

09



# Hypertext Preprocessor (PHP)

## Pemrograman Web

Danny Sebastian, S.Kom., M.M., M.T.

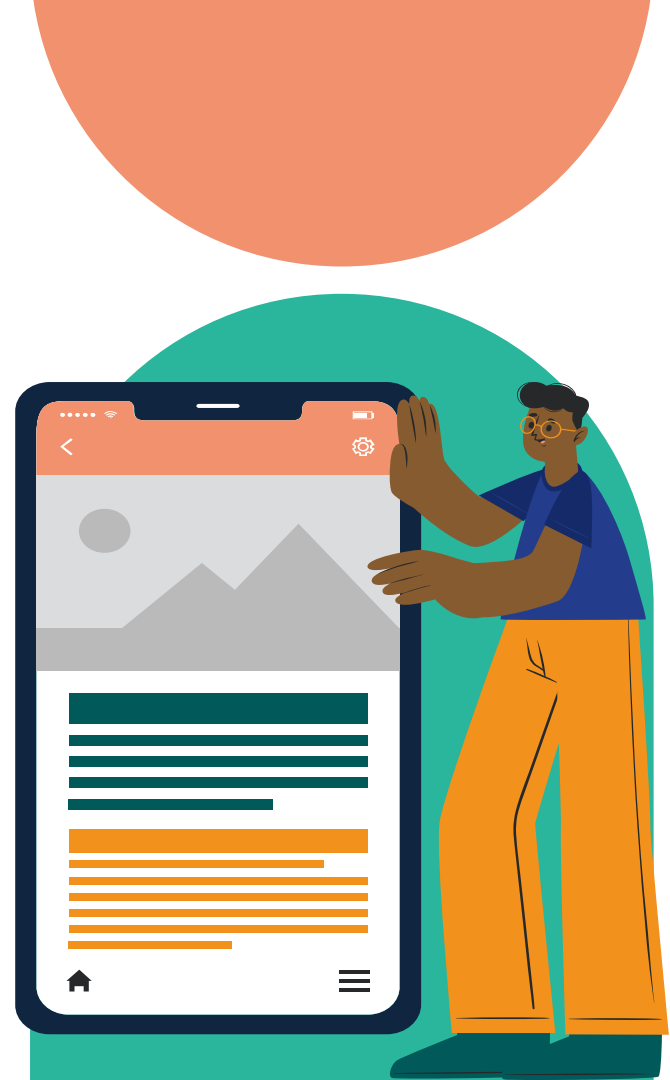
Maria Nila Anggia Rini, S.T., M.TI.

Agata Filiana, S.Kom., M.Sc.

**INFORMATIKA UKDW GENAP 2022/2023**

# TOPIK HARI INI

- ★ **SQL (Refresh)**
- ★ **SELECT Data**
- ★ **INSERT Data**
- ★ **UPDATE Data**
- ★ **DELETE Data**



# AKTIVITAS KELAS

Silahkan membentuk kelompok 2-3 orang per kelompok dan ikuti materi hari ini secara bersamaan. Di akhir, hasilnya akan dikumpul sebagai aktivitas kelas.



# PHP DATABASE














Hal mendasar untuk mengelola konten pada sebuah halaman website adalah SELECT, INSERT, UPDATE, dan DELETE. Kali ini kita akan belajar mengelola data pengguna. Topik yang digunakan untuk penjelasan bagian ini adalah entitas Mahasiswa.

Pada pengembangan website menggunakan PHP dan MariaDB/MySQL pada umumnya akan menggunakan XAMPP bundling. Di dalam XAMPP bundling terdapat phpMyAdmin sebagai website untuk mengelola data pada database. Untuk mengakses phpMyAdmin dapat melalui URL <http://localhost/phpmyadmin>

# KEMBALI KE SQL

Sambil mengingat kembali mata kuliah Sistem Basis Data. Siapkan XAMPP Anda dan lakukan langkah-langkah berikut ini:

1. Buat sebuah database baru dengan nama **universitas**
2. Pada database tersebut, buat sebuah tabel baru dengan nama **mahasiswa** dengan struktur seperti di bawah ini:

