

# **BAB 1**

---

## A. Definisi

### 1. PHP (*PHP Hypertext Preprocessor*)

Sejarah awal PHP adalah pendekatan dari *Personal Home Page* (situs personal), dulu PHP masih berbentuk *script* yang berfungsi sebagai pengolah data form dari *web server-side* yang memiliki sifat *open source*, dan memiliki nama *Form Interpreter* (FI). PHP (*Hypertext Preprocessor*) memiliki sintak yang mirip dengan ASP, Java, bahasa C, Perl dan memiliki kelebihan fungsi yang mudah dipahami dan spesifik. Versi terkini dari PHP adalah PHP7. Kelebihan PHP yaitu:

- a. Bahasa pemrograman PHP tidak memerlukan *Compiler* dalam penggunaanya.
- b. Memiliki sifat *open source*.
- c. Banyak sekali aplikasi PHP yang gratis dan siap untuk digunakan seperti PrestaShop, WordPress, dll.
- d. Bisa membuat web menjadi dinamis.
- e. PHP memiliki banyak dukungan dari berbagai *web server* contohnya saja Apache.
- f. PHP memiliki keunggulan lebih cepat dibandingkan dengan Java dan ASP.
- g. MySQL merupakan paket aplikasi dengan PHP.

Kekurangan PHP yaitu:

- a. PHP memiliki kelemahan keamanan.
- b. Kode PHP bisa dibaca oleh semua orang.
- c. Biaya untuk *encoding* membutuhkan biaya yang sangat mahal

## 2. CodeIgniter

CodeIgniter adalah sebuah kerangka kerja untuk *web* yang dibuat dalam format PHP. Format yang dibuat dapat digunakan untuk membuat sistem aplikasi *web* yang kompleks. CodeIgniter dapat mempercepat proses pembuatan *web*, karena *class* dan *module* yang dibutuhkan sudah tersedia dan *programmer* hanya tinggal menggunakannya kembali pada aplikasi *web* yang akan dibuat.

CodeIgniter memiliki banyak fitur yang membuatnya berbeda dengan *framework* lainnya. Tidak seperti beberapa *framework* PHP lainnya, dokumentasi untuk *framework* ini sangat lengkap, yang mencakup seluruh aspek dalam *framework*. CodeIgniter juga mampu berjalan pada lingkungan *Shared Hosting* karena memiliki ukuran yang sangat kecil, namun memiliki kinerja yang sangat luar biasa. Dari segi pemrograman, CodeIgniter kompatibel dengan PHP4 dan PHP5, sehingga akan berjalan dengan baik pada *web host* yang banyak dipakai saat ini. CodeIgniter menggunakan pola desain *Model-View-Controller* (MVC), yang merupakan cara untuk mengatur aplikasi web ke dalam 3 bagian yang berbeda. Pada intinya CodeIgniter juga membuat penggunaan ekstensif dari pola desain Singleton. Maksudnya adalah cara untuk *me-load class* sehingga jika *class* itu di panggil dalam beberapa kali, kejadian yang sama pada *class* tersebut akan digunakan kembali. Hal ini

sangat berguna dalam koneksi *database*, karena kita hanya ingin menggunakan satu koneksi setiap kali *class* itu digunakan.

### 3. Javascript

JavaScript dibuat dan didesain selama sepuluh hari oleh Brandan Eich, seorang karyawan Netscape, pada bulan September 1995. Awalnya bahasa pemrograman ini disebut Mocha, kemudian diganti ke Mona, lalu LiveScript sebelum akhirnya resmi menyandang nama JavaScript. Versi pertama dari bahasa ini hanya terbatas di kalangan Netscape saja. Fungsionalitas yang ditawarkan pun terbatas. Namun, JavaScript terus dikembangkan oleh komunitas developer yang tak henti-hentinya mengerjakan bahasa pemrograman ini.

JavaScript adalah salah satu bahasa pemrograman yang paling banyak digunakan dalam kurun waktu dua puluh tahun ini. Bahkan JavaScript juga dikenal sebagai salah satu dari tiga bahasa pemrograman utama bagi web developer:

- HTML: Memungkinkan Anda untuk menambahkan konten ke halaman web.
- CSS: Menentukan *layout, style*, serta keselarasan halaman *website*.
- JavaScript: Menyempurnakan tampilan dan sistem halaman web.

JavaScript dapat dipelajari dengan cepat dan mudah serta digunakan untuk berbagai tujuan, mulai dari meningkatkan fungsionalitas *website* hingga mengaktifkan permainan (*games*) dan *software* berbasis web. Selain itu,

terdapat ribuan template dan aplikasi JavaScript yang bisa Anda gunakan secara gratis dan semuanya ini berkat beberapa situs, seperti Github.

#### 4. JQuery

jQuery adalah library JavaScript yang populer. Bahasa pemrograman ini dibuat oleh John Resig, tepatnya pada tahun 2006, untuk memudahkan para *developer* dalam menggunakan dan menerapkan JavaScript di *website*. jQuery bukanlah bahasa pemrograman yang berdiri sendiri, melainkan bekerja sama dengan JavaScript. Dengan menggunakan jQuery, Anda bisa melakukan banyak hal.

Seperti yang kita ketahui, menulis kode bukanlah pekerjaan yang mudah dan terkadang menyulitkan, terlebih lagi kalau ada banyak string kode yang harus ditambahkan dan diaktifkan. Di sinilah jQuery memainkan perannya. Fungsi jQuery adalah meng-*compress* berbagai baris atau line kode ke dalam satu buah fungsi sehingga Anda tidak perlu menulis kembali semua baris kode hanya untuk menyelesaikan satu task. Salah satu alasan mengapa jQuery sangat populer dan banyak digunakan adalah kemampuan lintas platformnya. Secara otomatis, jQuery akan memperbaiki error serta punya fungsi yang sama seperti ketika dijalankan di browser, seperti Chrome, Firefox, Safari, MS-Edge, IE, Android, dan iOS.

Adanya jQuery juga memudahkan penggunaan Ajax. Ajax menganut sistem kerja yang asinkron, dan umumnya dimulai dari kode yang tersisa. Hal ini berarti kode yang ditulis dengan Ajax dapat berkomunikasi dengan server dan memperbarui kontennya tanpa harus me-*load* kembali halaman terlebih dulu. Sayangnya, tidak selamanya penggunaan Ajax lancar dan mulus-mulus saja. Setiap browser mengaktifkan Ajax Api dengan cara yang berbeda-beda.

Karena itulah, sebisa mungkin kode harus diatur agar bisa dijalankan di semua jenis browser.

## 5. Ajax

AJAX adalah sebuah singkatan dari *Asynchronous Javascript and XML* dan mengacu pada sekumpulan teknis pengembangan web (*web development*) yang memungkinkan aplikasi web untuk bekerja secara *asynchronous* (tidak langsung) – memproses setiap request (permintaan) yang datang ke *server* di sisi background.

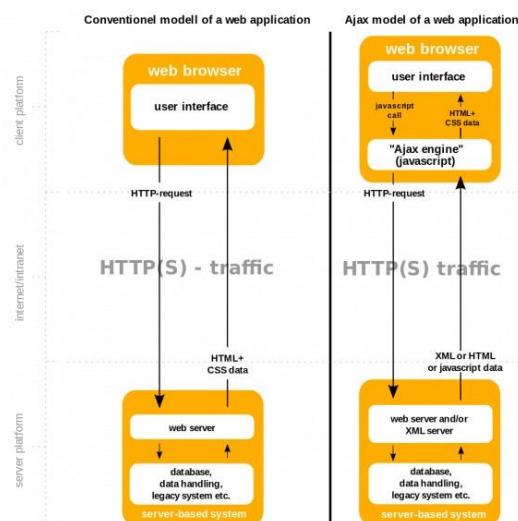
Layaknya HTML, XML atau *eXtensible Markup Language* adalah varian lain dari bahasa markup. Jika HTML dirancang untuk menampilkan data, maka XML dirancang untuk memuat dan membawa data. Baik JavaScript maupun XML bekerja secara *asynchronous* di dalam AJAX. Alhasil, aplikasi web yang menggunakan AJAX dapat mengirimkan dan menerima data dari *server* tanpa harus mereload keseluruhan halaman. Berikut adalah beberapa contoh dari penggunaan AJAX:

- Sistem *Voting* atau *Rating*
- *Chat Room*
- Notifikasi *Trending* di Twitter

AJAX bukanlah teknologi dan bukan pula bahasa pemrograman. Seperti yang telah dijelaskan sebelumnya, AJAX adalah sekumpulan teknik pengembangan web. Pada umumnya sistem ini terdiri atas:

- HTML/XHTML sebagai bahasa utama dan CSS untuk menampilkan data.
- *The Document Object Model (DOM)* untuk menampilkan data yang dinamis beserta interaksinya.
- XML untuk pertukaran data, sedangkan XSLT untuk manipulasi data. Sebagian besar *developer* mulai mengganti XML dengan JSON karena bentuknya yang mendekati JavaScript.
- Objek *XMLHttpRequest* untuk komunikasi tidak langsung (asynchronous).
- Bahasa pemrograman JavaScript untuk menyatukan semua teknologi ini.

Untuk memahami cara kerja AJAX secara keseluruhan, setidaknya Anda harus punya pemahaman teknis dasar terlebih dulu. Untungnya, prosedur umum dari cara kerja AJAX tidak begitu sulit. Lihat diagram dan tabel di bawah ini untuk perbandingannya.



Gambar 1 Diagram Alur AJAX

## 6. MySQL

MySQL adalah sebuah *database management system* (manajemen basis data) menggunakan perintah dasar SQL (*Structured Query Language*) yang cukup terkenal. *Database management system* (DBMS) MySQL multi pengguna dan multi alur ini sudah dipakai lebih dari 6 juta pengguna di seluruh dunia.



Gambar 2 Logo MySQL

MySQL adalah DBMS yang *open source* dengan dua bentuk lisensi, yaitu *Free Software* (perangkat lunak bebas) dan *Shareware* (perangkat lunak berpemilik yang penggunaannya terbatas). Jadi MySQL adalah *database server* yang gratis dengan lisensi GNU General Public License (GPL) sehingga dapat Anda pakai untuk keperluan pribadi atau komersil tanpa harus membayar lisensi yang ada.

Seperti yang sudah disinggung di atas, MySQL masuk ke dalam jenis RDBMS (Relational Database Management System). Maka dari itu, istilah semacam baris, kolom, tabel, dipakai pada MySQL. Contohnya di dalam MySQL sebuah database terdapat satu atau beberapa tabel. SQL sendiri merupakan suatu bahasa yang dipakai di dalam pengambilan data pada relational database atau database yang terstruktur. Jadi MySQL adalah database management system yang menggunakan bahasa SQL sebagai bahasa penghubung antara perangkat lunak aplikasi dengan database server.

Penggunaan dari MySQL sendiri tentu memiliki kelebihan dan kekurangannya, berikut adalah kelebihan dari MySQL:

- Mendukung integrasi dengan bahasa pemrograman lain
- Tidak membutuhkan RAM besar
- Mendukung *multi user*
- Bersifat *opensource*
- Struktur tabel yang fleksibel
- Tipe data yang bervariasi

Selain itu kekurangan dari penggunaan MySQL sebagai berikut:

- Kurang cocok untuk aplikasi *mobile* dan *game*
- Sulit mengelola *database* yang besar
- *Technical support* yang kurang bagus

# **BAB 2**

---

## B. Metode Penyelesaian Aplikasi

### 1. Definisi Scrum

*Scrum* yang masih merupakan bagian dari metodologi *agile*. *Scrum* merupakan sebuah proses pengembangan perangkat lunak dengan konsep *agile* yang dikembangkan oleh Jeff Sutherland dan Ken Schwaber pada tahun 1996. Model *scrum* biasanya digunakan karena menerapkan proses dengan siklus pendek berulang, secara aktif melibatkan pengguna untuk membangun, memprioritaskan, dan memverifikasi kebutuhan.

Kelebihan *scrum* adalah dapat menghasilkan produk sesuai dengan keinginan pengguna dan cocok untuk pengembangan sistem dengan skala kecil serta banyak perubahannya. Di sisi lain, *scrum* merupakan metodologi yang paling sering digunakan oleh banyak perusahaan besar karena merupakan salah satu metodologi yang menggunakan konsep *agile* dengan keunggulannya yaitu lebih cepat dan lebih efektif dalam pengembangan perangkat lunak.

Berikut merupakan beberapa aktifitas yang dilakukan dalam penerapan metodologi *scrum*:

1. *Backlog* atau yang lebih dikenal *product backlog* adalah daftar kebutuhan atau fitur yang memberikan nilai bisnis kepada klien. *Product backlog* dapat bertambah seiring dengan kebutuhan penggunanya.
2. *Scrum* membagi sebuah *project* ke dalam sebuah perulangan yang disebut sebagai *sprint*. Sebuah *sprint* biasanya memiliki durasi

pengerjaan hingga 4 minggu, dimana tim pengembang fokus dalam mencapai target yang jelas dan telah ditentukan.

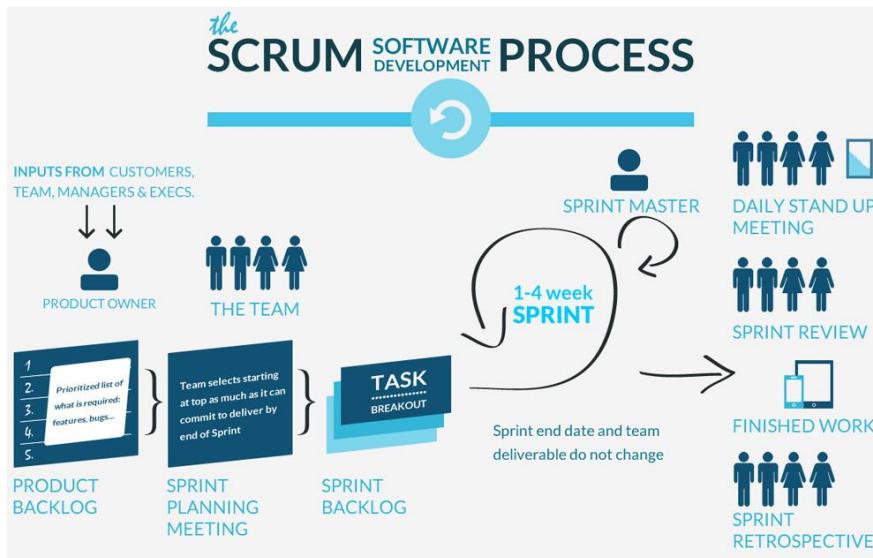
3. *Scrum meetings*, merupakan pertemuan rutin yang dilakukan perhari atau perminggu untuk mengevaluasi apa yang dikerjakan, hambatan, dan target penyelesaiannya. Dalam hal ini, penulis melakukan *scrum meetings* setiap minggu dengan membahas apa saja yang dikerjakan setiap minggunya serta hambatan apa yang dialami selama proses pengerjaan itu.
4. *Demo*, memberikan perubahan atau *update* kepada klien yang telah diimplementasikan agar dapat dilihat dan dievaluasi oleh klien.

Selain aktifitas yang telah disebutkan, ada beberapa aktor yang terlibat dalam pengembangan sistem menggunakan model *scrum*, yaitu:

1. *Product owner*, adalah orang yang bertanggung jawab atas keseluruhan sistem atau produk yang sedang dikembangkan
2. *Master scrum*, adalah orang yang mengeliminasi segala hambatan yang muncul dalam pengembangan proyeknya
3. *Team scrum*, adalah sekumpulan orang yang dapat terdiri dari *programmers*, *testers*, dan para ahli lainnya yang bertugas untuk mengembangkan produk dan memberikan kepuasan terhadap klien melalui produk yang dihasilkan.

## 2. Tahapan Metodologi Scrum

Untuk alur yang lebih jelas dari *Scrum*, maka lihatlah gambar 3 di bawah ini:



Gambar 3 Alur Scrum

Dengan diagram alur yang telah ditunjukkan pada gambar 3, alur tersebut dapat dibagi menjadi 3 tahapan utama yaitu tahap *pregame*, *game*, dan *post game*.

Berikut adalah tahapan – tahapan yang harus dilakukan:

1. Tahap *pregame* berfokus pada melakukan pengidentifikasi permasalahan yang terjadi, melakukan pengumpulan data yang relevan terkait dengan permasalahan yang terjadi serta melakukan perencanaan untuk membuat solusi dari masalah yang sedang dihadapi. Tahap *pregame* berisi aktifitas *scrum* seperti *backlog/product backlog*.

2. Tahap *game* dilakukan perancangan sesuai dengan perencanaan yang telah dibuat pada tahapan sebelumnya, biasanya pada tahapan ini berisi aktifitas *sprint* dan juga *scrum meetings*.
3. Sedangkan pada tahap *post game* dilakukan pengujian akhir guna melihat apakah sistem yang dikembangkan sudah memenuhi kebutuhan pengguna. Di dalam tahapan yang terakhir berisi aktifitas seperti *demo*.

Biasanya, untuk setiap *project* akan berbeda – beda *product backlog* dan *sprint* yang dilakukan tergantung bagaimana kompleksitas *project* itu atau faktor lainnya. Lalu untuk melakukan *scrum meetings* yang membahas kendala dan juga *milestone* dari *project* minimal satu minggu sekali yang seringkali harus menyesuaikan dengan jadwal dari *product owner*.

Untuk itu, sebelum dimulainya pengerjaan proyek perlu dilakukan pendefinisian yang jelas siapa saja *actor* yang terlibat dan juga *product backlog* apa saja yang akan dibuat. Selain itu *sprint* dari setiap *product backlog* juga perlu didefinisikan. Maka berikut adalah tabel yang mendefinisikan keseluruhan *actor*, *activity*, *product backlog*, dan juga *sprint* dari pengerjaan proyek.

Tabel 1 Tabel *Actor* dan *Activity Scrum*

<i>Activity and Actor</i>	<i>General</i>	
<i>Team Scrum</i>	M. Fauzan Aristyo	
	Muhamad Wahyunda	
	Dimas Haryoputra Maheswara	
	Riandaka Rizal	
<i>Scrum Master</i>	Riandaka Rizal	
<i>Product Owner</i>	M. Fauzan Aristyo	
	<i>Product Backlog</i>	<i>Sprint</i> (estimasi)
<i>Backlog</i>	<i>Create, update, delete, view</i> data peserta EPPM	1 minggu
	<i>Create, update, delete, view</i> data <i>workbook</i>	1 minggu
	<i>Create, update, delete, view</i> data <i>assessment</i> peserta EPPM	1 minggu
	Menu raport	3 hari
	Visualisasi data progres	4 hari
<b>Total Estimasi Pengerjaan</b>		4 minggu

# **BAB 3**

---

## C. Tutorial Pembuatan Aplikasi EPPM Go!

### 1. Inisiasi penggerjaan aplikasi

Untuk pembuatan aplikasi EPPM Go! ini akan dibagi menjadi 5 menu untuk admin dan juga 2 menu untuk *user*, menu yang ditujukan untuk admin adalah sebagai berikut:

- Menu Daftar Pekerja
- Menu *Workbook*
- Menu *Assessment*
- Menu Ruangkerja
- Menu Raport

Lalu untuk menu yang ditujukan untuk *user* adalah sebagai berikut:

- Menu *Workbook*
- Menu *Raport*

Untuk membuat aplikasi EPPM Go! Diperlukanlah sebuah logo sehingga aplikasi itu dapat dikenali oleh khalayak ramai, untuk logo harus dibuat dengan menarik dan atraktif yang dapat menarik perhatian dan minat orang untuk menggunakannya.



Gambar 4 Logo EPPM Go!

Ada tahapan-tahapan yang harus dilalui mulai dari inisiasi sampai dengan *testing*. Agar lebih terstruktur maka harus didefinisikan terlebih dahulu tahapannya, yaitu:

1. Download CodeIgniter berupa *zip file*
2. Inisiasi CodeIgniter dengan *unzip file*
3. Membuat *login*
4. Buat CRUD (*Create, Read, Update, Delete*) menu daftar pekerja.
5. Buat CRUD (*Create, Read, Update, Delete*) menu *workbook* baik untuk admin dan *user*.
6. Bagi fungsi yang bisa diakses oleh admin, dan batasi fungsi yang hanya bisa diakses oleh *user*
7. Buat CRUD (*Create, Read, Update, Delete*) menu *assessment* untuk admin.
8. Di dalam menu *assessment*, buat pembobotan nilai untuk penilaian yang akan dilakukan. 40% untuk nilai teori dan 60% untuk nilai praktik
9. Buat CRUD (*Create, Read, Update, Delete*) menu ruangkerja untuk admin.
10. Di dalam menu ruangkerja, buat fungsi *import* untuk mengimport *file excel* atau *csv* agar dapat efisien merubah data di dalam menu ruangkerja

11. Tampilkan hasil penilaian *assessment* di menu raport tanpa filter untuk admin dan filter sesuai dengan akun *user* jika digunakan oleh *user*.
12. Tambahkan fungsi *export* agar dapat diunduh di komputer masing masing.
13. Lalu pada halaman utama (*dashboard*) dari admin, tambahkan grafik untuk dapat memvisualisasikan data dari menu *workbook*.
14. Untuk grafik, gunakan ChartJS API agar dapat menggunakan grafik yang dinamis.

Pada halaman *user* atau pengguna, diperlukan sebuah gambar yang menarik agar mendapat kesan pertama dari pengguna itu sendiri berikut adalah gambar untuk halaman awal pengguna



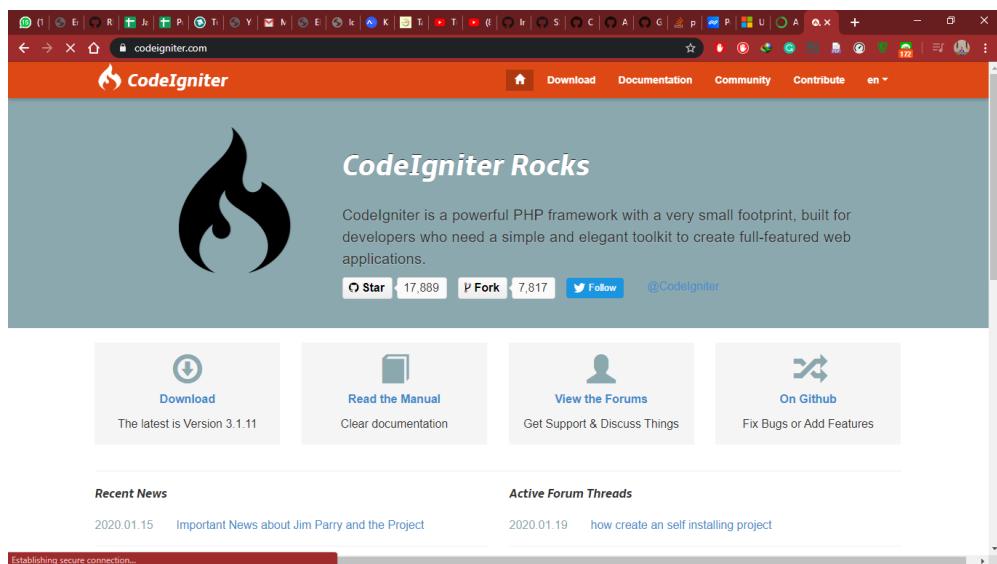
*Gambar 5 Halaman Awal Pengguna*

Dapat dilihat pada gambar 5, ada beberapa komponen dari gambar yang disesuaikan dengan kebiasaan dari pengguna sendiri. Hal ini merupakan suatu usaha untuk mendapatkan kesan pertama pengguna sendiri.

## 2. Pembuatan Aplikasi

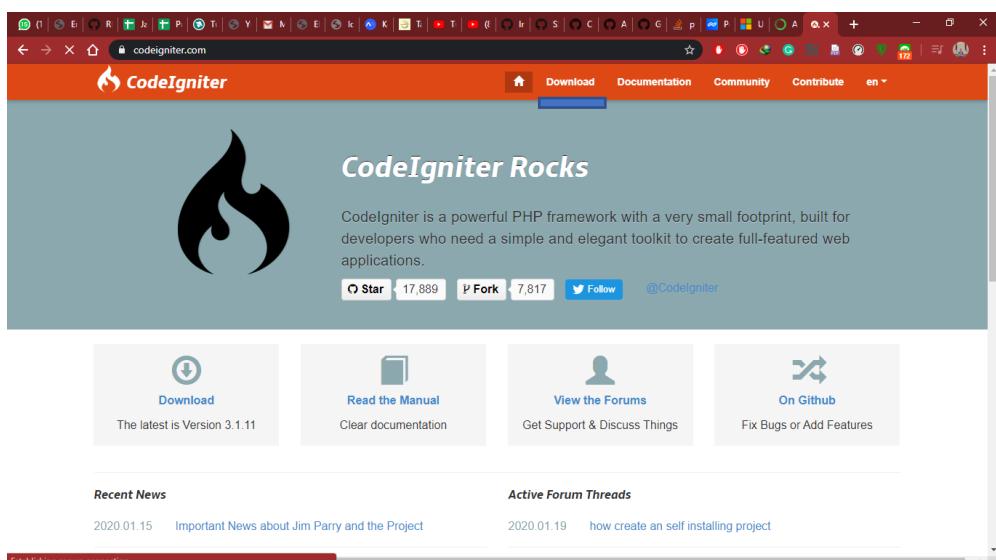
### 2.1. Inisiasi CodeIgniter

Untuk pembuatan aplikasi, maka harus disesuaikan dengan langkah yang telah diinisiasi sebelumnya. Maka hal pertama yang harus dilakukan adalah *download* CodeIgniter melalui *website* resminya yaitu [www.codeigniter.com](http://www.codeigniter.com)



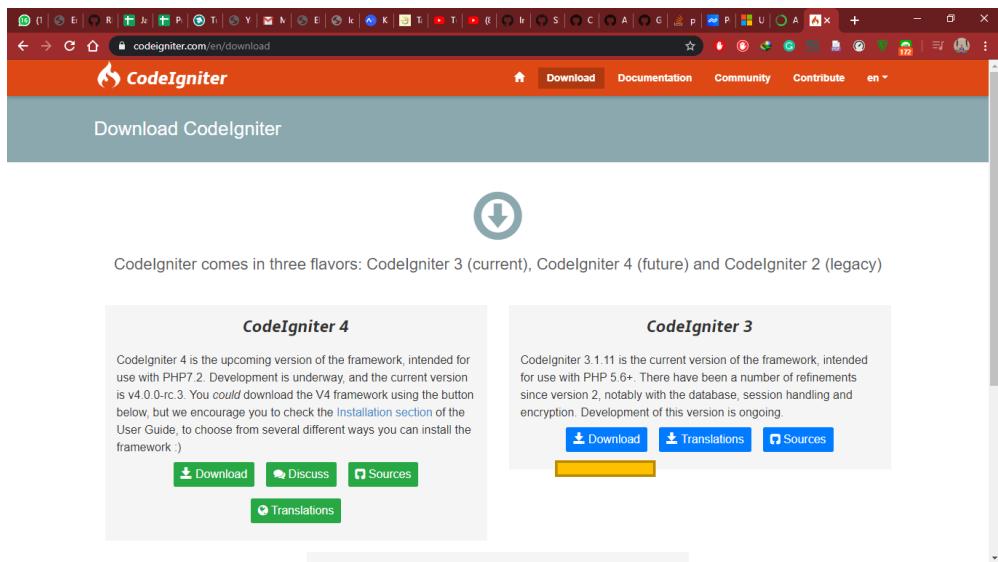
Gambar 6 Website Resmi CodeIgniter

Setelah membuka *website* CodeIgniter pilih menu *download* yang ada di atas menu bar.



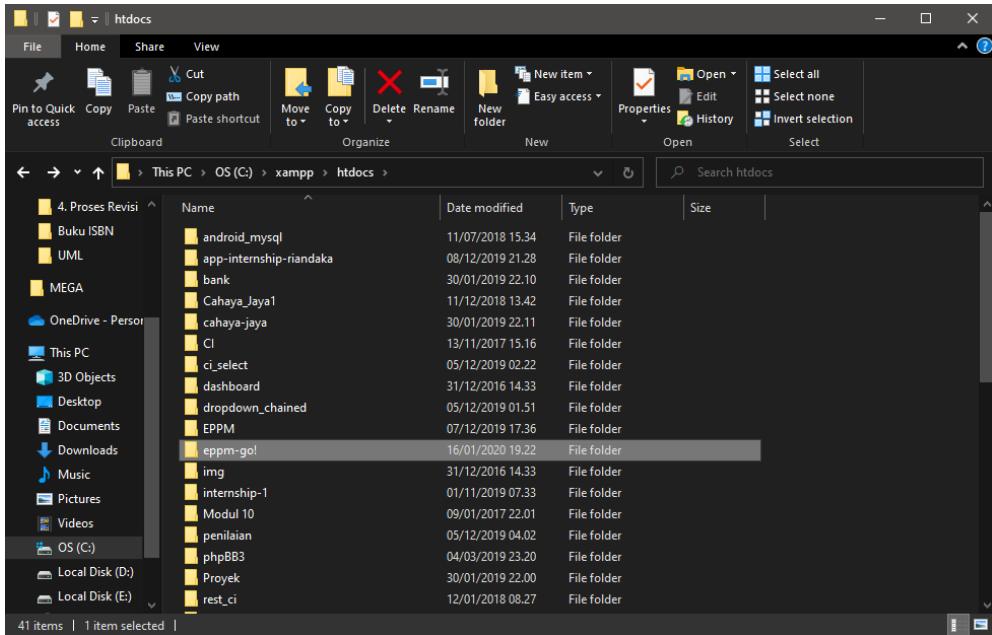
Gambar 7 Pilih Menu Download

Setelah berada di halaman *download* pilih python 3 untuk diunduh, karena saat ini python 3 adalah versi yang paling banyak digunakan oleh pengembang di dunia.



Gambar 8 CodeIgniter 3

Tunggulah beberapa saat untuk selesai mengunduh *file zip* dari CodeIgniter. Setelah itu *unzip file* CodeIgniter yang baru saja kita unduh ke dalam folder C://xampp/htdocs dan *rename* folder CodeIgniter menjadi eppm-go! Seperti yang ditunjukkan pada gambar 9.



