بسم رب الحسین(ع)

تمرین ۱۷ محمدعرفانی

۱.دانسته‌های خود را از لاراول بیان کنید و توضیح دهید پوشه های آن چه کاربردی دارند.

لاراول یک فریمورک php است که با معماری mvc طراحی شده است. Template engine استفاده شده در این فریمورک blade است.

ساختار :

پوشه app شامل core برنامه است. این فولدر شامل console,exceptions,http,models, providers است. console شامل تمام دستورات artisan است.exceptions حاوی تمام exception handler هاست. پوشه http شامل تمامی controller ها – middleware ها است. فولدر models شامل تمامی elequent model هایمان میباشد. Providers نیز شامل تمامی service provider هایمان میباشد. در فولدر بعدی bootstrap است .

The `bootstrap` directory contains the `app.php` file which bootstraps the framework. This directory also houses a `cache` directory which contains framework generated files for performance optimization such as the route and services cache files. You should not typically need to modify any files within this directory.

فولدر بعدی config شامل تمامی configuration هایمان میباشد.

فولدر database نیز شامل migration ها و model factories هایمان میباشد.

Lang مکانی است که زبان هایی که در برنامه استفاده میکنیم را قرار میدهیم.

Public شامل فایل index.php است که حکم ورودی برای درخواست هایمان و لودکردن فایل هایمان را دارد. این پوشه همچنین جاییست که در آن فایل های تصویر ،css و جاوااسکریپ در آن قرار میگیرد. همچنین شامل یک فایل htaccess نیز میباشد که برای دستورات اجرایی سرور است.

Resources: در این فولدر view هایمان قرار دارد. همچنین javascript , css هایی قرار دارد که بعدا میتوان کامپایل شوند و در پوشه public قرار گیرد.

Routes شامل تمامی route هایمان است.

storage :

The `storage` directory contains your logs, compiled Blade templates, file based sessions, file caches, and other files generated by the framework. This directory is segregated into `app`, `framework`, and `logs` directories. The `app` directory may be used to store any files

generated by your application. The `framework` directory is used to store framework generated files and caches. Finally, the `logs` directory contains your application's log files.

Test: شامل فایل های تست ما می‌باشد.

Vendor : نیز شامل نیازمندی های کامپوزر است.

۲. Artisan در لاراول چیست؟

Artisan اسم CLI لاراول است که شامل چندین دستور کاربردی برای استفاده در محیط CLI میباشد. برای استفاده از این دستورات باید در محیط CLI در پوشه اصلی باشیم. سپس دستوری که اول آن php artisan است را میدهیم. به طور مثال php artisan serve که یک سرور مجازی برای تست کردن کد به ما میدهد. یا با استفاده از php artisan make:controller controllerName یک کنترلر آماده ساخته می‌شود که کار مارا بسیار راحت میکند.با استفاده از دستور زیر میتوان تمامی دستورات را مشاهده کرد.

Php artisan list

۳.templating engine چیست، معروف‌ترین‌هارا نام ببرید، لاراول از کدام استفاده میکند؟

Template engine ها برای انتقال داده به صورت HTML با یک قالب مشخص استفاده میشود. این درصورتی است که

Template engines are mostly used for server-side applications that are run on only one server and are not built as APIs. The popular ones include Ejs, Jade, Pug, Mustache, HandlebarsJS,jinja2, and Blade.

در لاراول از Blade استفاده میشود.

Blade is the simple, yet powerful templating engine that is included with Laravel. Unlike some PHP templating engines, Blade does not restrict you from using plain PHP code in your templates. In fact, all Blade templates are compiled into plain PHP code and cached until they are modified, meaning Blade adds essentially zero overhead to your application. Blade template files use the `.blade.php` file extension and are typically stored in the `resources/views` directory.

۴. چرخه حیات درخواست در لاراول را توضیح دهید.

تمامی درخواست ها از فایل index.php که در پوشه public قرار دارد شروع میشود. در این فایل یک شی از app.php که در پوشه بوت‌استرپ قرار دارد ساخته میشود.سپس لاراول یک service container میسازد. در مرحله بعد به سمت درخواست به سمت کرنل ارسال میشود. این مرحله این وظایف را شامل میشود: configure error handling, configure logging ,detect the application environment.

کرنلی که در پوشه HTTP قرار دارد همچنین لیستی از میدلور هایی که درخواست باید از آنها عبور کنند. این میدلورها خواندن و نوشتن های HTTP SESSION ها را کنترل میکند. پس از آن کرنل service provider ها را فراخوانی میکند. RouteServiceProvider روت هایی که در پوشه routes هست را فراخوانی میکند. زمانی که درخواست به روتر رسید به سمت یک روت مشخص یا کنترلر ارسال میشود. در این بین درخواست باید از میدلور های مخصوص آن روتر نیز بگذرد و اگر گذشت و به کنترلر رسید ، یک response برگردانده میشود. در متود send داخل index.php این پاسخ به سمت وب سروری که درخواست را ارسال کرده بود میفرستد.

۵. روت‌های Resource چه کاربردی دارند؟

ما میتوانیم به وسیله دستور زیر یک resource controller بسازیم.

php artisan make:controller carController –resource

برای دسترسی به این کنترلر میتوان روت را به صورت زیر تعریف کنیم.

```
Route::resource('cars', CarsController::class);
```

با دستور زیر میتوانیم به تمامی روت هایی که به آن دسترسی داریم را ببینیم.

php artisan route:list

در کنترلر به متود های create -store- show -edit-update – delete دسترسی داریم که میتوانیم آن‌ها را به صورت دلخواه گسترش دهیم و استفاده کنیم.

۶. روت‌های Named چه کاربردی دارند؟

It allows you to refer to the routes when generating URLs or redirects to the specific routes. In short, we can say that the naming route is the way of providing a nickname to the route.

در انتهای route تعریف شده میتوان به وسیله تابع name به روت اسم دهیم.

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

از این به بعد میتوانیم در یک روت دیگر این روت را فراخوانی کنیم.

```php
Route::get('/',function (){
   return redirect()->route('home');
});
```

۷. تفاوت {!! !!} و {{ }} در Blade چیست؟

By default, Blade `{{ }}` statements are automatically sent through PHP's `htmlspecialchars` function to prevent XSS attacks. If you do not want your data to be escaped, you may use the following syntax: {! !}

به طور مثال:

```php
$first = "<b>Narendra Sisodia</b>";
```

And it is accessed, within Blade, with {{ $first }} then the output'll be:
Narendra Sisodia
But if it is accessed with {!! $first !!} then the output'll be:
Narendra Sisodia

۸. آدرس {cat}/{id}/{title}/ را به صورت زیر درنظر بگیرید

روت را به این صورت تعریف میکنیم

```php
Route::get('/{title}/{id}/{cat}',function ($title,$id,$cat){
    return view("$cat.index",['title'=>$title,'id'=>$id,'cat'=>$cat]);
})->whereAlpha('title')->whereNumber('id')->whereIn('cat',['sport','economy','values']);
```

در پوشه resources/views فولدرهای sport, economy , values قرار دارد که در هر کدام فایل index.blade.php قرار میگیرد. در هر فایل نیز میتوان به این صورت نتیجه را نشان دهیم.

```php
{{ "title=> ".$title.", id=>".$id.", cat=> ".$cat }}
```