Installation steps:

1. Install Composer (https://getcomposer.org/doc/00-intro.md#globally)

after installation: cmd prompt: type composer i.e (c:\>composer) it will be work correctly.

2. Install Node Js (https://nodejs.org/en/download/)

now check version (c:\> node -v and c:\>npm -v)

3. Install Laravel(https://laravel.com/docs/6.x) and install from C:\>composer global require laravel/installer

Now:(C:\>laravel) and check path (C:\> echo $path)

4. Now will start work in https://www.instagram.com/freecodecamp/

5. Create new Project in Laravel: (c:\> laravel new freeCodeGram (freeCodeGram is project name in laravel) )

6. Goto project: (C:\> cd freeCodeGram)

7. now open a project in Sublime Editor

8. C:\freeCodeGram\> php artisan

9. Run server: C:\freeCodeGram\> php artisan serve (Note: don't close this server in powershell until you are working on porject)

10. Goto Browser (google chrome): (http://127.0.0.1:8000/)

11. Run another PowerShell and goto working folder (C:\freeCodeGram\>)

12: How to view pages in editor (sublime):

open project (freeCodeGram): Goto Resources Folder then click on view Folder and it contain a file (welcom.blade.php which is rendering template)

now change title from laravel to freeCodeGram in welcome.blade.php file

13. to make authentication: C:\freeCodeGram\>php artisan make:auth (this command for old version i.e 5.2)

OR

https://dev.to/msamgan/php-artisan-make-auth-in-laravel-6-0-hc

new way for new version greater than 5.2 version i.e 6.0

type this in powershell

1. C:\freeCodeGram\>composer require laravel/ui

2. C:\freeCodeGram\>php artisan ui vue --auth

14. now check javascript npm version (C:\freeCodeGram\> npm -v) i.e 6.12.0

node js version: (C:\freeCodeGram\>node -v) i.e. v12.13.0

15. now install npm dependencies in project folder (C:\freeCodeGram\> npm install)

16. now compile npm file (C:\freeCodeGram\>npm run dev) Note: run this command 2 times)

17. How to view js in editor (sublime):

open project (freeCodeGram): Goto Resources (folder) goto js (folder) then click on app.js file but this not correct file

18. How to view correct js file in editor (sublime):

open project (freeCodeGram): Goto Public (folder) goto js (folder) and then click on app.js file now that is the correct file and never touch this file.

19. create new database file windows 10: open project (freeCodeGram): Goto Database (folder) and right click on database folder and create a new file with

name "database.sqlite" and save it and leave it blank

(but for Other system C:\freeCodeGram\>vim database/database.sqlite)

20. Make change ".env" file: open project (freeCodeGram in sublime): open file ".env file" replace the following code with new code:

old code New Code

DB\_CONNECTION=mysql DB\_CONNECTION=sqlite

DB\_HOST=127.0.0.1 remove remaining code (DB\_HOST, DB\_PORT, DB\_DATABASE, DB\_USERNAME, DB\_PASSWORD)

DB\_PORT=3306

DB\_DATABASE=laravel

DB\_USERNAME=root

DB\_PASSWORD=

21. Make migration: C:\freeCodeGram\>php artisan migrate

22. now goto locahost server in google chrome (http://127.0.0.1:8000): then click on register and fill register form and click on register button

it will give error:

23. terminate server in powershell (Ctrl + c) and again restart it in powershell (C:\freeCodeGram\> php artisan serve)

24. now Again goto locahost server in google chrome (http://127.0.0.1:8000): then click on register and fill register form and click on register button

25. "You are logged in" message will be displayed in browser (google chrome)

26. for checking this user, now logout

27. again log in AND enter "eamial:test@test.com", & "password=12345678"

28. open project (freeCodeGram)in sublime, then goto resources then views then layouts then click on app.blade.php:

id app section: change this {{ config('app.name', 'Laravel') }} with freeCodeGram and click on save and now check it. logo text will be changed:

29. goto project (freeCodeGram) - public folder - create new folder with name svg and then right click on svg folder - create new file(freeCodeGramLogo.svg)

30. now goto https://simpleicons.org and search "freeCode" file and click on this file then at will be download and then open in chrome

and right click inside and goto view source and then select all the code and copy.

31. now open freeCodeGramLogo.svg file and paste the source code and then save it.

32. agian goto project (freeCodeGram)in sublime, then goto resources then views then layouts then click on app.blade.php:

id app section: before "freeCodeGram" text create two <div> i.e.

<div><img src="/svg/freeCodeGramLogo.svg" style="max-height: 25px"></div>

<div>freeCodeGram</div>

now open browser check the result. the logo and text will be shown in 2 rows.

