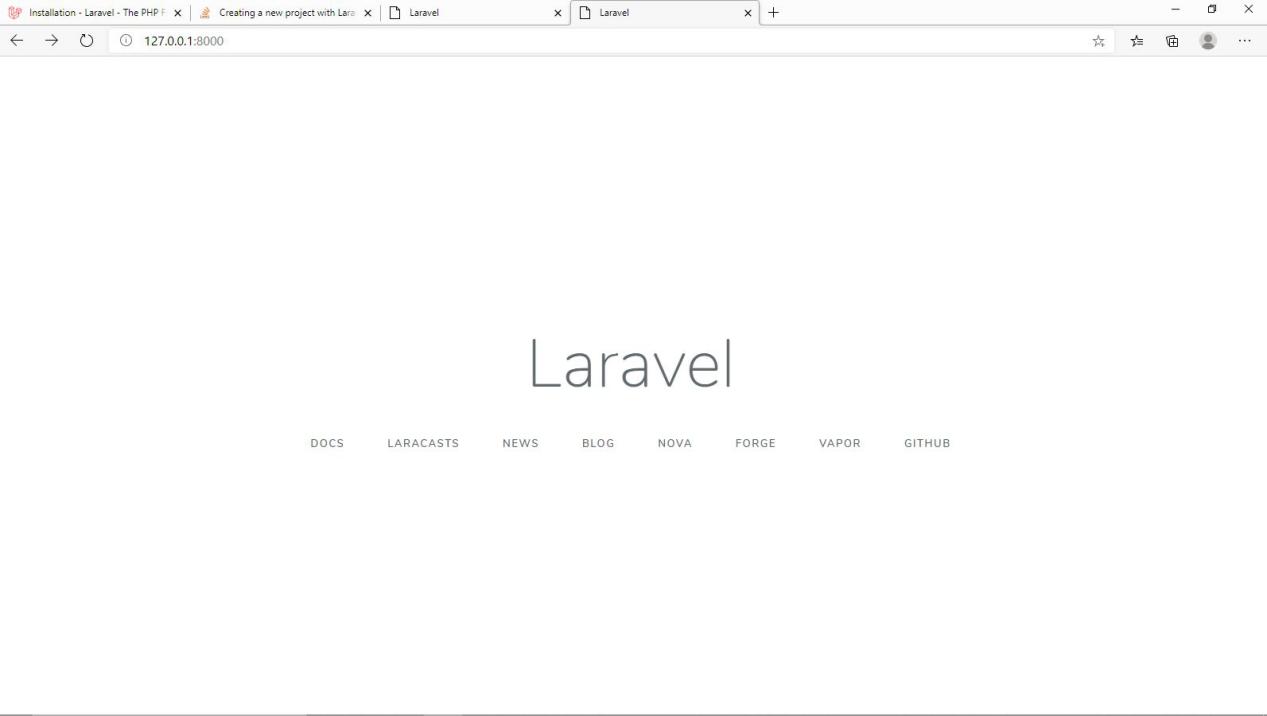
1. Open CMD with administrator.
2. open composer by typing the keyword “composer”
3. Instal laravel with ”composer global require laravel/installer”
4. Create new file with “composer create-project --prefer-dist laravel/laravel projectname”
5. Confirm installation success with “php artisan serve”,this give you an address to load in your browser
6. Once address is loaded into browser laravel welcome page should be returned
7. Route-:adding new directories or paths and telling laravel how to reach it
8. Artisan are set of command line tools type “php artisan’ to get all commands you can run with artisan
9. Typing “php artisan help ‘command’” gives you syntax you can work with,description ,examples
10. To build your controller you cd to your code root file “php artisan make:controller ‘controller name ’”
11. Your controller would be in “C:\xampp\htdocs\laravel\app\Http\Controllers\myFirstcontroller.php”
12. BLADE is a templating engine that comes with laravel
13. The .env file at the project root holds your database credentials and settings,it sets up the applicationn in ts current environment ,make sure the database and password then port is checked and corresponds to your database in xammp or anyother local server

NOTE proper naming in model and table creation make things easieer with laravel,

\*\*The model name should be singular eg app to create a model it is done from cmd type

“php artisan make:model modelName(singular)”;

Model created is founnd in C:\xampp\htdocs\laravel\app\App.php

\*\*table name should be plural eg apps to create table “php artisan make:migration creates\_tablename\_table --create Tablename(plural)”

File created is in C:\xampp\htdocs\laravel\database\migrations\2020\_06\_28\_143824\_create\_apps\_table.php

In this file you find and can add table attributes ,column name etc,then you migrate.

\*\*Migrate this is done in our cmd ,type “php artisan migrate” ,it migrates all tables in the migration folder to our server.

