapse;
12     margin-top: 20px;
13     box-shadow: 0 2px 10px rgba(45, 51, 107, 0.1);
14     border-radius: 8px;
15     overflow: hidden;
16 }
17
18 th, td {
19     padding: 12px 15px;
20     text-align: left;
21     border-bottom: 1px solid #ddd;
22 }
23
24 th {
25     background-color: var(--light-purple);
26     color: var(--dark-purple);
27     font-weight: bold;
28 }

```

```

29
30 tr:hover {
31     background-color: rgba(204, 242, 244, 0.3);
32 }
33
34 tr:nth-child(even) {
35     background-color: rgba(169, 181, 223, 0.1);
36 }

```

1. Bagian kode ini mengatur gaya untuk elemen **h1** dan tabel.
2. Judul **h1** diatur agar berada di tengah dengan margin bawah untuk memberikan jarak.
3. Tabel memiliki lebar 100% dari kontainer, dengan border-collapse untuk menghilangkan jarak antar sel.
4. Setiap sel tabel memiliki padding, dan baris tabel akan berubah warna saat di-hover.
5. Baris genap diberi latar belakang yang berbeda untuk meningkatkan keterbacaan.
6. Gaya ini membuat tabel terlihat lebih modern dan mudah dibaca.
7. Penggunaan warna latar belakang yang berbeda pada baris genap dan saat di-hover memberikan kontras yang baik, sehingga pengguna dapat dengan mudah membedakan antar baris.

```

1  /* Button styles */
2  .btn {
3      display: inline-block;
4      padding: 8px 12px;
5      background-color: var(--dark-purple);
6      color: white;
7      text-decoration: none;
8      border-radius: 4px;
9      transition: background-color 0.3s ease, transform 0.3s ease;
10     font-weight: 500;
11     border: none;
12     margin: 2px;
13 }
14
15 .btn:hover {
16     background-color: var(--mid-purple);
17     color: white;
18     transform: translateY(-2px);
19     box-shadow: 0 2px 10px rgba(45, 51, 107, 0.15);
20 }
21
22 .btn-edit {
23     background-color: var(--mid-purple);
24 }
25
26 .btn-edit:hover {
27     background-color: #6777b8;
28 }
29

```

```

30 .btn-delete {
31     background-color: #f44336;
32 }
33
34 .btn-delete:hover {
35     background-color: #da190b;
36 }
37
38 .header-action {
39     margin-bottom: 20px;
40     text-align: right;
41 }
42
43 .btn-secondary {
44     background-color: var(--light-purple);
45     color: var(--dark-purple);
46 }
47
48 .btn-secondary:hover {
49     background-color: var(--medium-blue);
50     color: var(--dark-purple);
51 }

```

1. Bagian kode ini mengatur gaya untuk tombol-tombol yang digunakan di halaman.
2. Tombol memiliki latar belakang ungu gelap, teks putih, dan border-radius untuk sudut yang halus.
3. Efek hover ditambahkan untuk memberikan umpan balik visual saat pengguna mengarahkan kursor ke tombol.
4. Tombol **Edit** dan **Delete** memiliki warna khusus untuk membedakan fungsinya.
5. Tombol **Batal** menggunakan kelas **btn-secondary** dengan warna latar belakang yang lebih terang.
6. Gaya ini membuat tombol terlihat modern dan responsif terhadap interaksi pengguna.

