

Admin Template Codeigniter 4

Jagowebdev.com

I. Pengantar

PHP Admin Template Jagowebdev dikembangkan untuk memudahkan dalam pengembangan aplikasi. Anda tidak perlu membuat dari awal manajemen usernya, seperti bagaimana user login, hak akses pada user tersebut, dll sehingga Anda dapat fokus dalam pengembangan aplikasi.

Berikut ini beberapa hal penting yang perlu diketahui:

II. Install Aplikasi

Untuk install aplikasi ikuti langkah berikut:

- a. Copy file php ke folder htdocs, misal jika di copy ke folder htdocs/admin_template maka struktur foldernya akan tampak seperti berikut:

DATA (D:) > xampp > htdocs > admin_template >

| Name | Type |
|----------------------------------|---------------|
| app | File folder |
| public | File folder |
| system | File folder |
| writable | File folder |
| .env | ENV File |
| .htaccess | HTACCESS File |
| admin-template-codeigniter-4.sql | SQL File |
| index.php | PHP File |
| license.txt | Text Document |
| README.md | MD File |
| spark | File |

- b. Buat database dengan nama yang dikehendaki, misal penjualan, selanjutnya load file admin-template-codeigniter-4.sql yang disertakan pada file download ke database tersebut.
- c. Edit file app/Config/App.php edit bagian \$baseURL sesuai dengan url dimana aplikasi diinstall dan edit app/Config/Database.php sesuai dengan konfigurasi database Anda. Untuk variabel lainnya bersifat opsional. Misal jika kita ekstrak file ke folder htdocs/admin_template seperti contoh diatas, maka \$baseURL nya adalah http://localhost/admin_template/

d. System requirements

Codeigniter 4 mensyaratkan versi PHP minimal adalah versi 7.2 sehingga untuk menjalankan Admin Template ini juga diperlukan PHP versi > 7.2. Direkomendasikan menggunakan PHP versi > 7.4 yang bisa diinstal menggunakan xampp portable versi 7.4.10 yang bisa didownload di <https://sourceforge.net/projects/xampp/files/XAMPP%20Windows/>

Mengatasi Error:

Biasanya ketika mulai menjalankan aplikasi, akan muncul pesan error

Fatal error: Uncaught Error: Call to undefined function CodeIgniter\locale_set_default() in...

Error ini disebabkan karena library Internationalization extension (Intl) belum aktif. Untuk mengaktifkannya, buka file php.ini dan uncomment (hilangkan tanda titik koma didepan) bagian extension=intl

Pada PHP 8, jangan lupa juga untuk mengaktifkan ekstensi GD dengan uncomment bagian extension=gd

Setelah itu **jangan lupa restart server** (apache)

Catatan: Agar lebih aman, sebaiknya menggunakan database MariaDB bukan MySQL, contoh bundle yang menggunakan database MariaDB adalah XAMPP

A. Alur Program

Alur program pada aplikasi Admin Template Codeigniter 4 ini adalah:

Router -> Filter (app/Filters/Bootstrap.php) -> Controller (app/Controllers/BaseController.php) -> Controller

Filter akan diakses sebelum Codeigniter mengakses controller, pada filter tersebut terdapat beberapa script konfigurasi umum, seperti custom CSRF, mendefinisikan nama module berdasarkan URL, dll.

Selanjutnya pada BaseController.php terdapat berbagai pengaturan global yang akan diterapkan ke semua controller.

III. Setup Awal Aplikasi

1. Membuat Menu

a. Membuat menu

Untuk membuat menu, klik menu website > Menu. Di halaman menu, klik Tambah Menu, isikan parameter, kemudian klik Submit

Data Menu

[+ Tambah Menu](#)

Tambah Menu [X]

Nama Menu:

URL:

Aktif: ☒ Jika tidak aktif, semua children tidak akan dimunculkan

Module: Untuk highlight menu dan parent

Use icon:

Role: Untuk memunculkan menu, assign role ke menu

[Cancel](#) [Submit](#)

Penting diperhatikan bahwa agar nantinya menu dapat terhighlight ketika module terkait menu tersebut dibuka, maka pada bagian isian Module, pilih module yang sesuai, yang pada contoh diatas kita isi Module produk. Jika module belum dibuat, Anda dapat membuatnya terlebih dahulu pada menu Module, atau Anda bagian Module dapat dikosongkan terlebih dahulu, kemudian mengisinya pada saat melakukan editing menu.

Menu yang dibuat akan berada di posisi paling atas pada hierarki menu, Anda dapat mengubah urutan dengan menggeser menu

Data Menu

[+ Tambah Menu](#)

| | | |
|--------------------|--|--|
| + Website | | |
| + Forms - CRUD | | |
| Data Tables | | |
| Data Tables (Ajax) | | |
| Produk | | |

[Save](#)

Selanjutnya klik Save.

b. Membuat Submenu

Untuk membuat submenu, caranya, buat menu seperti langkah sebelumnya, selanjutnya geser menu yang telah dibuat tadi menjadi submenu dari menu yang diinginkan, contoh kita buat menu E-Book dan kita jadikan submenu dari menu Produk:



Selanjutnya klik Save.

c. Menampilkan menu

Menu tidak serta merta tampil, agar dapat tampil, menu perlu di assign terlebih dahulu ke role, untuk assign ke role, klik menu Website > Assign Role > Menu Role.

d. Menampilkan Highlight Menu ketika halaman dibuka

Sama seperti pada penjelasan membuat menu, agar menu terhighlight ketika membuka halaman tertentu yang artinya module tertentu, maka kita perlu meng-assign menu tersebut ke module, caranya edit menu yang ingin diassign modulnya kemudian pada bagian Module, pilih module yang ingin diassign ke menu tersebut

Edit Menu

Nama Menu

Produk

URL

produk

Aktif

☒

Jika tidak aktif, semua children tidak akan dimunculkan


Module

produk | Produk (Aktif) ▼

Untuk highlight menu dan parent

Use icon

Ya ▼



Role

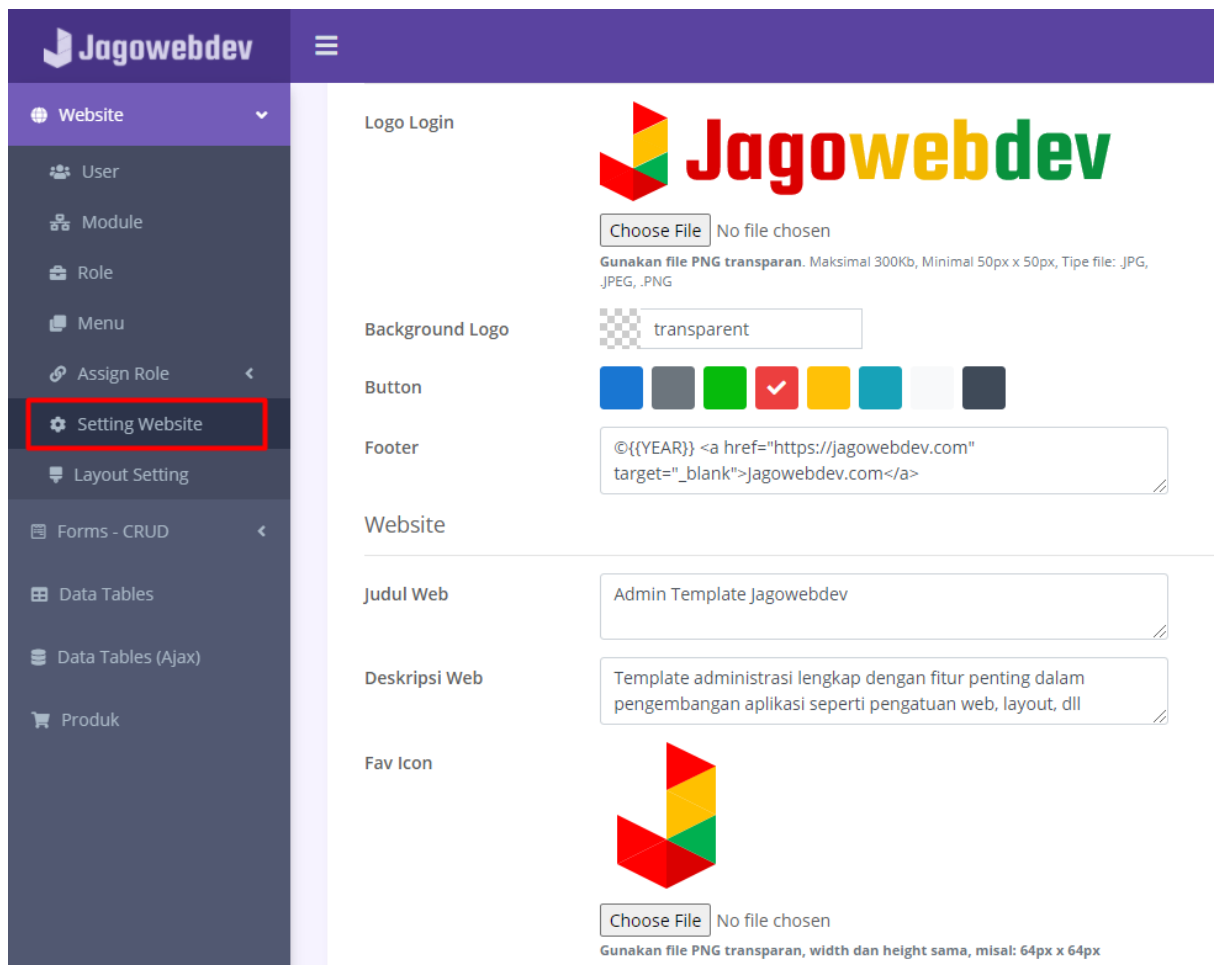
Untuk memunculkan menu, assign role ke menu

Cancel

Submit

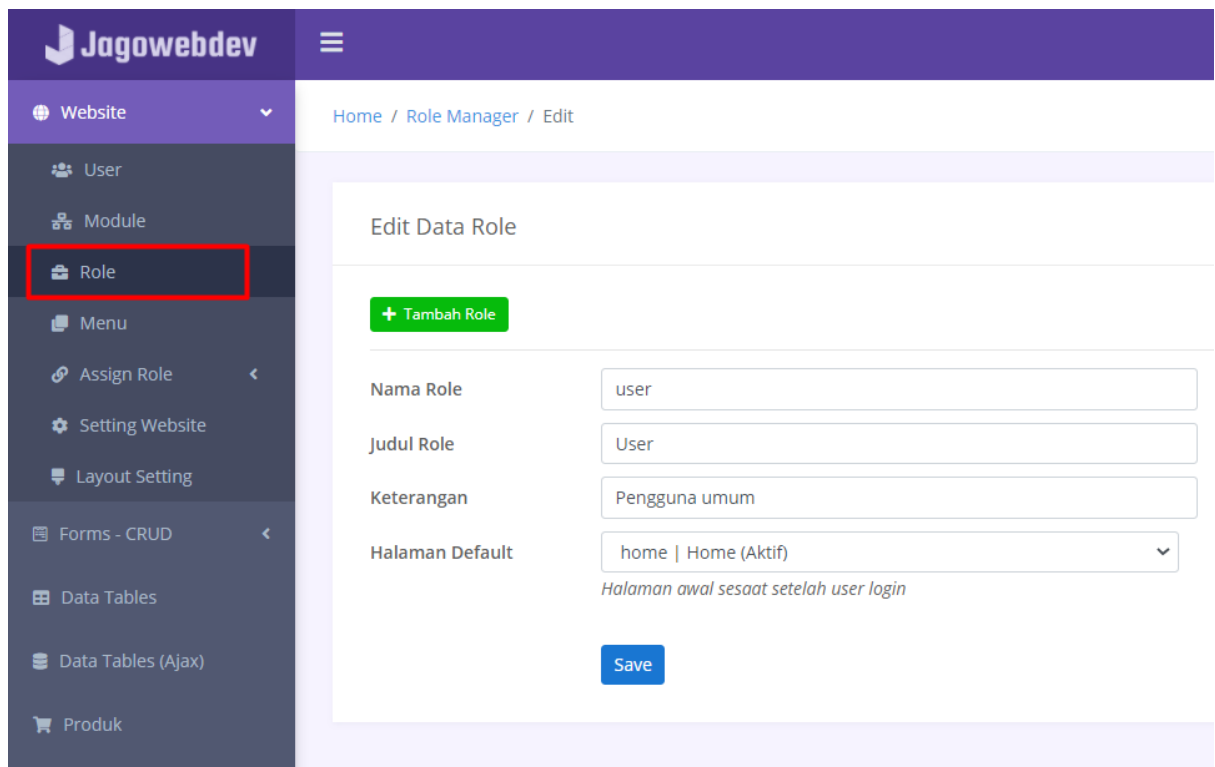
2. Mengubah logo

Pengaturan logo, baik logo pada favicon, halaman login, maupun halaman aplikasi dapat dilakukan melalui menu Setting Website, contoh sebagai berikut:



3. Membuat halaman default

Ketika user login, dapat langsung diarahkan ke halaman default sesuai dengan role user tersebut. Untuk membuat halaman default tersebut, masuk ke menu role, edit role yang ada kemudian pada halaman default pilih halaman default untuk role tersebut.



4. Mode Pengembangan

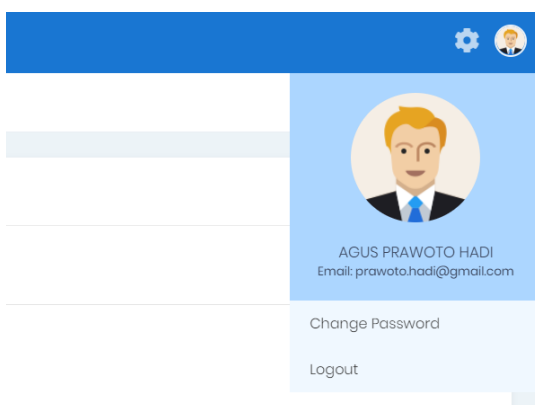
Secara default jika terjadi error, maka aplikasi akan memunculkan pesan error tersebut, Anda dapat menonaktifkan pesan error tersebut dan hanya menampilkan pesan error secara umum dengan cara mengedit file .htaccess kemudian ganti SetEnv CI_ENVIRONMENT development menjadi SetEnv CI_ENVIRONMENT production.

5. Login

Login pertama kali adalah username: admin, password: admin

6. Ubah Password

Untuk menjaga kerahasiaan dan keamanan password, ubah password hanya bisa dilakukan oleh user pemilik password sendiri. Caranya yaitu login ke akun, kemudian masuk ke menu Change Password yang ada di menu akun pojok kanan atas



IV. Membuat Module

Pada admin template ini, module tidak bisa dibuat secara otomatis, melainkan dibuat secara manual sesuai dengan lingkungan pengembangan CodeIgniter 4

A. Module Tanpa Database

Seperti ketika mengembangkan aplikasi menggunakan codeigniter, untuk membuat module/halaman, kita perlu membuat controller, misal kita akan membuat module produk. Pertama kita buat file controller dengan nama Produk.php di folder app/Controllers sebagai berikut:

| DATA (D:) > xampp > htdocs > admin_template > app > Controllers | | |
|---|-------------|------|
| Name | Type | Size |
| Builtin | File folder | |
| Base.php | PHP File | 4 KB |
| BaseController.php | PHP File | 8 KB |
| Data_tables.php | PHP File | 5 KB |
| Data_tables_ajax.php | PHP File | 5 KB |
| Home.php | PHP File | 1 KB |
| Image_upload.php | PHP File | 5 KB |
| Input_dinamis.php | PHP File | 5 KB |
| Login.php | PHP File | 3 KB |
| Multiple_fileupload.php | PHP File | 7 KB |
| Options_dinamis.php | PHP File | 6 KB |
| Produk.php | PHP File | 1 KB |
| Welcome.php | PHP File | 1 KB |

Misal file Produk.php tersebut kita isi script sebagai berikut:

```
<?php
namespace App\Controllers;

class Produk extends BaseController
{
    public function index()
    {
        echo 'Tes Module Produk';
    }
}
```

Selanjutnya, agar module dapat diakses, kita perlu mendaftarkan module ke sistem, caranya masuk kemenu module kemudian tambahkan module produk sebagai berikut:

Jagowebdev

Website / Module Manager / Add

Tambah Module

[+ Tambah Module](#) [Daftar Module](#)

Nama Module: Sesuai nama yang ada di URL

Judul Module:

Deskripsi:

Status:

[Save](#)

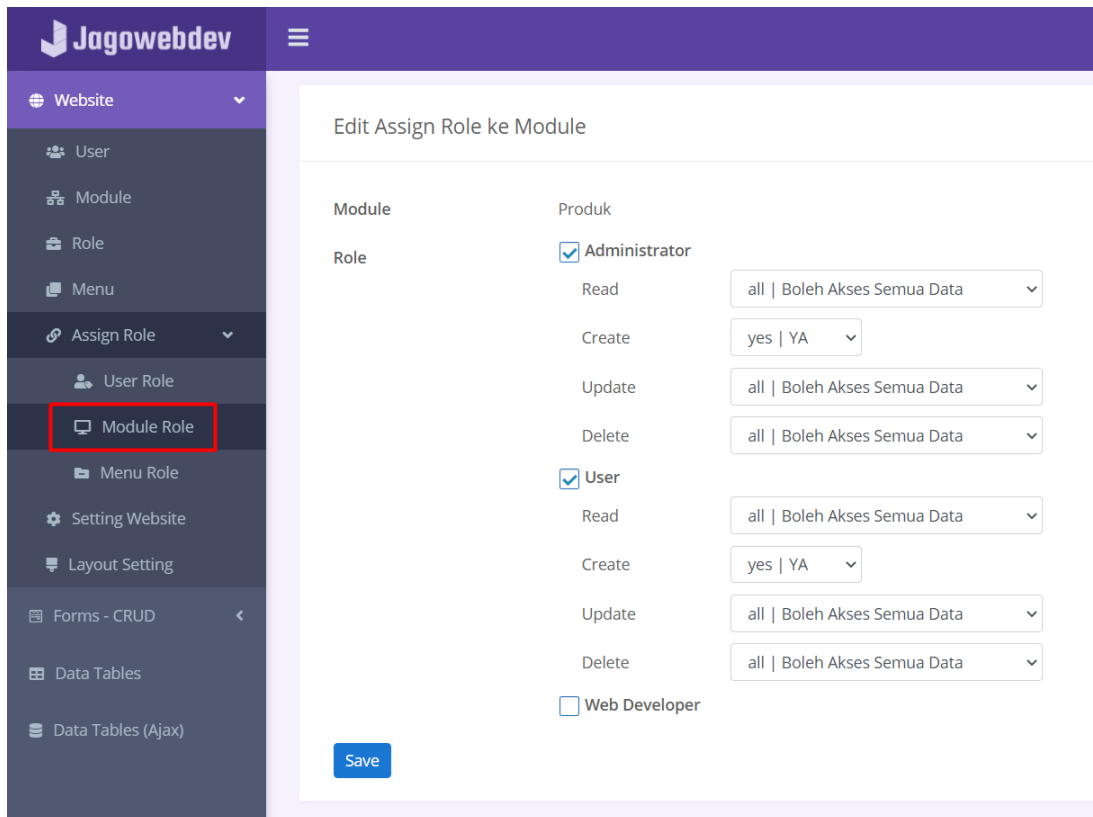
Selanjutnya kita tentukan, siapa yang boleh mengakses module tersebut, caranya masuk ke menu Assign Role, sub menu Module Role seperti tampak pada gambar berikut:

Jagowebdev

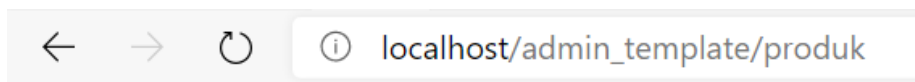
Website / Assign Role / Module Role

| | | | | |
|----|---------------------|----------------------|----------------------------------|---|
| 8 | menu-role | Menu - Role | Administrator | Edit Detail |
| 9 | image-upload | Image Upload | Administrator User | Edit Detail |
| 15 | setting | Web Setting | Administrator User | Edit Detail |
| 16 | setting-web | Setting Web | Administrator | Edit Detail |
| 21 | options-dinamis | Options Dinamis | Administrator User | Edit Detail |
| 22 | input-dinamis | Input Dinamis | Administrator User | Edit Detail |
| 25 | home | Home | Administrator User Web Developer | Edit Detail |
| 26 | multiple-fileupload | Multiple File Upload | Administrator User | Edit Detail |
| 27 | datatables | Data Tables | Administrator User Web Developer | Edit Detail |
| 28 | datatables-ajax | Data Tables Ajax | Administrator User Web Developer | Edit Detail |
| 29 | produk | Produk | | Edit Detail |

Selanjutnya pilih tombol edit pada module produk. Pada halaman Edit Assign Role ke Module, pilih role yang ingin diberi akses sebagai berikut:



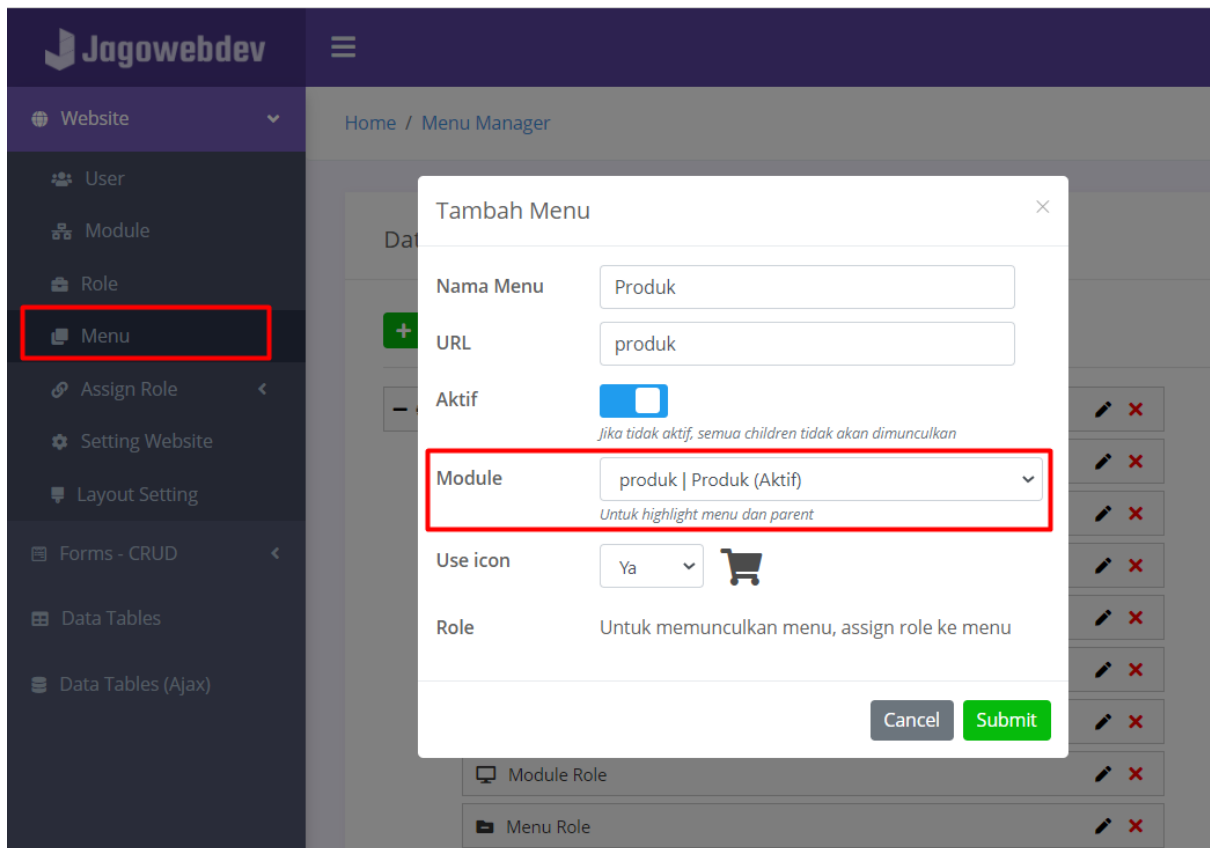
Setelah telah berhasil, maka module tersebut sudah bisa diakses, pada contoh ini, karena aplikasi saya letakkan di folder `htdocs/admin_template`, maka alamat module produk adalah http://localhost/admin_template/produk, contoh sebagai berikut:



Tes Module Produk

B. Membuat Menu Untuk Module

Agar module mudah diakses, maka alamat module tersebut perlu kita tambahkan pada menu, caranya, masuk ke menu "Menu" kemudian tambahkan klik Tambah Menu, pada isian menu, isikan detail menu, misal seperti contoh berikut:

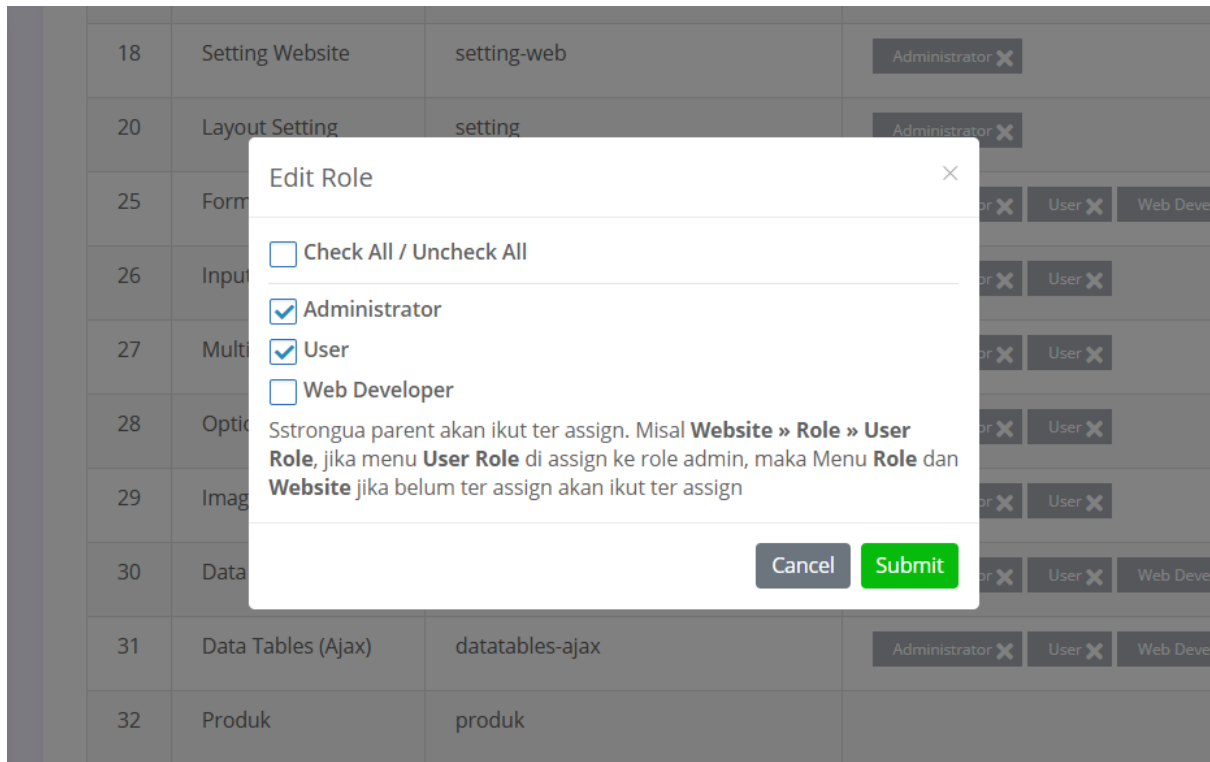


Penting diperhatikan bahwa pada bagian Module kita pilih module produk yang telah kita buat sebelumnya, hal ini bertujuan agar ketika module/halaman produk dibuka, menu Produk dapat ter highlight.

