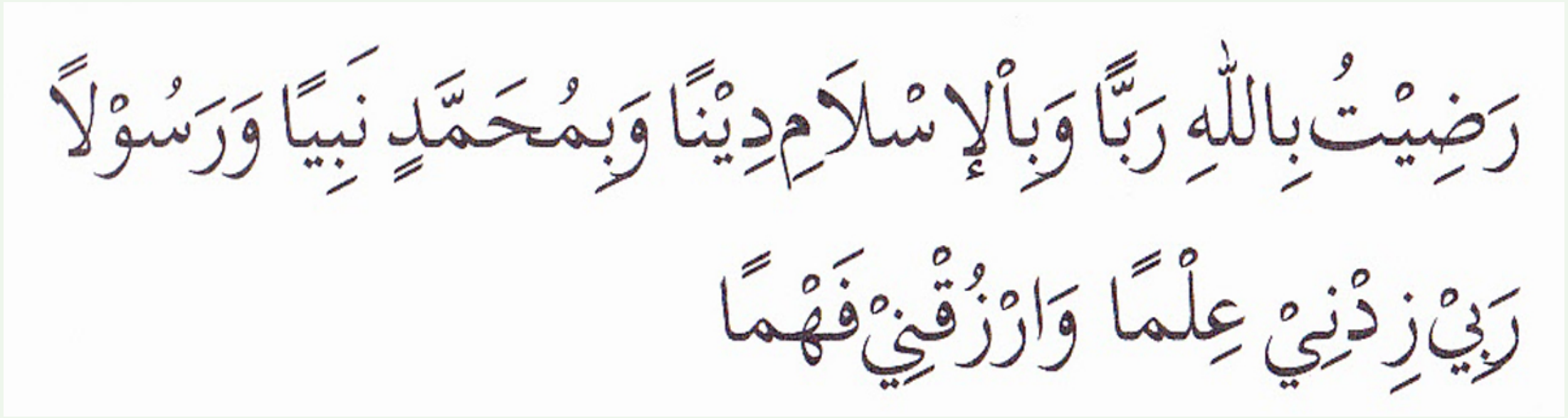
Persiapan

Sudah terinstall

1. PHP versi 7.2/XAMPP/WAMP
2. Mysql/Postgresql
3. Text Editor (Visual Studio Code)

DOA BELAJAR



"Kami ridho Allah SWT sebagai Tuhanku, Islam sebagai agamaku, dan Nabi Muhammad sebagai Nabi dan Rasul, Ya Allah, tambahkanlah kepadaku ilmu dan berikanlah aku kefahaman"

Universitas 'Aisyiyah Yogyakarta

LARAVEL IN ACTION

Sadr Lufti Mufreni, S.Kom, M.Sc.

2019

Bahan Kajian

Outlines

Pendahuluan

Instalasi

Otentifikasi

Model

Migrasi Database

Router

Controller

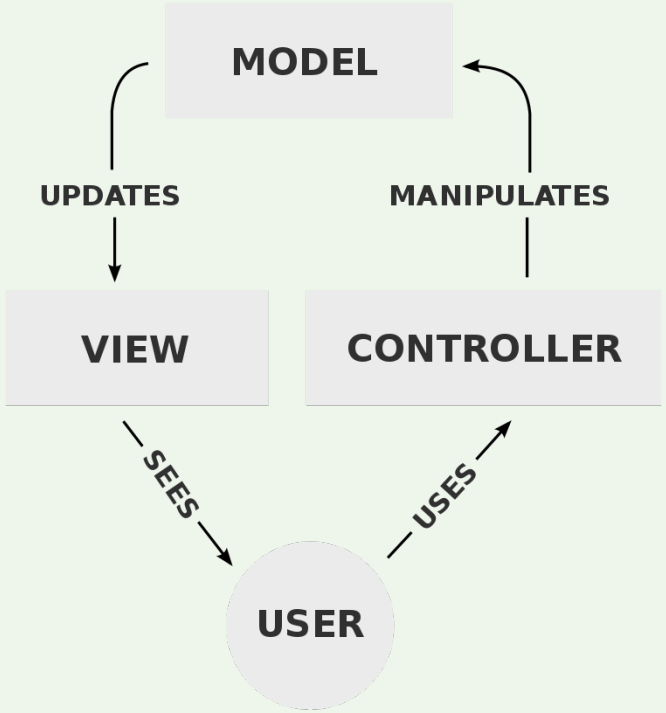
View

Pendahuluan

Laravel adalah web framework untuk PHP

Framework : kumpulan library yang dijadikan satu

Ide awal dari Ruby on Rails dengan konsep MVC (Model-View-Controller)



Pendahuluan

Framework lainnya:

1. CakePHP
2. CodeIgniter: paling populer 2010

Mempunyai komunitas yang kuat

Kelemahan: adaptasi terhadap teknologi baru kurang, versi terakhir belum mendukung PHP 7