Gambar 9 Folder eppm-go!

Setelah selesai membuat folder eppm-go! silakan lanjut ke langkah selanjutnya, yaitu membuat CRUD (*Create, Read, Update, Delete*) untuk menu daftar pekerja.

## 2.2. Login

Buat file dengan nama Login.php pada controller, nama file untuk controller selalu disarankan untuk berawalan huruf besar. Lalu buka file tersebut dan ketikkan seperti yang ditunjukkan pada gambar 10

```
<?php  
defined('BASEPATH') OR exit('No direct script access allowed');  
  
class Login extends CI_Controller {  
  
    public function __construct()  
    {  
        parent::__construct();  
        //Load library form validasi  
        $this->load->library('form_validation');  
        //Load model admin  
        $this->load->model('admin');  
    }  
}
```

Gambar 10 Controller Login.php

- Class Login extends CI\_Controller = maksudnya adalah membuat *class Login* dengan memakai *library* CI\_Controller
- Public function \_\_construct() = maksudnya adalah menggunakan function construct
- \$this->load->library('form\_validation') = maksudnya adalah *load library* form validasi
- \$this->load->mode('admin') = maksudnya adalah *load* model admin

Setelah itu kita lanjutkan kembali dengan mengetikkan sesuai dengan yang ditunjukkan oleh gambar 11

```

15 |     public function index()
16 |     {
17 |
18 |         if($this->admin->is_logged_in())
19 |         {
20 |             // jika memang session sudah terdaftar, maka redirect ke halaman dashboard
21 |             redirect("admin/dashboard");
22 |
23 |         // }elseif($this->user->is_logged_in()){
24 |
25 |             // redirect("user/dashboard");
26 |
27 |         }else{
28 |
29 |             //jika session belum terdaftar
30 |
31 |             //set form validation
32 |             $this->form_validation->set_rules('username', 'Username', 'required');
33 |             $this->form_validation->set_rules('password', 'Password', 'required');
34 |
35 |             //set message form validation
36 |             $this->form_validation->set_message('required', '<div class="alert alert-danger" style="margin-top: 3px">
37 |                 <div class="header"><b><i class="fa fa-exclamation-circle"></i> {field}</b> harus diisi</div></div>');
38 |
39 |             //cek validasi
40 |             if ($this->form_validation->run() == TRUE) {
41 |
42 |                 //get data dari FORM
43 |                 $username = $this->input->post("username", TRUE);
44 |                 $password = $this->input->post("password", TRUE);
45 |
46 |                 //checking data via model
47 |                 $checking = $this->admin->check_login('account', array('username' => $username), array('password' => $passw

```

*Gambar 11 Controller Login.php*

Seperti yang dilihat pada gambar 11, untuk function index akan mengecek terlebih dahulu apakah sudah ada *session* yang dibuat apa belum, jika sudah maka akan langsung *redirect* ke halaman *dashboard* admin. Jika belum maka akan membuat form validasi dengan *username* dan *password* yang harus diisi lalu akan dicek apakah data *username* dan *password* yang kita masukkan sudah ada atau belum,

```

19     //jika ditemukan, maka create session
20     if ($checking != FALSE) {
21         foreach ($checking as $apps) {
22
23             $session_data = array(
24                 'employee_id' => $apps->employee_id,
25                 'user_name'   => $apps->username,
26                 'user_pass'   => $apps->password,
27                 'user_email'  => $apps->user_email,
28                 'user_nama'   => $apps->employee_name,
29                 'directorate' => $apps->directorate,
30                 'role'        => $apps->role
31             );
32             //set session userdata
33             // print_r($session_data); die;
34             $this->session->set_userdata($session_data);
35
36             //redirect berdasarkan Level user
37             if($this->session->userdata("role") == "admin"){
38                 redirect('admin/dashboard');
39             }else{
40                 redirect('user/dashboard');
41             }
42         }
43     }else{
44
45         $error = 'Username atau Password Anda salah. Silakan ulangi Kembali';
46         '<div class="alert alert-danger" style="margin-top: 3px">';
47         '<div class="header"><b><i class="fa fa-exclamation-circle"></i> ERROR</b> username atau password salah!';
48         $this->load->view('login', $error);
49     }
50 }
```

Gambar 12 Controller Login.php

Jika sudah ada maka akan langsung dibuat *session*-nya dan akan *redirect* ke halaman awal sesuai dengan hak aksesnya masing masing. Jika itu admin maka akan diarahkan ke halaman *dashboard* admin, sedangkan jika *user* maka akan diarahkan ke halaman awal untuk pengguna.

```

92     public function logout()
93     {
94         $this->session->sess_destroy();
95         redirect('index.php/login');
96     }
97 }
```

Gambar 13 Controller Login.php

Lalu setelah itu, kita membuat function logout() yang berguna untuk keluar dari aplikasi dengan cara menghapus atau menghancurkan *session* yang sedang berjalan dan mengarahkannya ke halaman login. Lanjut lagi buat kembali *file* dengan nama Admin.php pada folder model

```
1 <?php
2 defined('BASEPATH') OR exit('No direct script access allowed');
3
4 class Admin extends CI_Model
5 {
6     //fungsi cek session Logged in
7     function is_logged_in()
8     {
9         return $this->session->userdata('id');
10    }
11 }
```

Gambar 14 Models Admin.php

Seperti yang dapat dilihat pada gambar 14, ada function is\_logged\_in() yang berguna untuk mengecek apakah *session* telah ada dan dibuat. Lalu hasil yang dikeluarkan dari function ini berupa userdata dengan membawa id dari user yang sedang *login*. Lalu dilanjutkan dengan function is\_role() yang ditunjukkan oleh gambar 15, dimana akan mengecek *role* dari setiap pengguna yang masuk sehingga akan diberikan hak akses sesuai dengan level atau *role*-nya.

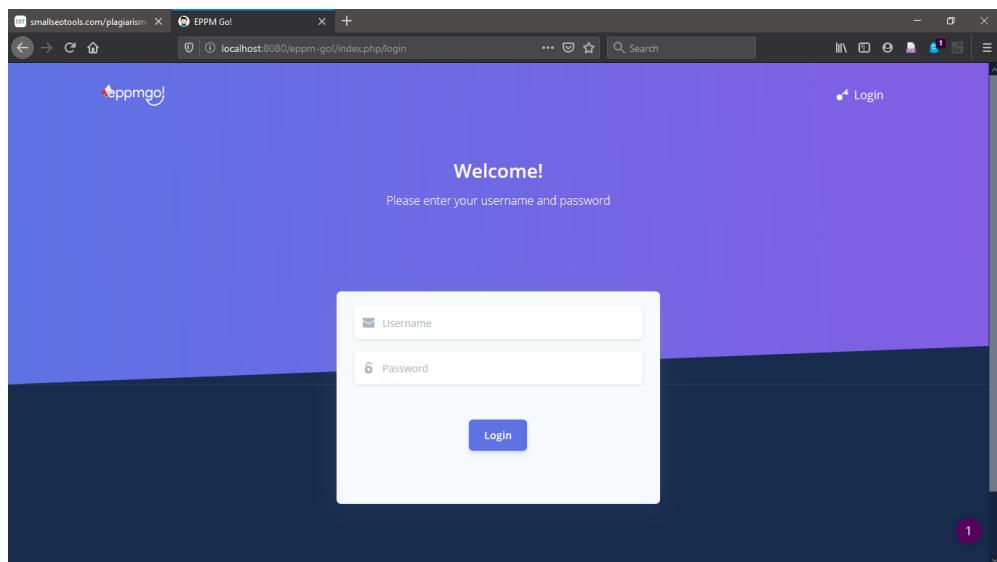
```
//fungsi cek level
function is_role()
{
    return $this->session->userdata('role');
```

Gambar 15 Models Admin.php

Lalu untuk *view*-nya silakan berkreasi sendiri menggunakan bootstrap pilihan anda sendiri. Disini saya menggunakan bootstrap dari argon dashboard. Pastikan pada tombol login sudah diisikan form methodnya seperti yang ditunjukkan pada gambar 16 berikut.

```
<form method="POST" action="=base_url();?&gt;index.php/login"&gt;</pre
```

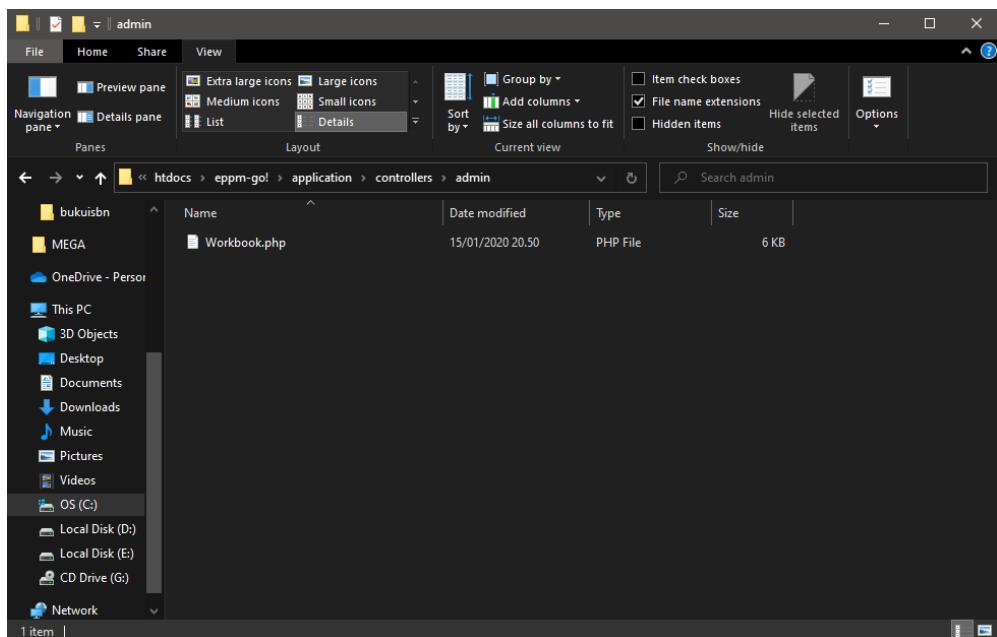
Gambar 16 Form Method Login



Gambar 17 Halaman Login

### 2.3. CRUD Menu Workbook

Untuk membuat CRUD menu daftar pekerja, mulai sekarang akan kita akan bekerja menggunakan HTML, PHP, dan juga *database*-nya. Untuk *database*-nya akan dijelaskan di lain bab. Pertama buatlah *file* bernama Workbook.php dan folder dengan nama admin, ini ditujukan untuk membedakan *file* untuk admin dan juga untuk *user*. Lalu tempatkan pada folder controller pada direktori eppm-go! Untuk penamaan *file* controller disarankan untuk selalu berawalan huruf kapital, agar penamaan *file* menjadi lebih rapi dan dapat dibedakan secara visual. Tetapi apabila lebih suka dibuat dengan berawalan huruf kecil maka sesuaikan saja dengan keinginan masing masing.



Gambar 18 Pembuatan Workbook.php

Setelah membuat file Workbook.php, maka selanjutnya yang akan kita lakukan adalah mulai untuk ngoding. Pada Workbook.php ketik kode berikut sesuai dengan yang ditunjukkan oleh gambar 19.

```
1 <?php
2 defined('BASEPATH') OR exit ('No direct script access allowed');
3
4 class Workbook extends CI_Controller
5 {
6     public function __construct()
7     {
8         parent::__construct();
9         $this->load->model("workbook_model"); //Load model workbook
10        $this->load->library('form_validation'); //Load library form validation
11        $this->load->helper(array('form', 'url'));
12        if($this->workbook_model->is_role() != "admin"){
13            redirect("index.php/login");
14        }
15    }
}
```

Gambar 19 Controller Workbook.php

Disini diawali dengan dengan me-*load* dari model workbook yang akan kita bahas selanjutnya, dan juga me-*load library* form validasi. Selain itu, pada controller Workbook.php akan terlebih dahulu mengecek *role* yang akan mengakses halaman *workbook* awal yang function-nya diambil dari model workbook, jika *role* yang masuk adalah admin maka akan langsung menuju dan mengakses halaman *workbook*. Sedangkan bila sebaliknya, maka akan langsung menuju halaman *login*.

Selanjutnya adalah membuat function index, yang di dalamnya terdapat sintaks yang berfungsi untuk me-*load* tampilan halaman *workbook* awal dengan mengambil data yang diinisiasi melalui method getAll() dari model *workbook\_model* seperti yang ditunjukkan oleh gambar 20.

```
17 |     public function index()
18 |     {
19 |         $data["workbook"] = $this->workbook_model->getAll(); //ambil data dari model
20 |         $this->load->view("admin/workbook/list_workbook", $data); //load view data model ke workbook
21 |     }
22 | }
```

Gambar 20 Controller Workbook.php

Kemudian, lanjut dengan membuat function add yang berfungsi untuk me-*load* tampilan halaman form input progres *workbook* dengan mengambil data *stages* dari modul pelatihan EPPM dari model *workbook\_model* seperti yang ditunjukkan oleh gambar 21.

```
23 |     public function add()
24 |     {
25 |         $data["workbook"] = $this->workbook_model->get_data_stage();
26 |         $this->load->view("admin/workbook/new_form", $data); //Load isi form workbook
27 |     }
28 | }
```

Gambar 21 Controller Workbook.php

Lalu dilanjutkan dengan membuat function aksi\_add yang berfungsi untuk menyimpan data yang telah kita *input*-kan pada form *workbook* dengan menggunakan model *workbook\_model* dan memberikan pesan apabila data telah sukses disimpan ke dalam *database* dan akan langsung mengarahkan ulang ke halaman *workbook* seperti yang ditunjukkan oleh gambar 22.

```
public function aksi_add()
{
    // print_r('tes'); die;
    $this->workbook_model->save(); //objek model redirect
    $this->session->set_flashdata('Sukses', 'Data Anda Berhasil Disimpan'); //pesan berhasil
    redirect('index.php/admin/workbook');
}
```

Gambar 22 Controller Workbook.php

Jika sudah selesai membuat function aksi\_add, maka selanjutnya adalah membuat function baru di dalam file *Workbook.php* dengan nama function edit yang berguna untuk menampilkan halaman edit *workbook* yang data form *input* *workbook* diambil dari *database* dan dilempar melalui model *workbook\_model* berdasarkan id yang dipilih seperti yang ditunjukkan oleh gambar 23 berikut.

```
public function edit($id=null)
{
    $data["workbook"] = $this->workbook_model->getById($id); //mengambil data berdasarkan id
    $data["status"] = ['Not Yet Started', 'On Progress', 'Acc. Done']; //mengambil data berdasarkan id
    if (!$data["workbook"]) show_404(); // jika tidak ada show error
    $this->load->view("admin/workbook/edit_form", $data); //Load edit form workbook
}
```

Gambar 23 Controller Workbook.php

Selanjutnya sama seperti function add, kita harus membuat function aksi\_edit untuk melakukan proses edit pada aplikasi. Function aksi\_edit berfungsi untuk memberikan perubahan pada form yang telah kita simpan sebelumnya. Function aksi\_edit dimulai dengan menginisiasikan *config* atau pengaturan tentang folder mana yang menjadi tempat penyimpanan gambar dan mana saja ekstensi untuk *file* gambar yang didukung. Lalu setelah itu akan me-load *library upload* dengan menggunakan *config* yang telah diinisiasi sebelumnya. Jika tidak ada gambar yang diubah atau di-upload maka hanya ubah data yang dimasukkan nilai yang baru dan bila sukses akan langsung diarahkan kembali menuju halaman *workbook* awal.

```
69 public function aksi_edit()
70 {
71     $config['upload_path'] = './upload/workbook'; //path folder
72     $config['allowed_types'] = 'gif|jpg|png|jpeg|bmp'; //type yang dapat diakses bisa anda sesuaikan
73
74
75     $this->load->library('upload', $config);
76     if ( ! $this->upload->do_upload('image')){
77         $id = $this->input->post('workbook_id');
78         $data = array (
79             'workbook_id' => $this->input->post('workbook_id'),
80             'stage_id' => $this->input->post('stage_id'),
81             'modul_id' => $this->input->post('modul_id'),
82             'employee_name' => $this->input->post('employee_name'),
83             'progress' => $this->input->post('progress'),
84             'status' => $this->input->post('status'),
85             'image' => $this->input->post('old_image')
86         );
87
88         // print_r($data); die;
89         $this->workbook_model->update($id,$data);
90         $this->session->set_flashdata('success', 'Data Anda Berhasil Diupdate');
91         redirect('index.php/admin/workbook');
92     }else{
93
94
95
96
97
98
99
100 }
```

#### Gambar 24 Controller Workbook.php

tetapi bila ada gambar yang diubah atau gambar yang baru saja di-*upload* maka masukkan seluruh data yang baru saja diubah ke dalam *database* dan bila sukses akan langsung diarahkan kembali ke halaman awal *workbook*

```
else{
    $id = $this->input->post('workbook_id');
    $image = $this->upload->data();
    $data = array (
        'workbook_id' => $this->input->post('workbook_id'),
        'stage_id' => $this->input->post('stage_id'),
        'modul_id' => $this->input->post('modul_id'),
        'employee_name' => $this->input->post('employee_name'),
        'progress' => $this->input->post('progress'),
        'status' => $this->input->post('status'),
        'image' => $image['file_name']
    );
    // print_r($data); die;
    $this->workbook_model->update($id,$data);
    $this->session->set_flashdata('success', 'Data Anda Berhasil Diupdate');
    redirect('index.php/admin/workbook');
}
```

Gambar 25 Controller Workbook.php

Setelah membuat fungsi untuk *add* dan *edit*, maka kita juga harus membuat fungsi untuk menghapus data yang salah. Oleh karena itu, kita buat function *delete* yang berfungsi untuk menghapus data yang kita pilih atau sesuai dengan id-nya. Jika sudah maka akan diarahkan kembali ke halaman awal *workbook*.

```
public function delete($id=null)
{
    if (!isset($id)) show_404();

    if ($this->workbook_model->delete($id)){
        redirect('index.php/admin/workbook');
    }
}
```

Gambar 26 Controller Workbook.php

Setelah selesai membuat Workbook.php untuk admin, selanjutnya adalah kita harus membuat modelsnya yang berfungsi untuk melakukan manipulasi data yang kita inginkan. Silakan membuat *file* dengan nama Workbook\_model.php di dalam folder models. Lanjut dengan mengetikkan *line* berikut sesuai dengan gambar 27.

```
1 <?php defined('BASEPATH') OR exit('No direct script access allowed');
2
3 class Workbook_model extends CI_Model
4 {
5     // private $_table = "workbook";
6
7     public $workbook_id;
8     public $stage_id;
9     public $modul_id;
10    public $employee_id;
11    public $employee_name;
12    public $status;
13    public $image = "default.jpg";
14 }
```

Gambar 27 Models Workbook\_model.php

Pada *line* di atas, kita terlebih dahulu menginisiasikan variabel variabel yang akan digunakan, dengan membuat *class* Workbook\_model dengan menggunakan *library* CI\_Model. Jika sudah selesai, kita harus membuat aturan atau *rules* mana saja *field* yang harus diisi seperti yang ditunjukkan oleh gambar 28 berikut ini.

```
15     public function rules()
16     {
17         return [
18             ['field' => 'stage_id',
19              'rules' => 'required'],
20
21             ['field' => 'modul_id',
22              'rules' => 'required'],
23
24             ['field' => 'employee_id',
25              'rules' => 'required'],
26
27             ['field' => 'employee_name',
28              'rules' => 'required'],
29
30             ['field' => 'progress',
31              'rules' => 'required'],
32
33             ['field' => 'status',
34              'rules' => 'required'],
35
36             ['field' => 'image',
37              'rules' => 'required'],
38
39         ];
40     }
41
```

Gambar 28 Models Workbook\_model.php

Lalu kita harus diatur pula hak akses, sehingga setiap pengguna hanya dapat menu yang sesuai dengan hak akses yang diberikan.

```
42     //fungsi cek level
43     function is_role()
44     {
45         return $this->session->userdata('role');
46     }
```

Gambar 29 Models Workbook\_model.php

Jika sudah selesai mengatur hak aksesnya, maka langsung kita ambil datanya dari *database* di tabel *workbook* dengan membuat function *getAll()*. Lalu agar menampilkan data pada halaman *workbook* menjadi lengkap maka kita melakukan teknik *join* yang menggabungkan salah satu *field* di tabel *parent* ke dalam tabel *children*-nya sesuai dengan foreign key-nya.

```
48     public function getAll()
49     {
50         $this->db->select('workbook.*', 'account.employee_name as yaww');
51         $this->db->join('account', 'workbook.employee_id = account.employee_id');
52         $this->db->from('workbook');
53         return $this->db->get()->result();
54     }
```

Gambar 30 Models Workbook\_model.php

Sama seperti function *getAll()*, kita membuat function *getById()* yang berfungsi untuk mengambil data berdasarkan id-nya pada tabel *workbook* di *database*. Dan kita juga melakukan teknik *join* yang menggabungkan field di tabel *parent* ke dalam tabel *children*-nya sesuai dengan foreign key-nya.

```
56     public function getById($id)
57     {
58         $this->db->select('workbook.*', 'account.employee_name as yaww');
59         $this->db->join('account', 'workbook.employee_id = account.employee_id');
60         $this->db->from('workbook');
61         $this->db->where('workbook_id', $id);
62         return $this->db->get()->row();
63     }
```

Gambar 31 Models Workbook\_model.php

Lanjut lagi, kita buat function `get_data_stage()` yang berfungsi untuk mengambil data *stages* pada tabel *workbook* di kolom *stage*. Setelahnya kita buat function `save()` yang berfungsi untuk menyimpan *input*-an pada form dan memasukkan data tersebut ke dalam tabel *workbook* di dalam *database* seperti yang ditunjukkan pada gambar 32.

```
65 |     public function get_data_stage()
66 |     {
67 |         $query = $this->db->get('stage');
68 |         return $query;
69 |     }
70 |
71 |     public function save()
72 |     {
73 |         // $post = $this->input->post();
74 |         // print_r($post); die;
75 |         // $this->workbook_id = $post["workbook_id"];
76 |         $data = array (
77 |             "stage_id" => $this->input->post("stage_id"),
78 |             "modul_id" => $this->input->post("modul_id"),
79 |             "employee_id" => $this->input->post("employee_id"),
80 |             "employee_name" => $this->input->post("employee_name"),
81 |             "progress" => $this->input->post("progress"),
82 |             "status" => $this->input->post("status"),
83 |             "image" => $this->uploadImage()
84 |         );
85 |
86 |         $this->db->insert("workbook", $data);
87 |     }
88 | }
```

Gambar 32 Models Workbook\_model.php

Jika sudah selesai, maka langkah selanjutnya adalah membuat function update, dimana data yang dimasukkan akan diupdate berdasarkan id yang sedang digunakan.

```
89 |     public function update($id, $data)
90 |     {
91 |         $this->db->where('workbook_id', $id);
92 |         $this->db->update("workbook", $data);
93 |     }
94 | }
```

Gambar 33 Models Workbook\_model.php

Jika sudah maka langkah selanjutnya adalah membuat function delete yang akan menghapus data yang dipilih dan akan dihapus sesuai dengan id-nya. Selanjutnya adalah membuat function \_uploadImage(). Lalu kita memberikan *config* tentang *upload* gambar. Disimpan di direktori mana, lalu jenis ekstensi apa saja yang dibolehkan, lalu besar *file size* yang diperbolehkan. Setelahnya baru *load library upload* dan menggunakan *config* yang baru saja diberikan. Jika sedang melakukan *upload* dengan nama *file* aslinya, maka replace nama *file* menjadi default.jpg.

Langkah terakhir dalam membuat workbook\_model.php adalah membuat function \_deleteImage() yang akan dihapus sesuai dengan id yang dipilih, lalu hapus data yang ada di dalam tabel workbook di *database* seperti yang ditunjukkan oleh gambar 34 di bawah ini.

```

public function delete($id)
{
    $this->_deleteImage($id);
    return $this->db->delete("workbook", array("workbook_id" => $id));
}

private function _uploadImage()
{
    $config['upload_path']          = './upload/workbook/';
    $config['allowed_types']        = 'gif|jpg|png';
    $config['file_name']            = $this->workbook_id;
    $config['overwrite']            = true;
    $config['max_size']             = 10240; // 10MB
    // $config['max_width']          = 1024;
    // $config['max_height']         = 768;

    $this->load->library('upload', $config);

    if ($this->upload->do_upload('image')) {
        return $this->upload->data("file_name");
    }

    return "default.jpg";
}

private function _deleteImage($id)
{
    $workbook = $this->getId($id);
    if ($workbook->image != "default.jpg") {
        $filename = explode(".", $workbook->image)[0];
        return array_map('unlink', glob(FCPATH."upload/workbook/$filename.*"));
    }
}

```

*Gambar 34 Models Workbook\_model.php*

Jika models dan controller telah kita buat, maka langkah selanjutnya adalah membuat view. Untuk view silakan mengunduh *bootstrap* sesuai dengan keinginan masing masing, disini saya masih menggunakan argon *dashboard bootstrap* dikarenakan *bootstrap* yang saya gunakan memiliki keseluruhan halaman yang dibutuhkan. Sehingga hanya diperlukan sedikit modifikasi dari argon *dashboard*-nya.

Buatlah folder admin di dalam folder views, lalu jika sudah membuat folder admin selanjutnya buat kembali sebuah folder dengan nama workbook. Nah pada folder workbook kita selanjutnya membuat list\_workbook.php. Mengapa banyak sekali membuat folder pada folder views? Ini dimaksudkan untuk merapikan direktori kerja kita, selain itu dalam penggerjaan aplikasi akan menjadi lebih efisien. Sehingga direktori dari beberapa folder yang telah dibuat akan terlihat menjadi seperti ini C://xampp/htdocs/eppm-go!/application/views/admin/workbook.

Ingat! Hanya ubah beberapa komponen di dalam *bootstrap* sesuai dengan yang kita perlukan. Jangan mengubah semua komponen, karena akan berakibat *layout* yang telah diatur menjadi berantakan. Lalu pada file list\_workbook.php jangan lupa tambahkan *link* atau URL (*Uniform Resource Locator*) menuju menu menu yang telah atau akan dibuat. *Link* akan ada di setiap halaman yang kita buat, jadi pastikan jangan lupa menaruh *link* pada tiap tiap halaman yang dibuat Seperti yang ditunjukkan oleh gambar 35 di bawah ini

```
<ul class="navbar-nav">
- 

```

Gambar 35 Link/URL Menu

Setelah membuat *link* atau URL selanjutnya kita akan membuat tabel untuk menampilkan data yang kita punya. Pertama buat terlebih dahulu kolom untuk nama *field*-nya.

```
<table class="table align-items-center table-flush">
  <thead class="thead-light">
    <tr>
      <th scope="col">ID Stage</th>
      <th scope="col">ID Modul</th>
      <th scope="col">Nama Pekerja</th>
      <th scope="col">Progress</th>
      <th scope="col">Status</th>
      <th scope="col">Lembar Pengesahan</th>
      <th scope="col"></th>
    </tr>
  </thead>
  <tbody>
```

Gambar 36 Inisiasi Tabel Workbook

Setelah membuat kolom untuk nama *field*-nya maka kita selanjutnya harus membuat kolom untuk menempatkan atau mewadahi data yang akan ditampilkan. Agar memudahkan pembuatan tabel, kita akan menggunakan DataTables yang berguna untuk membuat tabel secara instan hanya dengan menggunakan sintaks jQuery yang masih merupakan *library* untuk Javascript.

```
<tbody>
    <?php foreach ($workbook as $workbook): ?>
    <tr>
        <td width="150">
            <?php echo $workbook->stage_id ?>
        </td>
        <td>
            <?php echo $workbook->modul_id ?>
        </td>
        <td>
            <?php echo $workbook->yaww ?>
        </td>
        <td>
            <?php echo $workbook->progress ?>
        </td>
        <td>
            <?php echo $workbook->status ?>
        </td>
        <td>
            
        </td>
        <td width="250">
            <a href="<?php echo site_url('admin/workbook/edit/'.$workbook->workbook_id)?>">
                <i class="fas fa-edit"></i>Update</a>
            <a onclick="deleteConfirm('<?php echo site_url('admin/workbook/delete/'.$workbook->workbook_id) ?>')"
               href="#" class="btn btn-small text-danger"><i class="fas fa-trash"></i>Hapus</a>
            </td>
        </tr>
    <?php endforeach; ?>
</tbody>
</table>
```

Gambar 37 Views list\_workbook.php

Disini kita menggunakan perulangan *foreach* yang mengambil data dari *models* *workbook* dan diinisiasikan sebagai variabel \$workbook. Lalu memanggil *field* yang diperlukan ke dalam tabelnya. Karena ada perulangan *foreach* maka setiap variabel yang diinisiasikan akan diulang secara terus

menerus hingga semua datanya terpanggil. Lalu kita membuat sebuah tombol untuk melakukan perubahan atau *update* dengan mengambil id dari data yang kita pilih, begitupun dengan tombol hapus sama seperti tombol *update*. Dan jangan lupa kita akhiri perulangan *foreach*-nya.

