**بسم الله الرحمن الرحیم**

**تمرین هفتگی 16 – مکتب شریف دروه 70 php  
تهیه کننده : حسین مومنی**

1. **دانسته های خود را از لاراول بیان کنید و توضیح دهید پوشه های آن چه کاربردی دارد.**

**مسیر ریشه (root) پروژه شامل مسیرهای زیر میشه:**

**app**

**bootstrap**

**config**

**database**

**public**

**resources**

**routes**

**storage**

**tests**

**vendor**

**App (/app)**

**این مسیر شامل کدهای هسته برنامه‌ای هست که قراره بنویسیم. اکثر کلاس‌ها و فایل‌های برنامه ما توی این مسیر قرار میگیره. این مسیر خودش شامل چند مسیر دیگه میشه که توی ادامه اونها رو بررسی میکنیم.**

**Bootstrap (/bootstrap)**

**این مسیر شامل یک فایل هست به اسم app.php. وظیفه این فایل اینه که فریمورک رو راه‌اندازی کنه. این مسیر همچنین یک پوشه‌ای داره به اسم cache که شامل فایل‌هایی هست که فریمورک برای افزایش سرعت تولید میکنه. مثل فایل‌های کش تولید شده برای Routeها و سرویس‌ها.**

**Config (/config)**

**این مسیر همونطور که از اسمش مشخصه، شامل همه فایل‌های پیکربندی (configuration) برنامه هست. همه‌ی تنظیمات مربوط به برنامه‌ی ما توی این پوشه قرار میگیره. مثلا تنظیمات مربوط به دیتابیس، ایمیل و ...**

**Database (/database)**

**این مسیر شامل فایل‌های Seed, Migration و Model Factory پروژه هست. در فولدر Migration فایل هایی که مربوط به تولید جداول مورد نیاز دیتا بیس است تولید می شود که در دو متد up و down تولید شده اند که برای ایجاد جدول و نابود کردن جدول به کار می رود.**

**Public (/public)**

**این مسیر یک فایل داره به اسم index.php که یک ورودی برای همه درخواست‌هایی هست که به برنامه میاد. همچنین autoloading برنامه هم بر عهده این فایل هست. این مسیر همونطور که از اسمش پیدا هست، مسیر عمومی برنامه هست. پس میتونه شامل تصاویر، فایل‌های CSS و Javascript باشه. فی الواقع تمامی فایل‌های CSS و Javascript که از پوشه اصلی در resource خوانده می شوند در این فایل رندر و کامپایل می شوند که کاربر به آن ها دسترسی داشته باشد.**

**Resources (/resources)**

**این مسیر شامل فایل‌های view هست. همچنین شامل فایل‌های خام و کامپایل نشده LESS و SASS و جاوا اسکریپت هست. فایل‌های مربوط به زبان هم توی این مسیر هست (fa, en, fr , ...)**

**Routes (/routes)**

**ما توی این پوشه، تمام مسیرهایی که برای برنامه‌مون درنظر گرفتیم رو مشخص میکنیم. منظور از مسیر همون آدرس‌هایی هست که کاربر توی مرورگرش وارد میکنه. این مسیرها رو اینجا مشخص میکنیم.**

**بطور پیشفرض این پوشه شامل 4 تا فایل هست: channels.php - console.php - api.php - web.php**

**همه مسیرهایی که توی فایل web.php تعریف میکنیم، بطور پیشفرض شامل یک سری میدلویر هایی هستن. مثل Session، CSRF Protection، Cookie Encryption.**

**اگه پروژه شما Stateless یا RESTful API نیست، اکثر مسیرهایی که وجود داره توی همین فایل قرار میگیره.**

**از فایل api.php زمانی استفاده میکنیم که پروژه ما Stateless باشه. یعنی برای مثال برای اینکه کاربر رو احراز هویت کنیم باید توی هر درخواست، رشته مربوط به احراز هویت (Token) وجود داشته باشه. پس از سشن و کوکی توی این قسمت پروژه خبری نیست.**

**اگه بخوایم از طریق Console یک سری کارهایی رو توی برنامه انجام بدیم، از فایل console.php استفاده میکنیم.**

**Storage (/storage)**

**این مسیر برای ذخیره و نگهداری فایل‌های فریمورک و برنامه هست. این مسیر سه تا پوشه داره. app، framework و logs.**

**پوشه app مکان خوبیه که فایل‌های برنامه‌مون رو ذخیره کنیم.**

**پوشه framework شامل فایل‌هایی میشه که خود فریمورک تولید میکنه. مثل فایل‌های کامپایل شده Blade، فایل‌های سشن، فایل‌های کش و یک سری فایل دیگه که توسط فریمورک تولید میشه.**

