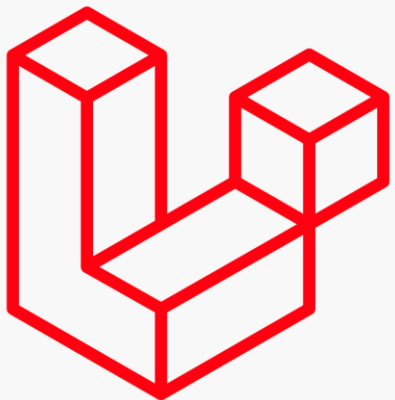


Laravel Training d1

Tarmizi Sanusi

Khirulnizam Abd Rahman

<http://fstm.kuis.edu.my/blog/laravel>



Module Outcome

Day 1

- Create Laravel Project
- Working with Database.
- Working with Model
- Working with Controllers
- Working with Views

What will be needed

- Laragon
- phpMyAdmin
- VS Code

2. Route to appropriate Laravel Controller

1. Submit User Request



Routing

routes/web.php

app/Http/Controllers

Controller

3. Interact with Data Model

4. Controller invokes results View

resources/views

View

5. Render view in users browser

app/

Model



Database

Create Laravel First Project

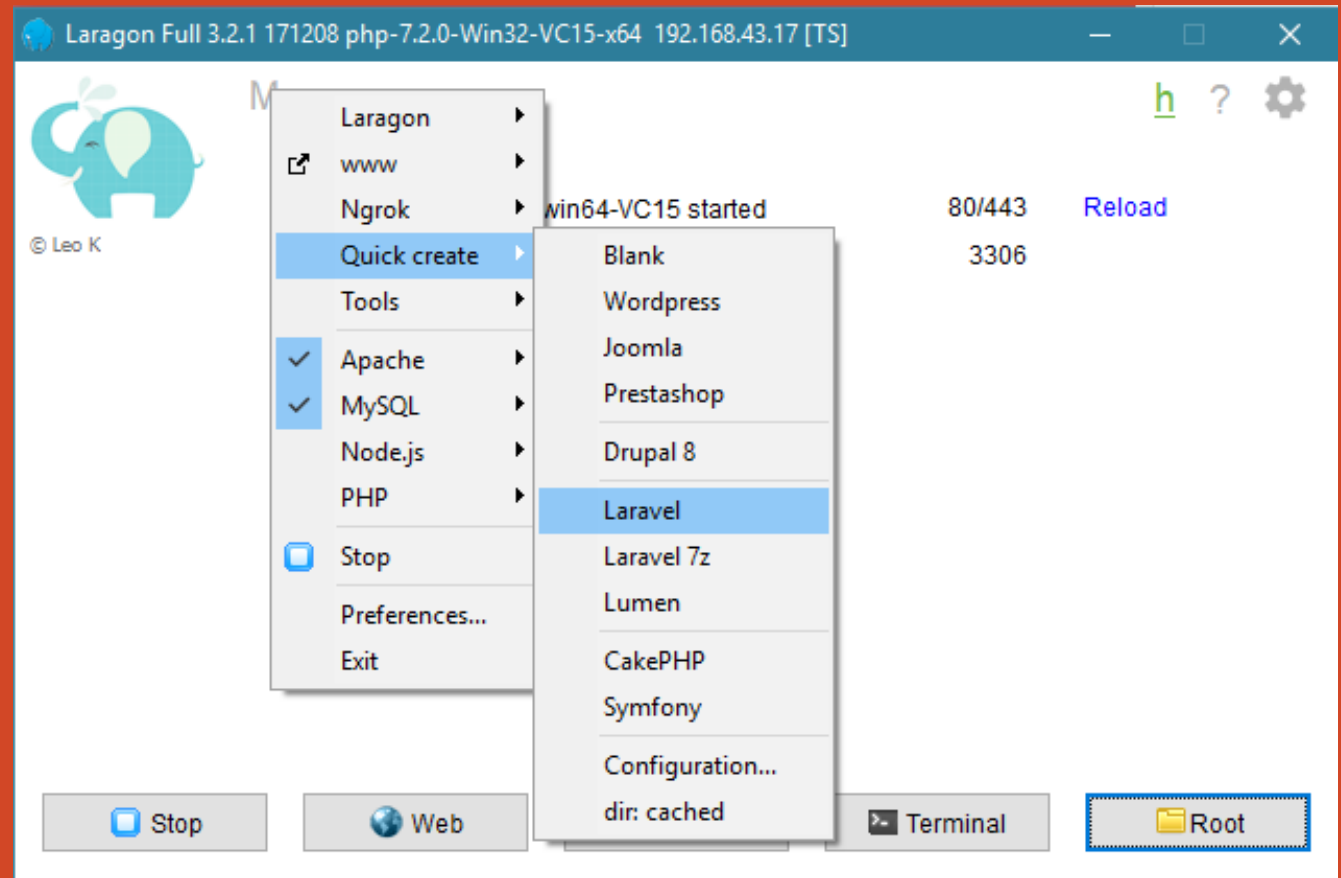
Install Laravel by issuing the Composer create-project command in your terminal

```
composer create-project --prefer-dist laravel/laravel project-name
```

Laravel

[DOCS](#)[LARACASTS](#)[NEWS](#)[BLOG](#)[NOVA](#)[FORGE](#)[VAPOR](#)[GITHUB](#)

With Laragon?
Never been EASIER.



Laragon

What is Laragon?

