Fisrt OF All :

1- ***New laravel installation :***

   composer create-project --prefer-dist laravel/laravel mct 5.2.29

2- ***Database configuration :***

   (change database name in .env file)

3- ***Setting Up Admin Views :***

   - **Php artisan make:auth**
   - **Go To Views and Create Admin Folder this folder contains index.blade.php**
     - **Also contains 3 folders [users – posts – categories]**
     - **Users ➔[index – create – edit]**
     - **Categories➔[index – edit]**
     - **Posts ➔[index – create – edit]**

4- Users table migration:

   - Add 2 integer columns to the usere table befor migration [role_id &is_active]
     The default value for is_active is 0
   - Php artisan make:model role –m
   - Add name to role table

5- Relation setup and data entry

   In the user model :

```
public  function  role(){

    return $this->belongsTo('App\Role');
}
```

   - then make migration : *php artisan migrate*

6- Admin controller and routes :

   Php artisan make:controller –resource AdminUsersController

   - Add route :

```
Route::resource('admin/users' , 'AdminUsersController');

Route::get('/admin' , function(){
    return view('admin.index');
});
```

7- Installing node js:

   Download node js and install it

   Then :

   Go to cmd and write :

```
Npm install -g gulp
npm install --global gulp-cli
```

8- Gulpe and Assets :

```
.style([
            'Libs/blog-post.css',
            'Libs/bootsrap.css',
            'Libs/styles.css'

       ],'./public/css/libs.css')

    .scripts([
            'Libs/jquery.js'


    ],'./public/js/libs.js')
```

- Then Write in terminal :

```
gulp
```

  This will compine all files into one file

9- Admin master file :

Go To Layouts and add admin.blade.php

Copy design from anywhere you want and paste it into this file

- To link to your css and js files that you combined :

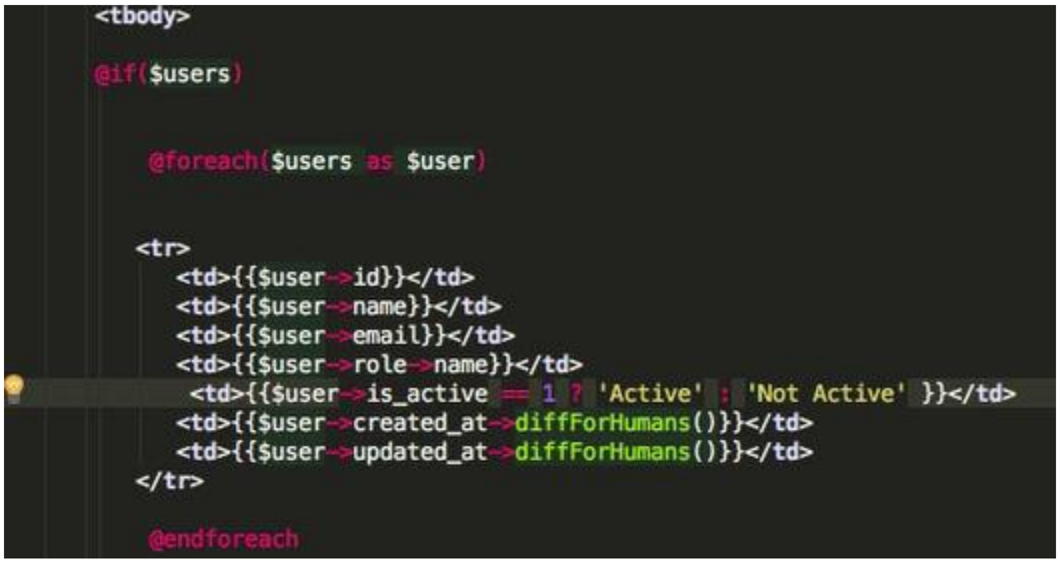
```
<link href="{{ asset('css/app.css') }}" rel="stylesheet">
<link href="{{ asset('css/libs.css') }}" rel="stylesheet">
<script src="{{ asset('js/libs.js') }}"></script>
```

10- If tou want to change anything in stylesheet after merge you can do it in the super css file [sass.scss] and re gulp

11- Displaying Users :

AadminUsersComtroller -> index

```
public function index()
{
    $users = User::all();
    return view('admin.users.index' , compact('users'));
}
```

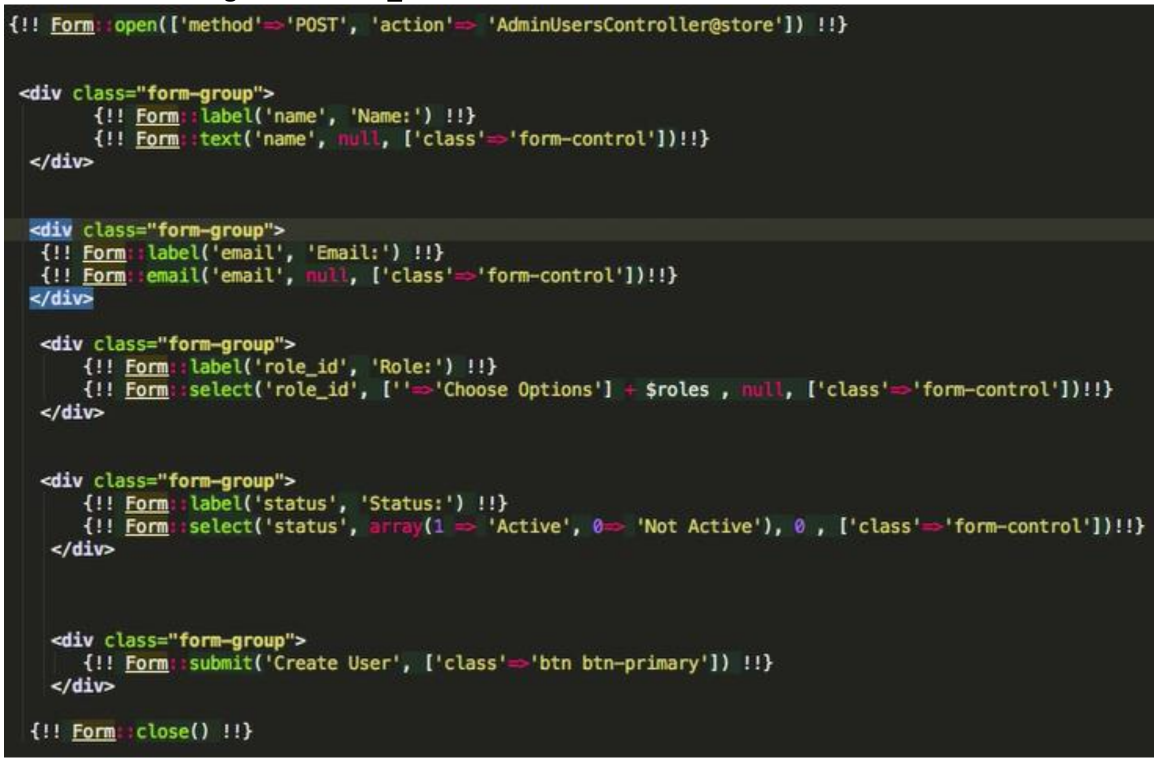


```
<tbody>

@if($users)

    @foreach($users as $user)

   <tr>
      <td>{{$user->id}}</td>
      <td>{{$user->name}}</td>
      <td>{{$user->email}}</td>
      <td>{{$user->role->name}}</td>
       <td>{{$user->is_active == 1 ? 'Active' : 'Not Active' }}</td>
      <td>{{$user->created_at->diffForHumans()}}</td>
      <td>{{$user->updated_at->diffForHumans()}}</td>
   </tr>

    @endforeach
```

12- Laravel collective html package
    Visit : https://laravelcollective.com/docs/5.1/html
    Follw the steps to install laravel collective package.
13- Create User Page
    Go to create function :

