|  |  |
| --- | --- |
| **Materi** | **Nilai** |
| PHP FRAMEWORK CI minggu ke 18 (video 10-19 Tutorial Codeigniter-4) | 95 |

**Saya Sudah Belajar dan Mengerti dan Saya BISA**

1. Tutorial CodeIgniter-4 pada bagian menginsert data. Jadi sebuah data dari form akan dimasukkan ke dalam database. Untuk memulai nya pada tblkategori tambahkan 1 kolom yang bernama keterangan disamping kolom kategori type nya VARCHAR lalu values nya 100, Kemudian disave. Sebelumnya di navbar atau menu nya select yang code untuk tampilannya diletakkan di file select.php ini dibuat menjadi sebuah tabel yang akan menampilkan data dari database tblkategori. Codenya:

<table border="1px">

    <tr>

        <th>No</th>

        <th>Kategori</th>

        <th>Keterangan</th>

    </tr>

    <?php $no = 1; ?>

    <?php foreach ($kategori as $key => $value) : ?>

        <tr>

            <td><?= $no++ ?></td>

            <td><?= $value['kategori'] ?></td>

            <td><?= $value['keterangan'] ?></td>

        </tr>

    <?php endforeach; ?>

</table>

Karena sebelumnya file ini sudah disambungkan ke tblkategori dengan menggunakan $kategori itu jadi nanti data dari kolom kategori dan kolom keterangan yang ada di tblkategori akan terlooping ke dalam tabelnya. Dan untuk kolom nomer dia akan melooping dari angka 1 dst. Untuk form yang akan menginsert data ke tblkategori diletakkan di navbar atau menu insert yang settingan tampilannya berada di file insert.php di dalam folder kategori yang ada di views. Code form nya seperti ini:

<form action="<?= base\_url() ?>/admin/kategori/insert" method="post">

    Kategori : <input type="text" name="kategori" required>

    <br>

    Keterangan : <input type="text" name="keterangan" required>

    <br>

    <input type="submit" name="simpan" value="SIMPAN">

</form>

Yang berarti di dalam form itu ada input text untuk Kategori dan Keterangan lalu input type submit untuk menyimpan dan action form ini nanti akan menyambungkan dengan function insert yang ada di folder kategori dan menjalankan isinya. Function insert ini diisi code:

public function insert()

    {

        $model = new Kategori\_M();

        $model->insert($\_POST);

        return redirect()->to(base\_url() . "/admin/kategori");

    }

Jadi function ini menerima data dari form karena method form nya post dan diterima function menggunakan $\_POST. Lalu di dalam file Kategori\_M.php di bagian isi dari class Kategori\_M nya ditambahi dengan protected $allowedFields = ['kategori', 'keterangan']; yang berarti dia akan menginsert data dari input text yang bernama kategori dan keterengan yang diambil dari function nya $model->insert($\_POST); dan function nya mengambil dari form yang ada di file insert.php. Selanjutnya setelah menginsert data pasti data ini juga otomatis masuk ke dalam tabel yang tampil di menu select, karena itu ditambahkan return redirect()->to(base\_url() . "/admin/kategori"); agar setelah menginsert langsung membuka tampilan dari menu select yang url nya localhost/restoran-ci4/admin/kategori.

2. Tutorial CodeIginiter-4 pada bagian Delete Data. Untuk menghapus data mula-mula di dalam file Kategori\_M.php yang ada di model ditambahkan protected $primaryKey = 'idkategori'; di dalam class Kategori\_M untuk mengambil data idkategori dari tblkategori, selanjutnya di dalam file select.php pada bagian tabelnya ditambahkan th untuk hapus dengan td nya seperti ini: <td><a href="<?= base\_url() ?>/admin/kategori/delete/<?= $value['idkategori'] ?>">Hapus</a></td> yang isi dari kolom ini adalah tulisan hapus dan apabila ditekan dia akan menjalankan url -localhost/restoran-ci4/admin/kategori/delete/(idkategori sesuai data yang mau dihapus). Karena code itu tadi menjalankan function delete, maka di dalam function delete nya diisi dengan:

$model = new Kategori\_M();

$model->delete($id);

return redirect()->to(base\_url() . "/admin/kategori");

yang berarti dia akan menyambungkan dengan class Kategori\_M yang ada di model dan mengambil tindakan untuk menghapus data sesuai dengan id yang dipilih lalu setelah mengahapus data akan langsung menuju ke tampilan sesuai redirect nya yaitu tampilan menu select.