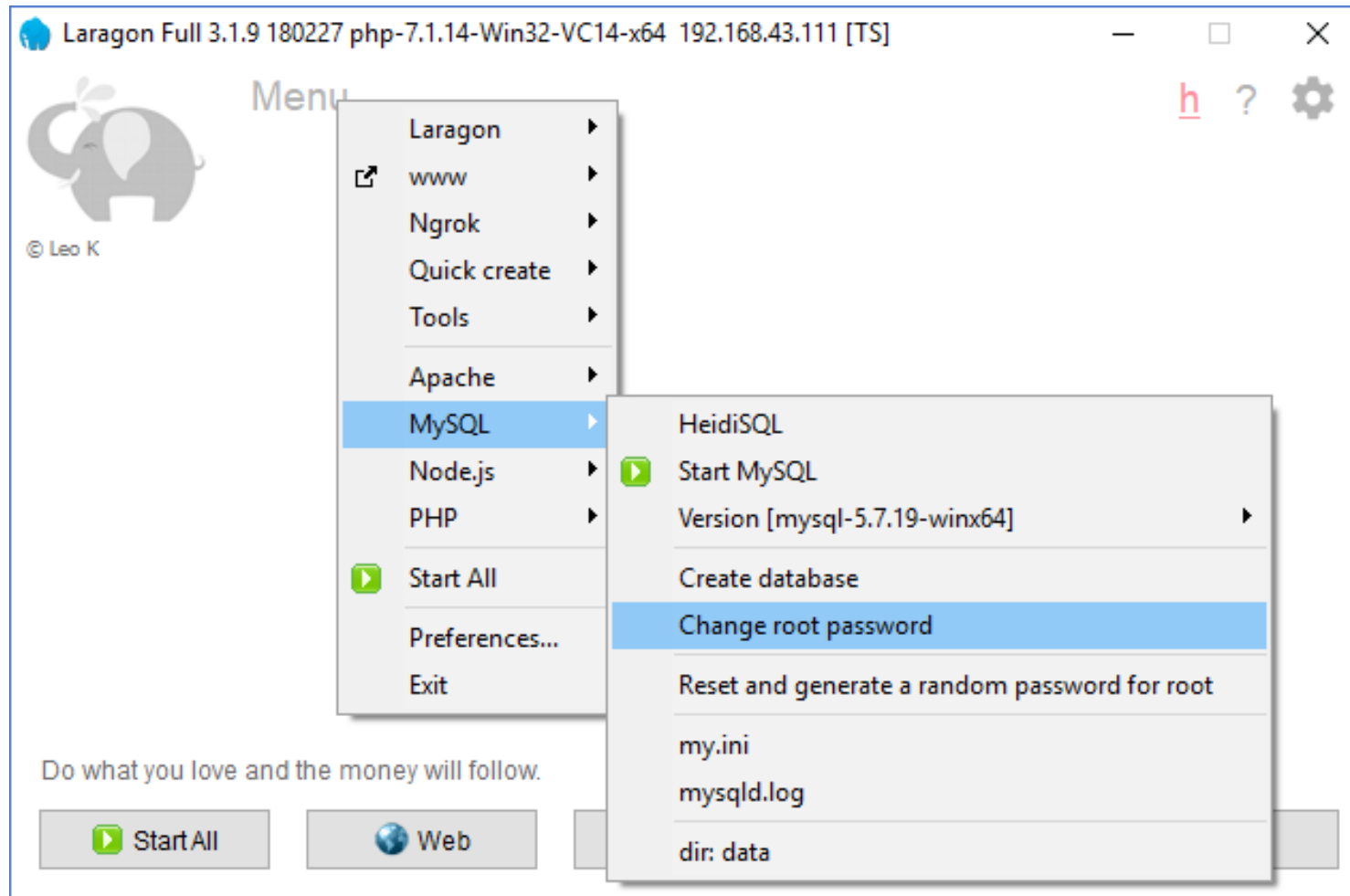
Laragon is a portable, isolated, fast & powerful universal development environment for PHP, Node.js, Python, Java, Go, Ruby. It is fast, lightweight, easy-to-use and easy-to-extend.

Why Laragon?

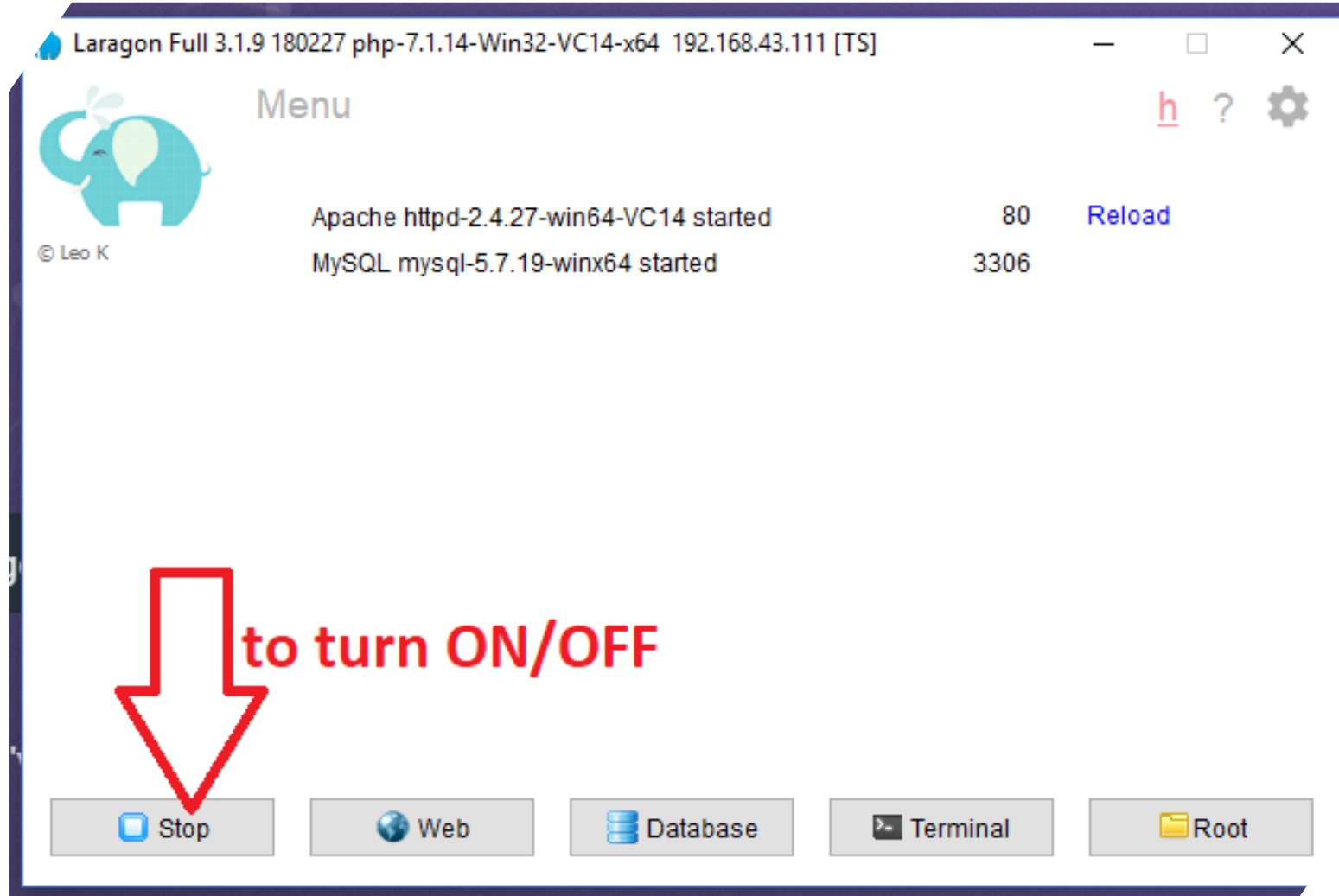
Laragon is great for building and managing modern web applications. It is focused on performance - designed around stability, simplicity, flexibility and freedom.

