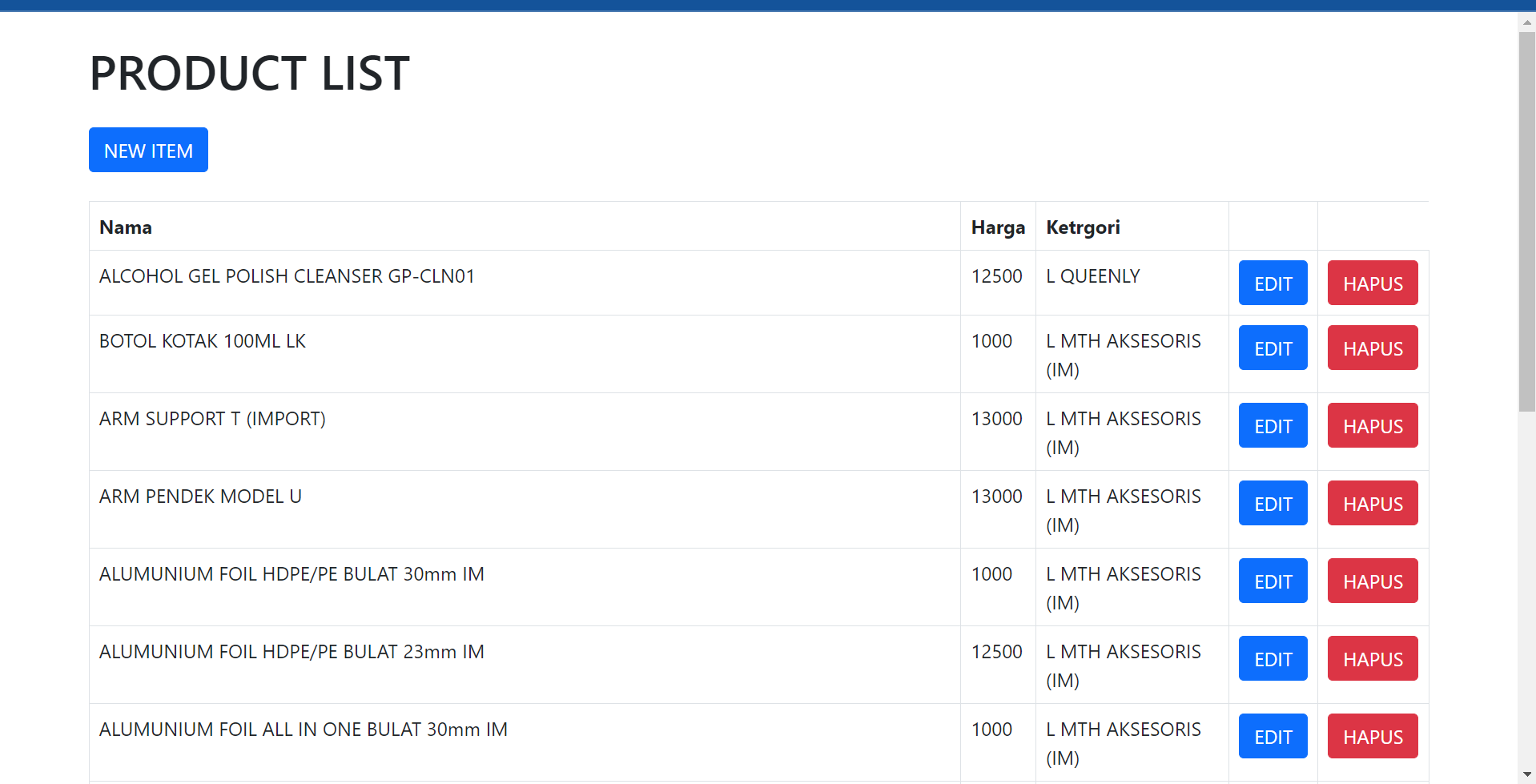
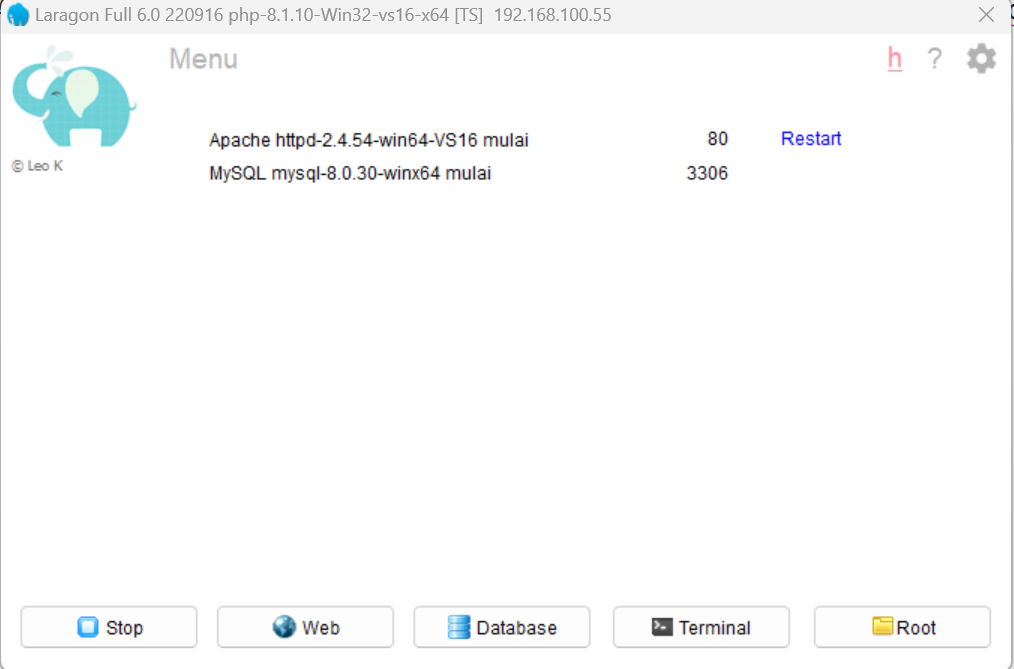
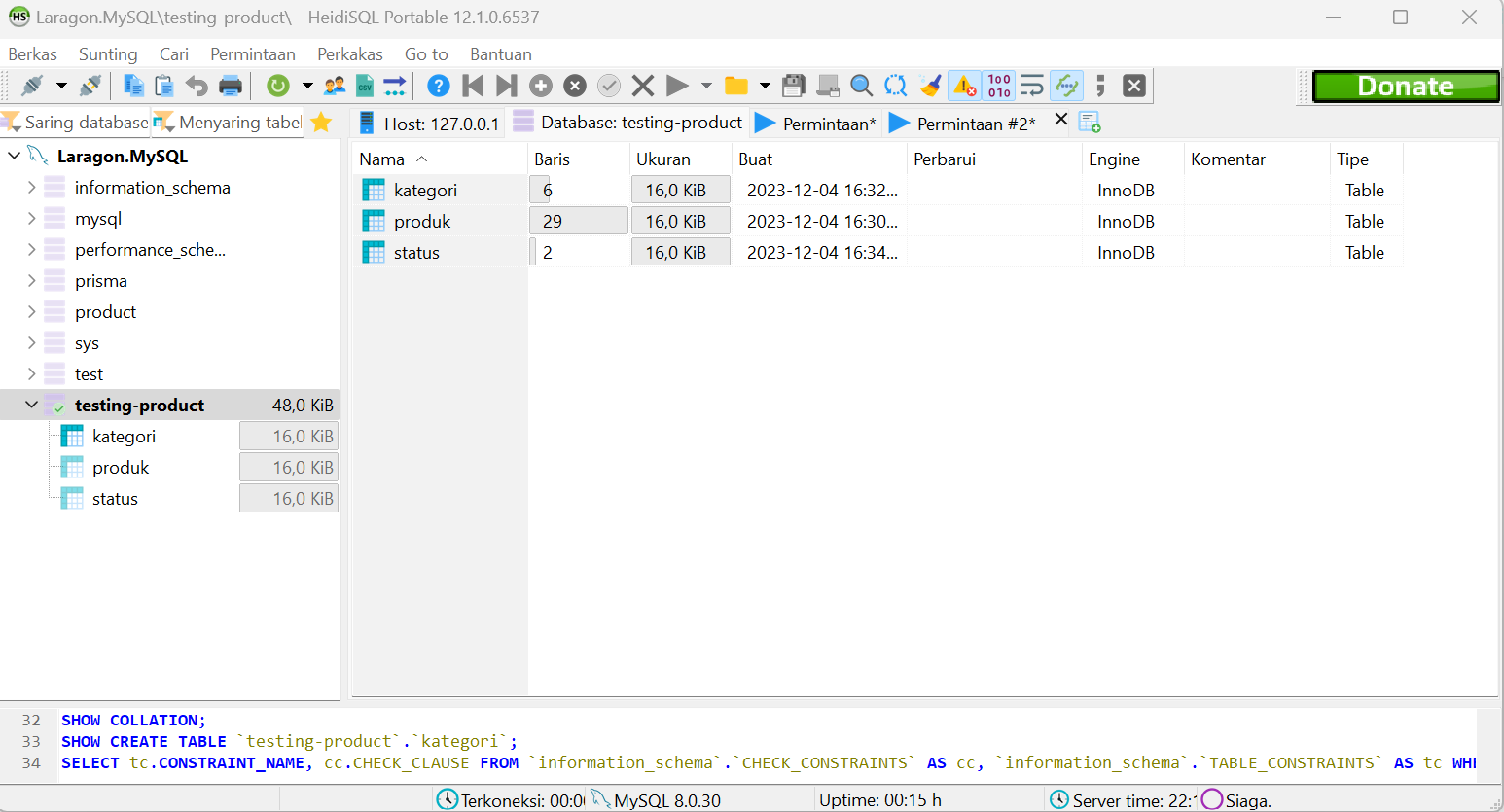
# DOKUMETASI APLIKASI DAFTAR BARANG DENGAN CODEIGNITER 3 DAN MYSQL