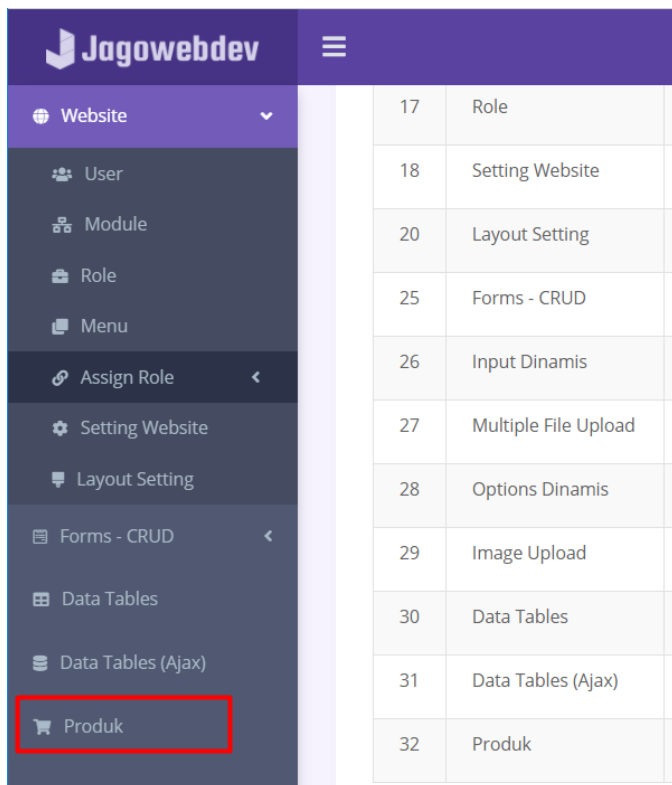
Menu yang telah kita buat tidak serta merta muncul di daftar menu, untuk memunculkannya, kita perlu meng assign menu tersebut ke role yang ada, untuk melakukannya, masuk ke menu Assign Role > Menu Role, kemudian pada menu produk, klik menu Edit sebagai berikut:

| | | | | | |
|-----------------|----|----------------------|--------------------------------------|----------------------------------|------|
| Website | 18 | Setting Website | setting-web | Administrator | Edit |
| User | 20 | Layout Setting | setting | Administrator | Edit |
| Module | 25 | Forms - CRUD | input-dinamis | Administrator User Web Developer | Edit |
| Role | 26 | Input Dinamis | input-dinamis?action=add | Administrator User | Edit |
| Menu | 27 | Multiple File Upload | multiple-fileupload?action=edit&id=1 | Administrator User | Edit |
| Assign Role | 28 | Options Dinamis | options-dinamis?action=edit&id=2 | Administrator User | Edit |
| User Role | 29 | Image Upload | image-upload?action=edit&id=1 | Administrator User | Edit |
| Module Role | 30 | Data Tables | datatables | Administrator User Web Developer | Edit |
| Setting Website | 31 | Data Tables (Ajax) | datatables-ajax | Administrator User Web Developer | Edit |
| Layout Setting | 32 | Produk | produk | | Edit |
| Forms - CRUD | | | | | |
| Data Tables | | | | | |

Selanjutnya pilih role yang ingin di assign dan simpan, misal seperti gambar berikut:



Setelah berhasil maka menu akan muncul disebelah kiri (jangan lupa refresh halaman)



C. Module Dengan Database

Pada codeigniter, untuk berinteraksi dengan database, kita perlu membuat model. Pada contoh kali ini kita buat model dengan nama ProdukModel.php dan kita letakkan di dalam folder app/Models sebagai berikut:

| | | |
|--|--------------|------|
| DATA (D:) > xampp > htdocs > admin_template > app > Models > | | |
| Name | Type | Size |
| Builtin | File folder | |
| .gitkeep | GITKEEP File | 0 KB |
| BaseModel.php | PHP File | 6 KB |
| DataTablesAjaxModel.php | PHP File | 5 KB |
| DataTablesModel.php | PHP File | 4 KB |
| ImageUploadModel.php | PHP File | 4 KB |
| InputDinamisModel.php | PHP File | 2 KB |
| MultipleFileuploadModel.php | PHP File | 7 KB |
| OptionsDinamisModel.php | PHP File | 6 KB |
| ProdukModel.php | PHP File | 1 KB |

Sebagai contoh, file ProdukModel.php kita isi script sebagai berikut:

```
<?php
namespace App\Models;

class ProdukModel extends \App\Models\BaseModel
{
    public function __construct() {
        parent::__construct();
    }

    public function getDataProduk() {
        $sql = 'SELECT * FROM produk';
        $result = $this->db->query($sql)->getResultArray();
        return $result;
    }
}
```

Selanjutnya kita ubah file controller Produk.php menjadi berikut:

```
<?php
namespace App\Controllers;
use App\Models\ProdukModel;

class Produk extends BaseController
{
    private $produkModel;

    public function __construct() {
        parent::__construct();
    }
}
```

```

        $this->produkModel = new ProdukModel;
    }

    public function index()
    {
        $produk = $this->produkModel->getDataProduk();
        echo '<pre>'; print_r($produk);
    }
}

```

Jika kita buka module produk, hasil yang kita peroleh adalah sebagai berikut:



```

Array
(
    [0] => Array
        (
            [id_produk] => 1
            [nama_produk] => Bluetooth Multi-Device Keyboard K480
            [deskripsi_produk] => Keyboard meja wireless untuk komputer, tablet, dan smartphone
        )

    [1] => Array
        (
            [id_produk] => 2
            [nama_produk] => USB Unifying Receiver
            [deskripsi_produk] => Receiver USB yang bisa digunakan untuk sebuah mouse atau keyboard unifying
        )

    [2] => Array
        (
            [id_produk] => 3
            [nama_produk] => M590 Multi-Device Silent
            [deskripsi_produk] => Mouse wireless hening untuk power user
        )

)

```

D. Handling Error Data Tidak Ditemukan

Jika data tidak ditemukan, Anda dapat dengan mudah memunculkan pesan error bahwa data tidak ditemukan, yaitu cukup dengan memanggil method `$this->errorDataNotFound()` method ini ada di file `app/Controllers/BaseController.php`

Sebagai contoh script controller Produk.php kita ubah sebagai berikut:

```

<?php
namespace App\Controllers;
use App\Models\ProdukModel;

class Produk extends BaseController
{
    private $produkModel;

    public function __construct() {
        parent::__construct();
        $this->produkModel = new ProdukModel;
    }

    public function index()

```

```

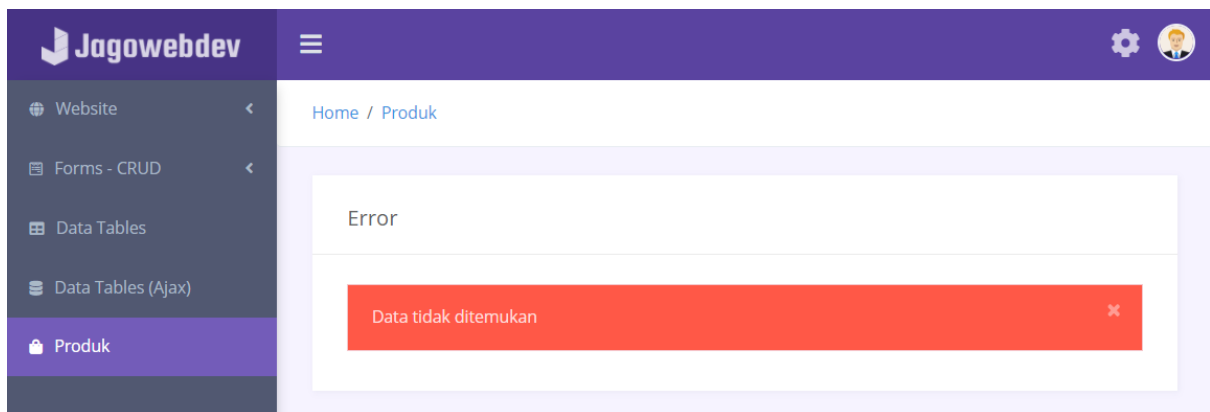
{
    $data = $this->data;
    $data['result'] = $this->produkModel->getDataProduk();

    if (!$data['result'])
        $this->errorDataNotFound();

    $this->view('produk-result.php', $data);
}
}

```

Maka, jika data produk tidak ditemukan, akan muncul pesan error sebagai berikut:




















E. Module Dengan View

Output module yang kita bahas sebelumnya merupakan output apa adanya. Anda dapat menggunakan header dan footer bawaan aplikasi sehingga tampilan module lebih menarik, untuk menggunakannya, Anda cukup memanggil method `$this->view()`. Method ini merupakan method bawaan aplikasi yang didesain agar dapat otomatis me-load file `header.php` dan `footer.php` yang ada di folder `app/Views/themes/modern/header`. Source code method ini dapat dilihat di file `app/Controllers/BaseController.php`

Selanjutnya mari kita membuat file view yang kita beri nama `produk-result.php`, letakkan file tersebut di folder `app/Views/themes/modern` sebagai berikut:

DATA (D:) > xampp > htdocs > admin_template > app > Views > themes > modern

| Name | Type | Size |
|--|-------------|------|
|  builtin | File folder | |
|  data-tables-ajax-result.php | PHP File | 2 KB |
|  data-tables-result.php | PHP File | 2 KB |
|  error.php | PHP File | 1 KB |
|  error-data-notfound.php | PHP File | 1 KB |
|  footer.php | PHP File | 1 KB |
|  header.php | PHP File | 8 KB |
|  image-upload-form.php | PHP File | 5 KB |
|  image-upload-result.php | PHP File | 2 KB |
|  multiple-fileupload-form.php | PHP File | 8 KB |
|  multiple-fileupload-result.php | PHP File | 3 KB |
|  options-dinamis-form.php | PHP File | 5 KB |
|  options-dinamis-result.php | PHP File | 3 KB |
|  penghadap-form.php | PHP File | 3 KB |
|  penghadap-result.php | PHP File | 2 KB |
|  produk-result.php | PHP File | 1 KB |
|  welcome.php | PHP File | 1 KB |

Selanjutnya kita isi file produk-result.php dengan script sebagai berikut:

```
<table class="table table-striped table-bordered table-hover">
  <thead>
    <tr>
      <th>No</th>
      <th>Nama Produk</th>
      <th>Deskripsi Produk</th>
    </tr>
  </thead>
  <tbody>
    <?php
    $no = 1;
    foreach ($result as $val) {
      echo ' <tr>
        <td>' . $no . '</td>
        <td>' . $val['nama_produk'] . '</td>
        <td>' . $val['deskripsi_produk'] . '</td>
      </tr>';
      $no++;
    }
    ?>
  </tbody>
</table>
```

Sedangkan script pada controller Produk.php kita ubah menjadi berikut:


```

<?php
namespace App\Controllers;
use App\Models\ProdukModel;

class Produk extends BaseController
{
    private $produkModel;

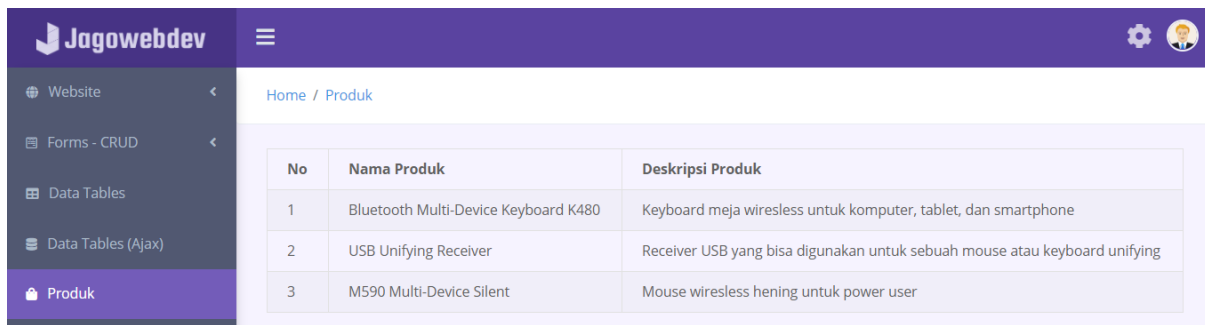
    public function __construct() {
        parent::__construct();
        $this->produkModel = new ProdukModel;
    }

    public function index()
    {
        $data = $this->data;
        $data['result'] = $this->produkModel->getDataProduk();
        $this->view('produk-result.php', $data);
    }
}

```

Perhatikan bahwa pada file controller Produk.php kita definisikan variabel \$data dengan properti \$this->data. Properti \$this->data ini berisi berbagaimacam data yang dapat kita gunakan pada file view, diantaranya \$config dan \$session, data data ini dapat dilihat pada script app/Controllers/BaseController.php selanjutnya variabel \$data kita isi data result hasil dari query tabel produk dari file model

Jika kita jalankan module produk, hasil yang kita peroleh adalah sebagai berikut:



| No | Nama Produk | Deskripsi Produk |
|----|--------------------------------------|--|
| 1 | Bluetooth Multi-Device Keyboard K480 | Keyboard meja wireless untuk komputer, tablet, dan smartphone |
| 2 | USB Unifying Receiver | Receiver USB yang bisa digunakan untuk sebuah mouse atau keyboard unifying |
| 3 | M590 Multi-Device Silent | Mouse wireless hening untuk power user |

Perhatikan bahwa breadcrumb yang muncul adalah Home / Produk, menyesuaikan dengan nama module yang didaftarkan

Agar tampilan lebih menarik, kita tambahkan beberapa elemen dan style pada output diatas sebagai berikut:

```

<div class="card">
    <div class="card-header">
        <h5 class="card-title"><?=$current_module['judul_module']?></h5>
    </div>

    <div class="card-body">
        <table class="table table-striped table-bordered table-hover">
            <thead>
                <tr>

```

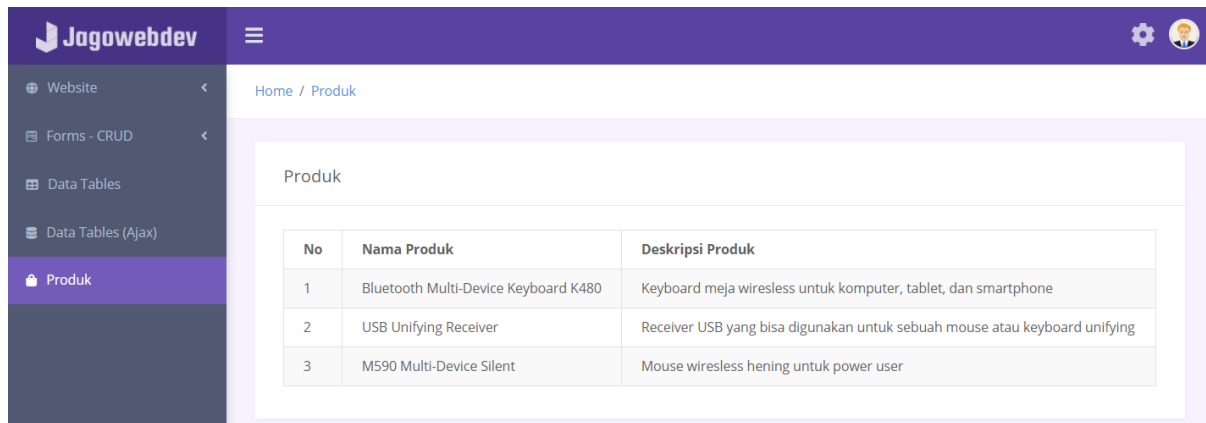
```

        <th>No</th>
        <th>Nama Produk</th>
        <th>Deskripsi Produk</th>
    </tr>
</thead>
<tbody>
<?php
$no = 1;
foreach ($result as $val) {
    echo ' <tr>
        <td>' . $no . '</td>
        <td>' . $val['nama_produk'] . '</td>
        <td>' . $val['deskripsi_produk'] . '</td>
    </tr>';
    $no++;
}
?>
</tbody>
</table>
</div>
</div>

```

Class CSS card, card-header, card-body merupakan style bawaan aplikasi admin template, anda dapat memodifikasi style ini pada file public/themes/modern/buiulin/css/site.css

Output yang dihasilkan adalah sebagai berikut:



The screenshot shows the Jagowebdev admin dashboard. On the left is a sidebar menu with options: Website, Forms - CRUD, Data Tables, Data Tables (Ajax), and Produk (highlighted). The main content area shows a breadcrumb 'Home / Produk' and a section titled 'Produk'. Below this is a table with the following data:

| No | Nama Produk | Deskripsi Produk |
|----|--------------------------------------|--|
| 1 | Bluetooth Multi-Device Keyboard K480 | Keyboard meja wireless untuk komputer, tablet, dan smartphone |
| 2 | USB Unifying Receiver | Receiver USB yang bisa digunakan untuk sebuah mouse atau keyboard unifying |
| 3 | M590 Multi-Device Silent | Mouse wireless hening untuk power user |

F. Edit, Delete, dan Tambah Data

Bagian ini akan membahas bagaimana membuat script untuk edit, delete, dan tambah data.

F.1. Menambahkan tombol edit dan delete

Pertama tama mari kita bahas cara membuat tombol untuk edit dan delete data.

Untuk menambahkan tombol edit dan delete pada tabel dimana data ditampilkan, kita dapat menggunakan fungsi `button_action()`, fungsi ini ada di file `app/Helpers/html_helper.php`, untuk menyertakan file tersebut, kita cukup menjalankan perintah `helper('html')`

Sebagai contoh script table pada file `app/views/produk-result.php` kita ubah menjadi berikut:

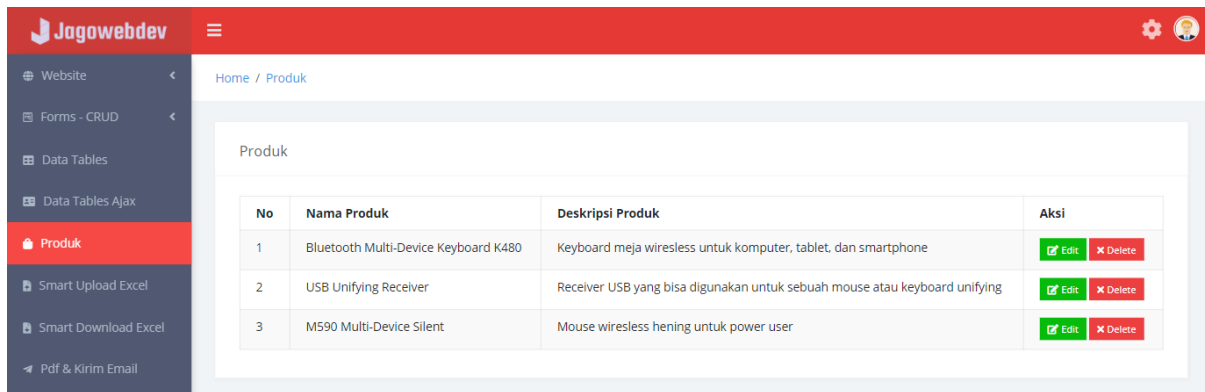
```

<div class="card">
  <div class="card-header">
    <h5 class="card-title"><?=$current_module['judul_module']?></h5>
  </div>

  <div class="card-body">
    <div class="table-responsive">
      <table class="table table-striped table-bordered table-hover">
        <thead>
          <tr>
            <th>No</th>
            <th>Nama Produk</th>
            <th>Deskripsi Produk</th>
            <th>Aksi</th>
          </tr>
        </thead>
        <tbody>
          <?php
            helper('html');
            $no = 1;
            foreach ($result as $val) {
              echo
                '
                <tr>
                  <td>' . $no . '</td>
                  <td>' . $val['nama_produk'] . '</td>
                  <td>' . $val['deskripsi_produk'] . '</td>
                  <td>' . btn_action([
                    'edit' => ['url' =>
                      '/edit?id=' . $val['id_produk']]
                    , 'delete' => ['url' => ''
                      , 'id' =>
                        $val['id_produk']
                      , 'delete-
                        title' => 'Hapus data produk: <strong>' . $val['nama_produk'] . '</strong> ?'
                      ]
                    ]) .
                  </td>
                </tr>'
                . $no++;
            }
          <?>
        </tbody>
      </table>
    </div>
  </div>
</div>

```

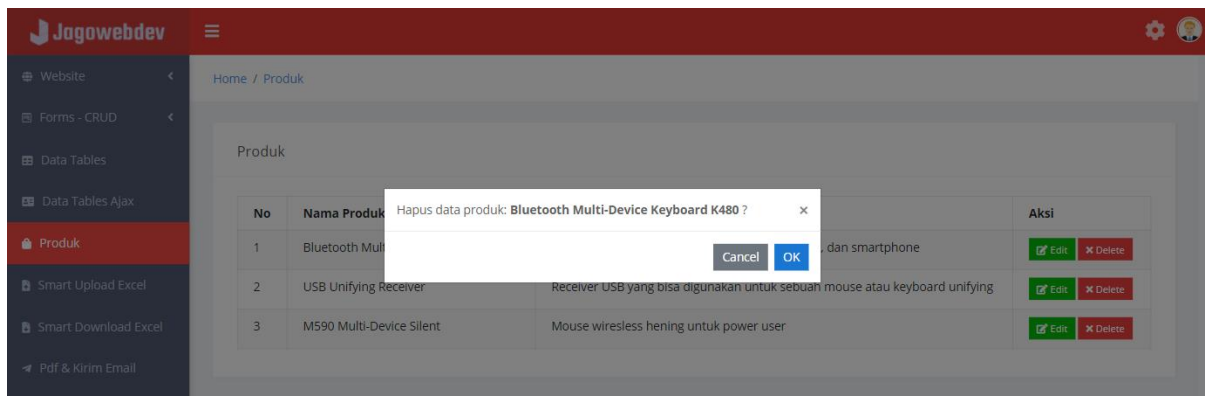
Jika kita jalankan, hasil yang kita peroleh adalah sebagai berikut:



Fungsi `btn_action()` diatas akan menghasilkan HTML sebagai berikut:

```
<div class="form-inline btn-action-group">
  <a href="/edit?id=1" class="btn btn-success btn-xs mr-1">
    <span class="btn-label-icon"><i class="fa fa-edit pr-1"></i></span> Edit
  </a>
  <form method="post" action="">
    <button type="submit" data-action="delete-data" data-delete-title="Hapus
data produk: <strong>Bluetooth Multi-Device Keyboard K480</strong> ?" class="btn btn-
danger btn-xs">
      <span class="btn-label-icon"><i class="fa fa-times pr-
1"></i></span> Delete
    </button>
    <input type="hidden" name="delete" value="delete">
    <input type="hidden" name="id" value="1">
  </form>
</div>
```

Ketika tombol delete di klik maka akan muncul konfirmasi penghapusan data sebagai berikut:



Konfirmasi ini muncul otomatis, karena terdapat script global yang otomatis menangkap event ketika button dengan `data-action="delete-data"` di klik, script ini ada di file `public/themes/modern/builtin/js/site.js`. Adapun scriptnya adalah sebagai berikut:

```
$( 'table' ).on( 'click', '[data-action="delete-data"]', function( e ) {
  e.preventDefault();
  var $this = $( this )
    , $form = $this.parents( 'form:eq(0)' );
  bootbox.confirm( {
    message: $this.attr( 'data-delete-title' ),
```

```
        callback: function(confirmed) {
            if (confirmed) {
                $form.submit();
            }
        },
        centerVertical: true
    });
});
}}
```

F.2. Edit Data

Selanjutnya mari kita bahas cara membuat script untuk edit data.