Laragon improves web development. Developers all over the world are using Laragon to make apps quickly and easily. It is used by thousands of developers with loves. You can check out the Testimonials to see how users think of Laragon and it's features page for more details.

Change DB root password

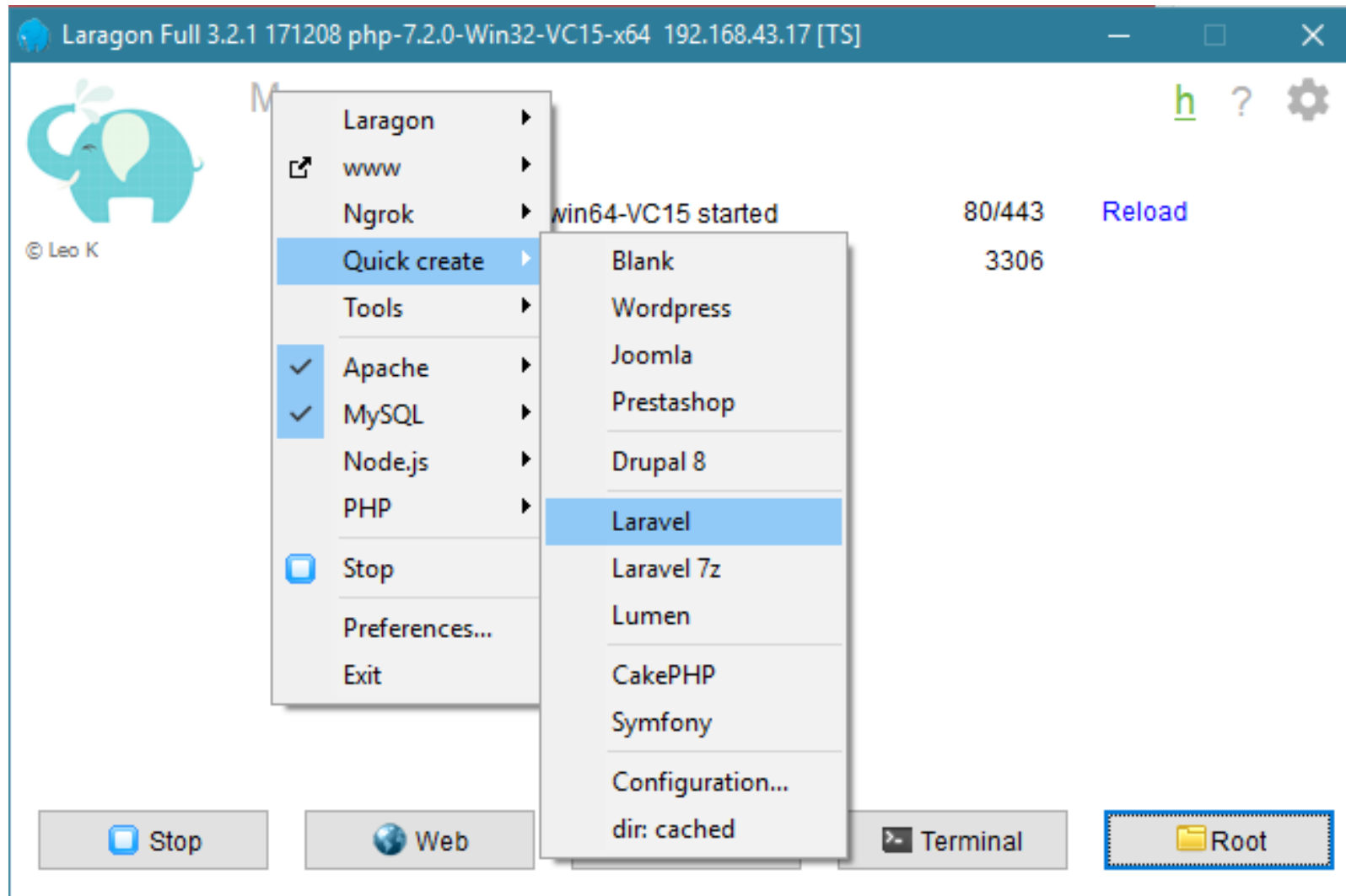


Turn On Laragon