33. now fixing logo side by side:

goto project (freeCodeGram)in sublime, then goto resources then views then layouts then click on app.blade.php:

id app section update this:

<a class="navbar-brand d-flex" href="{{ url('/') }}">

<div><img src="/svg/freeCodeGramLogo.svg" style="height: 20px; border-right: 1px solid #333" class="pr-3"></div>

<div class="pl-3 pt-1">freeCodeGram</div>

[Note: pr=padding right, pl=padding left, pt=padding top]

now open browser check the result.

34. goto project (freeCodeGram)in sublime, resources folder - saas folder - open app.scss file and insert this fonts from font.google.com website:

at line number 2: @import url('https://fonts.googleapis.com/css?family=Open+Sans:400,700');

and delete the older fonts.

35. goto project (freeCodeGram)in sublime, resources folder - saas folder - open \_varaibles.scss file

at line 5 replace "Nunito" by "Open Sans"

36. goto another powershell: now run npm file (C:\freeCodeGram\>npm run dev) and refresh browser

37. goto project (freeCodeGram)in sublime, resources folder - saas folder - open \_varaibles.scss file

at line 2 change the body color: "#ffffff

38. goto another powershell: now run npm file (C:\freeCodeGram\>npm run dev) and refresh browser

39. now goto project (freeCodeGram)in sublime, then goto resources then views then click on home.blade.php:

change the complete code according to our need.

40. logout user (i.e. zia) from website.

41. adding new col in register form:

project (freeCodeGram)in sublime, then goto resource folder - view folder - auth folder - open "register.blade.php file "

duplicate (copy) email address row: complete div: i.e line 28 - 40

paste this after line 41.

change the all instance of "email" to "username"

after that save a file.

42. Validation:

project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "RegisterController.php file "

goto line 53 and copy complete line of 'email'

and paste it into line 54. now with this code:

'username' => ['required', 'string', 'max:255', 'unique:users'],

save it.

43. Insert feild into table:

project (freeCodeGram)in sublime, then goto app folder - Databe folder - migrations folder- "2014\_10\_12\_000000\_create\_users\_table.php file "

goto line 19 and copy complete line of 'email'

and paste it into line 20. now with this code:

$table->string('username')->unique();

save it.

44. create data feild after validation and table feild insertion:

project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "RegisterController.php file "

goto line 69 and copy complete line of 'email'

and paste it into line 70. now with this code:

'username' => $data['username'],

save it.

45. goto browser and fill register form and click on register.

46. goto powershell: (tinker command work in the background)

C:\freeCodeGram\>php artisan tinker

>>>User::all(); [this will show users]

>>>exit [exit from tinker]

47. make migration with fresh table and delete old migration:

C:\freeCodeGram\>php artisan migrate:fresh

48. inserting username in fillable ftn:

project (freeCodeGram)in sublime, then goto app folder - "User.php file "

goto line 19 and copy 'email', and paste it and change with: 'username',

save file.

49. goto browser and fill register form and click on register because all old data erased.

50. now again go to tinker.

C:\freeCodeGram\>php artisan tinker

>>>User::all(); [this will show users]

>>>exit [exit from tinker]

51. displaying username in browser:

project (freeCodeGram)in sublime, then goto resources then views then layouts then click on app.blade.php:

goto line 55, replace 'name' with 'username'

save file and check browser. name=zia will be replace by username=ziakhan

52. make new controller i.e. profiles controller.

C:\freeCodeGram\>php artisan make:controller ProfilesController

now checking a controller in sublime editor:

project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "ProfilesController.php file

53. now goto project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "HomeController.php file

copy line 24 - 27 and

paste into line 9 of Profilescontroller.php

54. Now goto project (freeCodeGram)in sublime, then goto routes folder - web.php

now replace 'HomeController' with 'ProfilesController' at line 20.

save it

55. now close server and restart again by powershell

56. make changes in profilecontroller.php

<?php

namespace App\Http\Controllers;

use App\User;

use Illuminate\Http\Request;

class ProfilesController extends Controller

{

public function index($user)

{

$user =User::find($user);

return view('home', [

'user' => $user,

]);

}

}

57. make changes in web.php

line 20 Route::get('/profile/{user}', 'ProfilesController@index')->name('profile.show');

58. now goto project (freeCodeGram)in sublime, then goto resources then views then click on home.blade.php:

change line 10: 'freeCodeCamp' with {{ $user->username }}

59. creating a model (profile): C:\freeCodeGram\>php artisan make:model Profile -m (Profile=model name, -m=make new migration into table)

60. project (freeCodeGram)in sublime, then goto database folder - migrations folder - \_create\_profiles\_table.php

insert this code after line 17:

$table->unsignedBigInteger('user\_id');

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

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

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

$table->timestamps();

$table->index('user\_id');

61. now make migrations in powershell: C:\freeCodeGram\>php artisan migrate

62. Now goto project (freeCodeGram)in sublime, then goto app folder - Profile.php:

insert this code:

class Profile extends Model

{

public function user()

{

return $this->belongsTo(User::class);

}

}

63. Now goto project (freeCodeGram)in sublime, then goto app folder - User.php:

insert this code at line 39:

public function profile()

{

return $this->hasOne(Profile::class);

}

64. now again go to tinker. (manually enter profile and set)

C:\freeCodeGram\>php artisan tinker

>>> $profile = new \App\Profile(); [adding new profile ]

[ App\Profile {#3018}] new profile id will be like this.