F.2.1. Menambahkan Halaman Edit

Untuk menambahkan halaman edit, kita buat method edit pada controller misal pada controller produk (app/Controllers/Produk.php). Sebagai contoh, pada file Produk.php kita ubah script menjadi sebagai berikut:

```
<?php
namespace App\Controllers;
use App\Models\ProdukModel;

class Produk extends BaseController
{
    public function __construct() {
        parent::__construct();
        $this->model = new ProdukModel;
    }

    public function index()
    {
        $data = $this->data;
        $data['result'] = $this->model->getProduk();

        if (!$data['result'])
            $this->errorDataNotFound();

        $this->view('produk-result.php', $data);
    }

    public function edit()
    {
        if (empty($_GET['id']))
            $this->errorDataNotFound();

        $produk = $this->model->getProdukById($_GET['id']);

        if (!$produk)
            $this->errorDataNotFound();

        $this->data['title'] = 'Edit Data Produk';
        $this->data['produk'] = $produk;
        $this->view('produk-form.php', $this->data);
    }
}
```

Pada method edit diatas, kita memanggil method `getProdukById()` yang ada pada model. Untuk itu mari kita tambahkan method `getProdukById()` pada file model `app/Models/ProdukModel.php` sebagai berikut:

```
<?php
namespace App\Models;

class ProdukModel extends \App\Models\BaseModel
{
    public function __construct() {
        parent::__construct();
    }

    public function getProduk() {
        $sql = 'SELECT * FROM produk';
        $result = $this->db->query($sql)->getResultArray();
        return $result;
    }

    public function getProdukById($id) {
        $sql = 'SELECT * FROM produk WHERE id_produk = ?';
        $result = $this->db->query($sql, $id)->getRowArray();
        return $result;
    }
}
```

Selanjutnya kita buat file view dengan nama `produk-form.php` yang kita letakkan di folder `app/Views`. Adapun script pada file tersebut adalah sebagai berikut:

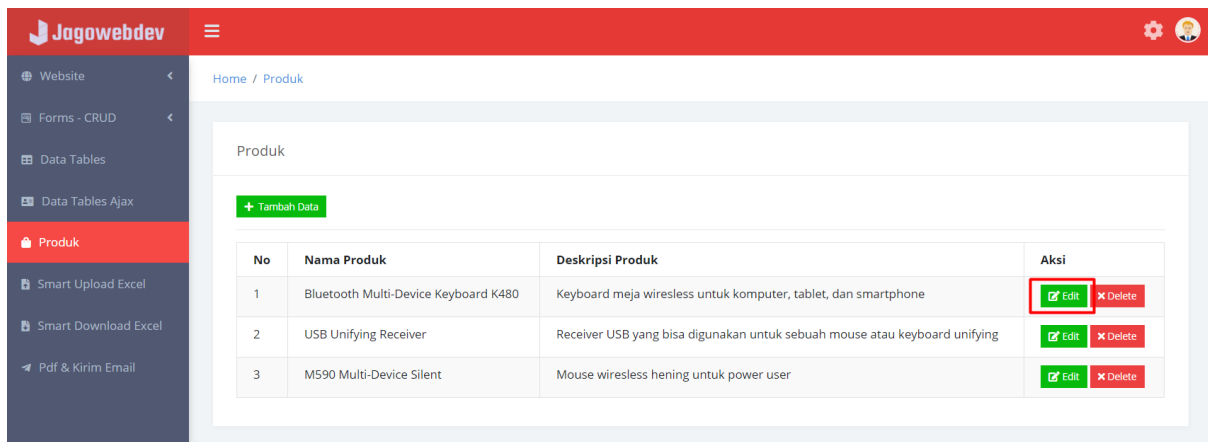
```
<div class="card">
    <div class="card-header">
        <h5 class="card-title"><?=$title?></h5>
    </div>
    <div class="card-body">
        <?php
        if (!empty($message)) {
            show_message($message);
        } ?>
        <form method="post" action="<?=current_url(true)?>" class="form-horizontal">
            <div class="form-group row">
                <label class="col-sm-3 col-md-2 col-lg-3 col-xl-2 col-form-label">Nama Produk</label>
                <div class="col-sm-5">
                    <input class="form-control" type="text"
name="nama_produk" value="<?=set_value('nama_produk', @$produk['nama_produk'])?>"
required="required"/>
                </div>
            </div>
            <div class="form-group row">
                <label class="col-sm-3 col-md-2 col-lg-3 col-xl-2 col-form-label">Deskripsi Produk</label>
                <div class="col-sm-5">
                    <textarea class="form-control"
name="deskripsi_produk"><?=set_value('deskripsi_produk',
@$produk['deskripsi_produk'])?></textarea>
                </div>
            </div>
        </form>
    </div>
</div>
```

```

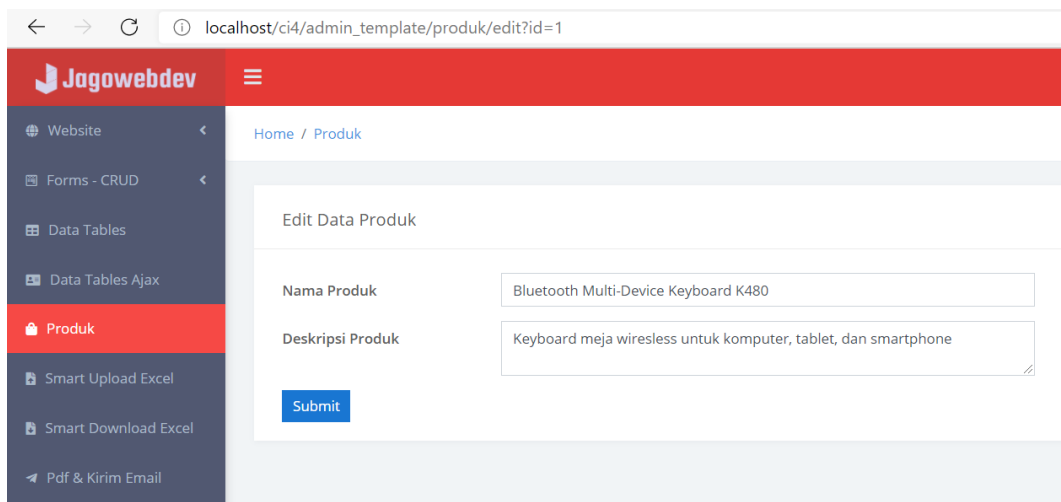
</div>
<div class="form-group row mb-0">
  <div class="col-sm-5">
    <button type="submit" name="submit" value="submit"
class="btn btn-primary">Submit</button>
    <input type="hidden" name="id"
value="<?=@$_GET['id_produk']?>" />
  </div>
</div>
</form>
</div>
</div>

```

Selanjutnya jika kita klik tombol edit



maka kita akan diarahkan ke url http://localhost/ci4/admin_template/produk/edit?id=1 dan tampilan yang kita peroleh adalah sebagai berikut:



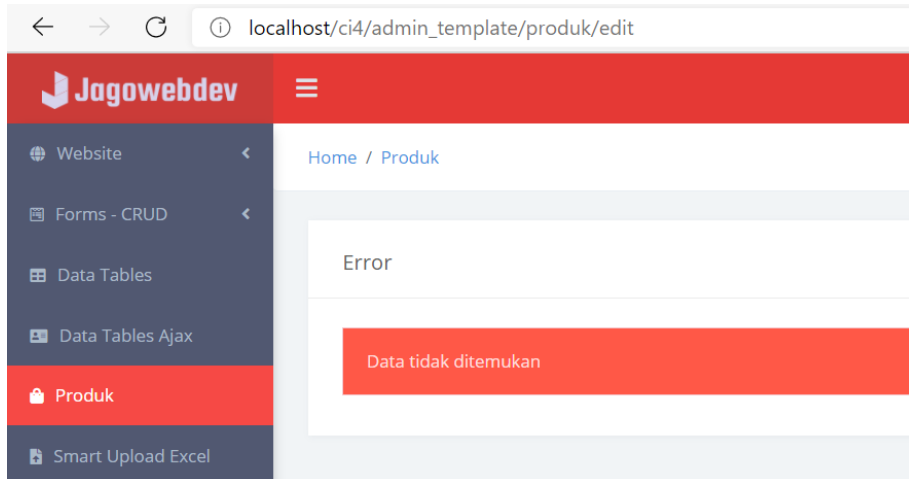
Pada script controller Produk.php, kita lakukan validasi request untuk mengantisipasi error ketika halaman edit dibuka. Pertama kita cek apakah ada id pada variabel \$_GET sebagai berikut:

```
// Script lain
```

```
if (empty($_GET['id']))  
    $this->errorDataNotFound();
```

```
// Script lain
```

Selanjutnya jika kita tes dengan menghilangkan parameter id, maka hasil yang kita peroleh adalah sebagai berikut:



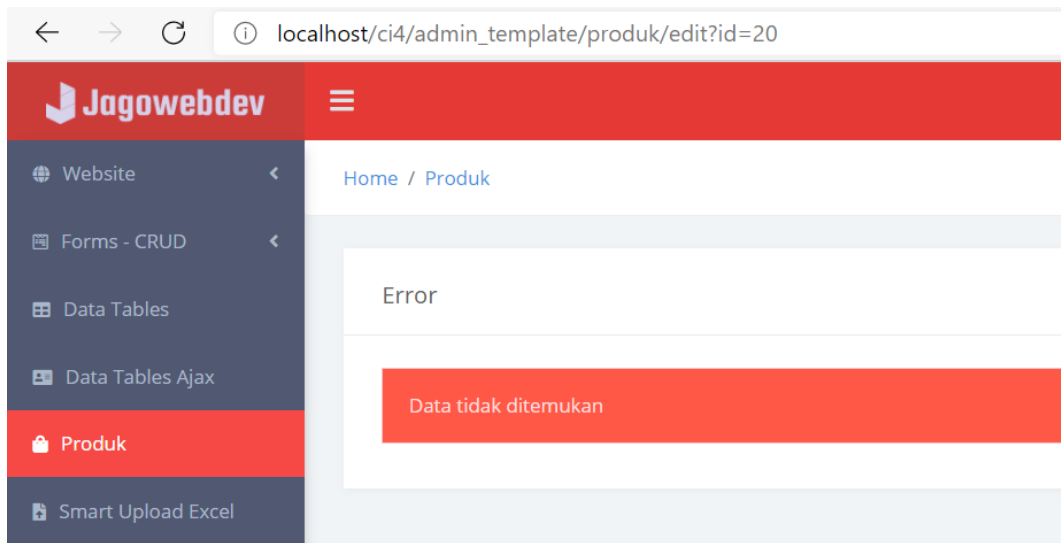
Selanjutnya, kita juga melakukan pengecekan apakah data yang di edit tersedia, script yang digunakan adalah

```
// Script lain
```

```
$produk = $this->model->getProdukById($_GET['id']);  
if (!$produk)  
    $this->errorDataNotFound();
```

```
// Script lain
```

Hal ini untuk mengantisipasi user memasukkan sembarang id, misal di database produk, id hanya ada 3, namun bisa jadi user mengubah id pada url menjadi misal 20



Catatan: method `$this->errorDataNotFound()` ada di file `app/Controllers/BaseController.php`

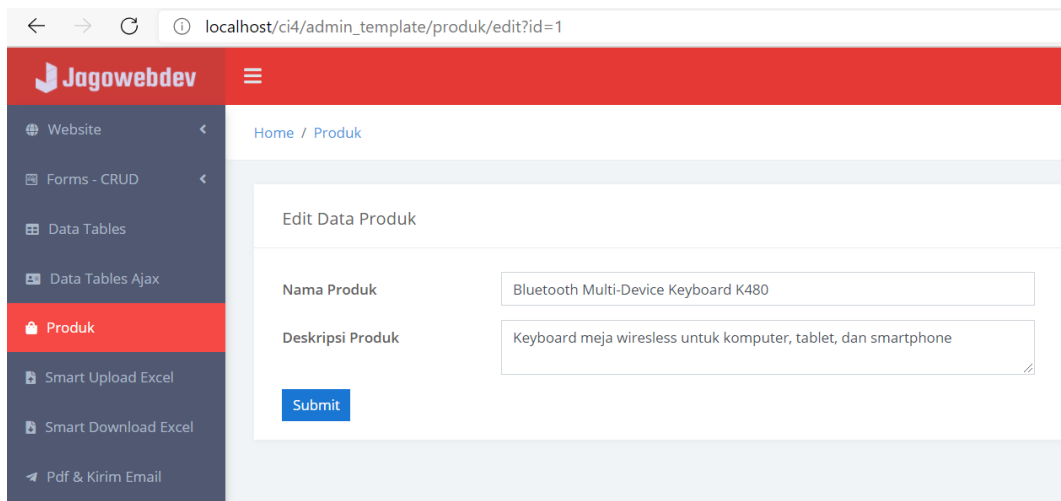
Selanjutnya, script pada file `produk-form.php`, terdapat fungsi `current_url(true)`. Fungsi ini (dengan nilai argumen `true`) akan menghasilkan url beserta query string yang ada, sehingga ketika halaman form tersebut dibuka atribut `action` akan bernilai http://localhost/ci4/admin_template/produk/edit?id=1 sebagai berikut:

```
<form method="post" action="http://localhost/admin_template/produk/edit?id=1"
class="form-horizontal">
```

Pada file `produk-form.php` juga terdapat fungsi `set_value()` sebagai berikut:

```
<input class="form-control" type="text" name="nama_produk"
value="<?=set_value('nama_produk', @$produk['nama_produk'])?>"
required="required"/>
```

Fungsi `set_value()` ini memiliki dua argumen, argumen pertama adalah index dari variabel `$_POST`, sedangkan argumen kedua berisi nilai yang akan digunakan ketika index pada variabel `$_POST` tidak ditemukan, pada contoh diatas, jika data `$_POST['nama_produk']` tidak ditemukan, maka fungsi `set_value` akan mengambil nilai pada variabel `$produk['nama_produk']`, sehingga ketika form pertama kali dibuka akan langsung berisi data produk yang ingin diedit



Selanjutnya, ketika data kita ubah, misal Nama Produk menjadi Bluetooth Multi-Device Keyboard dan kita klik submit, maka Nama Produk akan tetap Bluetooth Multi-Device Keyboard bukan Bluetooth Multi-Device Keyboard K480, hal ini karena nilai `$_POST['nama_produk']` berisi nilai Bluetooth Multi-Device Keyboard sehingga fungsi `set_value('nama_produk', @$produk['nama_produk'])` menghasilkan nilai Bluetooth Multi-Device Keyboard

F.2.2. Submit Data

Selanjutnya kita buat script untuk menyimpan data.

Ketika form edit produk kita submit, maka data akan dikirim melalui method post dan oleh PHP disimpan pada variabel `$_POST`

Untuk menyimpan data form yang disubmit, kita ubah script pada method edit menjadi berikut:

```
public function edit()
{
    if (empty($_GET['id']))
        $this->errorDataNotFound();

    $message = [];
    if (!empty($_POST['submit'])) {
        $error = validate_form();
        if ($error) {
            $message['status'] = 'error';
            $message['message'] = $error;
        } else {
            $update = $this->model->updateData();
            if ($update) {
                $message['status'] = 'ok';
            }
        }
    }
}
```

```

        $message['message'] = 'Data berhasil disimpan';
    } else {
        $message['status'] = 'error';
        $message['message'] = 'Data gagal disimpan';
    }
}

$produk = $this->model->getProdukById($_GET['id']);

if (!$produk)
    $this->errorDataNotFound();

$this->data['title'] = 'Edit Data Produk';
$this->data['produk'] = $produk;
$this->data['id_produk'] = $produk['id_produk'];
$this->view('produk-form.php', $this->data);
}

private function validateForm() {
    $validation = \Config\Services::validation();
    $validation->setRule('nama_produk', 'Nama Produk', 'trim|required');
    $validation->setRule('deskripsi_produk', 'Deskripsi Produk', 'trim|required');
    $validation->withRequest($this->request)->run();
    return $validation->getErrors();
}

```

Pada script diatas, kita tambahkan method `validateForm()` untuk melakukan validasi inputan user. Selanjutnya pada `ProdukModel.php` kita tambahkan method `updateData()` sebagai berikut:

```

public function updateData() {
    $data_db['nama_produk'] = $_POST['nama_produk'];
    $data_db['deskripsi_produk'] = $_POST['deskripsi_produk'];
    $update = $this->db->table('produk')->update($data_db, ['id_produk' => $_POST['id']]);
    return $update;
}

```

Pada method edit diatas, pertama tama kita cek apakah variabel `$_POST['submit']` tidak kosong, jika ya, maka pertama tama kita panggil method `validateForm()` untuk memvalidasi inputan yang ada, Pada method tersebut, jika variabel `$_POST['nama_produk']` dan atau variabel `$_POST['deskripsi_produk']` kosong, maka variabel `$error` akan berisi pesan error.

Kembali ke method edit, jika hasil `$this->validateForm();` bernilai false, yang artinya tidak terdapat error, maka data disimpan (diupdate) menggunakan perintah `$this->model->updateData();`

F.3. Delete Data

Setelah membuat script edit data, kita lanjutkan dengan script delete data.

Seperti telah dibahas sebelumnya, ketika kita klik tombol delete, maka akan muncul popup konfirmasi apakah kita akan menghapus data. Jika tombol OK di klik, maka form delete data akan tersubmit. Ketika form tersebut tersubit, maka data form delete akan dikirim menggunakan method post.

Selanjutnya untuk melakukan delete data, pada file app/Controllers/Produk.php method index kita buat scriptnya menjadi berikut:

```
public function index()
{
    $data = $this->data;

    if ($this->request->getPost('delete'))
    {
        $delete = $this->model->deleteProdukById($_POST['id']);
        if ($delete) {
            $data['message'] = ['status' => 'ok', 'message' => 'Data
produk berhasil dihapus'];
        } else {
            $data['message'] = ['status' => 'warning', 'message' =>
'Tidak ada data yang dihapus'];
        }
    }

    $data['result'] = $this->model->getProduk();

    if (!$data['result'])
        $this->errorDataNotFound();

    $this->view('produk-result.php', $data);
}
```

Berikutnya, pada file model app/Models/ProdukModel.php kita tambahkan method deleteProdukById() sebagai berikut:

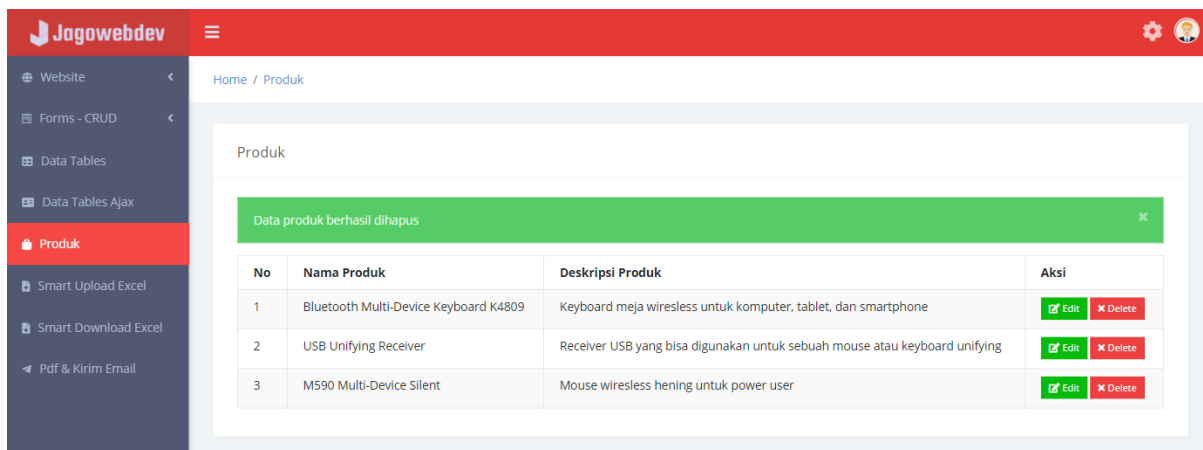
```
public function deleteProdukById($id) {
    $delete = $this->db->table('produk')->delete(['id_produk' => $id]);
    return $delete;
}
```

Pada file controller Produk.php method index, jika \$_POST['delete'] didefinisikan (\$this->request->getPost('delete')) yang artinya ada form delete yang disubmit, maka kita jalankan perintah \$this->model->deleteProdukById(\$_POST['id'])

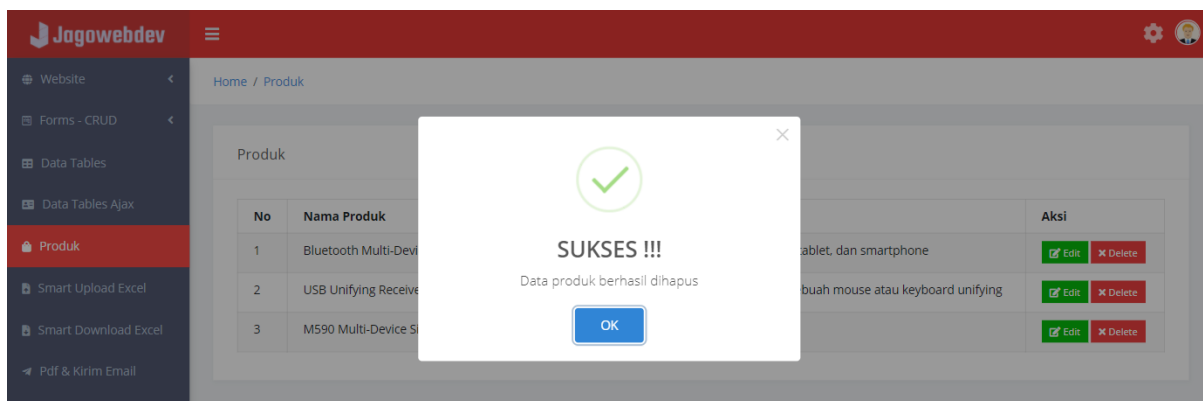
Selanjutnya, untuk menampilkan pesan hasil proses delete data, kita ubah script pada app/Views/produk-result.php bagian <div class="card-body"> menjadi sebagai berikut:

```
<div class="card-body">
<?php
    if (!empty($message)) {
        show_message($message);
    } ?>
<div class="table-responsive">
    <table class="table table-striped table-bordered table-hover">
```

Jika query delete berhasil dijalankan, maka akan muncul pesan Data produk berhasil dihapus



Untuk menampilkan pesan, kita juga dapat menggunakan alert popup. Secara default, ketika aplikasi dijalankan, aplikasi meload library SweetAlert2 (disertakan di script header public/themes/modern/header.php) sehingga kita tinggal menggunakannya saja. Untuk menggunakannya kita tinggal panggil dengan fungsi show_message(), sebagai contoh pada file app/Views/produk-result.php, fungsi show_message() kita ganti dengan show_alert(). Selanjutnya ketika kita menghapus data, maka pesan peringatannya berganti menjadi pop up sebagai berikut:



Catatan: fungsi show_message() dan show_alert() ada di file helper app/Helpers/util_helper.php file helper tersebut disetiing otomatis terload ketika aplikasi dijalankan.

F.4. Tambah Data

Selanjutnya mari kita buat script untuk tambah data.