Pendahuluan

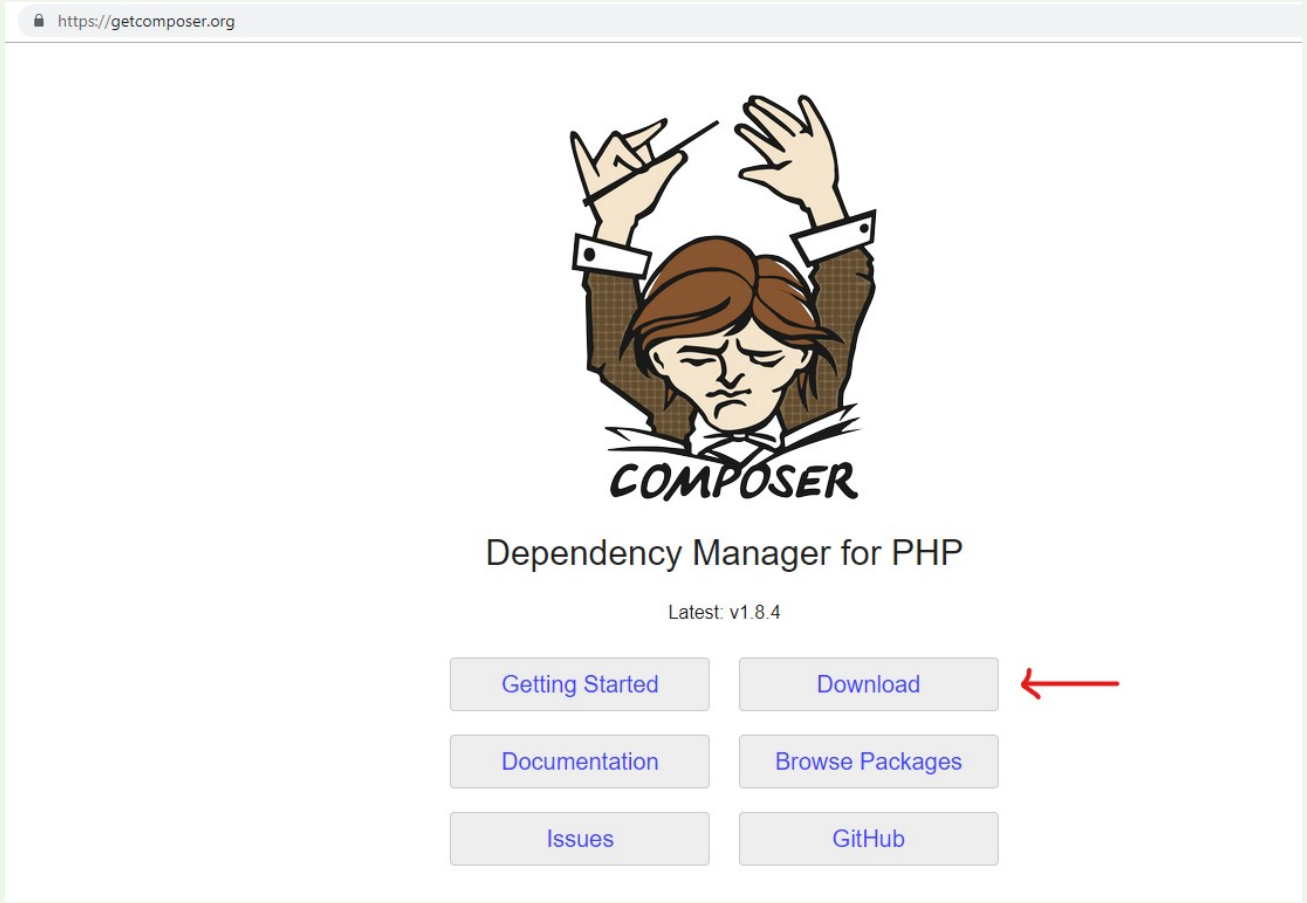
Laravel : open source, mendukung PHP 7

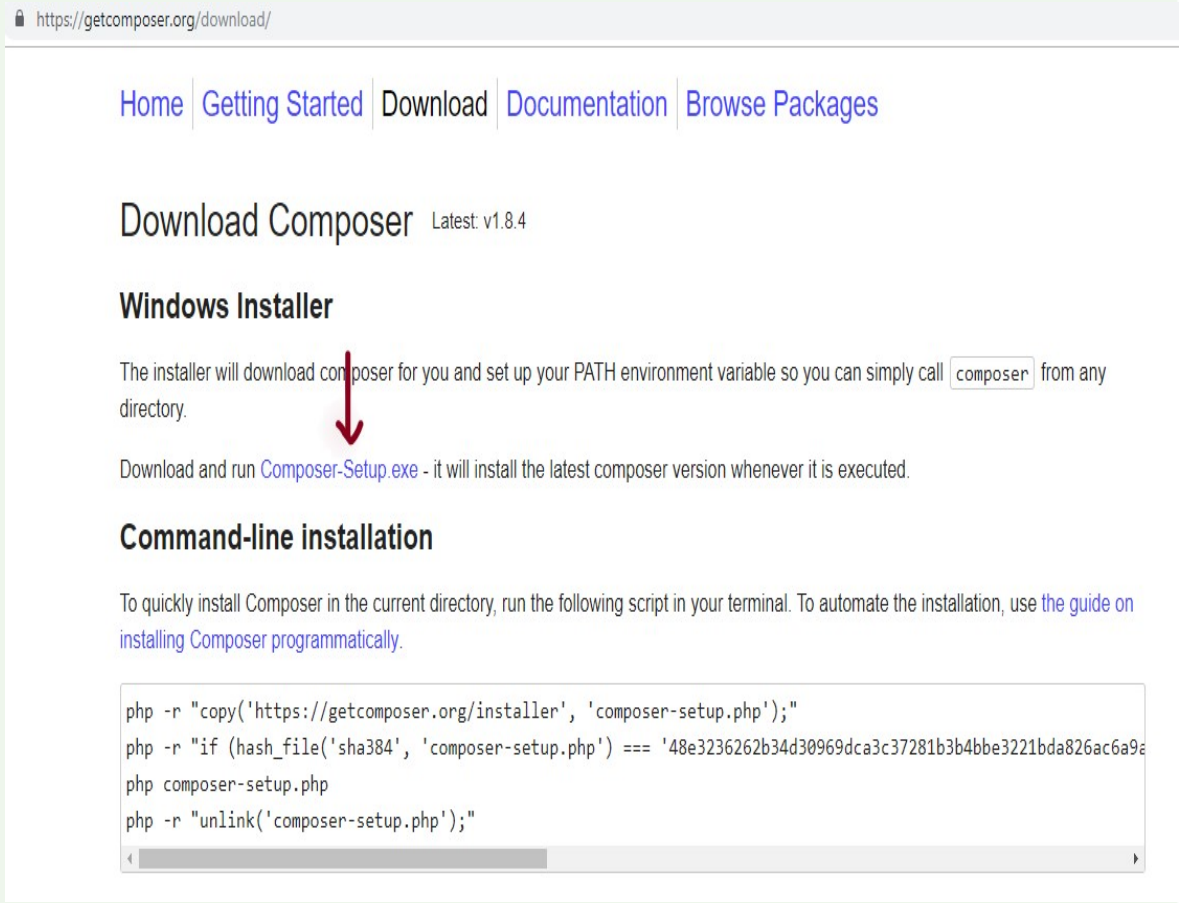
Fitur

1. Composer : mengelola dependecy
2. Eloquent ORM : model yang terhubung ke database, database independent
3. Query Builder
4. Restful controller : mendukung HTTP verb (POST, PUT, DELETE, GET)
5. Blade template : untuk halaman web yang dinamis
6. Artisan CLI : command line untuk manipulasi projek