>>> $profile->title = 'Cool Title';

=> "Cool Title" [return output]

>>> $profile->description = 'description';

=> "description" [return output]

>>> $profile->user\_id = 1;

=> 1 [return output]

>>> $profile->save();

=> true [return output]

>>> $profile->user

=> App\User {#3029

id: "1",

name: "zia",

email: "test@test.com",

username: "zia khan",

email\_verified\_at: null,

created\_at: "2019-10-24 07:02:10",

updated\_at: "2019-10-24 07:02:10",

}

>>> $profile

=> App\Profile {#3018

title: "Cool Title",

description: "description",

user\_id: 1,

updated\_at: "2019-10-25 04:54:30",

created\_at: "2019-10-25 04:54:30",

id: 1,

user: App\User {#3029

id: "1",

name: "zia",

email: "test@test.com",

username: "zia khan",

email\_verified\_at: null,

created\_at: "2019-10-24 07:02:10",

updated\_at: "2019-10-24 07:02:10",

},

}

>>> $profile->user

=> App\User {#3029

id: "1",

name: "zia",

email: "test@test.com",

username: "zia khan",

email\_verified\_at: null,

created\_at: "2019-10-24 07:02:10",

updated\_at: "2019-10-24 07:02:10",

}

>>> $user=App\User::find(1);

=> App\User {#3033

id: "1",

name: "zia",

email: "test@test.com",

username: "zia khan",

email\_verified\_at: null,

created\_at: "2019-10-24 07:02:10",

updated\_at: "2019-10-24 07:02:10",

}

>>> $user->profile

>>> $user->profile->url = 'freecodecamp.org';

>>> $user->save();

>>> $user->push();

>>>exit [exit from tinker]

65. now goto project (freeCodeGram)in sublime, then goto resources then views then click on home.blade.php:

change line 16-18:

<div class="pt-4 font-weight-bold">{{ $user->profile->title }}</div>

<div>{{ $user->profile->description }}</div>

<div><a href="#">{{ $user->profile->url }}</a></div>

66. make changes in profilecontroller.php (handling profile id does not exist in database)

in line 12 replace this with old code: $user =User::findOrFail($user);

67. creating a model (Post): C:\freeCodeGram\>php artisan make:model Profile -m (Profile=model name, -m=make new migration into table)

68. project (freeCodeGram)in sublime, then goto database folder - migrations folder - \_create\_posts\_table.php (make changes)

$table->bigIncrements('id');

$table->unsignedBigInteger('user\_id');

$table->string('caption');

$table->string('image');

$table->timestamps();

$table->index('user\_id');

69. now make migrations in powershell: C:\freeCodeGram\>php artisan migrate

70. Now goto project (freeCodeGram)in sublime, then goto app folder - User.php:

insert this code at line 40 before profile funtion:

public function posts()

{

return $this->hasMany(Post::class);

}

71. Now goto project (freeCodeGram)in sublime, then goto app folder - Post.php:

insert this code:

class Post extends Model

{

public function user()

{

return $this->belongsTo(User::class);

}

}

72. now goto project (freeCodeGram)in sublime, then goto resources then views then click on home.blade.php:

make changes line 10:

<div class="d-flex justify-content-between align-items-baseline">

<h1>{{ $user->username }}</h1>

<a href="#">Add New Post</a>

</div>

73. make changes in web.php:

add this code at line 19: Route::get('/p', 'PostController@create');

74. now make PostController in powershell: C:\freeCodeGram\>php artisan make:controller PostController

75. project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "PostController.php file

class PostController extends Controller

{

public function create()

{

return view('posts.create');

}

}

76. create/make posts folder in views folder.

77. create/make profiles folder in views folder.