Setelah itu, untuk sintaks jQuery dari DataTables kita taruh dibawah setelah sintaks pembuatan tabel. Jangan lupa untuk menyisipkan tag <script> karena masih satu bagian dari JavaScript. Selain itu, karena kita menggunakan *resources* dari luar ada beberapa cara yang bisa digunakan, yaitu dengan cara:

1. Mengunduh *library*-nya dan disimpan pada folder *assets* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini akan menggunakan *local resource* dari PC atau laptop kita.
2. Menyalin *link CDN* (*Content Delivery Network*) yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini hanya memerlukan akses internet. *Link CDN* DataTables untuk CSS adalah //cdn.datatables.net/1.10.20/css/jquery.dataTables.min.css dan *link CDN* DataTables untuk JavaScript adalah //cdn.datatables.net/1.10.20/js/jquery.dataTables.min.js

```
<script>$(document).ready( function () {
|   $('table').DataTable();
}
);
</script>
```

Gambar 38 Sintaks DataTables

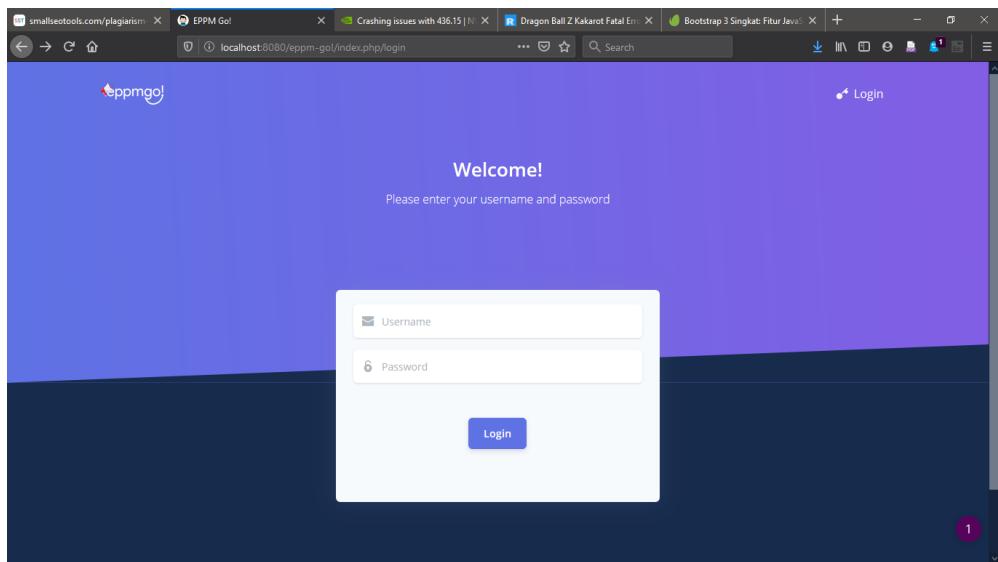
Lalu kita akan menggunakan modals yang juga masih bagian dari JavaScript, yang akan digunakan untuk tombol hapus. Modals adalah sebuah kotak dialog kecil yang biasanya muncul di atas halaman yang sedang kita buka. Sehingga apabila tombol hapus tidak sengaja ditekan maka akan menampilkan dialog konfirmasi untuk menghapus sebuah data, seperti yang ditunjukkan oleh gambar 39 berikut.

```
<script>
function deleteConfirm(url){
    $('#btn-delete').attr('href', url);
    $('#deleteModal').modal();
}
</script>
```

Gambar 39 Modal Tombol Hapus

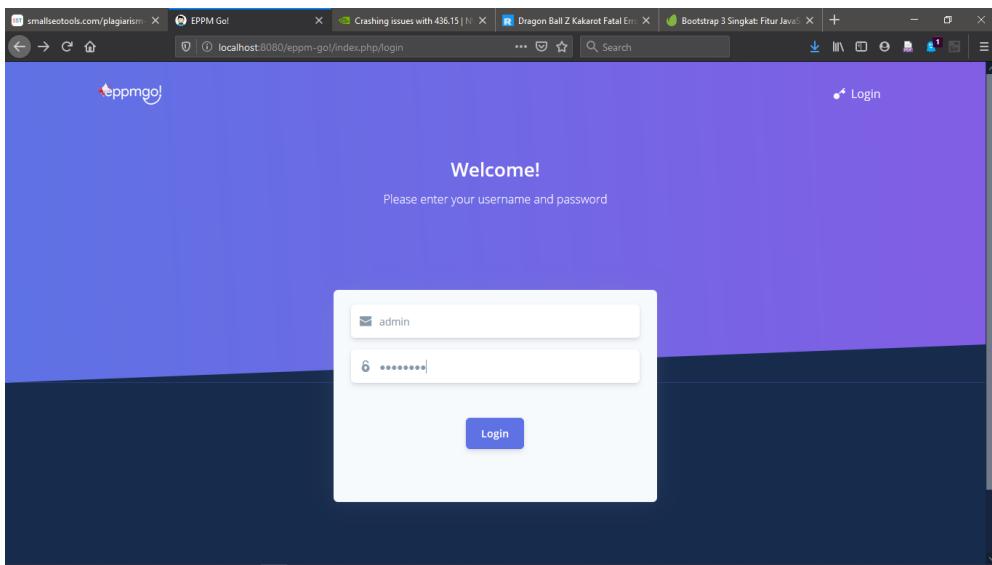
Jika controller, models, dan views telah kita buat maka langkah selanjutnya adalah mencoba untuk memanggil halaman web yang baru saja kita buat. Apakah terdapat error saat dibuka atau kan sudah sesuai dengan yang kita inginkan? Untuk itu silakan coba membuka *browser* atau peramban sesuai dengan preferensi kita. Dan coba ketikkan localhost:8080/eppm-go!/index.php/login.

Untuk pengaturan localhost masing masing pengguna pasti akan berbeda beda, dikarenakan *port* yang digunakan untuk mengakses *localhost*-nya. Karena saya menggunakan *port* 8080 untuk mengakses *localhost* makanya ada tambahan nama *port* di samping *localhost*.



Gambar 40 Halaman Login

Berhasil masuk ke halaman *login*, maka kita harus mengisi *username* dan *password* agar bisa mengakses halaman berikutnya. Karena halaman *workbook* yang baru saja kita buat adalah halaman untuk admin maka kita akan menggunakan *username* admin dan juga *password* admin123. Karena admin yang mengakses maka semua menu dan fitur fitur yang disediakan dapat diakses oleh admin. Sedangkan untuk pengguna akan mendapatkan limitasi terhadap menu dan fitur yang dapat digunakan.



Gambar 41 Halaman Login

A screenshot of a web browser showing the 'Workbook' list page. The interface includes a sidebar with navigation links like Dashboard, Daftar Peserta, Workbook (which is selected and highlighted in blue), Assessment, Data Ruangkerja, Report, and Logout. The main content area is titled 'Workbook' and features a table with columns for PROGRESS, STATUS, and LEMBAR PENGESAHAN. The table lists several entries, each with a small profile picture, a progress bar icon, an 'Update' button, and a 'Hapus' button. The rows show various status updates such as 'On Progress', 'Acc. Done', and 'Acc. Pending'.

Gambat 42 Halaman List Workbook

Setelah halaman *workbook* selesai dibuat, maka selanjutnya adalah membuat form *update* untuk dapat mengubah beberapa data yang kita masukkan sebelumnya. Untuk form *update* baru akan muncul setelah mengklik tombol *update*, maka langsung saja kita buat sebuah *file* dengan nama edit\_form.php pada direktori yang sama yaitu C://xampp/htdocs/eppm-go!/application/views/admin/workbook.

```
<div class="card mb-3">
  <div class="card-header">
    <a href="=site_url('admin/workbook/')?><i class="fas fa-arrow-left"></i>
    Kembali</a>
  </div>
  <div class="card-body">

    <form action="=base_url();?&gt;index.php/admin/workbook/aksi_edit" method="post" enctype="multipart/form-data"&gt;

      &lt;input type="hidden" name="workbook_id" value="<?=echo $workbook-&gt;workbook_id?&gt;" /&gt;

      &lt;div class="form-group"&gt;
        &lt;label for="name"&gt;Stage&lt;/label&gt;
        &lt;input class="form-control" type="text" name="stage_id" placeholder="Stage" value="<?=echo $workbook-&gt;stage_id?&gt;" readonly&gt;
        &lt;div class="invalid-feedback"&gt;
          &lt;?php echo form_error('stage') ?&gt;
        &lt;/div&gt;
      &lt;/div&gt;

      &lt;div class="form-group"&gt;
        &lt;label for="name"&gt;Modul&lt;/label&gt;
        &lt;input class="form-control" type="text" name="modul_id" min="0" placeholder="Modul" value="<?=echo $workbook-&gt;modul_id?&gt;" readonly&gt;
        &lt;div class="invalid-feedback"&gt;
          &lt;?php echo form_error('modul_id') ?&gt;
        &lt;/div&gt;
      &lt;/div&gt;</pre
```

Gambar 43 Form edit\_form.php

Sesuai dengan yang ditunjukkan oleh gambar 43, disini kita membuat sebuah form menggunakan sintaks bawaan yang disediakan oleh *bootstrap argon dashboard*. Pada tiap *textbox* pada form, kita memanggil kembali data data yang akan diubah sesuai dengan id yang dipilih, sama seperti yang kita lakukan sebelumnya pada halaman list\_workbook.php

```

<div class="form-group">
    <label for="name">Nama Pekerja</label>
    <input class="form-control" type="text" name="employee_name" min="0" placeholder="Nama Pekerja" value=<?php echo $workbook->yaww ?>
        <?php echo form_error('employee_name') ?>
    </div>
</div>

<div class="form-group">
    <label for="name">Progress*</label>
    <input class="form-control" type="text" name="progress" min="0" placeholder="Progress" value=<?php echo $workbook->progress ?>" required
        <?php echo form_error('progress') ?>
    </div>
</div>

<div class="form-group">
    <label for="name">Status*</label>
    <select class="form-control" name="status">
        <?php foreach ( $status as $status ) : ?>
        <?php if ( $status == $workbook->status ) : ?>
            <option value=<?php echo $status; ?>><?php echo $status; ?></option>
        <?php else : ?>
            <option value=<?php echo $status; ?>><?php echo $status; ?></option>
        <?php endif; ?>
        <?php endforeach; ?>
    </select>
    <div class="invalid-feedback">
        <?php echo form_error('status') ?>
    </div>
</div>

```

Gambar 44 Form edit\_form.php

Dapat dilihat pada gambar 43 sampai gambar 44, pada *textbox stage*, modul, nama pekerja, dan juga progres hanya memanggil data berdasarkan variabel yang telah kita definisikan sebelumnya pada models. Sedangkan untuk status, pada gambar dicoba untuk *foreach* untuk menampilkan data status akan dipilih dalam bentuk *combo box*. Selanjutnya untuk *textbox* lembar pengesahan sama saja dengan *textbox* lainnya hanya memanggil data berdasarkan variabel yang telah didefinisikan sebelumnya. Lalu ada tombol simpan untuk menyimpan perubahan perubahan yang dilakukan di form ini ke dalam *database*. Seperti yang ditunjukkan oleh gambar 45 berikut ini.

```
<div class="form-group">
    <label for="name">Lembar Pengesahan</label>
    <input class="form-control-file" type="file" name="image" />
    <input type="hidden" name="old_image" value="<?php echo $workbook->image ?>" />
    <div class="invalid-feedback">
        <?php echo form_error('image') ?>
    </div>
</div>
</div>

<input class="btn btn-success" type="submit" name="btn" value="Simpan" />
</form>
```

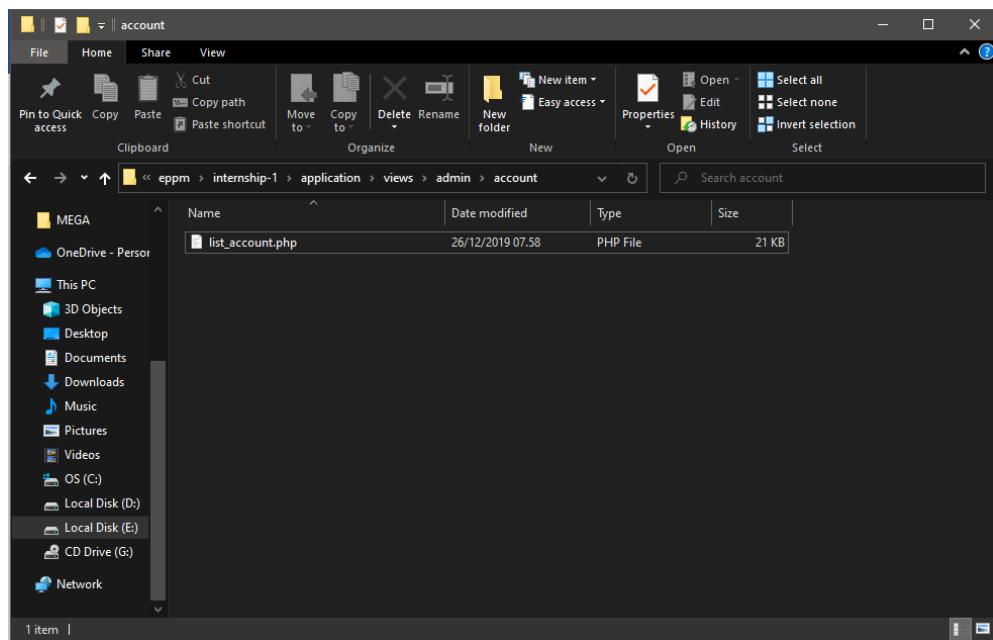
Gambar 45 Form edit\_form.php

Dengan ini, selesailah sudah halaman *workbook* untuk admin. Mengapa tidak ada form untuk meng-*input* data *workbook* yang baru? Karena hal itu dilakukan untuk pengguna atau *user*. Jadi admin hanya untuk memantau apakah sudah melaksanakan *workbook* atau belum, nah untuk form tersebut baru akan kita buat pada tahapan *workbook* untuk *user* pada beberapa sub bab berikutnya. Lanjut terus ya bacanya!

## 2.4. CRUD Menu Daftar Pekerja

Untuk CRUD menu daftar pekerja, ini sebenarnya sama saja seperti CRUD menu *workbook*. Ada beberapa hal yang sama diulangi dan ada beberapa hal yang diganti menyesuaikan dengan kebutuhan pada menu yang akan dibuat. Maka langsung saja kita buat folder *account* pada direktori controller seperti berikut C://xampp/htdocs/eppm-go!/application/controller/admin/account.

Pada folder *account* buatlah sebuah *file* controller dengan nama Account.php lalu isikan di dalam *file* Account.php seperti yang ditunjukkan pada gambar 46 berikut ini.



Gambar 46 Controller Account.php

Seperti pada *file* controller Workbook.php maka kita terlebih dahulu membuat sebuah class dengan nama Account dengan menggunakan *library* CI\_Controller. Lalu membuat function \_\_construct() yang akan me-load model account\_model dan juga me-load *library* form validasi. Karena sudah me-load model, maka kita gunakan function is\_role() pada account\_model yang akan mengecek apabila *role* yang masuk ke dalam aplikasi bukan admin maka akan diarahkan kembali ke halaman *login*.

Lalu setelah itu kita membuat function index() yang medefinisikan variabel \$data yang datanya diambil melalui function getAll() pada account\_model. Jika sudah maka akan menampilkan halaman list\_account dengan membawa data yang sudah didefinisikan pada variabel \$data sebelumnya. Seperti yang terlihat pada gambar 47 berikut ini.

```
<?php
defined('BASEPATH') OR exit ('No direct script access allowed');

class Account extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        $this->load->model("account_model"); //Load model Account
        $this->load->library('form_validation'); //Load Library form validation
        if($this->account_model->is_role() != "admin"){
            redirect("index.php/login");
        }
    }

    public function index()
    {
        $data["account"] = $this->account_model->getAll(); //ambil data dari model
        $this->load->view("admin/account/list_account", $data); //load view data model ke account
    }
}
```

Gambar 47 Controller Account.php

Jika sudah maka kita lanjut untuk membuat function add() dengan mendefinisikan variabel \$account pada account\_model dan juga variabel \$validation pada form validasi. Lalu variabel \$validation tersebut kita terapkan *rules*. Setelahnya kita tampilkan halaman tambah data atau form tambah data.

```
2 |     public function add()
3 |     {
4 |         $account = $this->account_model; //objek model
5 |         $validation = $this->form_validation; //objek form validation
6 |         $validation->set_rules($account->rules()); //terapkan rules
7 |
8 |         if ($validation->run()){
9 |             $account->save();
10 |             $this->session->set_flashdata('Sukses', 'Data Anda Berhasil Disimpan'); //pesan berhasil
11 |         }
12 |
13 |         $this->load->view("admin/account/new_account"); //load isi form account
14 |     }
```

Gambar 48 Controller Account.php

Setelahnya kita tambahkan function edit() yang akan mengambil datanya berdasarkan id yang dipilih, lalu akan ditampilkan kembali halaman form edit dengan data dari id yang sudah dipilih.

```
public function edit($id=null)
{
    if(!isset($id)) redirect('admin/account/list_account');

    $account = $this->account_model;
    $validation = $this->form_validation;
    $validation->set_rules($account->rules());

    if ($validation->run()){
        $account->update();
        $this->session->set_flashdata('success', 'Data Anda Berhasil Diupdate'); //pesan berhasil
    }

    $data["account"] = $account->getById($id); //mengambil data berdasarkan id
    if (!$data["account"]) show_404(); // jika tidak ada show error
    $this->load->view("admin/account/edit_account", $data); //Load edit form account
}
```

Gambar 49 Controller Account.php

Jangan lupa untuk membuat function delete() dengan mengambil data berdasarkan id yang kita pilih, sehingga hanya data yang dipilih saja yang terhapus. Jika sudah berhasil terhapus maka akan diarahkan kembali ke halaman daftar pekerja.

```
public function delete($id=null)
{
    if (!isset($id)) show_404();

    if ($this->account_model->delete($id)){
        redirect('index.php/admin/account');
    }
}
```

Gambar 50 Controller Account.php

Dengan ini, kita telah selesai membuat *file* controller Account.php. Maka langkah selanjutnya adalah kita haruss membuat *file* models account\_model.php pada direktori models di localhost kita. Setelah selesai membuat *file* tersebut langsung saja kita isikan *file*-nya dengan isian seperti yang ditunjukkan pada gambar 51 berikut.

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Account_model extends CI_Model
{
    private $_table = "account";

    public $employee_id;
    public $directorate;
    public $employee_name;
    public $username;
    public $password;
    public $user_email;
    public $role;

    public function rules()
    {
        return [
            ['field' => 'employee_id',
            'label' => 'No. Pekerja',
            'rules' => 'required'],

            ['field' => 'employee_name',
            'label' => 'Nama Pekerja',
            'rules' => 'required'],

            ['field' => 'username',
            'label' => 'username',
            'rules' => 'required'],

            ['field' => 'password',
            'label' => 'password',
            'rules' => 'required']
        ];
    }
}
```

Gambar 51 Models Account\_model.php

Seperti yang ditunjukkan gambar 51 di atas, terlebih dahulu kita harus mendefinisikan variabel yang akan digunakan untuk menu daftar pengguna. Jika sudah didefinisikan, maka lanjut dengan membuat function rules() dimana ini merupakan aturan aturan yang akan kita gunakan untuk menu ini. Dapat dilihat hanya ada 4 *field* yang diberi aturan tidak boleh kosong alias wajib diisi!.

Jika sudah selesai, maka kita selanjutnya membuat function is\_role() yang akan berfungsi untuk mengecek *role* yang masuk ke dalam aplikasi berdasarkan *session* yang aktif. Lalu kita juga membuat function getAll() yang berfungsi untuk mengambil semua data pada tabel account. Nah sama seperti function getAll() kita juga membuat function getById() yang berfungsi untuk mengambil data berdasarkan id-nya. Disini employee\_id dijadikan *identifier* dari id yang akan diambil. Untuk lebih lengkapnya silakan lihat gambar 52 di bawah ini.

```
//fungsi cek level
function is_role()
{
    return $this->session->userdata('role');
}

public function getAll()
{
    return $this->db->get_where($this->_table, array('role' => 'user'))->result();
}

public function getById($id)
{
    return $this->db->get_where($this->_table, ["employee_id" => $id])->row();
}
```

Gambar 52 Models Account\_model.php

Langkah selanjutnya adalah membuat function save() yang berfungsi untuk menyimpan data yang telah kita *input* kan ke dalam *database* pada tabel account. Begitupun dengan function edit() yang dibuat setelah function save() selesai, function edit() akan menampilkan data yang telah disimpan sebelumnya berdasarkan id yang dipilih. Lalu apabila telah mengubah data yang diperlukan maka akan langsung meng-*update* data tersebut sesuai dengan id yang dipilih.

Sama seperti function edit(), function delete() juga mengambil data berdasarkan id yang dipilih akan tetapi peruntukannya berbeda. Jika edit() untuk mengubah data, maka delete() untuk menghapus data yang kita pilih. Untuk lengkapnya dapat dilihat pada gambar 53 di bawah ini.

```
public function save()
{
    $post = $this->input->post();
    $this->employee_id = $post["employee_id"];
    $this->directorate = $post["directorate"];
    $this->employee_name = $post["employee_name"];
    $this->username = $post["username"];
    $this->password = $post["password"];
    $this->user_email = $post["user_email"];
    $this->role = $post["role"];
    $this->db->insert($this->_table, $this);
}

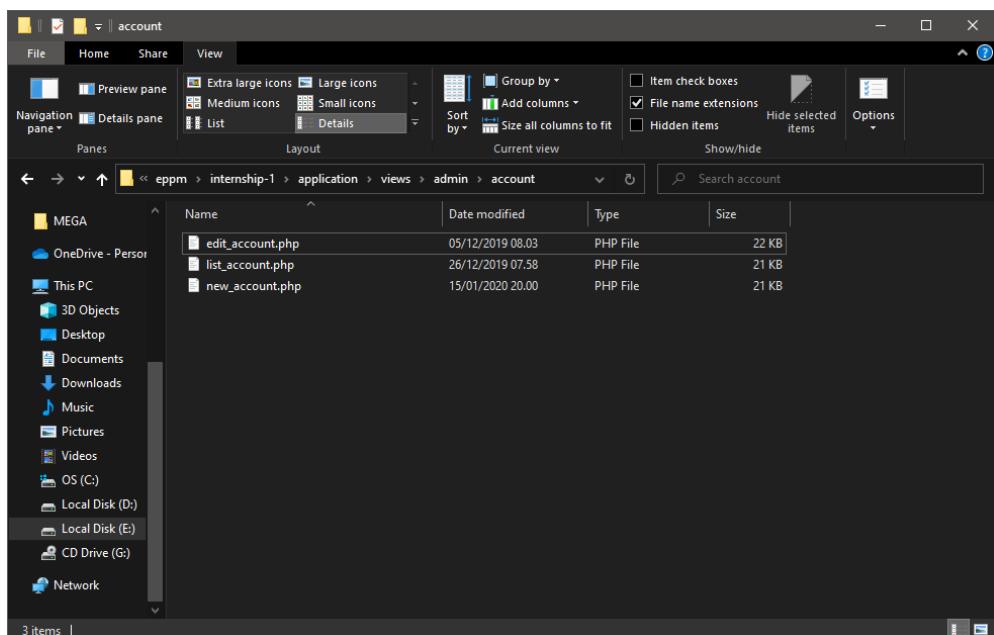
public function update()
{
    $post = $this->input->post();
    $this->employee_id = $post["employee_id"];
    $this->directorate = $post["directorate"];
    $this->employee_name = $post["employee_name"];
    $this->username = $post["username"];
    $this->password = $post["password"];
    $this->user_email = $post["user_email"];
    $this->role = $post["role"];
    $this->db->update($this->_table, $this, array('employee_id' => $post['employee_id']));
}

public function delete($id)
{
    return $this->db->delete($this->_table, array("employee_id" => $id));
}
```

Gambar 53 Models Account\_model.php

Mantap! Selesai sudah tahapan-tahapan yang diperlukan untuk membuat file models Account\_model.php. Nah bila temen-temen yang lain sudah maka bisa langsung dilanjutkan untuk membuat file views-nya yang berfungsi untuk menampilkan datanya ke dalam sebuah *user interface* yang lebih nyaman untuk dilihat oleh pengguna. Untuk itu buat folder account pada direktori views dan admin, dan buat kembali sebuah file dengan nama list\_account.php, edit\_account.php dan juga new\_account.php.

Sekarang fokusnya akan kita alihkan hanya kepada file list\_account.php yang berguna untuk menampilkan seluruh data yang ada pada database di dalam tabel account.



Gambar 54 File Account

Jangan lupa, untuk setiap menu dan halaman yang akan dibuat kita harus selalu menaruh *link* atau URL yang akan mengarah pada 5 menu utama yang akan dibuat seperti yang ditunjukkan pada gambar 55 di bawah ini.

```
<ul class="navbar-nav">
<li class="nav-item" >
    <a class=" nav-link " href="=php echo base_url(); ?index.php/admin" ><i class="ni ni-tv-2 text-primary"></i> Das
    </a>
</li>
<li class="nav-item" class=" active" >
    <a class="nav-link active " href="=php echo base_url(); ?admin/account" >
        | <i class="ni ni-bullet-list-67 text-red"></i> Daftar Peserta
    </a>
</li>
<li class="nav-item">
    <a class="nav-link " href="=php echo base_url(); ?admin/workbook" >
        | <i class="ni ni-bullet-list-67 text-red"></i> Workbook
    </a>
</li>
```

Gambar 55 Views Link atau URL

Lanjut lagi, kita membuat sebuah tabel dengan terlebih dahulu membuat kolom untuk nama *field*-nya seperti yang ditunjukkan oleh gambar 56 di bawah ini.

```
<div class="row">
    <div class="col">
        <div class="card shadow">
            <div class="card-header border-0">
                <h3 class="mb-0">Daftar Peserta</h3>
                <a href="=php echo site_url('admin/account/add') ?"><i class="fas fa-plus"></i> Tambah Baru</a>
            </div>
            <div class="table-responsive">
                <table id="table" class="table align-items-center table-flush" >
                    <thead class="thead-light">
                        <tr>
                            <th scope="col">Nomor Pekerja</th>
                            <th scope="col">Direktorat</th>
                            <th scope="col">Nama Pekerja</th>
                            <th scope="col">Username</th>
                            <!-- <th scope="col">Password</th> -->
                            <th scope="col">Email</th>
                            <!-- <th scope="col">Role</th> -->
                            <th scope="col"></th>
                        </tr>
                    </thead>
```

Gambar 56 Views list\_account.php

Setelahnya baru kita masukkan datanya dengan cara memanggil variabel yang telah didefinisikan sebelumnya pada bagian models. Lalu kita melakukan perulangan *foreach* untuk mengulang pemanggilan data sampai datanya telah terpanggil semua, sehingga tidak hanya satu data saja yang dipanggil.

Lalu jangan lupa buat juga tombol *update* dan juga *hapus*, dimana tombol *update* akan mengambil id dari data yang dipilih dan akan langsung diarahkan menuju form *update*. Sedangkan untuk tombol *hapus* akan mengambil id dari data yang dipilih untuk kita hapus. Oh iya pada menu daftar pekerja disini juga menggunakan DataTables, jadi cara penggunaannya masih sama seperti yang telah kita lakukan di menu workbook. Untuk lebih jelasnya lihat gambar 57 berikut ini.

```
<tbody>
<?php foreach ($account as $account): ?>
<tr>
    <td width="150">
        <?php echo $account->employee_id ?>
    </td>
    <td width="150">
        <?php echo $account->directorate ?>
    </td>
    <td>
        <?php echo $account->employee_name ?>
    </td>
    <td width="150">
        <?php echo $account->username ?>
    </td>
    <!-- <td width="150">
        <?php echo $account->password ?>
    </td> -->
    <td>
        <?php echo $account->user_email ?>
    </td>
    <!-- <td>
        <?php echo $account->role ?>
    </td> -->
    <td width="250">
        <a href=<?php echo site_url('admin/account/edit/'.$account->employee_id)?>">
            class="btn btn-small"><i class="fas fa-edit"></i>Update</a>
        <a onclick="deleteConfirm('<?php echo site_url('admin/account/delete/'.$account->employee_id) ?>')>
            href="#" class="btn btn-small text-danger"><i class="fas fa-trash"></i>Hapus</a>
        </td>
    </tr>
<?php endforeach; ?>
</tbody>
</table>
```

Gambar 57 Views list\_account.php

Seperti yang sudah dijelaskan barusan, kita masih menggunakan DataTables jadi kita harus menggunakan sintaks jQuery untuk menggunakan *resources* dari DataTables. Setelah itu, untuk sintaks jQuery dari DataTables kita taruh dibawah setelah sintaks pembuatan tabel. Jangan lupa untuk menyisipkan tag <script> karena masih satu bagian dari JavaScript. Selain itu, karena kita menggunakan *resources* dari luar ada beberapa cara yang bisa digunakan, yaitu dengan cara:

1. Mengunduh *library*-nya dan disimpan pada folder *assets* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini akan menggunakan *local resource* dari PC atau laptop kita.
2. Menyalin *link CDN (Content Delivery Network)* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini hanya memerlukan akses internet. *Link CDN* DataTables untuk CSS adalah //cdn.datatables.net/1.10.20/css/jquery.dataTables.min.css dan *link CDN* DataTables untuk JavaScript adalah //cdn.datatables.net/1.10.20/js/jquery.dataTables.min.js

```
</div>
<script>$(document).ready( function () {
    $('table').DataTable();
} );
</script>
</div>
```

Gambar 58 Views list\_account.php

Jangan lupa gunakan modal untuk konfirmasi sebelum dilakukannya penghapusan sebuah data, seperti yang ditunjukkan pada gambar 59 berikut ini.

```
<script>
function deleteConfirm(url){
    $('#btn-delete').attr('href', url);
    $('#deleteModal').modal();
}
</script>
```

Gambar 59 Views list\_account.php

Karena untuk list sudah selesai, maka kita akan lanjutkan kembali ke halaman form tambah data atau form *input* karena sudah dibuat sebelumnya *file* new\_account.php maka langsung saja diberi isian seperti yang akan ditunjukkan oleh gambar 60, dan 62 berikut.