Instalasi

1. Sebelum menginstall Laravel, install dulu Composer





Instalasi

1. Sebelum menginstall Laravel, install dulu Composer

Eksekusi

composer

Jangan lupa update PATH

2. Install Laravel dengan mengeksekusi perintah ini di command line

composer global require laravel/installer

Jangan lupa update PATH

3. Masuk ke direktori htdocs

Instalasi

4. Buat template dengan mengeksekusi perintah ini di command line

laravel new <name projek>

atau

laravel new <name projek> --auth

--auth digunakan untuk membuat otentifikasi secara otomatis

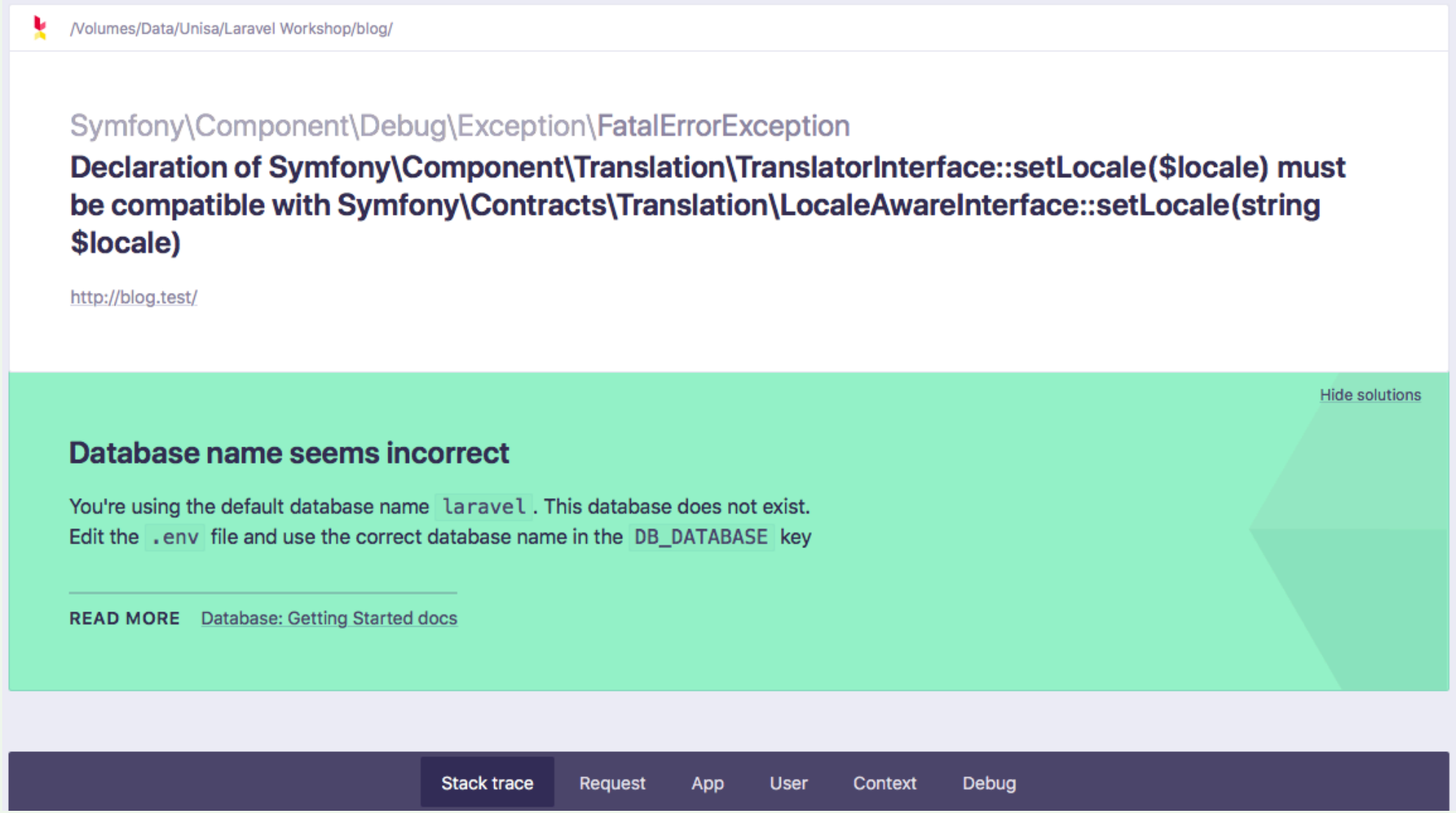
5. Masuk ke direktori <nama projek> dan eksekusi

php artisan serve

6. Buka browser dan kunjungi

http://localhost/<nama projek>

Instalasi


/Volumes/Data/Unisa/Laravel Workshop/blog/
Symfony\Component\Debug\Exception\FatalErrorException
Declaration of Symfony\Component\Translation\TranslatorInterface::setLocale($locale) must be compatible with Symfony\Contracts\Translation\LocaleAwareInterface::setLocale(string $locale)
http://blog.test/
Hide solutions
Database name seems incorrect
You're using the default database name laravel. This database does not exist.
Edit the .env file and use the correct database name in the DB_DATABASE key
READ MORE
Database: Getting Started docs
Stack trace
Request
App
User
Context
Debug

Instalasi

7. Buka file .env di root direktori projek baru
8. Ubah DB_CONNECTION menjadi pgsql untuk database Postgresql dan port ke 5432
9. Bila database belum ada silahkan dibuat databasenya

Contoh di postgresql, buka admin sql

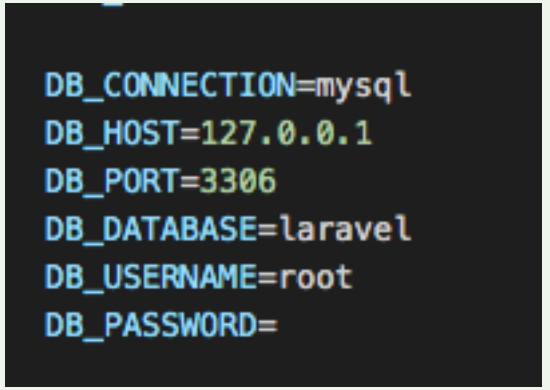
create database laravel

create user myblog with encrypted password ‘myblog’

grant all privileges on database laravel to myblog ;

10. Buka terminal/command prompt, masuk ke direktori projek baru, eksekusi di terminal/command promp

php artisan migrate



Instalasi

Laravel

DOCS LARACASTS NEWS BLOG NOVA FORGE VAPOR GITHUB

Instalasi

Setelah berhasil membuat projek, Laravel membuat direktori berikut

1. **app** : kode untuk controller/model
2. boostrap : boostrap/cache dibuat writable
3. config : berisi konfigurasi dari projek: app.php, database.php
4. database : file yang diperlukan untuk migrasi
5. **public** : menyimpan static file/images/javascript/css
6. **resources** : kode untuk view
7. **routes** : berisi route projek: web.php, api.php
8. storage : writable permission, compiled stuff
9. tests : testing framework for PHPUnit
10. vendor : composer dependencies

Otentifikasi

Untuk membuat otentifikasi Laravel menyediakan fitur otomatisnya.