78. now move home.blade.php file into profiles folder and rename it by index.blade.php

79. goto ProfilesController.php and make changes in line 13: replace home by profiles.index

return view('profiles.index',

80. now duplicate home.labde.php and paste it in posts folder and rename it create.blade.php

now delete all lines from 5 - 35

81. now goto project (freeCodeGram)in sublime, then goto resources folder - views folder - auth folder then click on register.blade.php

copy from line 14 - 25

82. now goto create.blade.php and paste line 14 - 15 of register.blade.php file

now all instances of word 'name' by 'caption'

83. now duplicate home.labde.php and paste it in posts folder and rename it create.blade.php

update file with this code:

@extends('layouts.app')

@section('content')

<div class="container">

<form action="/p" enctype="multipart/form-data" method="post">

@csrf

<div class="row">

<div class="col-8 offset-2">

<div class="row">

<h1>Add New Post</h1>

</div>

<div class="form-group row">

<label for="caption" class="col-md-4 col-form-label">Post Caption</label>

<input id="caption"

type="text"

class="form-control {{ $errors->has('caption') ? ' is-invalid' : '' }}"

name="caption"

value="{{ old('caption') }}"

required autocomplete="caption" autofocus>

@if ($errors->has('caption'))

<span class="invalid-feedback" role="alert">

<strong>{{ $errors->first('caption') }}</strong>

</span>

@endif

</div>

<div class="row">

<label for="image" class="col-md-4 col-form-label">Post Image </label>

<input type="file" class="form-control-file" id="image" name="image">

@if ($errors->has('image'))

<strong>{{ $errors->first('image') }}</strong>

@endif

</div>

<div class="row pt-4">

<button class="btn btn-primary">Add New Post</button>

</div>

</div>

</div>

</form>

</div>

@endsection

84. make changes in web.php:

add this code at line 21: Route::post('/p', 'PostController@store');

85. project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "PostController.php file

add this function at line 14:

public function store()

{

$data = request()->validate([

'caption'=> 'required',

'image'=> ['required', 'image'],

]);

auth()->user()->posts()->create($data);

dd(request()->all());

}

86. Now goto project (freeCodeGram)in sublime, then goto app folder - Post.php:

insert this code in 9:

class Post extends Model

{

protected $guarded = [];

public function user()

{

return $this->belongsTo(User::class);

}

}

87. now go to tinker. C:\freeCodeGram\>php artisan tinker

>>> Post::all();

[!] Aliasing 'Post' to 'App\Post' for this Tinker session.

=> Illuminate\Database\Eloquent\Collection {#3028

all: [

App\Post {#3029

id: "1",

user\_id: "1",

caption: "Caption for image",

image: "C:\xampp\tmp\phpCD3A.tmp",

created\_at: "2019-10-26 07:11:06",

updated\_at: "2019-10-26 07:11:06",

},

],

}

>>>

88. to proctect from un-authorized user to insert images: project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "PostController.php file

at this function code at line 9:

public function \_\_construct()

{

$this->middleware('auth');

}

89. show the proper path of an image: project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "PostController.php file

add this function at line 27:

dd(request('image')->store('uploads', 'public'));

90. watching an image in folder: project (freeCodeGram)in sublime, then goto storage folder - app folder - public folder - uploads folder: image will be store here and just click on it, it will be display there but image link will not be display on browser.

91. to create storage link now go to shell= C:\freeCodeGram\>php artisan storage:link

it will create a storage link .

92. to adding proper link of authorized user to insert images: project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "PostController.php file

at store function code at line :

public function store()

{

$data = request()->validate([

'caption'=> 'required',

'image'=> ['required', 'image'],

]);

$imagePath = (request('image')->store('uploads', 'public'));

auth()->user()->posts()->create([

'caption' => $data['caption'],

'image' => $imagePath,

]);

return redirect('/profile/'.auth()->user()->id);

}

93. now goto index.blade.php file: remove data from line 25 to end of this row and new code here:

<div class="row pt-4">

@foreach($user->posts as $post)

<div class="col-4">

<img src = "/storage/{{ $post->image }}" class="w-100">

</div>

@endforeach

</div>

And now insert href in line 12: i.e.

<a href="/p/create">Add New Post</a>

94. now go to tinker for image deletion/truncation. C:\freeCodeGram\>php artisan tinker

>>> Post::truncate();

95. Now goto browser: add new post: (<http://127.0.0.1:8000/profile/1>) and add three posts with images.

96. now goto app folder – providers folder - User.php (model):

Update this function:

public function posts()

{

return $this->hasMany(Post::class)->orderBy('created\_at', 'DESC'); }

97. now goto index.blade.php file: remove data from line 15 to end of this row and new code here:

<div class="pr-5"><strong>{{ $user->posts->count() }}</strong> posts</div>

And also update line 27: <div class="col-4 pb-4">

98. Adding new image library: C:\freeCodeGram\> composer require intervention/image

99. Now goto project (freeCodeGram)in sublime, then goto app folder - HTTP folder - Controllers folder - Auth folder- "PostController.php file” using image library:

Add this code at line 6: use Intervention\Image\Facades\Image;

Add this code at line 30 - 31: $image = Image::make(public\_path("storage/{$imagePath}"))->fit(1200, 1200);

$image->save();

100. Displaying caption of an image: now goto index.blade.php file at line 28:

<a href="/p/{{ $post->id }}">

<img src = "/storage/{{ $post->image }}" class="w-100">

</a>

101. Now go to web.php file: create new route:

Route::get('/p/{post}', 'PostController@show');