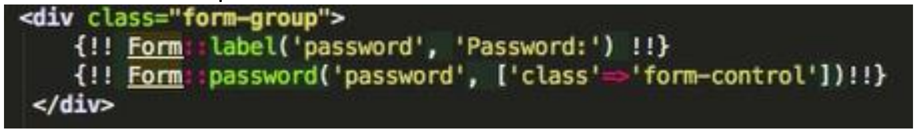
```
public function create()
{
    $roles = Role::lists('name', 'id')->all();
    return view('admin.users.create',compact('roles'));
}
```

    then go to create.blade.php
    make sure to change status to is_active



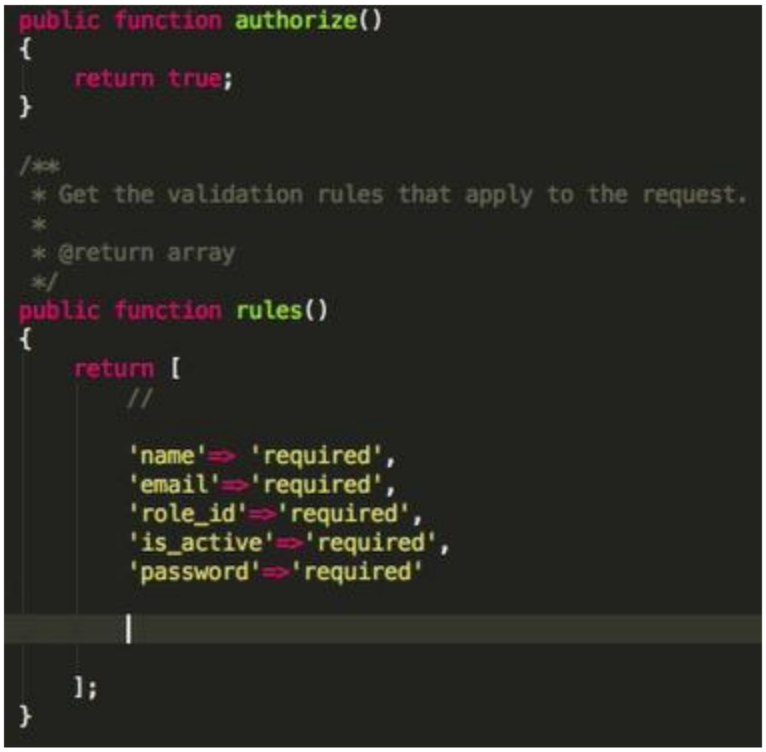
```
{!! Form::open(['method'=>'POST', 'action'=> 'AdminUsersController@store']) !!}


 <div class="form-group">
        {!! Form::label('name', 'Name:') !!}
        {!! Form::text('name', null, ['class'=>'form-control'])!!}
  </div>


  <div class="form-group">
   {!! Form::label('email', 'Email:') !!}
   {!! Form::email('email', null, ['class'=>'form-control'])!!}
  </div>

   <div class="form-group">
       {!! Form::label('role_id', 'Role:') !!}
       {!! Form::select('role_id', [''=>'Choose Options'] + $roles , null, ['class'=>'form-control'])!!}
   </div>


   <div class="form-group">
       {!! Form::label('status', 'Status:') !!}
       {!! Form::select('status', array(1 => 'Active', 0=> 'Not Active'), 0 , ['class'=>'form-control'])!!}
    </div>



    <div class="form-group">
       {!! Form::submit('Create User', ['class'=>'btn btn-primary']) !!}
    </div>

  {!! Form::close() !!}
```

14- Password Field and Custom Request:
    Add Text field for password afer status



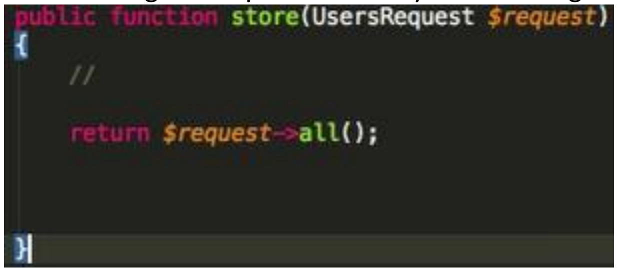
```
 <div class="form-group">
     {!! Form::label('password', 'Password:') !!}
     {!! Form::password('password', ['class'=>'form-control'])!!}
 </div>
```

    Then you can make request file :
    Php artisan make:request UsersRequest



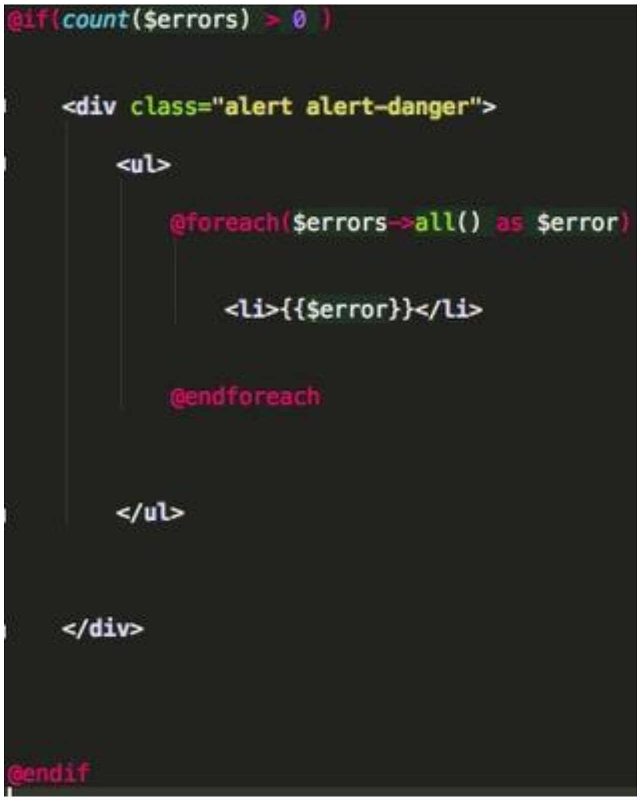
```php
public function authorize()
{
    return true;
}

/**
 * Get the validation rules that apply to the request.
 *
 * @return array
 */
public function rules()
{
    return [
        //

        'name'=> 'required',
        'email'=>'required',
        'role_id'=>'required',
        'is_active'=>'required',
        'password'=>'required'


    ];
}
```

Afetr making the request file now you can change the request in the store function



```php
public function store(UsersRequest $request)
{
    //

    return $request->all();


}
```

15- Displaying errors and including with blade:
Create Foder called includes and add new blade file on it named form_error:

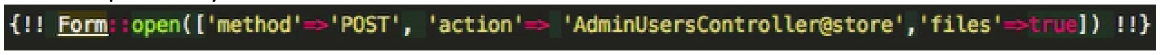


```
@if(count($errors) > 0 )

    <div class="alert alert-danger">

        <ul>

            @foreach($errors->all() as $error)

                <li>{{$error}}</li>

            @endforeach


        </ul>


    </div>



@endif
```

Then Go To create Page at the end of it and add this line :
@include('includes.form_error');

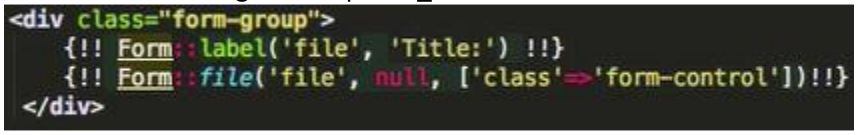
16- Adding upload file feature to form :

- First go to database and add column named [photo_id]
- Add files option to your form:



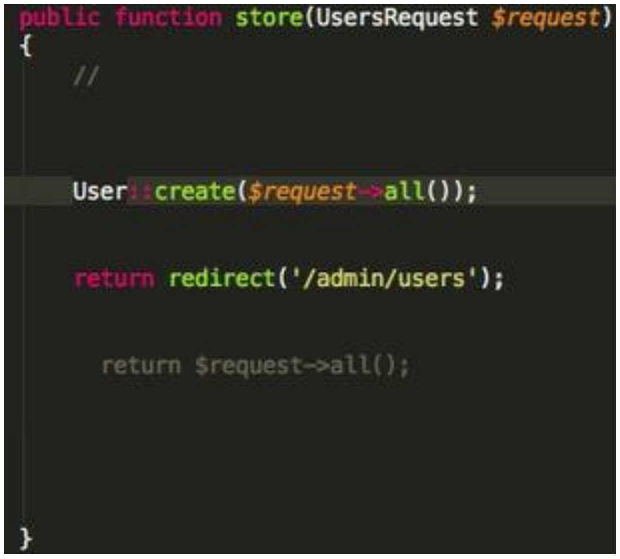
```
{!! Form::open(['method'=>'POST', 'action'=> 'AdminUsersController@store','files'=>true]) !!}
```

- Add file field to make the upload after password :
  Make Sure To Change file to photo_id



```
<div class="form-group">
    {!! Form::label('file', 'Title:') !!}
    {!! Form::file('file', null, ['class'=>'form-control'])!!}
 </div>
```