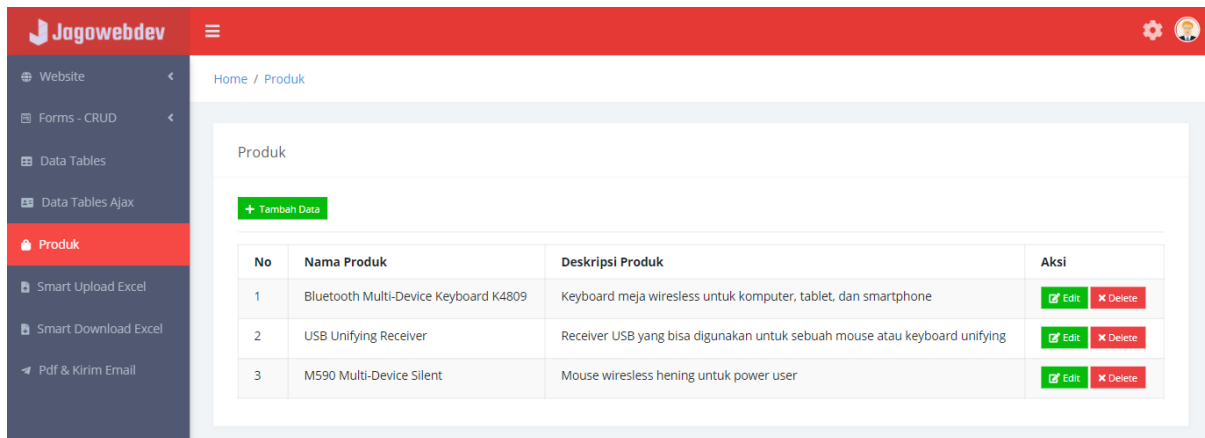
F.4.1. Menambahkan tombol Tambah Data

Pertama tama, mari kita buat script untuk menampilkan tombol data. Caranya, pada script app/Views/produk-result.php kita tambahkan script link tambah data sebagai berikut:

```
<div class="card-body">
    <?php
        if (!empty($message)) {
            show_alert($message);
        } ?>
    <a href="<?=$config->baseURL?>produk/add" class="btn btn-success btn-
xs"><i class="fas fa-plus pr-1"></i> Tambah Data</a>
<hr/>
```

```
<div class="table-responsive">
    <table class="table table-striped table-bordered table-hover">
```

Ketika halaman produk dibuka, akan muncul tombol + Tambah Data sebagai berikut:



Selanjutnya pada script app/Controllers/Produk.php, kita tambahkan method add sebagai berikut:

```
namespace App\Controllers;
use App\Models\ProdukModel;

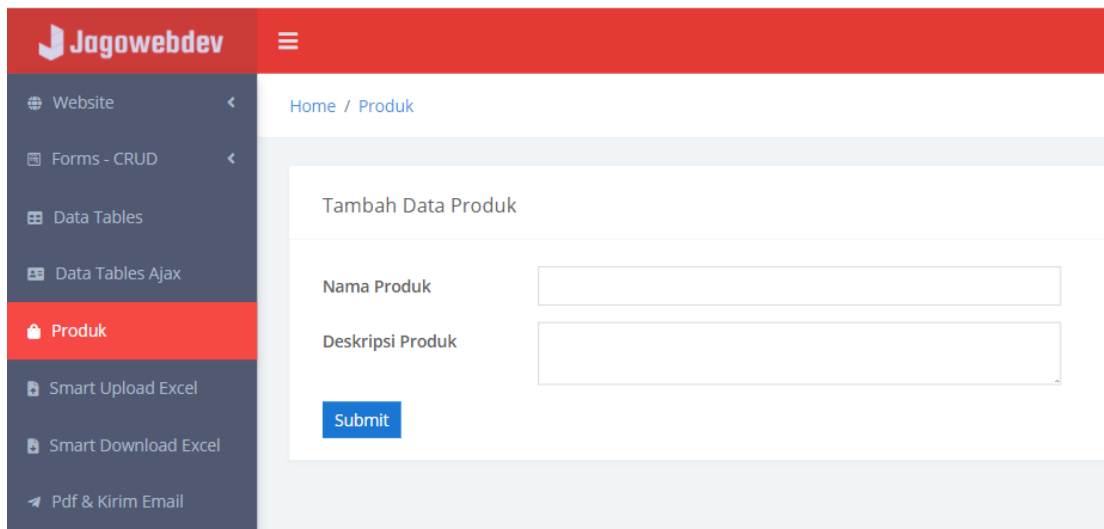
class Produk extends BaseController
{
    // Script lain

    public function edit()
    {
        // Script
    }

    public function add() {
        $this->data['title'] = 'Tambah Data Produk';
        $this->view('produk-form.php', $this->data);
    }

    private function validateForm() {
        // Script
    }
}
```

Perhatikan bahwa kita cukup menggunakan satu form untuk tambah dan edit data, yaitu file app/Views/produk-form.php, tidak perlu membuat form baru untuk tambah data (produk-form.php kita gunakan untuk form tambah dan edit data). Dengan menggunakan satu form, maka maintenance akan menjadi lebih mudah. Selanjutnya, ketika tombol + Tambah Data di klik, maka halaman akan diarahkan ke <http://localhost/ci4/admin template/produk/add> dan tampilan yang dihasilkan adalah sebagai berikut:



F.4.2. Submit Data

Selanjutnya, pada controller produk (app/Controllers/Produk.php) kita tambahkan script submit data sehingga script method add menjadi berikut:

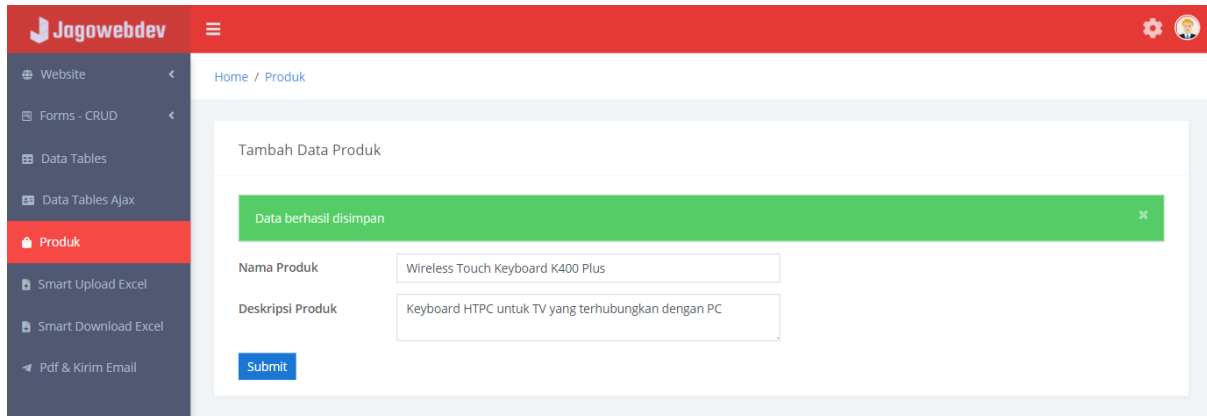
```
public function add() {  
  
    $this->data['title'] = 'Tambah Data Produk';  
  
    $message = [];  
    if (!empty($_POST['submit'])) {  
        $error = $this->validateForm();  
        if ($error) {  
            $message['status'] = 'error';  
            $message['message'] = $error;  
        } else {  
            $query = $this->model->insertData();  
            if ($query) {  
                $message['status'] = 'ok';  
                $message['message'] = 'Data berhasil disimpan';  
            } else {  
                $message['status'] = 'error';  
                $message['message'] = 'Data gagal disimpan';  
            }  
        }  
    }  
  
    $this->data['message'] = $message;  
    $this->view('produk-form.php', $this->data);  
}
```

Berikutnya kita tambahkan method insertData() pada file model app/Models/ProdukModel.php sebagai berikut:

```
public function insertData() {  
    $data_db['nama_produk'] = $_POST['nama_produk'];  
    $data_db['deskripsi_produk'] = $_POST['deskripsi_produk'];  
    $insert = $this->db->table('produk')->insert($data_db);  
}
```

```
}  
    return $insert;  
}
```

Selanjutnya, mari kita tes dengan menambahkan data pada form dan klik disubmit. Jika data berhasil disimpan, maka akan muncul pesan bahwa data berhasil disimpan:



Skenario selanjutnya adalah, ketika data berhasil disubmit, maka posisi form adalah edit data, sehingga ketika form disubmit kembali, maka tidak menambah data, melainkan update data yang baru saja disimpan. Untuk keperluan tersebut, maka kita perlu mendapatkan data id_produk yang baru saja tersimpan kemudian meletakkannya pada form.

Jika kita lihat kembali script app/Views/produk-form.php, maka letak id_produk ada di input dengan name id di sebelah bawah tombol submit, ketika halaman tambah data dibuka, script yang muncul adalah sebagai berikut:

```
<div class="form-group row mb-0">  
    <div class="col-sm-5">  
        <button type="submit" name="submit" value="submit" class="btn btn-  
primary">Submit</button>  
        <input type="hidden" name="id" value=""/>  
    </div>  
</div>
```

Sedangkan script php pada file app/Views/produk-form.php adalah sebagai berikut:

```
<div class="form-group row mb-0">  
    <div class="col-sm-5">  
        <button type="submit" name="submit" value="submit" class="btn btn-  
primary">Submit</button>  
        <input type="hidden" name="id" value="<?=@$_GET['id']?>"/>  
    </div>  
</div>
```

Pada script produk-form.php, agar id_produk tidak hanya mengambil data dari variabel \$_GET saja, kita ubah bagian @\$_GET['id'] menjadi \$id_produk sebagai berikut:

```
<input type="hidden" name="id" value="<?=@$id_produk?>"/>
```


Dengan model seperti itu, nantinya setelah form disubmit dan data berhasil disimpan (ditambah), input name="id" tersebut akan terisi data id_produk dari data yang baru saja disimpan.

Selanjutnya, mari kita ubah script app/Controllers/Produk.php method add sehingga ketika data berhasil disimpan, submit data berikutnya adalah update data bukan insert data

```
$id_produk = $this->model->insertData();
if ($id_produk) {
    $this->data['id_produk'] = $id_produk;
    $message['status'] = 'ok';
    $message['message'] = 'Data berhasil disimpan';
} else {
    $message['status'] = 'error';
    $message['message'] = 'Data gagal disimpan';
}
```

Jangan lupa untuk menambahkan script pada method edit karena form produk-form.php perlu data id_produk. Pada method edit, kita tambahkan script `$this->data['id_produk'] = $produk['id_produk'];` sebagai berikut:

```
public function edit()
{
    // Script lainnya

    $this->data['produk'] = $produk;
    $this->data['id_produk'] = $produk['id_produk'];
    $this->view('produk-form.php', $this->data);
}
```

Selanjutnya pada script model ProdukModel.php method insertData(), kita ubah sehingga ketika data berhasil disimpan, method akan menghasilkan data id dari data yang baru saja disimpan tersebut:

```
public function insertData() {
    $data_db['nama_produk'] = $_POST['nama_produk'];
    $data_db['deskripsi_produk'] = $_POST['deskripsi_produk'];
    $query = $this->db->table('produk')->insert($data_db);
    if ($query)
        return $this->db->insertID();

    return $query;
}
```

Selanjutnya bisa dicoba untuk memasukkan data kemudian Submit, kemudian, masih pada form yang sama, ubah data tersebut dan klik Submit. Cek pada halaman produk, jika berhasil maka data yang bertambah adalah data terakhir ketika kita melakukan perubahan data.

F.5. Merapikan Script

Selanjutnya, jika kita perhatikan, script pada method add dan edit isinya mirip, bedanya, pada method add method model yang dipanggil adalah insertData(), sedangkan pada method edit, method model adalah updateData().

Agar lebih efisien dan tidak terjadi pengulangan penulisan script, pada controller dan model produk kita buat sebuah method yang dapat digunakan untuk menyimpan maupun mengupdate data

Controller app/Controllers/Produk.php

Pada controller Produk.php. kita buat method baru dengan nama saveData() yang merupakan gabungan dari method add() dan edit(), dengan adanya method baru tersebut, kita sesuaikan script pada method add() dan edit(). Hasil akhir script controller Produk.php adalah sebagai berikut:

```
<?php
namespace App\Controllers;
use App\Models\ProdukModel;

class Produk extends BaseController
{
    public function __construct() {
        parent::__construct();
        $this->model = new ProdukModel;
    }

    public function index()
    {
        $data = $this->data;

        if ($this->request->getPost('delete'))
        {
            $delete = $this->model->deleteProdukById($_POST['id']);
            if ($delete) {
                $data['message'] = ['status' => 'ok', 'message' => 'Data produk
berhasil dihapus'];
            } else {
                $data['message'] = ['status' => 'warning', 'message' => 'Tidak
ada data yang dihapus'];
            }
        }

        $data['result'] = $this->model->getProduk();

        if (!$data['result'])
            $this->errorDataNotFound();

        $this->view('produk-result.php', $data);
    }

    public function edit()
    {
        if (empty($_GET['id']))
            $this->errorDataNotFound();

        if (!empty($_POST['submit'])) {
            $result = $this->saveData();
            $this->data['message'] = $result['message'];
        }

        $produk = $this->model->getProdukById($_GET['id']);
    }
}
```

```

        if (!$produk)
            $this->errorDataNotFound();

        $this->data['title'] = 'Edit Data Produk';
        $this->data['produk'] = $produk;
        $this->data['id_produk'] = $produk['id_produk'];
        $this->view('produk-form.php', $this->data);
    }

    public function add() {

        $this->data['title'] = 'Tambah Data Produk';

        if (!empty($_POST['submit'])) {
            $result = $this->saveData();
            $this->data['message'] = $result['message'];
            $this->data['id_produk'] = $result['id_produk'];
        }

        $this->view('produk-form.php', $this->data);
    }

    private function saveData()
    {
        $result = [];
        $id_produk = '';
        if (!empty($_POST['submit'])) {
            $error = $this->validateForm();
            if ($error) {
                $result['status'] = 'error';
                $result['message'] = $error;
            } else {

                $save = $this->model->saveData(@$_POST['id']);

                if ($save['query']['message'] == '') {
                    $result['status'] = 'ok';
                    $result['message'] = 'Data berhasil disimpan';
                } else {
                    $result['status'] = 'error';
                    $result['message'] = 'Data gagal disimpan';
                }

                $id_produk = $save['id_produk'];
            }
        }
        return ['message' => $result, 'id_produk' => $id_produk];
    }

    private function validateForm() {
        $validation = \Config\Services::validation();
        $validation->setRule('nama_produk', 'Nama Produk', 'trim|required');
        $validation->setRule('deskripsi_produk', 'Deskripsi Produk', 'trim|required');
        $validation->withRequest($this->request)->run();
        return $validation->getErrors();
    }
}

```

Model app/Models/ProdukModel.php

Pada model ProdukModel.php, method insertData() dan updateData() kita gabung menjadi satu method baru yang kita beri nama saveData(). Setelah dilakukan perubahan, maka isi script ProdukModel.php menjadi sebagai berikut:

```
<?php
namespace App\Models;

class ProdukModel extends \App\Models\BaseModel
{
    public function __construct() {
        parent::__construct();
    }

    public function getProduk() {
        $sql = 'SELECT * FROM produk';
        $result = $this->db->query($sql)->getResultArray();
        return $result;
    }

    public function getProdukById($id) {
        $sql = 'SELECT * FROM produk WHERE id_produk = ?';
        $result = $this->db->query($sql, $id)->getRowArray();
        return $result;
    }

    public function saveData($id) {
        $data_db['nama_produk'] = $_POST['nama_produk'];
        $data_db['deskripsi_produk'] = $_POST['deskripsi_produk'];
        $id_produk = $id;

        $builder = $this->db->table('produk');
        if (empty($id)) {
            $builder->insert($data_db);
            $id_produk = $this->db->insertID();
        } else {
            $builder->update($data_db, ['id_produk' => $_POST['id']]);
        }

        return ['query' => $this->db->error(), 'id_produk' => $id_produk];
    }

    public function deleteProdukById($id) {
        $delete = $this->db->table('produk')->delete(['id_produk' => $id]);
        return $delete;
    }
}
```

V. Data Tables

Admin template ini telah menyertakan pengelolaan tabel menggunakan library datatables (non ajax), contoh penerapannya ada pada menu Data Tables.

A. Cara Kerja dan Contoh

Pada data tables (yang bukan ajax), pertama tama kita menampilkan semua data tabel terlebih dahulu baru kemudian dengan script data tables tabel tersebut ditampilkan sesuai dengan kriteria yang telah ditentukan seperti data ditampilkan per halaman dengan jumlah tertentu, ditambahkan fitur search data, fitur pengurutan data pada kolom tertentu (sort data), dll

B. Menampilkan Tabel

Tabel HTML kita tampilkan seperti biasa, contoh script untuk menampilkan tabel dapat dilihat di file app\Views\themes\modern\data-tables-result.php sebagai berikut:

```
<table class="table table-striped table-bordered table-hover data-tables" id="data-tables">
  <thead>
    <tr>
      <th>No</th>
      <th>Foto</th>
      <th>Nama</th>
      <th>TTL</th>
      <th>ALAMAT</th>
      <th>NPM</th>
      <th>PRODI</th>
      <th>FAKULTAS</th>
      <th>Aksi</th>
    </tr>
  </thead>
  <tbody>
    <?php
    helper('html');

    $i = 1;
    foreach ($result as $key => $val)
    {
      $image = 'noimage.png';
      if ($val['foto']) {
        if (file_exists('public/images/foto/' . $val['foto'])) {
          $image = $val['foto'];
        }
      }

      echo ' <tr>
        <td>' . $i . '</td>
        <td><div class="list-foto"></div></td>
        <td>' . $val['nama'] . '</td>
        <td>' . $val['tempat_lahir'] . ', ' .
format_tanggal($val['tgl_lahir']) . '</td>
        <td>' . $val['alamat'] . '</td>
        <td>' . $val['npm'] . '</td>
        <td>' . $val['prodi'] . '</td>
        <td>' . $val['fakultas'] . '</td>
        <td>' . btn_action([
          'edit' => ['url' => current_url() . '/edit?id=' .
$val['id_mahasiswa']]
          , 'delete' => ['url' => ''
            , 'id' => $val['id_mahasiswa']
            , 'delete-title' => 'Hapus data mahasiswa:
<strong>' . $val['nama'] . '</strong> ?'
          ] .
        '</td>
```

```

        </tr>';
        $i++;
    }

    $settings['order'] = [2, 'asc'];
    $settings['columnDefs'][] = ['targets' => [0,1,8], 'orderable' => false];
    ?>
</tbody>
</table>

```

Jika halaman ditampilkan maka HTML yang dihasilkan adalah:

```

<table class="table table-striped table-bordered table-hover data-tables" id="data-tables">
  <thead>
    <tr>
      <th>No</th>
      <th>Foto</th>
      <th>Nama</th>
      <th>TTL</th>
      <th>ALAMAT</th>
      <th>NPM</th>
      <th>PRODI</th>
      <th>FAKULTAS</th>
      <th>Aksi</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>1</td>
      <td><div class="list-foto"></div></td>
      <td>Sony Haryanto</td>
      <td>Surabaya, 01 Februari 1997</td>
      <td>Jl. Kenanga No, 137, Surabaya</td>
      <td>22222</td>
      <td>Sistem Perangkat Lunak</td>
      <td>Teknik Informatika</td>
      <td>
        <div class="form-inline btn-action-group">
          <a href="http://localhost/ci4/admin_template/data-tables/edit?id=2"
class="btn btn-success btn-xs me-1 btn btn-success btn-xs me-1">
            <span class="btn-label-icon"><i class="fa fa-edit pe-
1"></i></span> Edit
          </a>
          <form method="post" action="">
            <button type="submit" data-action="delete-data" data-
delete-title="Hapus data mahasiswa: <strong>Sony Haryanto</strong> ?" class="btn btn-danger btn-
xs">
              <span class="btn-label-icon"><i class="fa fa-times
pe-1"></i></span> Delete
            </button>
            <input type="hidden" name="delete" value="delete"/>
            <input type="hidden" name="id" value="2"/>
          </form>
        </div>
      </td>
    </tr>
    <tr>
      <td>2</td>
      <td><div class="list-foto"></div></td>
      <td>Budi Kurniawan</td>
      <td>Jakarta, 05 Juli 2000</td>

```

```
        <td>Jl. Angrek No. 09, Jakarta</td>
        ...
    </tr>
</tbody>
</table>
```

C. File Javascript dan CSS Library dataTables

Selanjutnya Untuk dapat menggunakan datatables, kita perlu menyertakan file javascript dan CSS datatables. Untuk file javascript, yang disertakan adalah sebagai berikut:

```
<script type="text/javascript"
src="http://localhost/ci4/admin_template/public/vendors/datatables/dist/js/jquery.dataTables.min.js"></script>
<script type="text/javascript"
src="http://localhost/ci4/admin_template/public/vendors/datatables/dist/js/dataTables.bootstrap5.min.js"></script>
```

Pada Admin Template kali ini, datatables yang kita gunakan menggunakan theme Bootstrap 5 sehingga kita perlu menyertakan file/dataTables.bootstrap5.min.js seperti pada contoh diatas.

Selanjutnya file CSS yang kita sertakan adalah

```
<link rel="stylesheet" type="text/css"
href="http://localhost/ci4/admin_template/public/vendors/datatables/dist/css/dataTables.bootstrap5.min.css"/>
```

Semua script diatas ada di file app\Views\themes\modern\header.php

Selanjutnya kita perlu membuat script untuk membuat tabel tersebut diatas memiliki tampilan datatable. Script Javascript untuk menjalankan dataTables ada pada file javascript public/themes/modern/js/data-tables.js. Pada script tersebut kita menginisiasi Data Tables dengan perintah berikut:

```
const table = $('#data-tables').DataTable(settings);
```

Parameter settings ada di file yang sama sebagai berikut:

```
const $setting = $('#dataTables-setting');
let settings = {};
if ($setting.length > 0) {
    settings = $.parseJSON($('#dataTables-setting').html());
}
```

Parameter diatas merupakan parameter default yang akan diterapkan pada data tables yang akan kita buat. Pada script diatas terdapat variabel settings (**let settings**). Isi dari variabel ini app\Views\themes\modern\data-tables-ajax-result.php (**settings = \$.parseJSON(\$('#dataTables-setting').html());**) hal ini dilakukan untuk memudahkan pengaturan setting disatu tempat yaitu di file data-tables-ajax-result.php, anda dapat menyesuaikannya sesuai dengan kebutuhan.

D. Setting

Pada file `app\Views\themes\modern\data-tables-ajax-result.php` script untuk mendefinisikan setting data tables adalah sebagai berikut:

```
<?php
$settings['order'] = [2, 'asc'];
$settings['columnDefs'][] = ['targets' => [0,1, 8], 'orderable' => false];
?>
<span id="dataTables-setting" style="display:none"><?=json_encode($settings)?></span>
```

Ketika halaman ditampilkan, maka HTML yang dihasilkan adalah sebagai berikut:

```
<span id="dataTables-setting" style="display:none">
{"order":[2,"asc"],"columnDefs":[{"targets":[0,1,8],"orderable":false}]}
</span>
```


Dengan demikian pendefinisian datatables (pada file `public/themes/modern/js/data-tables.js`):

```
const table = $('#data-tables').DataTable(settings);
```


Sama dengan:

```
const table = $('#data-tables').DataTable(
    {"order":[2,"asc"],"columnDefs":[{"targets":[0,1,8],"orderable":false}]}
);
```

Setting diatas bertujuan untuk mengurutkan data berdasarkan kolom ketiga (index nomor 2) secara ascending

| No | Foto | Nama ↑ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ | Prodi ↑↓ | Fakultas ↑↓ | Action |
|----|---|--------------|--------------------------|------------------------------|--------|-------------------------|--------------------|---|
| 1 |  | Ahmad Basuki | Surakarta, 10 Maret 2000 | Jl. Bendar No. 77, Surakarta | 55555 | Sistem Informasi Modern | Teknik Informatika | Edit Delete |

dan kolom dengan index 0 (kolom pertama), kolom dengan index 1 (kolom kedua) dan kolom dengan index 8 (kolom ke 7) tidak memiliki fitur sort

| No | Foto | Nama ↑↓ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ | Prodi ↑↓ | Fakultas ↑↓ | Action |
|----|---|--------------|--------------------------|------------------------------|--------|-------------------------|--------------------|---|
| 1 |  | Ahmad Basuki | Surakarta, 10 Maret 2000 | Jl. Bendar No. 77, Surakarta | 55555 | Sistem Informasi Modern | Teknik Informatika | Edit Delete |

E. Button Copy, Excel, Pdf, Print

Pada data tables (non ajax) kita dapat menambahkan button untuk berbagai keperluan, yaitu button copy untuk mengcopy semua data, button excel untuk ekspor semua data ke format excel, button pdf untuk mengekspor semua data ke format pdf, dan button print untuk mencetak data.

