Web Programming with The Laravel Framework

#7 - The Architecture Concept of Laravel, Part-1

Steven Lolong

steven.lolong@uni-tuebingen.de

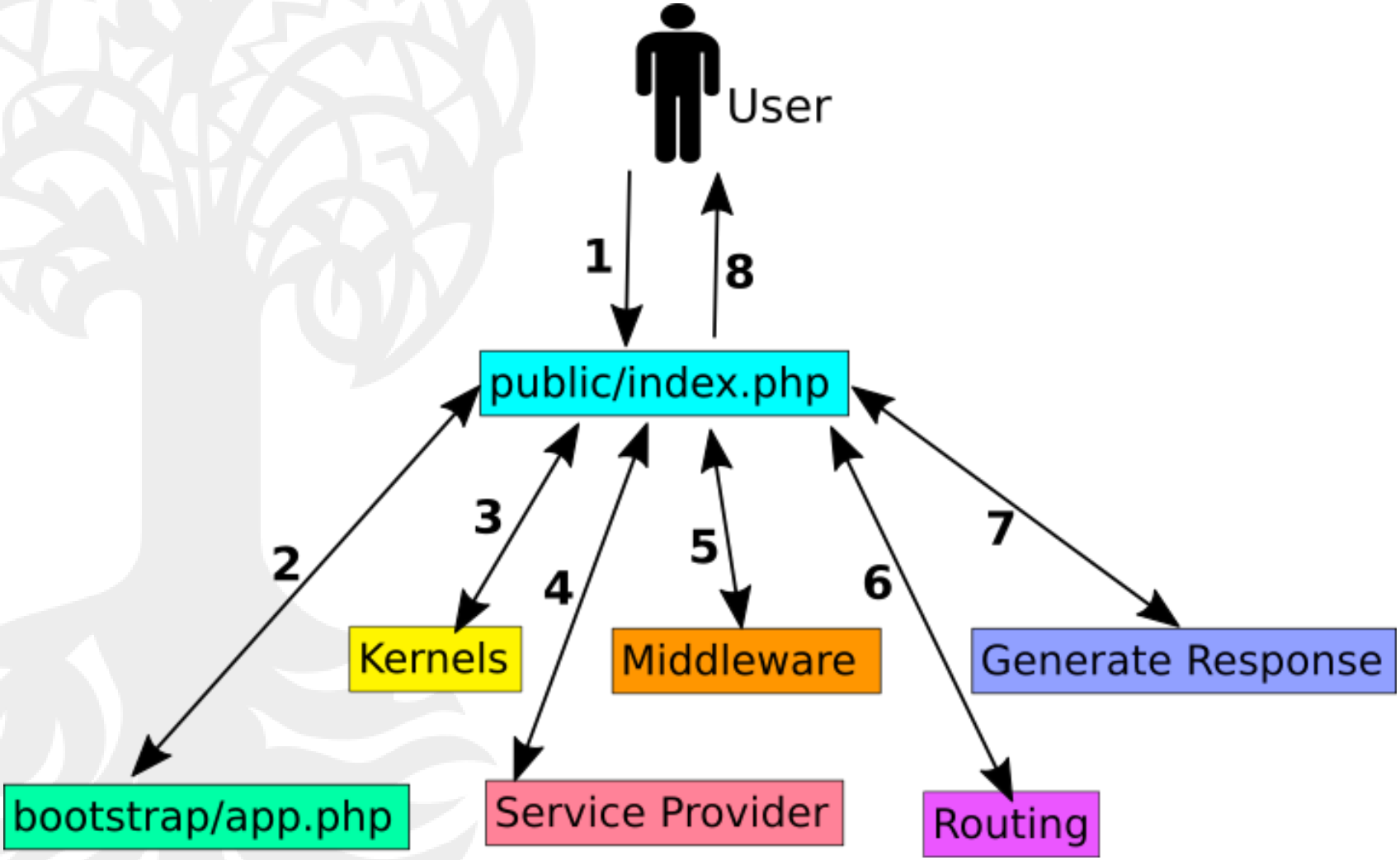
Programming Language Research Group
Tuebingen University

18 May 2022

Eberhard Karls
Universität
Tübingen

Summary

Request Lifecyle



Request Lifecyle

1. Request from user to *public/index.php*.
2. *index.php* loads *bootstrap/app.php* and create an instance of the application (service container).
3. Loading HTTP kernel (*app/Http/Kernel.php*) or Console kernel.
4. Kernel loads the service providers where the service providers for the application configuration are in the *config/app.php*.
5. Kernel defines a list of HTTP middleware
6. The service provider loads route files contained within the application's *routes* directory. The routing will map *address* to *actionhandler*.
7. Once the route of the controller method returns a response. The response will travel back outward through the route's middleware (optional).
8. The *index.php* file calls the *send* method on the returned response and sends the response content to the user's web browser.

Service Provider, Bootstrapped and Facades

Service Provider

Service providers are the central place of all Laravel application bootstrapping. Your application, as well as all of Laravel's core services, are bootstrapped via service providers.

Bootstrapped

In general, it means registering things, including registering service container bindings, event listeners, middleware, and even routes. Service providers are the central place to configure your application.

Facades

Laravel facades serve as "static proxies" to underlying classes in the service container, providing the benefit of a terse, expressive syntax while maintaining more testability and flexibility than traditional static methods. All of Laravel's facades are defined in the *Illuminate\Support\Facades* namespace.