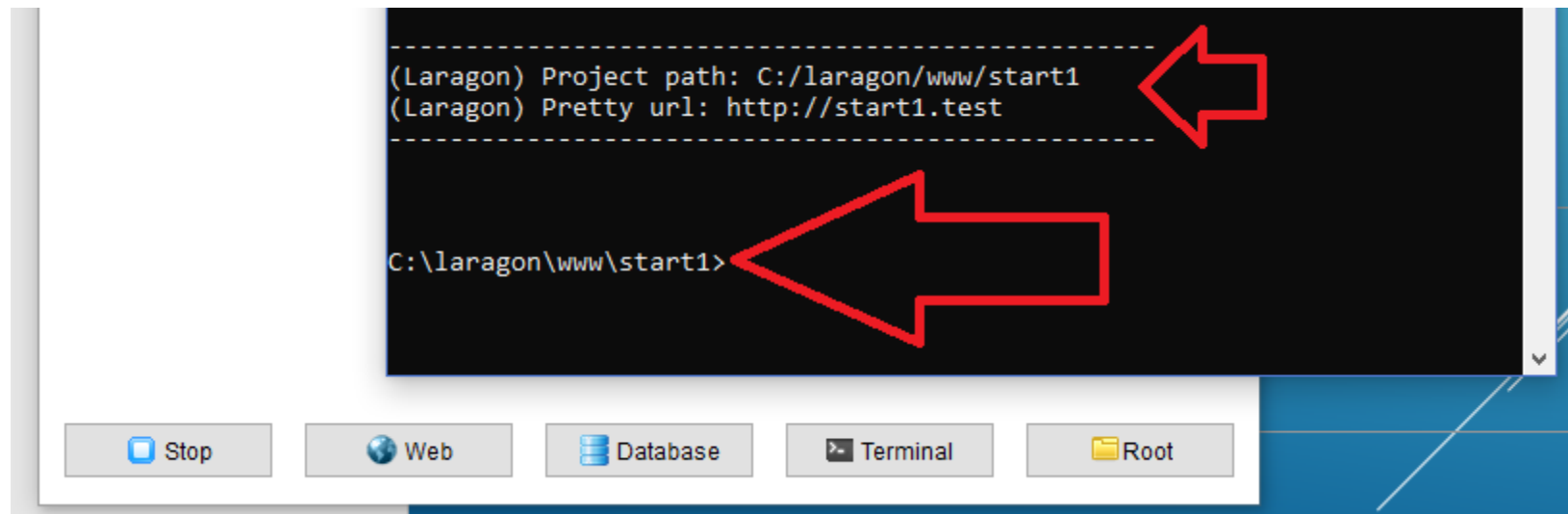


By default Laragon will served port 80 for Apache and port 3306 for DB

Create Project



Project with Pretty URL

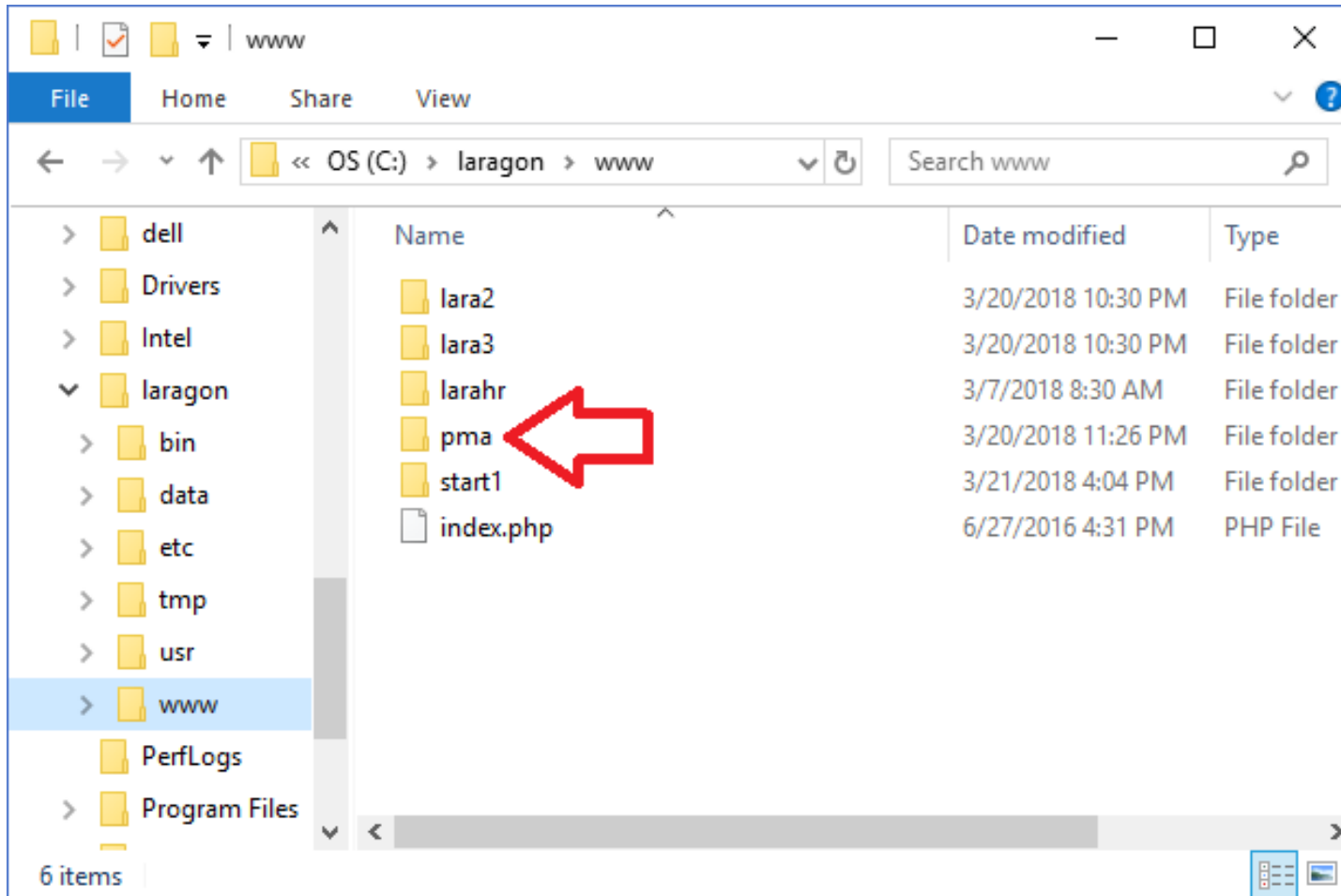


Test the Pretty url in browser

Update DB credentials in .env

```
8
9 DB_CONNECTION=mysql
10 DB_HOST=127.0.0.1
11 DB_PORT=3306
12 DB_DATABASE=start1
13 DB_USERNAME=root
14 DB_PASSWORD=abc123
15
```