```

<div class="card mb-3">
    <div class="card-header">
        <a href="=php echo site_url('admin/account/') ?&gt;"&gt;&lt;i class="fas fa-arrow-left"&gt;&lt;/i&gt; Back&lt;/a&gt;
    &lt;/div&gt;
    &lt;div class="card-body"&gt;
        &lt;form action="<?=php base_url('admin/account/add') ?&gt;" method="post" enctype="multipart/form-data"&gt;

            &lt;!-- &lt;input type="hidden" name="account_id" value="<?=php echo $account-&gt;employee_id?&gt;" /&gt; --&gt;

            &lt;div class="form-group"&gt;
                &lt;label for="name"&gt;No. Pekerja*&lt;/label&gt;
                &lt;input class="form-control" type="text" name="employee_id" placeholder="Nomor Pekerja" /&gt;
                &lt;div class="invalid-feedback"&gt;<?=php echo form_error('employee_id')?&gt;
            &lt;/div&gt;
        &lt;/div&gt;

        &lt;div class="form-group"&gt;
            &lt;label for="name"&gt;Direktorat&lt;/label&gt;
            &lt;input class="form-control" type="text" name="directorate" placeholder="Direktorat" /&gt;
            &lt;div class="invalid-feedback"&gt;<?=php echo form_error('directorate')?&gt;
        &lt;/div&gt;
    &lt;/div&gt;

    &lt;div class="form-group"&gt;
        &lt;label for="name"&gt;Nama Pekerja*&lt;/label&gt;
        &lt;input class="form-control" type="text" name="employee_name" placeholder="Nama Pekerja" /&gt;
        &lt;div class="invalid-feedback"&gt;<?=php echo form_error('employee_name')?&gt;
    &lt;/div&gt;
&lt;/div&gt;
</pre

```

Gambar 60 Views new\_account.php

```

<div class="form-group">
    <label for="name">Username*</label>
    <input class="form-control" type="text" name="username" placeholder="Username" />
    <div class="invalid-feedback">=php echo form_error('username')?&gt;
&lt;/div&gt;
&lt;/div&gt;

&lt;div class="form-group"&gt;
    &lt;label for="name"&gt;Password*&lt;/label&gt;
    &lt;input class="form-control" type="password" name="password" placeholder="Password" /&gt;
    &lt;div class="invalid-feedback"&gt;<?=php echo form_error('modul_id')?&gt;
&lt;/div&gt;
&lt;/div&gt;

&lt;div class="form-group"&gt;
    &lt;label for="name"&gt;Email&lt;/label&gt;
    &lt;input class="form-control" type="text" name="user_email" placeholder="Status" /&gt;
    &lt;div class="invalid-feedback"&gt;<?=php echo form_error('user_email')?&gt;
&lt;/div&gt;
&lt;/div&gt;

&lt;div class="form-group"&gt;
    &lt;label for="name"&gt;Role&lt;/label&gt;
    &lt;select class="form-control" name="role"&gt;<?=php echo form_error('role')?&gt;
        &lt;option value="admin"&gt;Admin&lt;/option&gt;
        &lt;option value="user"&gt;User&lt;/option&gt;
    &lt;/select&gt;
    &lt;div class="invalid-feedback"&gt;<?=php echo form_error('role')?&gt;
&lt;/div&gt;
&lt;/div&gt;
</pre

```

Gambar 61 Views new\_account.php

Dengan ini *file* new\_account.php sudah selesai dibuat. Maka langkah selanjutnya adalah mengisi *file* edit\_account.php yang sebelumnya telah kita buat. Langsung saja *file* tersebut kita isi dengan isian seperti yang ditunjukkan pada gambar 62 di bawah ini.



```
<div class="card mb-3">
  <div class="card-header">
    <a href="=php echo site_url('admin/account/') ?&gt;"&gt;&lt;i class="fas fa-arrow-left"&gt;&lt;/i&gt;Kembali&lt;/a&gt;
  &lt;/div&gt;
  &lt;div class="card-body"&gt;

    &lt;form action="<?=php base_url('admin/account/edit') ?&gt;" method="post" enctype="multipart/form-data"&gt;
      &lt;!-- &lt;input type="hidden" name="account_id" value="<?=php echo $account-&gt;employee_id?&gt;" /&gt; --&gt;

      &lt;div class="form-group"&gt;
        &lt;label for="name"&gt;No. Pekerja*&lt;/label&gt;
        &lt;input class="form-control" type="text" name="employee_id" placeholder="Nomor Pekerja" value="<?=php echo $account-&gt;employee_id?&gt;"&gt;
        &lt;div class="invalid-feedback"&gt;<?=php echo form_error('employee_id')?&gt;
      &lt;/div&gt;
    &lt;/div&gt;

    &lt;div class="form-group"&gt;
      &lt;label for="name"&gt;Direktorat&lt;/label&gt;
      &lt;input class="form-control" type="text" name="directorate" placeholder="Direktorat" value="<?=php echo $account-&gt;directorate?&gt;"&gt;
      &lt;div class="invalid-feedback"&gt;<?=php echo form_error('directorate')?&gt;
    &lt;/div&gt;
  &lt;/div&gt;</pre
```

Gambar 62 Views edit\_account.php

Sesuai dengan gambar 62 di atas, kita membuat sebuah *textbox* sesuai dengan *field* yang kita butuhkan. Lalu ambil datanya dengan memanggil variabel yang telah didefinisikan pada models Account\_model.php. Jangan lupa untuk memanggil datanya sesuai dengan id yang dipilih, maka panggil variabel employee\_id pada models melalui inisiasi variabel \$account di views. Untuk *textbox* selanjutnya akan sama saja seperti yang kita lakukan untuk *textbox* No. Pekerja.

```

<div class="form-group">
    <label for="name">Nama Pekerja*</label>
    <input class="form-control" type="text" name="employee_name" placeholder="Nama Pekerja" value=<?php echo $account->employee_na
    <div class="invalid-feedback"><?php echo form_error('employee_name')?>
    </div>
</div>

<div class="form-group">
    <label for="name">Username*</label>
    <input class="form-control" type="text" name="username" placeholder="Username" value=<?php echo $account->username?>"/>
    <div class="invalid-feedback"><?php echo form_error('username')?>
    </div>
</div>

<div class="form-group">
    <label for="name">Password*</label>
    <input class="form-control" type="password" name="password" placeholder="Password" value=<?php echo $account->password?>"/>
    <div class="invalid-feedback"><?php echo form_error('password')?>
    </div>
</div>

<div class="form-group">
    <label for="name">Email</label>
    <input class="form-control" type="text" name="user_email" placeholder="Email" value=<?php echo $account->user_email?> "/>
    <div class="invalid-feedback"><?php echo form_error('user_email')?>
    </div>
</div>

```

Gambar 63 Views edit\_account.php

Sama saja seperti yang telah kita lakukan, maka tinggal kita ikuti saja langkah – langkah di atas. Lalu apabila ada salah satu *textbox* yang harus diisi, karena sudah menerapkan *rules* pada models kita tinggal menggunakananya saja.

Jika *file* edit\_account.php sudah selesai dibuat, maka selesai sudah tahapan yang harus dijalani untuk membuat menu daftar pekerja. Untuk itu kita langsung jalankan pada browser halaman yang baru saja kita buat, dengan memanggil *link* seperti ini localhost:8080/eppm-go!/admin/account. Lalu jangan lupa saat kita membuka *link* di atas, akan diarahkan pertama kali untuk melakukan *login* maka tinggal masukkan *username* dan *password* yang sama

untuk admin. Setelahnya kita langsung menuju halaman daftar pekerja yang akan terlihat seperti yang ditunjukkan pada gambar 64 di bawah ini

The screenshot shows a web browser window with multiple tabs open. The active tab is 'localhost:8080/eppm-go/admin/account'. The page has a purple header with the EPPM Go! logo and a user profile for 'Jessica Jones'. On the left, there's a sidebar with navigation links: Dashboard, Daftar Peserta (selected), Workbook, Assessment, Data Ruangkerja, Raport, Logout, DOCUMENTATION, Getting started, Foundation, and Components. The main content area is titled 'Daftar Peserta' with a 'Tambah Baru' button. It features a search bar and a table with columns: NAMA PEKERJA, USERNAME, EMAIL, Update, and Hapus. The table contains seven rows of employee data:

NAMA PEKERJA	USERNAME	EMAIL	Update	Hapus
User	user	user@email.com	<input checked="" type="checkbox"/>	
pekerja	pekerja	pekerja@email.com	<input checked="" type="checkbox"/>	
Budi Jaya	budi	budijaya@gmail.com	<input checked="" type="checkbox"/>	
Rizal	rizal	rizal	<input checked="" type="checkbox"/>	
Riandaka Rizal	riandakar	riandakarizal@gmail.com	<input checked="" type="checkbox"/>	
Petrokimia	Bambang Purwito H.	bambangpurwi	<input checked="" type="checkbox"/>	
Petrokimia	Widodo Tri Rahardjo	widodotri	<input checked="" type="checkbox"/>	

Gambar 64 Halaman Daftar Pekerja

Saat kita mengklik menu daftar pekerja, maka yang akan pertama kali ditampilkan adalah tabel dari data yang telah kita masukkan sebelumnya. Lalu kita coba untuk menambahkan data baru ke dalam tabel tersebut, dengan mengklik tombol tambah baru di pojok kiri atas yang akan mengarahkan ke halaman form tambah baru atau form *input* seperti yang ditunjukkan pada gambar 65 di bawah ini.

The screenshot shows a web-based application interface for 'eppmgo'. On the left, there's a sidebar with navigation links: Dashboard, Account (which is currently selected), Workbook, Assessment, Logout, Documentation, Getting started, Foundation, and Components. The main content area has a header with a back button and a search bar. Below this, there's a form titled 'Add Account'. The form fields are: 'No. Pekerja\*' (Nomor Pekerja), 'Direktorat' (Direktorat), 'Nama Pekerja\*' (Nama Pekerja), 'Username\*' (Username), 'Password\*' (Password), and 'Email' (Status). A small purple vertical bar on the right side of the page has a number '1' in a circle.

Gambar 65 Halaman Form Input

Tinggal kita masukkan data pekerja yang akan ditambahkan, lalu jika sudah selesai mengisi semua *textbox*-nya maka tinggal klik tombol simpan pada akhir halaman atau di bawah halaman daftar pekerja seperti yang ditunjukkan pada gambar 66 di bawah ini.

The screenshot shows a web-based application interface for 'EPPM Go!' with a purple header bar. The main content area is a form for adding a new account. It includes fields for 'Email' (containing 'ema'), 'Password\*' (containing '\*\*\*\*\*'), 'Email' (containing 'ema@email.com'), and 'Role' (containing 'User'). A green 'Simpan' button is visible. Below the form, a note says '\*Harus Dilisi'. At the bottom left, there's a footer with the text '© 2019 PT. Pertamina (Persero)'. On the left side, there's a sidebar with navigation links: 'Dashboard', 'Account', 'Workbook', 'Assessment', 'Logout', 'DOCUMENTATION', 'Getting started', 'Foundation', and 'Components'. The URL in the browser address bar is 'localhost:8080/eppm-go/admin/account/add'.

Gambar 66 Halaman Form Input

Maka hasilnya akan terlihat pada tabel di halaman awal daftar pekerja saat awal kita mengklik halaman daftar pekerja. Dan data yang baru saja kita tambahkan adalah seperti yang ditunjukkan pada gambar 67 di bawah ini. Seluruh data yang kita tambahkan pada setiap halaman bersifat *real-time* atau akan ter-*update* sesuai dengan waktu sekarang, sehingga ini merupakan salah satu keunggulan yang dipunyai oleh aplikasi yang sedang kita bangun atau *develop*. Membuat aplikasi seperti ini tidaklah sulit, hanya membutuhkan kesabaran dan ketekunan.

+ Tambah Baru				
Show 10 entries				
NOMOR PEKERJA	DIREKTORAT	NAMA PEKERJA	USERNAME	EMAIL
2	User	User	user	user@email.com
111223	tes	pekerja	pekerja	pekerja@email.com
111234	Internship FHCI	Budi Jaya	budi	budijaya@gmail.com
112234	Magang FHCI	Rizal	rizal	rizal
123456	Internship FHCI	Riandaka Rizal	riandakar	riandakarizal@gmail.com
232122	SDM	Ema Ainun Novia	ema	ema@email.com
705423	Megaprojek Pengolahan dan Petrokimia	Bambang Purwito H.	bambangpurwi	bampur@pertamina.com
706169	Megaprojek Pengolahan dan Petrokimia	Widodo Tri Rahardjo	widodotri	widodo.rahardjo@pertamina.com
706655	Megaprojek Pengolahan dan Petrokimia	Arie Wisianto	arie.wisianto	arie.wisianto@pertamina.com

Gambar 67 Halaman Daftar Pekerja

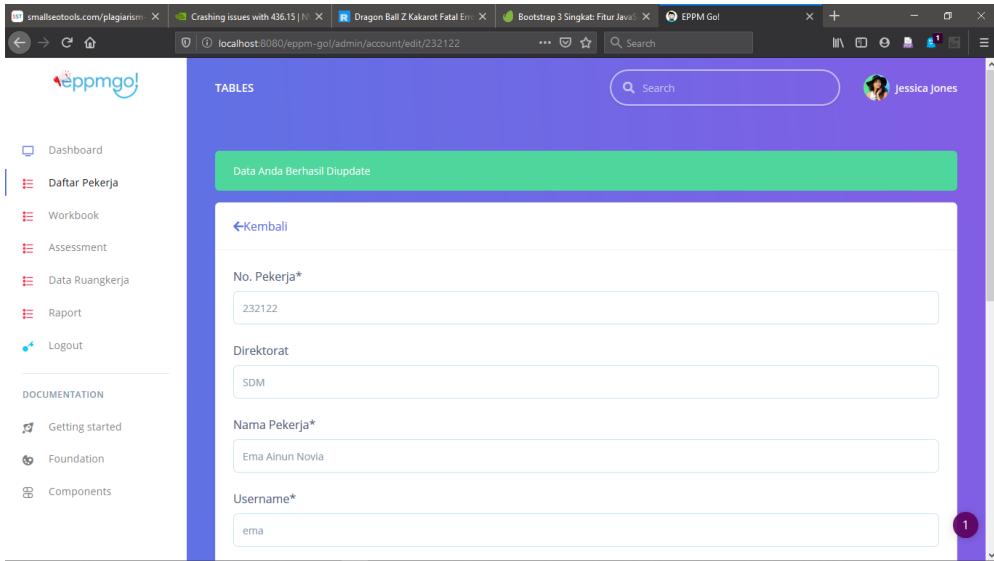
Lanjut kembali kita akan mencoba mengubah atau meng-*update* data dari pengguna Ema Ainun Novia, maka kita langsung klik saja tombol *update* yang tersedia di samping tabelnya. Setelah di klik kita akan diarahkan menuju halaman form *update* seperti yang ditunjukkan pada gambar 68 di bawah ini.

The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'localhost:3080/eppm-go/admin/account/edit/232122'. The page content is a form for updating employee data. The left sidebar has a blue header 'eppmgo!' and a navigation menu with items like 'Dashboard', 'Daftar Pekerja' (selected), 'Workbook', 'Assessment', 'Data Ruangkerja', 'Raport', and 'Logout'. Below this is a 'DOCUMENTATION' section with links to 'Getting started', 'Foundation', and 'Components'. The main form area has the following fields:

No. Pekerja*	232122
Direktorat	SDM
Nama Pekerja*	Ema Ainun Novia
Username*	ema
Password*	*****
Email	ema@email.com
Role	(This field is partially visible)

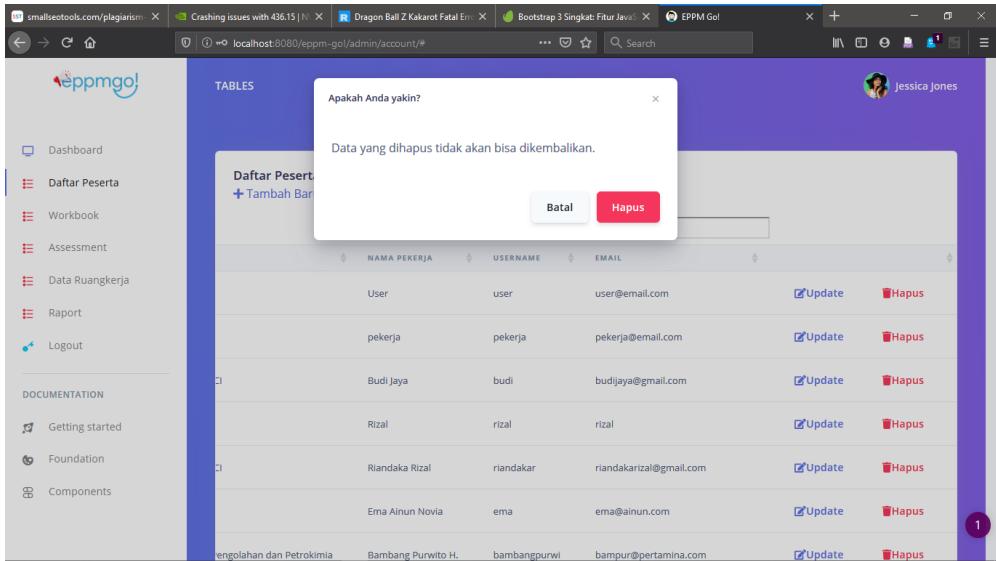
Gambar 68 Halaman Form Update

Setelah ada di halaman form *update*, maka kita tinggal mengubah data yang perlu diubah pada *textbox textbox* yang dirasa perlu. Setelah kita selesai mengubah beberapa *field* maka langsung disimpan saja hasil perubahannya dengan mengklik tombol simpan di bawah halaman form *update*, maka akan muncul notif kalau datanya telah berhasil kita *update* seperti yang ditunjukkan pada gambar 69 di bawah ini.



Gambar 69 Halaman Form Update

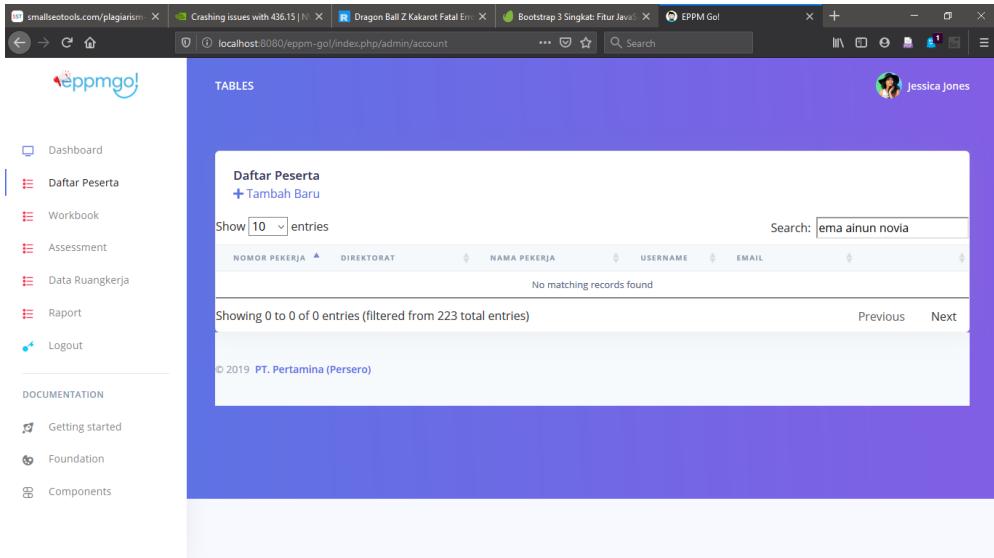
Lalu untuk menghapus data dari pengguna Ema Ainun Novia, tinggal kita klik tombol hapus di samping tombol *update*, maka akan muncul sebuah jendela kecil atau modal yang berguna untuk melakukan konfirmasi sebelum datanya dihapus. Sehingga kita tidak perlu takut atau kaget ketika tombol hapus tidak sengaja tertekan. Seperti yang ditunjukkan pada gambar 70 di bawah ini.



Gambar 70 Modal Tombol Hapus

Jika kita ingin benar benar menghapus data pengguna tersebut, maka langsung saja kita klik hapus. Jika tidak ingin, kita tinggal batalkan saja dengan menekan tombol batal. Karena kita ingin menghapus data pengguna tersebut maka kita tekan tombol hapus. Untuk memastikan bahwa datanya telah dihapus, kita akan mengujinya dengan mencari nama pengguna di kolom pencarian yang disediakan di atas tabel.

Karena data tersebut telah dihapus dari tabel dan juga *database*, maka tidak ditemukan hasil pencarian dari nama penggunanya. Seperti yang dapat kita lihat pada gambar 71 di bawah ini.



Gambar 71 Halaman Daftar Pekerja

Nah sangat mudahkan? Karena menu daftar pekerja telah selesai dibuat kita akan lanjut membuat untuk menu selanjutnya, yaitu menu *assessment* yang bertujuan untuk memberikan penilaian dari pelatihan yang sedang dijalani. Untuk menu *assessment* sendiri, dalam penilaianya akan ada pembobotan nilai sehingga kita juga harus mengatur fungsi matematikanya, agar hasil penilaianya sesuai dengan keinginan dan pembobotan nilai yang kita berikan

Untuk menu *assessment*, dalam memberikan sebuah penilaian pastinya diperlukan sebuah standar kelulusan sehingga hanya nilai yang memenuhi nilainya saja yang mendapat predikat lulus. Oleh karena itu, apabila rata rata nilai setelah dibobotkan lebih dari 70 maka akan mendapat predikat lulus. Bila nilai yang didapatkan setelah dibobotkan ternyata kurang dari 70 maka akan mendapat predikat tidak lulus.

## 2.4. CRUD Menu *Assessment*

Seperti nama untuk menu ini, *assessment* yaitu yang berarti penilaian. Jadi pada menu ini akan diberikan penilaian dari modul pelatihan yang telah diselesaikan, sehingga peserta pelatihan bisa mendapatkan nilai dari apa yang telah mereka kerjakan.

Sesuai dengan apa yang telah dijelaskan pada sub-bab sebelumnya, untuk menu *assessment* ini ada pembobotan nilai setiap kriteria penilaianya. Contoh untuk kriteria penilaian A bobotnya 10% dan nilai yang diberikan adalah 80. Jadi nilai untuk penilaian A adalah 0,8, sehingga berapapun nilai yang diperoleh untuk kriteria penilaian A akan dikalikan dengan bobot yang telah kita tentukan untuk penilaianya. Selain itu, ada juga standar kelulusan minimum yang telah ditetapkan yang menjadi acuan agar para peserta pelatihan dapat melampaui standar yang ditetapkan. Standar kelulusannya sendiri minimum rata – rata nilai 70. Bila nilai rata – rata yang diperoleh oleh peserta pelatihan lebih dari 70, maka peserta tersebut lulus. Dan sebaliknya apabila nilai rata – rata yang diperoleh peserta tersebut kurang dari 70, maka peserta tersebut dianggap tidak lulus pelatihannya. Coba lihat ilustrasi dari gambar 72 di bawah ini.

RATA - RATA	STATUS LULUS
78.3	Lulus
81.65	Lulus
83.95	Lulus
68.85	Tidak Lulus
77.5	Lulus
78.25	Lulus
84.4	Lulus
76.6	Lulus
79.15	Lulus
75.75	Lulus

Gambar 72 Ilustrasi Standar Kelulusan

Baiklah untuk mari kita bagi kriteria penilaian untuk *assessment* dan juga bobot penilaian dari kriteria penilaianya. Kriteria penilaianya dibagi menjadi 2 yaitu teori dan praktik

- a. Kriteria penilaian pertama adalah, definisi. Dapat menjelaskan definisi tentang suatu teori dengan bobot nilai 10%
- b. Kriteria penilaian kedua adalah, tujuan. Dapat menjelaskan tujuan dari suatu teori dengan bobot nilai 10%

- c. Kriteria penilaian ketiga adalah, pelaksanaan. Dapat menjelaskan pelaksanaan tentang teori tersebut di dunia nyata dengan bobot nilai 20%
- d. Kriteria penilaian keempat adalah, contoh praktik. Dapat menjelaskan contoh praktik dari teori yang diterapkan dengan bobot nilai 15%
- e. Kriteria penilaian kelima adalah, analisis. Dapat menjelaskan analisis dari suatu permasalahan yang dialami dengan bobot nilai 20%
- f. Kriteria penilaian keenam adalah, solusi. Dapat menjelaskan solusi dari suatu permasalahan yang dialami dengan bobot nilai 25%

Dengan ini bobot nilai untuk teori adalah 40% dan bobot nilai untuk praktik adalah 60%. Karena bobot nilai dan juga standar kelulusan telah kita definisikan, langsung saja kita membuat *file* Assessment.php pada direktori controller di dalam folder admin. Lalu kita isikan *file* Assessment.php tersebut dengan isian sebagai berikut.

```
<?php
defined('BASEPATH') OR exit ('No direct script access allowed');

class Assessment extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        $this->load->model("assessment_model"); //Load model workbook
        $this->load->library('form_validation'); //Load Library form validation
        $this->load->helper(array('form', 'url'));
        if($this->assessment_model->is_role() != "admin"){
            redirect("index.php/login");
        }
    }
}
```

Gambar 73 Controller Assessment.php

Disini kita membuat sebuah class dengan nama Assessment dengan menggunakan *library* dari *CI\_Controller*. Lalu kita gunakan function *\_\_construct()* yang disediakan *CodeIgniter* dengan me-*load* models *assessment\_model* dan juga *library* dari form validasi. Lalu jika ada pengguna yang melakukan *login* maka function *is\_role()* yang di-*load* dari *assessment\_model* mengecek apakah *role* pengguna tersebut adalah bukan admin, jika benar maka arahkan kembali ke halaman *login*.

```

public function index()
{
    $data["assessment"] = $this->assessment_model->getAll(); //ambil data dari model
    // print_r('tes'); die;
    $this->load->view("admin/assessment/list_assessment", $data); //load view data model ke workbook
}

public function get_stage()
{
    $data['data'] = $this->assessment_model->get_data_stage();
    $this->load->view('admin/assessment/new_assessment', $data);
}

public function get_modul()
{
    $id = $this->input->post('id');
    $data = $this->assessment_model->get_data_modul($id);
    echo json_encode($data);
}

public function add()
{
    $data["assessment"] = $this->assessment_model->get_data_stage();
    $this->load->view("admin/assessment/new_assessment", $data); //Load isi form workbook
}

```

Gambar 74 Controller Assessment.php

Setelah selesai membuat *line* di atas, maka kita lanjut untuk mengetik isian seperti yang ada di gambar 74 di atas. Buat function index() dengan mengambil data dari assessment\_model menggunakan function getAll() lalu tampilkan halaman *assessment* dengan membawa data yang telah diambil dari assessment\_model. Selanjutnya kita membuat function get\_stage() dengan mengambil data kembali dari assessment\_model melalui function get\_data\_stage() lalu tampilkan halaman form tambah baru *assessment* dengan membawa datanya dari assessment\_model. Lalu kita kembali membuat function get\_modul() dengan mengambil id dari *stage* dan ambil data modul dari assessment\_model sesuai dengan id *stage* yang dipilih. Jangan lupa membuat function add() dengan isian yang sama seperti function get\_stage() yang telah dijelaskan sebelumnya

```

public function aksi_add()
{
    // print_r('tes'); die;
    $this->session->set_flashdata('Sukses', 'Data Anda Berhasil Disimpan'); //pesan berhasil
    $this->assessment_model->save(); //objek model redirect
    redirect('index.php/admin/assessment');
}

public function edit($id=null)
{
    $data["assessment"] = $this->assessment_model->getById($id); //mengambil data berdasarkan id
    if (!$data["assessment"]) show_404(); // jika tidak ada show error
    $this->load->view("admin/assessment/edit_assessment", $data); //Load edit form workbook
}

```

Gambar 75 Controller Assessment.php

Setelah membuat function add() tambahkan kembali function aksi\_add() dengan menggunakan function save() yang diambil melalui assessment\_model. Lanjut kembali membuat function edit() dengan mengambil data berdasarkan id melalui assessment\_model menggunakan function getById(). Lalu tampilkan halaman form edit *assessment* dengan membawa data yang sebelumnya telah diambil menggunakan function getById().

```

function aksi_edit()
{
    $id = $this->input->post('assessment_id');

    $nilai_definisi = $this->input->post('nilai_definisi');
    $nilai_definisi *= 0.1;
    $nilai_tujuan = $this->input->post('nilai_tujuan');
    $nilai_tujuan *= 0.1;
    $nilai_langkah = $this->input->post('nilai_langkah');
    $nilai_langkah *= 0.2;
    $nilai_contoh = $this->input->post('nilai_contoh');
    $nilai_contoh *= 0.15;
    $nilai_analisis = $this->input->post('nilai_analisis');
    $nilai_analisis *= 0.2;
    $nilai_solusi = $this->input->post('nilai_solusi');
    $nilai_solusi *= 0.25;
    $total_nilai = $nilai_definisi + $nilai_tujuan + $nilai_langkah + $nilai_contoh + $nilai_analisis + $nilai_solusi;
    $status_lulus = $nilai_definisi + $nilai_tujuan + $nilai_langkah + $nilai_contoh + $nilai_analisis + $nilai_solusi;

    if($total_nilai >= 70){
        $status_lulus = "Lulus";
    }else{
        $status_lulus = "Tidak Lulus";
    }
}

```

Gambar 76 Controller Assessment.php

Pada gambar 76 di atas, kita membuat function aksi\_edit() kita mengambil id dari data yang kita pilih lalu mendefinisikan kriteria penilaian ke dalam variabelnya masing masing, jangan lupa untuk memberikan bobot nilai pada setiap variabel dari masing masing kriteria penilaian. Jika sudah maka keseluruhan nilanya akan kita jumlahkan, dan diberi sebuah pengkondisian bila nilai lebih dari sama dengan 70, maka mendapatkan predikat lulus. Bila tidak maka mendapat predikat tidak lulus. Pada *line* sintaks di atas, kita terlebih dahulu mendefinisikan variabel penilaian dan juga bobot nilainya jika sudah selesai maka tinggal menyimpan dan memasukkan datanya ke dalam *database* pada tabel assessment. Dan jangan lupa untuk membuat function delete() untuk menghapus data yang dipilih. Seperti yang ditunjukkan oleh gambar 77 berikut ini.

