**Laravel Blog Post Website**

1. Wamp ဖြင့္

2. getcompser.org ကေန window Installer ကို ေရြး Down ပီး Install လုပ္ ေနရင္ choose the command line php မွာ wamp/bin/php/php 7.3.12/php.exe ကို ေရြးေပးရမယ္။

3. wamp ထဲမွာ Shift + Right Click ႏွိတ္ပီး open command window here ကို ႏွိတ္လိုက္ power shall ေပါလာမယ္ အဲ့ထဲမွာ php-v (or)composer-v ကို ႏွိတ္ရင္ php (or)composer အေၾကာင္းေပါလာရင္ composer install လုပ္တာ ေအာင္ျမင္တယ္ ။

4. wamp/www/folder အလြတ္တခုေဆာက္ထားပီး powershall ေခါပီး Laravel webite ကေန Composer ကေန laravel ကို Install လုပ္တဲ့ code ကိုရွာပီး Run ေပး folder ထဲၾကည့္ file ေရာက္။

5. Short Url လုပ္ခ်င္ရင္ server.php ကို copy ပြားပီး index.php လို႔ေျပာင္း public folder ထဲက htaccess file ကိုအျပင္ထုတ္ အာဆ့ိုရင္(localhost/bgpost /public ဆိုတာထည့္မေရးရေတာ့ဘူး )bgpost ထိပဲေရးယုံပဲ။

6. MD bootstrap Cdn ကို Down - Zip ဖိုင္ရမယ္။

7. Visual Studio Code လို႔ Google တြင္ရွာ Install လုပ္ပီး Extension မွာ laravel blade snippets ရွာ install

8. node.js software & npm software down install

9. Laravel collective form နဲ႕ေ၇းမယ္ဆိုရင္ <https://laravelcollective.com/docs/6.x/html> အဲ့ထဲက composer code ကို copy၊ powershall မွာ run / ၀င္မ၀င္ကို ကို ့ lar project ထဲက composer.json file မာ ၾကည့္လို႔ရတယ္

----------------------------------------- **Project စေရးမယ္** --------------------------------------------------

အရင္ဆုံး user table ထဲမွာ admin ဟုတ္ မဟုတ္ စစ္ဖို႔ (3)ေၾကာင္းထည့္

$table->boolean('admin')->default(0);

$table->boolean('confirmed')->default(0);

$table->boolean('active')->default(0);

Db ထဲမွာ Category ေဆာက္ဖို႔ (သို႔) Post (သို႔) တခုခုေဆာက္ေတာ့မယ္ဆိုရင္ model /migration/controller ရွိရမယ္ အဲ့တြက္

php artisan make:model Post –mc (Model၊Migrations၊Controller သုံးခုလုံးတခါတည္းေဆာက္ပီးသားဖစ္သြားမယ္)

၁။ migration folder ထဲက category ထဲမွာ ကိုထည့္ခ်င္တဲ့ Table column ထည့္္ / ၂။ Model file ထဲမွာ အဲ့ Table column ေတြရဲ႕ name တစ္ခုခ်င္းစီထည့္ / ၃။ web.php မာ ေတာ့ Route::resource('category',CategoryController::class); တစ္ခု ေရးထား /

၄။ Controller မွာေတာ့

use App\Models\Category;

use Validator; validator ေတြနဲ႕ စစ္မာဖစ္လို႔

use Session; Session flash ေတြလဲ သုံးမာဖစ္လို႔

**Category Section (Admin Site)**

Category ေတြ Create လုပ္ဖို႔တြက္ admin folder ထဲမွာ main.balde.php ႀကီးေဆာက္ထား။ ေနာက္ပီး category folder အသစ္ေဆာက္ပီး အဲ့ထဲမွာ creat.blade.php file ေဆာက္ ပီိး အဲ့ထဲ မာ admin ထဲက main.blade.php extends လုပ္ ပီး section ထဲမွာလည္း create form တခုေဆာက္ထား