```

1  /* Form styles */
2  .form-group {
3      margin-bottom: 15px;
4  }
5
6  label {
7      display: block;
8      margin-bottom: 5px;
9      font-weight: bold;
10     color: var(--dark-purple);
11 }
12
13 input[type="text"],
14 input[type="email"],
15 input[type="number"],
16 input[type="date"],
17 textarea {

```

```
18   width: 100%;
19   padding: 8px;
20   border: 1px solid var(--light-purple);
21   border-radius: 4px;
22   box-sizing: border-box;
23   background-color: var(--pale-white);
24   color: var(--dark-purple);
25   transition: border-color 0.3s ease;
26 }
27
28 input:focus,
29 textarea:focus {
30   outline: none;
31   border-color: var(--mid-purple);
32   box-shadow: 0 0 0 2px rgba(120, 134, 199, 0.2);
33 }
34
35 textarea {
36   height: 100px;
37   resize: vertical;
38 }
39
40 /* Detail page */
41 .detail-item {
42   margin-bottom: 15px;
43   border-bottom: 1px solid var(--light-purple);
44   padding-bottom: 10px;
45 }
46
47 .detail-item:last-child {
48   border-bottom: none;
49 }
50
51 .detail-label {
52   font-weight: bold;
53   display: inline-block;
54   width: 150px;
55   color: var(--mid-purple);
56 }
57
58 .actions {
59   margin-top: 20px;
60   text-align: center;
61 }
62
63 /* Alert messages */
64 .alert {
65   padding: 10px;
66   border-radius: 5px;
67   margin-bottom: 15px;
68 }
69
70 .alert-success {
71   background-color: #d4edda;
72   color: #155724;
73   border: 1px solid #c3e6cb;
```

```

74 }
75
76 .alert-danger {
77   background-color: #f8d7da;
78   color: #721c24;
79   border: 1px solid #f5c6cb;
80 }

```

1. Bagian kode ini mengatur gaya untuk form input dan halaman detail.
2. Setiap grup form memiliki margin bawah untuk memberikan jarak antar elemen.
3. Label ditampilkan sebagai blok dengan margin bawah, dan input memiliki padding, border, dan border-radius untuk tampilan yang bersih.
4. Efek fokus ditambahkan pada input untuk memberikan umpan balik visual saat pengguna berinteraksi dengan form.
5. Halaman detail menampilkan informasi transaksi dengan label yang tebal dan margin bawah untuk setiap item detail.
6. Tombol aksi di halaman detail diatur agar berada di tengah dengan margin atas.
7. Pesan alert ditambahkan untuk memberikan umpan balik kepada pengguna tentang status operasi (sukses atau gagal).