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1 id 	int(11)			No	None		AUTO_INCREMENT	 Change  Drop  More
<input type="checkbox"/>	2 nim	char(8)	utf8mb4_general_ci		Yes	NULL			 Change  Drop  More
<input type="checkbox"/>	3 nama	varchar(50)	utf8mb4_general_ci		Yes	NULL			 Change  Drop  More
<input type="checkbox"/>	4 jenis_kelamin	char(1)	utf8mb4_general_ci		Yes	NULL			 Change  Drop  More

# KEMBALI KE SQL

3. Pada tabel mahasiswa, lakukan insert data berikut ini:

NIM	Nama	Jenis Kelamin
71230554	David	L
71230556	Ursula	P
71230557	Lisa	P
71230558	Damai	P
71230562	Peter	L
71230565	Rena	P

# KEMBALI KE SQL

4. Pada tabel mahasiswa, terdapat kesalahan nama pada NIM 71230558, lakukan update nama dari “Damai” menjadi “Dami”.
5. Ternyata ada mahasiswa yang mengundurkan diri, lakukan penghapusan untuk NIM 71230557.
6. Buatlah SQL query untuk menghitung jumlah mahasiswa dengan jenis kelamin ‘P’.





**PHP**



# Syntax Dasar PHP

- Script PHP disimpan dalam file ber-ekstensi .php
  - Bisa 100% script PHP, bisa juga di-"campur" dengan HTML
  - Script PHP didefinisikan dalam tag pembuka: **<?php** dan penutup: **?>**
  - Tiap pernyataan diakhiri dengan **;** (titik koma)
- Script PHP jika disimpan dalam file ber-ekstensi .html tidak akan dijalankan oleh PHP Interpreter karena akan dianggap sebagai dokumen HTML biasa

```
<?php
    echo 'Hello World!';
    for($i=1; $i<=10; $i++){
        echo 'PHP is easy!';
    }
?>
```

# PHP Variable

- Diawali dengan **\$ (lambang dollar)**
- Bersifat dinamis dan loosely type
- Nama variabel bersifat case-sensitive
- Type variable pada PHP int, float, boolean, string, array, object, null
- Type string bisa menggunakan `' '` (petik tunggal) atau `" "` (petik ganda)
- Nama harus dimulai dengan huruf atau underscore ( `_` )
- Auto conversion
  - `$a = "1" + 1;` à dari string ke int (nilai `$a` = 2)
  - `$b = 3/2;` à dari int ke float (nilai `$b` = 1.5)
- Bisa juga menggunakan type casting
  - `$usia = (int) "21";` à dari string ke int

# Predefined variables

Pada PHP terdapat beberapa variables yang sudah terdefiniskan dan dapat diakses

**`$_SERVER`, `$_GET`, `$_POST`, `$_REQUEST`, `$_FILES`, `$_SESSION`, `$_COOKIE`, `$_ENV`**

## **`$_SERVER`**

- Berisi informasi server dan request yang sedang ditangani (berupa array)
- informasi tentang elemen – elemen (headers, path, lokasi script, dll) pada server.

## **`$_FILES`**

- Untuk mengakses variabel-variabel yang berhubungan dengan proses upload file

## **`$_SESSION`**

- Untuk mengakses variabel-variabel yang digunakan oleh beberapa halaman web yang berbeda. Biasanya digunakan untuk mengelola kegiatan pengguna dalam suatu website

# Predefined Variables

## `$_GET` dan `$_POST`

- Untuk mengakses variabel-variabel yang dikirimkan menggunakan form (saat submit form)
- Penggunaan tergantung pada atribut method pada form, menggunakan get atau post
- Contoh:

```
.....  
<form action="cek.php" method="get">  
    <input type="text" name="nama">  
    .....  
</form>
```