**Create Method**

url မွာ category/create ေရးရင္ create.blade.php သြားဖို႔တြက္ Controller ထဲမွာ

public function create(){

            return view('category.create'); < view/category/create.blade.php ကိုသြားပါလို႔ေရးထားတာ

        }

**Store Method**

Create Category button ကို ႏွိတ္ရင္ db ထဲ store ဖစ္ဖို႔ Category Controller ထဲမွာ အလိုေရးေပးရတယ္။

       public function store(Request $request){

             /\*   Write Second  \*/

            $validator= Validator::make($request->all(),[

                'name'=>'required | max:30 | min:3'

            ]);

            if($validator->fails()){

                return redirect('category/create')

                ->withInput()

                ->withErrors($validator);

            }

            /\*   Write First  \*/

            $category=new Category; <- Modle ထဲက Category နဲ႕ name တူရမယ္

            $category->name = $request->name; <- category table ထဲက name col မွာ user request

ေတြရဲ႕name ေတြထည့္မယ္

            $category->save();

            Session::flash('category\_create', 'New Category is Created'); <-alert ထုတ္ဖို႔

            return redirect('category/create'); <-submit btn ႏွိတ္ပီးရင္ category ထဲက create file ကိုပဲ ျပန္သြားပါ

        }

'New Category is Created' Alert ထုတ္ဖို႔အတြက္ category/create file ထဲမွာပဲ ျပခ်င္တဲ့ေနရာမွာ ဒီကုဒ္ကို ထည့္ေပး

  @if(Session::has('category\_create')) <-controller ထဲမွာ ေရးထားတဲ့ session ::flash ရဲ႕ Name

                    <div class="alert alert-success">

                        <em>{!!session('category\_create')!!}</em>

                        <button type="button" class="close" date-dismiss='alert' aria-label="Close">

                            <span aria-hidden="true">&times</span>

                        </button>

                    </div>

                @endif

**Index Method**

Db ထဲက category ေတြ ထုတ္ပီးျပဖို႔ category folder ထဲမွာ main.blade.php ေဆာက္၊ ပီးရင္ CategoryController ထဲမွာ

     public function index(){

            $categories=Category::all(); <- Category table ထဲက အားလုံးကို $categories လို႔ ေပးလိုက္တယ္

            return view('category.main') <- category folder ထဲက main.blade.php ကိုသြားမယ္

            ->with('categories',$categories); <-သြားရင္   categories ကိုပါေခၚသြားပါ

}

category/main file ထဲမွာ

 <tbody>

@foreach($categories as $category)

     <tr>

      <td>{!! $category->name !!}</td>

       <td><a href="{!!url('category/'.$category->id.'/edit')!!}">Edit</a></td>

        {!! Form::open(array('url'=>'category/'.$category->id,'method'=>"DELETE")) !!}

         {!! csrf\_field() !!}

          {!! method\_field('DELETE')!!}

<td><button class="btn btn-danger">Delete</button></td>

 {!! Form::close() !!}

         </tr>

  @endforeach

</tbody>

**Edit Method**

**CategoryController ထဲမွာ**

public function edit($id){

            $categories=Category::findorFail($id);

            return view ('category.edit')

            ->with('categories',$categories);

        }

category/edit.blade.php file တခုေဆာက္ (create.blade.php file ကိုပဲ copy ကူးပီး edit လို႔ေျပာင္းလိုက္ )

eidt.blade.php file ေရာက္သြားရင္ form ထဲမွာ category name ကို တန္းျပဖို႔အတြက္Form Model တေၾကာင္း ေရးေပးရတယ္။form open ေရးဖို႔မလို။

{!! Form:: model($categories, array('route'=>array('category.update',$categories->id),'method'=>'PUT')) !!}

 {!! Form::label('name','Edit Category') !!}

            {!! Form::text('name', null, array('class'=>'form-control'))!!}

             <br>

              {!! Form::submit('Update Category', array('class'=>'secondary-cart-btn'))!!}

              {!! Form::close() !!}

**Update Method**

**Update Category ကို ႏွိတ္ရင္ Update ဖစ္ဖို႔တြက္ CategoryController ထဲမွာ store method ထဲက ဟာေတြ copy ယူပီးေရးလဲရတယ္တူသေလာက္ရွိတယ္။**

        public function update(Request $request, $id){

            $validator= Validator::make($request->all(),[

                'name'=>'required | max:30 | min:3'