**پوشه logs که همونطور که از اسمش پیداست، برای ذخیره گزارش‌های برنامه‌ی ما استفاده میشه.**

**پوشه /storage/app/public مکان خوبیه که فایل‌های کاربر رو ذخیره کنیم. مثلا آواتار و فایل‌هایی که لازمه بصورت عمومی در دسترس باشن. البته برای دسترسی عمومی به این فایل‌ها باید symbolic link درست کنیم که به این مسیر اشاره کنه. با استفاده از دستور زیر توی خط فرمان میتونیم این symbolic link رو بسازیم:**

**php artisan storage:link**

**Tests (/tests)**

**این مسیر هم همونطور که از اسمش مشخصه شامل تست‌های برنامه ما هست.**

**Vendor (/vendor)**

**پکیج‌هایی که برای برنامه نیاز داریم، توسط composer توی این پوشه ذخیره و استفاده میشه.**

1. **Artisan در لاراول چیست؟**

**به طور خلاصه می توان گفت که Artisan در واقع کمکیار کاربر برای اجرا و کار با تمامی ابعاد لاراول است.**

**Artisan یک ابزار خط فرمان در فریم‌ورک لاراول است.**

**این ابزار وظیفه دارد تا انجام تغییرات در فریم‌ورک را ساده کند. فراخوانی این ابزار با استفاده از فایل اجرایی php به انجام می‌رسد**

**برخی از کارهایی که Artisan انجام می‌دهد عبارتند از:**

**فعال و غیر فعال کردن سایت**

**بهینه‌سازی فریم‌ورک برای عملکرد بهتر**

**به‌روزرسانی پایگاه داده با آخرین تغییرات**

**ایجاد کلید امنیتی برنامه که در کوکی‌ها و سشن‌ها استفاده می‌شود**

1. **Templating Engine چیست؟معروف ترین ها را نام ببرید و لاراول از کدامیک استفاده می کند؟**

**کار جایگزینی مقادیر در یک template توسط «موتور قالب» یا template engine صورت می‌گیرد. تمپلت‌انجینی برای ما مطلوب است که ما را از نوشتن کد و تگ html در غیر از فایل template بی‌نیاز کند. لیست معروف ترین آن ها به شرح ذیل است:**

**Foil  - SavantPHP - TBS  - TPLN – Blade که لاراول از موتور Blade استفاده کرده است در ساختار خود.**

1. **چرخه حیات درخواست در لاراول را توضیح دهید؟**

**مرحله اول : Auto Loader**

**به عنوان اولین مرحله ، درخواست از مرورگر کاربر شروع می شود ، سپس به سرور وب می رسد. سرور وب (Apache یا Nginx) درخواست داده شده را به فایلی در مسیر Laravel public / index.php هدایت می کند که یک نقطه شروع برای بارگذاری بقیه فریمورک است. این فایل autoloader ای که توسط composer ساخته شده است را لود میکند.**

**سپس با لود شدن autolader ، نمونه ای از برنامه Laravel از اسکریپت bootstrap / app.php بازیابی میشود. یعنی در اولین مرحله ، لاراول خودش نمونه ای از برنامه را ایجاد می کند. بطور مثال بوت استرپ را در این مرحله میشود اینگونه تشبیه کرد که وقتی بعنوان راننده سوار میشید، اولین مرحله چک کردن آینه ها و بستن کمربند و .... میباشد.**

**مرحله دوم: Kernel**

**مرحله بعدی در قسمت هسته برنامه( KERNEL) اعمال خواهد شد.**

**بسته به نوع درخواستی که وارد برنامه می شود ، درخواست دریافتی به HTTP kernel یا console kernel ارسال می شود.**

**این دو نوع KERNEL به عنوان مکان اصلی برای اجرای همه درخواست ها میباشند.**

**هسته یا کرنل HTTP ، در برنامه Http / Kernel.php قرار داده شده است. این kernel فقط یک درخواست دریافت می کند و یک پاسخ ارسال می کند. بوت استرپرهایی که توسط کلاس Kernel تعریف شده اند ، مسئولیت رسیدگی به خطا ، پیکربندی ورود به سیستم (configure logging) ، مسیریابی های تعریف شده در فایل env و سایر کارهایی را که باید قبل از انجام درخواست ارسالی انجام شود را برعهده دارد .**

**کرنل HTTP همچنین لیستی از middleware هایی که قبل از استفاده از برنامه باید مورد توجه قرار بگیرد را نیز تعریف خواهد کرد.**