Aplikasi ini menggunakan Codeigniter sebagai framework dan MYSQL sebagai database nya. Lalu saya menggunakan Laragon sebagai perangkat lunak untuk lokal server.



Saya menggunakan Laragon karena saya rasa Laragon lebih mudah digunakan daripada XAMPP dan juga aplikasi tersebut cukup populer sekarang. Lalu untuk interface management database saya menggunakan heidiSql yang juga sudah ada di dalam Laragon.



Pada HeidiSQL buat database dengan nama ‘testing-product’. Pada database buat table produk, kategori, dan status.

#### Table Produk

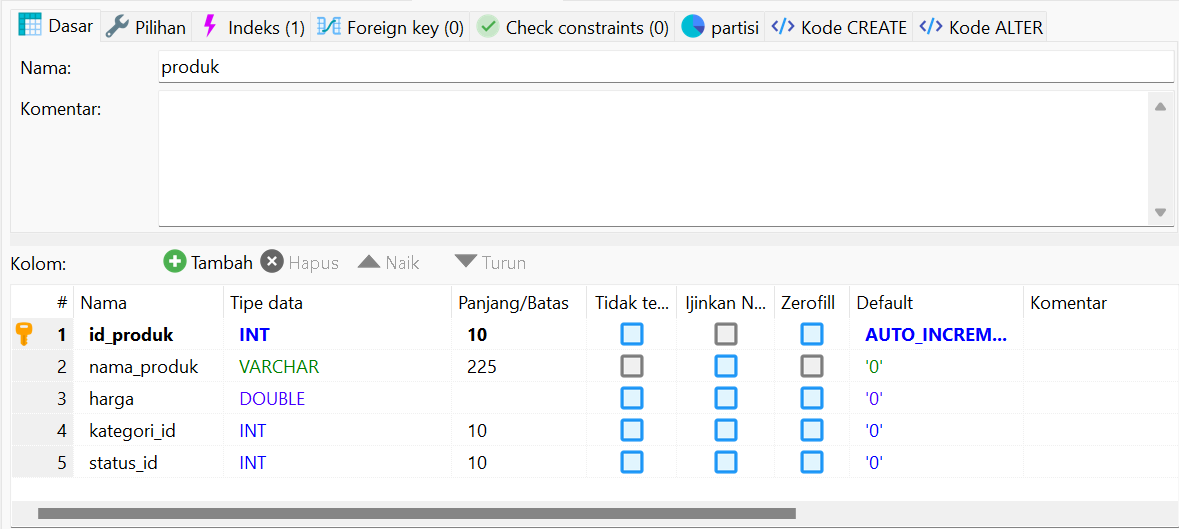


Table produk terdapat beberapa kolom:

* id\_produk : untuk menyimpan data id produk yang merupakan primary key dari data produk yang akan kita simpan dengan panjang maksimal 10. Data tersebut bertipe Interger karena kolom tersebut akan berisi data angka yang memiliki nilai default Auto Increment.
* nama\_produk : kolom ini akan menyimpan data nama produk dengan tipe data varchar dan panjang maksimal 225.
* harga : kolom ini untuk menyimpan data harga produk dengan tipe data double karena ada kemungkinan data harga akan disimpan dalam bentuk pecahan.
* kategori\_id : untuk menyimpan data id kategori yang akan digunakan untuk relasi table dengan table kategori.
* status\_id : untuk menyimpan data id status yang akan digunakan untuk relasi table dengan table status.