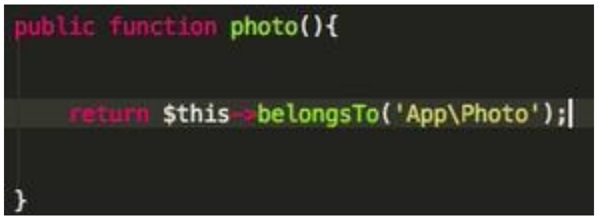
- Go To Store Method :DON'T FORGET MASS ASSIGNMENTS



```
public function store(UsersRequest $request)
{
    //


    User::create($request->all());


    return redirect('/admin/users');


      return $request->all();



}
```

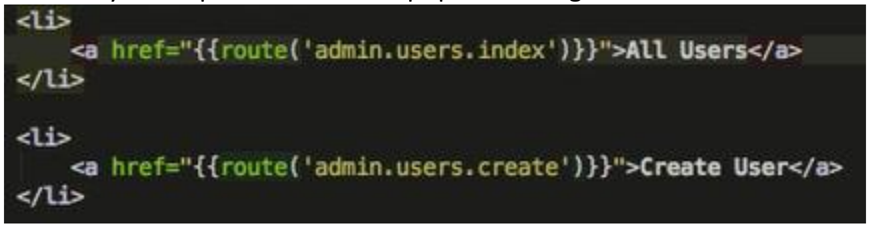
17- User photos migration - relation - mass-assignment:

- Php artisan make:model Photo –m
- Go To Photo Migrate File and add Column named file as string
- Go To Photo Model and Add
  Protected $fillable=['file'];
- Add The Relation into User Model:



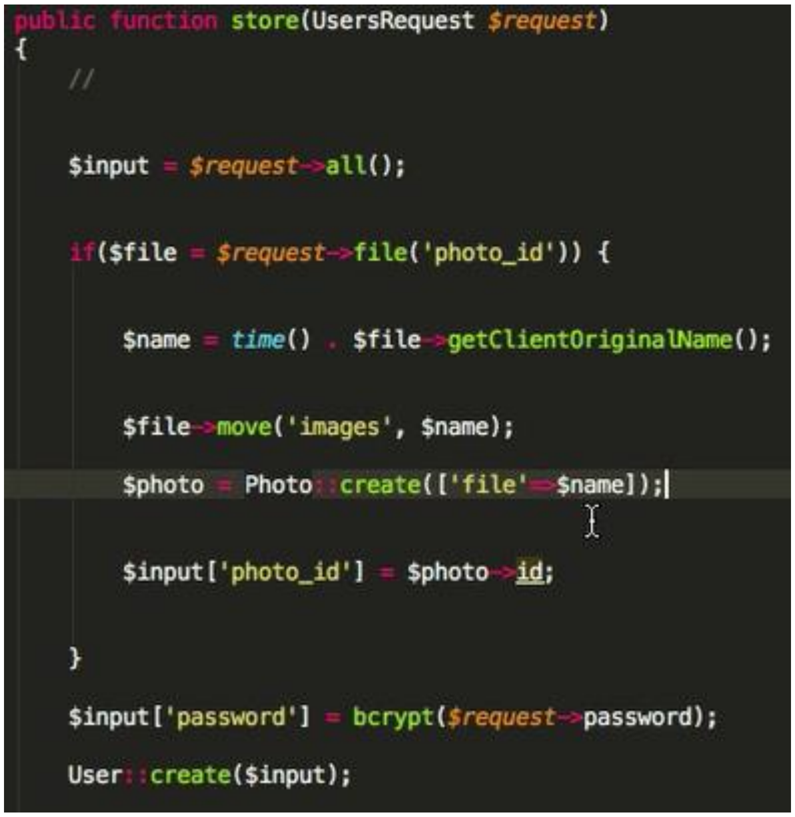
```
public function photo(){

    return $this->belongsTo('App\Photo');

}
```

18- Creating links :
Go To Layouts Open admin.blade.php and change links:



```
<li>
    <a href="{{route('admin.users.index')}}">All Users</a>
</li>

<li>
    <a href="{{route('admin.users.create')}}">Create User</a>
</li>
```

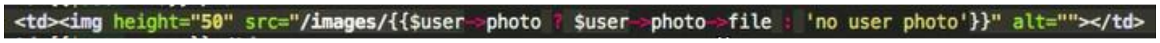
19- Adding User With Photo Into DataBase :



```php
public function store(UsersRequest $request)
{
    //


    $input = $request->all();


    if($file = $request->file('photo_id')) {


        $name = time() . $file->getClientOriginalName();


        $file->move('images', $name);

        $photo = Photo::create(['file'=>$name]);


        $input['photo_id'] = $photo->id;


    }

    $input['password'] = bcrypt($request->password);

    User::create($input);
```

- Add This Line at the end of last code :
  Return redirect('/admin/users');

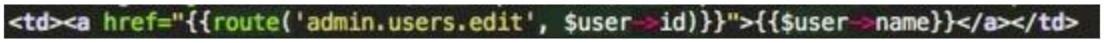
20- Displaying photos :
Go To Index.blade.php Page From Admin Folder :



```html
<td><img height="50" src="/images/{{$user->photo ? $user->photo->file : 'no user photo'}}" alt=""></td>
```

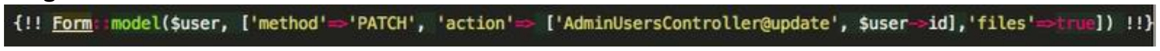
21- Edit users - setting up the form :

- Go To Index.blade.php Page From Admin Folder :
  Add Link On The Name:



```html
<td><a href="{{route('admin.users.edit', $user->id)}}">{{$user->name}}</a></td>
```

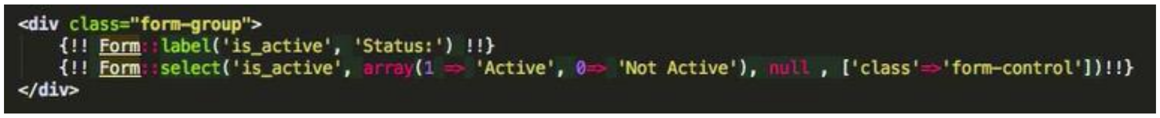
- Copy the create.blade.php into edit.blade.php
- Change the form Like this:



```php
{!! Form::model($user, ['method'=>'PATCH', 'action'=> ['AdminUsersController@update', $user->id],'files'=>true]) !!}
```

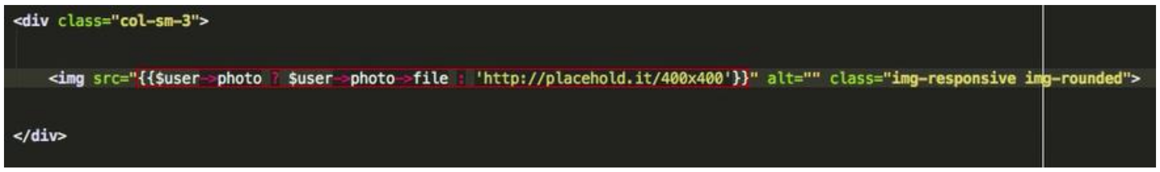
22- Edit user - displaying images and status :

- Change 0 to null

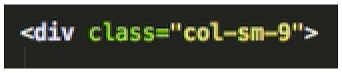


```html
<div class="form-group">
    {!! Form::label('is_active', 'Status:') !!}
    {!! Form::select('is_active', array(1 => 'Active', 0=> 'Not Active'), null , ['class'=>'form-control'])!!}
</div>
```

- To Display Photo :
  - Separate data and photo:
  - Make div for photo and other for data



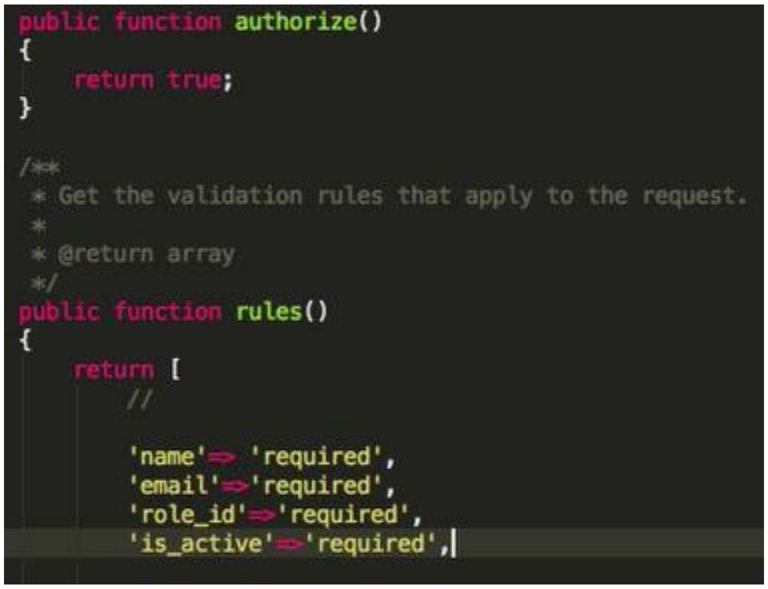
```
<div class="col-sm-3">

    <img src="{{$user->photo ? $user->photo->file : 'http://placehold.it/400x400'}}" alt="" class="img-responsive img-rounded">

</div>
```



```
<div class="col-sm-9">
```