3. Tutorial Codeigniter-4 pada bagian Update Data. Untuk mengupdate data mula-mula di dalam file select.php nya ditambahkan 1 kolom lagi untuk tempat update data nya, jadi th nya “ubah” dan td nya sama seperti td dari hapus, bedanya pada nama nya yaitu “ubah” dan pada url nya menggunakan function find. Lalu didalam function find yang ada di class Kategori, isinya seperti ini

public function find($id = null)

    {

        $model = new Kategori\_M();

        $kategori = $model->find($id);

        $data = [

            'judul' => 'UPDATE DATA',

            'kategori' => $kategori

        ];

        return view("kategori/update", $data);

    }

Jadi form ini akan mencari baris mana yang mau diubah dengan mengambil idkategorinya lalu apabila sudah ketemu selanjutnya dia akan menampilkan atau ke view folder kategori bagian file update.php, di dalam file ini terdapat sebuah form seperti ini

<form action="<?= base\_url() ?>/admin/kategori/update" method="post">

    Kategori : <input type="text" name="kategori" value="<?= $kategori['kategori'] ?>" required>

    <br>

    Keterangan : <input type="text" name="keterangan" value="<?= $kategori['keterangan'] ?>">

    <br>

    <input type="hidden" name="idkategori" value="<?= $kategori['idkategori'] ?>">

    <input type="submit" name="simpan" value="SIMPAN">

</form>

Ada input text untuk mengubah data dari kolom kategori, keterangan, dan untuk idkategorinya tetapi disembunyikan lalu submit bernama simpan, setelah mengisi form dan menekan submit selanjutnya dia akan menjalankan actionnya yang mengarah pada function update di dalam class Kategori. Di dalam function update isinya

public function update()

    {

        $model = new Kategori\_M();

        $model->save($\_POST);

        return redirect()->to(base\_url() . "/admin/kategori");

    }

Yang berarti dengan function ini dia akan mengirimkan data dengan menyimpan lalu memposting dari form tadi ke dalam tblkategori. Setelah melakukan update dia kembali ke halaman sesuai return redirect nya yaitu ke tampilan select.php. Karena update ini dimasukkan ke dalam tabel, maka di admin.php yang berisi tempat dari navbar untuk nav ul li nya update dihapus saja.

4. Tutorial CodeIgniter-4 pada bagian upload image atau upload gambar. Untuk mengupload sebuah gambar, dimulai dari membuat folder baru di dalam views yang bernama Menu, lalu di dalam menu dibuat form.php yang isinya berupa form seperti ini

<form action="<?= base\_url('/admin/menu/insert') ?>" method="post" enctype="multipart/form-data">

    Gambar : <input type="file" name="gambar" required>

    <br>

    <input type="submit" name="simpan" value="SIMPAN">

</form>

Jadi form ini berupa input file yang berfungsi untuk mengambil gambar dari penyimpanan computer lalu setelah menekan submit dia akan menuju sesuai action nya yaitu ke function insert yang ada di class menu di dalam folder admin controllers dan dengan adanya enctype="multipart/form-data"> untuk membawa data gambar yang akan diproses ke function insert. Di dalam function insert ini berisi:

public function insert()

    {

        $file = $this->request->getFile('gambar');

        $name = $file->getName();

        $file->move('./upload');

        echo $name . "Sudah diupload";

    }

Yang berarti akan mengambil sebuah file sesuai dengan yang dipilih dari input file bernama ‘gambar’ lalu diberi nama sesuai dengan nama yang dipilih tadi dan file ini dipindahkan(move) ke folder upload yang otomatis terbuat di dalam restoran-ci4. Dan apabila sudah diupload dan masuk ke dalam folder upload selanjutnya tampilan browser akan muncul tulisan (nama dari file) Sudah diupload.

5. Tutorial CodeIgniter-4 pada bagian Session. Untuk mengaktifkan session dibuatkan sebuah construct yang dapat digunakan untuk semua function agar session yang ada di dalam functionnya berfungsi. Codenya:

protected $session = null;

public function \_\_construct()

{

$this->session = \Config\Services::session();

}

Untuk itu agar setiap session yang ada di dalam function yang berbeda-beda berfungsi maka ditambahkan $this-> lalu session karena letak construct yang menyimpan session ini berada di luar function lainnya. Terdapat session create untuk pembuatan session,

public function create()

    {

        $tbluser = [

            'user'  => 'koki',

            'email'     => 'koki@gmail.com',

            'level'     => 'koki'

        ];

        $this->session->set($tbluser);

    }

Jadi dia akan membuat sebuah session dengan menggunakan session->set, lalu ada session yang berfungsi untuk membaca session yaitu session read,

public function read()

    {

        echo $this->session->get('user');

        echo "<br>";

        echo $this->session->get('email');

        echo "<br>";

        echo $this->session->get('level');

    }

Jadi dia akan membaca atau menampilkan sebuah session dengan menggunakan session->get, lalu ada session yang berfungsi untuk menghapus session yang dipilih dari data session yang sudah dibuat seperti ini

public function delete()

    {

        $this->session->remove('email');

    }

Jadi dia akan menghapus session yang bernama email dengan menggunakan session->remove, lalu ada session yang berfungsi untuk menghapus semua session yang sudah dibuat seperti ini

public function destroy()

    {

        $this->session->destroy();

    }

Jadi dengan menggunakan session->destroy dia akan menghapus seluruh data dari session yang telah dibuat, apabila ingin mengembalikan session yang sudah dibuat tadi maka kembali menggunkana function create yang berisi session->set.