Untuk memunculkan button ini, kita perlu menambahkan setting pada data tables, script tambahan setting adalah sebagai berikut (file `public/themes/modern/js/data-tables.js`):

```
const addSettings =
{
  "buttons":[
    { "extend": "copy"
      , "text": "<i class='far fa-copy'></i> Copy"
      , "className": "btn-light me-1"
    },
    { "extend": "excel"
      , "title": "Data Mahasiswa"
      , "text": "<i class='far fa-file-excel'></i> Excel"
      , "exportOptions": {
        columns: [2, 3, 4, 5, 6, 7],
        modifier: {selected: null}
      }
      , "className": "btn-light me-1"
    },
    // script lainnya
  ]
}
```

Selanjutnya script tambahan setting ini kita gabungkan dengan setting sebelumnya

```
// Merge settings
settings = {...settings, ...addSettings};
```

Jika tidak ingin menampilkan button, baris tersebut diatas tinggal di jadikan comment sebagai berikut:

```
// Merge settings
// settings = {...settings, ...addSettings};
```

VI. Data Tables Ajax

Admin template ini telah menyertakan pengelolaan tabel menggunakan library datatables ajax, contoh penerapannya ada pada menu Data Tables Ajax.

A. File Javascript dan CSS Library dataTables

Sama seperti sebelumnya, untuk dapat menggunakan datatables ajax, kita perlu menyertakan file javascript dan CSS datatables. Untuk file javascript, yang disertakan adalah sebagai berikut:

```
<script type="text/javascript"
src="http://localhost/ci4/admin_template/public/vendors/datatables/dist/js/jquery.dataTables.min.js"></script>
<script type="text/javascript"
src="http://localhost/ci4/admin_template/public/vendors/datatables/dist/js/dataTables.bootstrap5.min.js"></script>
```

Sama seperti contoh sebelumnya, datatables yang kita gunakan kali ini menggunakan theme Bootstrap 5 sehingga kita perlu menyertakan file/dataTables.bootstrap5.min.js seperti pada contoh diatas, Script diatas disertakan di file app\Views\themes\modern\header.php

Selanjutnya file CSS yang kita sertakan adalah

```
<link rel="stylesheet" type="text/css"
href="http://localhost/ci4/admin_template/public/vendors/datatables/dist/css/dataTables.bootstrap5.min.css"/>
```

File css diatas juga disertakan di file header.php

F. Script Menjalankan DataTables

Script Javascript untuk menjalankan dataTable ajax ada pada file javascript public/themes/modern/js/data-tables-ajax.js. Pada script tersebut kita menginisiasi Data Tables Ajax dengan perintah berikut:

```
table = $('#table-result').DataTable( settings );
```

Parameter settings ada di file yang sama sebagai berikut:

```
const column = $.parseJSON($('#dataTables-column').html());
const url = $('#dataTables-url').html();

const settings = {
  "processing": true,
  "serverSide": true,
  "scrollX": true,
  "ajax": {
    "url": url,
    "type": "POST"
  },
  "columns": column,
  "initComplete": function( settings, json ) {
    //
  }
}
```

Parameter diatas merupakan parameter default yang akan diterapkan pada data tables yang akan kita buat. Pada script diatas terdapat variabel column dan url. Isi dari variabel ini ada di file view app\Views\themes\modern\data-tables-ajax-result.php, penempatan variabel di halaman view ini dilakukan untuk memudahkan pengaturan setting disatu tempat yaitu di file data-tables-ajax-result.php, anda dapat menyesuaikannya sesuai dengan kebutuhan.

B.1. URL

Pada Datatables Ajax, pertama tama kita harus mendefinisikan alamat url dimana data akan diambil. Pada script diatas, alamat URL diambil dari element HTML dengan id dataTables-url url = \$('#dataTables-url').html(); element ini ada di file app\Views\themes\modern\data-tables-ajax-result.php sebagai berikut:

```
<span id="dataTables-url" style="display:none"><?=current_url() . '/getDataDT' ?></span>
```

Jika halaman ditampilkan maka element HTML yang digenerate adalah sebagai berikut:

```
<span id="dataTables-url" style="display:none">http://localhost/ci4/admin_template/data-tables-ajax/getDataDT</span>
```

Adapun bentuk data yang dihasilkan dari url diatas adalah sebagai berikut:

```
{
  "draw": "1",
  "recordsTotal": "16",
  "recordsFiltered": 16,
  "data": [
    {
      "id_mahasiswa": "8",
      "nama": "Ahmad Basuki",
      "email": "ahmad.basuki@yopmail.com",
      "npm": "55555",
      "tempat_lahir": "Surakarta",
      "tgl_lahir": "Surakarta, 10 Maret 2000",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatika",
      "alamat": "Jl. Bendar No. 77, Surakarta",
      "foto": "Ahmad Basuki.png",
      "qrcode_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "1",
      "tgl_edit": null,
      "id_user_edit": null,
      "ignore_search_foto": "<div class=\"list-foto\"><img
src=\"http://localhost/ci4/admin_template/public/images/foto/Ahmad Basuki.png\"/></div>",
      "ignore_search_no_urut": 1,
      "ignore_search_action": // Button
    },
    {
      "id_mahasiswa": "10",
      "nama": "Ahmad Fathoni",
      "email": "ahmad.fathoni@yopmail.com",
      "npm": "234545",
      // Data lainnya
    }
  ]
}
```

B.2. Kolom

Selanjutnya kita perlu mendefinisikan data mana saja yang akan kita isikan ke kolom tabel. Pada datatables format data kolom ini berbentuk array sebagai berikut:

```
settings = {
  "processing": true,
  "serverSide": true,
  "scrollX": true,
  "ajax": {
    "url": url,
    "type": "POST"
  },
  "columns": [
    { "data": "ignore_search_no_urut" },
    { "data": "ignore_search_foto" },
    { "data": "nama" },
    { "data": "alamat" },
    { "data": "tgl_lahir" },
    { "data": "alamat" },
  ]
}
```

```

        { "data": "npm" }
        { "data": "prodi" }
        { "data": "fakultas" }
        { "data": "ignore_search_action" }
    ],
    "initComplete": function( settings, json ) {
        //
    }
}

```

Pada contoh diatas, baris pada tabel akan diisi oleh data: `ignore_search_no_urut` pada kolom pertama, `ignore_search_foto` pada kolom kedua, nama pada kolom ketiga, dst. Data ini diambil dari data yang sudah diambil dari url yang telah kita definisikan sebelumnya:

```

{
    "draw": "1",
    "recordsTotal": "16",
    "recordsFiltered": 16,
    "data": [
        {
            "id_mahasiswa": "8",
            "nama": "Ahmad Basuki",
            "email": "ahmad.basuki@yopmail.com",
            "npm": "55555",
            "tempat_lahir": "Surakarta",
            "tgl_lahir": "Surakarta, 10 Maret 2000",
            "prodi": "Sistem Informasi Modern",
            "fakultas": "Teknik Informatika",
            "alamat": "Jl. Bendar No. 77, Surakarta",
            "foto": "Ahmad Basuki.png",
            "qrcode_text": null,
            "tgl_input": "2020-11-15",
            "id_user_input": "1",
            "tgl_edit": null,
            "id_user_edit": null,
            "ignore_search_foto": "<div class='list-foto'><img
src='\"http://localhost/ci4/admin_template/public/images/foto/Ahmad Basuki.png\"'/></div>",
            "ignore_search_no_urut": 1,
            "ignore_search_action": "// Button
        },
        {
            "id_mahasiswa": "10",
            "nama": "Ahmad Fathoni",
            "email": "ahmad.fathoni@yopmail.com",
            "npm": "234545",
            // Data lainnya
        }
        // Data lainnya
    ]
}

```

Agar mudah melakukan konfigurasi, data kolom tabel ini kita letakkan pada element HTML dengan id `dataTables-column` pada file `app\Views\themes\modern\data-tables-ajax-result.php`, selanjutnya pada file `data-tables-ajax.js` data tersebut kita ambil dengan perintah `column = $.parseJSON($('#dataTables-column').html());`

Script untuk mendefinisikan kolom (file `data-table-ajax-result.php`) adalah sebagai berikut:

```

$column =[
    'ignore_search_no_urut' => 'No'
    , 'ignore_search_foto' => 'Foto'
    , 'nama' => 'Nama'
    , 'tgl_lahir' => 'TTL'
    , 'alamat' => 'Alamat'
    , 'npm' => 'NPM'
    , 'prodi' => 'Prodi'
    , 'fakultas' => 'Fakultas'
    , 'ignore_search_action' => 'Action'
];

foreach ($column as $key => $val) {
    $column_dt[] = ['data' => $key];
}

<span id="dataTables-column" style="display:none"><?=json_encode($column_dt)?></span>

```

Ketika halaman digenerate maka variabel `$column_dt` akan menjadi:

```

<span id="dataTables-column"
style="display:none">[{"data":"ignore_search_urut"}, {"data":"foto"}, {"data":"nama"
}, {"data":"tgl_lahir"}, {"data":"alamat"}, {"data":"npm"}, {"data":"prodi"}, {"data":"
fakultas"}, {"data":"ignore_search_action"}]</span>

```

Penting diperhatikan:

Ketika terjadi perubahan data, seperti filter data, searching data, dan sort data, datatables akan mengirim data nama semua kolom ini, selanjutnya dari sisi server kita membuat script untuk melakukan pengolahan data berdasarkan kolom tersebut. Kita melakukan pengolahan data dengan melakukan query data pada database sehingga kita perlu mendefinisikan kolom yang tidak ada di database atau kolom yang tidak relevan untuk pengolahan data, pada contoh kali ini kita beri nama kolom tersebut dengan awalan **ignore_search_** sehingga nantinya ketika melakukan pengolahan data kolom ini akan diabaikan

B.3. Setting Tambahan

Selanjutnya kita dapat menambahkan parameter tertentu sesuai dengan keperluan seperti kolom mana yang akan diurutkan (sort) dan kolom mana yang tidak diberikan fitur sort (misal kolom yang berisi foto atau kolom yang berisi tombol edit dan delete). Penerapannya pada datatables adalah sebagai berikut:

```

settings = {
    "processing": true,
    "serverSide": true,
    "scrollX": true,
    "ajax": {
        "url": url,
        "type": "POST"
    },
    "columns": [
        { "data": "no_urut" },
        { "data": "ignore_search_foto" },
        { "data": "nama" },
        { "data": "alamat" },

```

```

        { "data": "tgl_lahir" },
        { "data": "alamat" },
        { "data": "npm" },
        { "data": "prodi" },
        { "data": "fakultas" },
        { "data": "ignore_search_action" }
    ],
    "order": [3, "desc"]
    "columnDefs": [
        { "targets": 1, "orderable": false },
        { "targets": 8, "orderable": false }
    ]
    "initComplete": function( settings, json ) {
        //
    }
}

```

Pada script diatas, tabel akan diurutkan berdasarkan kolom ketiga dengan model pengurutan descending **"order": [3, "desc"]** selanjutnya kolom yang tidak kita beri fitur sort nya adalah kolom ke 1 yaitu kolom foto dan kolom ke 8 yaitu kolom yang berisi tombol edit dan delete (urutan kolom dimulai dari 0)

```

"columnDefs": [
    { "targets": 1, "orderable": false },
    { "targets": 8, "orderable": false }
]

```

Agar memudahkan pengaturan, parameter tambahan ini juga kita letakkan di file `app\Views\themes\modern\data-tables-ajax-result.php` sebagai berikut

```

$column =[
    'ignore_search_no_urut' => 'No'
    , 'ignore_search_foto' => 'Foto'
    , 'nama' => 'Nama'
    , 'tgl_lahir' => 'TTL'
    , 'alamat' => 'Alamat'
    , 'npm' => 'NPM'
    , 'prodi' => 'Prodi'
    , 'fakultas' => 'Fakultas'
    , 'ignore_search_action' => 'Action'
];

$settings['order'] = [3,'desc'];
$index = 0;
$th = '';
foreach ($column as $key => $val) {
    $th .= '<th>' . $val . '</th>';
    if (strpos($key, 'ignore_search') !== false) {
        $settings['columnDefs'][] = ["targets" => $index, "orderable" => false];
    }
    $index++;
}

<span id="dataTables-setting" style="display:none"><?=json_encode($settings)?></span>

```


ketika halaman ditampilkan maka tag `` dengan id `dataTables-setting` akan berisi data JSON sebagai berikut:

```
<span id="dataTables-setting"
style="display:none">{"order":[3,"desc"],"columnDefs":[{"targets":1,"orderable":false}, {"targets":8,"orderable":false}]}</span>
```


Selanjutnya pada file data-tables-ajax.js setting tersebut kita ambil dan kita gabungkan dengan settings defaultnya

```
let $add_setting = $('#dataTables-setting');
if ($add_setting.length > 0) {
    add_setting = $.parseJSON($('#dataTables-setting').html());
    for (k in add_setting) {
        settings[k] = add_setting[k];
    }
}
```

Dengan tambahan setting pada contoh diatas maka: tabel akan diurutkan berdasarkan kolom ke 3 (kolom dengan index 2) secara ascending

| No | Foto | Nama ↑ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ | Prodi ↑↓ | Fakultas ↑↓ | Action |
|----|--|--------------|--------------------------|------------------------------|--------|-------------------------|--------------------|---|
| 1 |  | Ahmad Basuki | Surakarta, 10 Maret 2000 | Jl. Bendar No. 77, Surakarta | 55555 | Sistem Informasi Modern | Teknik Informatika | Edit Delete |

dan kolom dengan index 0 (kolom pertama), kolom dengan index 1 (kolom kedua) dan kolom dengan index 8 (kolom ke 7) tidak memiliki fitur sort

| No | Foto | Nama ↑↓ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ | Prodi ↑↓ | Fakultas ↑↓ | Action |
|----|---|--------------|--------------------------|------------------------------|--------|-------------------------|--------------------|---|
| 1 |  | Ahmad Basuki | Surakarta, 10 Maret 2000 | Jl. Bendar No. 77, Surakarta | 55555 | Sistem Informasi Modern | Teknik Informatika | Edit Delete |

B.4. Header dan Footer

Selanjutnya kita mendefinisikan header dan footer dari tabel yang ingin kita tampilkan, script yang kita gunakan adalah sebagai berikut:

```
$column = [
    'ignore_search_urut' => 'No'
    , 'foto' => 'Foto'
    , 'nama' => 'Nama'
    , 'tgl_lahir' => 'TTL'
    , 'alamat' => 'Alamat'
    , 'npm' => 'NPM'
    , 'prodi' => 'Prodi'
    , 'fakultas' => 'Fakultas'
    , 'ignore_search_action' => 'Action'
];

foreach ($column as $key => $val) {
    $th .= '<th>' . $val . '</th>';
    // Script lainnya
}
```

```

}

<table id="table-result" class="table display table-striped table-bordered table-hover"
style="width:100%">
  <thead>
    <tr>
      <th><?=$th?>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <th><?=$th?>
    </tr>
  </tfoot>
</table>

```

Ketika halaman ditampilkan, tag HTML yang dihasilkan adalah sebagai berikut:

```

<table id="table-result" class="table display table-striped table-bordered table-hover"
style="width:100%">
  <thead>
    <tr>
      <th>No.</th>
      <th>Foto</th>
      <th>Nama</th>
      <th>TTL</th>
      <th>Alamat</th>
      <th>NPM</th>
      <th>Prodi</th>
      <th>Fakultas</th>
      <th>Action</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <th>No.</th>
      <th>Foto</th>
      <th>Nama</th>
      <th>TTL</th>
      <th>Alamat</th>
      <th>NPM</th>
      <th>Prodi</th>
      <th>Fakultas</th>
      <th>Action</th>
    </tr>
  </tfoot>
</table>

```

B.5. Modifikasi Data Baris Tabel

Data yang akan kita tampilkan pada datatables diambil sesuai dengan url yang telah kita definisikan pada angka 1, yaitu http://localhost/ci4/admin_template/data-tables-ajax/getDataDT

Url diatas akan menghasilkan data JSON dengan format sebagai berikut:

```

{
  "draw": "1",
  "recordsTotal": "18",
  "recordsFiltered": "18",

```

```

"data": [
    {
        "id_mahasiswa": "17",
        "nama": "Wahyu Jatmiko",
        "email": "wahyujatmiko@email.com",
        "npm": "345432",
        "tempat_lahir": "Surakarta",
        "tgl_lahir": "Surakarta, 05 April 2002",
        "prodi": "Desain Grafis",
        "fakultas": "Desain Komunikasi Visual",
        "alamat": "Jl. Arjuna No. 14, Surakarta",
        "foto": "<div class='list-foto'><img
src='http://localhost:7777/ci4/admin_template/public/images/foto/noimage.png'></div>",
        "qrcode_text": null,
        "tgl_input": null,
        "id_user_input": null,
        "tgl_edit": "2021-08-17",
        "id_user_edit": "1",
        "ignore_search_urut": 1,
        "ignore_search_action": "<div class='form-inline btn-action-
group'><a href='http://localhost:7777/ci4/admin_template/data-tables-ajax/edit?id=17'
class='btn btn-success btn-xs me-1 btn btn-success btn-xs me-1'><span class='btn-
label-icon'><i class='fa fa-edit pe-1'></i></span> Edit</a><form method='post'
action='\"><button type='submit'\" data-action='delete-data' data-delete-title='Hapus
data mahasiswa: <strong>Wahyu Jatmiko</strong> ?'\" class='btn btn-danger btn-xs'><span
class='btn-label-icon'><i class='fa fa-times pe-1'></i></span> Delete</button><input
type='hidden'\" name='delete'\" value='delete'\"><input type='hidden'\" name='id'\"
value='17'\"></form></div>"
    },
    {
        "id_mahasiswa": "8",
        "nama": "Tendi Anshori",
        "email": "tendi.anshori@yopmail.com",
        "npm": "12345",
        "tempat_lahir": "Banyuwangi",
        "tgl_lahir": "Banyuwangi, 06 Agustus 2001",
        "prodi": "Desain Kreatif",
        "fakultas": "Desain Komunikasi Visual",
        "alamat": "Jl. Semarak No. 03, Banyuwangi",
        "foto": "<div class='list-foto'><img
src='http://localhost:7777/admin_template/public/images/foto/Tendi Anshori.png'></div>",
        "qrcode_text": null,
        "tgl_input": "2020-11-15",
        "id_user_input": "2",
        "tgl_edit": null,
        "id_user_edit": null,
        "ignore_search_urut": 3,
        "ignore_search_action": "<div class='btn-action-group'><a
href='http://localhost:7777/admin_template/datatables-ajax/edit?id=8'\" class='btn btn-
success btn-xs me-1'><span class='btn-label-icon'><i class='fa fa-edit pe-
1'></i></span> Edit</a><form method='post'\" action='\"><button type='submit'\" data-
action='delete-data'\" data-delete-title='Hapus data mahasiswa: <strong>Tendi
Anshori</strong> ?'\" class='btn btn-danger btn-xs'><span class='btn-label-icon'><i
class='fa fa-times pe-1'></i></span> Delete</button><input type='hidden'\"
name='delete'\" value='delete'\"><input type='hidden'\" name='id'\"
value='8'\"></form></div>"
    },
    {
        "id_mahasiswa": "7",
        "nama": "Intan Nurwiyati",
        "email": "intan.nurwiyati@yopmail.com",
        "npm": "12345",
        // Data lainnya
    },

```

```

    // Data lainnya
  }
}

```

Output data ini dapat dilihat dengan membuka developer tools pada browser sebagai berikut:

The screenshot shows the Jagowebdev website with a table of student data. The table has columns: No, Foto, Nama, TTL, Alamat, NPM, Prodi, and Fakultas. The data is as follows:

| No | Foto | Nama | TTL | Alamat | NPM | Prodi | Fakultas |
|----|------|---------------|--------------------------|---------------------------------|--------|-------------------------|-------------------|
| 1 | | Ahmad Basuki | Surakarta, 10 Maret 2000 | Jl. Bendar No. 77, Surakarta | 55555 | Sistem Informasi Modern | Teknik Informatik |
| 2 | | Ahmad Fathoni | Jember, 04 Juli 2000 | Jl. Ahmad Dahlan No. 03, Jember | 234545 | Sistem Informasi Modern | Teknik Informatik |
| 3 | | Anton | Serang, 22 | Jl. Sidorang | 12345 | Desain | Desain |

The browser's developer tools network tab is open, showing the response of the `getDataDT` endpoint. The response is a JSON object containing a list of student data.

```

{
  "draw": "1",
  "recordsTotal": "16",
  "recordsFiltered": "16",
  "data": [
    {
      "id_mahasiswa": "8",
      "nama": "Ahmad Basuki",
      "email": "ahmad.basuki@yopmail.com",
      "npm": "55555",
      "tempat_lahir": "Surabaya",
      "tgl_lahir": "10-03-2000",
      "alamat": "Jl. Bendar No. 77, Surakarta",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatik",
      "foto": "Ahmad Basuki.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "10",
      "nama": "Ahmad Fathoni",
      "email": "ahmad.fathoni@yopmail.com",
      "npm": "234545",
      "tempat_lahir": "Jember",
      "tgl_lahir": "04-07-2000",
      "alamat": "Jl. Ahmad Dahlan No. 03, Jember",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatik",
      "foto": "Ahmad Fathoni.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "7",
      "nama": "Anton Junaedi",
      "email": "anton.junaedi@yopmail.com",
      "npm": "12345",
      "tempat_lahir": "Serang",
      "tgl_lahir": "22-00-00",
      "alamat": "Jl. Sidorang",
      "prodi": "Desain",
      "fakultas": "Desain",
      "foto": "Anton.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "11",
      "nama": "Boni Simanjuntak",
      "email": "boni.simanjuntak@yopmail.com",
      "npm": "55555",
      "tempat_lahir": "Surabaya",
      "tgl_lahir": "10-03-2000",
      "alamat": "Jl. Bendar No. 77, Surakarta",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatik",
      "foto": "Boni Simanjuntak.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "3",
      "nama": "Budi Kurniawan",
      "email": "budi.kurniawan@yopmail.com",
      "npm": "77777",
      "tempat_lahir": "Surabaya",
      "tgl_lahir": "10-03-2000",
      "alamat": "Jl. Anggrek No. 09, Jember",
      "prodi": "Desain",
      "fakultas": "Desain",
      "foto": "Budi Kurniawan.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "15",
      "nama": "Budi Susanto",
      "email": "budi.susanto@yopmail.com",
      "npm": "44332",
      "tempat_lahir": "Surabaya",
      "tgl_lahir": "10-03-2000",
      "alamat": "Jl. Bendar No. 77, Surakarta",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatik",
      "foto": "Budi Susanto.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "16",
      "nama": "Budiman",
      "email": "budiman@email.com",
      "npm": "324543",
      "tempat_lahir": "Surabaya",
      "tgl_lahir": "10-03-2000",
      "alamat": "Jl. Bendar No. 77, Surakarta",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatik",
      "foto": "Budiman.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    },
    {
      "id_mahasiswa": "13",
      "nama": "Dendi Suandi",
      "email": "dendi.suandi@yopmail.com",
      "npm": "33554",
      "tempat_lahir": "Surabaya",
      "tgl_lahir": "10-03-2000",
      "alamat": "Jl. Bendar No. 77, Surakarta",
      "prodi": "Sistem Informasi Modern",
      "fakultas": "Teknik Informatik",
      "foto": "Dendi Suandi.png",
      "qr_code_text": null,
      "tgl_input": "2020-11-15",
      "id_user_input": "2"
    }
  ]
}

```

Atau pada browser langsung dibuka alamat tersebut kemudian klik kanan pilih View page source

The screenshot shows the browser's address bar with the URL `localhost/ci4/admin_template/data-tables-ajax/getDataDT`. The page content is partially visible, showing a table of student data. A right-click context menu is open, and the `View page source` option is highlighted.

