Code Guide
for

Laravel Folder Structure Coding Standard

Version 1.0

By
People
Software Development
Team

Revision History

Version	Description	Date	Author
1.0	Initial Draft	June 16, 2021	Waleed Ashraf
1.0.1	Initial Draft Modules	December 3, 2021	M. Zahoor Anwar

1. Introduction

This document is quick guide for developer to follow for standard naming standard in

- Laravel Folder Structure Practices
- Coding style Standards
- Code file name
- Method and variable name
- Migration file name
- Route name

Additionally, is provide quick over view of framework of folder structure for relevant technology stack frameworks used in peoplei.

2. Purpose

- A coding standard gives a uniform appearance to the codes written by different engineers.
- It improves readability, and maintainability of the code and it reduces complexity also.
- It helps in code reuse and helps to detect error easily.
- It promotes sound programming practices and increases efficiency of the programmers.

3. Project Folder Structure Practices

Evolve Laravel apps in a modular style using core, shared and feature modules

3.1 Laravel Directory Structure

|-- [+] app

|-- [+] Console

|-- Kernel.php

|-- [+] Exception

|-- Handler.php

|-- [+] Helpers

|-- [+] Http

|-- [+] Controllers

|-- [+] Api

|-- [+] Http

|-- [+] Middleware

|-- [+] Providers

|-- [+] config

|-- [+] database

|-- [+] factories

|-- [+] migrations

|-- [+] seeds

|-- [+] public

|-- [+] CSS

|-- [+] JS

|-- [+] images(optional)

|-- index.php (Handle Cors and Autoloader)

|-- [+] resources

|-- [+] js

|-- [+] components

|-- [+] lang

|-- [+] sass

|-- [+] views

|-- [+] routes

|-- api_v2.3.php (API Based version 2.3)

|-- api.php (API Based)

|--web.php (Session based for web)

|-- [+] storage

| -- [+] app

|-- [+] public

|-- [+] debugger

|-- [+] framework

|-- [+] cache

|-- [+] sessions

|-- [+] testing

|-- [+] views

|-- [+] tests (test cases for unit testing)

|-- [+] Feature

|-- [+] Unit

|-- [+] updates (work history and release notes)

|-- [+] vendor (Laravel Packages)

|-- .env (environment details)

|-- .gitignore (add files and folder don't want to upload on svn or repository)

|-- server.php (server configuration to set root path)

|-- composer (JSON file) (Laravel packages to use)

|-- package (JSON File) (Node packages lock file)

|-- package-lock (JSON File) (Node Packages)

|-- README.md (Read Me file)