102. Now go to PostController.php: add new function:

public function show(\App\Post $post)

{

return view('posts.show', compact('post'));

}

103. Now create new file in inner ‘posts folder’: show.blade.php to show the caption of images.

@extends('layouts.app')

@section('content')

<div class="container">

<div class="row">

<div class="col-8">

<img src="/storage/{{ $post->image }}" class="w-100">

</div>

<div class="col-4">

<div>

<h3>{{ $post->user->username }}</h3>

<p>{{ $post->caption }}</p>

</div>

</div>

</div>

</div>

@endsection

104. now check it in browser: when you will click on post-image, then each post-image display an image and its caption and description as well.

105. now go to index.blade.php to create new profile link.

Add this code at line 13, under Add new post</div>:

<div>

<a href="/profile/{{ $user->id }}/edit">Edit Profile</a>

</div>

106. Now go to web.php file: create new route for edit profile: add this code at line 28:

Route::get('/profile/{user}/edit', 'ProfilesController@edit')->name('profile.edit');

107. Now goto app folder - HTTP folder - Controllers folder - Auth folder- "ProfilesController.php file:

Update complete file:

<?php

namespace App\Http\Controllers;

use App\User;

use Illuminate\Http\Request;

class ProfilesController extends Controller

{

public function index(User $user)

{

return view('profiles.index', compact('user'));

}

public function edit(User $user)

{

return view('profiles.edit', compact('user'));

}

}

108. Now create new file in inner ‘profiles folder’: edit.blade.php:

109. Now go to web.php file: create new route for update profile: add this code at line 30:

Route::patch('/profile/{user}', 'ProfilesController@update')->name('profile.update');

110. Now goto app folder - HTTP folder - Controllers folder - Auth folder- "ProfilesController.php file:

Add new function/method:

public function update(User $user)

{

$data = request()->validate([

'title'=> 'required',

'description'=> 'required',

'url'=> 'url',

'image'=> '',

]);

auth()->$user->profile->update($data);

return redirect("/profile/{$user->id}");

}

111. now go to app folder - providers folder – Profile.php

Update this method to handle mass assignment error:

class Profile extends Model

{

protected $guarded = [];

public function user()

{

return $this->belongsTo(User::class);

}

}

Now goto browser update profile and it will be updated.

112. Policies: to restrict the un-authorized users.

C:\freeCodeGram\>php artisan make:policy ProfilePolicy -m Profile

Policy created successfully.

113. Now go to app folder – policies folder – ProfilePolicy.php

Update this function:

public function update(User $user, Profile $profile)

{

return $user->id == $profile->user\_id;

}

114. . Now goto app folder - HTTP folder - Controllers folder - Auth folder- "ProfilesController.php file:

Update these functions/methods:

public function edit(User $user)

{

$this->authorize('update', $user->profile);

return view('profiles.edit', compact('user'));

}

public function update(User $user)

{

$this->authorize('update', $user->profile);

$data = request()->validate([

'title'=> 'required',

'description'=> 'required',

'url'=> 'url',

'image'=> '',

]);

auth()->$user->profile->update($data);

return redirect("/profile/{$user->id}");

}

Now logout in browser and check <http://127.0.0.1:8000/profile/1/edit> it will give error 403.

115. now go to index.blade.php file and edit profile button will off when user is not login:

Update line 12 – 13 and 14 – 20

@can ('update', $user->profile)

<a href="/p/create">Add New Post</a>

@endcan

<div>

@can ('update', $user->profile)

<a href="/profile/{{ $user->id }}/edit">Edit Profile</a>

@endcan

</div>

116. Now go to web.php file: now arrange routes in this order then /p/create will be work

Route::get('/p/create', 'PostController@create');

Route::post('/p', 'PostController@store');

Route::get('/p/{post}', 'PostController@show');

117. Now go to ProfileController.php and update this code:

Add this to line 7: use Intervention\Image\Facades\Image;

Add this to update function at line 36 - 48:

if (request('image')) {

$imagePath = (request('image')->store('profile', 'public'));

$image = Image::make(public\_path("storage/{$imagePath}"))->fit(1000, 1000);

$image->save();

}

auth()->user()->profile->update(array\_merge(

$data,

['image' => $imagePath]

));

return redirect("/profile/{$user->id}");

118. now goto database folder – then \_create\_profile\_table.php

At line 22 enter this: $table->string('image')->nullable();

119. now migrate database: C:\freeCodeGram\> php artisan migrate:fresh (everything will be washout)

Now create user again – goto browser and click on register and so on ….

