LAPORAN PRAKTIKUM

**Jobsheet-2**

**Read dan Input Database serta Form Validation**

**Mata Kuliah Pemrograman Web Lanjut**



Oleh:

MINGGAR PUTRA DHEA R. NIM. 1841720111

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

**2020**

|  |  |
| --- | --- |
|  | Jurusan Teknologi Informasi Politeknik Negeri Malang  **Jobsheet-2: Read dan Input Database serta Form Validation**  **Mata Kuliah Pemrograman Web Lanjut**  **Pengampu:** Tim Ajar Pemrograman Web Lanjut  *Februari 2020* |

**Topik**

Menampilkan dan Menambahkan Data ke *Database* menggunakan CodeIgniter serta Menambahkan Form Validation

**Tujuan**

Mahasiswa diharapkan dapat:

1. Memahami cara menampilkan data dari *database* menggunakan CodeIgniter
2. Memahami cara menambahkan *form validation* menggunakan CodeIgniter
3. Memahami cara menambahkan data ke *database* menggunakan CodeIgniter

**Praktikum – Bagian 1: Menampilkan data dari *database* menggunakan codeIgniter**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Buka kembali folder CodeIgniter yang telah dibuat pada Jobsheet sebelumnya |
| 2 | Buka *controller* mahasiswa dan tambahkan *code* berikut (*line* 9)    Keterangan :   * Pada line 9 menjelaskan bawa *function* index ini akan melakukan *load* database. *Load* database ini hanya dilakukan di *function* index saja jika ada *function* lain maka kita harus melakukan *load* database lagi. Sehingga hal ini dirasa tidak praktis maka untuk itu kita harus tambahkan c*ode* agar *load* database ini dapat digunakan banyak *function.* |
| 3 | Tambahkan *code* berikut agar database dapat digunakan banyak *function* (*line* 7-12)    Keterangan:   * *Line* 7 – 12 adalah sebuah *function* construct (**PERHATIKAN ITU** **MENGGUNAKAN DOUBLE UNDERSCORE \_ \_ construct (tanpa** **spasi)**) * *Function* construct dijalankan pada saat classnya diinstansiasi * *Line* 10 digunakan untuk menjalankan fungsi construct pada class parent dimana class parent ini adalah CI\_Controller * *Line* 11 digunakan agar untuk mengload database (Semua fungsi di class mahasiswa dapat menggunakannya) * Jika kita sudah melakukan load database pada saat *function* construct maka pada *function* index kita comment agar tidak bekerja dua kali |
| 4 | Selain 2 cara diatas kita juga dapat melakukan proses *load* database pada **application/config/autoload** dan tambahkan pada database pada **autoload[‘libraries’]=array(‘database’)** seperti pada gambar |
| 5 | Untuk kasus ini kita gunakan cara yang terakhir (no 4) karena kita asumsikan bahwa semua *controller* membutuhkan database. Maka kita *comment code* function construct yang ada di *controller* mahasiswa |
| 6 | Setelah itu kita akan mengatur konfigurasi database. Untuk mengatur konfigurasi database maka buka folder **application/config/database.php** |
| 7 | Kita set konfigurasinya seperti pada gambar dibawah ini    Keterangan :   * *Line* 78 artinya *hostname* yang digunakan adalah localhost * *Line* 79 artinya *username* (phpmyadmin) yang digunakan apa * *Line* 80 artinya *password* (phpmyadmin) yang digunakan apa * *Line* 81 artinya nama database yang nanti digunakan |
| 8 | Jika kita lihat konfigurasi database yang kita buat maka kita perlu sebuah database dengan nama **kuliah\_CI**. Maka buat lah sebuah database dengan nama kuliah\_CI dan dengan tabel mahasiswa seperti pada gambar dibawah ini |
| 9 | Tambahkan beberapa data ke dalam tabel mahasiswa seperti pada gambar (pada contoh terdapat 3 data, jumlah data lebih banyak tidak masalah) |