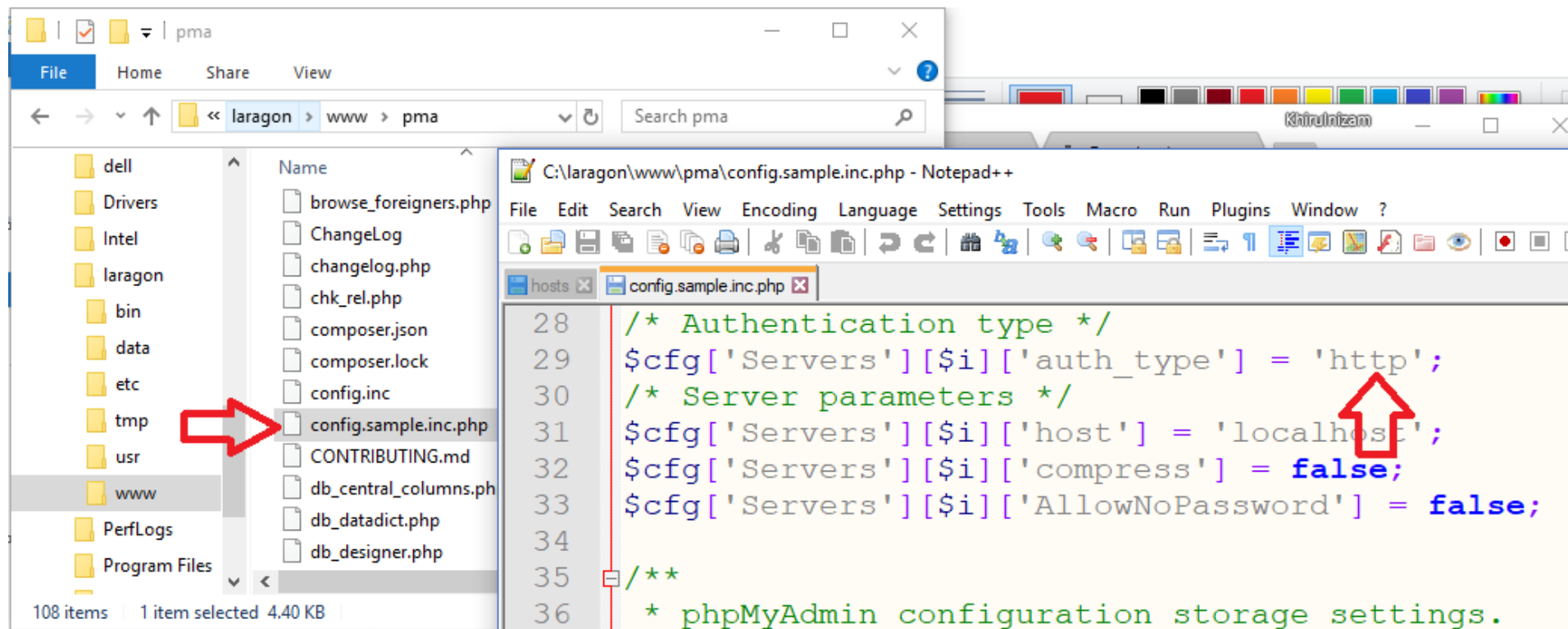
Download and install phpmyadmin.Net



Try access

<http://localhost/pma>

Phpmyadmin config.inc



Authentication Module

Laravel makes implementing authentication very simple. In fact, almost everything is configured for you out of the box.

