مستندات پروژه فروشگاه دانلود فایل

**تاریخ 1 فروردین 99:**

1. لوگو به سمت راست بالای سایت اضافه شد.
2. روت تماس با ما درست شد.

**تاریخ 2 فروردین 99:**

1. کنترلر ContactsController با دستور

php artisan make:controller ContactsController ایجاد شد و متد index برای نمایش صفحه اصلی ارتباط با ما درست شد.

1. برای بخش ارتباط با ما این جوری در نظر گرفتم که هرکسی بتواند بدون نیاز به رجیستر شدن در سایت، در بخش ارتباط با ما، پیام خود را ارسال نماید. بنابراین لازم است یک جدول جداگانه برای بخش contact us در نظر گرفته شود.

طرح اولم این بود که پیام های بخش contact us را هم در جدول comments ذخیره کنم. اما در این صورت نیاز است که user\_id هم داشته باشیم که نیاز به رجیستر و لاگین بودن کاربر است. پس یک جدول جدا برای "ارتباط با ما" در نظر گرفتم.

فیلدهای جدول contacts :

Id

First\_name

Last\_name

Email

Phone

Message

است که فیلد message رو از نوع text در نظر میگیریم.

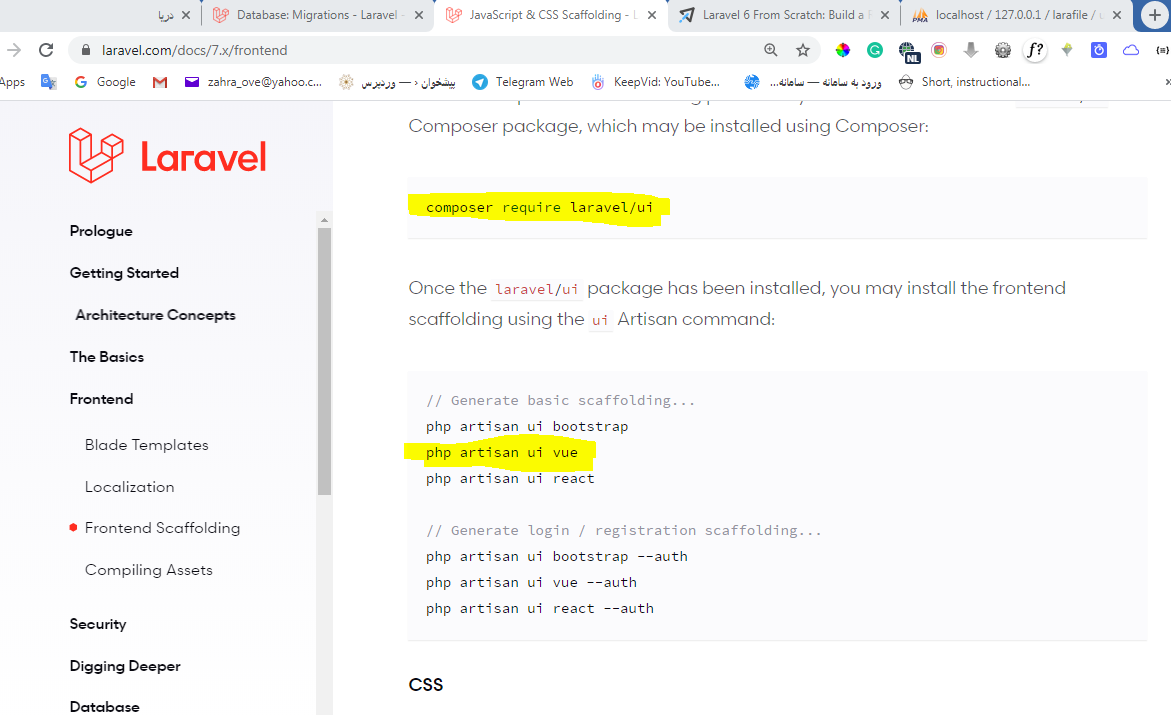
با دستور php artisan make:model Contact –m هم فایل مدل و هم فایل migration را میسازیم.

مدل Contact قرار نیست با مدل دیگری در ارتباط باشد.

در نهایت تابع store رو در ContactsController نوشتم و پیامهای کاربران و بازدیدکنندگان از این به بعد در جدول contacts ذخیره خواهد شد.

**3 فروردین 99:**

بخش auth را با وارد کردن دو دستور زیر به پروژ اضافه کردم



بعضی ها موقع زدن دستور composer require laravel/ui یک --dev هم به آخر این دستور اضافه میکنند. استاد جنتی گفتند دلیل اضافه کردن این –dev این است که آخرین ورژن موجود را بگیرد حتی اگر آن ورژن احیانا با یا مشکلی داشته باشد ولی آخرین وژن موجود برات نصب میشه.