Hasilnya adalah sebagai berikut:

```
view-source:localhost/ci4/admin_template/data-tables-ajax/getDataDT

1 { "draw":1,"recordsTotal":16,"recordsFiltered":16,"data":[{"id_mahasiswa":2,"nama":"Sony
Haryanto","email":"sony.haryanto@yopmail.com","npm":"22222","tempat_lahir":"Surabaya","tgl_lahir":"Surabaya, 01
Februari 1997","prodi":"Sistem Perangkat Lunak","fakultas":"Teknik Informatika","alamat":"Jl. Kenanga No, 137,
Surabaya","foto":"Sony Haryanto.png","qrcode_text":null,"tgl_input":"2020-11-
15","id_user_input":2,"tgl_edit":null,"id_user_edit":null,"ignore_search_foto":"<div class='list-foto'><img
src='http://localhost/ci4/admin_template/public/images/foto/Sony Haryanto.png'></div>";
"no_urut":1,"ignore_search_action":"<div class='form-inline btn-action-group'><a
href='http://localhost/ci4/admin_template/data-tables-ajax/edit?id=2' class='btn btn-success btn-xs me-1
1 btn btn-success btn-xs me-1'><span class='btn-label-icon'><i class='fa fa-edit pe-1'>
</i></span> Edit</a><form method='post' action='\"'><button type='submit'
data-action='delete-data' data-delete-title='Hapus data mahasiswa: <strong>Sony Haryanto</strong> ?'
class='btn btn-danger btn-xs'><span class='btn-label-icon'><i class='fa fa-times pe-1'>
</i></span> Delete</button><input type='hidden' name='delete'
value='delete'></div></div><input type='hidden' name='id' value='2'></div>"}]
```

Output diatas dihasilkan oleh method getDataDT yang ada pada file
app\\Controllers\\Data_tables_ajax.php. Adapun script pada method tersebut adalah sebagai berikut:

```
public function getDataDT() {

    $this->hasPermissionPrefix('read');

    $num_data = $this->model->countAllData( $this->whereOwn() );
    $result['draw'] = $start = $this->request->getPost('draw') ?: 1;
    $result['recordsTotal'] = $num_data;

    $query = $this->model->getListData( $this->whereOwn() );
    $result['recordsFiltered'] = $query['total_filtered'];

    helper('html');
    $id_user = $this->session->get('user')['id_user'];

    $no = $this->request->getPost('start') + 1 ?: 1;
    foreach ($query['data'] as $key => &$val)
    {
        $image = 'noimage.png';
        if ($val['foto']) {
            if (file_exists('public/images/foto/' . $val['foto'])) {
                $image = $val['foto'];
            }
        }


        $val['ignore_search_foto'] = '<div class="list-foto"></div>';
        $val['tgl_lahir'] = $val['tempat_lahir'] . ', ' . format_tanggal($val['tgl_lahir']);

        $val['no_urut'] = $no;
        $val['ignore_search_action'] = btn_action([
            'edit' => ['url' => $this->config->baseURL .
$this->currentModule['nama_module'] . '/edit?id=' . $val['id_mahasiswa']]
            , 'delete' => ['url' => ''
                , 'id' =>
                $val['id_mahasiswa']
                , 'delete-title' => 'Hapus
data mahasiswa: <strong>'.$val['nama'].'</strong> ?'
            ]
        ]);
        $no++;
    }

    $result['data'] = $query['data'];
    echo json_encode($result); exit();
}
```


Pada script diatas:

1. Kita definisikan kolom yang akan kita beri awalah ignore_search_
2. recordsTotal menunjukkan total semua data

| 10 |  | Fransisco Brian | Mei 2001 |
|----|---|-----------------|----------|
| No | Foto | Nama | |

Showing 1 to 10 of 16 entries

3. recordsFiltered menunjukkan jumlah data hasil filter, seperti ketika melakukan search data

| 3 |  | Budiman | Jakarta, 04 Februari 2001 |
|----|---|---------|---------------------------|
| No | Foto | Nama | TTL |

Showing 1 to 3 of 3 entries (filtered from 16 total entries)

B.6. Search, Sort, Show Number, dan Pagination

Ketika kita melakukan interaksi pada datatables yang menyebabkan perubahan data, maka datatables akan mengirim semua parameter yang ada ke server. Parameter yang dikirim ini dapat dilihat pada console browser, misal kita melakukan pencarian data dengan kata Budi, maka data yang dikirim adalah sebagai berikut:

Borderable%5D=true&columns%5B3%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B3%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B4%5D%5Bdata%5D=alamat&columns%5B4%5D%5Bname%5D=&columns%5B4%5D%5Bsearchable%5D=true&columns%5B4%5D%5Bborderable%5D=true&columns%5B4%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B4%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B5%5D%5Bdata%5D=npn&columns%5B5%5D%5Bname%5D=&columns%5B5%5D%5Bsearchable%5D=true&columns%5B5%5D%5Bborderable%5D=true&columns%5B5%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B5%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B6%5D%5Bdata%5D=prodi&columns%5B6%5D%5Bname%5D=&columns%5B6%5D%5Bsearchable%5D=true&columns%5B6%5D%5Bborderable%5D=true&columns%5B6%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B6%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B7%5D%5Bdata%5D=fakultas&columns%5B7%5D%5Bname%5D=&columns%5B7%5D%5Bsearchable%5D=true&columns%5B7%5D%5Bborderable%5D=true&columns%5B7%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B7%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B8%5D%5Bdata%5D=ignore_search_action&columns%5B8%5D%5Bname%5D=&columns%5B8%5D%5Bsearchable%5D=true&columns%5B8%5D%5Bborderable%5D=false&columns%5B8%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B8%5D%5Bsearch%5D%5Bregex%5D=false&order%5B0%5D%5Bcolumn%5D=2&order%5B0%5D%5Bdir%5D=asc&start=0&length=10&search%5Bvalue%5D=Budi&search%5Bregex%5D=false

Setiap kali ada perubahan data maka parameter yang dikirim sama seperti diatas hanya datanya saja yang berbeda, jika kita melakukan sorting data, misal kita urutkan data berdasarkan kolom ke tujuh (index kolom 6) sebeagai berikut:

| No | Foto | Nama ↑↓ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ | Prodi ↑↓ | Fakult |
|----|--|---------------|----------------------------|-------------------------------|--------|------------------------|---------------|
| 1 |  | Sony Haryanto | Surabaya, 01 Februari 1997 | Jl. Kenanga No, 137, Surabaya | 22222 | Sistem Perangkat Lunak | Teknik Inform |

Maka data yang terkirim adalah sebagai berikut:

Name

☐ getDataDT

×

Headers

Payload

Preview

Response

Initiator

Timing

Cookies

columns[8][searchable]: true

columns[8][orderable]: false

columns[8][search][value]:

columns[8][search][regex]: false

order[0][column]: 6

order[0][dir]: desc

start: 0

length: 25

search[value]:

Raw datanya adalah:

draw=20&columns%5B0%5D%5Bdata%5D=ignore_search_no_urut&columns%5B0%5D%5Bname%5D=&columns%5B0%5D%5Bsearchable%5D=true&columns%5B0%5D%5Bborderable%5D=false&columns%5B0%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B0%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B1%5D%5Bdata%5D=ignore_search_foto&columns%5B1%5D%5Bname%5D=&columns%5B1%5D%5Bsearchable%5D=true&columns%5B1%5D%5Bborderable%5D=false&columns%5B1%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B1%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B2%5D%5Bdata%5D=nama&columns%5B2%5D%5Bname%5D=&columns%5B2%5D%5Bsearchable%5D=true&columns%5B2%5D%5Bborderable%5D=true&columns%5B2%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B2%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B3%5D%5Bdata%5D=tgl_lahir&columns%5B3%5D%5Bname%5D=&columns%5B3%5D%5Bsearchable%5D=true&columns%5B3%5D%5Bborderable%5D=true&columns%5B3%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B3%5D%5Bsearch%5D%5Bregex%5D=false

```
lse&columns%5B4%5D%5Bdata%5D=alamat&columns%5B4%5D%5Bname%5D=&columns%5B4%5D%5Bsearchable%5D=true&
columns%5B4%5D%5Bborderable%5D=true&columns%5B4%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B4%5D%5Bsearch%
5D%5Bregex%5D=false&columns%5B5%5D%5Bdata%5D=npn&columns%5B5%5D%5Bname%5D=&columns%5B5%5D%5Bsearch
able%5D=true&columns%5B5%5D%5Bborderable%5D=true&columns%5B5%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B5
%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B6%5D%5Bdata%5D=prodi&columns%5B6%5D%5Bname%5D=&columns%
5B6%5D%5Bsearchable%5D=true&columns%5B6%5D%5Bborderable%5D=true&columns%5B6%5D%5Bsearch%5D%5Bvalue%
5D=&columns%5B6%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B7%5D%5Bdata%5D=fakultas&columns%5B7%5D%5
Bname%5D=&columns%5B7%5D%5Bsearchable%5D=true&columns%5B7%5D%5Bborderable%5D=true&columns%5B7%5D%5B
search%5D%5Bvalue%5D=&columns%5B7%5D%5Bsearch%5D%5Bregex%5D=false&columns%5B8%5D%5Bdata%5D=ignore_
search_action&columns%5B8%5D%5Bname%5D=&columns%5B8%5D%5Bsearchable%5D=true&columns%5B8%5D%5Bborder
able%5D=false&columns%5B8%5D%5Bsearch%5D%5Bvalue%5D=&columns%5B8%5D%5Bsearch%5D%5Bregex%5D=false&or
der%5B0%5D%5Bcolumn%5D=6&order%5B0%5D%5Bdir%5D=desc&start=0&length=25&search%5Bvalue%5D=&search%5
Bregex%5D=false
```

Semua data tersebut dikirim ke url sesuai dengan yang telah kita definisikan pada data-tables-ajax.js yaitu `http://localhost/ci4/admin_template/data-tables-ajax/getDataDT` Agar data yang ditampilkan akurat, parameter tersebut harus kita aplikasikan dengan benar.

Pada method `getDataDT` (file `app\Controlllers\Data_tables_ajax.php`), kita memanggil method `getListData` dari model `DataTablesAjaxModel` (file `app\Models\DataTablesAjaxModel.php`)

```
public function getDataDT() {

    // Script lainnya

    $query = $this->model->getListData( $this->whereOwn() );
    $result['recordsFiltered'] = $query['total_filtered'];

    // Script lainnya

}
```

Method `getListData()` ini yang bertanggung jawab mengimplementasikan semua parameter yang dikirim datatables. Script pada method `getListData()` adalah sebagai berikut:

```
public function getListData($where) {

    $columns = $this->request->getPost('columns');

    // Search
    $search_all = @$this->request->getPost('search')['value'];
    if ($search_all) {
        // Additional Search
        $columns[][ 'data' ] = 'tempat_lahir';
        foreach ($columns as $val) {

            if (strpos($val['data'], 'ignore_search') !== false)
                continue;

            if (strpos($val['data'], 'ignore') !== false)
                continue;

            $where_col[] = $val['data'] . ' LIKE "' . $search_all . '"';
        }
        $where .= ' AND ( ' . join(' OR ', $where_col) . ' ) ';
    }

    // Order
    $start = $this->request->getPost('start') ?: 0;
```

```

        $length = $this->request->getPost('length') ?: 10;

        $order_data = $this->request->getPost('order');
        $order = '';
        if (!empty($_POST) && strpos($_POST['columns'][$order_data[0]['column']]['data'],
'ignore_search') === false) {
            $order_by = $columns[$order_data[0]['column']]['data'] . ' ' .
strtoupper($order_data[0]['dir']);
            $order = ' ORDER BY ' . $order_by . ' LIMIT ' . $start . ', ' . $length;
        }

        // Query Total Filtered
        $sql = 'SELECT COUNT(*) AS jml_data FROM mahasiswa ' . $where;
        $data = $this->db->query($sql)->getRowArray();
        $total_filtered = $data['jml_data'];

        // Query Data
        $sql = 'SELECT * FROM mahasiswa
                ' . $where . $order;
        $data = $this->db->query($sql)->getResultArray();

        return ['data' => $data, 'total_filtered' => $total_filtered];
    }

```

B.6.1 Search

Pada script diatas, kita telah mengakomodir search data dan order data. Pada bagian search data kita melakukan pencarian data ke semua kolom tabel database yang kolomnya tidak mengandung awalah ignore_search atau ignore, daftar nama kolom ini sesuai dengan yang telah kita definisikan sebelumnya pada file app\Views\themes\modern\data-tables-ajax-result.php sebagai berikut:

```

$column =[
    'ignore_search_no_urut' => 'No'
    , 'ignore_search_foto' => 'Foto'
    , 'nama' => 'Nama'
    , 'tgl_lahir' => 'TTL'
    , 'alamat' => 'Alamat'
    , 'npm' => 'NPM'
    , 'prodi' => 'Prodi'
    , 'fakultas' => 'Fakultas'
    , 'ignore_search_action' => 'Action'
];

foreach ($column as $key => $val) {
    $column_dt[] = ['data' => $key];
}

<span id="dataTables-column" style="display:none"><?=json_encode($column_dt)?></span>

```

Adapun script searchnya adalah sebagai berikut:

```

$columns = $this->request->getPost('columns');

// Search
$search_all = @$this->request->getPost('search')['value'];
if ($search_all) {
    // Additional Search
    $columns[][ 'data' ] = 'tempat_lahir';
    foreach ($columns as $val) {

```

```

        if (strpos($val['data'], 'ignore_search') !== false)
            continue;

        if (strpos($val['data'], 'ignore') !== false)
            continue;

        $where_col[] = $val['data'] . ' LIKE "%' . $search_all . '"';
    }
    $where .= ' AND (' . join(' OR ', $where_col) . ') ' ;
}

```

Pada script diatas, pertama tama kita cek apakah parameter search (`@$this->request->getPost('search')['value']`) berisi data, jika ya maka kita lakukan proses pencarian.

Ketika melakukan pencarian, selain kolom yang telah kita definisikan, kita juga menambah kolom lain untuk dilakukan pencarian yaitu kolom tempat_lahir:

```

// Additional Search
$columns[['data'] = 'tempat_lahir';

```

Selanjutnya hasil dari variabel `$where` adalah:

```

WHERE 1 = 1 AND (nama LIKE "%Budi%" OR tgl_lahir LIKE "%Budi%" OR alamat LIKE "%Budi%" OR npm LIKE "%Budi%" OR prodi LIKE "%Budi%" OR fakultas LIKE "%Budi%" OR tempat_lahir LIKE "%Budi%")

```

B.6.2 Order, Show Number, Pagination

Pada script bagian order juga sama, kita cek apakah kolom mengandung kata ignore_search, jika tidak, maka tambahkan query SQL untuk order data

```

// Order
$start = $this->request->getPost('start') ?: 0;
$length = $this->request->getPost('length') ?: 10;

$order_data = $this->request->getPost('order');
$order = '';
if (!empty($POST) && strpos($POST['columns'][$order_data[0]['column']]['data'], 'ignore_search') === false) {
    $order_by = $columns[$order_data[0]['column']]['data'] . ' ' . 
    strtoupper($order_data[0]['dir']);
    $order = ' ORDER BY ' . $order_by . ' LIMIT ' . $start . ', ' . $length;
}

```

Pada script diatas, ketika kita klik kolom ke tujuh

| No | Foto | Nama ↑↓ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ | Prodi ↑↓ | Fakult |
|----|---|---------------|----------------------------|-------------------------------|--------|------------------------|---------------|
| 1 |  | Sony Haryanto | Surabaya, 01 Februari 1997 | Jl. Kenanga No, 137, Surabaya | 22222 | Sistem Perangkat Lunak | Teknik Inform |

Maka variabel order akan bernilai

```
ORDER BY prodi DESC LIMIT 0, 10
```

Selain order, script diatas juga telah mengakomodir Show Number dan Pagination, ketika kita merubah jumlah data yang ditampilkan, sebagai berikut:



| Show | 10 | entries | Search: <input type="text"/> | | | | |
|------|---|--------------|------------------------------|------------------------------|-------|-------------------------|-------------------|
| No | Foto | Nama | TTL | Alamat | NPM | Prodi | Fakultas |
| 1 |  | Ahmad Basuki | Surakarta, 10 Maret 2000 | Jl. Bendar No. 77, Surakarta | 55555 | Sistem Informasi Modern | Teknik Informatik |

Maka variabel `$length` akan berisi jumlah data yang ditampilkan, misal jika kita ubah nilai show menjadi 25, maka variabel `$order` akan bernilai:

```
ORDER BY prodi DESC LIMIT 0, 25
```

Demikian juga dengan pagination, jika kita klik halaman 2 dengan posisi nilai show 10, maka nilai variabel `$order` menjadi:

```
ORDER BY prodi DESC LIMIT 10, 10
```

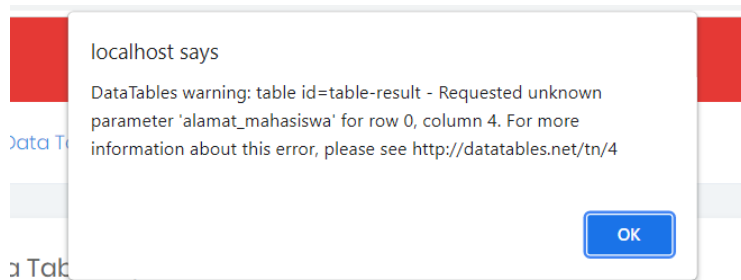
G. Troubleshooting

Ketika menjalankan datatables tidak jarang kita menemui error berupa popup windows yang berisi informasi kegagalan datatables untuk merender data. Umumnya error ini disebabkan karena 2 hal, yang pertama adalah kolom yang didefinisikan tidak ditemukan pada data yang ingin ditampilkan dan yang kedua adalah url tidak menghasilkan data sesuai dengan format yang telah ditentukan.

Untuk error pertama misal kita ubah definisi kolom dari alamat menjadi alamat_mahasiswa sebagai berikut:

```
$column =[
    'ignore_search_no_urut' => 'No'
    , 'ignore_search_foto' => 'Foto'
    , 'nama' => 'Nama'
    , 'tgl_lahir' => 'TTL'
    , 'alamat_mahasiswa' => 'Alamat'
    , 'npm' => 'NPM'
    , 'prodi' => 'Prodi'
    , 'fakultas' => 'Fakultas'
    , 'ignore_search_action' => 'Action'
];
```

Maka akan muncul pesan error sebagai berikut:



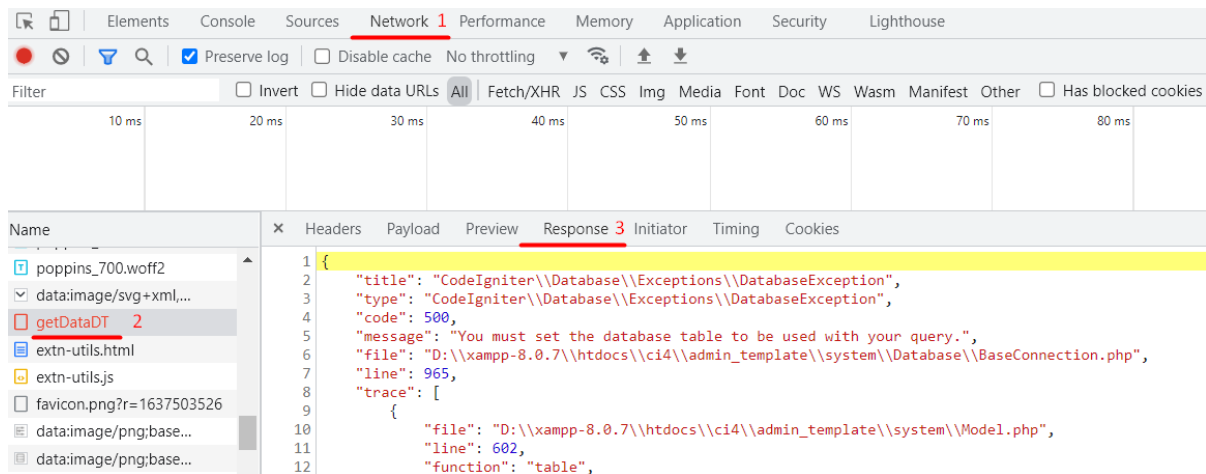
Meskipun terjadi error, maka data tetap ditampilkan, namun untuk kolom nama bernilai kosong

| No | Foto | Nama ↑↓ | TTL ↑↓ | Alamat ↑↓ | NPM ↑↓ |
|----|---|--------------|-----------------------------|-----------|--------|
| 1 |  | Ahmad Basuki | Surakarta, 10 Maret 2000 | | 55555 |

Error ini terjadi karena data yang diambil dari server tidak mengandung alamat_mahasiswa

```
{
  "draw": "1",
  "recordsTotal": "18",
  "recordsFiltered": "18",
  "data": [
    {
      "id_mahasiswa": "17",
      "nama": "Wahyu Jatmiko",
      "email": "wahyujatmiko@email.com",
      "npm": "345432",
      "tempat_lahir": "Surakarta",
      "tgl_lahir": "Surakarta, 05 April 2002",
      "prodi": "Desain Grafis",
      "fakultas": "Desain Komunikasi Visual",
      "alamat": "Jl. Arjuna No. 14, Surakarta",
      "foto": "<div class='list-foto'><img
src='http://localhost:7777/ci4/admin_template/public/images/foto/noimage.png'></div>",
      "qrcode_text": null,
      "tgl_input": null,
      "id_user_input": null,
      "tgl_edit": "2021-08-17",
      "id_user_edit": "1",
      "ignore_search_urut": 1,
      "ignore_search_action": // Button action
    }
    // Data lainnya
  ]
}
```

Selanjutnya pada bentuk error yang kedua, di mana data yang dihasilkan tidak sesuai dengan format yang telah ditentukan maka setelah muncul popup error, maka data tidak muncul. Untuk mengatasi hal ini, kita perlu mengidentifikasi output dari url pengambilan data, caranya dapat melalui tab network yang ada pada console browser sebagai berikut:



VII. Menggunakan RBAC

Admin Template ini sudah disertakan tools RBAC untuk mempermudah Anda mengatur akses data sesuai dengan role yang dimiliki oleh user.

A. Konfigurasi

Pengaturan global tools RBAC ini ada pada file `app/Config/App.php` variabel sebagai `$checkRoleAction` berikut:

```
public $checkRoleAction = ['enable_global' => true, 'field' => 'id_user_input'];
```

Jika nilai `enable_global` bernilai `true`, maka ketika halaman dibuka, sistem akan otomatis mengecek apakah role user boleh melakukan aksi `add`, `edit`, atau `delete` data. Berikut metode pengecekan nya:

- **Add dan Edit.** Sistem akan mengecek action dari controller, jika action bernilai `add` atau `edit`, maka akan dilakukan pengecekan apakah role boleh melakukan aksi `add` dan `edit`. Sehingga penting untuk mendefinisikan url untuk aksi tambah data menggunakan url http://localhost/nama_module/add dan `edit` data menggunakan url http://localhost/nama_module/edit
- **Delete.** System akan mengecek variabel `$_POST['delete']`, jika variabel tersebut didefinisikan maka akan dilakukan pengecekan apakah role diperbolehkan melakukan aksi `delete`, sehingga penting untuk menambahkan `<input type="hidden" name="delete">` pada form `delete`

Catatan: Pada global check (`enable_global = true`), hak akses hanya akan dicek jika role tersebut memiliki hak akses `No`, jika nilai hak akses `Own`, sistem tidak bisa melakukan pengecekan, pengecekan harus dilakukan di level module (dibahas dibawah).

Method yang digunakan untuk mengecek hak akses adalah `checkRoleAction()` untuk pengecekan global dan `cekHakAkses()` untuk pengecekan level controller. Kedua method tersebut ada di file `app/Controllers/BaseController.php`

Contoh penerapan pada level global: misal user biasa (username: user, pass: user) kita beri hak akses ke module produk sebagai berikut:

☒ User

| | |
|--------|-----------------------------------|
| Read | own Hanya Data Miliknya Sendiri |
| Create | no TIDAK |
| Update | own Hanya Data Miliknya Sendiri |
| Delete | own Hanya Data Miliknya Sendiri |

Pada hak akses diatas, user tidak diperbolehkan untuk menambah data dan hanya dapat membaca, mengedit, dan menghapus data miliknya sendiri.

Selanjutnya, jika user user membuka halaman produk dan mengklik tombol + Tambah Data

Produk

[+ Tambah Data](#)

| No | Nama Produk | Deskripsi Produk | Aksi |
|----|--------------------------------------|--|---|
| 1 | Bluetooth Multi-Device Keyboard K480 | Keyboard meja wireless untuk komputer, tablet, dan smartphone | Edit Delete |
| 2 | USB Unifying Receiver | Receiver USB yang bisa digunakan untuk sebuah mouse atau keyboard unifying | Edit Delete |
| 3 | M590 Multi-Device Silent | Mouse wireless hening untuk power user | Edit Delete |
| 4 | Wireless Touch Keyboard K400 Plus | Keyboard HTPC untuk TV yang terhubung dengan PC | Edit Delete |

Maka akan muncul pesan error sebagai berikut:

Jagowebdev

Home / Produk

Error

Role Anda tidak diperkenankan untuk menambah data

Selanjutnya mari kita ubah hak akses user agar dapat menambahkan data

| | |
|--|-------------------------------------|
| <input checked="" type="checkbox"/> User | |
| Read | own Hanya Data Miliknya Sendiri ▼ |
| Create | yes YA ▼ |
| Update | own Hanya Data Miliknya Sendiri ▼ |
| Delete | own Hanya Data Miliknya Sendiri ▼ |

Dengan hak akses diatas, user boleh menambah data, namun tetap hanya dapat membaca, mengedit, dan menghapus data miliknya sendiri. Selanjutnya jika user mengklik tombol + Tambah Data, akan muncul form isian data produk.