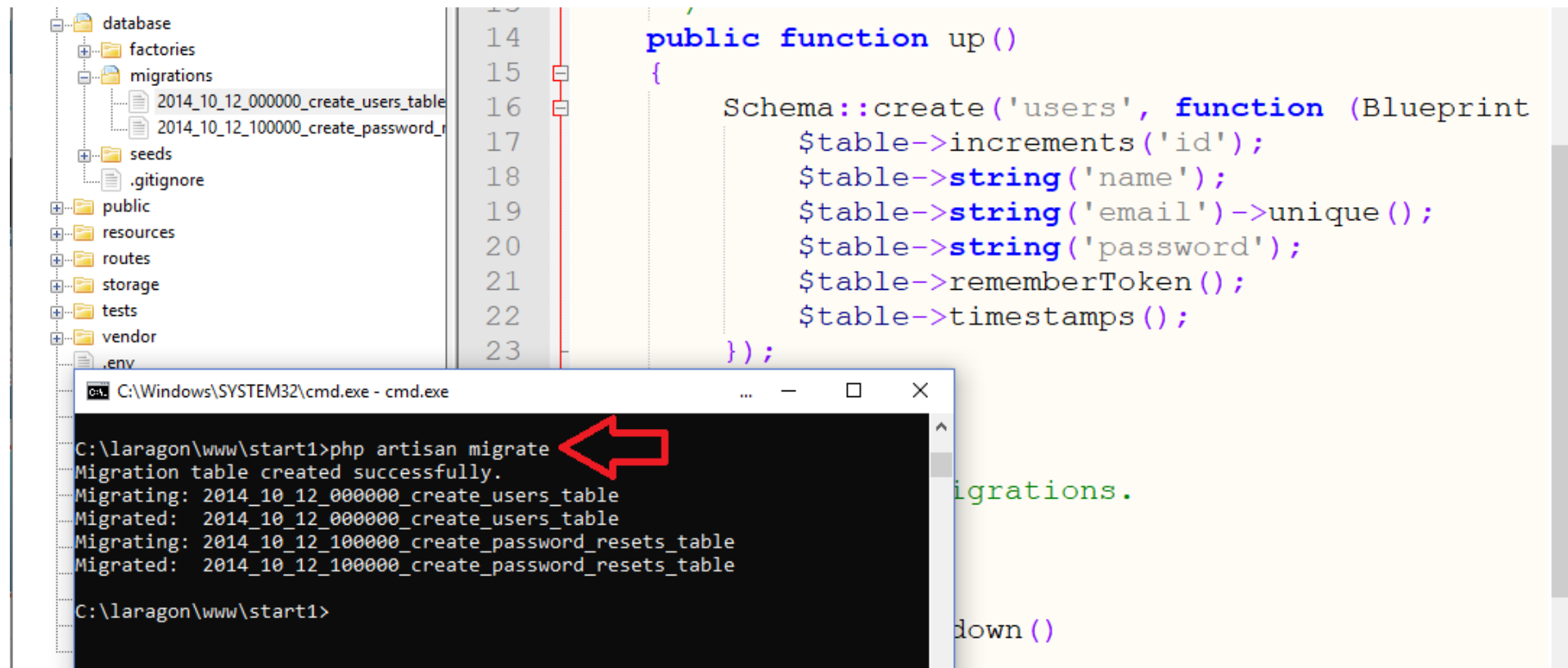
```
composer require laravel/ui --dev
```

```
php artisan ui vue --auth
```

```
npm install
```

```
npm run dev
```

Authentication Module



The screenshot displays a Laravel project's file structure on the left, including folders like `database`, `factories`, `migrations`, `seeds`, `public`, `resources`, `routes`, `storage`, `tests`, `vendor`, and `.env`. The `migrations` folder contains two files: `2014_10_12_000000_create_users_table` and `2014_10_12_100000_create_password_resets_table`. The main editor shows the `up()` method of a migration class, which uses the `Schema::create()` method to create a `users` table with columns for `id`, `name`, `email`, `password`, and `timestamps`. Below the editor, a terminal window titled `C:\Windows\SYSTEM32\cmd.exe - cmd.exe` shows the command `php artisan migrate` being executed. A red arrow points to this command. The terminal output indicates that the migration table was created successfully and that the specified migrations were applied.

```
14 public function up()  
15 {  
16     Schema::create('users', function (Blueprint  
17         $table->increments('id');  
18         $table->string('name');  
19         $table->string('email')->unique();  
20         $table->string('password');  
21         $table->rememberToken();  
22         $table->timestamps();  
23     });
```

```
C:\laragon\www\start1>php artisan migrate  
Migration table created successfully.  
Migrating: 2014_10_12_000000_create_users_table  
Migrated: 2014_10_12_000000_create_users_table  
Migrating: 2014_10_12_100000_create_password_resets_table  
Migrated: 2014_10_12_100000_create_password_resets_table  
  
C:\laragon\www\start1>
```

Login

E-Mail Address

Password

☐ Remember Me

Login

[Forgot Your Password?](#)

Reset Password

E-Mail Address

Send Password Reset Link

Register

Name

E-Mail Address

Password

Confirm Password

Register

Working with Database

Database Migrations

Migrations are like version control for your database.

Make Migrations

```
php artisan make:migration create_trainings_table
```

Run Migrations

```
php artisan migrate
```

```
public function up()
{
    Schema::create('trainings', function (Blueprint $table) {
        $table->bigIncrements('id');
        $table->string('title');
        $table->text('description');
        $table->string('trainer');
        $table->timestamps();

        $table->unsignedBigInteger('user_id');
        $table->foreign('user_id')->references('id')->on('users');
    });
}
```

Working with Database

Database Factories

Laravel has a feature called model factories that allows you to build fake data for your models.

Make Factory

```
php artisan make:factory TrainingFactory
```

Run Factory

```
php artisan tinker  
  
factory('App\Training',10)->create();
```

```
use App\Training;  
use Faker\Generator as Faker;  
  
$factory->define(Training::class, function (Faker $faker) {  
    return [  
        'title' => $faker->sentence(5),  
        'description' => $faker->text(),  
        'trainer' => $faker->name,  
        'user_id' => factory('App\User')->create()->id,  
    ];  
});
```

Working with Database

Database Seeds

Laravel includes a simple method of seeding your database with test data using seed classes.

Make Seeder

```
php artisan make:seeder TrainingsTableSeeder
```

Run Seeder

```
php artisan db:seed
```

On database/seeds/TrainingsTableSeeder.php

```
public function run()
{
    DB::table('trainings')->insert([
        'title' => 'Laravel Training 6 Days(Advanced)',
        'description' => 'The 6 Days training focus on enhancement API to Restful Architecture',
        'trainer' => 'Khirulnizam',
        'user_id' => factory('App\User')->create()->id,
    ]);
}
```

On database/seeds/DatabaseSeeder.php

```
public function run()
{
    $this->call(TrainingsTableSeeder::class);
}
```

How to create and run migrations, seeder and factory?

Here are the example.

Make Migrations

```
php artisan make:migration create_trainings_table
```

```
php artisan make:migration alter_articles_table_add_attachment
```

Run Migrations

```
php artisan migrate
```

Make Seeder

```
php artisan make:seeder TrainingsTableSeeder
```

Run Seeder

```
php artisan db:seed
```

Make Factory

```
php artisan make:factory TrainingFactory
```

Run Factory

```
php artisan tinker
```

```
factory('App\Training',10)->create();
```

Model

Models typically live in the app directory.

All Eloquent models
extend Illuminate\Database\Eloquent\Model class.