حالا قبلا دستور migrate را زده بودم و تمذم جدوال users و ... درست شده بود. وقتی auth را هم اضافه میکنم، یک فایل مایگریشن به نام create\_password\_reset\_table.php ایجاد می­شود.

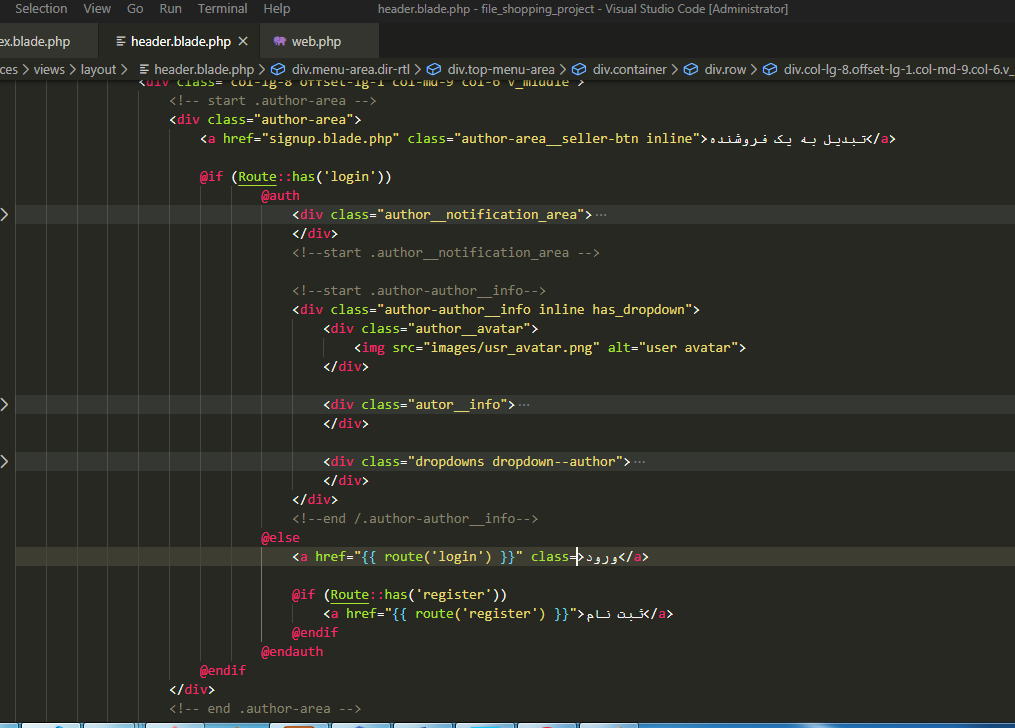
حالا چون من میخوام جدول users رو کامل تر کنم پس یک php artisan migrate:rollback میزنم و حالا چند ستون به فایل مایگریشن users اضافه میکنم به شکل زیر و دوباره دستور

php artisan migrate را میزنم تا جدول ها مجددا ساخته شوند.



* دو دکمه "ورود" و "ثبت نام" به منوی سایت اضافه شد. اگر کاربر در سایت لاگین کرده بود آنگاه دو دکمه ورود و ثبت نام بهش نشون داده نمیشه و به جاش نام و سایر مشخصاتش نمایش داده میشه. اگر لاگین نکرده باشه که دو دکمه ورود و ثبت نام به کاربر نمایش داده میشه.

کدش رو تو در بخش header.blade.php به این شکل اضافه کردم:



خب حالا که دو دکمه ثبت نام و ورود به صفحه اول اضافه شد باید لینکشون رو درست کنم. از اونجایی که تمی که خریدیم خودش صفحه ثبت نام و ورود رو داره پس از همون ها استفاده میکنم و دو صفحه رو در مسیر view قرار دادم و از خود صفحاتی که سیستم auth در پوشه auth در پوشه view در پوشه resources داره استفاده نکردم بلکه در پوشه views دو فایل به نام register.blade.php برای ثبت نام و یک فایل دیگر به نام login.blade.php برای لاگین ایجاد کردم.