120. now go to app folder - providers folder – User.php

Add this function at lin 40 – 48:

protected static function boot()

{

parent::boot(); //todo: chnage the autogenerated stub

static::created(function($user){

$user->profile()->create([

'title' => $user->username,

]);

});

}

121. now migrate database: C:\freeCodeGram\> php artisan migrate:fresh (everything will be washout)

Now create user again – goto browser and click on register – create new user and so on ….

122. now go to index.blade.php file:

Update this code at line 6 - 8:

<div class="col-4 p-5">

<img alt="" class="rounded-circle" src="/storage/{{ $user->profile->image }}">

</div>

123. Now go to browser and edit profile.

124. Now go to show.blade.php file:

Update this code after line 9:

<div class="col-4">

<div>

<div class="d-flex align-items-center">

<div class="pr-3">

<img src="/storage/{{ $post->user->profile->image }}" class="rounded-circle w-100" style="max-width: 40px;">

</div>

<div>

<div class="font-weight-bold">

<a href="/profile/{{ $post->user->id }}">

<span class="text-dark"> {{ $post->user->username }}</span>

</a>

<a href="#" class="pl-3">Follow</a>

</div>

</div>

</div>

<hr>

<p><span class="font-weight-bold">

<a href="/profile/{{ $post->user->id }}">

<span class="text-dark">{{ $post->user->username }}</span>

</a>

</span> {{ $post->caption }}

</p>

</div>

</div>

125. Now logout user 1 i.e test and register another user (new user2)

126. download “no-image-vailable” from internet and upload in user2 profile image

Then copy image address from user2 profile image (right click on profile image and click on copy image address)

127. now go to app folder - providers folder – Profile.php

Add new function after line 10:

public function profileImage()

{

return '/storage/' . ($this->image) ? $this->image : '/profile/YgN4G81RQCT5NhOHNXfUUjowURUyiW8e265NIy8b.jpeg'; //(image address pasted here)

}

128. Now go to show.blade.php file:

Update this code after line 10:

<div class="pr-3">

<img src="{{ $post->user->profile->profileImage() }}" class="rounded-circle w-100" style="max-width: 40px;">

</div>

129. Now goto indexblade.php file:

Update this code after line 6:

<div class="col-4 p-5">

<img src="{{ $user->profile->profileImage() }}" class="rounded-circle w-100">

</div>

130. Now go to ProfileController.php

update this code at line 36 - 48:

if (request('image')) {

$imagePath = (request('image')->store('profile', 'public'));

$image = Image::make(public\_path("storage/{$imagePath}"))->fit(1000, 1000);

$image->save();

$imageArray = ['image' => $imagePath];

}

auth()->user()->profile->update(array\_merge(

$data,

$imageArray ?? []

));

131. Now goto indexblade.php file:

Updating follow button at line 11 – 14:

<div class="d-flex align-items-center pb-4">

<div class="h4">{{ $user->username }}</div>

<button class="btn btn-primary ml-4">Follow</button>

</div>

132. Now go to Public (folder) – js folder – components folder – ExampleComponent.vue

“ExampleComponent.vue” this file by “FollowButton.vue”

133. No go to Public (folder) – js folder – components folder – app.js

Update line 22:

Vue.component('follow-button', require('./components/FollowButton.vue').default);

134. now go to index.blade.php and cut this code from line 13:

<button class="btn btn-primary ml-4">Follow</button>

Paste in FollowButton.vue in template, remove all data upto </template> then paste this.

<template>

<div>

<button class="btn btn-primary ml-4">Follow</button>

</div>

</template>

135. now go to index.blade.php

Update line 13:

<follow-button></follow-button>

136. now goto powershell:

C:\freeCodeGram\>npm run watch (you can run again again to watch changes)

When it will runs completely then go to browser and you will not find any changes.

But when ever you made changes in FollowButton.vue file it modify whole project.

e.g. <button class="btn btn-primary ml-4">Follow</button> by

<button class="btn btn-primary ml-4">**Follow ME**</button>

Now save it and goto browser check the Follow button.

137. Make controller for Follow button:

C:\freeCodeGram> php artisan make:controller FollowsController

Controller created successfully.

138. Now go to app folder – Http folder – controllers folder – FollowsController.php file:

<?php

namespace App\Http\Controllers;

use App\User;

use Illuminate\Http\Request;

class FollowsController extends Controller

{

public function store(User $user)

{

return $user->username;

}

}

139. add this code to FollowButton.vue

<template>

<div>

<button class="btn btn-primary ml-4" @click="followUser">Follow</button>

</div>

</template>

<script>

export default {

props: ['userId'],

mounted() {

console.log('Component mounted.')

},

methods: {

followUser() {

axios.post('/follow/' + this.userId)

.then(response => {

alert(response.data);

});

}

}

}

</script>

140. web.php: add new route:

At line 20 add this code:

Route::post('follow/{user}', 'FollowsController@store');

141. now go to index.blade.php

Update line 13:

<follow-button user-id="{{ $user->id }}"></follow-button>

Now browser will return correct id of user when you click on follow button.