Model

```
php artisan make:model Training
```

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Training extends Model
{

}
```

Model

```
protected $fillable = [  
    'title', 'description', 'trainer', 'user_id', 'attachment'  
];
```

```
public function user()  
{  
    return $this->belongsTo('App\User');  
}
```

```
protected $table= 'articles';
```

```
//define $article->submitted_date  
public function getSubmittedDateAttribute(){  
    return $this->created_at->format('d/m/Y');  
}
```

```
public function scopePublished($query){  
    return $query->where('published',1);  
}
```

Model used to define fillable attributes.

Model also can be used to define relationship

Model also can be used to to specify custom table.

Model also can be used to define mutator

Model also can be used to define local scope

Controller

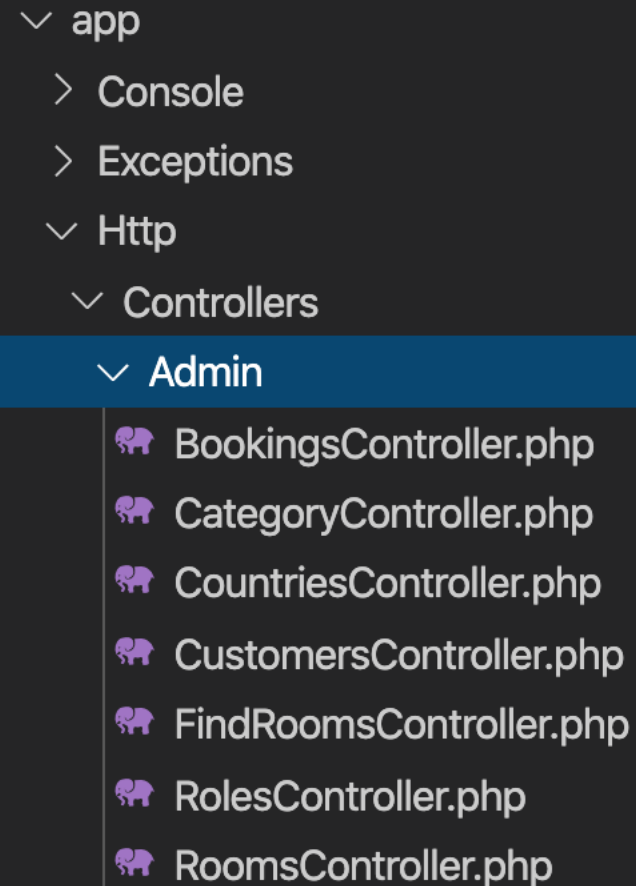
Instead of defining all of your request handling logic as Closures in route files, you may wish to organize this behavior using Controller classes.








Controllers can group related request handling logic into a single class.

Controllers are stored in the `app/Http/Controllers` directory.

Make Controller

```
php artisan make:controller TrainingController
```



- ✓ app
 - > Console
 - > Exceptions
 - ✓ Http
 - ✓ Controllers
 - ✓ Admin
 -  BookingsController.php
 -  CategoryController.php
 -  CountriesController.php
 -  CustomersController.php
 -  FindRoomsController.php
 -  RolesController.php
 -  RoomsController.php

Controller

7 method in controllers

Make Controller

```
php artisan make:controller TrainingController
```

Action

index

create

store

show

edit

update

destroy

Controller - Index

Description	URL	Controller Function	View File
Default page for showing all the training.	GET example.com/trainings	index()	app/views/trainings/index.blade.php

```
use App\Training;
```

```
class TrainingController extends Controller  
{
```

```
//
```

```
public function index()  
{
```

```
    $trainings = Training::all();
```

```
    return view('trainings.index')->with(compact('trainings'));
```

```
}
```

Controller - Create

Description	URL	Controller Function	View File
Show the form to create a new training.	GET example.com/trainings/create	create()	app/views/trainings/create.blade.php

```
public function create()
{
    return view('trainings.create');
}
```

Controller - Store

Description	URL	Controller Function	View File
Process the create form submit and save the training to the database.	POST example.com/trainings/create	store()	NONE

//Method 2

```
$user = auth()->user();  
$article = $user->trainings()->create($request->only('title','description','trainer'));
```

```
use Illuminate\Support\Facades\Auth;
```

```
class TrainingController extends Controller  
{
```

```
@ -17,4 +18,17 @@ public function create()  
{
```

```
    return view('trainings.create');
```

```
public function store(Request $request)  
{
```

```
    //Method 1
```

```
    $training = new Training();
```

```
    $training->title = $request->get('title');
```

```
    $training->description = $request->get('description');
```

```
    $training->trainer = $request->get('trainer');
```

```
    $training->user_id = Auth::id();
```

```
    $training->save();
```

```
    return redirect('/trainings');
```

```
}
```

Views

Views contain the HTML served by your application and separate your controller / application logic from your presentation logic.

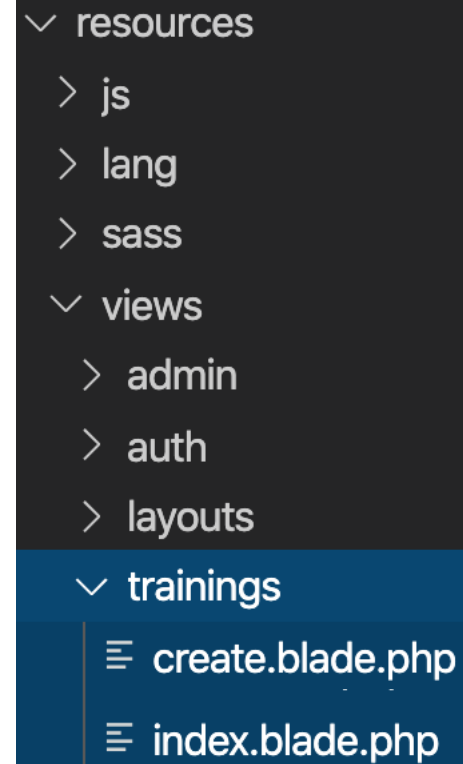
Views are stored in the resources/views directory