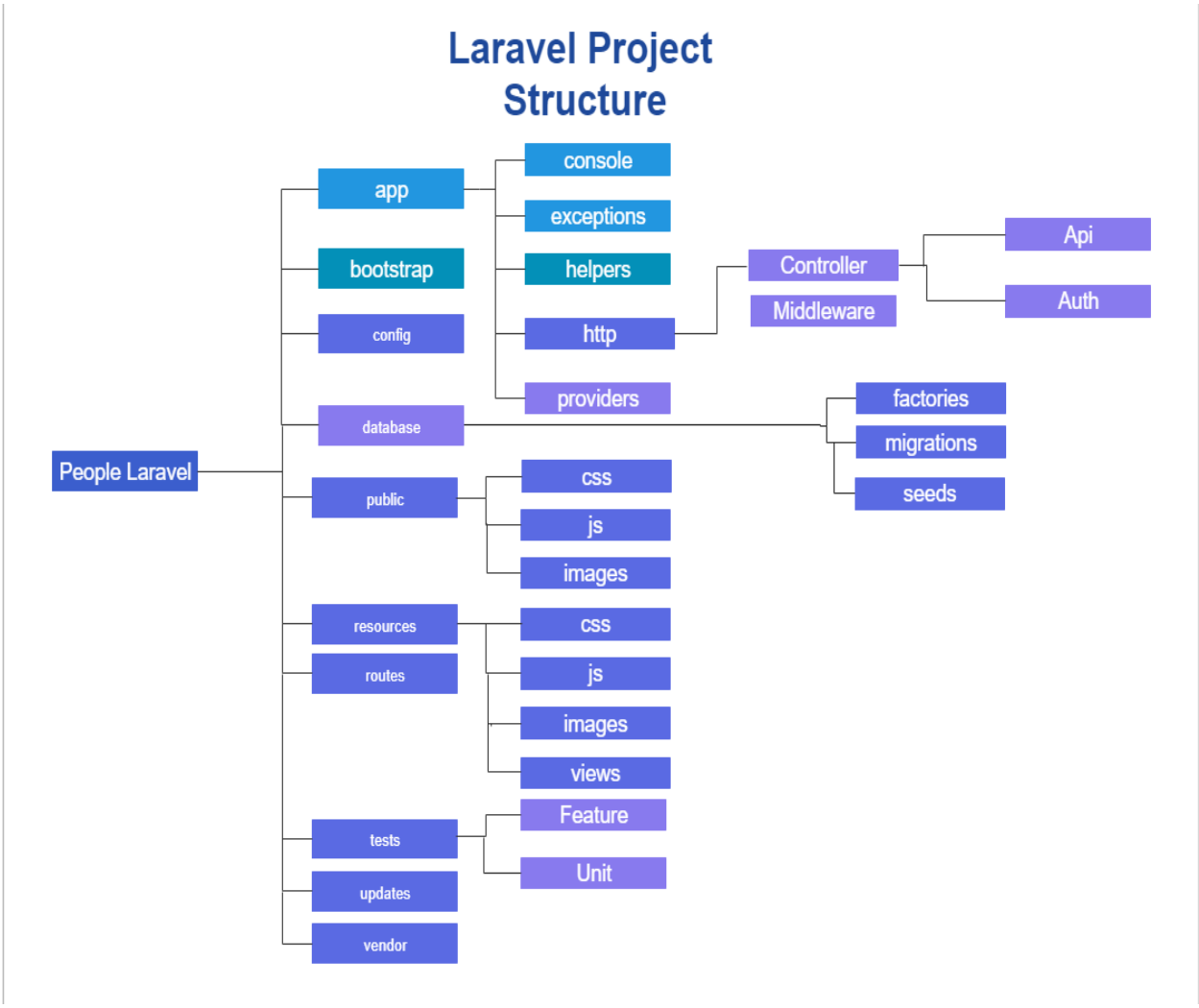


Fig 1.1

3.2 App Folder:

Which contains all the modules and components of your application. Every application has at least one module and one component. app folder contains the middleware for protecting routes and user according to roles and permissions.

3.3 Bootstrap Folder:

The **bootstrap directory** contains the app. php file which bootstraps the framework. This **directory** manages a cache **directory** which contains framework generated files for performance optimization such as the route and services cache files

3.4 Config Folder:

Where we can include various configuration settings and associate's parameters for the smooth functionality of Laravel application. Few selected files mentioned below,

- *app.php*
- *auth.php*
- *cache.php*
- *database.php*
- *session.php*

3.5 Database Folder

Where we can include various parameters for database functionalities. It will contain three directories.

- Factories
- Migrations
- Seeds

3.6 Public Folder

It is the root folder which helps in initializing the Laravel application.it will contain the following files and folders

- **.htaccess** − This file gives the server configuration
- **javascript ,css and images** − These files are considered as assets.
- **index.php** − This file is required for the initialization of a web application.

3.7 Resources Folder

Resource's directory contains the files which enhances your web application. The sub-folders included in this directory and their purpose is explained below

- **Js** These files are considered as assets.
- **Lang** This folder includes configuration for localization or internalization. (For multilanguage purpose)
- **View** − Views are the HTML files or templates which interact with end users and play a primary role in MVC architecture

3.8 Routes Folder

Routes Directory Contain the routes for Laravel application. it will contain the following files.

- Web.php (file defines routes that are for your web interface and manage by Web Middle Ware)
- Api.php (are stateless and are assigned the API middleware group)

- Channel.php (used for broadcast an event)
- Console.php (for executing commands in console bash)

3.9 Tests

All the unit test cases are included in this directory. The naming convention for naming test case classes is **camelCase** and follows the convention as per the functionality of the class.

3.10 Updates

Update directory contains the revision and history of work on Laravel people.

3.11 Vendor

Vendor Directory will contain the Laravel packages that we are using in project.

3.12 Tips to keep in mind:

1. Controllers, Models and migrations should be easily relatable and shouldn't create any confusion
2. Proper Migrations and Models
3. Make proper relations between models
4. Commenting code properly
5. Server-side error handling should be handled properly

4. Code Style Guide:

The wording of each guideline indicates how strong the recommendation is.

What	How	Good	Bad
Controller	Singular	ArticleController	ArticlesController
Route	Plural	articles:/1	article/1
Named route	snake_case with dot notation	Users.show_active	users.show-active/users-show-active
Model	Singular	User	Users
has One or belong to relationship	Singular	articleComment	articleComments/article_comment
All other relationship	Plural	articleComments	articleComment/article_comment
Table	Plural	article_comments	article_comment/articleComments
Pivot Table	Singular model names in	article_user	user_article,articles_users