| 10 | Setelah kita membuat sebuah *database*, maka langkah berikutnya adalah membuat sebuah model untuk melakukan proses *database*. Di dalam model ini kita akan membuat sebuah *function* untuk melakukan perintah ambil data dari *database*. Untuk lebih jelasnya ikuti langkah-langkah berikut:   1. Buatlah sebuah file model dengan nama **mahasiswa\_model** pada folder **application/models**      1. Tambahkan *function* getAllmahasiswa pada mahasiswa\_model.php     Keterangan :   * Pada *line* 11 membuat sebuah query *builder* yang digunakan untuk mengambil semua data * <https://www.codeigniter.com/user_guide/database/query_builder.html#selecting-data> * Pada *line* 15 digunakan untuk mengenerate hasil query dalam bentuk array * <https://www.codeigniter.com/user_guide/database/results.html> * <https://www.codeigniter.com/user_guide/database/results.html#result-arrays>  1. Selain menggunakan query builder kita juga bisa menggunakan cara lain seperti gambar dibawah ini |
| 11 | Setelah kita mempunyai sebuah model dengan nama **mahasiswa\_model** maka langkah berikutnya kita akan *load* model ini ke dalam ***controller* mahasiswa**.    Keterangan :   * *Line* 17 digunakan untuk melakukan *load* model dengan nama modelnya adalah **mahasiswa\_model** * *Line* 22 artinya ada data yang berupa array dengan isi arraynya adalah mahasiswa dimana terdapat isi berupa function **getAllmahasiswa()** yang ada pada model dengan nama **mahasiswa\_model** * *Line* 24 tambahkan **$data** |
| 12 | Jika kita lakukan var\_dump($mahasiswa) pada **index.php** pada **views/mahasiswa/index**    Keterangan   * $mahasiswa didapat dari mana ??? -> jika kita lihat *line* 22 pada *controller* mahasiswa terdapat [mahasiswa] * Jika kita lihat pada *controller* mahasiswa terdapat perintah/c*ode* seperti berikut: • $data['mahasiswa']=$this->mahasiswa\_model->getAllmahasiswa(); (*Code* pertama) • $this->load->view('mahasiswa/index',$data); (*Code* kedua) * Jika kita lihat *code* kedua, ada $data yang isi arraynya adalah mahasiswa dan isi mahasiswa tersebut adalah ***function* getAllmahasiswa** pada model dengan nama mahasiswa\_model. Dimana nanti isi dari **$data[‘mahasiswa’]** ini dikirimkan ke halaman ***view*** dengan **2 paramater** yaitu **mahasiswa/index** dan **$data** (yang isinya mahasiswa**,** sehingga jika mau var\_dump di halaman mahasiswa/index, kita menggunakan $mahasiswa. Jika pada mahasiswa/index ke var\_dump $data maka akan terjadi *error* seperti pada gambar dibawah ini)     Hal ini disebabkan karena tidak ketahui data pada mahasiswa/index.php pada *line* 2 berupa var\_dump($mahasiswa), maka pastikan terlebih dahulu pada *controller* mahasiswa.php sudah terdapat $data di *line* 13 dan 15. |
| 13 | Jika berhasil maka data akan tampil seperti pada gambar berikut |
| 14 | Agar semua isi dari tabel dapat tampil di halaman index mahasiswa maka kita harus lakukan proses ***looping***. Gunakan c*ode* berikut pada **views/mahasiswa/index** |
| 15 | Jika berhasil maka tampilan akan seperti pada gambar dibawah ini |
| 16 | Jika kita klik Home, halaman Home belum dapat tampil. Maka dari itu kita akan set agar semua dapat saling terhubung. Untuk itu kita membutuhkan yang namanya **base\_url**. untuk lebih jelasnya ikuti langkah-langkah berikut:   1. Tambahkan *code* berikut pada **views/template/header (line 26,27,28)**     Keterangan :   * *Line* 26, 27 ,28 tambahkan base\_url (pastikan **config/config.php** sudah diset) $config['base\_url'] = 'http://localhost/12.codeIgniter/code-kedua/'; (code\_kedua adalah nama project CI jadi sesuaikan dengan nama project CI kalian) * Agar halaman web mahasiswa dapat ditampilkan saat *user* mengklik mahasiswa, maka setelah **base\_url** tambahkan ***controller* mahasiswa**  1. Agar base\_url dikenali oleh sistem maka tambahkan konfigurasi pada **config/autoload.php** tambahkan url pada **$autoload[‘helper’]=array(‘url’)**      1. Jika berhasil maka semua halaman dapat ditampilkan pada saat *user* melakukan *event click*. |
|  | * Menu Home      * Menu Data Mahasiswa      * Menu About |