            ]);

            if($validator->fails()){

                return redirect('category/'.$categories->id.'/edit')

                ->withInput()

                ->withErrors($validator);

            }

            $category=Category::find($id);

            $category->name = $request->Input('name');

            $category->save();

            Session::flash('category\_update','Categories is Updated');

            return redirect('category');

        }

**Destory Method**

**CategoryController ထဲမွာ destroy method ေရးတာ**

public function destroy($id){

            $categories=Category::find($id);

            $categories->delete();

            Session::flash('category\_delete','Categories is Deleted');

            return redirect('category');

        }

**Validation FORM**

**CategoryController ရဲ႕ store method ထဲမွာ validator စစ္ထားတာနဲ႕ မရေသးဘူး**

**၁။folder တစ္ခုအသစ္ေဆာက္ပီးေတာ့ errors.blade.php ဖိုင္ တခုေဆာက္ပီး အဲ့ထဲမွာ ေအာက္ကတိုင္း ေရးေပးထား ၂။ကိုလုပ္ခ်င္တဲ့ form open အေပါမွာ @include(‘common/errors’) ေရးေပးရတယ္ အဒါမွ error ေတြလာျပတာ။**

@if (count($errors)>0)

{{-- Form error list --}}

<div class="alert alert-danger">

    <strong>Something is wrong</strong>

    <ul>

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

            <li>{!! $error !!}</li>

        @endforeach

    </ul>

</div>

@endif

**Post Section (Admin Site)**

php artisan make:model Post –mc (Model၊Migrations၊Controller သုံးခုလုံးတခါတည္းေဆာက္ပီးသားဖစ္သြားမယ္)

Post ေတြ crud လုပ္ပုံလုပ္နည္းနဲ့ form validation / error ေတြစစ္တာက အေပၚက category အတိုင္းပဲ

**create Post form မွာ category ကို selection အေန နဲ႕ ထုတ္ျပရမယ္ အဲ့တြက္**

၁။ PostController ထဲမွာ create method ထဲမွာ

public function create(){

        $categories=array();

        foreach(Category::all() as $category){

           $categories[$category->id]= $category->name;

        }

        return view('post.create')

        ->with('categories',$categories);

    }

၂။ Post/create file ရဲ႕ form မွာ

{!! Form::label('category\_id','Category Name') !!}

{!! Form::select('category\_id', $categories, array('class'=>'form-control'))!!}<br><br>

post create ကို click ႏွိတ္တဲ့အခါမွာ selection နဲ႕ ထုတ္ထားတဲ့ Category Name ေတြရဲ ့ id ေတြကို db ထဲက post table ထဲမွာေဆာက္ထားတဲ့ **'category\_id'** column ထဲမွာ ထည့္ခ်င္လို႔

**၁။ migration/posttable ထဲက** $table->id();  **ရဲ႕ေအာက္မွာ အဲ့ ၂ေၾကာင္းကို ထည့္ေပးရမယ္ ။** **အဲ့ ၂ေၾကာင္က ဘယ္ကရလဲဆိုရင္ laravel website /migration/foreign key constraint ထဲက ဟာ**

$table->unsignedBigInteger('category\_id'); <posts table ထဲမွာ 'category\_id' column ထည့္မယ္

$table->foreign('category\_id')->references('id')->on('categories'); < posts table ထဲမွာ category\_id column

ထဲမွာ categories table ထဲက 'id' ကိုထည့္မယ္ လို႔ေျပာတာ အဒါပီးရင္ migrate လုပ္လိုက္ေတာ့ အာဆ့ို

db ထဲမွာ post table ထဲက category\_id ကေသာ့အျဖဴပုံေလးနဲ႕ျပေနလိမ့္မယ္။

ပီးရင္model ထဲက post နဲ႕ category ၂ ဖိုင္ကို ခ်ိတ္ေပးရမယ္။

**၂။ model/Post ထဲမွာ**

public function category(){

        return $this->belongsTo('category');