Pembuatan table tersebut bila ditulis dalam query:

CREATE TABLE `produk` (

`id\_produk` INT NOT NULL AUTO\_INCREMENT,

`nama\_produk` VARCHAR(225) NOT NULL DEFAULT '0',

`harga` DOUBLE NOT NULL DEFAULT 0,

`kategori\_id` INT(11) NOT NULL DEFAULT 0,

`status\_id` INT(11) NOT NULL DEFAULT 0,

PRIMARY KEY (`id\_produk`)

)

COLLATE='utf8mb4\_0900\_ai\_ci';

#### **Table Ketegori**

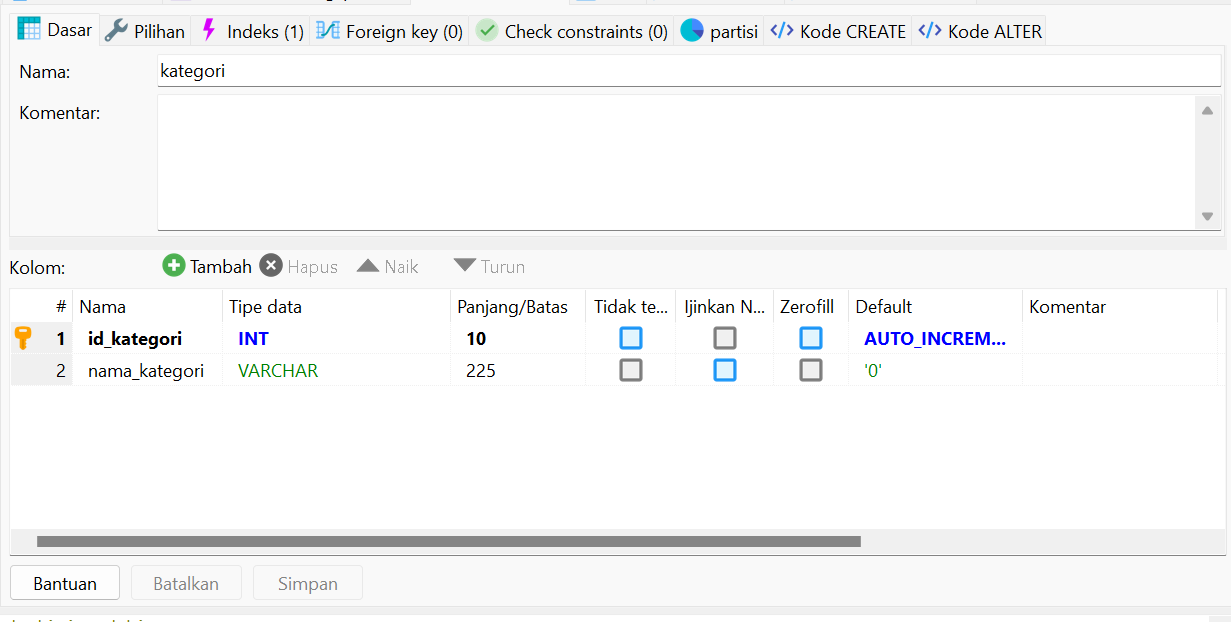


Table kategori terdapat beberapa kolom:

* id\_kategori : untuk menyimpan data id kategori yang merupakan primary key dari data produk yang akan kita simpan dengan panjang maksimal 10. Data tersebut bertipe Interger karena kolom tersebut akan berisi data angka yang memiliki nilai default Auto Increment.
  + - Nama\_kategori : kolom ini akan menyimpan data nama kategori dengan tipe data varchar dan panjang maksimal 225.

Pembuatan table tersebut bila ditulis dalam query:

CREATE TABLE `kategori` (

`id\_kategori` INT NOT NULL AUTO\_INCREMENT,

`nama\_kategori` VARCHAR(225) NOT NULL DEFAULT '0',

PRIMARY KEY (`id\_kategori`)

)

COLLATE='utf8mb4\_0900\_ai\_ci';

#### **Table Status**

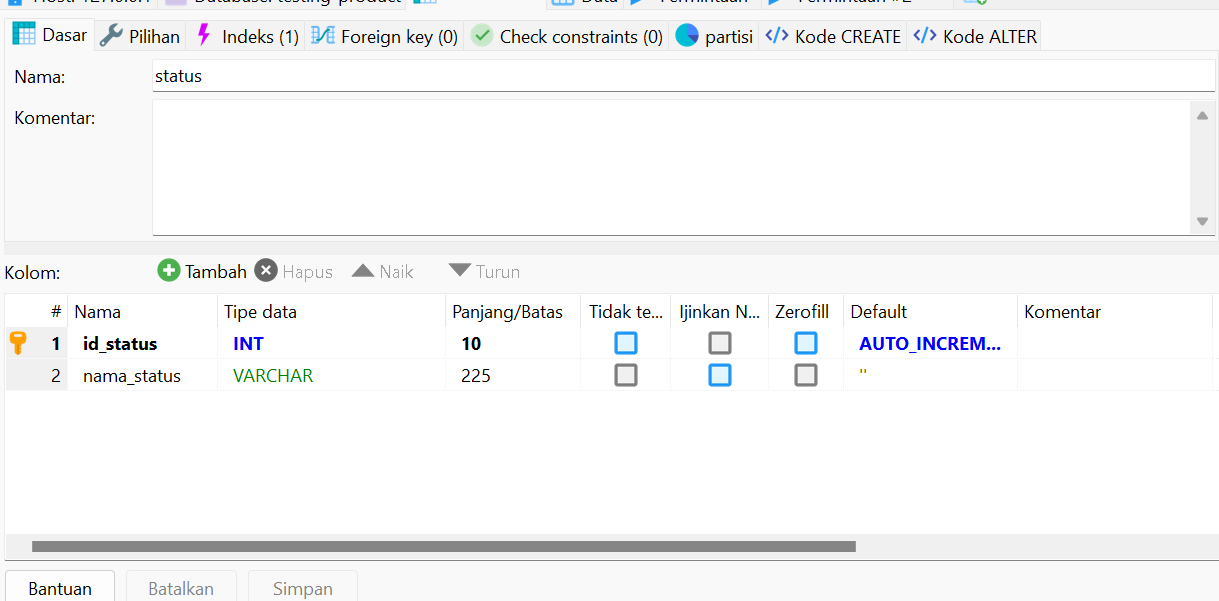


Table status terdapat beberapa kolom:

* id\_status : untuk menyimpan data id status yang merupakan primary key dari data produk yang akan kita simpan dengan panjang maksimal 10. Data tersebut bertipe Interger karena kolom tersebut akan berisi data angka yang memiliki nilai default Auto Increment.
  + - Nama\_status : kolom ini akan menyimpan data nama status dengan tipe data varchar dan panjang maksimal 225.

Pembuatan table tersebut bila ditulis dalam query:

CREATE TABLE `status` (

`id\_status` INT NOT NULL AUTO\_INCREMENT,

`nama\_status` VARCHAR(225) NOT NULL DEFAULT '',

PRIMARY KEY (`id\_status`)

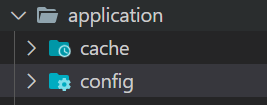
)

COLLATE='utf8mb4\_0900\_ai\_ci';

## Bagian Codeigniter

#### Setting Environment

Buka pada folder aplication/config.