6. Tutorial CodeIgniter-4 pada bagian Validation. Validation ini digunakan untuk memeriksa apakah sebuah data itu diijinkan untuk masuk ke sebuah database atau tidak. Jadi misalkan di dalam kolom kategori memiliki sebuah data dan semua data ini harus berbeda-beda dan tidak boleh ada data ganda yang ada didalamnya (jika sudah ada data itu maka tidak boleh menggunakan data yang sama) maka menggunakan validation. Untuk itu di dalam class Kategori\_M yang ada di dalam models ditambahkan

protected $validationRules = [

        'kategori' => 'alpha\_numeric\_space|min\_length[3]|is\_unique[tblkategori.kategori]'

    ];

    protected $validationMessages = [

        'kategori' => [

            'alpha\_numeric\_space' => 'Tidak boleh menggunakan simbol',

            'min\_length' => 'Minimal 3 huruf',

            'is\_unique' => 'Data sudah digunakan'

        ]

    ];

Dengan menambahkan itu, dari $validationRules ini berfungsi untuk mengatur agar data yang akan diinsert ke kolom kategori harus berupa huruf, angka, dan spasi (alpha\_numeric\_space) lalu huruf yang diinsert minimal harus tiga angka (min\_length[3]) dan data yang dimasukkan tidak boleh sama dengan yang sudah terdata di tblkategori bagian kategorinya (is\_unique[tblkategori.kategori]). Setelah itu apabila memasukkan insert data yang melawan aturan dari $validationRules maka $validationMessages berfungsi untuk memberikan pesan ketika terjadi kesalahan/error sesuai dibagian mana error nya. Agar 2 variable ini berfungsi maka di dalam function insert yang ada pada class Kategori di dalam controllers isinya diubah menjadi

$model = new Kategori\_M();

if ($model->insert($\_POST) === false) {

$error = $model->errors();

echo $error['kategori'];

} else {

   return redirect()->to(base\_url("/admin/kategori"));

}

Jadi ditambahkan sebuah if yang apabila data yang dimasukkan salah atau melawan aturan dari $validationRules maka dia akan menjalankan error dari kategori yang ada di $validationRules dan $validationMessages. Dan apabila insert sudah benar, berjalan, dan berhasil, maka dia akan kembali ke tampilan awal sesuai return redirect nya yaitu di tampilan menu kategori.

7. Tutorial CodeIgniter-4 pada bagian Menggunakan Bootstrap. Untuk mendapatkan bootstrap secara offline maka harus mendownload dulu di getbootstrap.com. seperti yang sudah dipelajari sebelumnya bahwa bootstrap ini berfungsi untuk merapikan tampilan browser tanpa haru mengatur css secara manual di coding, jadi bootstrap telah menyediakan element-element yang bisa langsung digunakan bagi penggunanya. Untuk mengaktifkan bootstrap maka ditambahkan

<link rel="stylesheet" href="<?= base\_url('/bootstrap/css/bootstrap.min.css') ?>">

Yang diletakkan di admin.php atau tempat nya view layout untuk menu navbar itu, dengan begitu file ini akan menyambungkan dengan folder bootstrap masuk ke folder css dan mengambil bootstrap.min.css untuk digunakan di browsernya nanti. Karena letak admin.php ini berada di dalam folder template yang ada di view maka link untuk menyambungkan ke bootstrapnya menggunakan base\_url yang berarti localhost/restoran-ci4/ lalu masuk ke folder bootstrap dan seterusnya. Dengan adanya base\_url ini tidak perlu mencari letak folder bootstrap dari file admin.php dan menyambungkannya secara manual. Jika sudah menyambungkan dengan bootstrap maka bisa ditambahkan container, row, col, dll sesuai keinginan yang nanti akan langsung otomatis diatur oleh bootstrap di browsernya.