- //pada cek.php (Script PHP)  
`$nama = $_GET['nama'];`

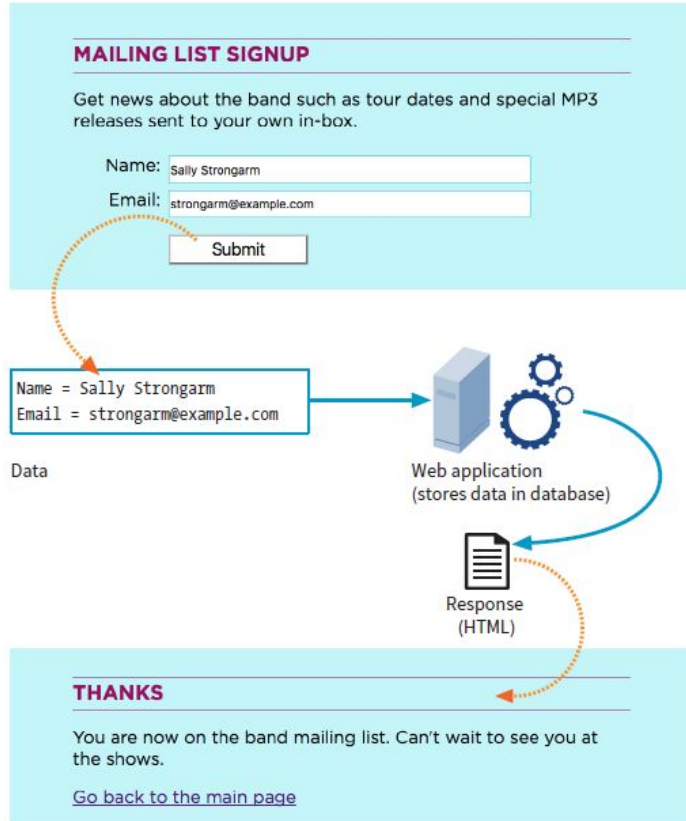
# Input PHP

PHP dapat menerima masukan dari berbagai macam sumber:

- Input dari Parameter URL
- **Input dari Form (masukan dari pengguna)**
- Input dari Cookies
- Input dari Session
- Input dari File



# FORM HTML



Pada bagian ini form HTML yang akan dibahas bersifat statis. Nanti saat PHP, kita akan melengkapi form dinamis.

Secara umum, gambar di samping menunjukkan cara kerja form secara dinamis.

# HTML Form

- Untuk mengirimkan data yang diisikan oleh pengguna
- Bergantung kepada HTML Input Type
- Tag **<form method="POST" action="submit.php">**
- Method yang digunakan biasanya adalah POST dan GET. Action adalah file php yang digunakan untuk memproses inputan pengguna. Dapat berisi syntax untuk insert, update, delete atau insert



# ELEMEN FORM HTML

```
<form action="form.php" method="get">
  <input type="hidden" name="inihidden" id="idhidden" value="NILA">
  username: <input type="text" name="iniuser" id="iduser">
  password: <input type="password" name="inipass" id="idpass">
  <input type="submit" value="KLIK ME">
  <input type="reset" value="RESET">
</form>
```

Atribut **action** digunakan untuk URL yang akan memproses isi formulir tersebut.

Atribut **method** digunakan untuk menentukan bagaimana informasi dikirimkan ke server. Bisa menggunakan POST atau GET

**Submit** yang berfungsi sebagai trigger (pemicu) pengiriman data dari form ke pemroses

Proses pengiriman data ke file form.php dengan menggunakan method POST, dengan data yang dibawa adalah yang berada di dalam form. Nama variabel akan menggunakan attribute **name**



# ELEMEN FORM HTML

```
<form action="form.php" method="post">  
  <input type="hidden" name="inihidden" id="idhidden" value="NILA">  
  username: <input type="text" name="iniuser" id="iduser">  
  password: <input type="password" name="inipass" id="idpass">  
  <input type="submit" value="KLIK ME">  
  <input type="reset" value="RESET">  
</form>
```

Berdasarkan kode diatas:

- a. **Berapa data yang dikirimkan ke file form.php?**
- b. **Bagaimana cara form.php menangkap data-data tersebut?**

# \$\_POST

## FILE index.php

```
<form action="form.php" method="post">
    <input type="hidden" name="inihidden" id="idhidden" value="NILA">
    username: <input type="text" name="iniuser" id="iduser">
    password: <input type="password" name="inipass" id="idpass">
    <input type="submit" value="KLIK ME">
    <input type="reset" value="RESET">
</form>
```

username:  password:

## FILE form.php

```
<?php
    print_r($_POST);
?>
```

Array ( [inihidden] => NILA [iniuser] => admin [inipass] => admin )

-----

Jika ingin akses satu persatu

```
<?php
    $hidden = $_POST['inihidden'];
    $username = $_POST['iniuser'];
    $password = $_POST['inipass'];
    echo "Data hidden: $hidden <br/> dengan user:
$username dan pass=$password"
?>
```

**Data hidden: NILA  
dengan user: admin dan pass=admin**

# \$\_GET



localhost/Progweb/PHPDua/2022/form.php?inihidden=NILA&iniuser=admin&inipass=admin