**مرحله سوم: Service Providers**

**مرحله بعداز گذشتن از kernelها، لود کردن ارائه دهندگان خدمات (Service Providers) هستند که بخشی از عملکرد های bootstrapping میباشند. Service Providers مورد نیاز برای برنامه در فایل config/app.php قرار دارند.**

**همینطور که متدهای تعریف شده صدا زده میشوند، تمام providers ثبت می شوند. پس از ثبت و تعریف همه ی providers ، روش های بوت providers ها صدا زده میشود. providers ها روش های بت متفاوتی دارند که در با استفاده از آنها صدا زده میشود که در مقالات آتی به آنها میپردازم.**

**مرحله چهارم: Dispatch Request**

**پس از راه اندازی برنامه و ثبت و راه اندازی همه Service Providers، درخواست برای ارسال به روتر تحویل داده می شود. که همان web.php یا app.php میباشد. سپس روتر درخواست را به یک مسیر یا کنترلر مدنظر ارسال می کند ، و همچنین در صورت نیاز هر middleware خاصی را برای مسیر اجرا می کند.**

**مرحله پنجم: Router**

**اکنون درخواست توسط روتر به کنترلر یا مسیر موردنظر منتقل می شود و ریکوئست ارسالی مطابق روند زیر در پایان قابل مشاهده خواهدبود:**

**روتر درخواست های HTTP را به کنترلر هدایت می کند یا با حذف کنترلر ، نمایش یا پاسخ ها را مستقیماً باز می گرداند. این مسیرها در فایل app/routes.php قرار دارند.کنترلر ها که در مسیر app/controllers/ قرار دارند، اقدامات خاصی را انجام داده و داده ها را به یک view ارسال می کند.**

**ویوها که در مسیر app/views/ قرار دارد، قالب داده ها را بصورت مناسبی جهت پاسخ در HTTP آماده میکند و ارائه میدهد.**

1. **روت های Resource چه کاربردی دارند؟**

**Route های resource کلا متدهای restful رو در خودشون جا دادند که شامل متدهای store, index, show, update, destroy, edit, create میشه.**

1. **روت های Named چه کاربردی دارند؟**

**با استفاده از این روت، این امکان وجود دارد که URL های مختلف را نام گزاری کرده و به راحتی با نام آن ها مثلا redirect کرد از هر جای برنامه که خواستیم و کد کمتری بزنیم و خطا به حداقل می رسد. مثل مثال زیر :**

Route::get(

'/user/profile',

[UserProfileController::class, 'show']

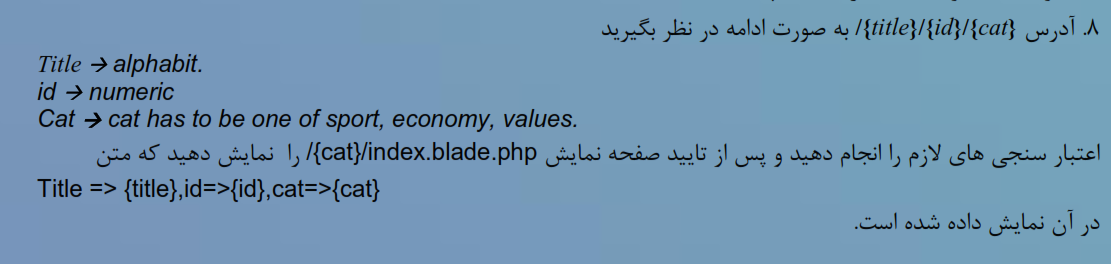
)->name('profile');

return redirect()->route('profile');

1. **تفاوت {!! !!} و {{}} در Blade چیست؟**

**در مورد {{}} به راحتی در محیط HTML به جای اینکه کد های php رو با باز کردن <?php ?> بنویسیم، با این علامت به راحتی کد های php رو داخلش می نویسیم. ولی اگر خواستیم در داخل کدهای php هم کدهای html بنویسیم نیاز است تا از علامت دوم استفاده کنیم که تک های html رو هم تفسیر کنه و چاپ نکنه به طور مستقیم.**

1. **به سوال زیر پاسخ دهید؟**

**کد درخواستی به شرح ذیل است:**

**Web.php**

Route::get('/{title}/{id}/{cat}', function ($title, $id, $cat) {

    compact("title","id","cat");

    return view("content",compact("title","id","cat"));

})->whereNumber('id')->whereAlpha('title')->whereIn('cat', ['sport', 'economy', 'values']);

**Index.blade.php**

<div>

    @yield('content')

</div>

**Content.blade.php**

@extends("$cat.index")

@section('content')

<div>

    <p>title</p>

    {{$title}}

    <p>id</p>

    {{$id}}

    <p>cat</p>

    {{$cat}}

</div>

@endsection