```

        }

        $data = array (
            'assessment_id' => $this->input->post('assessment_id'),
            'stage_id' => $this->input->post('stage_id'),
            'modul_id' => $this->input->post('modul_id'),
            'employee_id' => $this->input->post('employee_id'),
            'nilai_definisi' => $this->input->post('nilai_definisi'),
            'nilai_tujuan' => $this->input->post('nilai_tujuan'),
            'nilai_langkah' => $this->input->post('nilai_langkah'),
            'nilai_contoh' => $this->input->post('nilai_contoh'),
            'nilai_analisis' => $this->input->post('nilai_analisis'),
            'nilai_solusi' => $this->input->post('nilai_solusi'),
            'total_nilai' => $total_nilai,
            'status_lulus' => $status_lulus
        );

        // print_r($data); die;
        $this->assessment_model->update($id,$data);
        $this->session->set_flashdata('success', 'Data Anda Berhasil Diupdate');
        redirect('index.php/admin/assessment');

    }

    public function delete($id=null)
    {
        if (!isset($id)) show_404();

        if ($this->assessment_model->delete($id)){
            redirect('index.php/admin/assessment');
        }
    }
}

```

Gambar 77 Controller Assessment.php

Setelah menyelesaikan isian seperti di atas, maka selesai sudah langkah untuk membuat *file* controller Assessment.php dan akan dilanjutkan untuk membuat *file* models dengan nama Assessment\_model.php pada direktori models di dalam folder admin.

Jika sudah dibuat, maka langsung saja kita mendefinisikan class Assessment\_model dengan menggunakan *library* dari CI\_Model lalu kita definisikan juga variabel variabel yang akan digunakan untuk menu *assessment*. Setelahnya buat kembali function is\_role() yang berguna untuk mengetahui *role* dari setiap pengguna yang akan melakukan *login*. Lanjut lagi kita akan mengambil seluruh data dari tabel *assessment* pada *database* dengan membuat function getAll()

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Assessment_model extends CI_Model
{
    private $_table = "assessment";

    public $assessment_id;
    public $stage_id;
    public $modul_id;
    public $employee_id;
    public $nilai_definisi;
    public $nilai_tujuan;
    public $nilai_langkah;
    public $nilai_contoh;
    public $nilai_analisis;
    public $nilai_solusi;
    public $total_nilai;

    //fungsi cek Level
    function is_role()
    {
        return $this->session->userdata('role');
    }

    public function getAll()
    {
        $this->db->select('assessment.*', 'account.employee_name as yaww');
        $this->db->join('account', 'assessment.employee_id = account.employee_id');
        $this->db->from('assessment');
        return $this->db->get()->result();
        // print_r('tes'); die;
    }
}
```

Gambar 78 Models Assessment\_model.php

Lanjut lagi kita membuat function getById() dengan mengambil hanya id yang kita pilih pada halaman menu *assessment*. Langkah selanjutnya adalah membuat function get\_data\_stage() yang berfungsi untuk mengambil data *stage* dari tabel *stage* pada *database*. Dan kita buat juga function get\_data\_modul() yang berfungsi untuk mengambil data modul berdasarkan *stage* yang kita pilih nanti di halaman menu *assessment*.

Dapat dilihat disini, bahwa kedua function get\_data\_stage() dan get\_data\_modul() ini saling berhubungan karena data modul baru akan diambil bila sebelumnya telah memilih *stage*-nya.

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Assessment_model extends CI_Model
{
    private $_table = "assessment";

    public $assessment_id;
    public $stage_id;
    public $modul_id;
    public $employee_id;
    public $nilai_definisi;
    public $nilai_tujuan;
    public $nilai_langkah;
    public $nilai_contoh;
    public $nilai_analisis;
    public $nilai_solusi;
    public $total_nilai;

    //fungsi cek level
    function is_role()
    {
        return $this->session->userdata('role');
    }

    public function getAll()
    {
        $this->db->select('assessment.*', 'account.employee_name as yaww');
        $this->db->join('account', 'assessment.employee_id = account.employee_id');
        $this->db->from('assessment');
        return $this->db->get()->result();
        // print_r('tes'); die;
    }
}
```

Gambar 79 Models Assessment\_model.php

Nah kita teruskan untuk membuat function save() yang berfungsi untuk menyimpan data yang kita isikan pada form *input assessment* ke dalam *database*. Sama seperti function aksi\_edit() di file controller Assessment.php terlebih dahulu kita definisikan variabel untuk setiap kriteria penilaian dan juga pembobotannya. Jika sudah diberikan pengkodisian bila nilai lebih dari sama dengan 70 akan mendapatkan predikat lulus, dan bila kurang dari 70 akan mendapat predikat tidak lulus.

```
public function save()
{
    $nilai_definisi = $this->input->post('nilai_definisi');
    $nilai_definisi *= 0.1;
    $nilai_tujuan = $this->input->post('nilai_tujuan');
    $nilai_tujuan *= 0.1;
    $nilai_langkah = $this->input->post('nilai_langkah');
    $nilai_langkah *= 0.2;
    $nilai_contoh = $this->input->post('nilai_contoh');
    $nilai_contoh *= 0.15;
    $nilai_analisis = $this->input->post('nilai_analisis');
    $nilai_analisis *= 0.2;
    $nilai_solusi = $this->input->post('nilai_solusi');
    $nilai_solusi *= 0.25;
    $total_nilai = $nilai_definisi + $nilai_tujuan + $nilai_langkah + $nilai_contoh + $nilai_analisis + $nilai_solusi;
    $status_lulus = $nilai_definisi + $nilai_tujuan + $nilai_langkah + $nilai_contoh + $nilai_analisis + $nilai_solusi;

    if($total_nilai >= 70){
        $status_lulus = "Lulus";
    }else{
        $status_lulus = "Tidak Lulus";
    }
}
```

Gambar 80 Models Assessment\_model.php

Jika telah didefinisikan variabel variabelnya, langkah selanjutnya adalah membuat sintaks dengan isian yang akan memasukkan data data yang telah diisi pada form *input assessment* ke dalam *database*. Seperti yang ditunjukkan oleh gambar 81 di bawah ini.

```
$data = array (
    "stage_id" => $this->input->post("stage_id"),
    "modul_id" => $this->input->post("modul_id"),
    "employee_id" => $this->input->post("employee_id"),
    "nilai_definisi" => $this->input->post("nilai_definisi"),
    "nilai_tujuan" => $this->input->post("nilai_tujuan"),
    "nilai_langkah" => $this->input->post("nilai_langkah"),
    "nilai_contoh" => $this->input->post("nilai_contoh"),
    "nilai_analisis" => $this->input->post("nilai_analisis"),
    "nilai_solusi" => $this->input->post("nilai_solusi"),
    "total_nilai" => $total_nilai,
    "status_lulus" => $status_lulus
);

$this->db->insert("assessment", $data);
}
```

Gambar 81 Models Assessment\_model.php

Masih semangat? Lanjut saja kita membuat function *update* yang berfungsi untuk mengubah suatu data yang kita inginkan dengan mengambil id dari data yang dipilih. Jika sudah selesai mengubah datanya, maka data yang baru saja diubah akan dimasukkan ke dalam *database*. Begitupun dengan function *delete()* yang akan mengambil id dari data yang ingin kita hapus, jika benar benar ingin dihapus maka data tersebut akan terhapus pada halaman menu *assessment* dan juga *database*-nya. Seperti yang ditunjukkan pada gambar 82 di bawah ini.

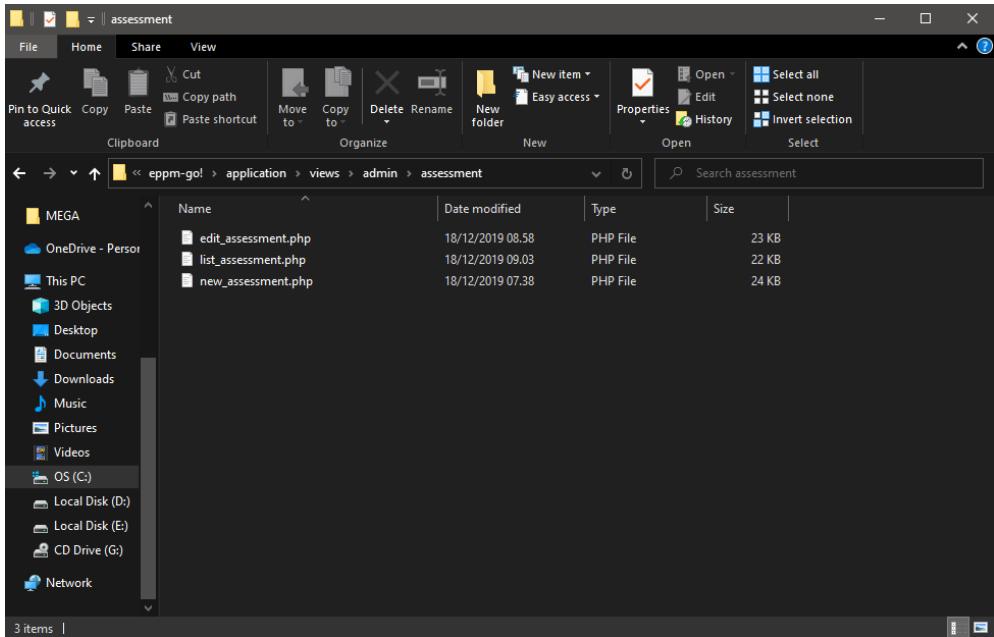
```
public function update($id, $data)
{
    $this->db->where('assessment_id',$id);
    $this->db->update("assessment", $data);
}

public function delete($id)
{
    // $this->_deleteImage($id);
    return $this->db->delete($this->_table, array("assessment_id" => $id));
}
```

Gambar 82 Models Assessment\_model.php

Kalau function delete() sudah selesai kita buat, maka selesai sudah langkah langkah pembuatan dari models Assessment\_model.php. Mari kita istirahat sejenak, coba renungkan setelah sampai sejauh ini apa saja yang telah teman teman dapatkan saat membaca tutorial ini? Apakah ada manfaatnya untuk pribadi teman teman semua? Ataukah sudah sejauh ini teman teman masih tidak mudeng dari apa yang sudah dibaca? Coba renungkan!

Baiklah dengan telah selesainya models Assessment\_model.php maka langkah terakhir dalam membuat menu *assessment* adalah bagian views. Untuk itu silakan buat terlebih dahulu folder *assessment* dan langsung membuat *file* dengan nama list\_assessment.php, new\_assessment.php, dan juga edit\_assessment.php di dalamnya pada direktori views di dalam folder admin.



Gambar 83 File Views Assessment

Jika sudah dibuat ketiga *file* tersebut, lanjut untuk mengisi *file* list\_assessment.php. Kembali saya ingatkan, disini saya menggunakan *bootstrap argon dashboard* dan hanya memodifikasinya sesuai dengan kebutuhan pada tutorial kali ini. Kembali ke topik, kita buat sebuah tabel untuk menampung atau mewadahi data yang akan kita ambil dari *database* dan isikan sesuai dengan gambar 84 di bawah ini.

```
<!-- table -->
<div class="row">
  <div class="col">
    <div class="card shadow">
      <div class="card-header border-0">
        <h3 class="mb-0">Assessment</h3>
        <a href=<?php echo site_url('admin/assessment/add') ?>><i class="fas fa-plus"></i> Tambah Baru</a>
      </div>
      <div class="table-responsive">
        <table class="table align-items-center table-flush">
          <thead class="thead-light">
            <tr>
              <th scope="col">ID Stage</th>
              <th scope="col">ID Modul</th>
              <th scope="col">Nama Pekerja</th>
              <!-- <th scope="col">Nilai Definisi</th>
              <th scope="col">Nilai Tujuan</th>
              <th scope="col">Nilai Langkah</th>
              <th scope="col">Nilai Contoh</th>
              <th scope="col">Nilai Analisis</th>
              <th scope="col">Nilai Solusi</th> -->
              <th scope="col">Total Nilai</th>
              <th scope="col">Status Kelulusan</th>
              <th scope="col"></th>
            </tr>
          </thead>
```

Gambar 84 Views list\_assessment.php

Dapat dilihat pada gambar 84, ada tabel yang di *comment* – *comment* membuat baris atau *line* yang dipilih tidak akan memengaruhi program – ini dikarenakan sebelumnya baik untuk menampilkan keseluruhan kolom tapi lebih baik lagi hanya menampilkan kolom yang dibutuhkan saja.

Kembali lagi kita menggunakan perulangan *foreach* untuk menampilkan datanya sehingga keseluruhan datanya dapat ditampilkan tidak hanya satu data saja yang ditampilkan, seperti yang ditunjukkan pada gambar 85 di bawah ini tentang penggunaan dari perulangan *foreach*.

```
<tbody>
    <?php foreach ($assessment as $assessment): ?>
    <tr>
        <td width="150">
            <?php echo $assessment->stage_id ?>
        </td>
        <td>
            <?php echo $assessment->modul_id ?>
        </td>
        <td>
            <?php echo $assessment->yaww ?>
        </td>
        <!-- <td>
            <?php echo $assessment->nilai_definisi ?>
        </td>
        <td>
            <?php echo $assessment->nilai_tujuan ?>
        </td>
        <td>
            <?php echo $assessment->nilai_Langkah ?>
        </td>
        <td>
            <?php echo $assessment->nilai_contoh ?>
        </td>
        <td>
            <?php echo $assessment->nilai_analisis ?>
        </td>
        <td>
            <?php echo $assessment->nilai_solusi ?>
        </td> -->
        <td>
            <?php echo $assessment->total_nilai ?>
        </td>
```

Gambar 85 Views list\_assessment.php

Lalu kita kembali membuat tombol untuk *update* dan *hapus* untuk mengubah data yang kita pilih dan menghapus data yang kita pilih, kita buat tombol tombol tersebut di samping datanya. Kembali saya ingatkan, untuk menampilkan tabel pada setiap halaman kita menggunakan DataTables.

```
<td width="250">
    <a href="=php echo site_url('admin/assessment/edit/'.$assessment-&gt;assessment_id)?" class="btn btn-small"><i class="fas fa-edit"></i>Update</a>
    <a onclick="deleteConfirm('=php echo site_url('admin/assessment/delete/'.$assessment-&gt;assessment_id)?'" href="#" class="btn btn-small text-danger"><i class="fas fa-trash"></i>Hapus</a>
</td>
```

Gambar 86 Views list\_assessment.php

Karena, masih menggunakan DataTables jadi kita harus menggunakan sintaks jQuery untuk menggunakan *resources* dari DataTables. Setelah itu, untuk sintaks jQuery dari DataTables kita taruh dibawah setelah sintaks pembuatan tabel. Jangan lupa untuk menyisipkan tag <script> karena masih satu bagian dari JavaScript. Selain itu, karena kita menggunakan *resources* dari luar ada beberapa cara yang bisa digunakan, yaitu dengan cara:

1. Mengunduh *library*-nya dan disimpan pada folder *assets* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini akan menggunakan *local resource* dari PC atau laptop kita.

- Menyalin *link CDN* (*Content Delivery Network*) yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini hanya memerlukan akses internet. *Link CDN* DataTables untuk CSS adalah //cdn.datatables.net/1.10.20/css/jquery.dataTables.min.css dan *link CDN* DataTables untuk JavaScript adalah //cdn.datatables.net/1.10.20/js/jquery.dataTables.min.js

```
<script>$(document).ready( function () {
|   $('table').DataTable();
} );
</script>
```

Gambar 87 Views list\_assessment.php

Untuk tombol hapus, kita harus selalu menggunakan konfirmasi sebelum dilakukannya penghapusan data. Ini berguna untuk mencegah dan meminimalisir terjadinya penghapusan data secara tidak sengaja, dan kita menggunakan modal untuk konfirmasinya seperti yang ditunjukkan gambar 88 di bawah ini

```
<script>
function deleteConfirm(url){
  $('#btn-delete').attr('href', url);
  $('#deleteModal').modal();
}
</script>
```

Gambar 88 Views list\_assessment.php

Maka selesai sudah *file* list\_assessment.php yang kita buat, langkah selanjutnya adalah membuat *file* new\_assessment.php yang berfungsi untuk memberikan sebuah form penilaian untuk disimpan dan dimasukkan ke dalam *database*. Oke langsung saja isikan *file* tersebut dengan isian seperti yang ditunjukkan pada gambar 89 di bawah ini.

```
<div class="card mb-3">
    <div class="card-header">
        <a href="=php echo site_url('admin/assessment/') ?&gt;"&gt;&lt;i class="fas fa-arrow-left"&gt;&lt;/i&gt; Kembali&lt;/a&gt;
    &lt;/div&gt;
    &lt;div class="card-body"&gt;
        &lt;form action="<?=base_url();?&gt;index.php/admin/assessment/aksi_add" method="post" enctype="multipart/form-data"&gt;
            &lt;!-- &lt;input type="hidden" name="assessment_id" value="<?=php echo $assessment-&gt;assessment_id?&gt;" /&gt; --&gt;

            &lt;div class="form-group"&gt;
                &lt;label for="Stage"&gt;Stage*&lt;/label&gt;
                &lt;select class="form-control" name="stage_id" placeholder="Stage" id="stage_id" required&gt;
                    &lt;option&gt; - Pilih Stage -&lt;/option&gt;
                    &lt;?php
                        foreach ($assessment-&gt;result() as $baris) {
                            echo "&lt;option value='".$baris-&gt;stage_id."'&gt;".$baris-&gt;stage_name."&lt;/option&gt;";
                        }
                    &gt;
                &lt;div class="invalid-feedback"&gt;&lt;?php echo form_error('stage_id')?&gt;
                &lt;/div&gt;
                &lt;/select&gt;
            &lt;/div&gt;

            &lt;div class="form-group"&gt;
                &lt;label for="Modul"&gt;Modul*&lt;/label&gt;
                &lt;select class="form-control" name="modul_id" placeholder="Modul" id="modul_id" required&gt;
                    &lt;option value=""&gt; - Pilih Modul -&lt;/option&gt;
                    &lt;div class="invalid-feedback"&gt;&lt;?php echo form_error('modul_id')?&gt;
                    &lt;/div&gt;
                    &lt;/select&gt;
            &lt;/div&gt;
        &lt;/form&gt;
    &lt;/div&gt;
</pre
```

Gambar 89 Views new\_assessment.php

Disini terdapat 2 *textbox* yang berbentuk form *dropdown*. Karena *field stage* dan juga modul ini saling berhubungan maka kita akan menggunakan AJAX untuk membuat *dropdown chained* atau sebuah form *dropdown* yang saling berkaitan.

```

<div class="form-group">
    <label for="No. Pekerja">No. Pekerja*</label>
    <input class="form-control" type="text" name="employee_id" placeholder="Nomor Pekerja" required>
    <div class="invalid-feedback"><?php echo form_error('employee_id')?>
    </div>
</div>

<div class="form-group">
    <label for="Nilai Definisi">Nilai Untuk Definisi</label>
    <input class="form-control" type="text" name="nilai_definisi" placeholder="Nilai Definisi" required>
    <div class="invalid-feedback"><?php echo form_error('nilai_definisi')?>
    </div>
</div>

<div class="form-group">
    <label for="Nilai Tujuan">Nilai Untuk Tujuan</label>
    <input class="form-control" type="text" name="nilai_tujuan" placeholder="Nilai Tujuan" required>
    <div class="invalid-feedback"><?php echo form_error('nilai_tujuan')?>
    </div>
</div>

<div class="form-group">
    <label for="nilai_langkah">Nilai Untuk Langkah-Langkah</label>
    <input class="form-control" type="text" name="nilai_langkah" placeholder="Nilai Langkah" required>
    <div class="invalid-feedback"><?php echo form_error('nilai_langkah')?>
    </div>
</div>

```

Gambar 90 Views new\_assessment.php

Sama saja seperti pada menu *workbook* dan juga daftar pekerja yang sebelumnya telah dibuat, disini kita juga membuat sebuah *textbox* untuk dapat mengisi formnya. Dapat dilihat pada gambar 90 ini, kita sudah membuat *textbox* untuk *field* No. pekerja, dan juga kriteria – kriteria penilaian yang sudah dibahas sebelumnya.

```

<div class="form-group">
    <label for="nilai_contoh">Nilai Untuk Contoh Penerapan</label>
    <input class="form-control <?php echo form_error('nilai_contoh')? 'is-invalid': ''?>" type="text" name="nilai_contoh" placeholder="Nilai Contoh" required/>
    <div class="invalid-feedback"><?php echo form_error('nilai_contoh')?>
    </div>
</div>

<div class="form-group">
    <label for="nilai_analisis">Nilai Untuk Analisis</label>
    <input class="form-control <?php echo form_error('nilai_analisis')? 'is-invalid': ''?>" type="text" name="nilai_analisis" placeholder="Nilai Analisis" required/>
    <div class="invalid-feedback"><?php echo form_error('nilai_analisis')?>
    </div>
</div>

<div class="form-group">
    <label for="nilai_solusi">Nilai Untuk Solusi</label>
    <input class="form-control <?php echo form_error('nilai_solusi')? 'is-invalid': ''?>" type="text" name="nilai_solusi" placeholder="Nilai Solusi" required/>
    <div class="invalid-feedback"><?php echo form_error('nilai_solusi')?>
    </div>
</div>

<input class="btn btn-success" type="submit" name="btn" value="Simpan" />
</form>

```

Gambar 91 Views new\_assessment.php

Jika semua *textbox* dari seluruh *field* yang dibutuhkan telah selesai dibuat, jangan lupa untuk memberikan tombol simpan untuk dapat menyimpan isian isian pada *textbox* yang akan diisi nanti ke dalam *database* kita. Sempat disinggung pada AJAX sebelumnya, karena *stage* dan modul itu saling berhubungan satu sama lain maka untuk memfasilitasi agar dapat membuat *dropdown chained* yaitu dengan AJAX. Sama seperti *JavaScript*, sintaks AJAX ditempatkan pada bagian bawah sebelum tag </body> dan kita diwajibkan untuk memberikan *link resources* CDN agar dapat menggunakan AJAX. *Link* CDN untuk AJAX sendiri ditempatkan di atas HTML pada bagian tag <head> dengan *link Content Delivery Network* berikut ini <https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js>.

Kalau lebih suka menggunakan AJAX dengan mengunduh *resources*-nya maka unduh terlebih dahulu dan taruh pada direktori *assets*. Berikut sintaks AJAX yang berada pada bawah tag </body> seperti yang ditunjukkan pada gambar 92 berikut ini.

```
<script> $(document).ready(function(){
    $('#stage_id').change(function(){
        // console.log('tes');
        var id=$(this).val();
        // console.log(id);
        $.ajax({
            url : "<?php echo base_url();?>index.php/admin/assessment/get_modul",
            method : "POST",
            data : {id: id},
            async : false,
            dataType : 'json',
            success: function(data){
                var html = '';
                var i;
                for(i=0; i<data.length; i++){
                    console.log(data)
                    html += '<option value='+data[i].modul_id+'>'+data[i].modul_name+'</option>'
                }
                $('#modul_id').html(html);
                //nu bener modul_id
            }
        });
    });
});</script>
```

Gambar 92 Sintaks AJAX

Seperti yang dapat kita lihat pada gambar 92 di atas, disinilah pembuktian bahwa untuk *field stage* dan modul itu saling berhubungan. Karena pada setiap *stage* memiliki modul yang berbeda – beda. Sehingga apabila kita memilih *stage* 1 maka kita juga hanya dapat memilih modul modul yang terdaftar pada *stage* 1.

Selesai sudah tahapan untuk membuat *file* new\_assessment.php, tinggal satu langkah terakhir yaitu membuat *file* edit\_assessment.php yang kurang lebih memiliki langkah yang sama. Sehingga ini akan lebih memudahkan kita dalam membuat *file* edit\_assessment.php

```
<!-- Table -->
<div class="card mb-3">
  <div class="card-header">
    <a href="php echo site_url('admin/assessment/') ?&gt;"<i class="fas fa-arrow-left">
```

Gambar 93 Views edit\_assessment.php

Dapat dilihat ini adalah langkah yang sama seperti menu menu yang sebelumnya telah kita buat. Tinggal membuat *textbox* untuk form dan panggil datanya sesuai dengan id yang dipilih sebelumnya. Mulai dari sini kita buat terlebih dahulu *textbox* sesuai dengan *field* yang dibutuhkan, seperti yang ditunjukkan pada gambar 94 sampai gambar 96 di bawah ini

```
<div class="form-group">
    <label for="name">No. Pekerja*</label>
    <input class="form-control <?php echo form_error('employee_id') ? 'is-invalid':'' ?>" type="text" name="employee_id" min="0" placeholder="No. Pekerja" value="<?php echo $assessment->employee_i
    <div class="invalid-feedback">
        <?php echo form_error('employee_id') ?>
    </div>
</div>

<div class="form-group">
    <label for="nilai_definisi">Nilai Untuk Definisi</label>
    <input class="form-control <?php echo form_error('nilai_definisi') ? 'is-invalid':'' ?>" type="text" name="nilai_definisi" min="0" placeholder="Nilai Definisi" value="<?php echo $assessment->nila
    <div class="invalid-feedback">
        <?php echo form_error('nilai_definisi') ?>
    </div>
</div>
```

Gambar 94 Views edit\_assessment.php

```
<div class="form-group">
    <label for="name">No. Pekerja*</label>
    <input class="form-control <?php echo form_error('employee_id') ? 'is-invalid':'' ?>" type="text" name="employee_id" min="0" placeholder="No. Pekerja" value="<?php echo $assessment->employee_i
    <div class="invalid-feedback">
        <?php echo form_error('employee_id') ?>
    </div>
</div>

<div class="form-group">
    <label for="nilai_definisi">Nilai Untuk Definisi</label>
    <input class="form-control <?php echo form_error('nilai_definisi') ? 'is-invalid':'' ?>" type="text" name="nilai_definisi" min="0" placeholder="Nilai Definisi" value="<?php echo $assessment->nila
    <div class="invalid-feedback">
        <?php echo form_error('nilai_definisi') ?>
    </div>
</div>
```

Gambar 95 Views edit\_assessment.php

```

        <div class="form-group">
    <label for="nilai_analisis">Nilai Untuk Analisis</label>
    <input class="form-control" type="text" name="nilai_analisis" min="0" placeholder="Nilai Analisis" value=<?php echo $assessment->nila
    <div class="invalid-feedback">
        <?php echo form_error('nilai_analisis') ?>
    </div>
    </div>
    <div class="form-group">
    <label for="nilai_solusi">Nilai Untuk Solusi</label>
    <input class="form-control" type="text" name="nilai_solusi" min="0" placeholder="Nilai Solusi" value=<?php echo $assessment->nila
    <div class="invalid-feedback">
        <?php echo form_error('nilai_solusi') ?>
    </div>
    </div>
    </div>

    <input class="btn btn-success" type="submit" name="btn" value="Simpan" />
</form>