142. Make migration:

C:\freeCodeGram> php artisan make:migration creates\_profile\_user\_pivot\_table --create profile\_user

143. now goto app folder – database folder – migrations folder – “\_\_ \_creates\_profile\_user\_pivot\_table.php” file

Add this code at line 18 – 19:

$table->unsignedBigInteger('profile\_id');

$table->unsignedBigInteger('user\_id');

144. now migrate database:

C:\freeCodeGram> php artisan migrate

Migrating: 2019\_10\_28\_074002\_creates\_profile\_user\_pivot\_table

Migrated: 2019\_10\_28\_074002\_creates\_profile\_user\_pivot\_table (0.44 seconds)

145. Now go to project (freeCodeGram)in sublime, then goto app folder - "User.php file "

Add this function at line 56 - :

public function following()

{

return $this->belongsToMany(Profile::class);

}

146. Now go to project (freeCodeGram)in sublime, then goto app folder - "Profile.php file "

Add this function at line 18 - :

public function followers()

{

return $this->belongsToMany(User::class);

}

147. Now go to app folder – Http folder – controllers folder – FollowsController.php file:

Update this function:

public function store(User $user)

{

return auth()->user()->following()->toggle($user->profile);

}

Now go to browser

148. Now go to FollowButton.vue file:

Update this method:

methods: {

followUser() {

axios.post('/follow/' + this.userId)

.then(response => {

console.log(response.data);

});

}

}

149. Now go to tinker: C:\freeCodeGram\>php artisan tinker

>>> $user = User::find(1);

[!] Aliasing 'User' to 'App\User' for this Tinker session.

=> App\User {#3038

id: "1",

name: "test user",

email: "test@test.com",

username: "test",

email\_verified\_at: null,

created\_at: "2019-10-28 04:47:27",

updated\_at: "2019-10-28 04:47:27",

}

>>> $user->following

=> Illuminate\Database\Eloquent\Collection {#3044

all: [],

}

>>> exit

150. Now again go to FollowButton.vue file:

151. index.blade.php file:

Update line 13:

<follow-button user-id="{{ $user->id }}" follows="{{ $follows }}"></follow-button>

152. ProfilesController.php

Update this function:

public function index(User $user)

{

$follows = (auth()->user()) ? auth()->user()->following->contains($user->id) : false;

return view('profiles.index', compact('user', 'follows'));

}

153. Now again go to FollowButton.vue file: and update this file completely:

<template>

<div>

<button class="btn btn-primary ml-4" @click="followUser" v-text="buttonText"></button>

</div>

</template>

<script>

export default {

props: ['userId', 'follows'],

mounted() {

console.log('Component mounted.')

},

data: function () {

return {

status: this.follows,

}

},

methods: {

followUser() {

axios.post('/follow/' + this.userId)

.then(response => {

this.status = ! this.status;

console.log(response.data);

});

}

},

computed: {

buttonText() {

return (this.status) ? 'Unfollow' : 'Follow';

}

}

}

</script>

154. Now go to followController.php:

Add this function (\_\_construct):

public function \_\_construct()

{

$this->middleware('auth');

}

155. Now again go to FollowButton.vue file:

Update this line 22 – 36:

methods: {

followUser() {

axios.post('/follow/' + this.userId)

.then(response => {

this.status = ! this.status;

console.log(response.data);

})

.catch(errors => {

if (errors.response.status == 401) {

window.location = '/login';

}

});

}

},

156. index.blade.php file:

Update line 33 – 34:

<div class="pr-5"><strong>{{ $user->profile->followers->count() }}</strong> followers</div>

<div class="pr-5"><strong>{{ $user->following->count() }}</strong> following</div>

157. Laravel Telescope: very important command to look in laravel (search about it on google)

Install it by the following command:

C:\freeCodeGram> composer require laravel/telescope (it will download telescope package)

C:\freeCodeGram> php artisan telescope:install (it will install telescope)

Now migrate it:

C:\freeCodeGram> php artisan migrate

158. to view laravel telescope on browser: apply this command (it is very powerful tool)

<http://127.0.0.1:8000/telescope/requests>

159. Now go to project (freeCodeGram)in sublime, then goto app folder – Http folder – controllers folder – Auth folder – LoginController.php file:

Update line 28 (remove “home”)

protected $redirectTo = '/';

160. now go to web.php file: update this code in 14 – 16:

(Route::get('/', function () { return view('welcome');});

By this and move to line 19, just under 'PostController@store'

Route::get('/','PostController@index');

161. now go to “PostController.php file”

Create a function under the (public function \_\_construct()):

use App\Post;

use App\User;

use Illuminate\Http\Request;

use Intervention\Image\Facades\Image;

class PostController extends Controller

{

public function \_\_construct()

{

$this->middleware('auth');

}

public function index()

{

$users = auth()->user()->following()->pluck('profiles.user\_id');

$posts = Post::whereIn('user\_id', $users)->latest()->get();

return view('posts.index', compact('posts'));

}

162. Now go to project (freeCodeGram)in sublime, then goto resources folder – views folder – posts folder –

create a new file (index.blade.php file) of posts.