## FILE index.php

```
<form action="form.php" method="get">
    <input type="hidden" name="inihidden" id="idhidden" value="NILA">
    username: <input type="text" name="iniuser" id="iduser">
    password: <input type="password" name="inipass" id="idpass">
    <input type="submit" value="KLIK ME">
    <input type="reset" value="RESET">
</form>
```

username:  password:

## FILE form.php

```
<?php
    print_r($_GET);
?>
```

Array ( [inihidden] => NILA [iniuser] => admin [inipass] => admin )

-----  
Jika ingin akses satu persatu

```
<?php
    $hidden = $_GET['inihidden'];
    $username = $_GET['iniuser'];
    $password = $_GET['inipass'];
    echo "Data hidden: $hidden <br/> dengan user:
$username dan pass=$password"
?>
```

**Data hidden: NILA  
dengan user: admin dan pass=admin**

# Perbedaan

Menggunakan method POST

Pada method POST, Variabel yang dikirimkan tidak ditampilkan di address

 localhost/Progweb/PHPDua/2022/form.php

Menggunakan method GET

Pada method GET, Variabel yang dikirimkan ditampilkan di address, yaitu setelah tanda tanya ?

 localhost/Progweb/PHPDua/2022/form.php?inihidden=NILA&iniuser=admin&inipass=admin

**\$\_GET tidak aman untuk data sensitif (misalnya password).**

# PHP ISSET Function

- PHP isset() function digunakan untuk mengecek apakah sebuah variabel sudah di-set atau belum
- Digunakan untuk mengecek apakah variabel kosong
- Dapat berguna untuk mengecek apakah sebuah button sudah di klik/submit
- isset() function akan mengembalikan nilai true atau false.
- isset() function menghasilkan true jika variabel ada.
- Sintax:

**isset(variabel)**

```
if(isset($_GET["Klik Me"])){  
    print_r($_GET);  
    $hidden = $_GET['inihidden'];  
    $username = $_GET['iniuser'];  
    $password = $_GET['inipass'];  
    echo "Data hidden: $hidden <br/> dengan user: $username dan  
    pass=$password";}   
else echo "Salah method kek-nya"
```

# PHP & DB

# AKSES MYSQL VIA PHP

Proses penggunaan MySQL dengan PHP:

1. Membuat koneksi ke MySQL.
2. Menentukan database yang ingin dipakai.
3. Membangun query.
4. Menjalankan query.
5. Mengambil hasil query dan menampilkan pada halaman web.
6. Melakukan tahapan 3-5 hingga mendapatkan semua data yang diinginkan.
7. Menutup koneksi MySQL.



# SELECT DATA

1. Buat sebuah halaman PHP baru dengan nama **index.php**. Tujuan akhir pada halaman ini adalah menampilkan semua mahasiswa pada sebuah tabel HTML yang diambil dari tabel mahasiswa.
1. Pertama-tama kita akan menyiapkan informasi yang dibutuhkan untuk membuat koneksi database. Fungsi `mysqli_connect()` digunakan untuk membuka koneksi SQL dengan parameter nama server, username, password, dan nama database.

```
// open connection
$servername = "127.0.0.1";
$username = "progweb";
$password = "progweb";
$databasename = "php2";
$conn = mysqli_connect($servername, $username, $password, $databasename) or die("Koneksi gagal.");
```



# SELECT DATA

3. Untuk mengambil data pada tabel mahasiswa dapat dilakukan query SELECT yang ditampung pada variabel `$sql`. Untuk menjalankan query maka dapat menggunakan fungsi `mysqli_query()` dengan parameter koneksi dan query SQL. Hasil dari query ditampung pada variabel `$result`.

```
//SELECT  
$sql = "SELECT * FROM mahasiswa";  
$result = mysqli_query($conn, $sql); //untuk run query
```

# SELECT DATA

4. Untuk memastikan query tersebut berhasil jalan, maka dapat dilakukan pengecekan dengan melihat jumlah baris yang dihasilkan dari query dengan fungsi `mysqli_num_rows()`.

```
if(mysqli_num_rows($result) > 0) {  
  
} else {  
    echo "Tabel mahasiswa kosong.";  
}
```