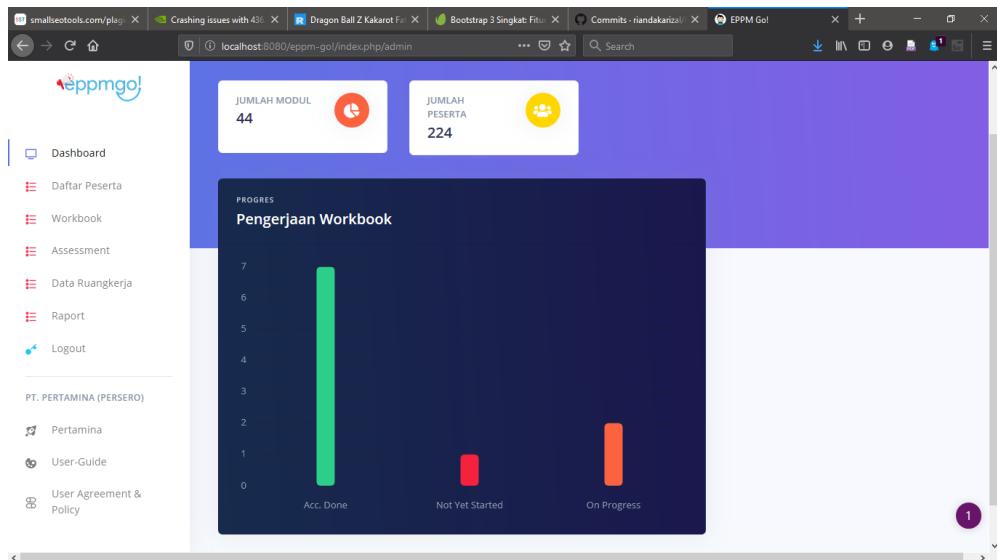
```

Gambar 96 Views edit\_assessment.php

Dengan ini selesailah sudah keseluruhan proses untuk membuat menu *assessment* yaitu controller, models, dan juga views-nya. Karena itu marilah kita coba menu *assessment* ini dengan menggunakan *browser* atau peramban yang menjadi preferensi kita masing – masing.

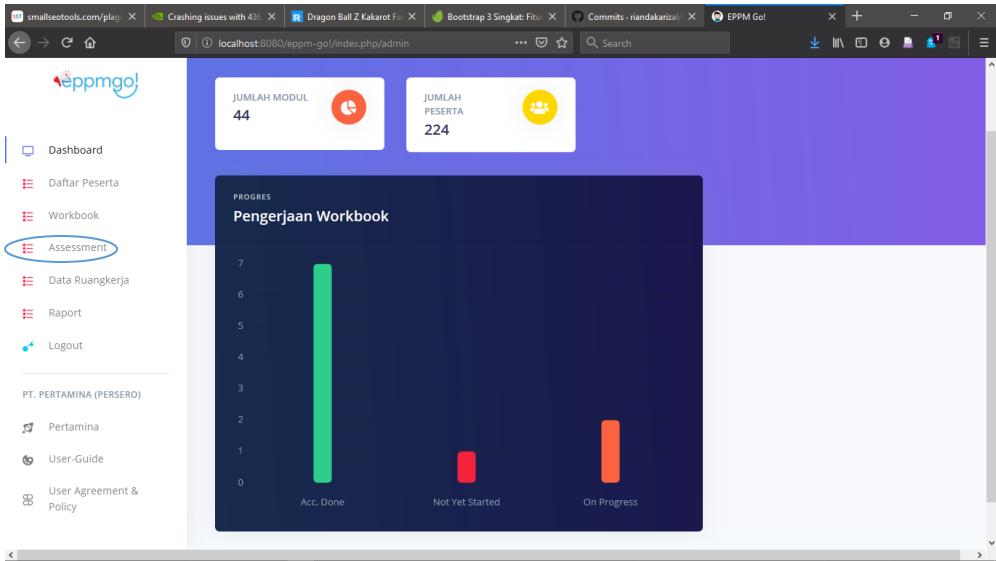
Bukalah *browser*-nya lalu ketikkan `localhost:8080/eppm-go!/admin/assessment` yang akan menuju langsung ke menu *assessment*. Tetapi akan diarahkan ke dalam halaman *login* untuk terlebih dahulu melakukan *login* dengan *username* dan *password* untuk admin. Kenapa hal ini terjadi? Karena bila suatu pengguna walaupun admin tidak diatur penggunaan *session* maka hal ini akan membuat aplikasi *website* yang kita sudah rancang dan kembangkan akan tampak tidak elok, dan terlihat seperti memiliki cacat.

Lanjut bila sudah masuk ke dalam aplikasi EPPM Go! maka akan terlihat seperti yang ditunjukkan pada gambar 97 di bawah ini.



Gambar 97 Halaman Awal Admin

Jika sudah seperti di atas, maka langsung kita memilih menu *assessment* karena kita ingin mencoba apakah sudah berhasil yang baru saja kita buat sebelumnya? Pilih menu *assessment* pada menu bar yang terdapat di samping layar laptop atau PC kalian



Gambar 98 Memilih Menu Assessment

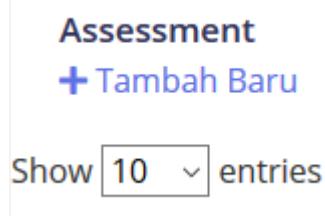
The screenshot shows the 'Assessment' list page. The sidebar on the left is identical to the one in the previous screenshot. The main content area has a title 'Assessment' with a '+ Tambah Baru' button. Below it is a search bar with a dropdown showing 'Show 10 entries'. A table lists ten rows of assessment data. The columns are: ID STAGE, ID MODUL, NAMA PEKERJA, TOTAL NILAI, STATUS KELULUSAN, Update button, and Hapus button. The last row of the table has a circled number '1' next to it. The table rows are:

ID STAGE	ID MODUL	NAMA PEKERJA	TOTAL NILAI	STATUS KELULUSAN	Update	Hapus
1	2.1	admin	78.3	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	2.1	Abdul Karim Amarullah	81.65	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	2.2	Riandaka Rizal	83.95	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	2.2	Riandaka Rizal	68.85	Tidak Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	3.1	admin	77.5	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	2.1	Riandaka Rizal	78.25	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	1.1	Muhammad Fauzan Aristyo	84.4	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	1.1	Riandaka Rizal	76.6	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>
1	6.2	Muhammad Fauzan Aristyo	70.15	Lulus	<input type="button" value="Update"/>	<input type="button" value="Hapus"/>

Gambar 99 Halaman Menu Assessment

Seperti yang dapat dilihat pada gambar 97, untuk halaman awal dari menu *assessment* sudah berhasil berjalan tanpa ada kendala. Datanya dapat terpanggil semua dan juga standar kelulusan yang sudah diterapkan pada sintaks models sebelumnya juga sudah berjalan. Sekali lagi dapat dilihat, bahwa nilai rata – rata menjadi acuan dari status kelulusan masing masing peserta. Apabila nilai rata – rata lebih dari 70 maka lulus, dan jika yang terjadi adalah sebaliknya nilai rata – rata kurang dari 70 maka tidak lulus.

Mari mencoba untuk menambahkan suatu penilaian baru untuk satu pengguna, untuk menambahkan penilaian pada peserta dapat dilakukan dengan cara menekan atau mengklik tombol tambah baru pada pojok kiri atas dari halaman menu *assessment* seperti yang ditunjukkan pada gambar 100 di bawah ini.



Gambar 100 Tombol Tambah Baru

Dengan menekan tombol tersebut, kita akan diarahkan menuju halaman form tambah baru *assessment* yang berguna untuk memasukkan data data penilaian pada form yang telah disediakan. Seperti yang ditunjukkan pada gambar 101 di bawah ini.

The screenshot shows a web-based application interface for managing assessments. On the left, there's a sidebar with a blue header containing the logo 'eppmgo!' and several menu items: Dashboard, Daftar Pekerja, Workbook, Assessment, Data Ruangkerja, Report, Logout, DOCUMENTATION, Getting started, Foundation, and Components. The main content area has a white background with a purple header bar at the top. The header bar contains the URL 'localhost:8080/eppm-go/admin/assessment/add' and a search bar. Below the header, the page title is '← Kembali'. The form itself has several input fields: 'Stage\*' with a dropdown placeholder '- Pilih Stage -'; 'Modul\*' with a dropdown placeholder '- Pilih Modul -'; 'No. Pekerja\*' with a text input placeholder 'Nomor Pekerja'; 'Nilai Untuk Definisi' with a text input placeholder 'Nilai Definisi'; 'Nilai Untuk Tujuan' with a text input placeholder 'Nilai Tujuan'; and 'Nilai Untuk Langkah-Langkah' with a text input placeholder 'Nilai Langkah'. A small red circle with the number '1' is located in the bottom right corner of the form area.

Gambar 101 Halaman Form Tambah Baru

Nah disini, kita dapat membuktikan apakah *dropdown* dari *field stage* dan modul memang berhubungan apa tidak? Klik saja pada *stage* dan pilih *stage 3* maka modul yang dapat dipilih adalah modul yang hanya terdapat pada *stage 3* saja. Seperti yang dapat kita lihat pada gambar 102 berikut ini.

[Kembali](#)

Stage\*

Stage 3

Modul\*

Plan Maintenance and Compliance

Plan Maintenance and Compliance

Resource Loading and Leveling  
Critical Path Optimization  
Design MS Tracking and Review  
Lookback Reviews  
Review Architecture  
Problem Solving and Root Cause Analysis  
Strategic Procurement Involvement  
Commercial Optimization Toolkit  
Commercial Negotiation  
Contract Administration Team  
Change Order Management  
Claims Management  
Integrated Ramp Up Planning

The screenshot shows a user interface for selecting a stage and a module. The 'Stage\*' field is populated with 'Stage 3'. The 'Modul\*' field has a dropdown menu open, displaying a list of items under 'Plan Maintenance and Compliance'. One item, 'Plan Maintenance and Compliance', is highlighted with a blue background, indicating it is the selected option. The list includes various management and optimization tasks.

Gambar 102 Dropdown Stage dan Modul

Lanjut kita mengisi form tersebut sesuai dengan penilaian yang ingin diberikan, jika sudah maka tinggal menyimpannya saja dengan menekan tombol simpan pada bagian bawah dari halaman ini. Lalu data yang baru saja kita masukkan dan simpan sudah dapat dilihat pada halaman awal menu *assessment*. Seperti yang dapat dilihat pada gambar 103 untuk tombol simpan dan juga gambar 104 untuk data yang sudah tersimpan.

Nilai Untuk Definisi  
60

Nilai Untuk Tujuan  
65

Nilai Untuk Langkah-Langkah  
77

Nilai Untuk Contoh Penerapan  
82

Nilai Untuk Analisis  
70

Nilai Untuk Solusi  
69

**Simpan**

Gambar 103 Tombol Simpan Halaman Assessment

ID STAGE	ID MODUL	NAMA PEKERJA	TOTAL NILAI	STATUS KELULUSAN	
3	1.7	Rizal	71.45	Lulus	<a href="#"></a> <a href="#"></a>

Gambar 104 Data yang Sudah Disimpan

Lalu bagaiman jika sebelumnya pada pengisian form di halaman form tambah baru penilaian ada nilai yang salah? Tenang saja, kita dapat merubahnya dengan menekan tombol *update* yang telah disediakan pada bagian samping kanan tabel. Klik tombol tersebut maka kita akan diarahkan menuju halaman form *update*

Bila sudah selesai mengubah data yang dikiranya perlu perbaikan, maka langsung saja disimpan perubahan tersebut dengan menekan kembali tombol simpan yang juga disediakan pada bagian bawah halaman form *update* ini. Dan ternyata setelah ada data yang diubah status lulus dari peserta yang bernama rizal menjadi tidak lulus. Seperti yang ditunjukkan pada gambar 105 dan juga gambar 106 berikut ini.

The screenshot shows a web browser window with the URL `localhost:8080/eppm-go/admin/assessment/edit/17`. The page displays an assessment form for a participant named Rizal. The form contains six input fields with the following values:

Kriteria	Nilai
Nilai Untuk Tujuan	60
Nilai Untuk Langkah	65
Nilai Untuk Contoh Penerapan	77
Nilai Untuk Analisis	82
Nilai Untuk Solusi	70

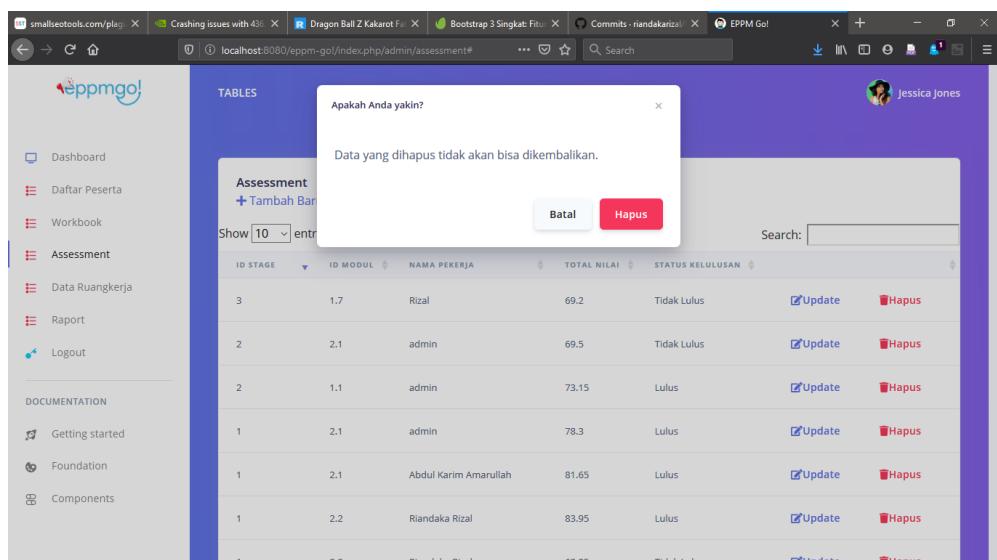
At the bottom of the form is a large green button labeled "Simpan".

Gambar 105 Halaman Form Update

ID STAGE	ID MODUL	NAMA PEKERJA	TOTAL NILAI	STATUS KELULUSAN	Actions
3	1.7	Rizal	69.2	Tidak Lulus	<a href="#">Update</a> <a href="#">Hapus</a>

Gambar 106 Perubahan Data Penilaian

Lalu apabila ternyata peserta atas nama rizal belum saatnya dilakukan *assessment* maka tinggal dihapus saja data dari peserta dengan nama rizal yang diberikan penilaian untuk modul 1.7 pada *stage* 3 dengan menekan tombol hapus berwarna merah di samping tombol *update*. Lalu apakah anda yakin ingin menghapus data ini? Apakah benar bahwa ini bukanlah suatu kesalahan? Maka akan keluar suatu jendela konfirmasi untuk menghapus data yang baru saja kita pilih seperti yang ditunjukkan pada gambar 107 berikut ini.



Gambar 107 Jendela Konfirmasi Hapus Data

Tekan saja tombol hapus untuk benar benar menghapus data ini dari aplikasi *website* dan juga *database*. Dan selesai, data sudah terhapus. Untuk menunjukkan bahwa datanya sudah benar benar terhapus, yaitu cari saja modul 1.7 pada kolom pencarian yang sudah disediakan. Dan hasilnya ditunjukkan pada gambar 108 berikut ini

Show	10	entries	Search:	1.7
ID STAGE	▲	ID MODUL	▼	NAMA PEKERJA
TOTAL NILAI	▼	STATUS KELULUSAN	▼	
No matching records found				
Showing 0 to 0 of 0 entries (filtered from 13 total entries)	Previous	Next		

Gambar 108 Data Sudah Terhapus

Dan ya, data sudah benar benar terhapus karena hasil pencarian dari modul 1.7 menunjukkan tidak ada data yang sesuai dengan *keyword* pada kolom pencarian. Dengan ini membuktikan bahwa untuk halaman menu *assessment* sudah dapat berfungsi dengan baik, karena tidak ditemukannya ada kesalahan ataupun *error* yang terjadi selama melakukan proses CRUD.

Selesai sudah pembuatan menu *assessment*, maka langkah selanjutnya yang harus dilakukan adalah membuat menu ruangkerja dimana ini akan meng-*import* data data ruangkerja. Karena pelatihan ini bekerja sama dengan ruangkerja, yang akan memberikan hasil penilaian *e-learning* dalam bentuk *file spreadsheet* atau yang biasa dikenal *file excel*. Kenapa diberikan fitur *import* pada menu ini? Karena data yang di-*update* sangat banyak dari ruangkerja perharinya, sehingga sangat tidak efisien apabila kita harus meng-*update* data yang berubah satu satunya.

Maka untuk memudahkan dan juga membuat proses *update* data ini menjadi lebih efisien, disediakanlah fitur *import*.

## 2.5. CRUD Menu Ruangkerja

Menu selanjutnya yang akan dibuat adalah menu ruangkerja. Dimana data data yang diterima admin akan dimasukkan ke dalam aplikasi *website* EPPM Go! ini. Kenapa harus ada menu ini? Karena pelatihan yang dijalankan bekerja sama dengan ruangkerja untuk modul pelatihan *e-learning* jadi para peserta diharuskan lulus terlebih dahulu dari pelatihan ruangkerja ini. Sebelum kita menuju penjelasan *tutorial* maka ada baiknya dijelaskan terlebih dahulu apa itu ruangkerja

Ruangkerja merupakan sebuah *platform* yang dikeluarkan oleh ruangguru yang bertujuan untuk meningkatkan kemampuan para pekerja di perusahaan tertentu. Dengan adanya pelatihan dari ruangkerja, maka perusahaan dapat menghemat banyak sekali *resources* yang dikeluarkan untuk pelatihan konvensional pada umumnya.



*Gambar 109 Ruangkerja*

Selain itu, ruangkerja memiliki keunggulan berupa beberapa fitur yang sangat menarik untuk digunakan karena ruangkerja memahami beragam kebutuhan terkait proses pelatihan di era modern ini. Berikut adalah fitur-fitur dari ruangkerja yang secara khusus disesuaikan untuk pengguna:

- a. *Mobile Learning*, pelatihannya dikemas untuk dapat diakses melalui *smartphone*
- b. *Learning Journey*, memiliki pendekatan belajar yang terstruktur dan sistematis
- c. *Powerful User Interface*, UI/UX yang menarik agar peserta nyaman
- d. *Collaboration*, adanya forum untuk diskusi yang membuka ruang untuk kolaborasi
- e. *Leaderboard*, adanya peringkat untuk menambah semangat belajar
- f. *Notifications*, informasi penugasan pelatihan atau *reminder* akan dikirim secara otomatis melalui email atau *push notification*
- g. *Certificate*, peserta pelatihan akan mendapat sertifikat bila telah menyelesaikan pelatihan dengan baik

Selain untuk pengguna, pastinya setiap pelatihan memiliki administrator yang bertugas untuk melakukan proses pengawasan dari pelatihan yang sedang dijalankan. Maka ruangkerja pun memfasilitasi fitur-fitur yang diberikan untuk administrator dalam proses pengawasan:

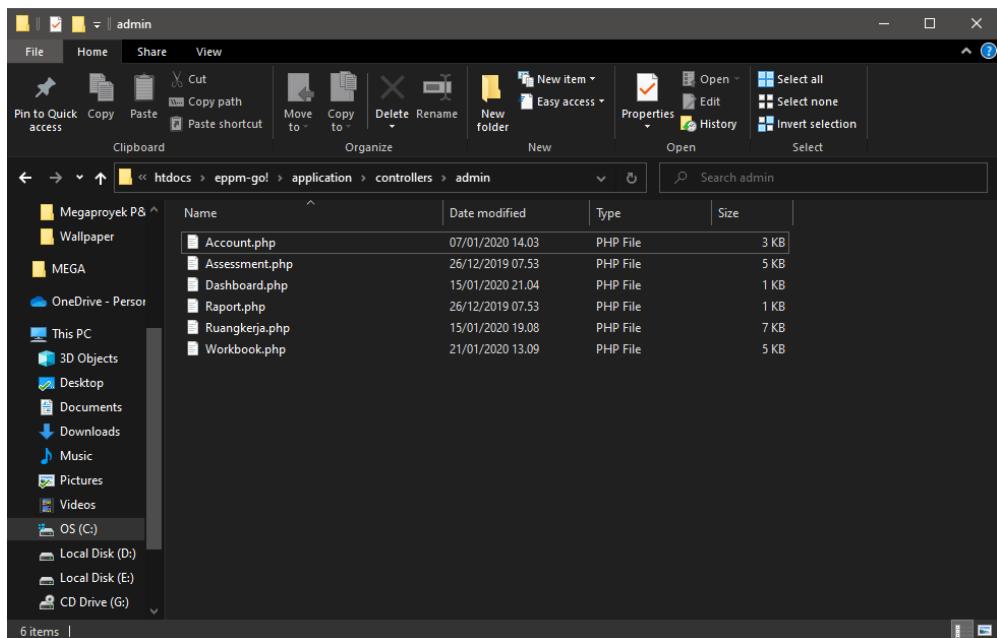
- a. *Monitoring Dashboard*, pemantauan pelatihan yang bersifat menyeluruh dan *seamless*.
- b. *Create Course*, dapat membuat kelas pelatihan yang sesuai dengan kebutuhan perusahaan
- c. *User Grouping & Enrollment*, adanya fitur untuk *enrollment* peserta pelatihan secara masal

Selain fitur-fitur yang ditawarkan, keunggulan dari ruangkerja adalah layanan yang mengcover keseluruhan proses pelatihan dari awal hingga pelatihan itu selesai, maka tak heran banyak perusahaan multinasional yang menggunakan jasa dari ruang kerja. Berikut adalah layanan yang ditawarkan oleh ruangkerja:

- a. *Pre-Onboarding*, yang meliputi analisis kebutuhan, penetapan tujuan, dan juga *strategy mapping*
- b. *Learning Design*, yang meliputi kebutuhan konseptual, kurasi konten, dan juga *storyboarding*
- c. *Content Development*, yang meliputi produksi video dan juga proses penyuntingan dan animasi
- d. *Implementation*, yang meliputi uji penerimaan pengguna, dan juga *go live*
- e. *Change Management Support*, yang meliputi *training of trainer*, *customer success support*, dan juga *data analysis*
- f. *Technical Support*, yang meliputi layanan pengguna 24 jam seminggu, dan juga perbaikan operasional

Setelah mendapatkan penjelasan mengenai apa itu ruangkerja, alangkah baiknya kita melanjutkan materi kita. Yaitu membuat CRUD menu ruangkerja dengan fitur *import* data csv.

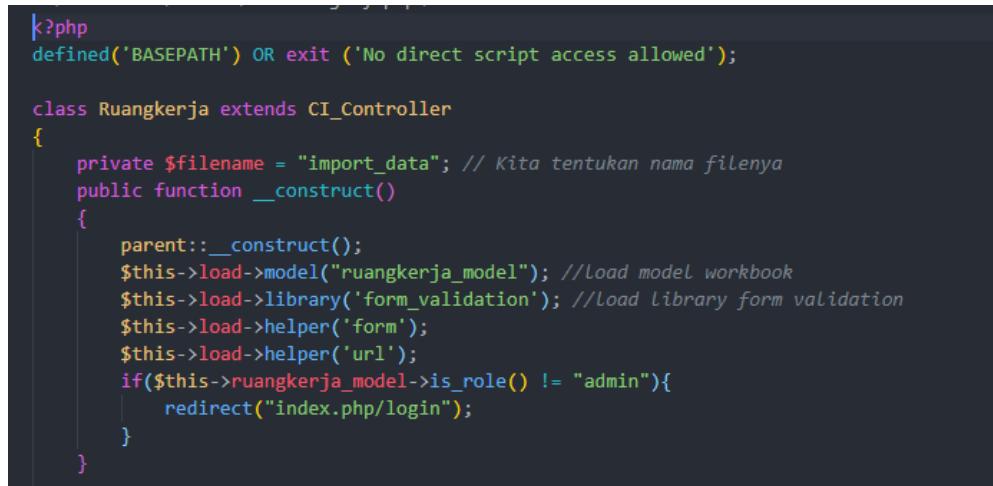
Langkah pertama adalah kita harus membuat *file* Ruangkerja.php, di dalam folder admin pada direktori *controllers* seperti yang ditunjukkan pada gambar 110 berikut ini.



Gambar 110 File Ruangkerja

Di dalam *file* Ruangkerja.php kita akan mengisikannya dengan sintak sintak untuk *controllers* yang berfungsi untuk menghubungkan *models* dan juga *views*. Selain itu, *controllers* berfungsi untuk melakukan pemrosesan dari data data yang dilemparkan oleh *models*.

Selanjutnya langsung saja kita isikan *file* Ruangkerja.php dengan isian yang ditunjukkan oleh gambar 111 di bawah ini.



```
<?php
defined('BASEPATH') OR exit ('No direct script access allowed');

class Ruangkerja extends CI_Controller
{
    private $filename = "import_data"; // Kita tentukan nama filenya
    public function __construct()
    {
        parent::__construct();
        $this->load->model("ruangkerja_model"); //Load model workbook
        $this->load->library('form_validation'); //Load Library form validation
        $this->load->helper('form');
        $this->load->helper('url');
        if($this->ruangkerja_model->is_role() != "admin"){
            redirect("index.php/login");
        }
    }
}
```

Gambar 111 Controllers Ruangkerja.php

Pada gambar 111 berikut, kita terlebih dahulu membuat sebuah *class* dengan nama Ruangkerja dengan menggunakan *library* dari CI\_Controller. Selanjutnya adalah kita menginisiasikan sebuah variabel yang secara privat, yang artinya hanya akan digunakan oleh variabel itu sendiri saja. Berikan nama dari variabel tersebut \$filename. Variabel ini berfungsi untuk menentukan nama *file* dari csv dan excel yang akan kita *import* nanti. Kemudian kita membuat sebuah *function \_\_construct()* yang di dalam *function* itu kita me-*load* models ruangkerja\_model, me-*load* library form validasi dan juga *helper*-nya. Lalu kita bila ada pengguna yang akan menggunakan menu ruangkerja maka dicek terlebih dahulu *role* dari pengguna tersebut. Jika *role* yang masuk bukan admin maka akan diarahkan kembali menuju *login*.

```
public function index()
{
    $data["ruangkerja"] = $this->ruangkerja_model->getAll(); //ambil data dari model
    // print_r('tes'); die;
    $this->load->view("admin/ruangkerja/list_ruangkerja", $data); //Load view data model ke workbook
}
```

Gambar 112 Controllers Ruangkerja.php

Langkah selanjutnya adalah memberi isian Ruangkerja.php seperti yang ditunjukkan pada gambar 112 di atas. Pertama kita harus membuat sebuah *function index()* di dalamnya inisiasikan variabel yang berfungsi untuk mengambil semua data dari tabel ruangkerja di *database*. Lalu tampilkan data tersebut pada halaman *list\_ruangkerja*. Setelah membuat *function index()* maka langkah selanjutnya yang harus ditempuh adalah kita membuat *function edit()* yang di dalamnya inisiasikan variabel yang berguna untuk mengambil data berdasarkan id yang dipilih, lalu tampilkan data dari id yang dipilih pada halaman *edit\_ruangkerja*.

Selanjutnya adalah membuat *function aksi\_edit()* dengan menyimpan data yang baru saja kita ubah dan simpan ke dalam *database*. Pertama inisiasikan variabel \$id yang akan menyimpan data dari id-nya. Lalu kita inisiasikan kembali variabel \$data dalam bentuk *array* yang akan menyimpan data yang banyak ke dalam satu variabel saja sebelum disimpan ke dalam *database*. Setelah itu masukkan perubahan yang terjadi ke dalam *database* dan bila datanya berhasil disimpan maka akan diarahkan kembali ke halaman ruangkerja. Seperti yang ditunjukkan pada gambar 113 berikut ini.

```

public function edit($id=null)
{
    $data["ruangkerja"] = $this->ruangkerja_model->getById($id); //mengambil data berdasarkan id
    if (!$data["ruangkerja"]) show_404();// jika tidak ada show error
    $this->load->view("admin/ruangkerja/edit_ruangkerja", $data); //load edit form workbook
}

public function aksi_edit()
{
    $id = $this->input->post('ruangkerja_id');

    $data = array (
        "ruangkerja_id" => $this->input->post("ruangkerja_id"),
        "employee_id" => $this->input->post("employee_id"),
        "employee_name" => $this->input->post("employee_name"),
        "user_email" => $this->input->post("user_email"),
        "start_time" => $this->input->post("start_time"),
        "posttest_score" => $this->input->post("posttest_score"),
        "status" => $this->input->post("status"),
        "deadline_course" => $this->input->post("deadline_course"),
        "clear_time" => $this->input->post("clear_time"),
        "directorate" => $this->input->post("directorate")
    );
    // print_r($data); die;
    $this->ruangkerja_model->update($id,$data);
    $this->session->set_flashdata('success', 'Data Anda Berhasil Diupdate');
    redirect('index.php/admin/ruangkerja');
}

```

*Gambar 113 Controllers Ruangkerja.php*

Setelah kita membuat *function* seperti yang ditunjukkan oleh gambar 113 di atas, maka untuk melengkapi lagi *file* Controllers yang sedang kta buat tambahkan *function* untuk *delete* dan juga *import file excel* dan *csv*. Seperti yang ditunjukkan oleh gambar 114 dan juga gambar 115 di bawah ini

```

public function delete($id=null)
{
    if (!isset($id)) show_404();

    if ($this->ruangkerja->delete($id)){
        redirect('index.php/admin/ruangkerja');
    }
}

```

Gambar 114 Controllers Ruangkerja.php

Pada gambar 114, kita membuat *function* *delete()* yang berfungsi untuk menghapus data dari *database* dan juga aplikasi *website* dengan mengambil id dari data yang dipilih.

```

public function formImport(){
    $data = array(); // Buat variabel $data sebagai array

    if(isset($_POST['preview'])){ // Jika user menekan tombol Preview pada form
        // Lakukan upload file dengan memanggil function upload yang ada di ModelAdmin.php
        $upload = $this->ruangkerja_model->upload_file($this->filename);

        if($upload['result'] == "success"){ // Jika proses upload sukses
            // Load plugin PHPExcel nya
            include APPPATH.'third_party/PHPExcel/PHPExcel.php';

            $excelreader = new PHPExcel_Reader_Excel2007();
            $loadexcel = $excelreader->load('excel/'.$this->filename.'.xlsx'); // Load file yang tadi diupload
            $sheet = $loadexcel->getActiveSheet()->toArray(null, true, true ,true);

            // Masukan variabel $sheet ke dalam array data yang nantinya akan dikirim ke file form.php
            // Variabel $sheet tersebut berisi data-data yang sudah diinput di dalam excel yang sudah diupload
            $data['sheet'] = $sheet;
        }else{ // Jika proses upload gagal
            $data['upload_error'] = $upload['error']; // Ambil pesan error uploadnya untuk dikirim ke form
        }
    }
    $this->load->view('admin/ruangkerja/import_ruangkerja', $data);
}

```

Gambar 115 Controllers Ruangkerja.php

Lalu selanjutnya adalah membuat *function* formImport() lalu membuat variabel \$data dimana datanya akan diubah menjadi sebuah *array*. Lalu jika pengguna menekan tombol *import* lakukan *upload file* dengan memanggil *function upload* pada models. Jika proses *upload* selesai maka akan me-*load plugin* dari excelnya. Jika berhasil, maka tampilkan data data tersebut dengan menampilkan halaman import\_ruangkerja.

Selanjutnya adalah membuat *function* import() yang akan berfungsi untuk memasukkan data dari kolom dan baris yang ada di excel yang akan kita lakukan *import*. Untuk lebih jelasnya silakan perhatikan gambar 116 di bawah ini.