```

1  /* Responsive styles */
2  @media (max-width: 768px) {
3    .container, .containeridx {
4      padding: 15px;
5    }
6
7    th, td {
8      padding: 8px 10px;
9    }
10
11    .btn {
12      padding: 6px 10px;
13      font-size: 0.9rem;
14    }
15  }

```

1. Bagian kode ini mengatur gaya responsif untuk tampilan di perangkat dengan lebar maksimum 768px.
2. Padding pada kontainer dan tabel dikurangi untuk mengoptimalkan ruang pada layar kecil.
3. Padding pada tombol juga disesuaikan agar tetap nyaman digunakan pada perangkat mobile.
4. Gaya responsif ini memastikan bahwa website tetap terlihat baik dan mudah digunakan di berbagai ukuran layar.

2.8 Hasil Praktikum

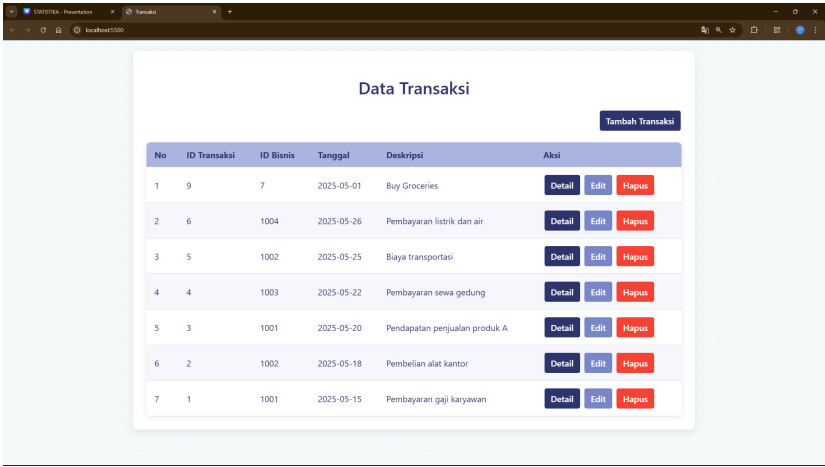


Figure 2.1: Tampilan Awal

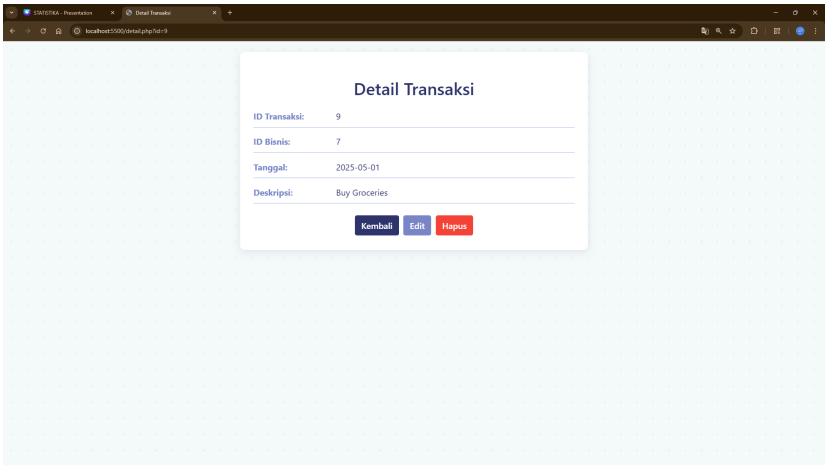


Figure 2.2: Detail Transaksi

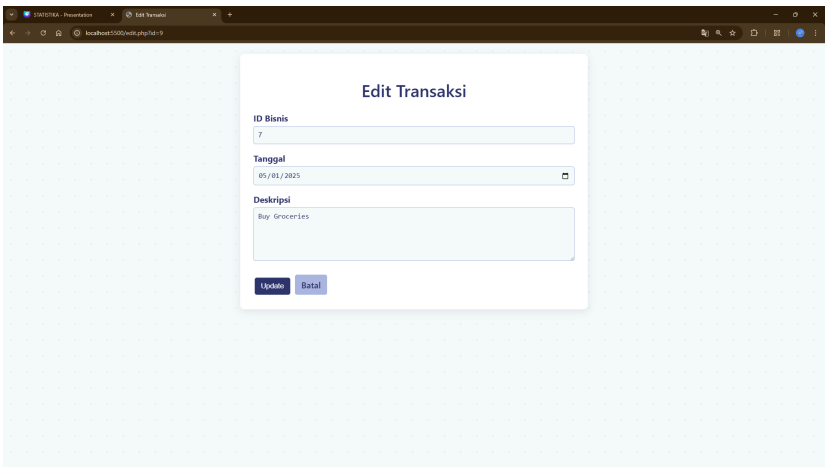


Figure 2.3: Edit Transaksi

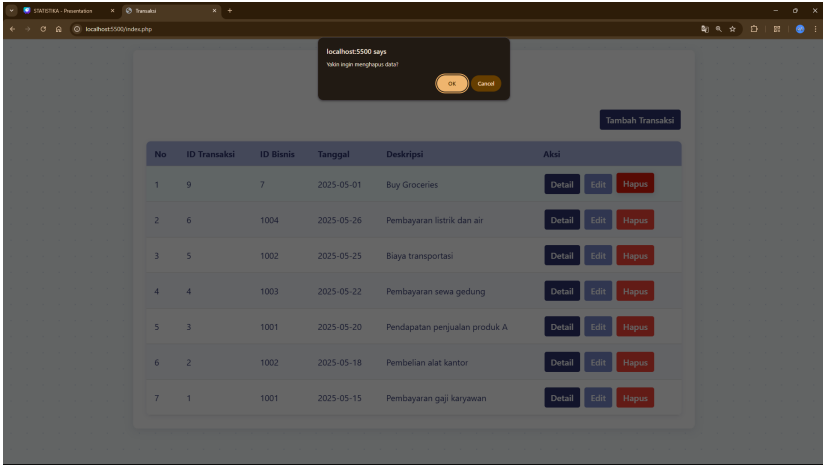


Figure 2.4: Hapus Transaksi

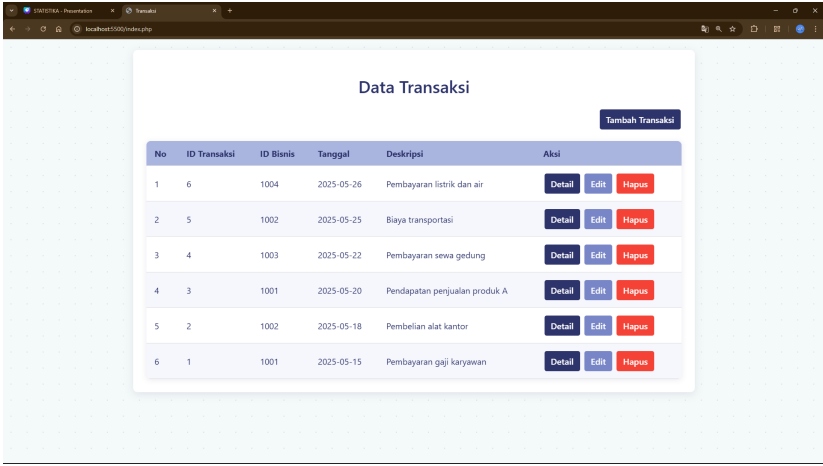


Figure 2.5: Hasil Setelah Hapus Transaksi

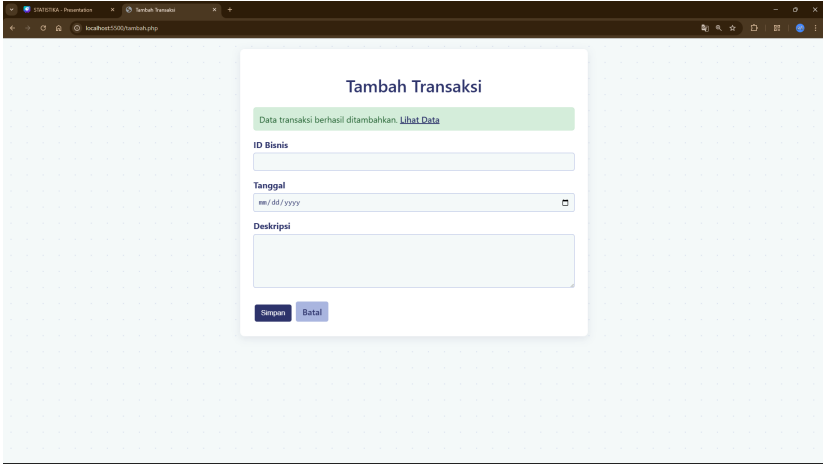
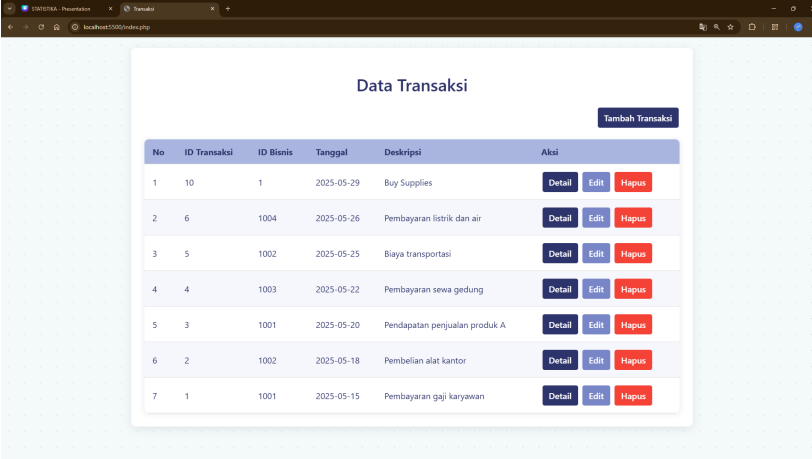


Figure 2.6: Tambah Transaksi