Caranya: install package laravel/ui dengan cara mengeksekusi di terminal

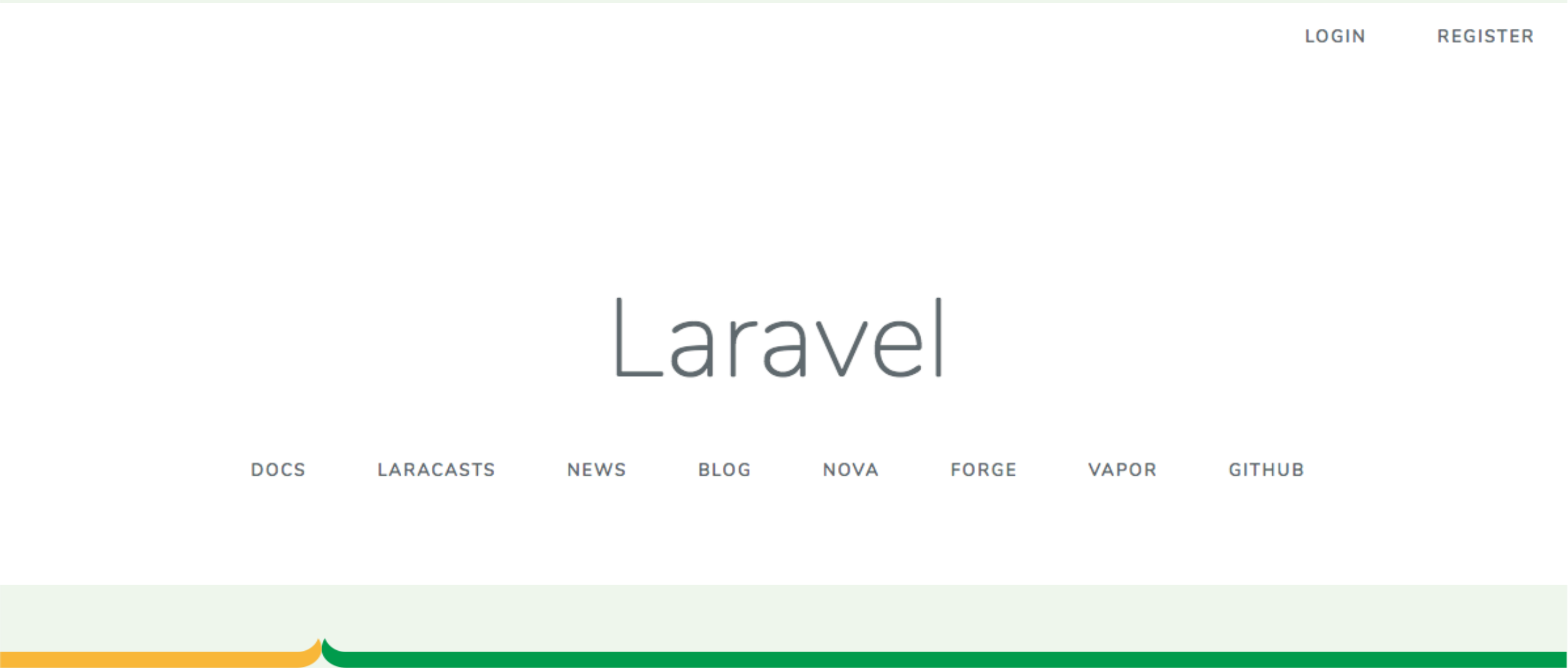
composer require laravel/ui --dev

php artisan ui bootstrap –auth

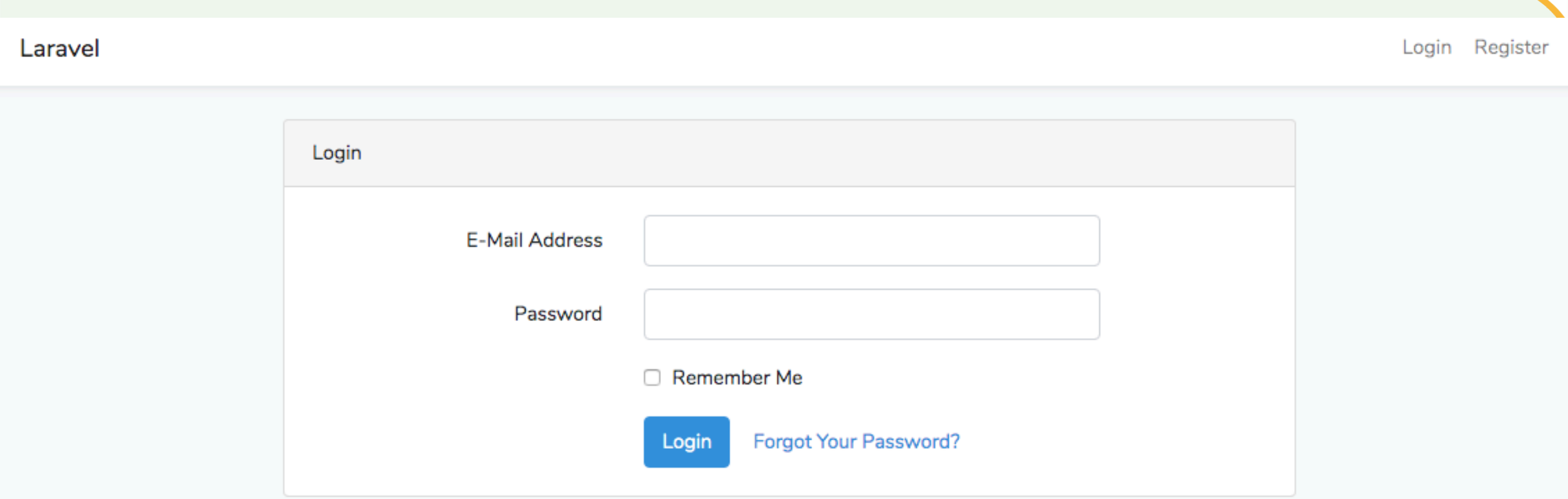
Ubah file .env APP_URL sesuai dengan nama webnya

```
APP_DEBUG=true
APP_URL=http://blog.test
```

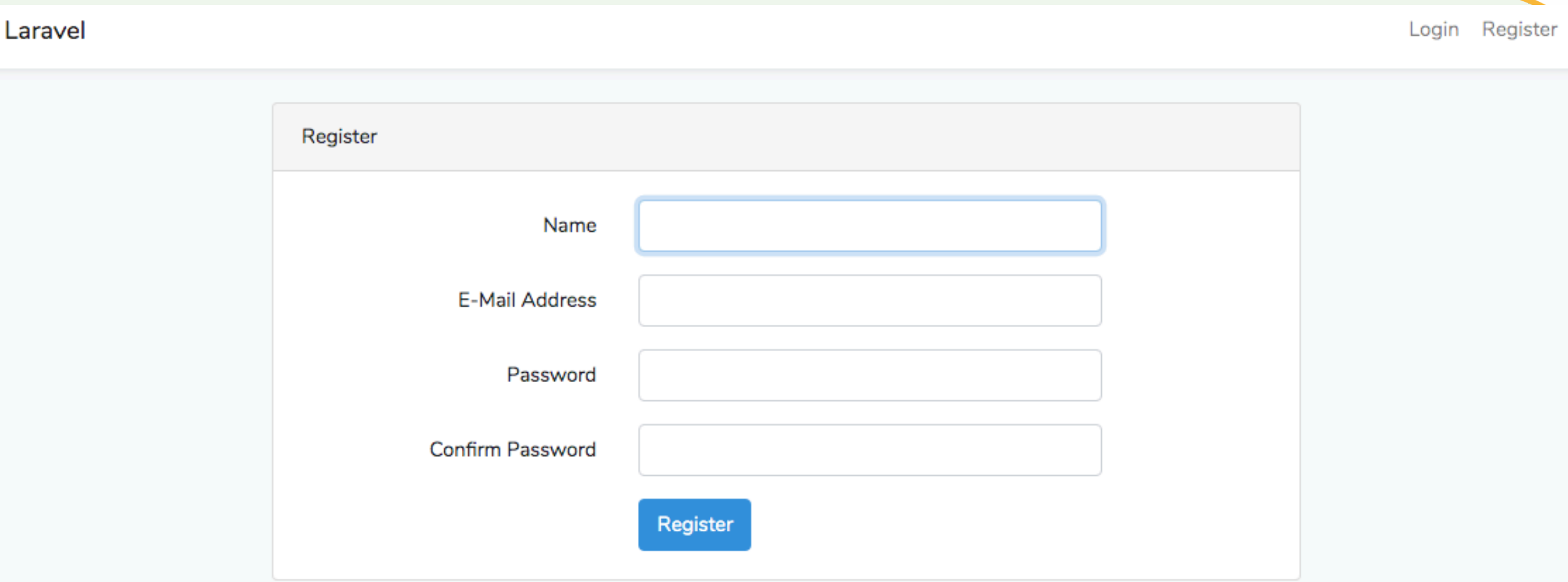
Otentifikasi



Otentifikasi



Otentifikasi



Membutuhkan setting mail server

Otentifikasi

Di .env, ubah mail settings

```
MAIL_DRIVER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=null
MAIL_PASSWORD=null
MAIL_ENCRYPTION=null
```

Otentifikasi

Untuk membuat verifikasi email, ketika mendaftar

Caranya:

1. Buka file User.php di direktori app
2. Untuk menambahkan fitur verifkasi email, User implementasi Illuminate\Contracts\Auth\MustVerifyEmail contract

```php
use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;

class User extends Authenticatable implements MustVerifyEmail
{
    use Notifiable;
```