حالا از کنترلرهای خود سیستم auth استفاده میکنم ولی باید یه کم تغییرشون بدیم. چرا و چگونه؟

در RegisterController.php که در پوشه auth از پوشه controller از پوشه http از پوشه app قرار دارد یک trait داریم به نام RegistersUsers که در این trait یک متد به نام showRegistrationForm وجود دارد که این خط را بر میگرداند:

return view('auth.register');

چون میخواد از view های خود سیستم auth استفاده کند. من خط زیر را به شکل اینجوزی تغییرش دادم:

return view('register');

پس میره در پوشه views و فایل register.blade.php را میخواند یعنی همون فایلی که در مرحله قبل خودمون ساختیم.

بخش loginController.php هم همین روال رو باید براش پیش بریم یعنی در این کنترلر یک trait داریم به نام AuthenticatesUsers و واردش میشیم و یک متد هست به نام showLoginForm که مسیرش به صورت پیش فرض به این شکل است:

        return view('auth.login');

و با به صورت زیر مینویسیسمش:

        return view('login');

تا فایلی که خودمون برای صفحه لاگین در پوشه views ساختیم را بخواند.

این تا اینجا

**4 فرودین 99:**

* برای بخش ثبت نام وقتی کاربر اطلاعاتش رو وارد میکرد یک ارور میگرفتم با این مفهوم که فیلد username مقدار default نداره. این به خاطر اینکه در مدل User باید تمام فیلدهای حدول users رو در متغیر $fillable تعیریف کنم. وقتی این کار رو کردم همه چیز حل شد.

حالا بعدش رفتم سراغ بخش redirect

یعنی کاربر معمولی بعد از لاگین یا رجیستر به پروفایلش هدایت بشه و ادمین به پنل مدیریت هدایت بشه.

قبل از این مرحله باید یه کاری کنم که هرکسی در سایت رجیستر کرد به صورت پیش فرض، نقش user معمولی بهش تخصیص داده شود.چون در مرحله قبل میخواستم بر اساس نقش هرکاربر، بخش redirectTo رو هندل کنم.

اول مدل Role رو به همراه فایل migration میسازیم با این دستور:

**php artisan make:model Role -m**

و بعد وارد فایل مایگریشنش میشیم و نام ستون های جدول roles رو اضافه میکنیم به این شکل:

    public *function* up()

    {

*Schema*::create('roles', *function* (*Blueprint* $table) {

            $table->id();

            $table->string('name');

            $table->string('description')->nullable();

            $table->timestamps();

        });

    }

حالا دستور زیر را میزنیم تا مایگریشن انجام شود و جدول roles ساخته شود:

**php artisan migrate**

یک اشتباه کردم اونم اینکه مدل Role را باید در پوشه Models میساختم که درستش کردم. چه طور؟ فایل مدل را move کردم به پوشه Models و فقط namespace را تغییر دادم به

**namespace App\Models;**

خب حالا بین جدول users و جدول roles ارتباط چند به چند وجود دارد. یعنی هر کاربر میتواند چند نقش داشته باشد و هر نقش میتواند متعلق به چند کاربر باشد. پس در مدل ها این روابط رو وارد میکنم. در مدل User این تابع را مینویسیم:

        # many to many relationship between users and roles table

        public *function* roles()

        {

            return $this->belongsToMany('App\Models\Role');

        }

و در مدل Role هم تابع زیر را:

    # many to many relationship between users and roles table

    public *function* users()

    {

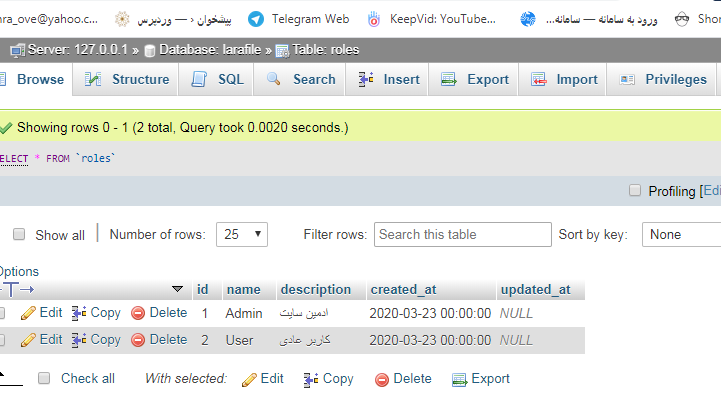
        return $this->belongsToMany('App\User');

    }

دقت کن که مدل User در پوشه app قرار دارد ولی مدل Role در پوشه app 🡺 Models قرار دارد. این را در توابع بالا باید درست وارد کرد.

تا اینجا ارتباط جدول کاربران و نقش ها ایجاد شد که رابطه چند به چند است.

حالا بریم سراغ تعریف نقش ها در جدول roles:



حالا باید کاری کنم هرکسی ثبت نام کرد به صورت خودکار نقش User را داشته باشد یعنی نقش کاربر معمولی.