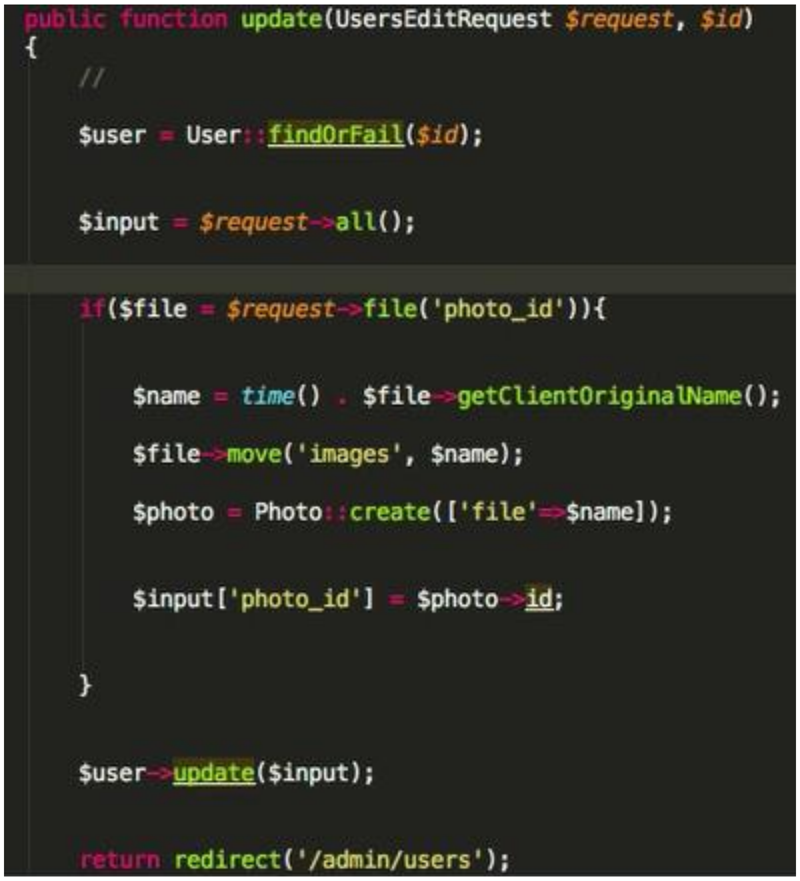
23- Update User :

- Go To edit.blade.php and put the form into div class="row"
- Make another request :
  Php artisan make:request UsersEditRequest



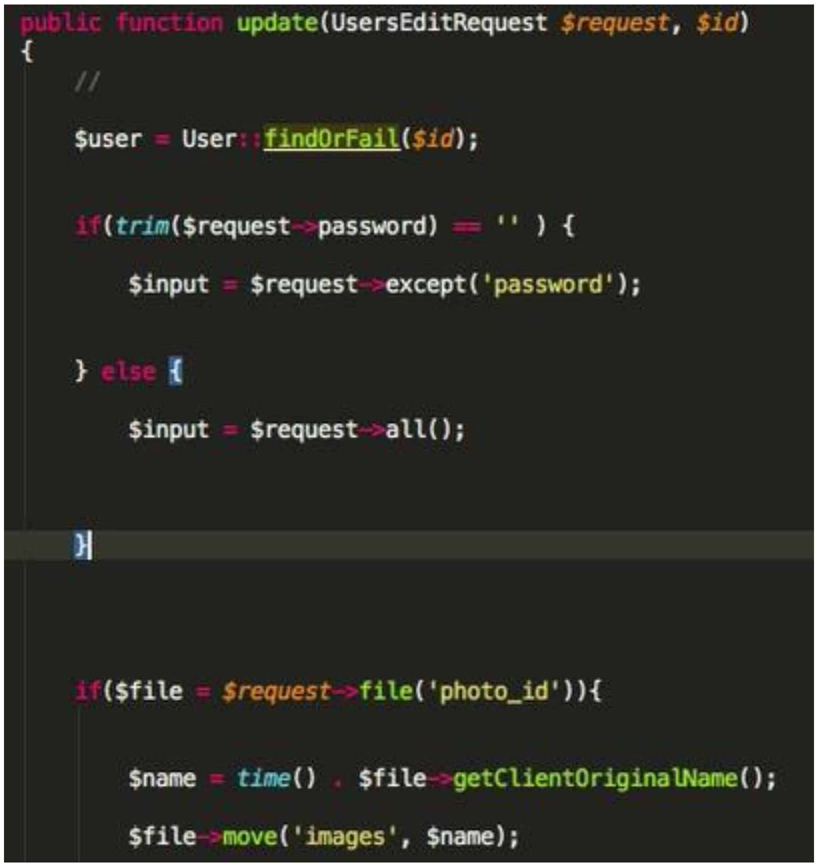
```
public function authorize()
{
    return true;
}

/**
 * Get the validation rules that apply to the request.
 *
 * @return array
 */
public function rules()
{
    return [
        //

        'name'=> 'required',
        'email'=>'required',
        'role_id'=>'required',
        'is_active'=>'required',
```

- Go To Update Function :

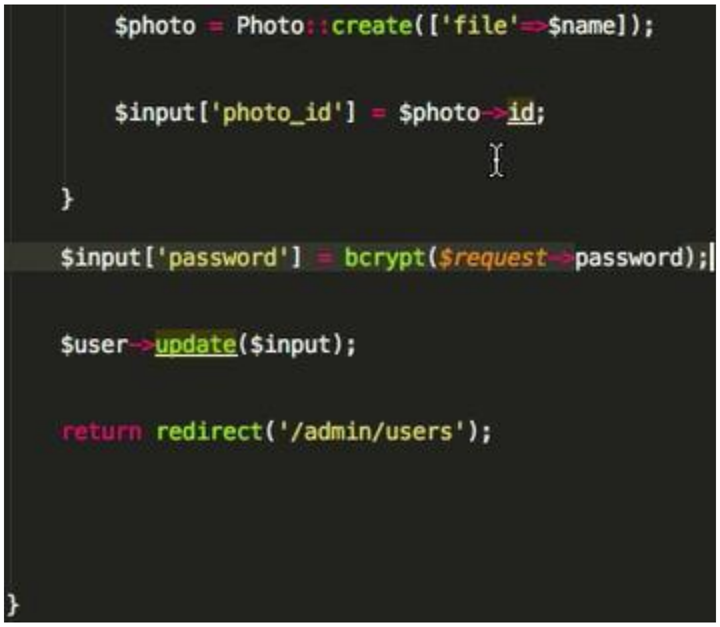


```php
public function update(UsersEditRequest $request, $id)
{
    //

    $user = User::findOrFail($id);


    $input = $request->all();


    if($file = $request->file('photo_id')){


        $name = time() . $file->getClientOriginalName();

        $file->move('images', $name);

        $photo = Photo::create(['file'=>$name]);


        $input['photo_id'] = $photo->id;


    }


    $user->update($input);


    return redirect('/admin/users');
```

- We Can Check for password if it empty:



```php
public function update(UsersEditRequest $request, $id)
{
    //

    $user = User::findOrFail($id);


    if(trim($request->password) == '' ) {

        $input = $request->except('password');

    } else {

        $input = $request->all();


    }



    if($file = $request->file('photo_id')){

        $name = time() . $file->getClientOriginalName();

        $file->move('images', $name);
```



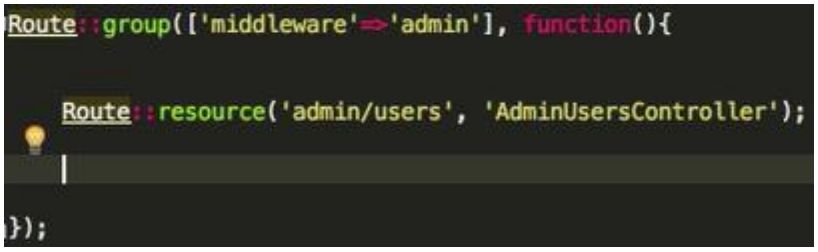
```php
        $photo = Photo::create(['file'=>$name]);

        $input['photo_id'] = $photo->id;


    }

    $input['password'] = bcrypt($request->password);

    $user->update($input);

    return redirect('/admin/users');



}
```