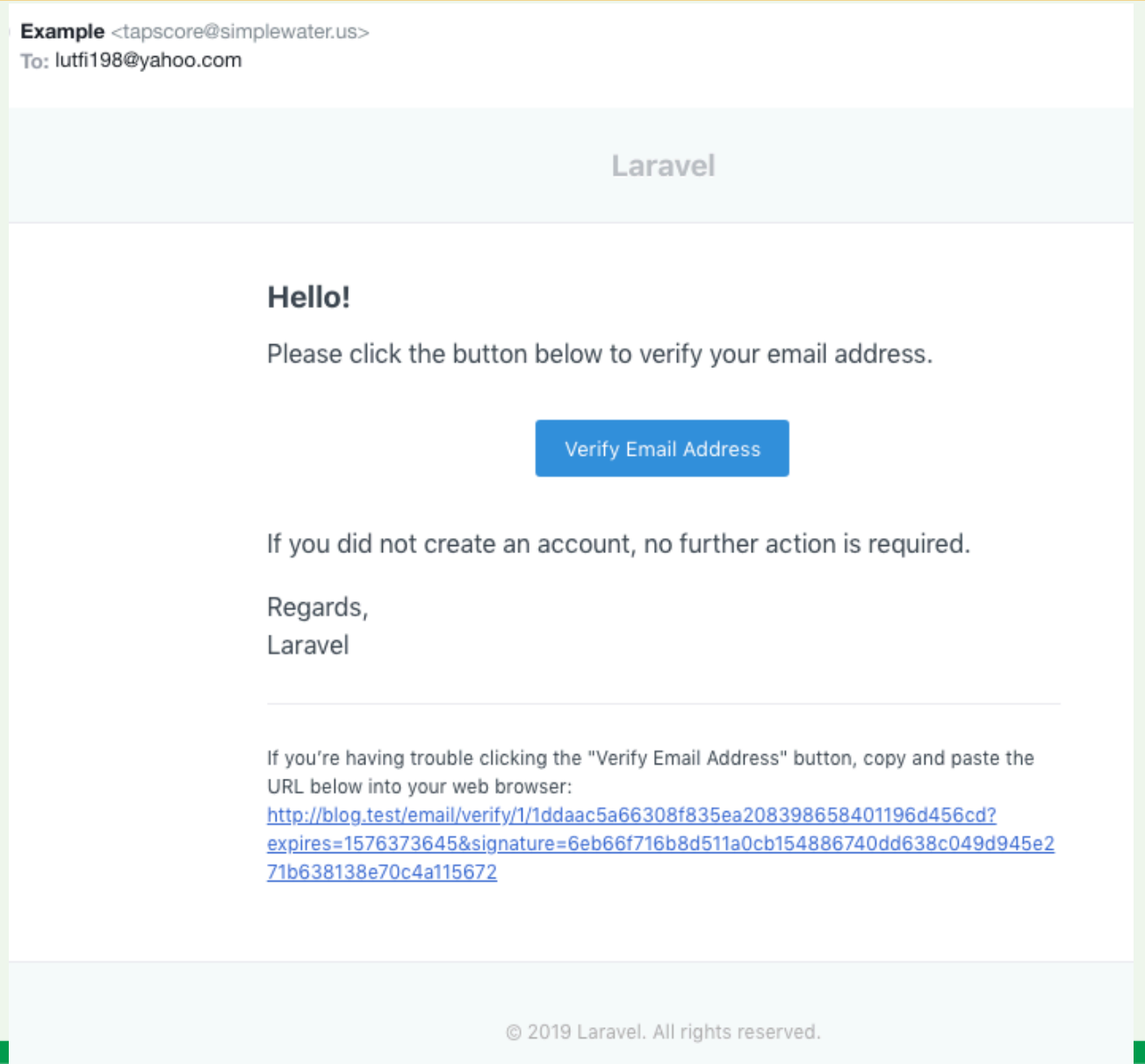
Otentifikasi

3. Buka file web.php di direktori routes
4. Tambahkan ['verify' => true] di Auth::routes
5. Tambahkan ->middleware(‘verified’) pada route yang diwajibkan user untuk verifikasi email terlebih dahulu

```php
Route::get('/', function () {
    return view('welcome');
});

Auth::routes(['verify' => true]);

Route::get('/home', 'HomeController@index')->name('home')->middleware('verified');
```

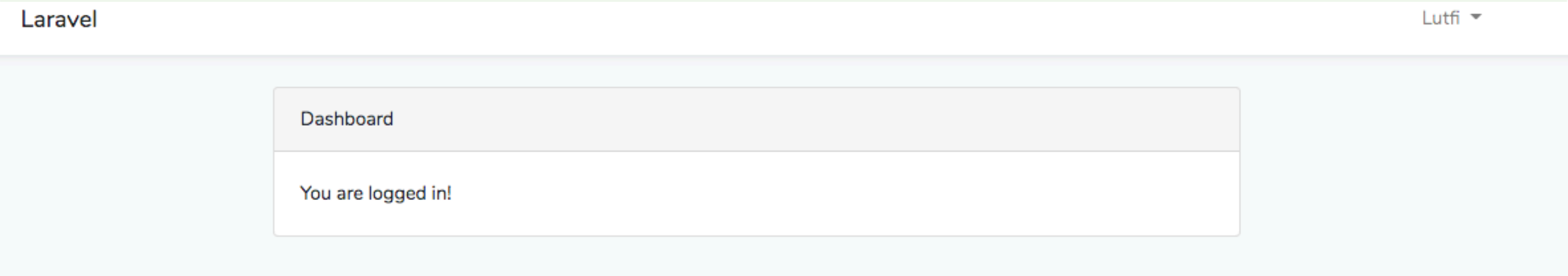
Otentifikasi

Hasil Register dengan Verifikasi Email

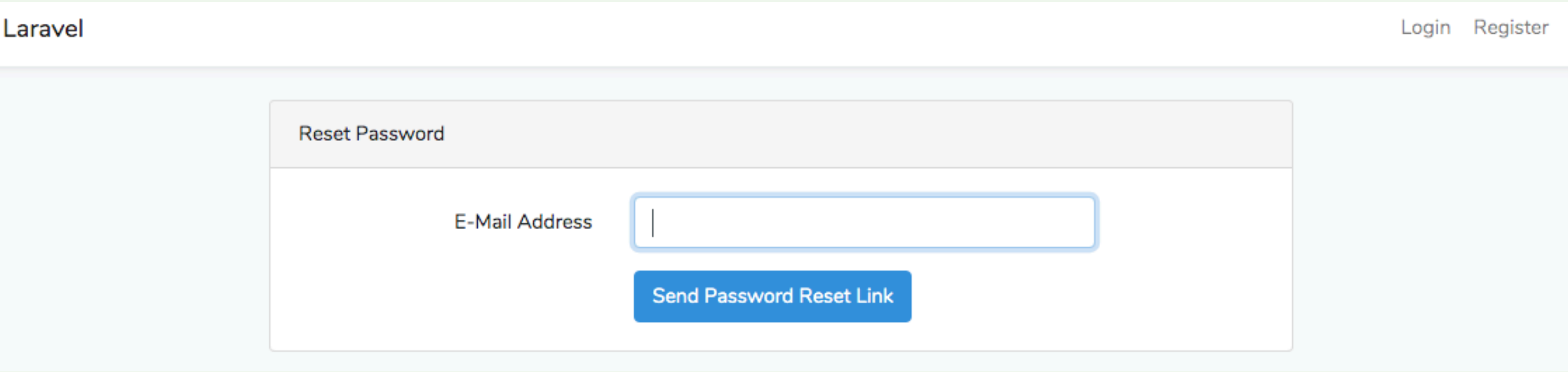


Otentifikasi

Hasil Register dengan Verifikasi Email



Otentifikasi



Otentifikasi

Laravel

Hello!