```
public function formImport(){
    $data = array(); // Buat variabel $data sebagai array

    if(isset($_POST['preview'])){ // Jika user menekan tombol Preview pada form
        // Lakukan upload file dengan memanggil function upload yang ada di ModelAdmin.php
        $upload = $this->ruangkerja_model->upload_file($this->filename);

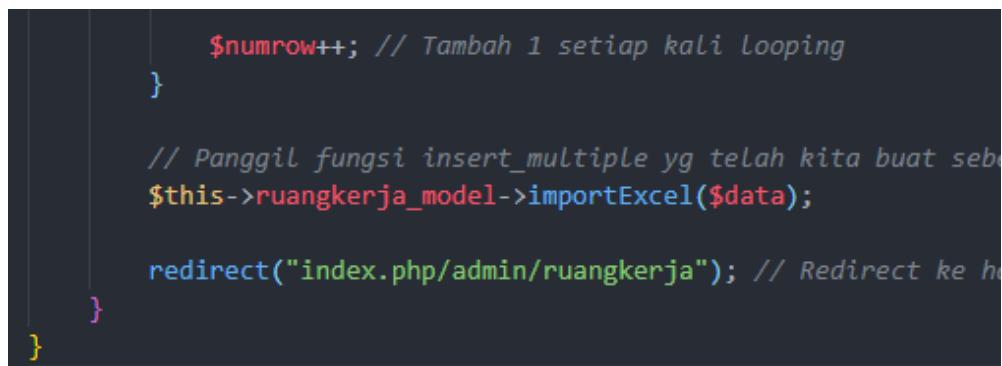
        if($upload['result'] == "success"){ // Jika proses upload sukses
            // Load plugin PHPExcel nya
            include APPPATH.'third_party/PHPExcel/PHPExcel.php';

            $excelreader = new PHPExcel_Reader_Excel2007();
            $loadexcel = $excelreader->load('excel/'.$this->filename.'.xlsx'); // Load file yang tadi diupload
            $sheet = $loadexcel->getActiveSheet()->toArray(null, true, true ,true);

            // Masukan variabel $sheet ke dalam array data yang nantinya akan di kirim ke file form.php
            // Variabel $sheet tersebut berisi data-data yang sudah diinput di dalam excel yang sudah diupload
            $data['sheet'] = $sheet;
        }else{ // Jika proses upload gagal
            $data['upload_error'] = $upload['error']; // Ambil pesan error uploadnya untuk dikirim ke form
        }
    }
    $this->load->view('admin/ruangkerja/import_ruangkerja', $data);
}
```

Gambar 116 Controllers Ruangkerja.php

Setelahnya kita akan lakukan *looping* dimana apabila ada banyak data baru yang ingin diimport maka akan dilakukan *looping* untuk memasukkan data tersebut satu persatu ke dalam *database*. Jika berhasil maka akan diarahkan kembali ke menu ruangkerja. Seperti yang ditunjukkan pada gambar 117 berikut ini.



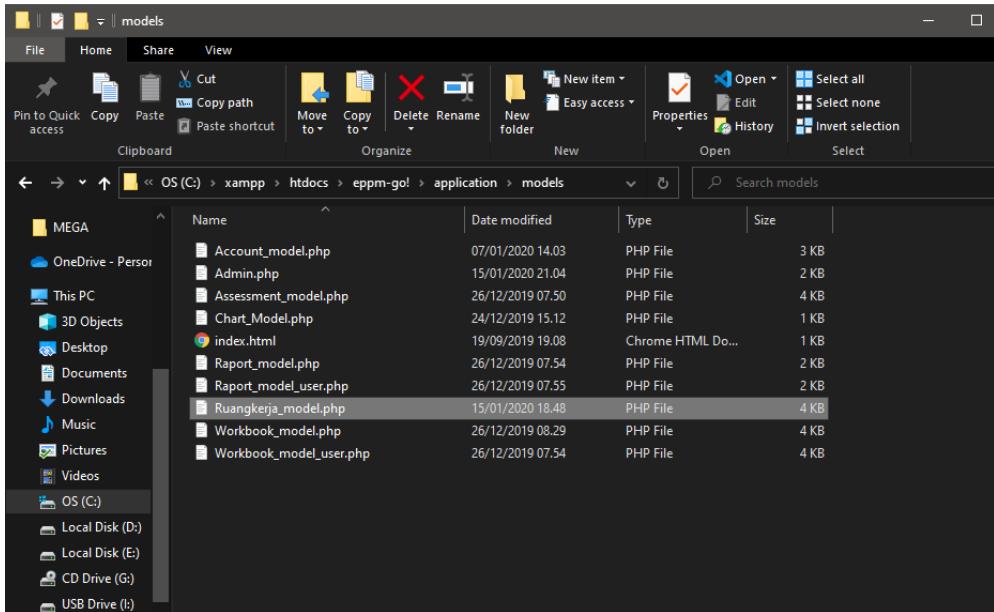
```
        $numrow++; // Tambah 1 setiap kali looping
    }

    // Panggil fungsi insert_multiple yg telah kita buat sebelumnya
    $this->ruangkerja_model->importExcel($data);

    redirect("index.php/admin/ruangkerja"); // Redirect ke halaman awal
}
}
```

Gambar 117 Controllers Ruangkerja.php

Dengan ini, selesai sudah tahapan tahapan dan komponen yang harus dibuat untuk menu ruangkerja. Maka langkah selanjutnya untuk menyempurnakan menu ruangkerja ini adalah membuat ruangkerja\_model yang berfungsi untuk melakukan manipulasi datanya. Langsung saja buat terlebih dahulu sebuah *file* dengan nama ruangkerja\_model.php pada folder admin di direktori models dari *localhost* kita. Seperti yang ditunjukkan oleh gambar 118 di bawah ini.



Gambar 118 Ruangkerja\_model.php

Jika sudah dibuat *file*-nya langsung kita berikan isian untuk *file* ruangkerja\_model.php tersebut, dengan membuat sebuah *class* terlebih dahulu dengan nama Ruangkerja\_model dengan menggunakan *library* dari CI\_Model.

Lanjut kita menginisiasikan terlebih dahulu variabel variabel yang akan kita gunakan, jika sudah buatlah *function* dengan nama getAll() yang akan mengambil seluruh data di dalam *database* seperti yang dapat kita lihat pada gambar 119 berikut ini

```
<?php defined('BASEPATH') OR exit('No direct script access allowed');

class Ruangkerja_model extends CI_Model
{
    private $_table = "ruangkerja";

    public $ruangkerja_id;
    public $employee_id;
    public $employee_name;
    public $user_email;
    public $start_time;
    public $posttest_score;
    public $status;
    public $deadline_course;
    public $clear_time;
    public $directorate;

    public function getAll()
    {
        return $this->db->get($this->_table)->result();
        // print_r('tes'); die;
    }
}
```

Gambar 119 Models Ruangkerja\_model.php

Jika sudah selesai, maka langsung kita lanjutkan ke tahapan selanjutnya dalam pembuatan Ruangkerja\_model.php seperti yang ditunjukkan pada gambar 120 di bawah ini, dimana kita akan membuat *function* upload\_file() agar dapat mengimport file excel yang akan digunakan untuk menu ruangkerja ini.

```

// Fungsi untuk melakukan proses upload file
public function upload_file($filename){
    $this->load->library('upload'); // Load Librari upload

    $config['upload_path'] = './excel/';
    $config['allowed_types'] = 'xlsx';
    $config['max_size'] = '2048';
    $config['overwrite'] = true;
    $config['file_name'] = $filename;

    $this->upload->initialize($config); // Load konfigurasi uploadnya
    if($this->upload->do_upload('file')){ // Lakukan upload dan Cek jika proses upload berhasil
        // Jika berhasil :
        $return = array('result' => 'success', 'file' => $this->upload->data(), 'error' => '');
        return $return;
    }else{
        // Jika gagal :
        $return = array('result' => 'failed', 'file' => '', 'error' => $this->upload->display_errors());
        return $return;
    }
}

```

*Gambar 120 Models Ruangkerja\_model.php*

Setelah membuat *function*-nya maka kita *load* terlebih dahulu *library upload* dan membuat konfigurasi untuk disimpan dimana *file*-nya lalu ekstensi apa yang diperbolehkan untuk *diupload*, ukuran maksimum *file* yang boleh *diupload* lalu apakah bisa *dioverwrite* dan nama *file*-nya.

Setelah membuat konfigurasinya, maka gunakan konfigurasi tersebut. Lalu lakukan *upload* dan cek apabila proses *upload* berhasil berikan pesan berhasil dan bila gagal maka berikan juga pesan gagal dengan memunculkan notifikasi galat.

```

public function autoIncrementRuangkerja(){
    return $this->db->query("ALTER TABLE ruangkerja AUTO_INCREMENT =1;");
}

public function importExcel($data)
{
    return $this->db->insert_batch('ruangkerja', $data);
}

public function getById($id)
{
    return $this->db->get_where($this->_table, ["ruangkerja_id" => $id])->row();
}

public function get_data_stage()
{
    $query = $this->db->get('stage');
    return $query;
}

```

*Gambar 121 Models Ruangkerja\_model.php*

Jika sudah selesai membuat *function* upload\_file() maka kita membuat *function* autoIncrementRuangkerja() yang mana berfungsi untuk menambahkan atau merubah id dari ruangkerja secara otomatis. Seperti yang dapat dilihat pada gambar 121 terdapat *query* untuk menambahkan id dari ruangkerja +1.

Lanjut langkah selanjutnya adalah membuat *function* importExcel() yang berfungsi untuk melakukan *import excel* ke dalam tabel ruangkerja di dalam *database*. Data data yang ada pada excel akan dimasukkan ke dalam *database*. Lalu lanjut kita membuat *function* getById yang akan mengambil id berdasarkan data yang dipilih nanti, dimana id-nya akan diambil melalui tabel ruangkerja di *database*. Tak lupa kita membuat *function* get\_data\_stage() untuk mengambil data *stage* pada tabel *stage* di dalam *database*.

```

public function update($id, $data)
{
    $this->db->where('ruangkerja_id', $id);
    $this->db->update("ruangkerja", $data);
}

public function delete($id)
{
    // $this->_deleteImage($id);
    return $this->db->delete($this->_table, array("ruangkerja_id" => $id));
}

//fungsi cek Level
function is_role()
{
    return $this->session->userdata('role');
}

}

```

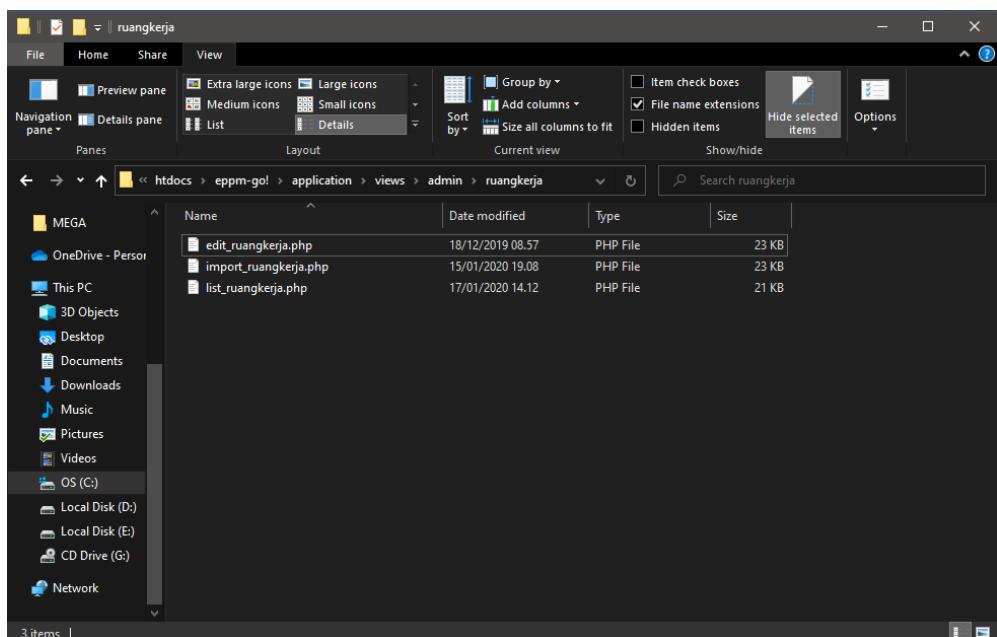
Gambar 122 Models Ruangkerja\_model.php

Sekarang kita akan membuat *function* yang berfungsi untuk menyimpan perubahan data yang akan kita masukkan ke dalam *database*. Yaitu kita membuat *function* update() yang akan meyimpan segala perubahan data dengan mengambil id sebagai tanda pengenal dan juga meyimpan perubah dari setiap *field*.

Lalu kita membuat *function* delete() yang akan menghapus data dari *database* dengan mengambil id dari data yang kita pilih lalu akan menghapus data tersebut. Tak lupa kita juga membuat *function* untuk mengecek *role* yang masuk ke dalam sistem, sehingga akan mengecek *role* dari *session* yang sedang aktif.

Untuk lebih jelasnya maka dapat melihat beberapa baris sintak yang ditunjukkan oleh gambar 122 di atas.

Dengan ini selesai sudah langkah langkah yang diperlukan untuk membuat models Ruangkerja\_model.php. Jadi tinggal satu langkah lagi sebelum menu ruangkerja dapat kita coba, yaitu membuat views atau tampilannya. Maka langsung saja kita membuat folder ruangkerja pada folder admin di direktori views. Ini adalah *path* direktorinya C:\xampp\htdocs\eppm-go!\application\views\admin\ruangkerja. Lalu di dalamnya buatlah 3 *file* dengan nama list\_ruangkerja.php, edit\_ruangkerja.php, dan juga import\_ruangkerja.php seperti yang ditunjukkan oleh gambar 123 di bawah ini.



Gambar 123 File Ruangkerja

Lalu setelah dibuat ketiga *file* tersebut, langsung saja kita ke sintak inti dari *file* list\_ruangkerja.php seperti yang akan ditunjukkan oleh gambar 124 berikut ini, dimana kita akan membuat tabel yang akan menampung atau mewadahi data data yang akan ditampilkan.

```
<!-- Table -->
<div class="row">
<div class="col">
    <div class="card shadow">
        <div class="card-header border-0">
            <h3 class="mb-0">Ruangkerja</h3>
            <br>
            <a href=<?= base_url();?>admin/ruangkerja/formImport" class="btn btn-info">Import</a>
        </div>
        <div class="table-responsive">
            <table align-items-center table-flush>
                <thead class="thead-light">
                    <tr>
                        <th scope="col">ID Ruangkerja</th>
                        <!-- <th scope="col">ID Modul</th> -->
                        <th scope="col">Nomor Pekerja</th>
                        <th scope="col">Nama Pekerja</th>
                        <th scope="col">Email</th>
                        <th scope="col">Start Time</th>
                        <th scope="col">Post Test Score</th>
                        <th scope="col">Status</th>
                        <th scope="col">Deadline Course</th>
                        <th scope="col">Clear Time</th>
                        <th scope="col">Direktorat</th>
                        <th scope="col"></th>
                    </tr>
                </thead>
                <tbody>
```

Gambar 124 Views list\_ruangkerja.php

Setelah selesai dibuat tabelnya, saatnya kita panggil datanya sesuai dengan kolom dari *field* setiap datanya. Sama seperti yang sudah dilakukan pada menu menu sebelumnya, maka kita juga membuat perulangan dengan metode *foreach* untuk menampilkan datanya sampai habis

Untuk itu, lakukan *foreach* dengan datanya dengan mengambil data tersebut dari tabel ruangkerja melalui inisiasi variabel \$ruangkerja pada models sebelumnya. Untuk lebih detailnya maka dapat dilihat pada gambar 125 berikut ini.

```
<tbody>
    <?php foreach ($ruangkerja as $ruangkerja): ?>
    <tr>
        <td width="150">
            <?php echo $ruangkerja->ruangkerja_id ?>
        </td>
        <td>
            <?php echo $ruangkerja->employee_id ?>
        </td>
        <td>
            <?php echo $ruangkerja->employee_name ?>
        </td>
        <td>
            <?php echo $ruangkerja->user_email ?>
        </td>
        <td>
            <?php echo $ruangkerja->start_time ?>
        </td>
        <td>
            <?php echo $ruangkerja->posttest_score ?>
        </td>
        <td>
            <?php echo $ruangkerja->status ?>
        </td>
        <td>
            <?php echo $ruangkerja->deadline_course ?>
        </td>
        <td>
            <?php echo $ruangkerja->clear_time ?>
        </td>
        <td>
            <?php echo $ruangkerja->directorate ?>
        </td>
```

Gambar 125 Views list\_ruangkerja.php

Lalu jangan lupa, kita juga harus memberika tombol *update* dan juga hapus jikalau sewaktu waktu ada data yang perlu diubah atau dihapus. Lalu jika sudah selesai akhiri perulangan *foreach*-nya seperti yang ditunjukkan pada gambar 126 berikut ini.

```
<td width="250">
    <a href="php echo site_url('admin/ruangkerja/edit/'.$ruangkerja-&gt;ruangkerja_id);?"&gt;
        class="btn btn-small"&gt;&lt;i class="fas fa-edit"&gt;&lt;/i&gt;Update&lt;/a&gt;
    &lt;a onclick="deleteConfirm('<?php echo site_url('admin/ruangkerja/delete/'.$ruangkerja-&gt;ruangkerja_id);?'&gt;" href="#" class="btn btn-small text-danger"&gt;&lt;i class="fas fa-trash"&gt;&lt;/i&gt;Hapus&lt;/a&gt;
&lt;/td&gt;
&lt;/tr&gt;
&lt;?php endforeach; ?&gt;</pre
```

Gambar 126 Views list\_ruangkerja.php

Karena, masih menggunakan DataTables jadi kita harus menggunakan sintaks jQuery untuk menggunakan *resources* dari DataTables. Setelah itu, untuk sintaks jQuery dari DataTables kita taruh dibawah setelah sintaks pembuatan tabel. Jangan lupa untuk menyisipkan tag <script> karena masih satu bagian dari JavaScript. Selain itu, karena kita menggunakan *resources* dari luar ada beberapa cara yang bisa digunakan, yaitu dengan cara:

1. Mengunduh *library*-nya dan disimpan pada folder *assets* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini akan menggunakan *local resource* dari PC atau laptop kita.

2. Menyalin *link CDN* (*Content Delivery Network*) yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini hanya memerlukan akses internet. *Link CDN* DataTables untuk CSS adalah //cdn.datatables.net/1.10.20/css/jquery.dataTables.min.css dan *link CDN* DataTables untuk JavaScript adalah //cdn.datatables.net/1.10.20/js/jquery.dataTables.min.js

```
<script>$(document).ready( function () {
|   $('table').DataTable();
} );
</script>
```

Gambar 127 Views list\_ruangkerja.php

Untuk tombol hapus, kita harus selalu menggunakan konfirmasi sebelum dilakukannya penghapusan data. Ini berguna untuk mencegah dan meminimalisir terjadinya penghapusan data secara tidak sengaja, dan kita menggunakan modal untuk konfirmasinya seperti yang ditunjukkan gambar 128 di bawah ini

```
<script>
function deleteConfirm(url){
  $('#btn-delete').attr('href', url);
  $('#deleteModal').modal();
}
</script>
```

Gambar 128 Views list\_ruangkerja.php

Maka kita coba terlebih dahulu, apakah tabelnya berhasil dibuat? Apakah datanya juga terpanggil? Maka kita buka *browser* dan ketikkan url `localhost:8080/eppm-go!/login`. Setelah melakukan login maka langsung menuju menu ruangkerja

The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'localhost:8080/eppm-go!/admin/ruangkerja'. The page content is a table titled 'Ruangkerja' with the following data:

ID RUANGKERJA	NOMOR PEKERJA	NAMA PEKERJA	EMAIL	START TIME	POST TEST
1	755303	Abdul Karim Amarullah	abdul.amarullah@pertamina.com	2019-06-13 02:21:30	100
2	718027	Abdullah	abdullah2708@pertamina.com	0000-00-00 00:00:00	0
3	750787	Achmad Hattary	achmad.hattary@pertamina.com	0000-00-00 00:00:00	0
4	10026617	Achmad Riza Pamula	achmad.pamula@pertamina.com	0000-00-00 00:00:00	0
5	755304	Achmad Syafiudin	achmad.syafiudin@pertamina.com	2019-06-12 05:56:43	73
6	714163	Ali Dwiyono Haritsyah	ad_haritsyah@pertamina.com	0000-00-00 00:00:00	0

Gambar 129 Views Menu Ruangkerja

Dapat dilihat pada gambar 129 di atas, tabel dan datanya berhasil dibuat dan dipanggil ke dalam menu ruangkerja. Maka hal selanjutnya adalah kita harus membuat form *update* yang akan muncul bila menekan tombol *update* dan juga membuat form *import* yang akan muncul bila menekan tombol *import*.

Maka kita lanjut ke *file* selanjutnya, yaitu edit\_ruangkerja.php dimana di dalam *file* ini kita akan membuat form *update* untuk merubah data yang ingin kita rubah. Langsung ke inti pembahasan, maka kita akan membuat form untuk setiap *field* dari ruangkerja. Seperti yang ditunjukkan pada gambar 130 di bawah ini.

```
<!-- Table -->


<div class="card-header">
    <a href="php echo site_url('admin/ruangkerja/') ?&gt;"<i class="fas fa-arrow-left"></i>
    Kembali</a>
  </div>
  <div class="card-body">

    <form action="php echo base_url();?&gt;index.php/admin/ruangkerja/aksi_edit" method="post" enctype="multipart/fo

    &lt;input type="hidden" name="ruangkerja_id" value="<?php echo $ruangkerja-&gt;ruangkerja_id?&gt;" /&gt;

    &lt;div class="form-group"&gt;
      &lt;label for="name"&gt;No. Pekerja&lt;/label&gt;
      &lt;input class="form-control &lt;?php echo form_error('employee_id') ? 'is-invalid':'?&gt;" type="text" name="employee_id" min="0" placeholder="No. Pekerja" value="<?php echo $ruangkerja-&gt;employee_i
      &lt;div class="invalid-feedback"&gt;
        &lt;?php echo form_error('employee_id') ?&gt;
      &lt;/div&gt;
    &lt;/div&gt;

    &lt;div class="form-group"&gt;
      &lt;label for="employee_name"&gt;Nama Pekerja&lt;/label&gt;
      &lt;input class="form-control &lt;?php echo form_error('employee_name') ? 'is-invalid':'?&gt;" type="text" name="employee_name" min="0" placeholder="Nama Pekerja" value="<?php echo $ruangkerja-&gt;employe
      &lt;div class="invalid-feedback"&gt;
        &lt;?php echo form_error('employee_name') ?&gt;
      &lt;/div&gt;
    &lt;/div&gt;
  &lt;/div&gt;</pre



Gambar 130 Views edit_ruangkerja.php



Dapat dilihat pada gambar 130 di atas, kita membuat terlebih dahulu textbox agar dapat menampung datanya. Lalu jika textbox sudah terbuat maka akan langsung dipanggil datanya untuk dapat diubah di dalam textbox.



125


```

```

<div class="form-group">
    <label for="user_email">Email</label>
    <input class="form-control" type="text" name="user_email" min="0" placeholder="Email" value="<?php echo $ruangkerja->user_email ?>" />
    <div class="invalid-feedback">
        <?php echo form_error('user_email') ?>
    </div>
</div>

<div class="form-group">
    <label for="start_time">Start Time</label>
    <input class="form-control" type="date" name="start_time" min="0" placeholder="Start Time" value="<?php echo $ruangkerja->start_time ?>" />
    <div class="invalid-feedback">
        <?php echo form_error('start_time') ?>
    </div>
</div>

<div class="form-group">
    <label for="posttest_score">Post Test Score</label>
    <input class="form-control" type="text" name="posttest_score" min="0" placeholder="Post Test Score" value="<?php echo $ruangkerja->posttest_score ?>" />
    <div class="invalid-feedback">
        <?php echo form_error('posttest_score') ?>
    </div>
</div>

<div class="form-group">
    <label for="status">Status</label>
    <input class="form-control" type="text" name="status" min="0" placeholder="Status" value="<?php echo $ruangkerja->status ?>" required/>
    <div class="invalid-feedback">
        <?php echo form_error('status') ?>
    </div>

```

*Gambar 131 Views edit\_ruangkera.php*

Sama seperti yang kita lakukan sebelumnya, pada gambar 131 di atas masih merupakan kelanjutan dari langkah yang sedang kita lakukan pada gambar 130 sebelumnya.

```

<div class="form-group">
    <label for="deadline_course">Deadline Course</label>
    <input class="form-control <?php echo form_error('deadline_course') ? 'is-invalid':'' ?>" type="date" name="deadline_course" min="0" placeholder="Deadline Course" value="<?php echo $ruangkerja->de
    <div class="invalid-feedback">
        <?php echo form_error('deadline_course') ?>
    </div>
</div>

<div class="form-group">
    <label for="clear_time">Clear Time</label>
    <input class="form-control <?php echo form_error('clear_time') ? 'is-invalid':'' ?>" type="date" name="clear_time" min="0" placeholder="Clear Time" value="<?php echo $ruangkerja->clear_time ?
    <div class="invalid-feedback">
        <?php echo form_error('clear_time') ?>
    </div>
</div>

<div class="form-group">
    <label for="directorat">Direktorat</label>
    <input class="form-control <?php echo form_error('directorat') ? 'is-invalid':'' ?>" type="text" name="directorat" min="0" placeholder="Direktorat" value="<?php echo $ruangkerja->directorat
    <div class="invalid-feedback">
        <?php echo form_error('directorat') ?>
    </div>
</div>

</div>

<input class="btn btn-success" type="submit" name="btn" value="Simpan" />
</form>

```

Gambar 132 Views edit\_ruangkerja.php

Setelah semua *textbox* untuk setiap *field* dari tabel ruangkerja dibuat, maka jangan lupa untuk membuat tombol simpan dimana yang akan berfungsi untuk menyimpan perubahan yang baru saja kita lakukan pada form *update* ini. Untuk itu selesaikan sudah tahapan yang diperlukan untuk pembuatan form *update* maka kita akan lakukan ujicoba ala ala dengan mengetik url localhost:8080/eppm-go!/login pada *browser* dan lakukan *login*. Setelah itu masuk ke menu ruangkerja dan pilih satu data untuk dilakukan *update* seperti yang ditunjukkan oleh gambar 133 berikut ini.

The screenshot shows a web-based application interface for managing employee data. On the left, there is a sidebar with a blue header containing the logo 'eppmgo!' and several menu items: Dashboard, Daftar Peserta, Workbook, Assessment, Data Ruangkerja, Raport, and Logout. Below this is a 'DOCUMENTATION' section with links to Getting started, Foundation, and Components.

The main content area has a purple header labeled 'TABLES'. A modal window titled 'Kembali' is open, prompting for 'No. Pekerja\*' (Employee No.) with the value '718027', 'Nama Pekerja' (Name) with the value 'Abdullah', 'Email' with the value 'abdullah2708@pertamina.com', 'Start Time' (Start Date) with the placeholder 'mm / dd / yyyy', and 'Post Test Score' (Post Test Score) with the value '0'. In the top right corner of the modal, there is a user profile picture of a woman and the name 'Jessica Jones'.

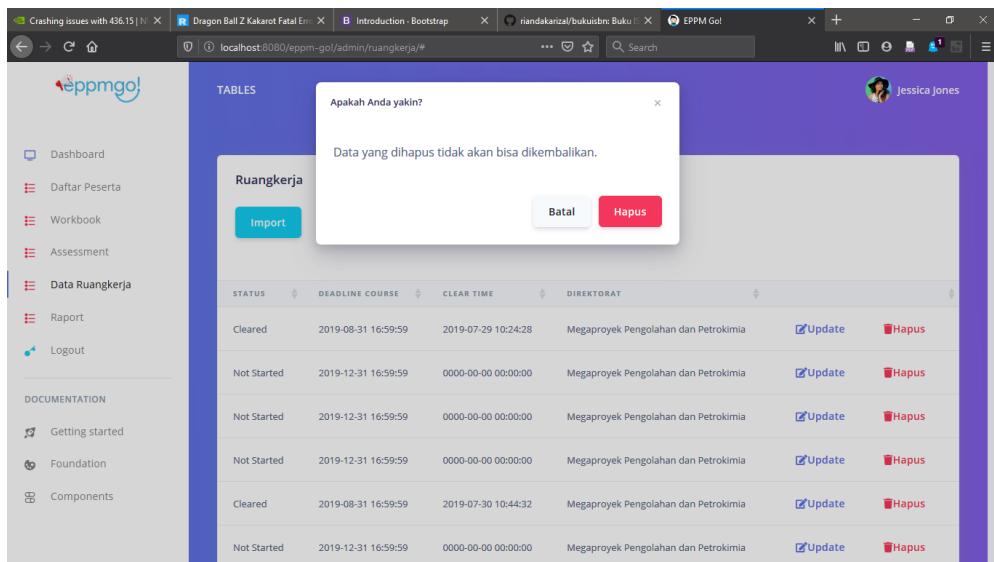
*Gambar 133 Views Form Update*

Dapat dilihat, kita sudah membuka form *update* dan kita akan coba lakukan perubahan data dan akan kita coba simpan

This screenshot shows the same application interface as the previous one, but with changes made to the form fields. The 'Post Test Score' field now contains the value '0'. The 'Status' field is set to 'Not Started'. The 'Deadline Course' field has the placeholder 'mm / dd / yyyy'. The 'Clear Time' field also has the placeholder 'mm / dd / yyyy'. The 'Direktorat' field contains the value 'Megaprojek Pengolahan dan Petrokimia'. At the bottom of the modal, a large green button labeled 'Simpan' (Save) is visible.

*Gambar 134 Views Form Update*

Dan datanya berhasil kita simpan ke dalam *database*. Kita coba juga hapus salah satu data dengan menekan tombol hapus seperti yang ditunjukkan pada gambar 135 berikut ini.



Gambar 135 Views Tombol Hapus

Dan akan muncul notifikasi untuk mengkonfirmasi apakah datanya benar benar akan dihapus? Jika iya maka hapus saja datanya, tetapi karena ini merupakan data yang penting, jadi tidak akan ada data dari ruangkerja yang akan dihapus.

Setelah selesai membuat form *update*, maka yang akan kita lakukan selanjutnya adalah membuat form *import* maka langsung saja kita buka *file* dari import\_ruangkerja.php yang sebelumnya telah kita buat, dan langsung saja ke bagian inti dari import\_ruangkerja.php seperti yang ditunjukkan pada gambar 136 berikut ini.

```
<!-- Table -->
<div class="row">
  <div class="col">
    <div class="card shadow">
      <div class="card-header border-0">
        <h3 class="mb-0">Ruang Kerja</h3>
        <br>
        <form method="post" action=<?php echo base_url("index.php/admin/ruangkerja/formImport"); ?>" enctype="multipart/form-data">
          <p><label>Select Excel File</label>
          <input type="file" name="file" required /></p>
          <input type="submit" name="preview" value="Preview" class="btn btn-info" />
        </form>

      <?php
      if(isset($_POST['preview'])){ // Jika user menekan tombol Preview pada form
        if(isset($upload_error)){ // Jika proses upload gagal
          echo "<div style='color: red;'>".$upload_error."</div>"; // Muncul pesan error upload
          die; // stop skrip
        }
      }

      // Buat sebuah tag form untuk proses import data ke database
      echo "<form method='post' action='".$base_url("index.php/admin/ruangkerja/import")."'>";
    </div>
  </div>
</div>
```

Gambar 136 Views import\_ruangkera.php

Pada gambar 136 di atas, kita akan membuat sebuah *preview* dari *file excel* yang kita masukkan. Lalu apabila pada saat proses *upload* maka akan memunculkan pesan galat dan menghentikan prosesnya. Selanjutnya kita membuat sebuah tag dari form untuk proses *import* data ke dalam *database*.