24- Security - middleware :

- Php artisan make:middleware Admin
- Go To kernel and add :

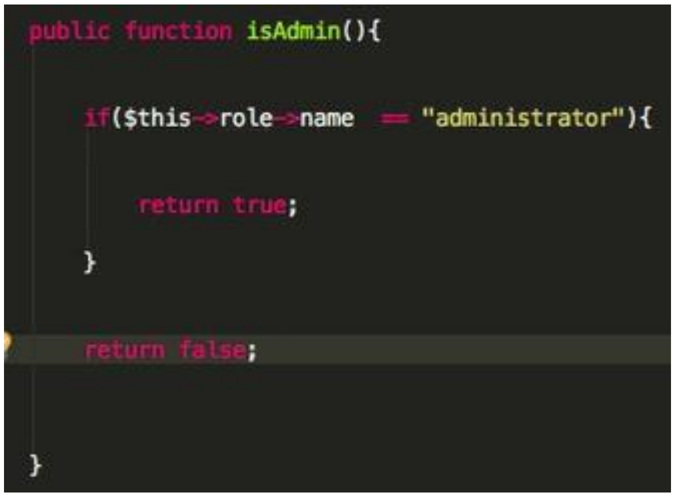
‘admin’=> \App\Http\Middleware\Admin::class,

- Go To Routes and Create Route Group:

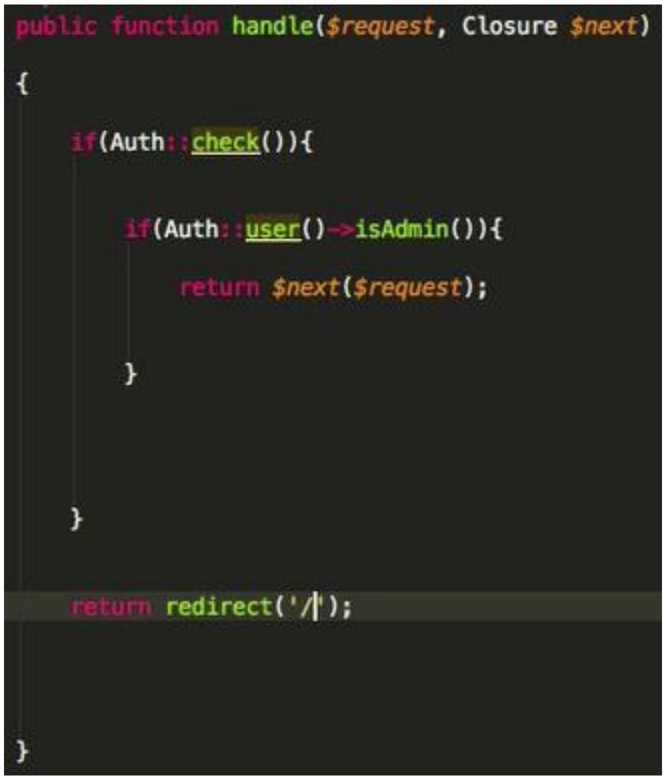


```
Route::group(['middleware'=>'admin'], function(){

    Route::resource('admin/users', 'AdminUsersController');


});
```

- Into user class :



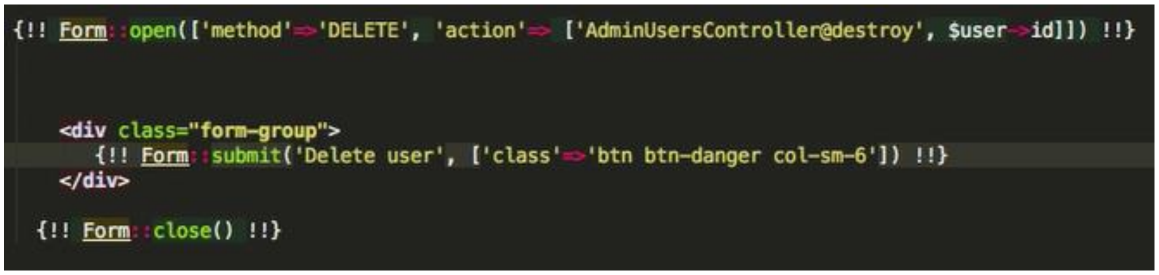
```
public function isAdmin(){

    if($this->role->name  == "administrator"){

        return true;

    }

    return false;


}
```



```
public function handle($request, Closure $next)

{

    if(Auth::check()){


        if(Auth::user()->isAdmin()){

            return $next($request);


        }




    }


    return redirect('/');




}
```

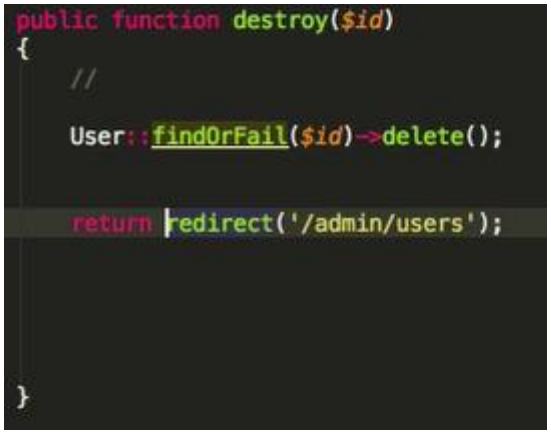
25- Deleteing Users :

- Go To Edit and Make a from contains a button fo deleting



```
{!! Form::open(['method'=>'DELETE', 'action'=> ['AdminUsersController@destroy', $user->id]]) !!}



    <div class="form-group">
        {!! Form::submit('Delete user', ['class'=>'btn btn-danger col-sm-6']) !!}
    </div>

  {!! Form::close() !!}
```

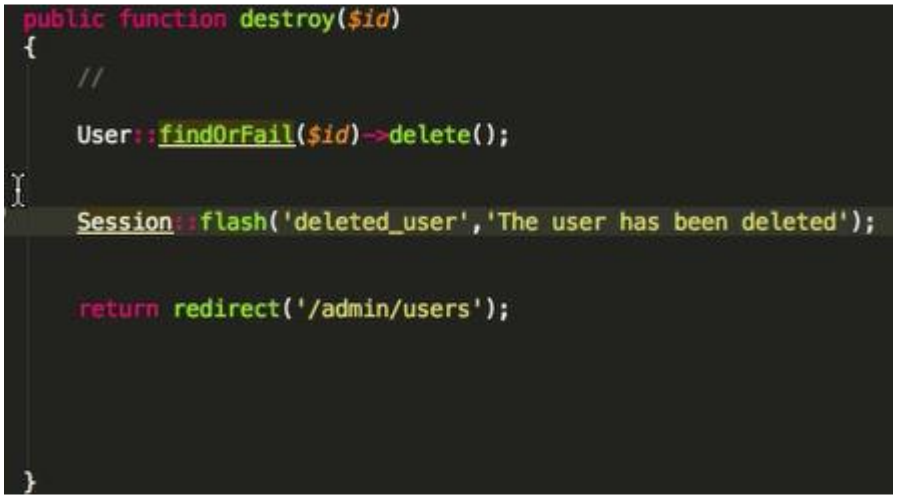
- Go To Controller.Destroy() :



```
public function destroy($id)
{
    //

    User::findOrFail($id)->delete();


    return redirect('/admin/users');





}
```

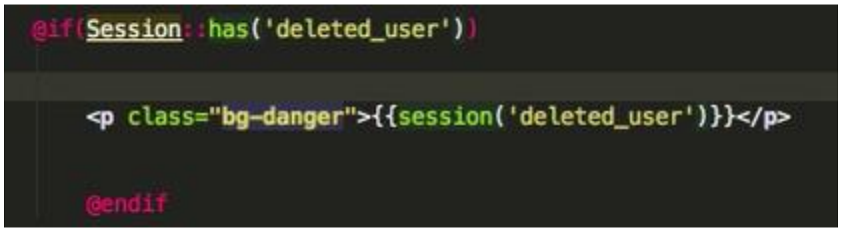
26- Messages and feedbacks :

- Change destroy method to be like this :