**Praktikum – Bagian 2: Membuat Form Input dan Menambahkan Form Validation**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Buatlah sebuah *button* tambah data pada views/mahasiswa/index.php (*line* 3-7). Tambahkan *code* pada line 3-7 untuk membuat sebuah *button* |
| 2 | Dikarenakan **mahasiswa\_model** ini nantinya akan digunakan oleh semua *function* di *controller* mahasiswa, maka “ $this->load->model('mahasiswa\_model'); “ yang awalnya hanya berada pada **function index,** kita ubah ke ***function* construct** agar semua *function* dapat menggunakan **mahasiswa\_model**. |
| 3 | Setelah itu tambahkan ***function* tambah** pada ***controller* mahasiswa** dan tambahkan *code* berikut.  (Tips : jika menggunakan vsCode **untuk menduplikat 1 baris dapat menggunakan alt+shift+panah\_bawah**)    Keterangan :   * Pada*line* 32, **$this->load->view(‘mahasiswa/tambah’)** artinya bahwa ***function* tambah** akan melakukan *load view* pada folder mahasiswa, dengan nama filenya adalah tambah |
| 4 | Setelah kita membuat ***function* tambah** pada *controller* mahasiswa, selanjutnya buatlah ***form* tambah** pada **views/mahasiswa/tambah.php**    Keterangan :   * Pada *line* 33, terdapat atribut class dengan *value* btn btn-primary float-right yang artinya kita membuat sebuah *button* dan posisi buttonnya ada di kanan * Ganti ***value*** dari *line* 28-31 menjadi seperti di bawah ini **(Agar di tabel tidak tersimpan *value* berupa angka)** |
| 5 | Maka akan dihasilkan halaman seperti di bawah ini. |
| 6 | Setelah berhasil membuat *form* tambah data, selanjutnya buka file **index** pada **views/mahasiswa/index.php** dan kita atur **href** nya    Keterangan :   * Tambahkan *code* pada line 5 agar pada saat *user* melakukan *event* *click* maka *user* akan ditampilkan halaman tambah. |
| 7 | Tambahkan validasi menggunakan form\_validation. Contoh validasi *form* adalah seperti *field* nama harus diisi, nim harus angka dsb. Untuk lebih jelasnya ikuti langkah berikut ini :   1. Tambahkan form\_validation   https://www.codeigniter.com/user\_guide/libraries/form\_validation.html?highlight=form#the-form (*searching controller*)   1. Buka *controller* mahasiswa dan tambahkan form\_validation      1. Setelah kita tambahkan *form validation*, maka langkah berikutnya menambahkan *rule*-nya  * https://www.codeigniter.com/user\_guide/libraries/form\_validation.html?highlight=form#the-form (*searching pipe* data) * Untuk melihat jenis-jenis *rule* bisa dilihat pada ***rule reference*** * $this->form\_validation->set\_rules('fieldname','fieldlabel','trim|required|min\_length[5]|max\_length[12]'); * Jika kita lihat untuk menambahkan *rules* ada 3 parameter yaitu   1. fieldname = nama dari htmlnya, semisal <input type="text" class="form-control" id="nama" name="nama"> maka yang disebut dengan fieldname adalah nama   2. fieldlabel = ini adalah alias digunakan untuk menampilkan kesalahannya (jika terjadi *error*)   3. *Rule* yang digunakan  1. Sebelum kita tambahkan *rule*-nya maka terlebih dahulu tambahkan **atribut name** pada **views/mahasiswa/tambah**      1. Setelah kita tambahkan **atribut *name*** maka tambahkan ***rule*** pada ***controller*/mahasiswa** seperti pada gambar dibawah ini (*line* 35-46)      1. Tambahkan pesan *error* pada **views/mahasiswa/tambah** (*line* 11)      1. Hasilnya saat *field* nama tidak diisi maka akan muncul keterangan the Nama *Field is required*      1. Tambahkan *alert* agar pesan *error* lebih menarik setelah *code validation error*  * <https://getbootstrap.com/docs/4.1/components/alerts/>      * Tambahkan tanda **;** pada akhir **validation\_errors()** sehingga menjadi seperti berikut<?= validation\_errors(); ?>  1. Akan dihasilkan seperti gambar di bawah ini      1. Setelah kita berhasil memberikan ***rules validation*** pada **nama**, maka tambahkan juga untuk **nim** dan **email**.      1. Hasilnya seperti berikut ini.      1. Kita juga dapat mengecek ke *valid* an email dengan menambahkan rule pada email seperti pada gambar dibawah ini menggunakan **valid\_email (*line* 37)**      1. Hasilnya sebagai berikut.      1. Selain itu kita juga dapat memberikan *rules* harus diisi dengan angka menggunakan *rules* **numeric**      1. Akan dihasilkan alert seperti berikut ini. |