اول از همه با استفاده از migration، جدول میانی role\_user را میسازیم با دستور زیر:

php artisan make:migration create\_role\_user\_table --create=role\_user

و کد زیر را درش مینویسیم:

    public *function* up()

    {

*Schema*::create('role\_user', *function* (*Blueprint* $table) {

            $table->id();

            $table->foreignId('role\_id')

                  ->constrained()

                  ->onUpdate('cascade');

            $table->foreignId('user\_id')

                  ->constrained()

                  ->onUpdate('cascade');

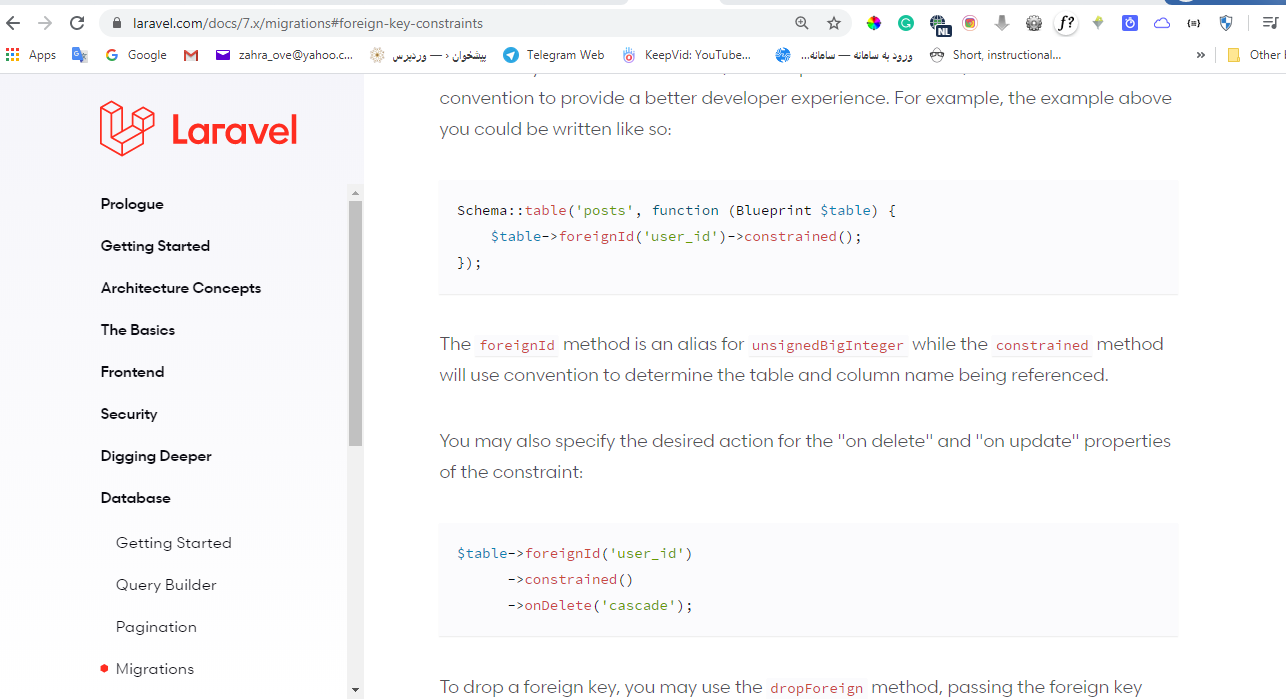
            $table->timestamps();

        });

    }

دقت کردی که در لاراول 7 چقدر خوشگل میشه foreign key رو مشخص کرد؟ ☺

توضیحات خود لاراول:



* قبل از رفتن به مرحله بعد، یک مدل و فایل مایگریشن برای genders درست کردم و در جدول users، ستون gender را به gender\_id تغییر دادم و foreigh key شد به ستون id از جدول users.

حالا که دو نقش اصلی رو تعریف کردیم میریم سراغ بخش redirectTo بر اساس نقش هر کاربر.

**5 فرودین 99:**

من یک تغییری دادم در جدول genders. از اونجایی که ستون gender\_id در جدول users به عنوان foreign key تعریف شده و من میخواستم این ستون nullable باشد ولی موفق نشدم بفهمم چه طور میشه یک ستون از نوع foreign key را nullable کرد پس اومد علاوه بر woman و man یک فیلد به نام none با آیدی 3 در نظر گرفتم که اگر کاربر مقداری را برای این فیلد در نظر نگرفت آنگاه مقدار 3 به سمت دیتابیس فرستاده میشود که به معنای جنیست تعیین نشده است.