	alphabetical order		
Table Column	snake_case without model name	meta_title	MetaTitle,article_meta_title
Foreign Key	Singular modal name with _id suffix	article_id	ArticleId,id_article,articles_id
Primary Key		Id	custom_id
Migration		2017_01_01_0000_create_article_table	2017_01_01_0000_ articles
Method	camelCase	getAll	get_all
Function	snake_case	abort_if	abortif
Method in resource controller	more infos:table	Store	saveArticle
Method in test class	camelCase	testGuestCannotSeeArticle	Test_guest_cannot_see_article
Model Property	snake_case	$model->model_property	$model->modelProperty
Variable	camelCase	$anyOtherVariable	$any_other_variable
Collection	descriptive plural	$activeUsers=User::active()->get()	$active,$data
Object	descriptive,singular	$activeUser= User::active()->first()	$users,$obj
Config and language file index	snake_case	articles_enabled	ArticlesEnabled,articles-enabled
View	Kebab-case	Show-filtered.blade.php	showFiltered.blade.php,show_filtered.blade.php
Config	Kebab-case	google-calendar.php	googleCalendar.php,google_calendar.php
Contract(interface)	adjective or noun	Authenticatable	Authentication Interface,IAuthentication
Trait	adjective	Notifiable	NotificationTrait

4.1 Deployment

1. Remove All dd and debuggers from Projects files.

2. Remove all extra comments from project files.
3. Set environments properly.

5. Modules:

nwidart/Laravel-modules is a Laravel package which was created to manage your large Laravel app using modules. A module is like a Laravel package, it has some views, controllers or models

5.1 Requirements:

The modules package requires **PHP 7.1** or higher and Laravel 7. *

5.2 Installation:

Please refer to the following Link: Installation and Setup | Laravel Modules Docs (nwidart.com)

5.3 Creating Module:

Creating a module is simple and straightforward. Run the following command to create a module.
php artisan module:make <module-name>

Replace <module-name> by your desired name
For further information: creating-module

5.4 Common Artisan Commands:

5.4.1 module:migrate:

Migrate the given module, or without a module an argument, migrate all modules.

php artisan module:migrate <module-name>

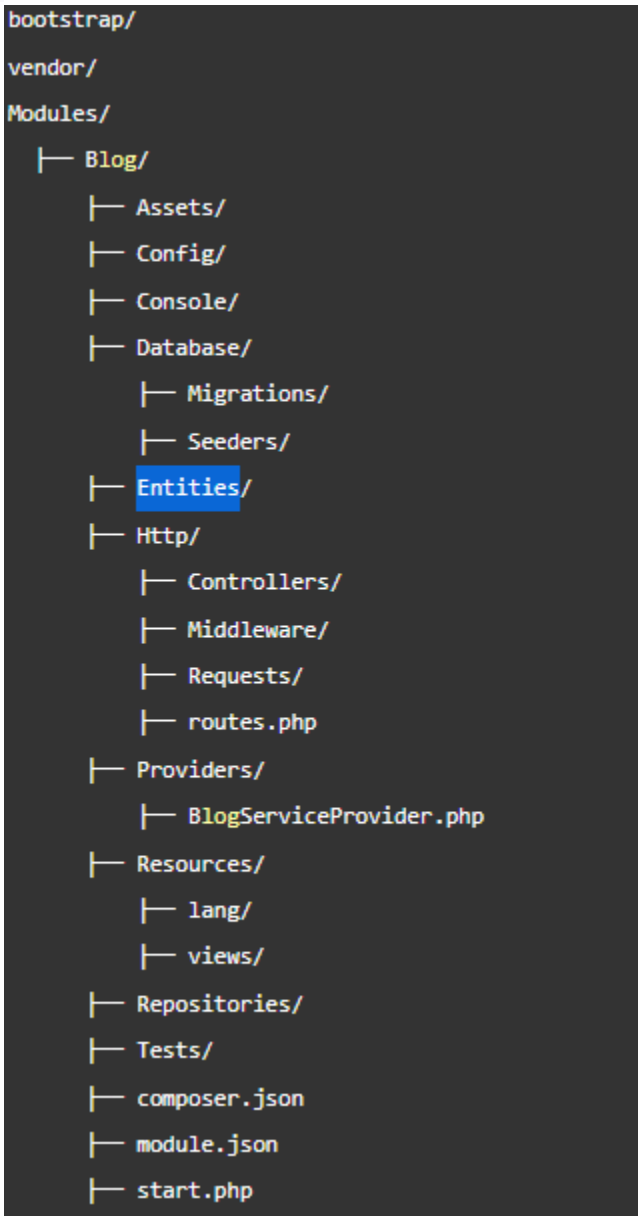
5.4.2 module:make-controller:

Generate a controller for the specified module.
php artisan module:make-controller <controller-name> <module-name>

5.4.3 module:make-model:

Generate a model for the specified module.
php artisan module:make-model <model-name> <module-name>
For further commands : Modules Command

5.5 Folder Structure:



NOTE:

Models will be created in Modules\<module-name>\Entities.
Please Do not create further subfolder(s) in Entities.
Right Approach: Modules\<module-name>\Entities\<model-file.php>.
Wrong Approach: Modules\<module-name>\Entities\Models\<model-file.php>