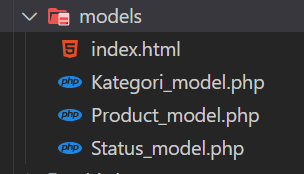
Sesuaikan beberapa hal seperti nama database, username, password pada aplication/config/database.php . Pada aplikasi kali ini username diisi root dan password dikosongi dan nama database sesuaikan dengan nama database yang telah dibuat seperti dibawah ini:



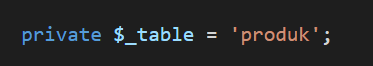
Pada aplication/config/config.php, di bagian base\_url kita isikan port <http://localhost:4000> seperti di bawah. Supaya url tersebut menjadi basic dari aplikasi yang dibuat.



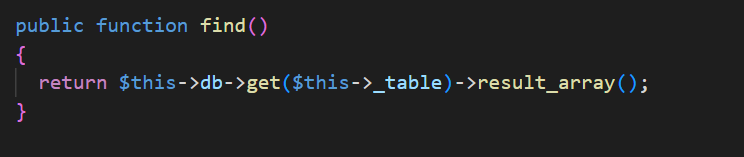
#### Setting Model

Di sini dibuat 3 model untuk masing-masing table. Table Product, Kategori dan Status untuk menghandle data yang akan dimasukkan ke database

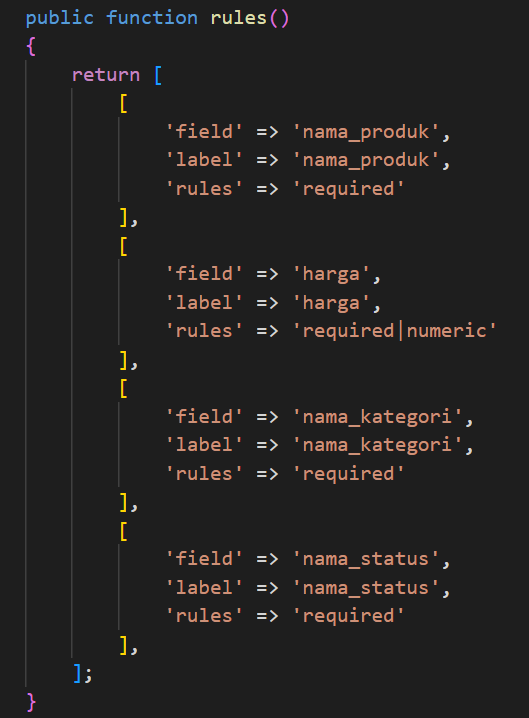
Pada masing-masing model buat variable private unutk menampung nama table.



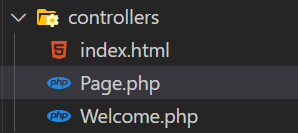
#### Method untuk memanggil data Status dan Kategori

Method find digunakan untuk memanggil semua data di kategori dan status, lalu di return dalam bentuk array agar bisa di looping

#### Method rules untuk validator di product\_model

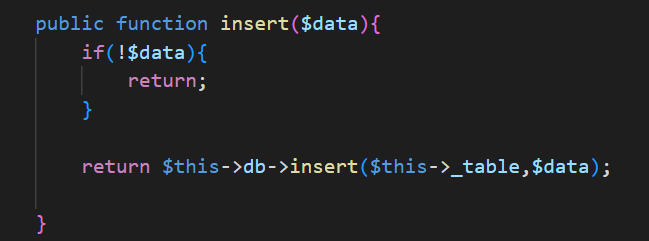
Method Rules mengembalikan nilai sebuah array yang berisi nama field, label dan rules yang akan diterapkan pada view nanti

#### Setting Controller

Buat file Page.php untuk pada controller. Page.php akan menghadle semua request di halaman view

#### Input Data

* Buat method pada class Product\_model pada folder model dengan nama insert



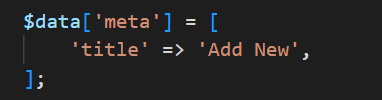
#### 

Method insert ini menampung parameter data yang akan dikirim dari controller. Dalam method tersebut terdapat logika if dimana jika parameter kosong maka langsung return tanpa ada nilai yang dikembalikan. Tapi jika ada maka return masukkan data dengan menerima parameter nama db dan $data

* Buat method pada class Page di folder controller dengan nama create



- Dalam method create buat variable data yang pertama menampung array meta untuk title



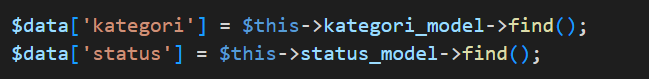
- Lalu Load Library Validator



- Load Model yang akan digunakan pada method create



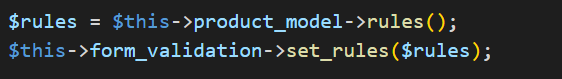
- Panggil method find di dalam kategori\_model dan status\_model untuk untuk mengambil data kategori dan status lalu masukkan dalam array variable $data



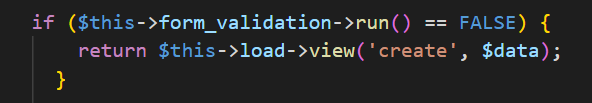
- Lalu buat logika dimana jika input method sama dengan post maka lakukan baris kode didalamnya



- Di dalam logika if terdapat baris kode untuk memvalidasi. Pertama kita buat variable $rules yang memanggil method rules didalam product model lalu set\_rules dengan parameter variable $rules



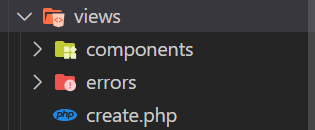
- Setelah itu terdapat logika if, dimana jika validation sama dengan FALSE maka akan load view create;