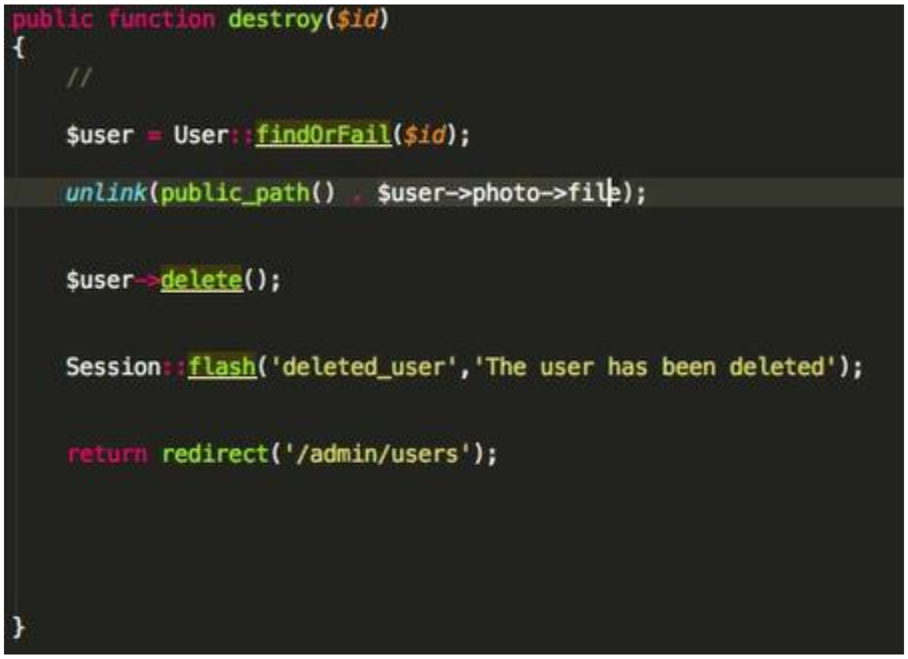
```
public function destroy($id)
{
    //

    User::findOrFail($id)->delete();

    Session::flash('deleted_user','The user has been deleted');

    return redirect('/admin/users');



}
```

- Then you can go to index.blade.php :



```
@if(Session::has('deleted_user'))

    <p class="bg-danger">{{session('deleted_user')}}</p>

    @endif
```

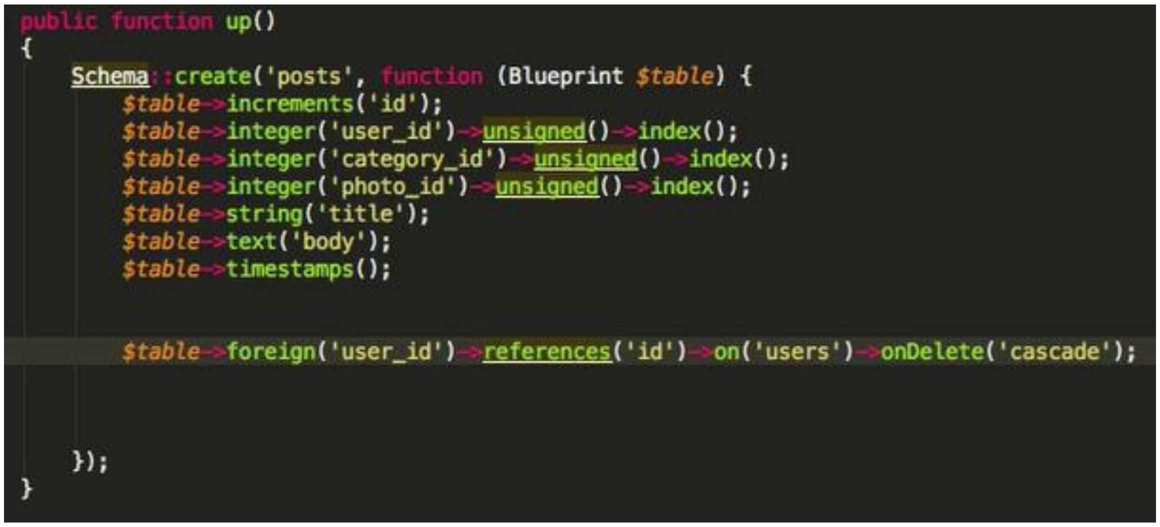
27- Delete User Image From Images :



```
public function destroy($id)
{
    //

    $user = User::findOrFail($id);

    unlink(public_path() . $user->photo->file);

    $user->delete();

    Session::flash('deleted_user','The user has been deleted');

    return redirect('/admin/users');



}
```

Posts Admin

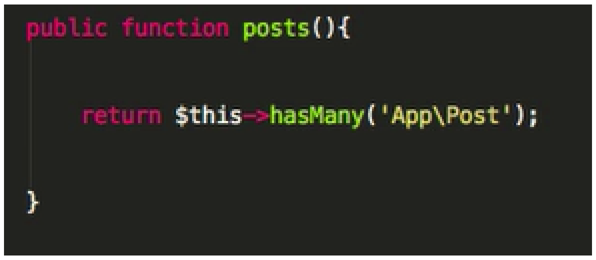
1- Make a controller named AdminPostsController
2- Make route for it
3- Fix links in the layout
4- Make Posts Model and migration



```
public function up()
{
    Schema::create('posts', function (Blueprint $table) {
        $table->increments('id');
        $table->integer('user_id')->unsigned()->index();
        $table->integer('category_id')->unsigned()->index();
        $table->integer('photo_id')->unsigned()->index();
        $table->string('title');
        $table->text('body');
        $table->timestamps();


        $table->foreign('user_id')->references('id')->on('users')->onDelete('cascade');



    });
}
```

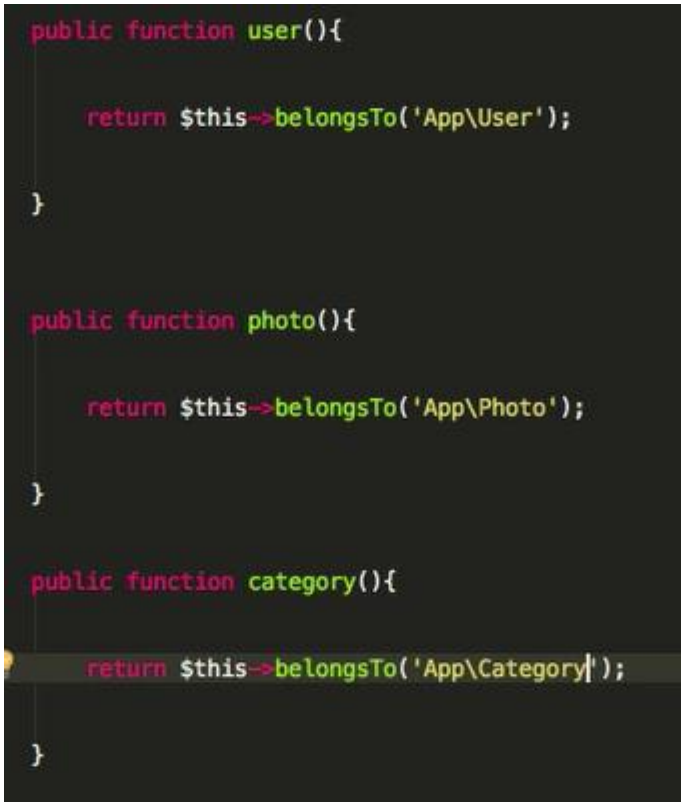
5- Make Categories Model and migration
6- RelationShips:

In user model :



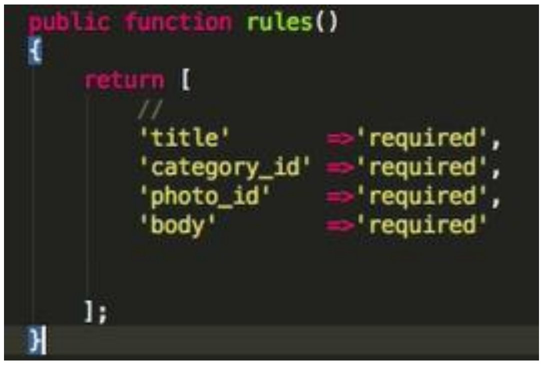
```
public function posts(){


    return $this->hasMany('App\Post');


}
```

In Post Model:



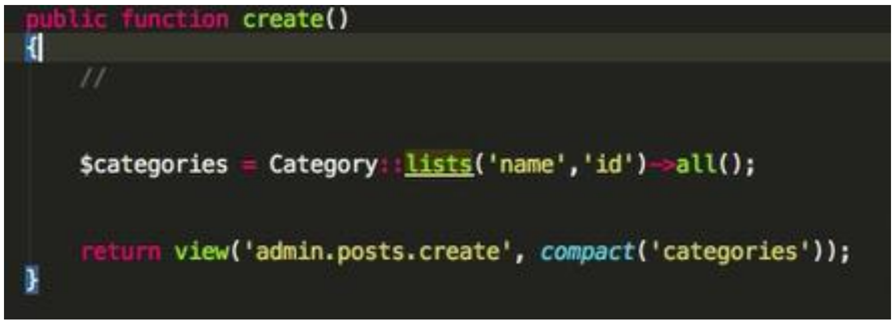
```
public function user(){

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

}


public function photo(){

    return $this->belongsTo('App\Photo');

}

public function category(){

    return $this->belongsTo('App\Category');

}
```