# SELECT DATA

5. Berikutnya akan disiapkan sebuah tabel HTML yang akan digunakan untuk menampung hasil query. Tabel ini akan dibuat ketika hasil dari pengecekan baris lebih dari 0. Buat header untuk tabel daftar mahasiswa. Header yang ditampilkan adalah NIM, nama, jenis kelamin, dan sebuah kolom tambahan untuk edit dan delete.

```
if(mysqli_num_rows($result) > 0) {  
    ?>  
    <table border=1>  
        <thead>  
            <th>NIM</th>  
            <th>Nama</th>  
            <th>Jenis Kelamin</th>  
            <th>Action</th>  
        </thead>  
    </table>  
    <?php  
} else {  
    echo "Tabel mahasiswa kosong."  
}  
?>
```

# SELECT DATA

6. Selanjutnya akan dilakukan perulangan pada setiap baris dari hasil query dan setiap baris akan dimasukkan pada kolom yang sesuai menggunakan tag `<tr>` dan `<td>`. Untuk mengambil data per baris fungsi `mysqli_fetch_assoc()` dapat digunakan. Pada kasus ini setiap baris ditampung pada variabel `$row`. Untuk mengakses data per kolom pada baris tersebut dapat menggunakan `$row["nama_kolom"]`.

```
<?php
while($row = mysqli_fetch_assoc($result)) {
    echo "<tr>";
    echo "<td>".$row["nim"]."</td>";
    echo "<td>".$row["nama"]."</td>";
    echo "<td>".$row["jenis_kelamin"]."</td>";
    echo "<td></td>";
    echo "</tr>";
}
?>
```

# SELECT DATA

7. Silahkan jalankan kode Anda dan lihat hasilnya!



# INSERT DATA

Insert into, atau yang sering disebut dengan CREATE data digunakan untuk menambah data mahasiswa. Pada bagian ini kita akan mencoba membuat sebuah form yang digunakan untuk menerima data mahasiswa baru. Logika berpikir untuk tahap ini:

1. Tampilkan form input
2. Pada saat disubmit, periksa inputan sudah sesuai atau belum.
3. Bila input sudah sesuai, buat koneksi ke database, kemudian isikan data menggunakan query INSERT
4. Berikan notifikasi input sudah berhasil disimpan
5. Bila pada langkah 2 masih ada data yang tidak sesuai, berikan notifikasi kesalahan



# INSERT DATA

1. Buat halaman **form-mhs.php**, halaman ini akan menampilkan sebuah form untuk melakukan edit dan insert data mahasiswa.
2. Isikan form dengan input type masing-masing field, berikan parameter action ke file yang akan mengelola input tersebut. Pada kasus ini file yang mengelola input adalah form-mhs.php atau dirinya sendiri.

```
<form action="form-mhs.php" method="POST">
  nim : <input type="number" name="nim" max="8"> <br>
  nama : <input type="text" name="nama" max="50"> <br>
  kelamin : <input type="radio" value="L" name="jenis_kelamin">Laki-laki
            <input type="radio" value="P" name="jenis_kelamin">Perempuan <br>
            <input type="submit">
</form>
```

Tambahkan validasi pada HTML untuk:

- NIM harus 8 karakter, hanya boleh angka saja, dan harus diisi
- Nama maksimum 50 karakter dan harus diisi

# INSERT DATA

3. Karena `action="form-mhs.php"` atau dirinya sendiri, maka pada bagian atas, sebelum tag `<html>`, tambahkan untuk membuka koneksi

```
// open connection
$servername = "127.0.0.1";
$username = "progweb";
$password = "progweb";
$databasename = "php2";
$conn = mysqli_connect($servername, $username, $password, $databasename) or die("Koneksi gagal.");
```



# INSERT DATA

4. Selanjutnya, kita melakukan pengecekan, apakah ada data pada variable `$_POST`. Bila ada, maka lakukan proses insert ke database. Variabel ini merupakan variabel global yang dipakai untuk metode POST. Selanjutnya query akan dieksekusi dan dicek apakah berhasil menambahkan data atau tidak dengan fungsi `mysqli_query()`.

```
if($_POST){  
    $nim = $_POST["nomer"];  
    $nama = $_POST["nama"];  
    $kelamin = $_POST["jenis_kelamin"];  
    $sql = "INSERT INTO mahasiswa (nim, nama, jenis_kelamin) VALUES ('".$nim."', '".$nama."', '".$kelamin."')";  
    if(mysqli_query($conn, $sql)) {  
        echo "Berhasil mengisi data.";  
    } else {  
        echo "Gagal mengisi data.";  
    }  
}
```