@extends('layouts.app')

@section('content')

<div class="container">

@foreach($posts as $post)

<div class="row">

<div class="col-6 offset-3">

<a href="/profile/{{ $post->user->id }}">

<img src="/storage/{{ $post->image }}" class="w-100">

</a>

</div>

</div>

<div class="row pt-4 pb-2">

<div class="col-6 offset-3">

<div>

<p>

<span class="font-weight-bold">

<a href="/profile/{{ $post->user->id }}">

<span class="text-dark">{{ $post->user->username }}</span>

</a>

</span> {{ $post->caption }}

</p>

</div>

</div>

</div>

@endforeach

<div class="row">

<div class="col-12 d-flex justify-content-center">

{{ $posts->links() }}

</div>

</div>

</div>

@endsection

163. now go to “PostController.php file”

Update the following function:

public function index()

{

$users = auth()->user()->following()->pluck('profiles.user\_id');

$posts = Post::whereIn('user\_id', $users)->with('user')->latest()->paginate(5);

return view('posts.index', compact('posts'));

}

164. To clear entries in telescope:

C:\freeCodeGram> php artisan telescope:clear

(output: Telescope entries cleared!)

Now go to browser and refresh this page (<http://127.0.0.1:8000/telescope/requests>)

165. Now go to ProfilesController.php (cache function increase your website speed at browser)

Update this function:

public function index(User $user)

{

$follows = (auth()->user()) ? auth()->user()->following->contains($user->id) : false;

$postCount = Cache::remember(

'count.posts.' . $user->id,

now()->addSeconds(30),

function() use ($user) {

return $user->posts->count();

});

$followersCount = Cache::remember(

'count.followers.' . $user->id,

now()->addSeconds(30),

function() use ($user) {

return $user->profile->followers->count();

});

$followingCount = Cache::remember(

'count.following.' . $user->id,

now()->addSeconds(30),

function() use ($user) {

return $user->following->count();

});

return view('profiles.index', compact('user', 'follows', 'postCount', 'followersCount', 'followingCount'));

}

166. go to profile folder - index.blade.php file: (update line 31 – 35)

<div class="d-flex">

<div class="pr-5"><strong>{{ $postCount }}</strong> posts</div>

<div class="pr-5"><strong>{{ $followersCount }}</strong> followers</div>

<div class="pr-5"><strong>{{ $followingCount }}</strong> following</div>

</div>

167. sending emails: (mailtrap.io search on google - it is an actually virtual inbox for email)

1. goto mailtrap.io

2. sign in with google account:

3. your account will be opened:

4. click on demo inbox and it will be empty but you will see integrations

5. copy user\_name and password:

:user\_name => '46aa112df2b33a',

:password => 'cc96923c87fbb5',

6. Now goto project (freeCodeGram)in sublime, open (.env) file

Update line 26 – 27

MAIL\_USERNAME=46aa112df2b33a

MAIL\_PASSWORD=cc96923c87fbb5

168. Now goto project (freeCodeGram)in sublime, app folder – providers folder - User.php file:

At line 8 add this code:

use Illuminate\Support\Facades\Mail;

update this function:

static::created(function($user){

$user->profile()->create([

'title' => $user->username,

]);

Mail::to($user->eamil)->send(new NewUserWelcomeMail());

});

169. make mail:

C:\freeCodeGram> php artisan make:mail NewUserWelcomeMail -m emails.welcome-email

(Mail created successfully. - output)

170. Now goto project (freeCodeGram)in sublime, app folder – mail folder – NewUserWelcomeMail.php

171. Now goto project (freeCodeGram)in sublime, resources folder – views folder – emails folder - welcome-email.blade

@component('mail::message')

# Welcome freeCodeGram

This is a community of fellow developers and we love that you have joined us.

All the best,<br>

Zia Ur Rehman

@endcomponent

172. now goto web.php file to route for email.

Update line 13 – 19:

use App\Mail\NewUserWelcomeMail;

Auth::routes();

Route::get('/email', function () {

return new NewUserWelcomeMail();

});

173. now goto freeCodeGram project: App folder – Http folder – Controllers folder - Auth folder - RegisterController.php

Update line 31: protected $redirectTo = '/'; just remove (‘/home)

174. Finally goto

1. <https://www.youtube.com/coderstape>

2. <https://github.com/coderstape/freeCodeGram> (Full coded course)

3. <https://laravel.com/docs/6.x#installation> (very important for reading)

4. <https://laravel-news.com/>