You are receiving this email because we received a password reset request for your account.

Reset Password

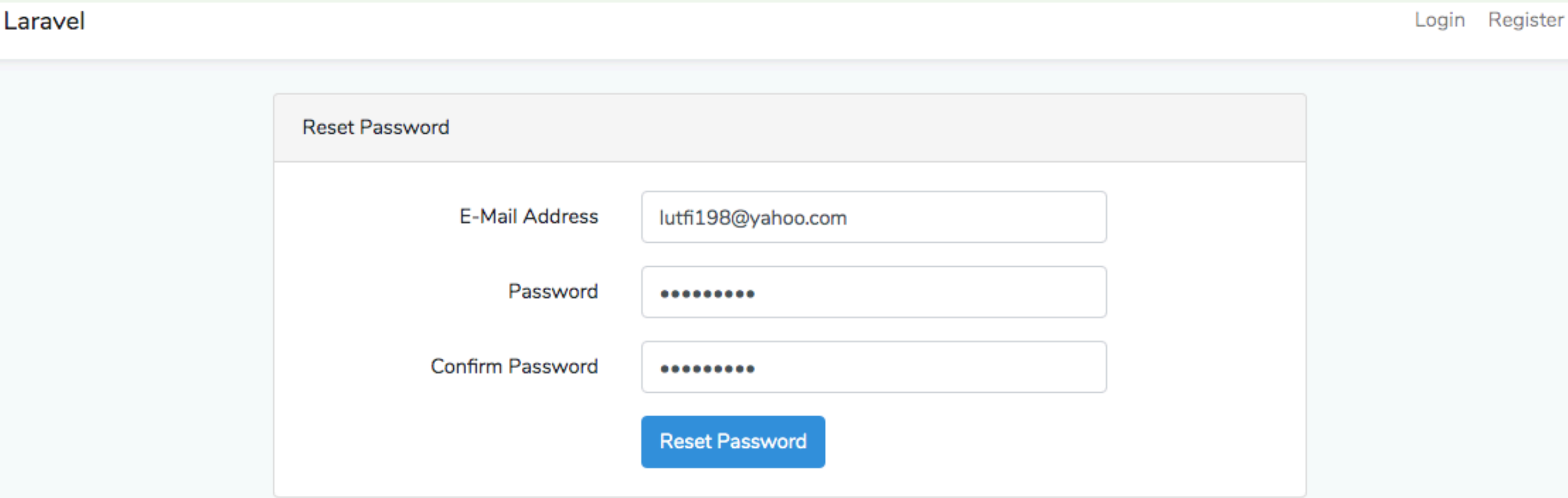
This password reset link will expire in 60 minutes.

If you did not request a password reset, no further action is required.

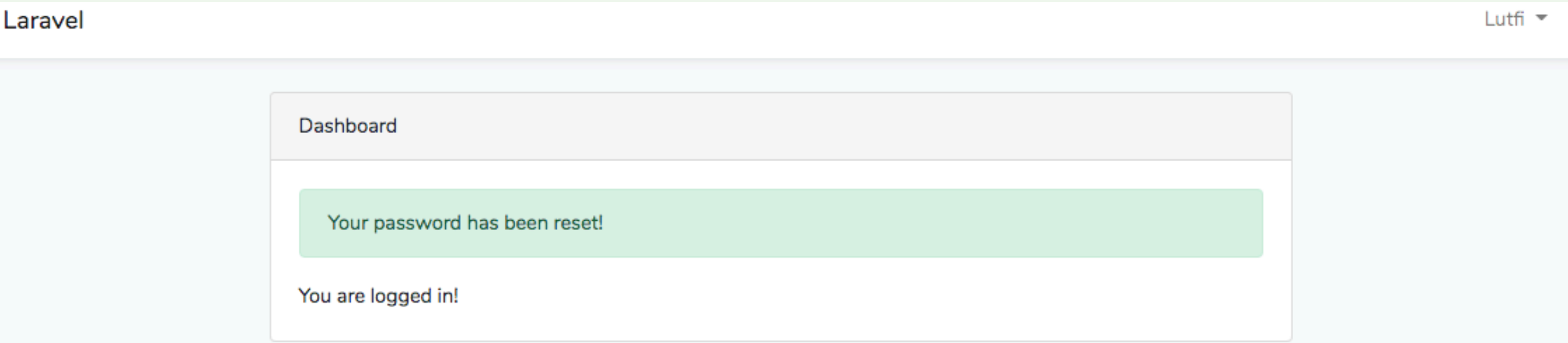
Regards,
Laravel

If you're having trouble clicking the "Reset Password" button, copy and paste the URL below into your web browser:
http://localhost/password/reset/4eef145e292cd456e8cb25efc6bc841420d85ad5c50a72c65ccfd5df1aaebe92?email=lutfi198%40yahoo.com

Otentifikasi



Otentifikasi



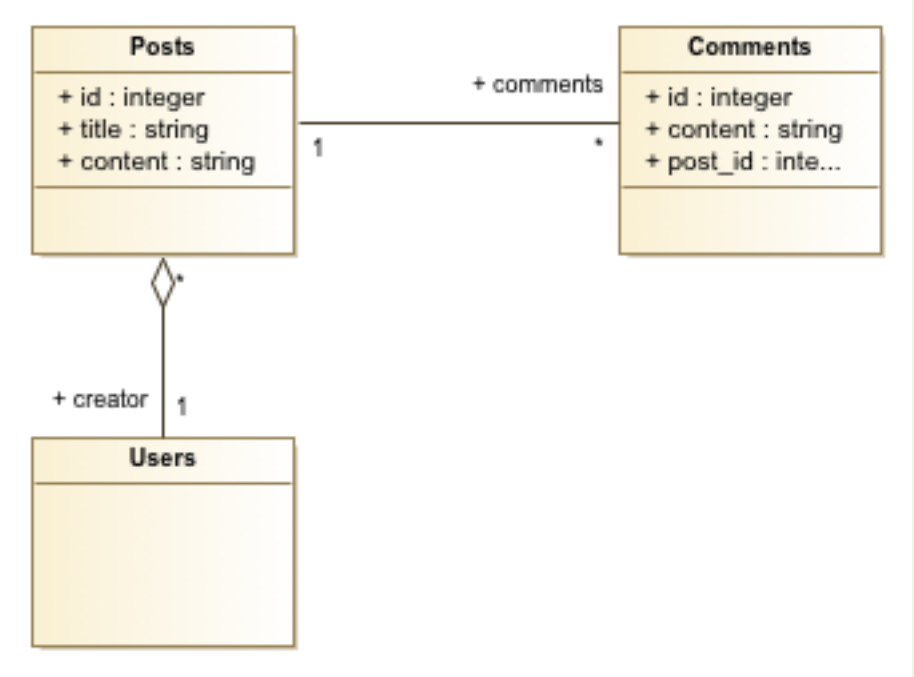
Model

Laravel menggunakan Eloquent sebagai Object Relation Model

Setiap Table di database mempunyai 1 Model class di PHP

Eloquent secara default menggunakan

Auto increment primary key (integer)



Model

Membuat class model secara otomatis

1. Menggunakan terminal eksekusi perintah ini di direktori projek

php artisan make:model Post –m

Otomatis membuat file Post.php berisi class dan migration file

unisa
Universitas 'Aisyiyah
Yogyakarta

Model

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Post extends Model
{
    //
}
```

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePostsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('posts', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->text('title')->nullable();
            $table->text('content')->nullable();
            $table->bigInteger('user_id');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('posts');
    }
}
```