Installation via Composer (Option-1)

Requirement

composer, composer is dependency Manager for PHP

Create Laravel Project

composer create-project laravel/laravel app-name

Installation via Composer (Option-2)

Install laravel as a global composer dependency

Download Laravel Installer

composer global require laravel/installer

Setting path in system

- macOS: $HOME/.composer/vendor/bin
- Windows: %USERPROFILE% \AppData \Roaming \Composer \vendor \bin
- GNU / Linux Distributions: $HOME/.config/composer/vendor/bin or $HOME/.composer/vendor/bin

Create Laravel Project

laravel new app-name

Kick off

Start the internal web server of Laravel. Jump into the application folder and run this command:

```
php artisan serve
```

Global Configuration

Global configuration file is *.env* in root folder

.env contains configuration of:

- Application name
- Environment
- Encryption key
- Debugging information
- Application address
- Logging
- Database connection
- Session driver
- Cache
- Queue
- etc.

Detail Configuration

Configuration files are in directory *config*:

- *app.php*, application configuration
- *auth.php*, authentication (if you use authentication package)
- *logging.php*, setting log
- *mail.php*, mail setting
- *session.php*, session setting
- *database.php*, database connection and driver
- *view.php*, to modify view directory
- *filesystem.php*, application driver for filesystem

Directory

- *config*, contains all files for configuration
- *app*, all files and folders of *controller*, *model*, *provider*, etc.
- *database*, contains seed and migration files for database.
- *public*, you can put all your files or folders that accessible by user. ex.: *.js, *.css, *.jpg, *.png, etc.
- *resources/views*, folder to put view file.
- *route*, containing your address that mapped to the function.
- The other folders are used by the Framework for its file.

Routing

Routing

- Routing in Laravel means creating your address and mapping it to a function (usually, a function in the controller class)
- Routing file for the web application is *route/web.php*
- Route format: Route::method(uri,callback)
  - Route is facades Route of Framework.
  - method is [get | post | put | patch | delete | options | any | match | redirect |].
  - uri is a uri address of application.
  - callback is function of application.

Get

Route::get(uri, callback)

```
use Illuminate\Support\Facades\Route;
Route::get('/say', function(){
  return "Halo!";
});
```

Listing: map to a function

```
use App\Http\Controllers\HomeController;
Route::get('/home', [HomeController::class, 'index']);
```

Listing: map to a class method

Match

Route::match([method], uri, callback)

```
1 Route::match(['get','post'], '/sayit', function(){
2   return "Hello!";
3 });
```

Listing: match

Any

Route::any(uri, callback)

```
1 Route::any('/usingany', function(){
2   return "Hello, this is from any";
3 });
```

Listing: any

Redirect

Route::redirect(from, to, 301)

or

Route::permanentRedirect(from, to)

```
1 Route::redirect('/nopage', '/usingany', 301);
```

Listing: redirect

View

Route::view(uri, view-file-name)

view-file-name is a view file in folder *resources/views/* with extention named *.blade.php*

```
1 Route::view("/welcome", "welcome");
```

Listing: view

Parameter

Route::get(uri{param}, function(param))

The parameter in address must be in the *curlybracket*

```
1 Route::get('/user/{id}/nm/{uname}', function($uid, $uname){
2     echo 'User id: ' . $uid . '<br/>';
3     echo 'Name: ' . $uname;
4 });
```

Listing: required parameter

Fallback

Route::fallback(callback)

Fallback will be executed when no other route matches the incoming request.

```
1 Route::fallback(function () {
2     return "address not found!";
3 });
```

Listing: fallback

Named Route

Route::...()->name('url-name')

```php
Route::any('/named', function(){
   $myurlName = route('withname');
   echo $myurlName;
})->name('withname');

Route::any('/urlnamed', function(){
     return redirect()->route('withname');
});
```

Listing: named route

Controller

Eberhard Karls
Universität
Tübingen

Create

Create controller file using **artisan** command
php artisan make:controller Controler-Name

Or

Create a controller file in the folder **app/Http/Controllers/**.
The file name must be the same as the controller class name.

Example

Create controller file using **artisan** command

php artisan make:controller Example

```
1 use Illuminate\Http\Request;
2
3 class Example extends Controller
4 {
5
6 }
```

Listing: Example controller class

Example

Add a function **hello()**

```
1 class Example extends Controller
2 {
3   public function hello(){
4         return "Hello, this is from controller";
5     }
6 }
```

Listing: function hello

Map Address to Controller in Routing

Route the address to a function in routing file (*routes/web.php*)

```
1 use App\Http\Controllers\Example;
2 Route::any('/conexample', [Example::class, 'hello']);
```

Listing: map the function

View

Create

- Create **a view file** in folder **resources/views/** or organise the file by group it into sub-folder under **resources/views/**.
- Use **.php** as file extention even the file is html file.
- Load view from controller **return view(filename, [variables])**.
- To access file in the sub-folder with **.** or **/**
- Passing data to views: **return view (subfoldername.filename), [data]**

Example - Load from Routing File

Create a file in views: **say.blade.php**

```
1 <html>
2   <body>
3     <h1>Halo, {{ $usrName }} !</h1>
4   </body>
5 </html>
```

Listing: view say

Add a routing in: **routes/web.php**

```
1 Route::get("/sayview", function(){
2     return view('say', ['usrName' => 'Steven']);
3 });
```

Listing: load view with data

Eberhard Karls Universität Tübingen

Example - Load from Controller

Create a file in **views/front/**: **say.blade.php**

```
<html>
  <body>
    <h1>Halo, {{ $usrName }} !</h1>
  </body>
</html>
```

Listing: view say

Add a function in controller **Example.php**

```
  public function say(){
      return view('front.say', ['usrName' => 'Klaus']);
  }
```

Listing: function say in controller

Add a routing in: **routes/web.php**

```
Route::any('/sayexample', [Example::class, 'say']);
```

Listing: route the address

Eberhard Karls Universität Tübingen