# INSERT DATA

Variabel `$_POST` merupakan sebuah variabel global yang digunakan pada metode POST. Variabel tersebut dapat mengakses data pada formulir dan merupakan associative array. Maka apabila pada formulir tersebut memiliki teksboks "Nama" dengan name = "nama" maka dapat diakses dengan `$_POST["nama"]`.

```
if($_POST){
    $nim = $_POST["nomer"];
    $nama = $_POST["nama"];
    $kelamin = $_POST["jenis_kelamin"];
    $sql = "INSERT INTO mahasiswa (nim, nama, jenis_kelamin) VALUES ('".$nim."', '".$nama."', '".$kelamin."')";
    if(mysqli_query($conn, $sql)) {
        echo "Berhasil mengisi data.";
    } else {
        echo "Gagal mengisi data.";
    }
}
```

# INSERT DATA

5. Apabila sudah selesai, pastikan koneksi ditutup.

```
// close connection  
mysqli_close($conn);
```



# UPDATE DATA

UPDATE data digunakan untuk mengubah data mahasiswa. Pada bagian ini kita akan menggunakan form mahasiswa yang sudah dibuat pada tahap sebelumnya. Logika berpikir untuk tahap ini:

1. Pada halaman SELECT atau daftar mahasiswa, kolom action, tambahkan sebuah tombol/link menuju ke halaman form dengan membawa parameter GET id dari data yang hendak di edit.
2. Pada form mahasiswa, apabila ada data GET id, maka tambahkan sebuah `<input type="hidden" value="id">`
3. Pada saat disubmit, Periksa inputan sudah sesuai atau belum.
4. Bila input sudah sesuai, buat koneksi ke database, kemudian isikan data menggunakan query UPDATE dengan WHERE adalah id dari input type hidden
5. Berikan notifikasi input sudah berhasil diubah
6. Bila pada langkah 3 masih ada data yang tidak sesuai, berikan notifikasi kesalahan

# UPDATE DATA

1. Buka file **index.php**.
1. Pada bagian perulangan data, kolom terakhir, tambahkan link untuk ke halaman form-mhs.php. Yang perlu diperhatikan adalah, id data GET yang dikirimkan.

```
while($row = mysqli_fetch_assoc($result)) {  
    echo "<tr>";  
    echo "<td>".$row["nim"]."</td>";  
    echo "<td>".$row["nama"]."</td>";  
    echo "<td>".$row["jenis_kelamin"]."</td>";  
    echo "<td>";  
        echo "<a href='form-mhs.php?id=".$row['id']."'>Edit</a>";  
    echo "</td>";  
    echo "</tr>";  
}
```

# UPDATE DATA

3. Pada halaman **form-mhs.php**, buat sebuah percabangan, bila ada data GET, maka proses pengambilan data awal/data yang hendak diubah. Lakukan pencarian ke database untuk mahasiswa yang memiliki id GET. `$_GET` merupakan variabel super global yang digunakan untuk mengambil data dari form dengan metode GET. Variabel ini penggunaannya sama dengan variable `$_POST`.

```
if($_GET){
    $id = $_GET['id'];
    $sql = "SELECT * FROM mahasiswa WHERE id='".$_GET['id']."'";
    $result = mysqli_query($conn, $sql);
    if(mysqli_num_rows($result) > 0) {
        $row = mysqli_fetch_assoc($result);
        $oldNim = $row["nim"];
        $oldNama = $row["nama"];
        $oldKelamin = $row["jenis_kelamin"];
    }
    else {
        echo "Data yang hendak diedit tidak ada.";
    }
}
```

# UPDATE DATA

4. Pada bagian `<form>`, tambahkan input hidden untuk id, dan isikan data awal berdasarkan data dari database.

```
<form action="form-mhs.php" method="POST">
  <?php
    if($id != null) {
      <input type="hidden" name="id" value="<?php echo $id; ?>">
    }
  ?>
  nim : <input type="number" name="nim" max="8" value="<?php echo $oldNim; ?>"> <br>
  nama : <input type="text" name="nama" max="50" value="<?php echo $oldNama; ?>"> <br>
  kelamin : <input type="radio" value="L" name="jenis_kelamin" <?php echo ($oldKelamin=='L')?'checked':''; ?> >Laki-laki
            <input type="radio" value="P" name="jenis_kelamin" <?php echo ($oldKelamin=='P')?'checked':''; ?> >Perempuan <br>
            <input type="submit">
</form>
```