**Praktikum – Bagian 3: Menambahkan Data ke Database / Insert Data**

Setelah kita berhasil membuat **form**\_**validation** langkah berikutnya adalah memasukkan data ke database **(insert database)**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Buka **controller/mahasiswa** dan tambahkan *code* berikut pada ***function* tambah** agar pada saat **form**\_**validation** bernilai ***true,*** maka akan ditambahkan data dengan nama modelnya -> **model\_mahasiswa** dengan nama *function* **tambahdatamhs()** dan jika sudah ditambah ke database maka akan *redirect* ke *controller* mahasiswa (*line* 46 - 47) |
| 2 | Jika kita lihat *code* pada ***controller* mahasiswa(line 46)** terdapat **model** dengan nama **mahasiswa\_model** dan *function*-nya **tambahdatamhs().** Untuk itu kita harus membuat sebuah *function* dengan nama **tambahdatamhs()** |
| 3 | Buatlah sebuah *function* **tambahdatamhs()** pada **models/mahasiswa\_model**   * <https://www.codeigniter.com/user_guide/database/query_builder.html?highlight=insert#inserting-data> * Tambahkan *code* seperti pada gambar dibawah ini     Keterangan:   * *Line* 22-27 adalah sebuah array dimana nantinya isi array adalah data-data yang akan dikirim (POST) ke database * *Line* 23 “nama” => $this ->input->post(‘nama’,true) * nama->alias * Parameter nama pada post adalah name pada html seperti pada gambar dibawah ini * Parameter *true* digunakan untuk mengaktifkan htmlspecialchar      * *Line* 29 adalah proses *insert* data ke database dengan **2 parameter** yaitu **nama tablenya** dan **objectnya** (dalam hal ini data yang akan dimasukkan ke database) |
| 4 | Langkah berikutnya kita coba untuk tambahkan data.  Jika berhasil maka data akan tampil di halaman **index**.**php** a. redirect('mahasiswa','refresh');(*line* 47 gambar *controller* mahasiswa)  b. Hasil |

***-- Selamat Mengerjakan –***