7- Request fro create :

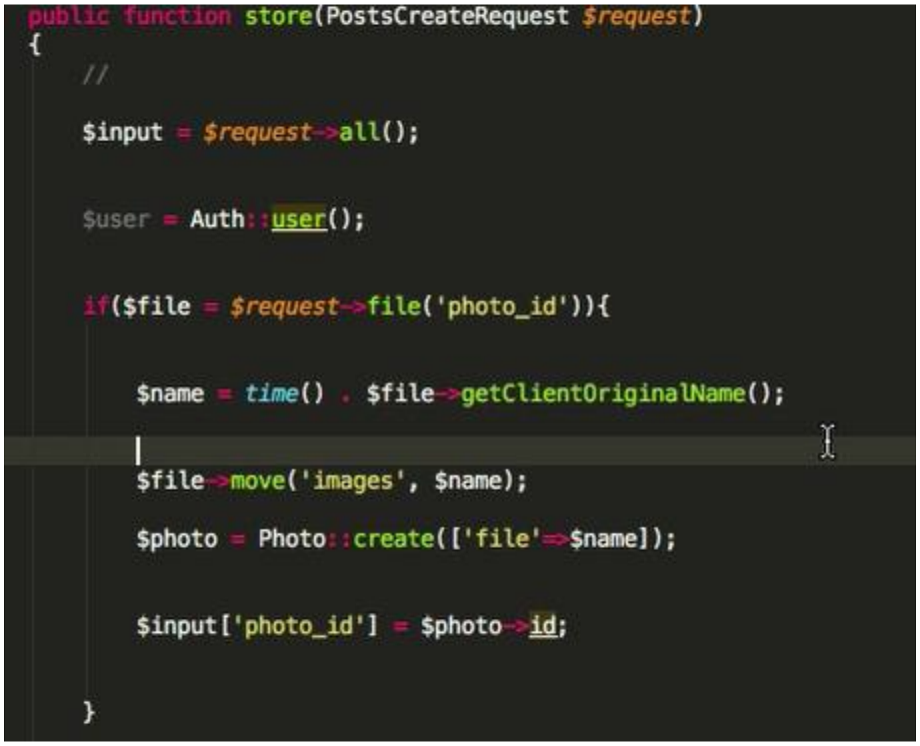


```
public function rules()
{
    return [
        //
        'title'       =>'required',
        'category_id' =>'required',
        'photo_id'    =>'required',
        'body'        =>'required'

    ];
}
```

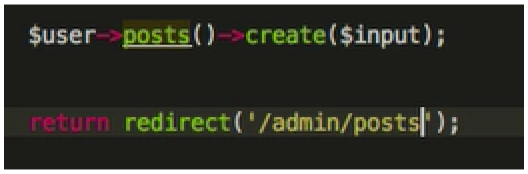
8- Create post With category :



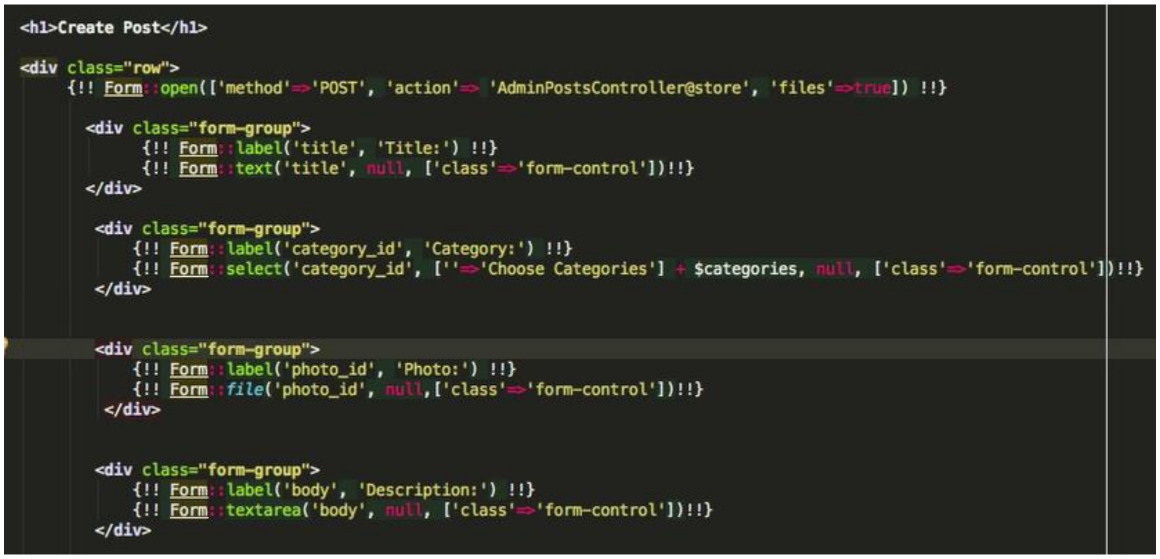
```
public function create()
{
    //


    $categories = Category::lists('name','id')->all();


    return view('admin.posts.create', compact('categories'));
}
```



```php
public function store(PostsCreateRequest $request)
{
    //

    $input = $request->all();


    $user = Auth::user();


    if($file = $request->file('photo_id')){


        $name = time() . $file->getClientOriginalName();


        $file->move('images', $name);

        $photo = Photo::create(['file'=>$name]);


        $input['photo_id'] = $photo->id;


    }
```



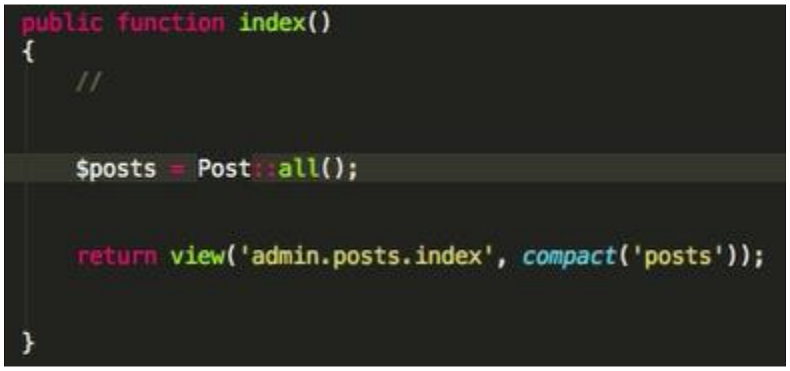
```php
$user->posts()->create($input);


return redirect('/admin/posts');
```



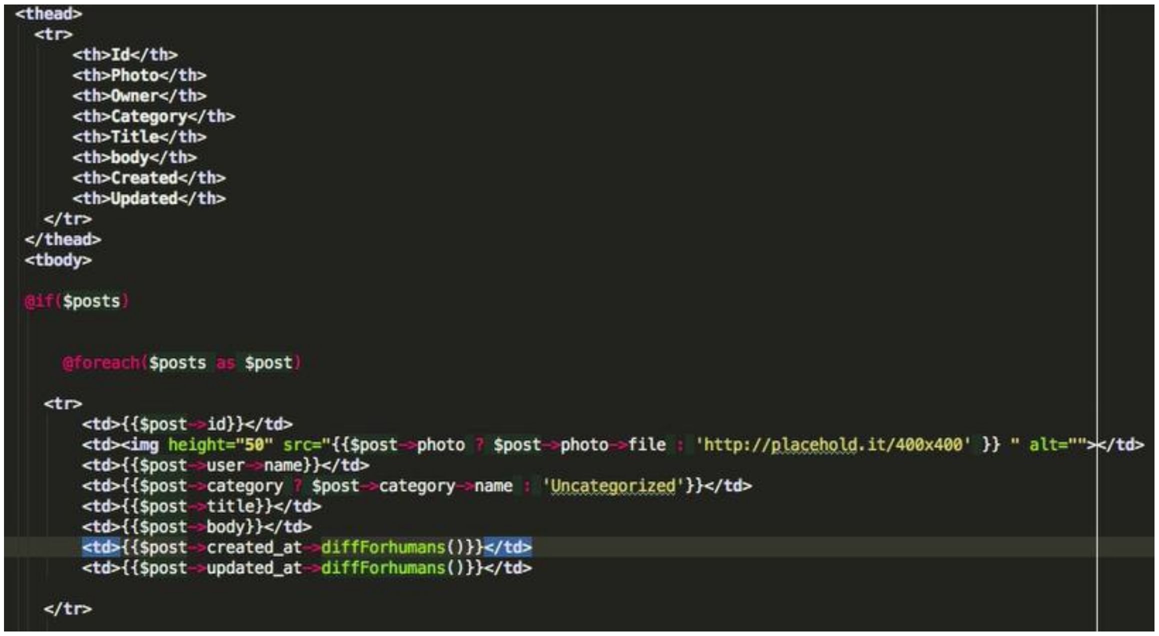
```html
<h1>Create Post</h1>

<div class="row">
    {!! Form::open(['method'=>'POST', 'action'=> 'AdminPostsController@store', 'files'=>true]) !!}

      <div class="form-group">
            {!! Form::label('title', 'Title:') !!}
            {!! Form::text('title', null, ['class'=>'form-control'])!!}
      </div>

       <div class="form-group">
           {!! Form::label('category_id', 'Category:') !!}
           {!! Form::select('category_id', [''=>'Choose Categories'] + $categories, null, ['class'=>'form-control'])!!}
       </div>


       <div class="form-group">
           {!! Form::label('photo_id', 'Photo:') !!}
           {!! Form::file('photo_id', null,['class'=>'form-control'])!!}
        </div>


       <div class="form-group">
           {!! Form::label('body', 'Description:') !!}
           {!! Form::textarea('body', null, ['class'=>'form-control'])!!}
       </div>
```