The screenshot shows a web browser displaying a page titled "Data Transaksi". At the top right of the page is a button labeled "Tambah Transaksi". Below this is a table with the following columns: "No", "ID Transaksi", "ID Bisnis", "Tanggal", "Deskripsi", and "Aksi". The table contains 7 rows of data. Each row in the "Aksi" column has three buttons: "Detail" (dark blue), "Edit" (light blue), and "Hapus" (red).

No	ID Transaksi	ID Bisnis	Tanggal	Deskripsi	Aksi
1	10	1	2025-05-29	Buy Supplies	Detail Edit Hapus
2	6	1004	2025-05-26	Pembayaran listrik dan air	Detail Edit Hapus
3	5	1002	2025-05-25	Biaya transportasi	Detail Edit Hapus
4	4	1003	2025-05-22	Pembayaran sewa gedung	Detail Edit Hapus
5	3	1001	2025-05-20	Pendapatan penjualan produk A	Detail Edit Hapus
6	2	1002	2025-05-18	Pembelian alat kantor	Detail Edit Hapus
7	1	1001	2025-05-15	Pembayaran gaji karyawan	Detail Edit Hapus

Figure 2.7: Hasil Setelah Tambah Transaksi

2.9 GitHub

<https://github.com/bukunya/ppw-crud>

Chapter 3

Kesimpulan

3.1 Kesimpulan

Pada praktikum ini, mahasiswa berhasil memahami dan mengimplementasikan konsep CRUD (Create, Read, Update, Delete) dalam pengembangan aplikasi web menggunakan PHP dan MySQL. Berikut adalah poin-poin utama yang dicapai:

1. Mahasiswa mampu membuat aplikasi web sederhana yang dapat mengelola data transaksi dengan fitur CRUD.