    }

**၃။ model/Category ထဲမွာ**

public function posts(){

        return $this->hgasMany('post');

    }

**Post View (Client Site)**

**Db ထဲက post ေတြ အကုန္လုံးထုတ္ျပဖို႔တြက္**

၁။web.php မွာ Route index ေရးထား။

Route::get('/',[StoreController::class,'index']); <- get('/') ဆိုတာက homepg ကိုေျပာတာ urlbar မွာ ('/') ေရးလိုက္ရင္ Store Controller ထဲက index() မွာေရးတဲ့ အတိုင္း အလုပ္လုပ္ပါ လို႔ေျပာတာ။

၂။ storecontroller ထဲမွာ index() နဲ႕ ေရးတာက post ေတြအားလုံးျပတဲ့ home pg

public function index(){

        $categories=Category::all();

        return view('store.main') < route ထဲမွာေရးထားတဲ့ ('/' ဒီurl က) store/main.blade.php ကို သြားမယ္။

->with('posts', Post::orderBy('created\_at','DESC')->get())   <-To Show All Post by DESC

        ->with('categories',$categories);  /\* To Show All Category \*/

    }

Index () နဲ႕ေရးထာတာမ္ို႔ လို႔ foreach နဲ႕ loop ပတ္ေပးရမယ္ 'store.main' file ထဲမွာ Post ေရာ Category ရွိတဲ့ေနရာမွာ loop ပတ္ပီးထုတ္ေပး။

   @foreach ($posts as $post)

          <div class="card mb-4">

               <h2 class="card-title">{!! $post->title !!}</h2>

                <p class="card-text">{!! $post->short\_desc !!}</p>

                <a class="btn btn-primary" href="{{ url("/store/view/$post->id") }}">Read more</a>

          </div>

     @endforeach

**Post ေတြရဲ႕ Readmore ကိုႏွိတ္ရင္ relative post ပဲျပဖို႔အတြက္**

၁။Store/main file ကို copy လုပ္ပီး view.blade.php အရင္ ေဆာက္ထား

၂။Read More မွာ foreach **နဲ႕ ထုတ္ထားတဲ့ $post**  ရဲ႕ id ကို url နဲ႕ ထုတ္ေပးရတယ္

<a class="btn btn-primary" href="{{ url("/store/view/$post->id") }}">Read more →</a>

**၃။ web.php မွာ detail Route တခုေရးေပး**

Route::get('/store/view/{postid}',[StoreController::class,'detail']); <- url မွာ/store/view/{post ရဲ႕id ဘာပဲလာလာ} StoreC ထဲက detail() အတိုင္းအလုပ္လုပ္ပါ။ carli bracket { } မွာေရးတာက ဘာ id ဘဲလာလာလိုေ့ျပာတာ။

**StoreController ထဲမွာ detail() ေရး**

 public function detail($id){ <- $id က readmore မွာ ထုတ္ထားတဲ့ url ထဲက /$post->id ကိုေျပာတာ

ဘာလို႔လဲဆိုရင္ အဲ့ url နဲ႕ web.php မွာ detail route ေရးထားခဲ့လို႔ detail($id) နဲ႕တူတာ။

$categories=Category::all(); <- 'store.view ကိုသြားတဲ့ခါ Category ထဲက data ေတြပါ pass လုပ္ခ်င္လို႔ေရးထားတာ။

        $data= Post::find($id); <- url ဘားမွာ ရွိတဲ့ id ကို ယူရွာပီးေတာ့ သက္ဆိုင္ရာ post ေတြျပမာ။

        return view('store.view',['post'=>$data]) <- data pass လုပ္ဖို႔ ->with နဲ႕ မေရးပဲ arrary ပုံစံေျပာင္းေ၇းထား

တာ post ကို store.view ထဲမွာ{!!$post->title!!}လို႔ထုတ္ေပး႐ုံပဲ။

        ->with('categories',$categories); <- categoires ကိုေတာ့ store.view ထဲမွာ foreach နဲ႕ loop ပတ္