Model

Membuat class model secara otomatis

2. Menggunakan terminal eksekusi perintah ini di direktori projek

php artisan make:model Comment –m

Otomatis membuat file Comment.php berisi class dan migration file

unisa
Universitas 'Aisyiyah
Yogyakarta

Model

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Comment extends Model
{
    //
}
```

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateCommentsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('comments', function (Blueprint $table) {
            $table->bigIncrements('id');
            $table->text('content')->nullable();
            $table->bigInteger('post_id');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('comments');
    }
}
```

Model

Migration

Laravel memudahkan untuk mengupdate database

Eksekusi perintah

php artisan migrate

Model

Relationship:

1. One to One menggunakan hasOne (Post hasOne User)
2. One to Many menggunakan hasMany (Post hasMany Comment)
3. One to Many (Inverse) menggunakan belongsTo (Comment hasOne Post)
4. Many to Many menggunakan belongsToMany

Post hasMany Comment karena post_id adanya di table comments

Model

Relationship:

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Post extends Model
{
    public function creator()
    {
        return $this->belongsTo('App\User', 'user_id', 'id');
    }

    public function comments()
    {
        return $this->hasMany('App\Comment');
    }
}
```

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Comment extends Model
{
    public function post()
    {
        return $this->belongsTo('App\Post');
    }
}
```

```php
class User extends Authenticatable implements MustVerifyEmail
{
    use Notifiable;

    public function posts()
    {
        return $this->hasMany('App\Post');
    }
```

Controller

Controller: yang mengatur antara View dan Model

Untuk mengakses controller dari View memerlukan Routing. Routing menggunakan HTTP Verbs

Disimpan di web.php atau api.php

```
Route::get($uri, $callback);
Route::post($uri, $callback);
Route::put($uri, $callback);
Route::patch($uri, $callback);
Route::delete($uri, $callback);
Route::options($uri, $callback);
```

Controller

Untuk routing yang memerlukan ijin akses menggunakan middleware.
Middleware adalah fungsi yang digunakan sebelum mengeksekusi request
Tugas middleware yang terpenting adalah filtering request, contoh ijin akses.

Untuk mengedit routing, web.php dan api.php bisa digunakan.
web.php untuk yang berhubungan dengan routing halaman web
api.php untuk yang berhubungan dengan API Ajax

Universitas 'Aisyiyah Yogyakarta

Controller

web.php

```
Route::get('/', function () {
    return view('welcome');
});

Auth::routes(['verify' => true]);

Route::get('/home', 'UserController@posts')->middleware('verified', 'auth');
Route::post('/posts', 'UserController@create_or_update_post')->middleware('verified', 'auth');

Route::post('/comments', 'HomeController@create_comment');
```

api.php

```
Route::get('/posts/{id}', 'HomeController@post');

Route::middleware('auth:api')->get('/user', function (Request $request) {
    return $request->user();
});

Route::middleware('auth:api')->post('/posts', 'UserController@create_or_update_post');
Route::middleware('auth:api')->delete('/posts/{id}', 'UserController@delete_post');
```

Controller

Untuk membuat controller dilakukan dengan mengeksekusi perintah berikut:

php artisan make:controller UserController

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Post;
use Illuminate\Support\Facades\Auth;

class UserController extends Controller
{
    //
    public function posts(Request $request)
    {
        return view('home');
    }

    public function create_or_update_post(Request $request)
    {
}

    public function delete_post(Request $request, $id)
    {
    }
}
```

View

Menampilkan tampilan di browser. Menggunakan blade template.

Gabungan html, css, javascript, dan kata kunci spesial.

Sebagai dasar kita menggunakan home.blade.php.

Kita akan menggunakan jquery di cdn, edit file resources/views/layouts/app.blade.php

```
<!-- Scripts -->
<script src="{{ asset('js/app.js') }}" defer></script>
<script
src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
integrity="sha256-pasqAKBDmFT4eHoN2ndd6lN370kFiGUFyTiUHWhU7k8="
crossorigin="anonymous"></script>
```

Universitas
Yogyakarta

```blade
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-12">
            <table class="table table-striped">
                <thead>
                    <tr>
                        <th style="width:50px;">No.</th>
                        <th style="width:350px; text-align: center;">Judul</th>
                        <th style="text-align: center;">Isi</th>
                        <th style="width:100px; text-align: center;">Dimodifikasi Tanggal</th>
                        <th style="width:150px; text-align: center;">
                            <button type="button" id="btn-add">Buat Baru</button>
                        </th>
                    </tr>
                </thead>
                <tbody>
                    @foreach ($posts as $post)
                    <tr>
                        <td>
                           {{$loop->iteration}}
                        </td>
                        <td>
                            {{$post->title}}
                        </td>
                        <td>
                            {{$post->content}}
                        </td>
                        <td>
                            {{$post->updated_at}}
                        </td>
                        <td>
                            <button type="button" class="btn-edit" data-id="{{$post->id}}">Edit</button>
                            <button type="button" class="btn-delete" data-id="{{$post->id}}">Delete</button>
                        </td>
                    </tr>
                    @endforeach
                </tbody>
            </table>
        </div>
    </div>
</div>
```

```html
<div class="modal" id="edit-post-modal" tabindex="-1" role="dialog">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="modal-title">Modal title</h5>
          <button type="button" class="close" data-dismiss="modal" aria-label="Close">
            <span aria-hidden="true">&times;</span>
          </button>
        </div>
        <div class="modal-body">
            <form class="form-horizontal">
                <input type="hidden" name="post_id" id="post_id" value="">
                <div class="form-group">
                    <label  class="col-sm-2 control-label" for="title">Judul</label>
                    <div class="col-sm-10">
                        <input class="form-control" type="text" id="title" name="title" value="" placeholder="Silahkan isi judul" />
                    </div>
                </div>
                <div class="form-group">
                    <label  class="col-sm-2 control-label" for="content">Isi</label>
                    <div class="col-sm-10">
                        <textarea class="form-control" style="min-height: 200px;" name="content" id="content"></textarea>
                    </div>
                </div>
            </form>
        </div>
        <div class="modal-footer">
          <button type="button" class="btn btn-primary" id="btn-save">Save</button>
          <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
        </div>
      </div>
    </div>
  </div>
```

View

Ajax

Untuk mengaktifkan ajax, API authentication harus diaktifkan. Diperlukan perubahan database untuk tabel users. Eksekusi perintah

php artisan make:migration enable_api_user_access

View

php artisan migrate

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class EnableApiUserAccess extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::table('users', function ($table) {
            $table->string('api_token', 80)->after('password')
                                ->unique()
                                ->nullable()
                                ->default(null);
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::table('users', function ($table) {
            $table->dropColumn(['api_token']);
        });
    }
}
```

View

Tambahkan kode berikut LoginController di app\Http\Controllers\Auth\LoginController.php