8. Tutorial CodeIgniter-4 pada bagian Paging. Jadi paging ini adalah membagi data yang ada di dalam kolom ke beberapa halaman atau menjadi beberapa halaman agar perhalamannya tidak menampilkan terlalu banyak data. Untuk menggunakan paging bisa copy code dari codeigniter.com yang bagian ini $pager = \Config\Services::pager(); diletakkan di dalam function read dalam class Kategori, code ini berfungsi untuk menjalankan services nya dulu atau menjalankan paging ini, apabila sudah jalan masih di dalam function read pada bagian $data nya di dalam kurung staplesnya ditambahkan

'kategori' => $model->paginate(2, 'group1'),

'pager' => $model->pager

Yang berarti dia akan mengatur halamannya menggunakan paginate itu, setiap halamannya hanya memuat 2 data, jadi misalkan ada 6 data, dan tiap halamannya hanya 2 data, maka halaman yang terbuat adalah 3 halaman. Lalu ada group1 itu adalah nama yang digunakan untuk link yang nantinya bisa disambungkan ke file lain. Setelah itu masuk ke dalam select.php dibawah tabel ditambahkan <?= $pager->links('group1', 'bootstrap') ?> yang berarti dia menyambungkan paging yang diambil dari link gorup1 tadi. Lalu ada bootstrap yang digunakan untuk mempercantik tampilan dari paging nya. Bagaimana link tau bootstrap ini akan menyambungkan ke file bootstrapnya? Maka pada folder app masuk ke folder config lalu cari Pager.php pada bagian public $templates nya ditambahkan 'bootstrap'     => 'App\Views\template\paging' yang berarti bootstrap ini menyambungkan ke file paging.php yang ada di folder template pada view. Maka di dalam file paging.php diisi sebuah code yang dicopas dari codeigniter.com dimana code ini memiliki fungsi untuk mengatur tampilan apada bagian pagingnya.

9. Tutorial CodeIgniter-4 pada bagian View Cells. Jadi view cell ini berfungsi untuk memudahkan apabila ada kesalahan mudah untuk dicari dimana kesalahannya dan bisa memotong-motong untuk dimasukkan ke dalam view\_cell apabila di dalam view terlalu banyak. Misalkan di dalam file form.php yang ada di dalam view ini ditambahkan

<div class="row">

    <?= view\_cell('\App\Controllers\Admin\Menu::option') ?>

</div>

Yang berarti dia akan menjalankan function option yang ada di dalam class Menu, selanjutnya di dalam function option ini berisi

public function option()

    {

        $model = new Kategori\_M();

        $kategori = $model->findAll();

        $data = [

            'kategori' => $kategori

        ];

        return view('template/option', $data);

    }

yang berarti dia akan menampilkan file option yang ada di dalam folder template di dalam view dan $data nya untuk digunakan di file option, di dalam file option berisi

<select name="kategori" id="kategori">

    <?php foreach ($kategori as $key => $value) : ?>

        <option value="<?= $value['idkategori'] ?>">

            <td><?= $value['kategori'] ?></td>

        </option>

    <?php endforeach; ?>

</select>

Di dalam file option ini merupakan tempat untuk membuat select option yang isinya diambil dari tblkategori dengan mengambil data idkategorinya lalu yang ditampilkan adalah data dari kategorinya. Jadi apabila di dalam form.php terlalu banyak code, maka dibuat view\_cell yang menyambungkan dengan function option dan function option menyambungkan dengan file option.php, kesimpulannya di dalam tampilan browser form.php juga akan tampil bagian untuk tampilan dari file option.php.

10. Tutorial CodeIgniter-4 pada bagian Session Flashdata. Jadi Session Flashdata ini digunakan untuk menyimpan pesan eror diatas dari formnya apabila data yang digunakan tidak sesuai ketentuan dan bisa di echo di file lain. Untuk itu cara mengaktifkannya di dalam function insert

public function insert()

    {

        $model = new Kategori\_M();

        if ($model->insert($\_POST) === false) {

            $error = $model->errors();

**session()->setFlashdata('info', $error['kategori']);**

**return redirect()->to(base\_url("/admin/kategori/create"));**

        } else {

            return redirect()->to(base\_url("/admin/kategori"));

        }

    }

Apabila terjadi kesalahan maka dia akan menyimpan apa kesalahannya dengan key yang bernama info, jadi seperti key nya adalah info dan value nya $error[‘kategori’] lalu setelah mengetahui error dia akan kembali menampilkan function create yang ada di dalam class Kategori. Tetapi ketika dia menampilkan function create dia tidak akan menampilkan pesan error nya bagian apa, untuk itu di dalam file insert.php ditambahi

<?php

echo session()->getFlashdata('info');

?>

Yang berarti dia akan menampilkan key info yang bervalue pesan error nya, jadi ketika terjadi kesalahan dia akan kembali menampilkan form create nya dan dia juga menampilkan pesan error nya diatas form.

**Saya Belum Mengerti**

1.