خب این تا اینجا.

امروز فیلمهای ACL، guard، policy رو نگاه کردم.

---------------------------------------------------------------------------

**6 فرودین 99:**

خب بحث redirect کاربر بر اساس نقش انجام شد. در ادامه کامل ترش میکنم و ممکن هست روت هاش عوض بشه.

در حالت عادی پس از رجیستر یا لاگین شدن کاربر به صفحه home هدایت میشود. من میخوام کاری کنم که اگر کاربر وارد شده ادمین باشد به داشبورد ادمین هدایت شود و اگر کاربر معمولی باشد به پروفایل خودش هدایت شود.

برای این کار در دو فایل RegisterController.php و LoginController.php خط زیر را کامنت میکنم:

    // protected $redirectTo = RouteServiceProvider::HOME;

و به جاش متد زیر را اضافه کن به دو فایل مذکور:

    public  *function* redirectTo()

    {

        //get authenticated user object

        $user = *Auth*::user();

        //if user has Admin role then redirect to admin panel otherwise redirect to profile page

        if($user->isAdmin())

        {

            return route('admin.dashboard');

        }

        else

        {

            return route('user.dashboard');

        }

    }

این روش در داکیومنت های خود لاراول هم آمده ولی الان نتونستم پیدا کنم

نکته مهمی که میخوام بگم این هست که موقع return کردن در متد redirectTo حتما چیزی که return میشه یا باید از نوع string باشه یا باید از تابع route استفاده کنی و تابع redirect اصلا جواب نمیده !!!

ببین برای پروژه جدید که عکسش رو گذاشتم چه طوری هندلش کدم. واسه پروژه قدیمی که فروشگاه عادی بود هم این مدلی کد زدم.

    public *function* redirectTo()

    {

        //get authenticated user object

        $user = *Auth*::user();

        //if user has Admin role then redirect to admin panel otherwise redirect to profile page

        if($user->hasRole('Admin'))

        {

            return '/admin';

        }

        else

        {

            return '/profile';

        }

    }

میبینی که چیزی که در هر حالت return شده از نوع string بوده. که در واقع url ها هستن که نوشتیم بدون استفاده از تابع reditect.

فقط یک مساله نهایی مونده و اون هم این هست که کاربری که لاگین کرده اگر دوباره بخواد به صفحه رجیستر دسترسی پیدا کند به صفحه home هدایت میشود. که این به دلیل این است که در فایل RedirectIfAuthenticated.php که در پوشه middleware قرار دارد این تابع نوشته شده است:

    public *function* handle($request, *Closure* $next, $guard = null)

    {

        if (*Auth*::guard($guard)->check()) {

            return redirect(*RouteServiceProvider*::HOME);

        }

        return $next($request);

    }

ما در فایل RouteServiceProvider.php که در پوشه providers قرار دارد این خط را اضافه میکنیم:

public const ROOT = '/';

و سپس در فایل RedirectIfAuthenticated.php تابع ذکر شده را به شکل زیر تغییر میدهیم:

    public *function* handle($request, *Closure* $next, $guard = null)

    {

        if (*Auth*::guard($guard)->check()) {

            // return redirect(RouteServiceProvider::HOME);

            return redirect(*RouteServiceProvider*::ROOT);

        }

        return $next($request);

    }

یعنی هرگاه کاربر لاگین شده قصد دسترسی به صفحه رجیستر را داشت اون رو به صفحه اصلی سایت هدایت کن و اجازه دسترسی به صفحه رجیستر را بهش نده. این اتفاق وقتی کاربر لاگین شده بخواد به صفحه لاگین هم دسترسی پیدا کند اتفاق می افتد یعنی به صورت خوردکار به صفحه اصلی سایت منتقل میشود.

تا اینجا روی گیت آپلود شد.

حالا بحث دسترسی ها میمونه که کاربر معمولی اجازه دسترسی به پنل ادمین رو نداشته باشه که باید از middleware استفاده کنم. و همینطور کاری کنم که کسی بتونه صفحه پروفایل کاربری رو ببینه که حتما لاگین کرده باشه در این صورت از middleware('auth') استفاده میشه برای اون روت بالا آورنده صفحه پروفایل.

خب قبل از اون من کنترلرها را به دو بخش تقسیم میکنم.

بخش Admin

بخش Front

کارهایی که کاربر میخواد انجام بده مثل دیدن صفحه پروفایل، ویرایش صفحه پروفایل و آپلود عکس و ... رو در یک UserController در پوشه Front قرار میدم.