Livewire

Setup je Laravel macam biasa.. buat model dan buat migration file untuk create table yang terkait dengan model..

Then, baru install livewire

composer **require** livewire/livewire

then, kita boleh mula.. mula dengan wat livewire component

php artisan make:livewire posts

so kita dapati, dia create dua tempat..

app/Http/Livewire/Posts.php

resources/views/livewire/posts.blade.php

dekat file app/Http/Livewire/Posts.php **//file ni ibarat controller aaa untuk livewire**

sini kita akan tulis render(), resetInputFields(), store(), edit(), cancel(), update(), and delete().

Bahagian ni ada beberapa part..

First dekat luar function (global variable):

**public** $posts, $title, $body, $post\_id;

public $counter = 0;

public function render() //function utama untuk rendering page tu

{

$this->posts = Post::all(); //ni incase kit ana retrieve dari model

return view('livewire.posts');

}

\*tak perlu passing variable dah.. yang declare dekat global semua boleh direct access dkt livewire component.

**Route macam biasa – Cuma perlu letak tag panggil component livewire yang mana**

Next, default route kita dekat welcome kan? Welcome.blade.php.. so dalam case ni.. biar je, kita guna route default..

Replace code dalam Welcome.blade.php

<!DOCTYPE html>

<html>

<head>

    <title>Livewire</title>

    <link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" rel="stylesheet">

    @livewireStyles

</head>

<body>

    <div class="container">

        <!-- part ni panggil livewire view component -->

        @livewire('posts')

    </div>

    @livewireScripts

</body>

</html>

* Tengok ada panggil component posts………… @livewire(‘posts’)

So dekat livewire view pulak, yang digenerated tadi.. resources/views/livewire/posts.blade.php

<div>

    <h1 style="margin-top : 150px;">

        {{ $counter }}

    </h1>

    <button wire:click="tambahCounter" type="button" class="btn btn-dark">Add(+)</button>

</div>

\*kita boleh terus guna {{ $counter }} yang ada dekat global tadi..

Then, kita boleh guna tag wire:click sebagai listener yang untuk buat realtime ni… dan panggil function mana yang ada dalam posts.php tadi

**Form submit/v-bind tapi dalam livewire**

<input type="text" class="form-control" wire:model.defer="limit" aria-describedby="LimitHelp" placeholder="Enter Limit">

wire:model=’namafunction’ //ni nama function dalam livewire punya controller.

\*untuk setiap perubahan dekat input, dia akan update terus dekat livewire controller punya global variable.. ibarat onChange=”ubah variable” //tapi ubah variable to dekat server side..

wire:model.defer=’namafunction’ //ni pulak adalah untuk, tak sentiasa update.. just update global variable kat controller tu, once disubmit..

**Once dah declare variable dekat livewire controller tu.. as GLOBAL**

\*kita tak yah passing dah dekat view /livewire component.. boleh terus access je..

@foreach($history as $hist)

                ({{ $hist->bil }}) images has {{ $hist->semasa }}

        @endforeach

\*terus mcm biasa – mcm kt laravel biasa

**Keep on update/any changes on database update to render (realtime)**

<h1 style="margin-top : 150px;" wire:poll='pollingChanges'>

        {{ $counter }} <br>

    </h1>

\*hanya letak tag wire:poll=’namafunction’ jee…

Actually, letak kt memana pun tak apa, asal dkt function tu update necessary things.

Misalnya,

public function pollingChanges()

    {

        $this->counter = counter::select('count')->first()->count;

        $this->history = sejarah::latest()->limit(5)->get();

    }

\*So sini dia update 2 values… so as nature of livewire, setiap kali global variable dia berubah, dia render component yang perlu kan? So begitulah..

**Storage Path (Laravel)**

storage\_path('app/downloaded/' . $this->arr[$i] . '')

\*ni merujuk ke folder storage/app/downloaded