# UPDATE DATA

5. Pada bagian INSERT data, buat percabangan untuk UPDATE data.

```
if($_POST){  
    if($_POST['id'] != null){  
        //update  
  
    } else {  
        //insert  
        $nim = $_POST["nim"];  
        $nama = $_POST["nama"];  
        $kelamin = $_POST["jenis_kelamin"];  
        $sql = "INSERT INTO mahasiswa (nim, nama, jenis_kelamin) VALUES ('.$nim.', '.$nama.', '.$kelamin.')";  
        if(mysqli_query($conn, $sql)) {  
            echo "Berhasil mengisi data.";  
        } else {  
            echo "Gagal mengisi data.";  
        }  
    }  
}
```



# UPDATE DATA

6. Tambahkan logika untuk melakukan UPDATE ke database

```
if($_POST['id'] != null){  
    //update  
    $nim = $_POST["id"];  
    $nim = $_POST["nim"];  
    $nama = $_POST["nama"];  
    $kelamin = $_POST["jenis_kelamin"];  
    $sql = "UPDATE mahasiswa set nim='".$nim."', nama='".$nama."', jenis_kelamin='".$jenis_kelamin."' WHERE id='".$id.'";  
    if(mysqli_query($conn, $sql)) {  
        echo "Berhasil mengubah data.";  
    } else {  
        echo "Gagal mengubah data.";  
    }  
} else {
```

Tambahkan link untuk kembali ke halaman index.php dari halaman update!

# DELETE DATA

DELETE data digunakan untuk menghapus data mahasiswa. Logika berpikir untuk tahap ini:

1. Pada halaman SELECT atau daftar mahasiswa, kolom action, tambahkan sebuah tombol/link yang membawa parameter GET id dari data yang hendak di hapus.
2. Lakukan query untuk DELETE data
3. Berikan notifikasi hapus sudah berhasil diubah



# DELETE DATA

1. Buka file **index.php** atau file untuk daftar mahasiswa
1. Pada bagian perulangan data, kolom terakhir, tambahkan link untuk ke halaman delete-mhs.php. Yang perlu diperhatikan adalah, id data GET yang dikirimkan

```
while($row = mysqli_fetch_assoc($result)) {  
    echo "<tr>";  
    echo "<td>".$row["nim"]."</td>";  
    echo "<td>".$row["nama"]."</td>";  
    echo "<td>".$row["jenis_kelamin"]."</td>";  
    echo "<td>";  
        echo "<a href='form-mhs.php?id=".$row['id']."'>Edit</a>";  
        echo "<a href='delete-mhs.php?id=".$row['id']."'>Delete</a>";  
    echo "</td>";  
    echo "</tr>";  
}
```

# DELETE DATA

3. Buat halaman baru **delete-mahasiswa.php**. Pada awal halaman, buat koneksi ke database.

```
// open connection
$servername = "127.0.0.1";
$username = "progweb";
$password = "progweb";
$databasename = "php2";
$conn = mysqli_connect($servername, $username, $password, $databasename) or die("Koneksi gagal.");
```

4. Lakukan pemeriksaan id pada method GET.

# DELETE DATA

5. Delete mahasiswa dengan id pada parameter GET.

```
if($_GET){  
    //DELETE  
    $id=$_GET['id'];  
    $sql = "DELETE FROM mahasiswa WHERE id='".$id."'";  
    if(mysqli_query($conn, $sql)) {  
        echo "Berhasil menghapus data. Jumlah data dihapus=".mysqli_affected_rows($conn);  
    } else {  
        echo "Gagal menghapus data data.";  
    }  
}
```

# DELETE DATA

6. Lakukan close connection.

```
// close connection  
mysqli_close($conn);
```

Tambahkan link untuk kembali ke halaman index.php dari halaman update!

# AKTIVITAS KELAS

Pastikan proses CRUD jalan!

Aktivitas tambahan: pada tiga halaman yang sudah kita buat, terdapat koneksi database yang terus menerus ditulis di atas. Bagaimana caranya agar tidak perlu melakukannya berkali-kali? Cari cara praktis untuk melakukan ini!



# THANKS!

Ada pertanyaan?