Category::all(); လို႔ေရးထားလို႔္။

    }

**Loop ပတ္ပီး ထုတ္ထားတဲ့ Category ေတြကို ႏွိတ္ရင္ relative post ေတြပဲ ျပခ်င္ရင္ ။**

**၁။** url("/store/category/$category->id ေရးေပးထားရတယ္။

@foreach ($categories as $category)

 <ul class="list-unstyled mb-0">

<li><a href="{{url("/store/category/$category->id")}}">{!! $category->name !!}</a></li>

   </ul>

@endforeach

၂။ အဲ့ url("/store/category/$category->id")  **အတြက္ Route ေရးေပးရမယ္**

Route::get('/store/category/{catid}',[StoreController::class,'getCategory']);<-url မွာ/store

/category /{category ရဲ႕id ဘာပဲလာလာလိုေျပာတာ}

၃။ StoreController ထဲမွာ

public function getCategory($id){ <- $id က web.php မွာ ေရးထားတဲ့ url ('/store/category/{category

ရဲ႕id ဘာပဲလာလာလိုေျပာတာ}')

  $categories=Category::all();

  $posts=DB::table('posts')->where('category\_id','=',$id)->get(); <- Db ထဲက post table ထဲက 'category\_id

က $id နဲ႕ တူတာကို get() နဲ႕ယူမယ္။

return view('store.category') <- store/category file ထဲမွာ datapass လုပ္မယ္

        ->with('posts',$posts)

        ->with('categories',$categories);

    }

**SEARCH POST**

**FORM**

<div class="col-lg-4">

            <!-- Search widget-->

   <div class="card mb-4">

        <div class="card-header">Search</div>

             <div class="card-body">

         {!! Form::open(array('url'=>'store/search','method'=>'get')) !!}

                    <div class="input-group">

          {!! Form::text('keyword',null, array('placeholder'=>'Search','class'=>'form-control'))!!}

           {!!Form::submit('search',array('class'=>'btn btn-primary'))!!}

                    </div>

                </div>

            </div>

**ROUTE ေရး**

Route::get('/store/search/',[StoreController::class,'getSearch']);

**Store Controller ထဲမွာ**

 public function getSearch(Request $request){

        $keyword=$request->input('keyword');

        $categories=Category::all();

        if($keyword !=""){

            return view('store.search')

                ->with('posts', Post::where('title', 'LIKE','%'.$keyword.'%')->get())

                ->with('keyword',$keyword)

                ->with('categories',$categories);

        }

    }

**Authentication**

1. Powershall ေခၚ **compose require laravel/ui** ကို run

2. Bootstrap ui သုံးမာမို႔လို႔ **Php artisan ui bootstrap** ကို run(ui vue /ui react ေတြလဲသုံးလို႔ရတယ္)

3. Php artisan ui:auth ကို run (auth folder ရယ္ တျခား folder ေတြ file ေတြ တိုးလာမယ္)

4. Login & Register theme မလွလို႔ ခ်ိန္းဖို့ auth/login ထဲမွာ extend က layouts.app ကိုို ျဖတ္ပီး layouts .main ကို extends လုပ္

5. Register လုပ္ပီးရင္ ကို႔ရဲ႕ Home page ကို ၫႊန္းဖို႔ homecontroller ထဲက index method ထဲက code ေတြ copy လုပ္ပီး StoreController ထဲက index method ထဲမွာ ထည့္လိုက္။

6. Login/register/logout လုပ္ဖို႔

layouts->main.php file ထဲမွာ login မလုပ္ရေသးတဲ့ လူေတြအတြက္ @guest ထဲမွာ login နဲ႕ register code menu ေတြ ေရး ပီး Login လုပ္ပီးသားတဲ့လူေတြတြက္ @auth ထဲမွာ logout နဲ႕ ကိုျပခ်တဲ့ menu ေတြ ေရးေပးရတယ္။