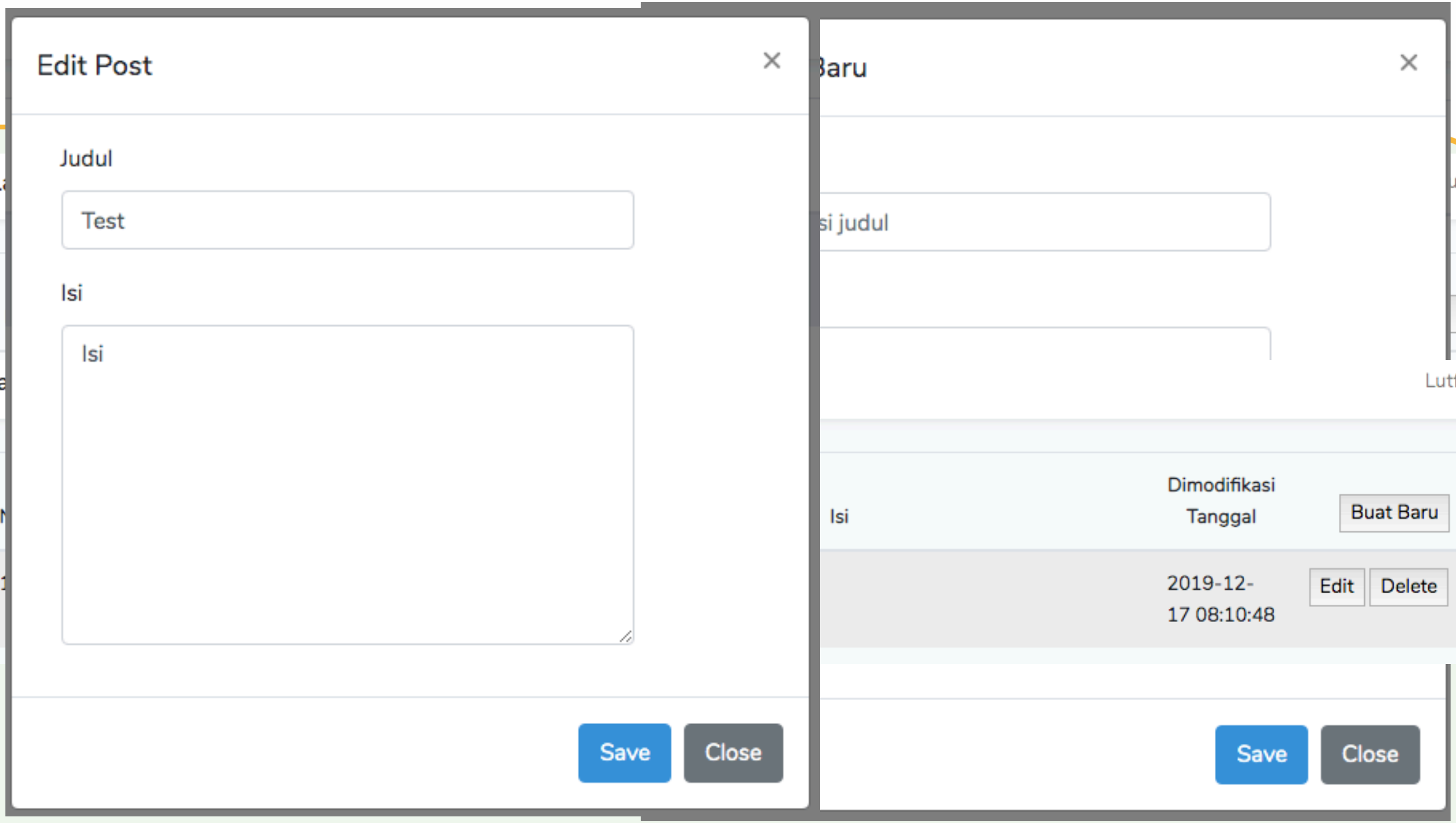
```php
public function __construct()
{
    $this->middleware('guest')->except('logout');
}

protected function authenticated(Request $request, $user)
{
    $user->api_token = \Str::random(80);
    $user->save();
}
```

UserController.php diubah menjadi

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Post;
use Illuminate\Support\Facades\Auth;

class UserController extends Controller
{
    public function posts(Request $request)
    {
        $posts = Auth::user()->posts;
        return view('home',
            ['posts' => $posts,
             'api_token' => Auth::user()->api_token]);
    }

    public function create_or_update_post(Request $request)
    {
        if ($request->has('id') && strlen($request->id) > 0)
        {
            $post = Post::find($request->id);
        }
        else
        {
            $post = new Post();
            $post->user_id = Auth::id();
            $post->save();
        }
        $post->title = $request->title;
        $post->content = $request->content;
        $post->save();
        return response()->json([
            'result' => 'OK',
            'data' => $post
        ]);
    }

    public function delete_post(Request $request, $id)
    {
        if ($id)
        {
            $post = Post::find($id);
            $post->delete();
            return response()->json([
                'result' => 'OK'
            ]);
        }
        return response()->json([
            'result' => 'ERROR',
            'message' => 'Invalid ID'
        ]);
    }

    public function post(Request $request, $id)
    {
        if ($id)
        {
            $post = Post::find($id);
            return response()->json([
                'result' => 'OK',
                'data' => $post
            ]);
        }
        return response()->json([
            'result' => 'ERROR',
            'message' => 'Invalid ID'
        ]);
    }
}
```

```
<script>
    $( document ).ready(function() {
        $('#btn-add').click(function(e) {
            $('#modal-title').text('Buat Post Baru');
            $('#post_id').val('');
            $('#title').val('');
            $('#content').val('');
            $('#edit-post-modal').modal('show');
        });

        $('#btn-save').click(function(e) {
            $.post( "{{ url('posts') }}",
                    {
                        _token: "{{ csrf_token() }}",
                        id: $('#post_id').val(),
                        title: $('#title').val(),
                        content: $('#content').val(),
                        api_token: '{{$api_token}}',
                    },
                function(data, status){
                    if(data.result=='OK')
                    {
                        $('#edit-post-modal').modal('hide');
                        location.reload();
                    }
                    else
                    {
                        alert(data.message);
                    }
                });
        });
```

```
        $('.btn-edit').click(function(e) {
            $.get( "{{ url('api/posts') }}/"+$(this).data('id')+"?api_token={{$api_token}}",
                function(data, status){
                    if(data.result=='OK')
                    {
                        data = data.data;
                        $('#post_id').val(data.id),
                        $('#title').val(data.title),
                        $('#content').val(data.content),
                        $('#modal-title').text('Edit Post');
                        $('#edit-post-modal').modal('show');
                    }
                    else
                    {
                        alert(data.message);
                    }
                });
        });

        $('.btn-delete').click(function(e) {
            if (confirm("Apakah anda yakin ingin menghapus ini?"))
            {
                $.ajax({
                url: "{{ url('api/posts') }}/"+$(this).data('id')+"?api_token={{$api_token}}",
                type: 'DELETE',
                success: function(data) {
                    if(data.result=='OK')
                    {
                        location.reload();
                    }
                    else
                    {
                        alert(data.message);
                    }
                }
            });
            }
        });
    });
</script>
@endsection
```


Edit Post
Judul
Test
Isi
Isi
Save
Close
Lutfi
Isi
Dimodifikasi Tanggal
Buat Baru
2019-12-17 08:10:48
Edit
Delete
Save
Close

PENUTUP BELAJAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

اَللَّهُمَّ أَرِنَا الْحَقَّ حَقًّا وَارْزُقْنَا اتِّبَاعَه ُ وَأَرِنَا الْبَاطِلَ بَاطِلاً وَارْزُقْنَا اجْتِنَابَهُ

Ya Allah Tunjukkanlah kepada kami kebenaran sehinggga kami dapat mengikutinya,

Dan tunjukkanlah kepada kami keburukan sehingga kami dapat menjauhinya.


UNIVERSITAS 'AISYIYAH
YOGYAKARTA
unisa
Universitas 'Aisyiyah
Yogyakarta