1. php artisan migrate:fresh deletes all tables in that database and reuploads

TO RUN LARAVEL MYSQL in cmd

1. run cmd as admin
2. Cd to project root directory
3. Type “php artisa tinker” to open tinker shell

* View all query “\namespace name\classname:::all() ” in this case eg we have “\App\customer::all()”

namespace name and class name can be found in system model C:\xampp\htdocs\laravel\app\Customer.php

* Where: \App\customer::where('id',1)->get();//more than one

OR

\App\customer::where('id',1)->first();//used in the case we want only 1 result

we are updating the active attribute of customer with id 1

* To update a value:first the specific customer with the where query in a variable like so

“$customer=\App\customer::where('id',1)->first();”

Then

To update we do “$customer->active=0” then we save

“$customer->save()”

TO SEND A MAIL

* Run you cmd as usual
* Type “php artisan make:mail mailableName -m emails.welcome”

The above creates a mailable with name mailableName in mail folder under apps and an emails directory with class welcome inside the views

LARAVEL AUTHENTICATION(Auto login forms and registraton)

While in your poject directory in cmd you run

* “Composer require laravel/ui”

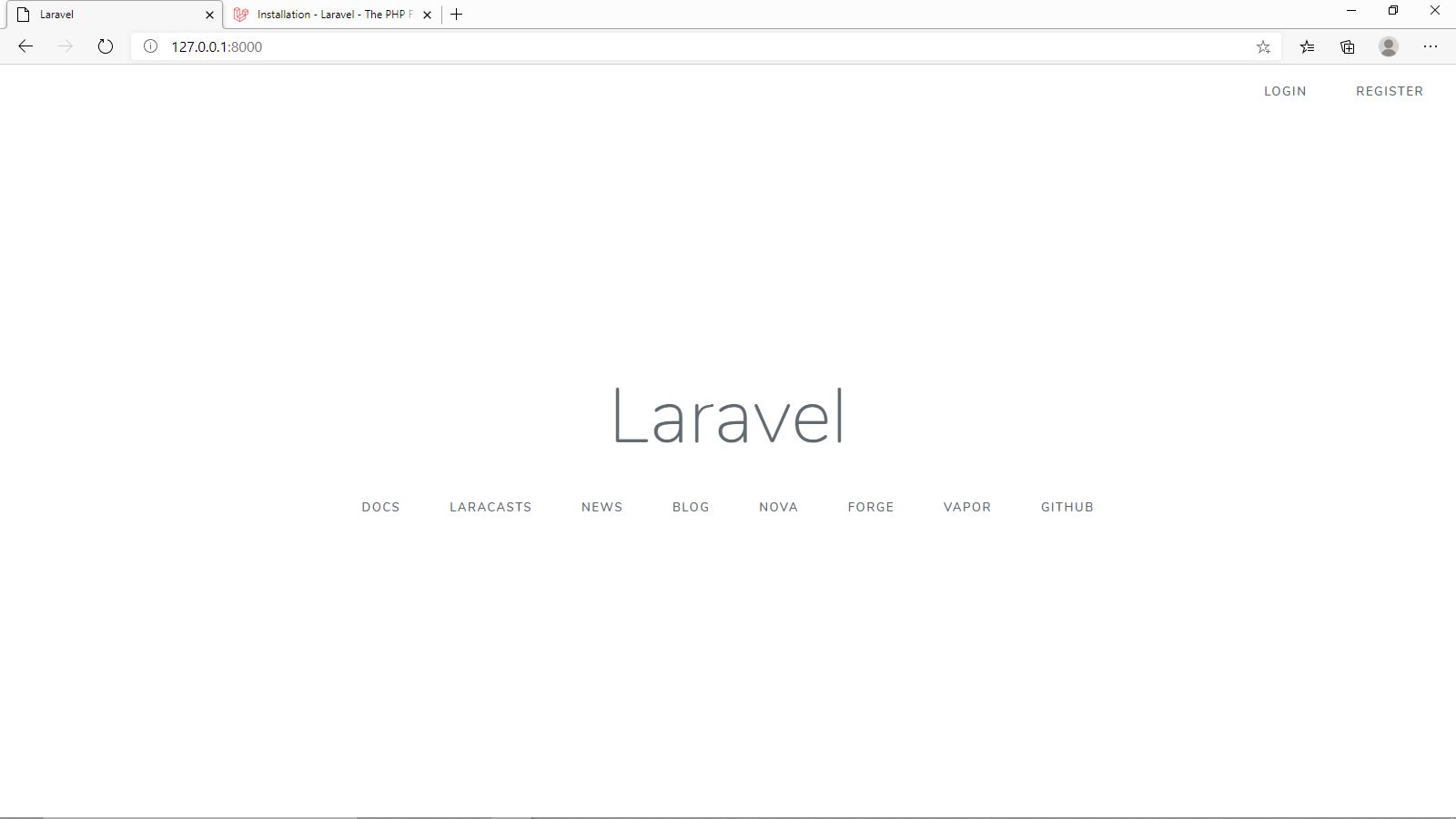
which adds authentication package with login and registration routes for front end registration/login

* To load this package type and Authentication scaffolding.