```
echo "<div class='table-responsive'>";
echo "<table class='table align-items-center table-flush'>
<thead class='thead-light'>
<tr>
<th>No.</th>
<th>Employee ID</th>
<th>Employee Name</th>
<th>User Email</th>
<th>Start Time</th>
<th>Posttest Score</th>
<th>Status</th>
<th>Deadline Course</th>
<th>Clear Time</th>
<th>Directorate</th>
</tr>
</thead>";

$numrow = 1;
$kosong = 0;
$no = -1;
```

Gambar137 Views import\_ruangkerja.php

Lalu setelahnya kita menginisiasikan variabel variabel yang dibutuhkan seperti yang dapat kita lihat pada gambar 137 di atas.

```
// Lakukan perulangan dari data yang ada di excel
// $sheet adalah variabel yang dikirim dari controller
echo "<tbody>";
foreach($sheet as $row){
$no++;
// Ambil data pada excel sesuai Kolom
$employee_id = $row['A'];

$employee_name = $row['B'];

$user_email = $row['C'];

$start_time = $row['D'];

$posttest_score = $row['E'];

$status = $row['F'];

$deadline_course = $row['G'];

$clear_time = $row['H'];

$directorate = $row['I'];
```

Gambar 138 Views import\_ruangkerja.php

Lanjut kita akan melakukan perulangan *foreach* dengan mengambil datanya dari excel dan disesuaikan dengan kolom pada tabel di *database* seperti yang dapat kita lihat pada gambar 138 di atas ini.

```

// Cek jika semua data tidak diisi
if($employee_id == "" && $employee_name == "" && $user_email == "" && $start_time == "" && $posttest_score == "" &&
continue;

if($numrow > 1){

$employee_id_td = ( ! empty($employee_id))? "" : " style='background: #E07171;'";
$employee_name_td = ( ! empty($employee_name))? "" : " style='background: #E07171;'";
$user_email_td = ( ! empty($user_email))? "" : " style='background: #E07171;'";
$start_time_td = ( ! empty($start_time))? "" : " style='background: #E07171;'";
$posttest_score_td = ( ! empty($posttest_score))? "" : " style='background: #E07171;'";
$status_td = ( ! empty($status))? "" : " style='background: #E07171;'";
$deadline_course_td = ( ! empty($deadline_course))? "" : " style='background: #E07171;'";
$clear_time_td = ( ! empty($clear_time))? "" : " style='background: #E07171;'";
$directorate_td = ( ! empty($directorate))? "" : " style='background: #E07171;'";
// Jika salah satu data ada yang kosong
if($employee_id == "" or $employee_name == "" or $user_email == "" or $start_time == "" or $posttest_score == "" or $status == "" or $deadline_course == "" or $clear_time == "" or $directorate == "") {
$kosong++;
}

}

```

Gambar 139 Views import\_ruangkerja.php

Dapat dilihat pada gambar 139 berikut, kita akan melakukan pengecekan terhadap baris apabila ada data yang tidak diisi maka akan memberikan sebuah tanda pada baris data tersebut. Lalu apabila ternyata ada data yang kosong maka akan dicek terlebih dahulu.

```
    }

    echo "<tr>";
    // echo "<td>".$no."</td>";
    echo "<td>".$no."</td>";
    echo "<td>".$employee_id_td.">".$employee_id."</td>";
    echo "<td>".$employee_name_td.">".$employee_name."</td>";
    echo "<td>".$user_email_td.">".$user_email."</td>";
    echo "<td>".$start_time_td.">".$start_time."</td>";
    echo "<td>".$posttest_score_td.">".$posttest_score."</td>";
    echo "<td>".$status_td.">".$status."</td>";
    echo "<td>".$deadline_course_td.">".$deadline_course."</td>";
    echo "<td>".$clear_time_td.">".$clear_time."</td>";
    echo "<td>".$directorate_td.">".$directorate."</td>";

    echo "</tr>";
    echo "</tbody>";
}

$numrow++; // Tambah 1 setiap kali Looping
}
```

Gambar 140 Views import\_ruangkerja.php

Kita lanjutkan disini dari data yang sudah dimasukkan tadi, maka akan dilakukan perulangan agar data yang ditampilkan bertambah 1 setiap kali *looping* terjadi proses *looping* selesai dan datanya sudah dikeluarkan semua.

```

echo "</table>";

// Cek apakah variabel kosong Lebih dari 0
// Jika Lebih dari 0, berarti ada data yang masih kosong
echo "<hr>";

// Buat sebuah tombol untuk mengimport data ke database
echo "<button class='btn btn-info btn-icon-split' type='submit' name='import'>Import</button>";
echo "<a href='".$base_url("index.php/ruangkerja")."'>Cancel</a>";
echo "</div>";

echo "</form>";
}

?>

```

Gambar 141 Views import\_ruangkerja.php

Karena, masih menggunakan DataTables jadi kita harus menggunakan sintaks jQuery untuk menggunakan *resources* dari DataTables. Setelah itu, untuk sintaks jQuery dari DataTables kita taruh dibawah setelah sintaks pembuatan tabel. Jangan lupa untuk menyisipkan tag <script> karena masih satu bagian dari JavaScript. Selain itu, karena kita menggunakan *resources* dari luar ada beberapa cara yang bisa digunakan, yaitu dengan cara:

1. Mengunduh *library*-nya dan disimpan pada folder *assets* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini akan menggunakan *local resource* dari PC atau laptop kita.
2. Menyalin *link CDN (Content Delivery Network)* yang akan dipanggil di atas untuk CSS di bagian tag *head* dan di bawah untuk JavaScript di bagian tag *footer*. Kekurangan cara ini hanya memerlukan akses internet. *Link CDN* DataTables untuk CSS adalah //cdn.datatables.net/1.10.20/css/jquery.dataTables.min.css dan *link*

CDN      DataTables      untuk      JavaScript      adalah  
//cdn.datatables.net/1.10.20/js/jquery.dataTables.min.js

```
<script>$(document).ready( function () {  
    $('#table').DataTable();  
} );  
</script>
```

Gambar 142 Views import\_ruangkerja.php

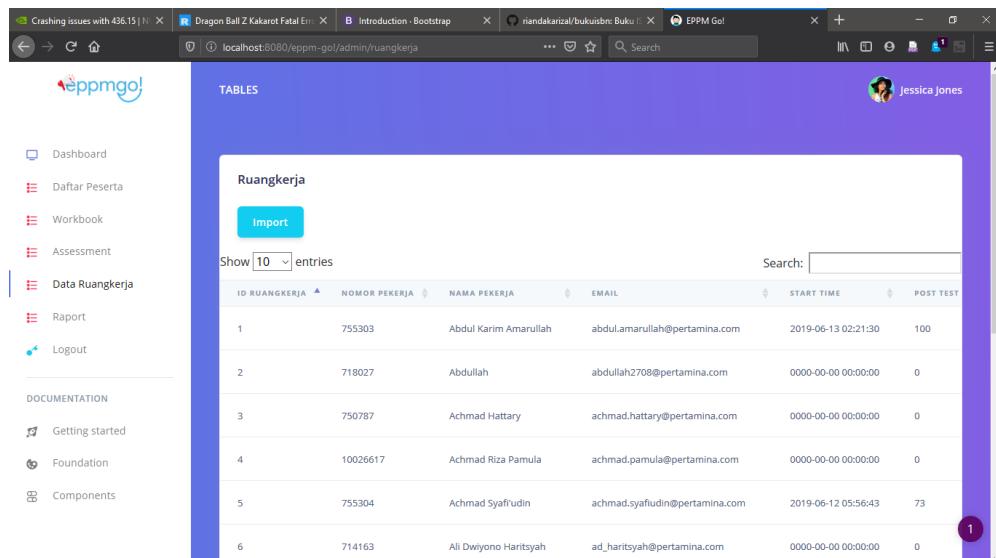
Untuk tombol hapus, kita harus selalu menggunakan konfirmasi sebelum dilakukannya penghapusan data. Ini berguna untuk mencegah dan meminimalisir terjadinya penghapusan data secara tidak sengaja, dan kita menggunakan modal untuk konfirmasinya seperti yang ditunjukkan gambar 143 di bawah ini

```
<script>  
function deleteConfirm(url){  
    $('#btn-delete').attr('href', url);  
    $('#deleteModal').modal();  
}  
</script>
```

Gambar 143 Views import\_ruangkerja.php

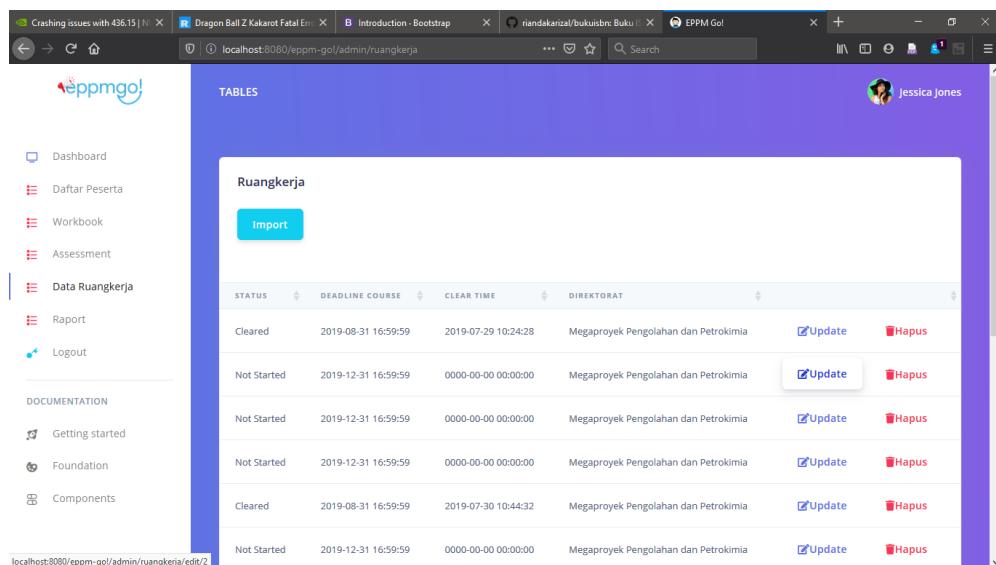
Setelah selesai dengan perulangan data, maka kita buat tombol untuk *import* agar menambahkan data tadi ke dalam *database*, dan juga sebuah tombol untuk membatalkan proses *import* data yang sedang kita lakukan.

Dengan ini selesai sudah langkah yang kita lakukan untuk membuat menu ruangkerja. Komponen dari menu ruangkerja adalah controllers, models, dan juga views. Dan dengan selesaiya isian pada gambar 141 di atas maka menu ruangkerja yang kita buat lengkap sudah. Selanjutnya adalah kita akan melakukan ujicoba ala ala apakah halaman yang baru saja kita buat bisa ditampilkan atau tidak. Lanjut buka url localhost:8080/eppm-go!/login dan lakukanlah *login*. Jika sudah *login*, langsung saja membuka menu ruangkerja



Gambar 144 Menu Ruangkerja

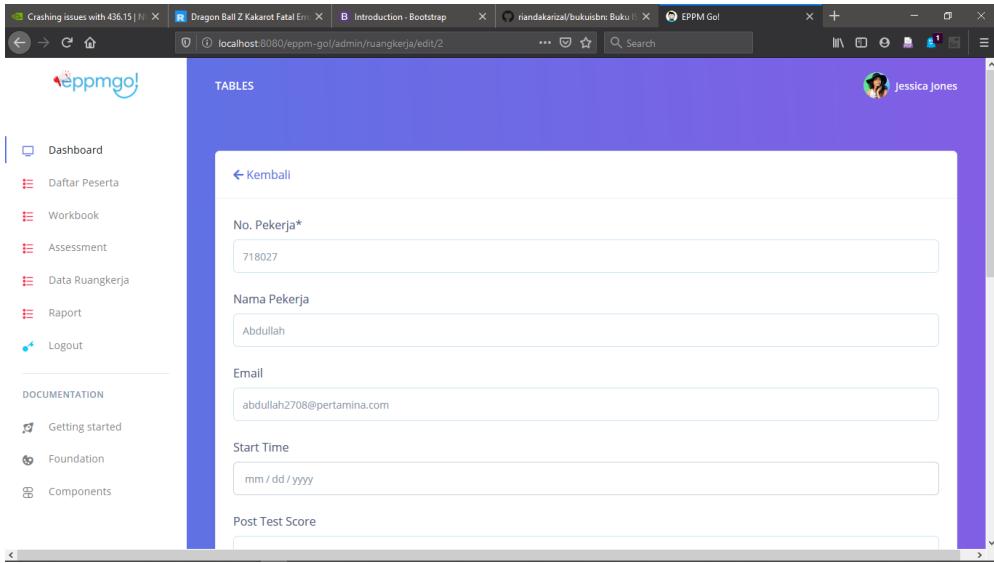
Setelah itu mari kita buka form *update* dengan menekan tombol *update* yang ada di samping data pada setiap baris data yang ditampilkan dan akan langsung mengarahkan ke halaman form *update*, seperti yang ditunjukkan pada gambar 145 dan juga gambar 146 berikut ini.



The screenshot shows a web browser window with multiple tabs open. The active tab is 'localhost:3080/eppm-go/admin/ruangkerja'. The page has a purple header with the title 'Ruangkerja' and a blue sidebar on the left containing navigation links like 'Dashboard', 'Daftar Peserta', 'Workbook', 'Assessment', 'Data Ruangkerja', 'Report', and 'Logout'. Below the sidebar is a table with columns: STATUS, DEADLINE COURSE, CLEAR TIME, DIREKTORAT, and two buttons: 'Update' and 'Hapus'. The 'Update' button is highlighted with a blue border. The table contains six rows of data. At the bottom of the table, there is a URL: 'localhost:3080/eppm-go/admin/ruangkerja/edit/2'.

STATUS	DEADLINE COURSE	CLEAR TIME	DIREKTORAT	Update	Hapus
Cleared	2019-08-31 16:59:59	2019-07-29 10:24:28	Megaprojek Pengolahan dan Petrokimia	<input checked="" type="button"/> Update	<input type="button"/> Hapus
Not Started	2019-12-31 16:59:59	0000-00-00 00:00:00	Megaprojek Pengolahan dan Petrokimia	<input checked="" type="button"/> Update	<input type="button"/> Hapus
Not Started	2019-12-31 16:59:59	0000-00-00 00:00:00	Megaprojek Pengolahan dan Petrokimia	<input checked="" type="button"/> Update	<input type="button"/> Hapus
Cleared	2019-08-31 16:59:59	2019-07-30 10:44:32	Megaprojek Pengolahan dan Petrokimia	<input checked="" type="button"/> Update	<input type="button"/> Hapus
Not Started	2019-12-31 16:59:59	0000-00-00 00:00:00	Megaprojek Pengolahan dan Petrokimia	<input checked="" type="button"/> Update	<input type="button"/> Hapus

Gambar 145 Tombol Update



Gambar 146 Halaman Form Update

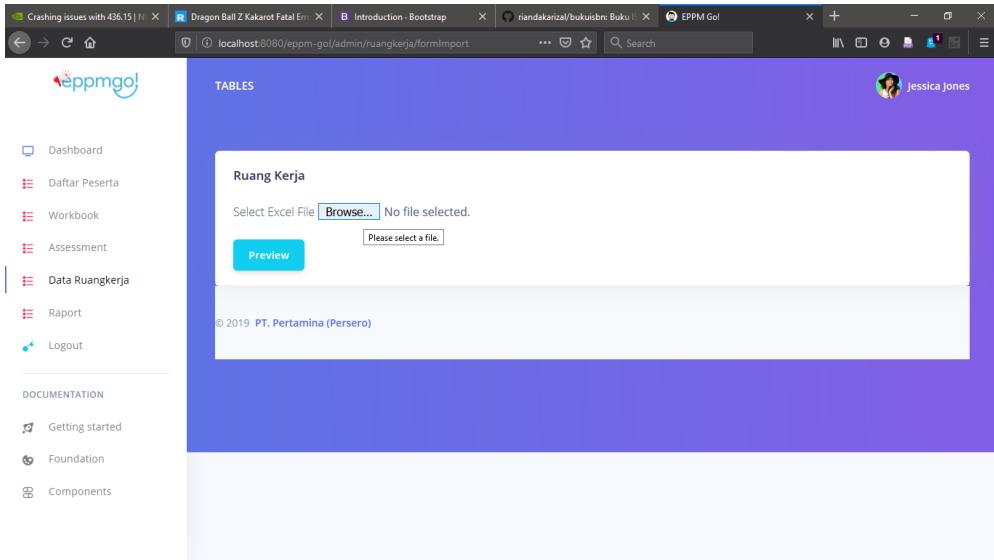
Karena sudah terbuka mari kita kembali ke halaman awal menu ruangkerja dan menguji cobakan halaman *import* dengan menekan tombol *import* yang terdapat pada kiri atas tabel seperti yang dapat kita lihat pada gambar 147 di bawah ini.

The screenshot shows a web-based application interface for 'EPPM Go!' with a purple header and sidebar. The sidebar on the left includes links for Dashboard, Daftar Peserta, Workbook, Assessment, Data Ruangkerja (highlighted with a blue border), Raport, and Logout. The main content area has a title 'TABLES' and a sub-section titled 'Ruangkerja'. It features a table with columns: ID RUANGKERJA, NOMOR PEKERJA, NAMA PEKERJA, EMAIL, START TIME, and POST TEST. The table contains 6 rows of data. An 'Import' button is located at the top left of the table area. A search bar and a dropdown menu for 'Show [10] entries' are also present. A user profile for 'Jessica Jones' is visible in the top right corner. A circled number '1' is positioned near the bottom right corner of the table area.

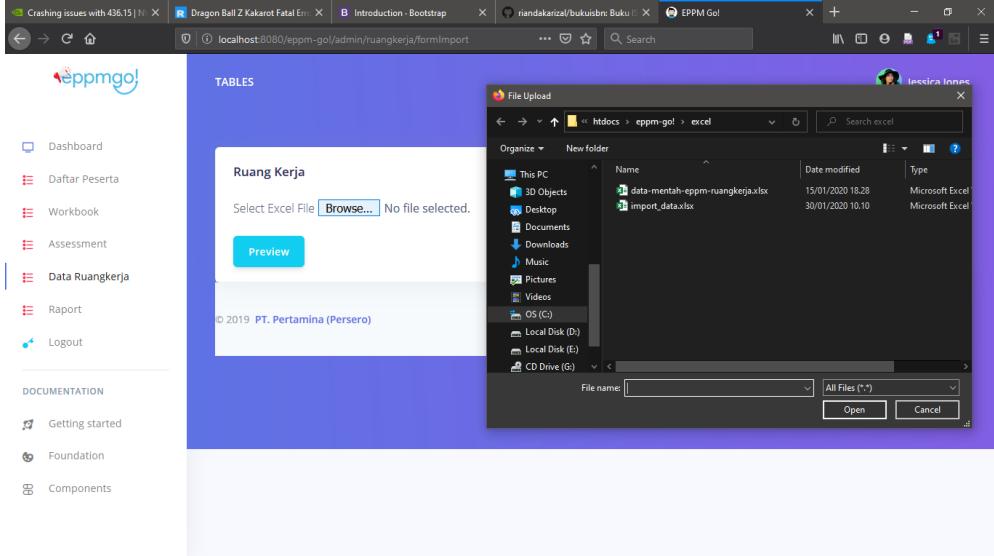
ID RUANGKERJA	NOMOR PEKERJA	NAMA PEKERJA	EMAIL	START TIME	POST TEST
1	755303	Abdul Karim Amarullah	abdulamarullah@pertamina.com	2019-06-13 02:21:30	100
2	718027	Abdullah	abdullah2708@pertamina.com	0000-00-00 00:00:00	0
3	750787	Achmad Hattary	achmad.hattary@pertamina.com	0000-00-00 00:00:00	0
4	10026617	Achmad Riza Pamula	achmad.pamula@pertamina.com	0000-00-00 00:00:00	0
5	755304	Achmad Syafiqudin	achmad.syafiqudin@pertamina.com	2019-06-12 05:56:43	73
6	714163	Ali Dwiyono Haritsyah	ad_haritsyah@pertamina.com	0000-00-00 00:00:00	0

Gambar 147 Tombol Import

Jika menekan tombol *import* maka akan mengarahkan kita menuju halaman form *import*. Dimana terlebih dahulu kita harus *upload file* excel yang akan dilakukan *import* dan tekan tombol *preview* apabila sudah memilih *file* excel yang ingin kita *import*. Seperti yang ditunjukkan pada gambar 148 dan juga gambar 149 berikut ini.



Gambar 148 Form Import



Gambar 149 Pilih File Excel

Jika sudah memilih *file* yang akan di-*import* maka langsung saja kita menekan tombol *preview* apakah benar ini adalah yang akan kita *import*. Seperti yang ditunjukkan pada gambar 150 berikut ini.

The screenshot shows a web-based application interface. On the left is a sidebar with a purple header and a white body containing navigation links: Dashboard, Daftar Peserta, Workbook, Assessment, Data Ruangkerja (selected), Raport, Logout, DOCUMENTATION, Getting started, Foundation, and Components. The main content area has a light blue header with the title 'Ruang Kerja'. Below it is a form with a 'Preview' button and a table. The table has columns: EMPLOYEE ID, EMPLOYEE NAME, USER EMAIL, START TIME, and POSTTER. There are 10 rows of data. Some cells contain numerical values (e.g., 100, 0, 73.3333, 93.3333) while others are empty or show 'None'. A red vertical bar highlights the 'POSTTER' column. A small number '1' is visible at the bottom right of the table.

EMPLOYEE ID	EMPLOYEE NAME	USER EMAIL	START TIME	POSTTER
755303	Abdul Karim Amarullah	abdulamarullah@pertamina.com	2019-06-13 2:21:30	100
718027	Abdullah	abdullah2708@pertamina.com	None	0
750787	Achmad Hattary	achmad.hattary@pertamina.com	None	0
10026617	Achmad Riza Pamula	achmad.pamula@pertamina.com	None	0
755304	Achmad Syaifudin	achmad.syaifudin@pertamina.com	2019-06-12 5:56:43	73.3333
714163	Ali Dwiyono Haritsyah	ad_haritsyah@pertamina.com	None	0
755305	Adi Kurnia Muktabar	adi.muktabar@pertamina.com	2019-06-13 6:13:36	93.3333
748861	Adithya Riswimbardi	adithya.riswimbardi@pertamina.com	None	0
755306	Adrian Satriaji Wiryawan	adrian.wiryawan@pertamina.com	2019-06-24 12:56:36	90

Gambar 150 Import File Excel

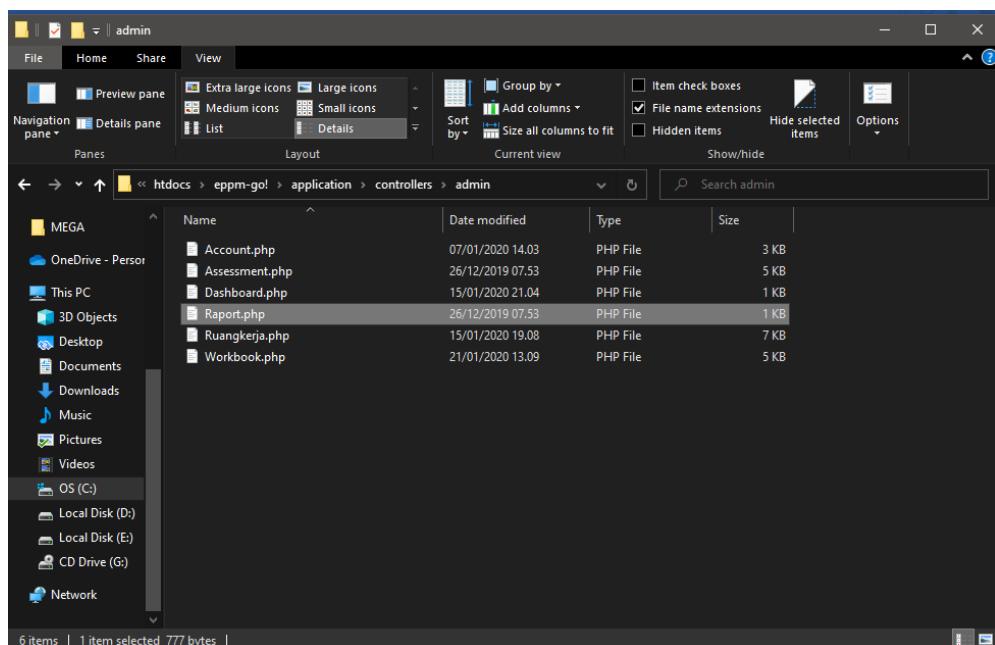
Dapat dilihat pada gambar 150 di atas, akan ada tanda bagi data yang kosong atau bernilai 0. Jika data tersebut sudah sesuai dengan data yang akan kita *import* maka langsung saja menuju akhir bawah dari halaman dan klik tombol import dan data tersebut akan ditambahkan ke dalam *database*.

Nah selesai sudah langkah-langkah yang harus kita lakukan untuk membuat menu ruangkerja, selanjutnya adalah kita akan membuat menu raport yang berfungsi untuk melihat nilai raport dari keseluruhan peserta pelatihan

## 2.6. CRUD Menu Raport

Di dalam menu raport, kita akan menampilkan semua data nilai dari peserta pelatihan dan juga membuat fitur dimana nilai nilai tersebut dapat kita unduh. Data nilai dapat diunduh dengan ekstensi .csv, .xlsx, .pdf, ataupun di-print yang mana ini akan memudahkan admin untuk proses pelaporan, sehingga admin tidak akan selalu membuka *website* untuk melihat nilai raportnya.

Untuk menu raport, tidak akan serumit dan sepanjang 4 menu yang telah kita sebelumnya. Untuk menu ini, hanya akan menjadi menu tersimpel karena kita hanya akan menampilkan data nilai dari para peserta. Langsung saja kita membuat *file* Raport.php untuk controllers di folder admin pada direktori controllers. Seperti yang ditunjukkan pada gambar 151 berikut ini.



Gambar 151 File Raport.php

Setelahnya, kita langsung saja memberikan isian pada *file* Raport.php seperti yang ditunjukkan pada gambar 152 berikut ini.

```
<?php
defined('BASEPATH') OR exit ('No direct script access allowed');

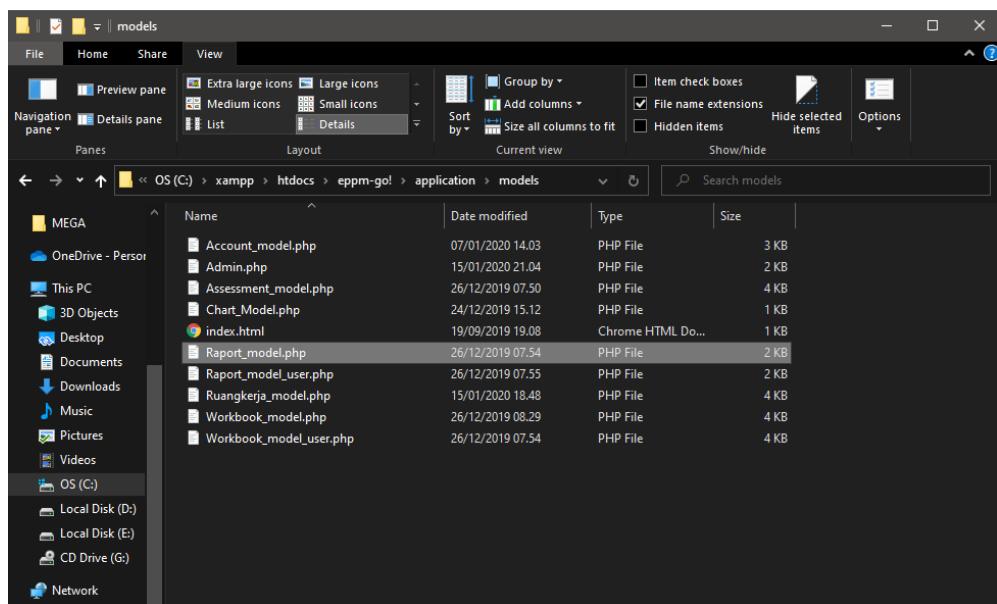
class Raport extends CI_Controller
{
    public function __construct()
    {
        parent::__construct();
        $this->load->model("raport_model"); //load model workbook
        $this->load->library('form_validation'); //Load library form validation
        $this->load->helper(array('form', 'url'));
        if($this->raport_model->is_role() != "admin"){
            redirect("index.php/login");
        }
    }

    public function index()
    {
        $data["raport"] = $this->raport_model->getAll(); //ambil data dari model
        // print_r($data["raport"]); die;
        $this->load->view("admin/raport/list_raport", $data); //Load view data model ke workbook
    }
}
```

Gambar 151 Controllers Raport.php

Pertama kita membuat sebuah *class* dengan nama Raport dengan menggunakan *library* dari CI\_Controller, selanjutnya adalah membuat *function* \_\_construct() yang akan me-load models rapport\_model, load form validasi. Lalu setelah me-load models rapport\_model maka akan menggunakan *function* yang ada pada rapport\_model yaitu is\_role() yang berfungsi untuk mengecek role dari pengguna yang masuk. Apabila role yang masuk bukan admin maka akan langsung diarahkan menuju halaman login.

Dan selesai sudah untuk *file controllers* Raport.php, sangat simple bukan? Maka langsung kita lanjutkan untuk membuat *file models* Raport\_model.php yang berfungsi untuk melakukan manipulasi data dari *database* menuju aplikasi *website* ataupun sebaliknya. Buatlah *file* tersebut di dalam folder models di direktori eppm-go! seperti yang ditunjukkan pada gambar 152 berikut ini.



Gambar 152 Models Raport\_model.php