B. Read Data

Untuk hak akses read data, agar nantinya dapat dilakukan pengecekan otomatis, kita perlu tambahkan kolom id_user_input pada tabel database, selanjutnya setiap kali menyimpan data, kita simpan data id_user ke kolom tersebut. Sebagai contoh kita ubah script pada model produk (file app/Models/ProdukModel.php) method saveData() sehingga dapat menyimpan id_user user yang menginput data, script sebelumnya adalah sebagai berikut:

```
public function saveData($id)
{
    $data_db['nama_produk'] = $_POST['nama_produk'];
    $data_db['deskripsi_produk'] = $_POST['deskripsi_produk'];
    $id_produk = $id;

    $builder = $this->db->table('produk');
    if (empty($id)) {
        $builder->insert($data_db);
        $id_produk = $this->db->insertID();
    } else {
        $builder->update($data_db, ['id_produk' => $_POST['id']]);
    }

    return ['query' => $this->db->error(), 'id_produk' => $id_produk];
}
```

Selanjutnya kita tambahkan script untuk menyimpan id_user yang menginput, sehingga script simpan data menjadi sebagai berikut:

```
public function saveData($id)
{
    $data_db['nama_produk'] = $_POST['nama_produk'];
    $data_db['deskripsi_produk'] = $_POST['deskripsi_produk'];
    $data_db['id_user_input'] = $this->session->get('user')['id_user'];
    $id_produk = $id;

    // Script lainnya
}
```

Selanjutnya mari kita coba untuk menyimpan data berikut:

Tambah Data Produk

Nama Produk

K380 Multi-Device Bluetooth Keyboard

Deskripsi Produk

Keyboard minimalis untuk komputer, tablet, dan ponsel

Submit

Setelah klik submit, maka tabel produk pada database akan tampak sebagai berikut:

| id_produk | nama_produk | deskripsi_produk | id_user_input | tgl_input | id_user_edit | tgl_edit |
|-----------|--------------------------------------|--|---------------|------------|--------------|------------|
| 1 | Bluetooth Multi-Device Keyboard K480 | Keyboard meja wireless untuk komputer, tablet,... | (NULL) | 2021-01-29 | (NULL) | 2021-01-29 |
| 2 | USB Unifying Receiver | Receiver USB yang bisa digunakan untuk sebuah... | (NULL) | 2021-01-29 | (NULL) | 2021-01-29 |
| 3 | M590 Multi-Device Silent | Mouse wireless hening untuk power user | (NULL) | 2021-01-29 | (NULL) | 2021-01-29 |
| 4 | Wireless Touch Keyboard K400 Plus | Keyboard HTPC untuk TV yang terhubung dengan de... | (NULL) | 2021-01-29 | (NULL) | 2021-01-29 |
| 5 | K380 Multi-Device Bluetooth Keyboard | Keyboard minimalis untuk komputer, tablet, dan... | 2 | 2021-01-29 | (NULL) | 2021-01-29 |

Pada contoh diatas, pada id_produk 5 yang baru saja kita tambahkan, id_user_input nya bernilai 2, yaitu id_user dari user yang login. Untuk baris yang lain (yang bernilai NULL) kita biarkan apa adanya.

Selanjutnya mari kita kembali ke data produk yang ditampilkan ke user. Sebelumnya, hak akses user untuk read data adalah own, namun ketika membuka halaman produk, semua data muncul

Produk

+ Tambah Data

| No | Nama Produk | Deskripsi Produk | Aksi |
|----|---------------------------------------|--|---|
| 1 | Bluetooth Multi-Device Keyboard K4809 | Keyboard meja wireless untuk komputer, tablet, dan smartphone | Edit Delete |
| 2 | USB Unifying Receiver | Receiver USB yang bisa digunakan untuk sebuah mouse atau keyboard unifying | Edit Delete |
| 3 | M590 Multi-Device Silent | Mouse wireless hening untuk power user | Edit Delete |
| 4 | Wireless Touch Keyboard K400 Plus | Keyboard HTPC untuk TV yang terhubung dengan PC | Edit Delete |
| 5 | K380 Multi-Device Bluetooth Keyboard | Keyboard minimalis untuk komputer, tablet, dan ponsel | Edit Delete |

Hal ini terjadi karena kita belum melakukan filter pada data. Untuk melakukannya kita tambahkan filter pada query pengambilan data. Pada app\Controllers\Produk.php method index kita memanggil method `$this->model->getProduk();` pada model. Selanjutnya, pada model, didalam method `getProduk()` terdapat query sebagai berikut:

```
$sql = 'SELECT * FROM produk';
```

Untuk memfilter data agar tampil sesuai dengan pemiliknya, kita tambahkan method `$this->whereOwn()` pada argumen `$this->model->getProduk()` yang ada pada controller `Produk.php` sehingga bentuk nya menjadi:

```
$data['result'] = $this->model->getProduk( $this->whereOwn() );
```

Method `$this->whereOwn()` ini ada di file `app/Controllers/BaseController.php` method ini akan otomatis menambahkan klausa `WHERE` sebagai berikut:

```
WHERE id_user_input = . $_SESSION['user']['id_user'];
```

Sehingga jika dijalankan, query diatas menjadi:

```
$sql = 'SELECT * FROM produk WHERE id_user_input = 2';
```

Perhatikan bahwa pada query diatas yang dilakukan pengecekan adalah kolom `id_user_input`. Fungsi `$this->whereOwn()` menggunakan kolom ini karena pada konfigurasi `public $checkRoleAction` bagian `field` yang ada di `app/Config/App.php` bernilai `id_user_input`.

```
public $checkRoleAction = ['enable_global' => true, 'field' => 'id_user_input'];
```

Selanjutnya, ketika kita jalankan, maka yang muncul hanya data dengan `id_user_input` yang bernilai 2 sebagai berikut

Home / Produk

Produk

+ Tambah Data

| No | Nama Produk | Deskripsi Produk | Aksi |
|----|--------------------------------------|---|-----------------------|
| 1 | K380 Multi-Device Bluetooth Keyboard | Keyboard minimalis untuk komputer, tablet, dan ponsel | <div>EditDelete</div> |

C. Edit Data

Pada setting role user yang telah kita definisikan sebelumnya, hak akses user untuk `update_data` adalah `Own` yang artinya hanya bisa mengupdate data miliknya sendiri. Agar user hanya dapat mengedit data miliknya sendiri, kita perlu tambahkan method `$this->cekHakAkses('update_data')` pada file controller `Produk.php` method `'edit'` sebagai berikut:

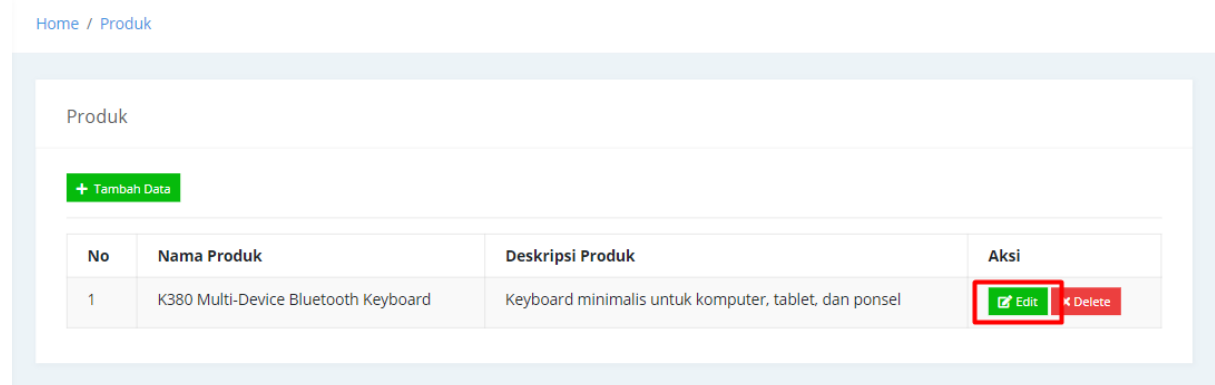
```
// Script lainnya
```

```
public function edit()
{
    $this->cekHakAkses('update_data');

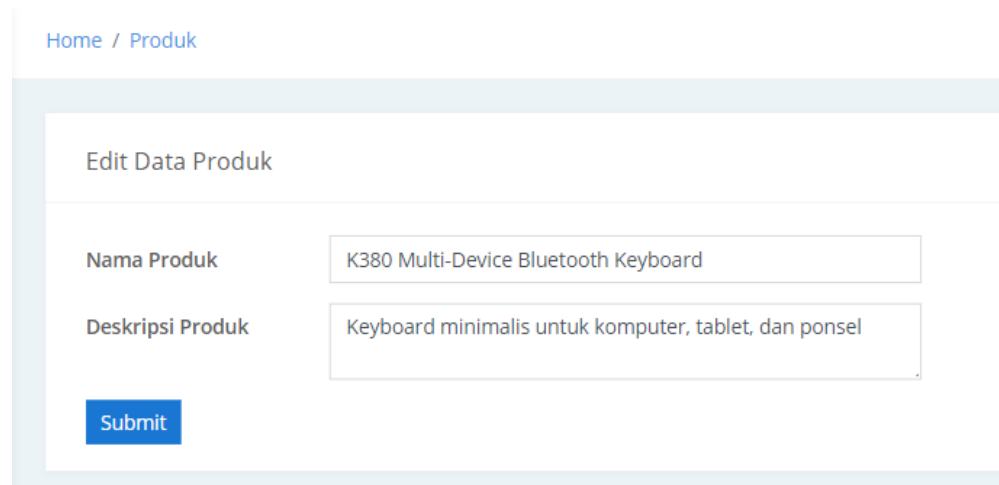
    if (empty($_GET['id']))
        $this->errorDataNotFound();

    //Script lainnya
}
```

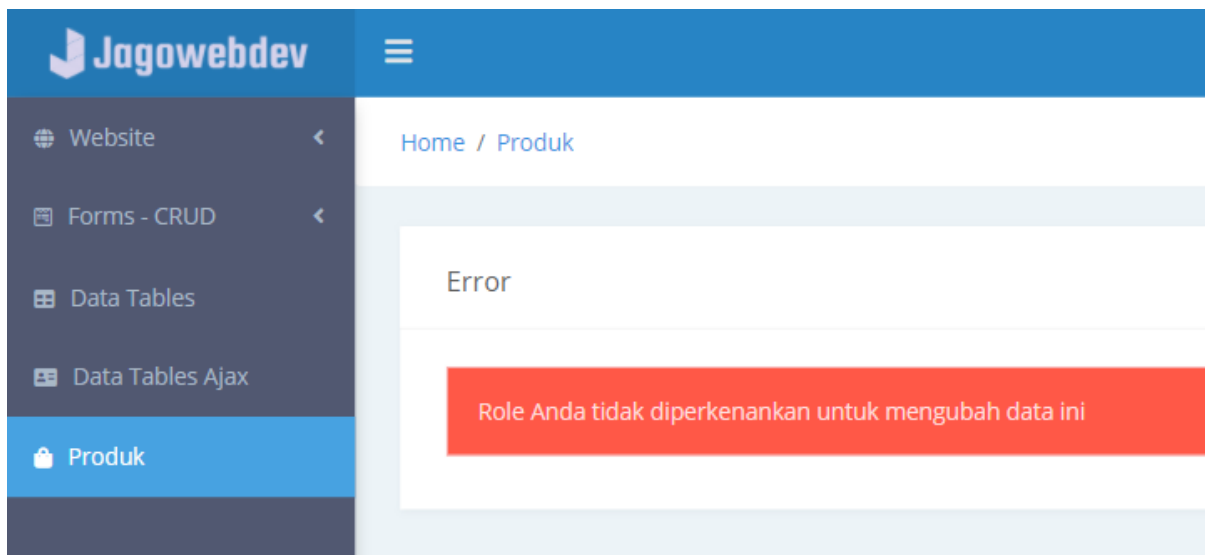
Selanjutnya jika kita klik tombol edit:



Maka kita akan diarahkan ke alamat http://localhost/ci4/admin_template/produk/edit?id=5 dengan tampilan sebagai berikut:



Jika parameter id kita ubah menjadi misal 4, sehingga url menjadi http://localhost/ci4/admin_template/produk/edit?id=4 maka yang kita dapati adalah sebagai berikut:



Method `$this->cekHakAkses('update_data')` yang kita tambahkan pada controller `Produk.php` ada di file `BaseController.php`, pada contoh diatas, method tersebut akan mengecek apakah produk dengan `id_produk 5` adalah milik user yang sedang login (`id_user 2`) sehingga boleh diedit, yaitu dengan melakukan pengecekan kolom `id_user_input`.

Method `$this->cekHakAkses('update_data')` tahu yang dicek tabel produk karena kita tidak mendefinisikan nama tabel pada method tersebut, sehingga method tersebut berasumsi bahwa nama tabel sama dengan nama module yaitu produk. Selanjutnya pada tabel produk, method akan mengambil data yang kolom `id_produk` nya bernilai 5. Method tahu kolom yang digunakan adalah `id_produk` karena kita tidak mendefinisikan nama kolom, sehingga method berasumsi bahwa kolom `id` untuk data produk adalah `id_ + nama tabel`

Parameter lengkap untuk method `$this->cekHakAkses()` adalah:

```
protected function cekHakAkses($action, $table_column = null, $column_check = null)
```

Sehingga, pada contoh diatas, jika method `$this->cekHakAkses()` kita tulis lengkap menjadi

```
$this->cekHakAkses('update_data', 'produk|id_produk', 'id_user_input');
```

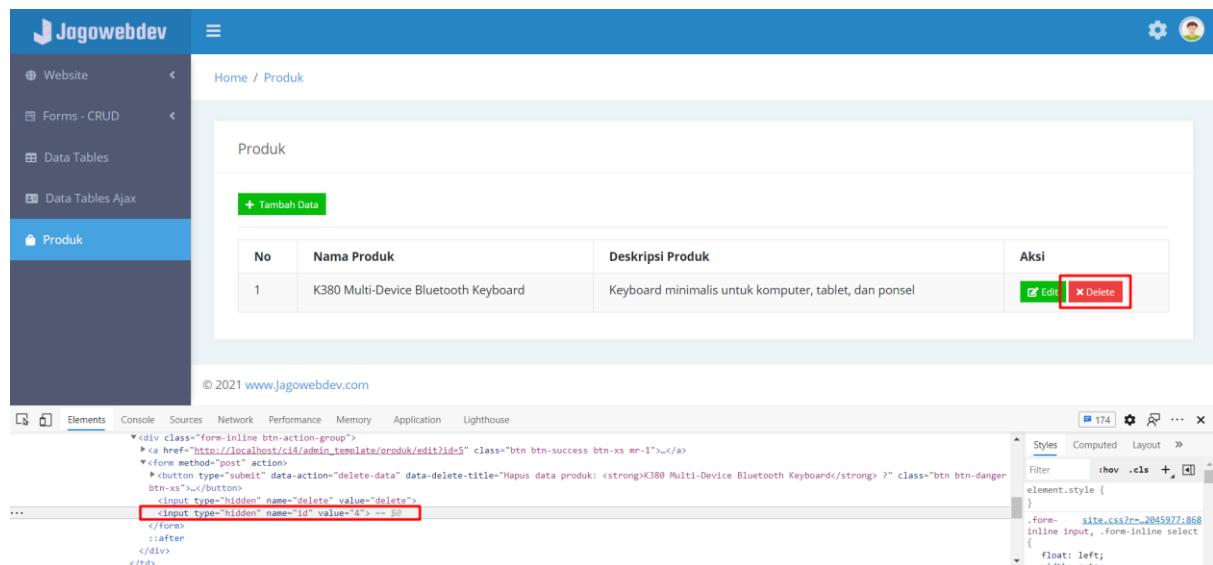
D. Delete Data

Selanjutnya mari kita atur parameter untuk delete data agar sesuai dengan hak akses yang telah ditetapkan. Pada contoh diatas, hak akses delete data untuk role user adalah `own` yang artinya user hanya dapat menghapus data miliknya sendiri. Untuk keperluan tersebut, caranya sama seperti edit, yaitu kita cukup tambahkan method `$this->cekHakAkses('delete_data')` sebagai berikut:

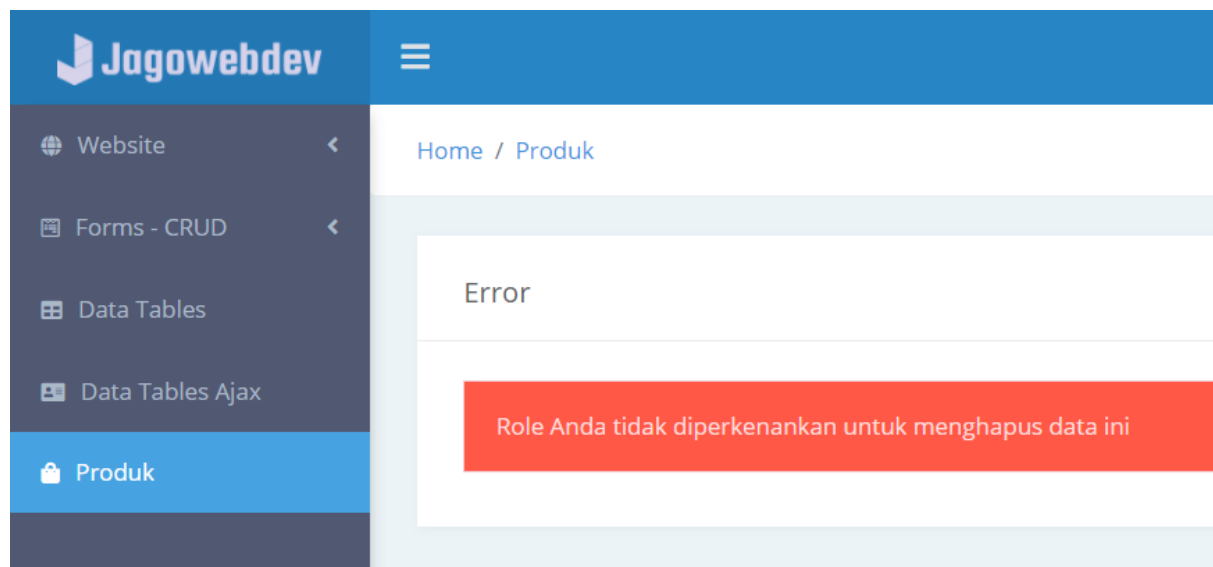
```
public function index()
{
    $data = $this->data;
    if ($this->request->getPost('delete'))
    {
        $this->cekHakAkses('delete_data');
        $delete = $this->model->deleteProdukById($_POST['id']);
    }
}
```

// Script lainnya

Selanjutnya jika kita tes dengan mengubah nilai id dan klik delete



Maka hasil yang kita peroleh adalah sebagai berikut:

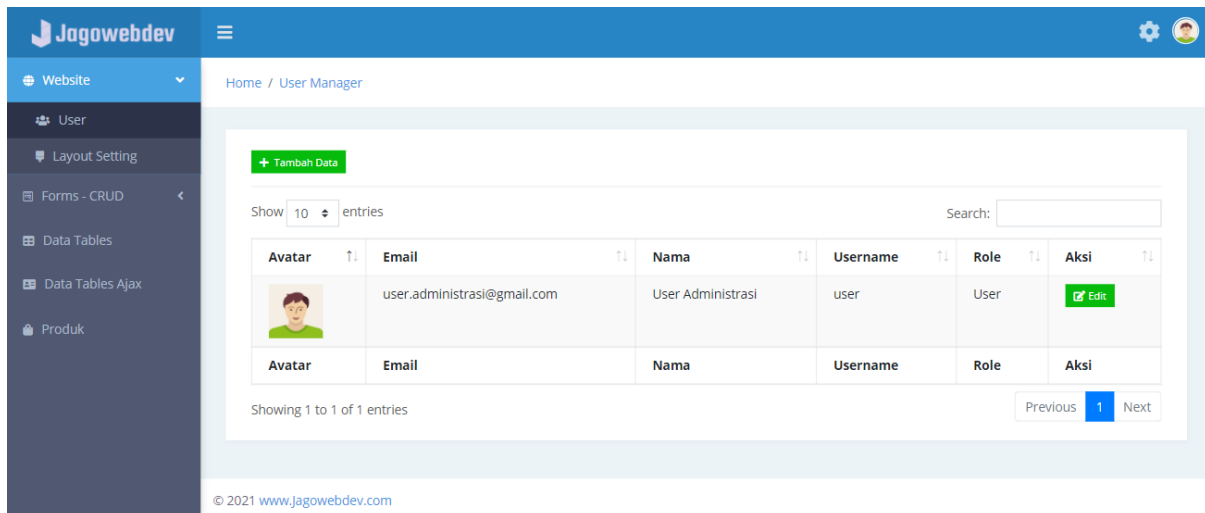


E. Studi Kasus

Contoh lain penerapan RABC adalah pada module User (app/Controllers/User.php).

Add Data

Pada module ini, user hanya dapat melihat data miliknya sendiri. Misal ketika user (username: user, pass: user) login dan membuka menu user, maka akan tampil data miliknya sendiri sebagai berikut:



Agar data user yang ditampilkan hanya miliknya sendiri, kita tambahkan metod `$this->whereOwn('id_user')` pada controller User sebagai berikut:

Pada method index:

```
$data['users'] = $this->model->getListUsers($this->actionUser, $this->whereOwn('id_user'));
```

Pada method `getDataDT`

```
// Script lainnya
$num_users = $this->model->countAllUsers($this->whereOwn('id_user'));
// Script lainnya
$query = $this->model->getListUsers($this->actionUser, $this->whereOwn('id_user'));
// Script lainnya
```

Perhatikan bahwa pada contoh diatas, kita tambahkan argumen `id_user` pada method `whereOwn()`, dengan argumen ini, maka method akan mengecek kolom `id_user` untuk dicocokkan dengan `id_user` yang login. Hal ini berbeda dengan contoh pada controller `Produk.php` dimana kita cukup menuliskan method `whereOwn()` tanpa argumen, karena pada controller `Produk.php`, kolom yang dicek adalah `id_user_input` (sesuai dengan parameter field pada property **protected** `$checkRoleAction` yang ada di file `app/Config/App.php`)

Edit Data

Agar user hanya dapat mengedit data miliknya sendiri, maka kita tambahkan method `cekHakAkses()` pada method `edit()` sebagai berikut:

```
public function edit()
{
    $this->cekHakAkses('update_data', 'user', 'id_user');

    // Script lainnya
```

Pada method `edit` tersebut, method `cekHakAkses()` kita tambahkan parameter kolom `id_user` karena kolom yang ingin kita cocokkan dengan `id user` yang login berbeda dengan kolom default yaitu `id_user_input`. Sedangkan parameter tabel yaitu `user`, boleh ditambahkan boleh tidak, karena

nama tabel sudah sama dengan nama module/controller yaitu user dan nama kolom untuk pengambilan data juga sudah sesuai yaitu id_ + nama tabel atau id_user, jika tidak ingin menambahkan parameter tabel, maka bentuk method menjadi:

```
$this->cekHakAkses('update_data', null, 'id_user');
```

Delete Data

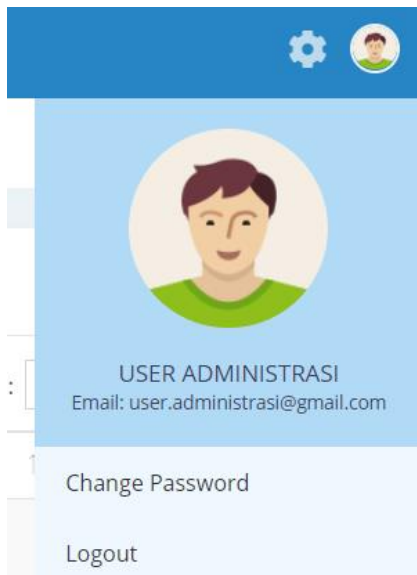
Untuk hak akses delete data, sama dengan aksi edit data, yaitu kita tambahkan method cekHakAkses() dengan parameter lengkap, sebagai berikut:

```
public function index()
{
    $this->cekHakAkses('read_data');
    if ($this->request->getPost('delete'))
    {
        $this->cekHakAkses('delete_data', 'user', 'id_user');
```

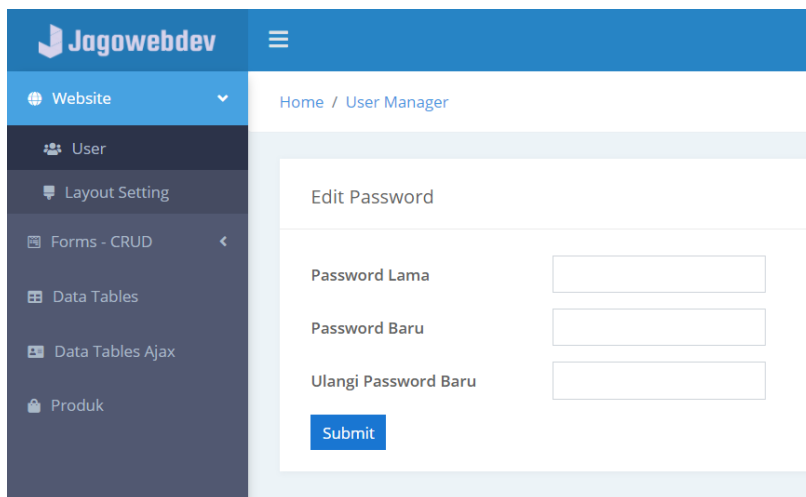
// Script lainnya

Update Password

Selanjutnya, pada menu user, menu update passwordnya terpisah dari menu edit dan delete, yaitu ada di menu user di pojok kanan atas



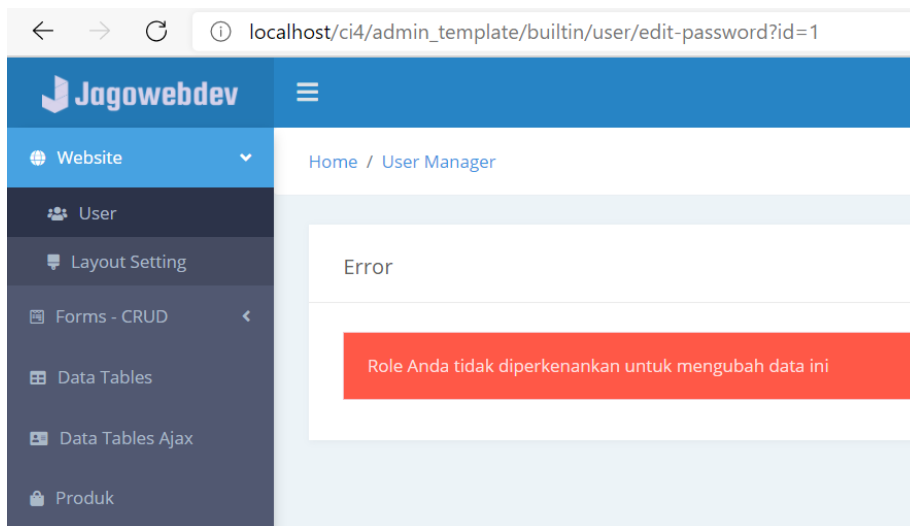
Ketika menu Change Password di klik, maka kita akan diarahkan ke url http://localhost/ci4/admin_template/user/edit-password?id=2 dengan tampilan halaman sebagai berikut:



Pada kondisi diatas, user dapat saja mengubah id pada url sehingga dapat mengubah password user lain, untuk mengatasi hal tersebut, sama seperti pada edit data yaitu ditambahkan method cekHakAkses() sebagai berikut:

```
public function edit_password()  
{  
    $this->cekHakAkses('update_data', 'user', 'id_user');  
}  
  
// Script lainnya
```

Dengan demikian, jika user mencoba mengakses data user lain misal dengan mengganti id pada url menjadi 1 maka akan muncul error sebagai berikut:



Contoh Lain

Contoh lain penerapan RBAC ini dapat dilihat pada contoh script datatables dan datatables-ajax

VIII. Menggunakan CSRF Token

Codeigniter telah menyediakan tools untuk mempermudah kita bekerja dengan CSRF token, tools tersebut dapat diaktifkan melalui konfigurasi pada app/Config/Filters.php

Ketika fitur CSRF tersebut diaktifkan, maka otomatis semua form akan dicek tokennya, jika ingin dikecualikan dari pengecekan, kita harus menuliskan daftar url yang tidak ingin dicek form nya.