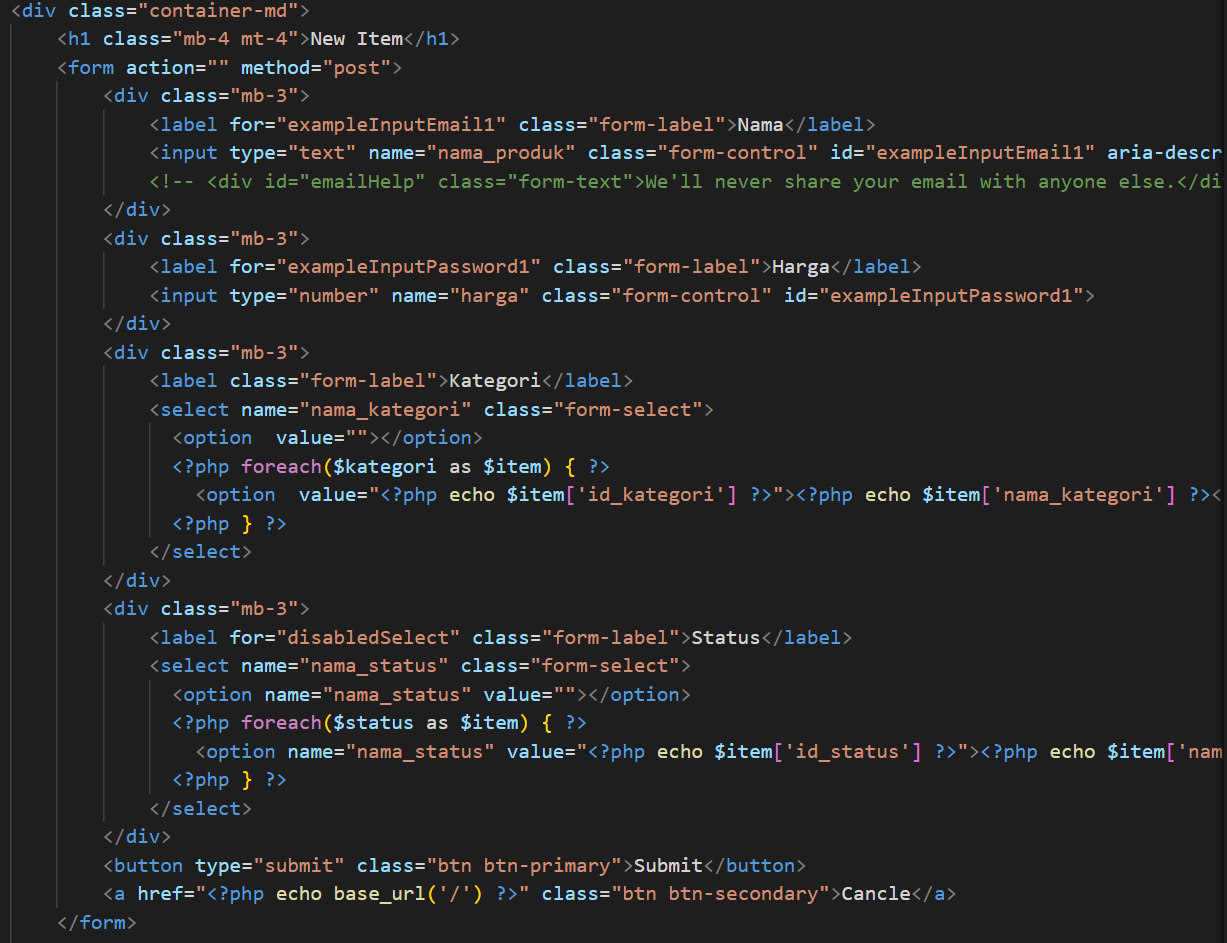
- Lalu buat variable $feedback dimana memasukkan array data-data yang disubmit melalui form di view lalu buat variable $feedback\_save yang berisikan method insert dalam product model dengan parameter $feedback setelah itu redirect ke halaman home



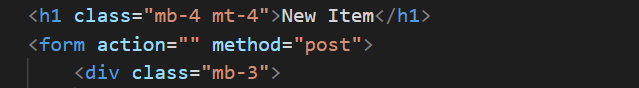
* Buat views untuk membuat baru item baru dengan nama create.php



- Isi dari create.php sebagai berikut

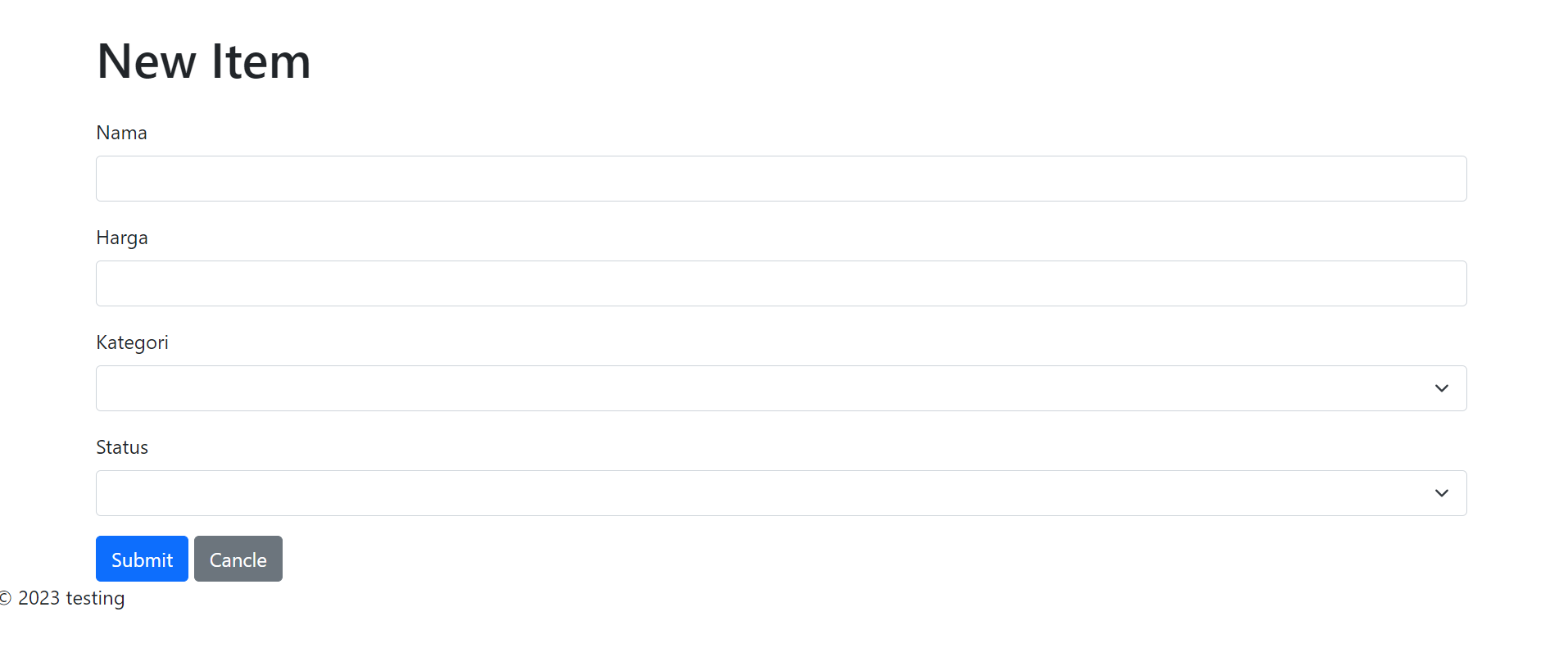


- Form dengan method post dan lalu looping data untuk select option.