“php artisan ui:auth”

then you run “php artisan serve”

* Your home page looks like whats below,notice we now have a login and register at the top right corner



* The login and register is functioning and all they need is a database connection which we discussed earlier
* To add styylinggs eg bootstrap,we go back to cmd and type “php artisan ui type --auth” type in our case now is bootstrap and auth means for forms and registration if it were just for general we type “ php artisan ui type” created components can be fouud in C:\xampp\htdocs\project-name\resources\sass
* Before compiling your CSS, install your project's frontend dependencies using the Node package manager (NPM): run “npm install” NOTE you might need t download nodejs and use its cmd to install this to your project directory
* Run “npm run dev” to finally compile your css and stylings,you should get somethngs emitted & compile sass and javascript files
* Serve and reefresh your page and your bootrap should be active

Getting user details in AUth system like laravel\_authentification

To Gets user id

* $data[user\_id]=auth()->user()->id;

RELATIONSHIPS

// relationship between user model and questionnnaire class  
 //meaning a user can be related or have many questionnaires

Done in the user model  
 return $this->hasMany(App\Questionniare::class);

**TINKER**

**In your project path on cmd we can run “php artisan tinker” this used to interact with our systems internal database.**

* **User::all(); ->gives all user details note user is a model from our app directory**

**ACCESSING OUR DATABASE FROM TINKER**

**FROM SUB TABLE**

* **Getting into a model “$modelName=new \App\tableName();**

**>>> $profile=new \App\profiles();**

* **Creating variables in the entered table name attribute**

**>>> $profile->title="Laravel is not easy";**

**>>> $profile->description="But we locomote";**

* **Then assign profile to a stance or user (like a where clause) and inserting the user\_id“$modelName->whereAttribue=1**

**>>> $profile->user\_id=2;**

* **Save effect “$modelName->save()”**

**>>> $profile->save();**

* **To display specific user “$modelName ->user**

**ACCESSING FROM THE MASTER TABLE**

**>>> $user=App\user::find(2);//picking a user id**

**>>> $user->profiles //accessing profile table with user id**

**>>> $user->profiles->url="test.org"; //updating or inserting value**

**>>> $user->profiles->save();//saving**

**LARAVEL QUERIES**

* {{ Auth::*user*()->username }}

**Above gives authenticated users username.**

* user::*all*(); //this is selecting all or a select all statement from user table or model don’t forget to import App/user , it can also be written as below if we don’t want to import model
* App\user::all()
* **User::find($user) //find specific user “$user” Is passed with the function argument and in a path brace in web route**
* **User::findorfail($user) //find specific user or if not found go to a 404 page “$user” Is passed with the function argument and in a path brace in web route**
* **request()->all() //returns all values of just inserted form or details**
* **modelName()::all() //gives all data in that models linked db**
* **post::create($data) //$data here is the variable that holds our validation array**
* **if we wanted to create to a specific user we need the authenticated username or identifying detail we use “auth->user()->post()->create($data)**

**user() and post() are models**

* **modelName()::truncate //wipes values in the table**

**LARAVEL HTTP VERBS**

| **Verb** | **Path** | **Action** | **Route Name** |
| --- | --- | --- | --- |
| GET | /photo  /resourse | index | photo.index for index page |
| GET | /photo/create  /resource/create | create | photo.create create or new eg ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;form |
| POST | /photo  /resourse | store | photo.store saves or store not  viewable |
| GET | /photo/{photo}  /resourse/resourse\_id | show | photo.show show specific  usually a compact id or route binding |
| GET | /photo/{photo}/edit | edit | photo.edit edit specific  usually a compact id or route binding |
| PUT/PATCH | /photo/{photo} | update | photo.update update specific  usually a compact id or route binding |
| DELETE | /photo/{photo} | destroy | photo.destroy delete specific  usually a compact id or route binding |

**IMAGE UPLOAD**

**For image uploads you need to run “php artisan storage:link” which creates a link between your main public folder and your storage folder so image can be viewed, it is ran once in a project.**

**We can resize images using the intervention library which id imported using “composer require intervention/image”**

<http://image.intervention.io/>

**POLICYS**

**This provides a form of restriction on a model “php artisan make:policy Policyname -m modelName”**

**This creates a new folder with name “policies” in the app folder which has our policy file (C:\xampp\htdocs\linstagram\app\Policies\Profilepolicy.php),in which we can pass arguments for all functions made in all controller and then authorice in our respective controller**

***MAKING MAIL MAILABLES AND MARKDOWN TEMPLATES FOR MAIL SENDING***

“php artisan make:mail filename --markdown=foldername\_in\_views.file\_name\_with\_markdown\_template”

This creates a mail folder in our app folder with filename and a view folder with our mark down file