7. Login ၀င္လာတဲ့လူက admin ဆို Admin Dashboard menu ျပမယ္ မဟုတ္ရင္ မျပဘူး အလိုလုပ္ခ်င္ရင္

**(Php artisan make:middleware Adminlogin)** file ေဆာက္ေပး ပီးရင္ Kernel.php မွာ protected $routemiddleware ထဲမွာ ( **‘admin’ => \App\Http\Middleware\Adminlogin::class,** ) တစ္ေၾကာင္းေရးေပး ပီးရင္ Adminlogin file ထဲမွာ admin နဲ႕ ညီမညီ ဆိုတဲ့ code ေရးေပးရမယ္။

8. url bar မွာ admin ကို ေက်ာ္၀င္လို႔မရေအာင္

-UserController file ေဆာက္ web.php မွာ UserController ကို resource တခုလုပ္ ပီး Code ေတြေရး

-PostController နဲ႕ CategoryController ေတြထဲမွာ Construct method ကို ေရးေပးရတယ္ PostController နဲ႕ CategoryController ကို admin ကပဲ ထိန္းခ်ဳပ္ရတဲ့အတြက္ Middleware နဲ႕ ထိန္းခ်ဳပ္ဖို႔

 public function \_\_construct(){

        $this->middleware('auth'); auth folder ကို ထိန္းခ်ဳပ္ဖို႔

        $this->middleware('admin'); admin folder ကို ထိန္းခ်ဳပ္ဖို႔

        }

9. Admin ကေန logout လုပ္ရင္ <a> ထဲက data-target=model နဲ႕ logout model ကိုဖ်က္ေပးရတယ္။

**Pagination**

၁။ StoreController ရဲ႕ index()method မွာ get ေနရာမွာ paginate အစားထိုးလိုက္တာပဲ

public function index(){

        $categories=Category::all();

        return view('store.main')

        /\*->with('posts', Post::orderBy('created\_at','DESC')->get())   To Show All Post \*/

        ->with('posts', Post::orderBy('created\_at','DESC')->paginate(2))   /\* To Show With pagination \*/

        ->with('categories',$categories);  /\* To Show All Category \*/

    }

၂။

{!! $posts->links() !!} store/main.blade.php ထဲမွာ pagination လုပ္မယ့္ေနရာမွာ ေရးေပး $posts ဆိုတာ foreach ရဲ႕ အမ်ားကိုယူထားတာ

၃။ google /laravel8/database/pagination မွာ Tailwind Css Framwork ကို click/ get started/Via CDN Link copy လုပ္ပီး ကို႔ရဲ႕ css file ထဲမွာ လာထည့္။

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

**Test Email Verified**

Powershall မွာ

1.composer require laravel/ui

2. php artisan ui bootstrap - -auth (view folder နဲ႕ Controller မွာ file ေတြ ေရာက္လာလိမ့္မယ္)

3. npm install (google မွာ Node.js download ရွာပီးဆြဲထားမွ npm install လုပ္လို႔ရမာ

4. npm run dev

5. model / user.php မွာ Authenticatable ေဘးမွာ **implements MustVerifyEmail** လို႔ ျပင္ေရးေပးရမယ္။

6. HomeController မွာ $this->middleware**([‘auth’ , ’verified’]);** ဒါမ်ိဳး ျပင္ ေရးေပးရမယ္ **။**

7. Web.php ထဲက Auth::routes([ ‘verify’ => true]);

8. Mailtrap.io မွာ အေကာင့္ဖြင့္ Intergrations မွာ Laravel ေရြး / maile copy က်လာမယ္ copy ကို႔ရဲ႕ env file ထဲက SMTP အပိုဒ္ မွာ paste ခ် ရင္ Mail from Name နဲ႕ Address ၂ ေၾကာင္း ကို ခ်န္ထား Mail from Name မွာ ဘာမွမလုပ္နဲ႕ Address မွာ Mailtrap.io ဖြင့္ထားတဲ့ ကို႔ရဲ႕ email ထည့္။

User Change Password

Auth / change\_password.blade.php တစ္ခုေဆာက္ အဲ့ထဲမွာ Form Model ေတြေရး UserController ထဲမွာ edit() နဲ႕ update() method ေတြေရး