Hal tersebut menurut kami kurang fleksibel karena untuk mengecualikan form dari pengecekan harus melalui url. Untuk itu, kami mengembangkan sendiri tools csrf yang mudah digunakan dan lebih fleksibel.

Tools CSRF yang kami kembangkan ini dapat diatur melalui file konfigurasi app/Config/App.php sebagai berikut:

```
public $csrf = [  
    'enable' => true,  
    'auto_settoken' => true,  
    'auto_check' => false,  
    'name' => 'csrf_token',  
    'expire' => 7200  
];
```

Mengaktifkan CSRF

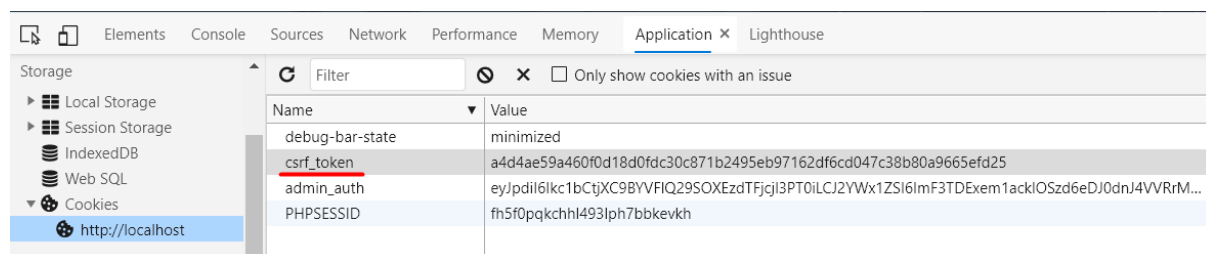
Untuk mengaktifkan fitur ini, ubah parameter enable menjadi true. Ketika aktif maka otomatis sistem akan me load file helper csrf_helper.php yang ada di folder app/Helpers. File helper tersebut berisi berbagai fungsi yang bermanfaat ketika bekerja dengan csrf token.

CSRF Otomatis

Tool csrf ini dapat dijalankan secara otomatis caranya ubah parameter auto_settoken dan auto_check menjadi true, misal sebagai berikut:

```
public $csrf = [  
    'enable' => true,  
    'auto_settoken' => true,  
    'auto_check' => true  
    'name' => 'csrf_token',  
    'expire' => 7200  
];
```

Ketika parameter auto_settoken bernilai true, maka setiap kali halaman dibuka, aplikasi akan membuat token dan menyimpannya di cookie browser. Nama cookie ini sesuai dengan nilai parameter name, pada contoh diatas nama cookie nya adalah csrf_token. Contoh sebagai berikut:



Selanjutnya, ketika parameter auto_check bernilai true, maka **setiap ada form dengan method post di submit**, sistem akan melakukan pengecekan apakah token yang disimpan di cookie tadi sesuai dengan token yang disubmit, sehingga penting diperhatikan bahwa ketika auto_check bernilai true, maka setiap membuat form, kita **harus** membuat input field yang menyimpan data

token. Untuk membuat field tersebut, kita cukup menggunakan fungsi `csrf_formfield()`. Contoh sebagai berikut:

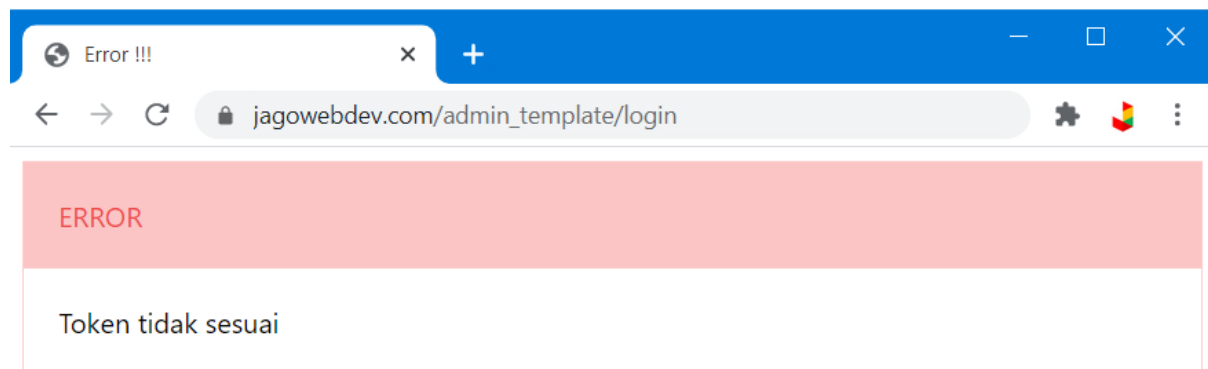
```
<form method="post" action="/login">
    <input type="text" name="username">
    <input type="password" name="password">
    <button type="submit" name="submit">Submit</button>
    <?php csrf_formfield(); ?>
</form>
```

Ketika form tersebut dibuka di browser, maka kode HTML yang dihasilkan adalah sebagai berikut:

```
<form method="post" action="/login">
    <input type="text" name="username">
    <input type="password" name="password">
    <button type="submit" name="submit">Submit</button>
    <input type="hidden" name="csrf_token"
value="a4d4ae59a460f0d18d0fdc30c871b2495eb97162df6cd047c38b80a9665efd25">
</form>
```

Note: Nama input yaitu `name="csrf_token"` sesuai dengan nilai pada parameter `name`

Selanjutnya, ketika form tersebut disubmit maka field `csrf_token` pada form akan dibandingkan dengan nilai `csrf_token` pada cookie (`$_POST['csrf_token'] == $_COOKIE['csrf_token']`), jika terjadi error, maka program akan otomatis berhenti dan muncul pesan error sebagai berikut:



Script tampilan error tersebut berada di file `app/Views/app_error.php`

CSRF Semi Otomatis

Untuk menggunakan metode ini, ubah parameter `auto_settoken` menjadi `true` sedangkan `auto_check` menjadi `false`, misal sebagai berikut:

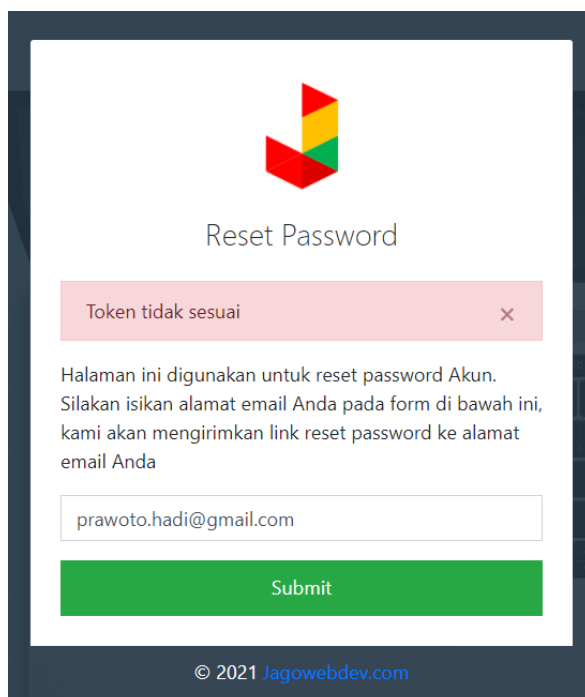
```
public $csrf = [
    'enable' => true,
    'auto_settoken' => true,
    'auto_check' => false,
    'name' => 'csrf_token',
    'expire' => 7200,
];
```


Pada metode ini, pengecekan token dilakukan secara manual menggunakan fungsi `csrf_validation()`, contoh seperti script berikut ini:

```
<?php
if (!empty($_POST['submit'])) {
    $cek_csrf = csrf_validation();
    if ($cek_csrf['status'] == 'error') {
        echo 'CSRF Error: ' . $cek_csrf['message'];
    }
}
?>
<form method="post" action="/login">
    <input type="text" name="username">
    <input type="password" name="password">
    <button type="submit" name="submit">Submit</button>
    <?php csrf_field(); ?>
</form>
```

Note: contoh CSRF Semi Otomatis ini dapat dilihat pada module register (`app/Controllers/Register.php`) dan recovery (`app/Controllers/Recovery.php`).

Contoh tampilan errornya adalah sebagai berikut:



CSRF Manual

Selain otomatis, tools csrf ini juga dapat digunakan secara manual. Dengan metode manual ini, maka token dibuat hanya ketika diperlukan saja, sehingga aplikasi lebih bersih.

Untuk menggunakan metode ini, pertama tama kita ubah parameter `auto_settoken` dan `auto_check` mejadi `false`, misal sebagai berikut:

```
public $csrf = [
    'enable' => true,
    'auto_settoken' => false,
    'auto_check' => false,
    'name' => 'csrf_token',
    'expire' => 7200
];
```

Selanjutnya, sebelum membuat form, pertama tama kita set terlebih dahulu token pada cookie, caranya, jalankan fungsi csrf_settoken(), selanjutnya pada form input kita buat field csrf dengan menjalankan fungsi csrf_formfield(). Contoh penerapannya adalah sebagai berikut:

```
<?php
csrf_settoken();
?>
<form method="post" action="/login">
    <input type="text" name="username">
    <input type="password" name="password">
    <button type="submit" name="submit">Submit</button>
    <?php csrf_field(); ?>
</form>
```

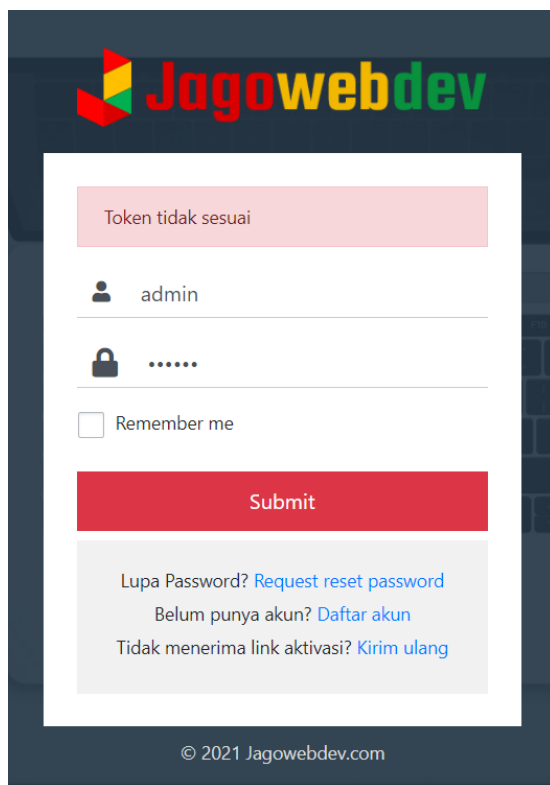
Selanjutnya untuk melakukan pengecekan, sama seperti pengecekan yang kita lakukan sebelumnya yaitu menggunakan fungsi csrf_validation() sebagai berikut:

```
<?php
csrf_settoken();

if (!empty($_POST['submit'])) {
    $cek_csrf = csrf_validation();
    if ($cek_csrf['status'] == 'error') {
        echo 'CSRF Error: ' . $cek_csrf['message'];
    }
}
?>
<form method="post" action="/login">
    <input type="text" name="username">
    <input type="password" name="password">
    <button type="submit" name="submit">Submit</button>
    <?php csrf_field(); ?>
</form>
```

NOTE: Contoh penerapan CSRF secara manual ada di module login (app/Controllers/Login.php).

Contoh tampilan errornya adalah sebagai berikut:



IX. Menggunakan Email

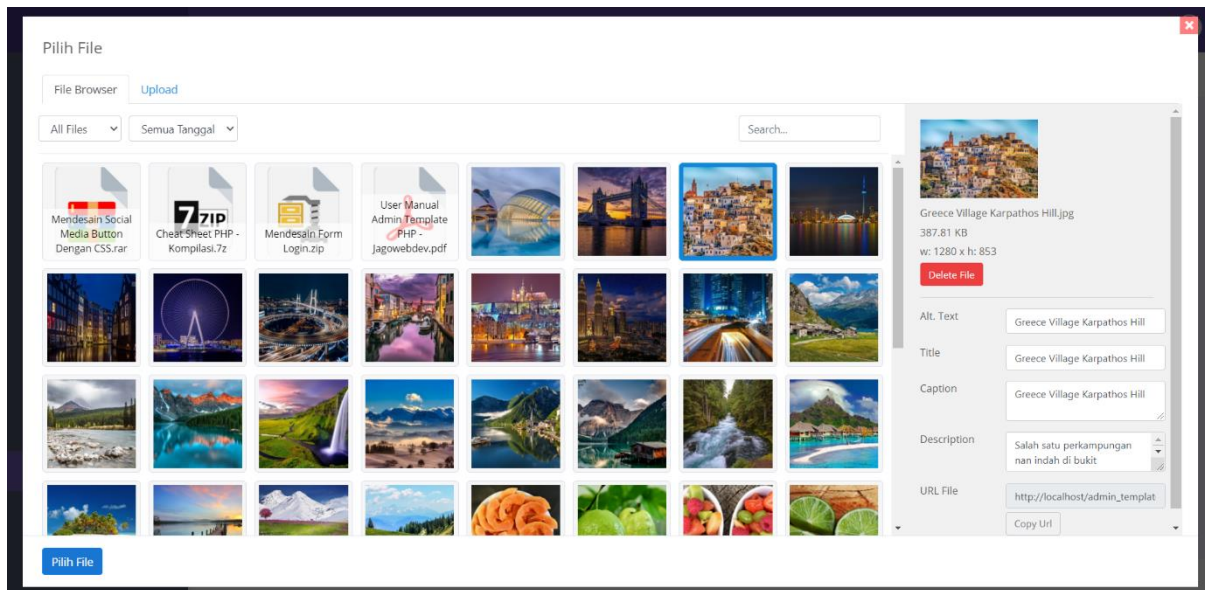
Aplikasi ini sudah menyertakan library PHP Mailer untuk mengirim email. Konfigurasi email yang ada di aplikasi ini dapat digunakan untuk mengirim email menggunakan berbagai layanan, mulai dari email Shared Hosting, Email Hosting, GMAIL standard authentication (less secure application), Gmail OAuth 2, dan AmazonSES

Untuk konfigurasi OAuth 2 token pada GMAIL dapat dibaca di file PDF Setting OAuth2 Gmail.pdf yang disertakan pada user manual. Konfigurasi email ada di file app/Config/EmailConfig.php. Untuk contoh scriptnya bisa melihat script kirim email ada di menu PDF & Send Email (app/Controllers/Pdfkirimemail.php).

X. Filepicker

Kami telah menyertakan plugin File Picker untuk memudahkan Anda memilih file untuk berbagai keperluan. File Picker ini kami kembangkan sendiri sehingga kompatibel dengan aplikasi Admin Template ini, disamping juga dapat digunakan di luar admin template ini.

File Picker ini berbeda dengan File Manager. Jika File Manager lebih seperti file browser, File Picker ini menggunakan database untuk menyimpan data file beserta atributnya. Tabel database yang digunakan adalah tabel **file_picker** Adapun tampilannya adalah sebagai berikut:



1. Lokasi Plugin

Plugin ini berlokasi di public/vendors/jwdfilepicker

2. Konfigurasi

Untuk dapat menggunakan plugin ini, perlu melakukan beberapa konfigurasi yaitu:

- File app/Config/Filepicker.php

```
<?php
namespace Config;

use CodeIgniter\Config\BaseConfig;
class Filepicker extends BaseConfig
{
    // Relative path
    public $uploadPath = 'public/files/uploads/';
    public $serverURL = 'filepicker/';
    public $iconPath = 'public/images/filepicker_images/';
    public $itemPerPage = 50;
    public $thumbnail = [
        'small' => ['w' => 250, 'h' => 250],
        'medium' => ['w' => 450, 'h' => 450]
    ];

    public function __construct()
    {
        $config = new \Config\App();
        $this->uploadURL = $config->baseURL . $this->uploadPath;
        $this->uploadPath = realpath(__DIR__ . '/../..') . '/' . $this->uploadPath;
        $this->serverURL = $config->baseURL . $this->serverURL;
        $this->iconURL = $config->baseURL . $this->iconPath;
        $this->iconPath = realpath(__DIR__ . '/../..') . '/' . $this->iconPath;
    }
}
```

- uploadPath: lokasi folder dimana file akan diupload

- serverURL: url backend untuk memproses file
 - iconPath: lokasi file icon yang digunakan oleh plugin
 - itemPerPage: menampilkan jumlah item ketika halaman di scroll
 - thumbnail: mengatur ukuran thumbnail ketika mengupload file
- b. Definisikan variabel javascript `filepicker_server_url` dan `filepicker_icon_url` Kedua variabel ini digunakan oleh plugin untuk pemrosesan data. Contoh pendefinisian variabel ini ada di file `app/Controllers/Filepicker.php` sebagai berikut:

```
$this->addJs('
    var filepicker_server_url = '' . $this->configFilepicker->serverURL . '';
    var filepicker_icon_url = '' . $this->configFilepicker->iconURL . '';
    , true
);
```

Kedua variabel ini nantinya akan digunakan untuk mendefinisikan variabel global pada plugin, yaitu di file `public/themes/modern/js/jwdfilepicker-default.js`

- c. File `public/themes/modern/js/jwdfilepicker-default.js`

```
$(document).ready(function() {
    jwdfilepicker.setDefaults({
        server_url : filepicker_server_url,
        icon_url : filepicker_icon_url
    });
});
```

Pendefinisian ini bertujuan untuk mendefinisikan secara global parameter `server_url` dan `icon_url` yang diperlukan oleh plugin. Selain global, pendefinisian ini juga dapat dilakukan secara manual ketika plugin dieksekusi.

- d. Scroll

Untuk menampilkan daftar file, plugin ini tidak menggunakan pagination, tetapi infinity scroll, jumlah item yang ditampilkan ketika awal plugin dibuka maupun ketika halaman dicroll di tentukan oleh nilai property `itemPerPage` yang ada di `app/Config/Filepicker.php`

3. Eksekusi Plugin

Contoh fungsi untuk mengeksekusi plugin adalah sebagai berikut:

```
jwdfilepicker.init({
    title : 'Pilih File',
    filter_file : '',
    onSelect: function ($elm) {
        var meta = JSON.parse($elm.find('.meta-file').html());
        $('filename').val(meta.nama_file);
        $('id-file-picker').val(meta.id_file_picker);
    }
});
```

Parameter `filter_file` digunakan untuk memfilter file yang akan ditampilkan pada file browser. Opsi yang tersedia adalah:

- `image`: hanya menampilkan file image
- `video`: hanya menampilkan file video
- `audio`: hanya menampilkan file audio
- `archive`: hanya menampilkan file archive (.zip, .rar, .7z, dan .gz)
- `document`: hanya menampilkan dokumen (pdf dan office / open office)

Contoh penggunaan filter: `filter_file : 'image video'` atau `filter_file : 'image'`

Contoh eksekusi plugin dapat dilihat di module File Picker Manager (app/Controllers/Filepicker), Gallery Drag n Drop (app/Controllers/Gallery), Artikel (app/Controllers/Artikel), dan Stream Download (app/Controllers/Filedownload).

4. Tipe File

File Picker ini mendukung semua tipe file, baik image, video, audio, dokumen, dll. Jika berbentuk image, maka akan ditampilkan preview image tersebut, jika bukan image, maka akan ditampilkan thumbnail icon yang sesuai beserta nama file nya. Thumbnail icon ini ada di folder `public/images/filepicker_images`.

5. Upload File

Plugin File Picker ini sudah disertakan drag and drop file uploader. File upload ini dapat digunakan untuk mengupload satu atau banyak file sekaligus. Jika yang diupload adalah file image, maka system akan otomatis membuat dua thumbnail untuk image tersebut yaitu small dan medium, dengan demikian akan diperoleh tiga file, yaitu file asli, file asli_small, dan file asli_medium. Ukuran thumbnail ini secara default 250px untuk thumbnail small, dan 450px untuk thumbnail medium (disesuaikan dengan nilai yang ada di variabel `$thumbnail`). Jika yang diupload bukan image, maka file akan diupload apa adanya.

6. Dependency

Plugin ini memerlukan beberapa dependency, yaitu:

a. jQuery

`public/vendors/jquery`

b. Dropzone

`public/vendors/dropzone`

c. Sweet Alert

`public/vendors/sweetalert`

d. PHP Image Workshop yang digunakan untuk membuat thumbnail image

`app/ThirdParty/Imageworkshop`

Mengelola File

Anda dapat mengelola file yang telah diupload melalui menu File Picker > File Picker Manager. Di halaman ini, Anda dapat upload, edit, maupun delete file yang telah diupload.

Tiny MCE

Plugin File Picker ini juga dapat diintegrasikan pada Tiny MCE, WYSIWYG text editor paling populer. Contoh penggunaannya ada di module artikel (menu File Picker > Artikel). Untuk konfigurasinya perlu menyertakan file `public/themes/modern/js/artikel.js` dan file `public/themes/modern/js/filepicker-tinymce.js`

XI. Library

Admin Template Jagowebdev ini sudah disertakan beberapa library php yang siap digunakan diantaranya:

1. PHP Mailer,
2. PHPXLSX Writer,
3. Spout PHP Excel Reader dan Writer
4. MPDF

Library tersebut disimpan di folder `app/ThirdParty/`

Selain library php, Admin Template ini juga sudah menyertakan library javascript dan css, diantaranya adalah datatables, bootstrap, bootbox, sweetalert2, dll. Library tersebut disimpan di folder `public/vendors`.

XII. Penutup

Semoga panduan ini dapat membawa manfaat bagi sobat semuanya dan semoga kedepannya lebih dapat disempurnakan lagi. Jika ada kritik dan saran mohon untuk dapat menghubungi kami melalui:

- Email: support@jagowebdev.com
- Email: prawoto.hadi@gmail.com
- Whats App: 0856 136 3962

Untuk mendapatkan update informasi update aplikasi maupun content Jagowebdev silakan follow social media kami di: Twitter @jagowebdev, Facebook Jagowebdev, dan Instagram Jagowebdev

Terima kasih,

Salam

Agus Prawoto Hadi

Terakhir diperbarui: 31 Mei 2021