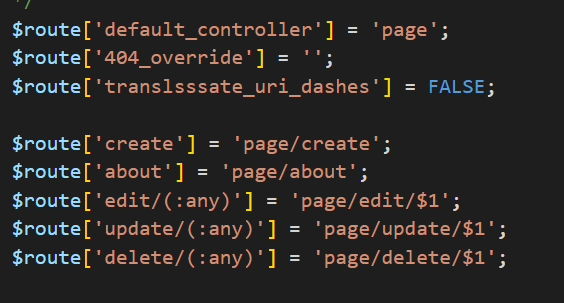




- Beginilah tampilan form input



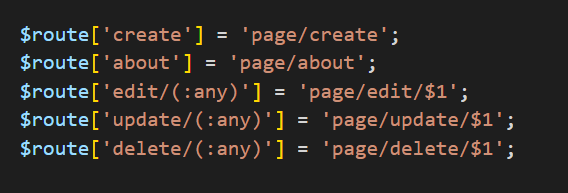
- Setting pada routes seperti dibawah ini



Default\_controller diisi dengan page tandanya ketika program dijalankan akan secara otomatis memanggil method index didalam controller page yang mana pada index controller kita panggil home.php



Lalu sesuaikan route seperti ini... $route[‘create’] menandakan nama route yang akan dipakai, sementara itu ’page/create’ merupakan methode yang akan dipanggil

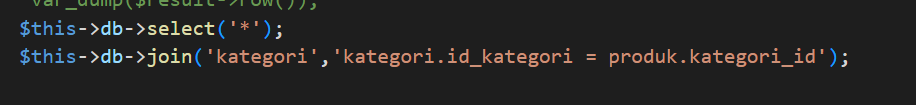


#### Menampilkan Data

* Buat method find di product\_model untuk menampilkan data

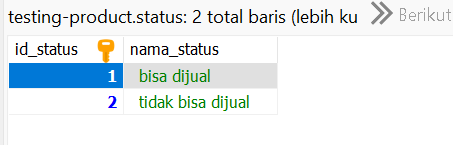


- method find kita joinkan data produk dengan kategori , melalui kolom kategori\_id di table produk dan id\_kategori unutuk mendapat nama kategori



- lalu return dengan menambahkan where status id di table produk samadengan 1 dimana 1 adalah id untuk item yang bisa dijual

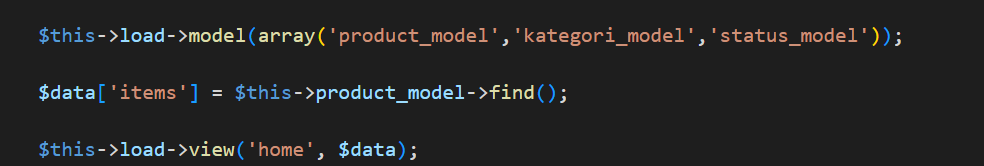




* Buat tambahkan beberapa baris kode di method index di controller page.php



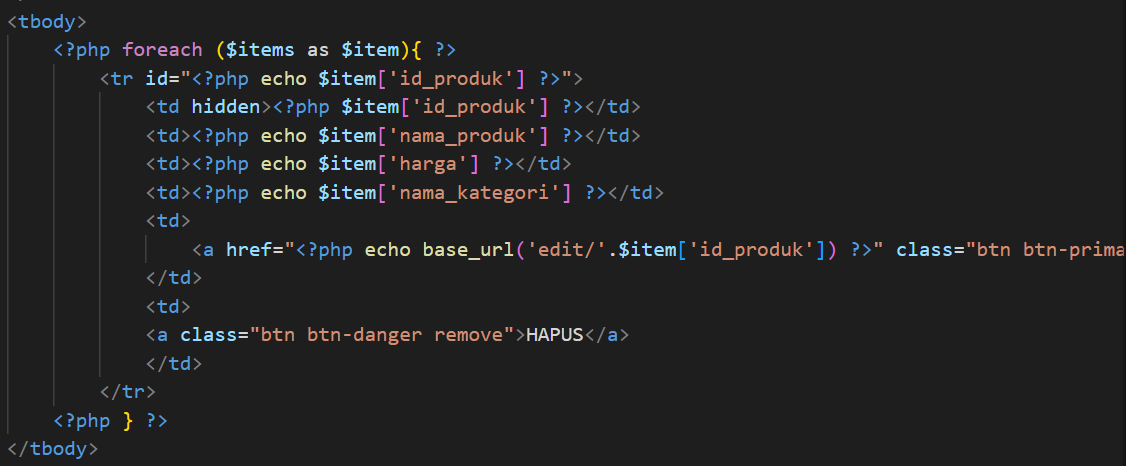
- load model yang akan dipakai lalu panggil find method pada product model masukkan kedalam variable array $data dan load view home bersama denga $data.



* Buat Home.php pada views untuk menampilkan data item.



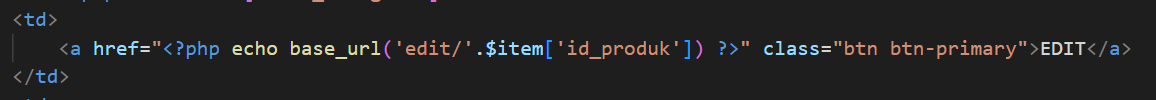
-Looping data yang dikirim ari controller dan tambahkan tombol hapus dan edit



- Dan juga tambahkan tombol new item yang mengarahkan ke halaman create



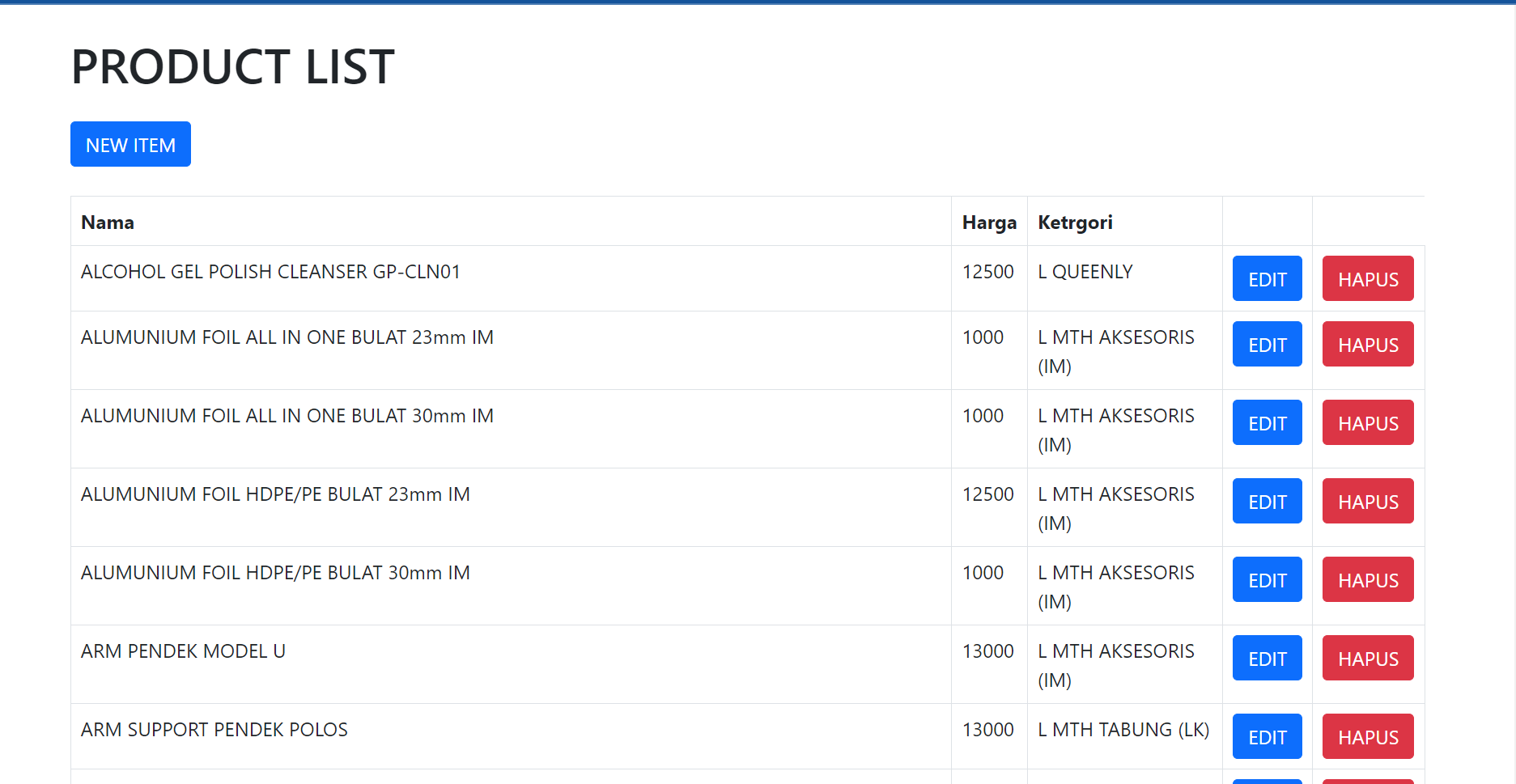
- Tombol edit panggil route dan lempar parameter berupa id item yang akan di edit