2. Operasi CREATE berhasil diimplementasikan untuk menambahkan data baru ke dalam database.
3. Operasi READ berhasil diimplementasikan untuk menampilkan data transaksi dalam bentuk tabel yang rapi dan informatif.
4. Operasi UPDATE berhasil diimplementasikan untuk memperbarui data transaksi yang sudah ada.
5. Operasi DELETE berhasil diimplementasikan untuk menghapus data transaksi dari database.
6. Mahasiswa juga berhasil menerapkan desain yang konsisten dan responsif menggunakan CSS, sehingga aplikasi terlihat modern dan mudah digunakan.
7. Penggunaan database dummy memastikan bahwa aplikasi dapat diuji dengan data yang realistis.
8. Dokumentasi dalam bentuk LaTeX memberikan penjelasan yang jelas tentang konsep dan implementasi yang dilakukan.

3.2 Referensi

1. Codecademy. (2025). What is CRUD? Explained. Codecademy. <https://www.codecademy.com/article/what-is-crud>
2. What Is CRUD? Create, Read, Update, and Delete — CrowdStrike. (2025). CrowdStrike.com. <https://www.crowdstrike.com/en-us/cybersecurity-101/observability/crud/>

3. Chris, K. (2022, June 15). CRUD Operations – What is CRUD? FreeCodeCamp.org.
<https://www.freecodecamp.org/news/crud-operations-explained/>
4. PPW1 — eLOK. (2025). Ugm.ac.id. <https://elok.ugm.ac.id/pluginfile.php/>