```
✓ resources
  > js
  > lang
  > sass
  ✓ views
    > admin
    > auth
    > layouts
  ✓ trainings
    ≡ create.blade.php
    ≡ edit.blade.php
    ≡ index.blade.php
```

Views

Views are stored in the resources/views directory

1. Go to resources/views
2. Create training folder
3. Inside training folder, create:
 1. create.blade.php
 2. index.blade.php



```

✓ resources
  > js
  > lang
  > sass
  ✓ views
    > admin
    > auth
    > layouts
  ✓ trainings
    ≡ create.blade.php
    ≡ index.blade.php

```

Views – Index View

Views are stored in the resources/views directory

1. Go to resources/views/trainings/index.blade.php

```
@extends('layouts.app')

@section('content')
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">Training Index</div>

                <div class="card-body">
                    <table class="table">
                        <thead>
                            <tr>
                                <th>ID</th>
                                <th>Title</th>
                                <th>Creator</th>
                                <th>Actions</th>
                            </tr>
                        </thead>
                        <tbody>
                            @foreach ($trainings as $training)
                                <tr>
                                    <td>{{ $training->id }}</td>
                                    <td>{{ $training->title }}</td>
                                    <td>{{ $training->user_id }}</td>
                                    <td>
                                        <a href="" class="btn btn-primary">Show</a>
                                        <a href="" class="btn btn-success">Edit</a>
                                        <a href="" class="btn btn-danger"
                                            onclick="return confirm('Are you sure?')">Delete</a>
                                    </td>
                                </tr>
                            @endforeach
                        </tbody>
                    </table>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

Views – Create View

Views are stored in the resources/views directory

1. Go to resources/views/trainings/create.blade.php

```
@extends('layouts.app')
```

```
@section('content')
```

```
<div class="container">
```

```
<div class="row justify-content-center">
```

```
<div class="col-md-8">
```

```
<div class="card">
```

```
<div class="card-header">Create Training</div>
```

```
<div class="card-body">
```

```
<form action="" method="POST">
```

```
@csrf
```

```
<div class="form-group">
```

```
<label for="title">Title</label>
```

```
<input type="text" class="form-control" name="title">
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="title">Description</label>
```

```
<textarea name="description" cols="20" rows="10" class="form-control"></textarea>
```

```
</div>
```

```
<div class="form-group">
```

```
<label for="title">Trainer</label>
```

```
<input type="text" class="form-control" name="trainer">
```

```
</div>
```

```
<div class="form-group">
```

```
<button type="submit" class="btn btn-primary">Create New Training</button>
```

```
<a href="" class="btn btn-link">Cancel</a>
```

```
</div>
```

```
</form>
```

```
</div>
```

```
</div>
```

```
</div>
```

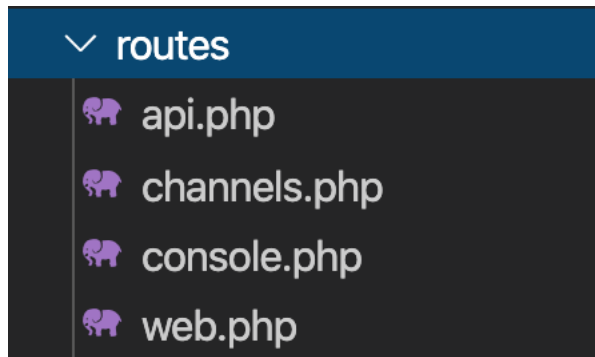
```
</div>
```

```
@endsection
```

Routing

All Laravel routes are defined in your route files, which are located in the routes directory.

<https://laravel.com/docs/6.x/routing>.

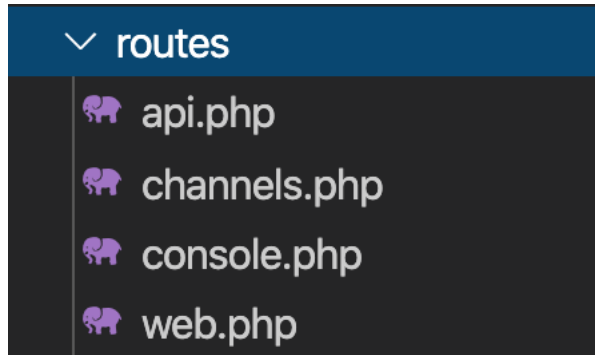


```
Route::get($uri, $callback);  
Route::post($uri, $callback);  
Route::put($uri, $callback);  
Route::patch($uri, $callback);  
Route::delete($uri, $callback);  
Route::options($uri, $callback);
```


Routing – Group and Named Route

All Laravel routes are defined in your route files, which are located in the routes directory.

<https://laravel.com/docs/6.x/routing>.



```
Route::group([
    'middleware' => ['auth'],
    'prefix' => 'trainings',
    'as' => 'training:'
], function() {
    Route::get('/', 'TrainingController@index')->name('index');
    Route::get('/create', 'TrainingController@create')->name('create');
    Route::post('/create', 'TrainingController@store')->name('store');
});
```

Routing – Group and Named Route

```
php artisan route:list
```

GET HEAD	trainings	training:index	App\Http\Controllers\TrainingController@index	web,auth
GET HEAD	trainings/create	training:create	App\Http\Controllers\TrainingController@create	web,auth
POST	trainings/create	training:store	App\Http\Controllers\TrainingController@store	web,auth

Hacks

Creating Index - <https://github.com/samtarmizi/laravel-spr/commit/1c925853f61da081f0e3ce298e58a4a423b66040>

Creating Create - <https://github.com/samtarmizi/laravel-spr/commit/7a5b5f3b643bcb8a265af3bb9f4e94e3aef95aa1>

Creating Store - <https://github.com/samtarmizi/laravel-spr/commit/2513b55c65bc4d27afa5004144ee2c7d41b3cbd7>