- sesuaikan routes seperti ini, dimana url edit akan ditambahkan parameter untuk item yg dituju. Route seperti ini berlaku pula untuk update dan hapus item



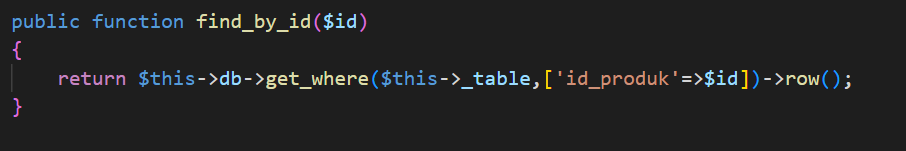
- Begini tampilan halaman awal untuk menampilkan data



#### Mengedit Data

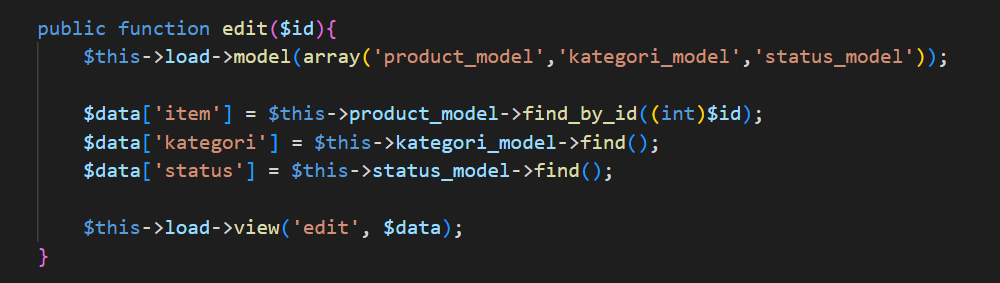
* Buat method find\_by\_id di product\_model untuk mendapatkan data

Method ini menerima parameter id dan mengembalikan data yang telah di filter dengan get\_where sesuai id parameter yg diterima



* Buat method edit di controller

- metode edit ini menerima parameter id. Memuat model dan menampung data-data pada array $data lalu memuat halaman edit dan melemparkan variable $data ebagai data yang akan dimuat



* Buat edit.php pada views untuk mengedit item.

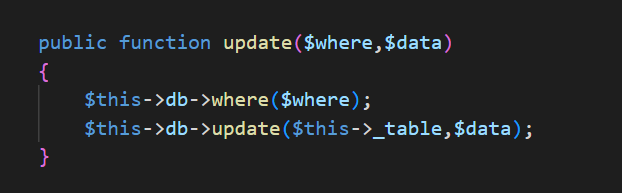
- tampilan edit kirang lebih sama seperti create tetapi edit.php melempar id saat submit untuk update item. Jadi tampilan form actionnya seperti ini



#### Menghapus Data

* Buat method update di product\_model

- Methode update menerima 2 parameter $where yang berisi array untuk find data dan $data array untuk mengupdate isi data

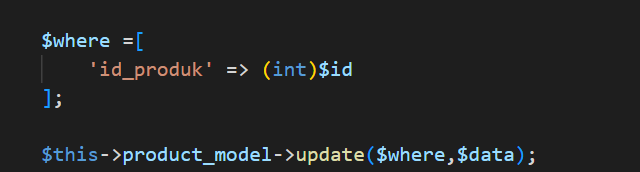


* Buat method update di page controller

- Method update kurang lebih sama dengan method insert dengan sedikit perbedaan



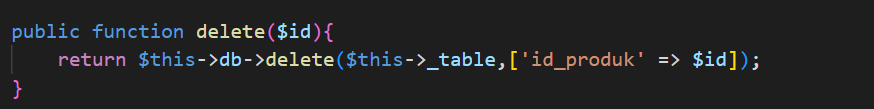
- kita memasukkan array where dan panggil method update pada product\_model dengan 2 parameter yg kita berikan



#### Menghapus Data

* Buat method delete di product\_model

- methode delete yang menerima 1 parameter id.



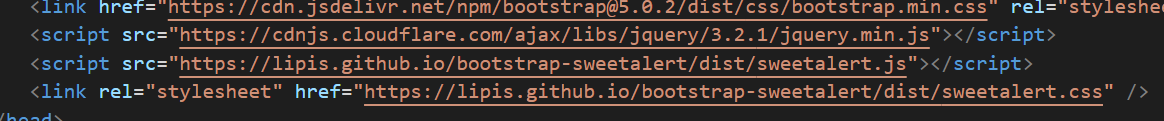
* Buat method delete di page controller

- dalam metode delete ada if dimana jika $id kosong maka munculkan 404. lalu coba jalankan product\_model delete dengan parameter $id jika sukses munculkan echo ‘delete successfully’.