After form close please add :
@include('includes.form_error');

9- Show post with category



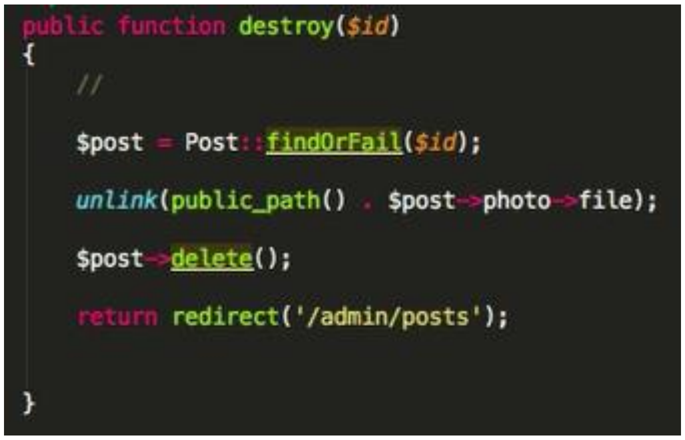
```php
public function index()
{
    //

    $posts = Post::all();

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

}
```



```html
<thead>
  <tr>
      <th>Id</th>
      <th>Photo</th>
      <th>Owner</th>
      <th>Category</th>
      <th>Title</th>
      <th>body</th>
      <th>Created</th>
      <th>Updated</th>
   </tr>
 </thead>
 <tbody>

 @if($posts)


     @foreach($posts as $post)

   <tr>
       <td>{{$post->id}}</td>
       <td><img height="50" src="{{$post->photo ? $post->photo->file : 'http://placehold.it/400x400' }} " alt=""></td>
       <td>{{$post->user->name}}</td>
       <td>{{$post->category ? $post->category->name : 'Uncategorized'}}</td>
       <td>{{$post->title}}</td>
       <td>{{$post->body}}</td>
       <td>{{$post->created_at->diffForhumans()}}</td>
       <td>{{$post->updated_at->diffForhumans()}}</td>

   </tr>
```

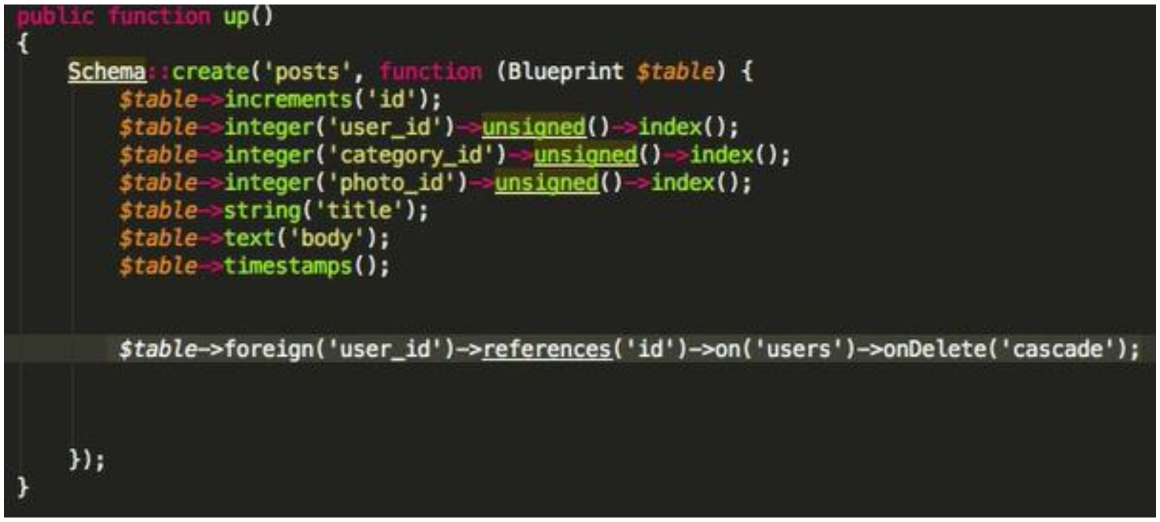
10- Delete Post :

- Add button in the edit form

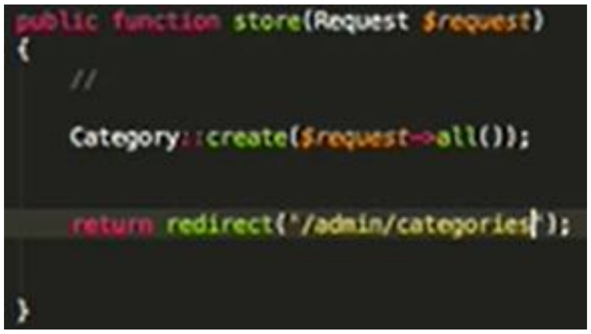


```php
public function destroy($id)
{
    //

    $post = Post::findOrFail($id);

    unlink(public_path() . $post->photo->file);

    $post->delete();

    return redirect('/admin/posts');


}
```

11- Delete With Relations:



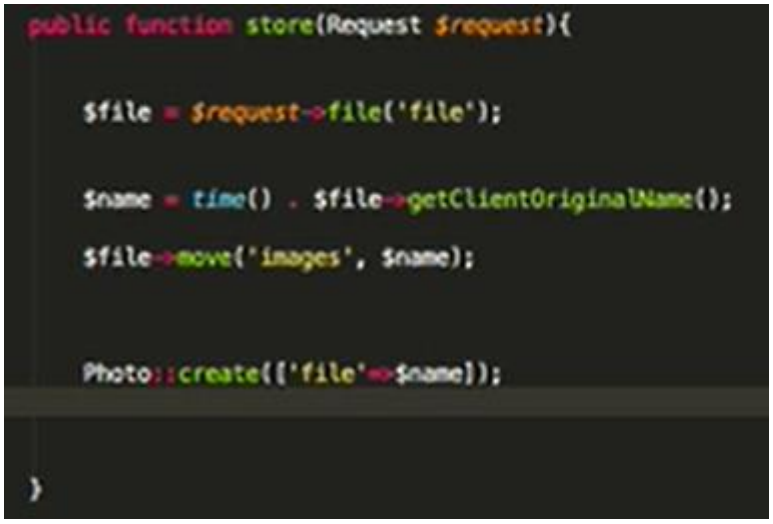
```
public function up()
{
    Schema::create('posts', function (Blueprint $table) {
        $table->increments('id');
        $table->integer('user_id')->unsigned()->index();
        $table->integer('category_id')->unsigned()->index();
        $table->integer('photo_id')->unsigned()->index();
        $table->string('title');
        $table->text('body');
        $table->timestamps();


        $table->foreign('user_id')->references('id')->on('users')->onDelete('cascade');



    });
}
```



```
public function store(Request $request)
{
    //

    Category::create($request->all());


    return redirect('/admin/categories');


}
```

Categories Admin :

- add AdminCategoriesController
- make route for this controller
- set up the index view to show all categories
- Display all categories in a table



```php
public function store(Request $request){

    $file = $request->file('file');

    $name = time() . $file->getClientOriginalName();

    $file->move('images', $name);


    Photo::create(['file'=>$name]);


}
```