* Konfirmasi delete pada page home

- disini kita menggunakan library Jquery dan sweetalert

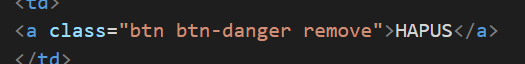


- tampilan Jquery



- Jquery akan mengambil button yang diklik dengan class remove



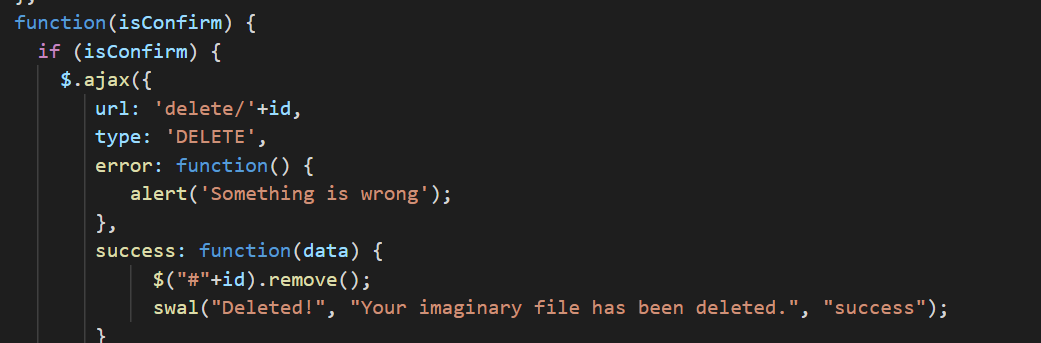


- Setelah itu buatkan variable untuk menangkap yang dipanggil melalui codeigniter

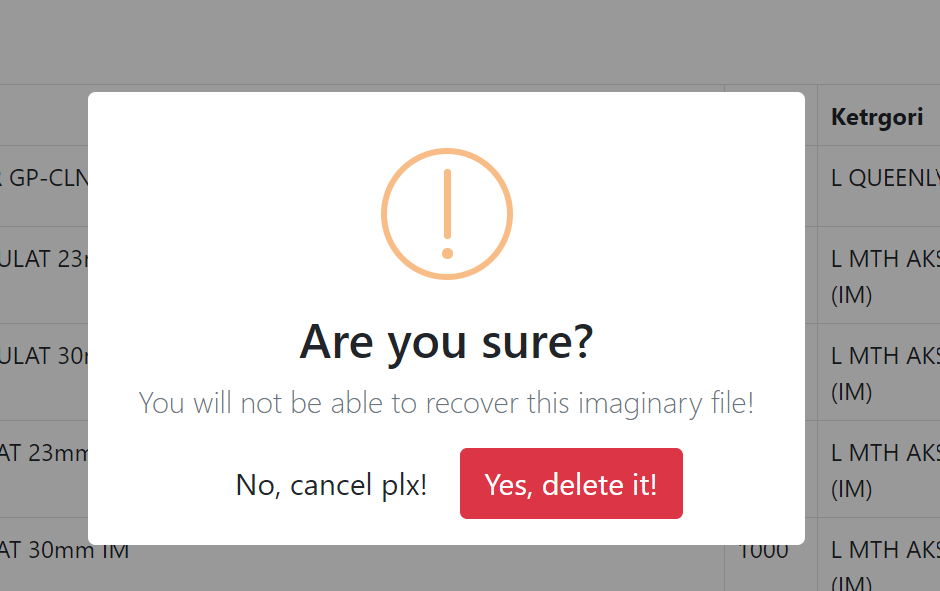




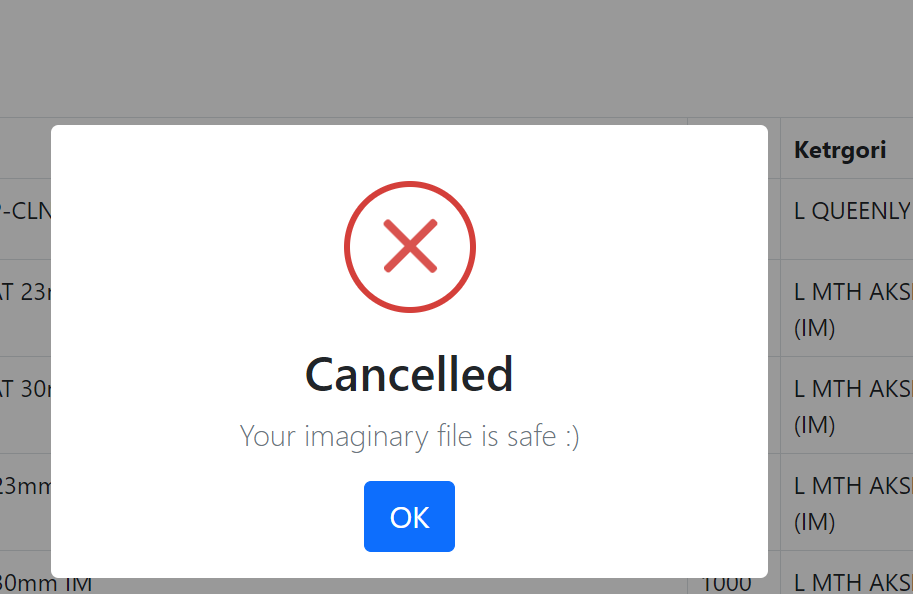
- jika dikonfimasi maka akan menjalankan url method delete + parameter



- tampilan konfirmasi



- tampilan jika tekan No

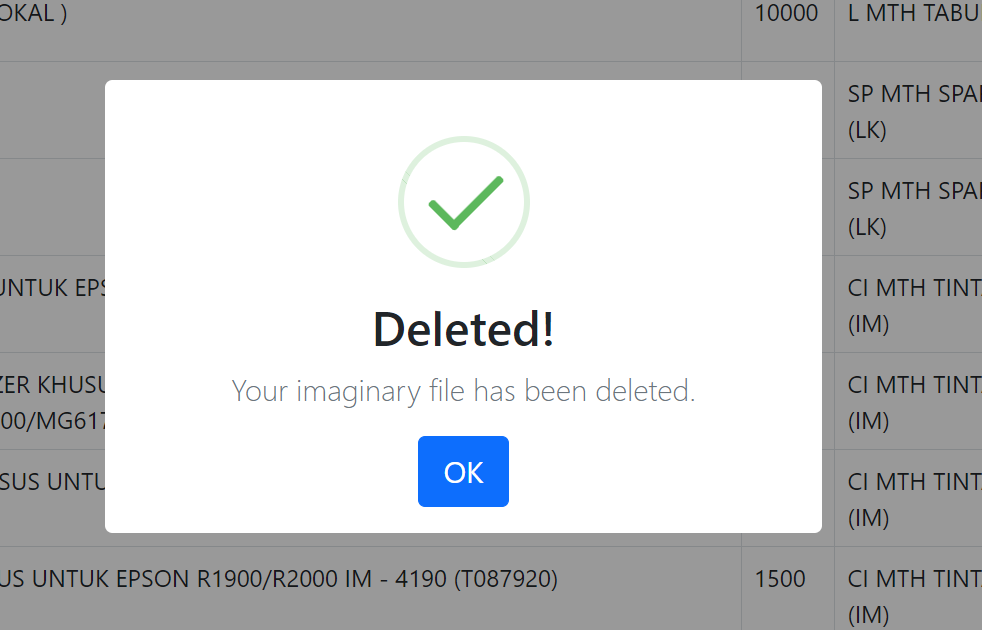


- contoh item yang akan dihapus





- jika sukses terhapus



- Pada database sudah tidak ada

