






STIW3044: Web Engineering
Semester A212
School of Computing, CAS, UUM

FRONT COVER

Final Exam Assignment

Name	Juanrico Alvaro
Matric No	702301
YouTube Presentation Link	https://youtu.be/_QpVRH1tlg0
Phone Number	(+60)136798361 / (+62)81283185508
GitHub Link	https://github.com/victorico123/MyTutor-Laravel
Submission Date	4/7/2022
Acknowledgment	I hereby acknowledge that the following works are from my effort in submitting this document. If found otherwise, severe action such as marks deduction or removal from the assignment can be taken against me.
Digital Signature	
Students Picture	

	STIW3044: Web Engineering Semester A212 School of Computing, CAS, UUM
Final Exam Assignment	
Given date: 3/7/2022	
Submission date: 7/7/2022	
100 Marks	

A212 WEB ENGINEERING FINAL (100 MARKS)**Instructions**

You have been assigned to develop submodules for a web application "Tutor Raya." The app allows users to find tutors and register for related subjects. Complete the following instructions by implementing the solution using the Laravel web application development framework. Your assignment is as follows.

1. Landing Page Module**(15 Marks)**

- a. Create a new Laravel project titled MyTutor and start with a new landing page that describes the MyTutor application. Create two links that allow for a new tutor to register and log in. Show your user HTML code. You can use lorem ipsum text generator for content if you wish to.

10 Marks[Welcome.blade.php](#)

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <title>Laravel</title>

    <!-- Fonts -->
```

```

    <link
href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;600;700&display
=swap" rel="stylesheet">

    <!-- Styles -->
    <style>
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
    <link href="{{ asset('css/landing.css') }}" rel="stylesheet">
    <link href="{{ asset('css/style.css') }}" rel="stylesheet">

    <!-- Scripts -->
    <script src="{{ asset('js/app.js') }}" defer></script>
</head>
<body class="antialiased">

    <div class='hero-header align-items-center align-content-center justify-
content-center'>

        <div class="parent-landing-card " style="height: 100vh;">
            <div class="card landing-card tabcontent">
                <div>
                    <object id="svg-teaching"
data="../../../assets/application/svg/teaching.svg" type="image/svg+xml"></object>
                </div>
                <div class="card-body text-dark">
                    <h3 class="card-title"><span
style="color:blue">My</span><span class="tutor-text">Tutor</span>
                    </h3>
                    <p class="card-text">"Application for a tutor booking
and payment."</p>

                    <hr>
                    @auth
                    <a href="{{ url('/home') }}">
                        <button class="btn btn-primary">Home</button>
                    </a>
                    @else
                    <a href="{{ route('login') }}">
                        <button class="btn btn-primary">Sign In</button>
                    </a>
                    <a href="{{ route('register') }}">
                        <button class="btn btn-primary">Sign Up</button>
                    </a>
                    @endauth
                </div>
            </div>
        </div>
    </div>
</body>

```

```
</html>
```

b. Show your route entry for 1.a.

2 Marks

```
<?php

use Illuminate\Support\Facades\Auth;

Route::get('/', [App\Http\Controllers\LandingController::class, 'landing'])->name('landing');
```

```
<?php

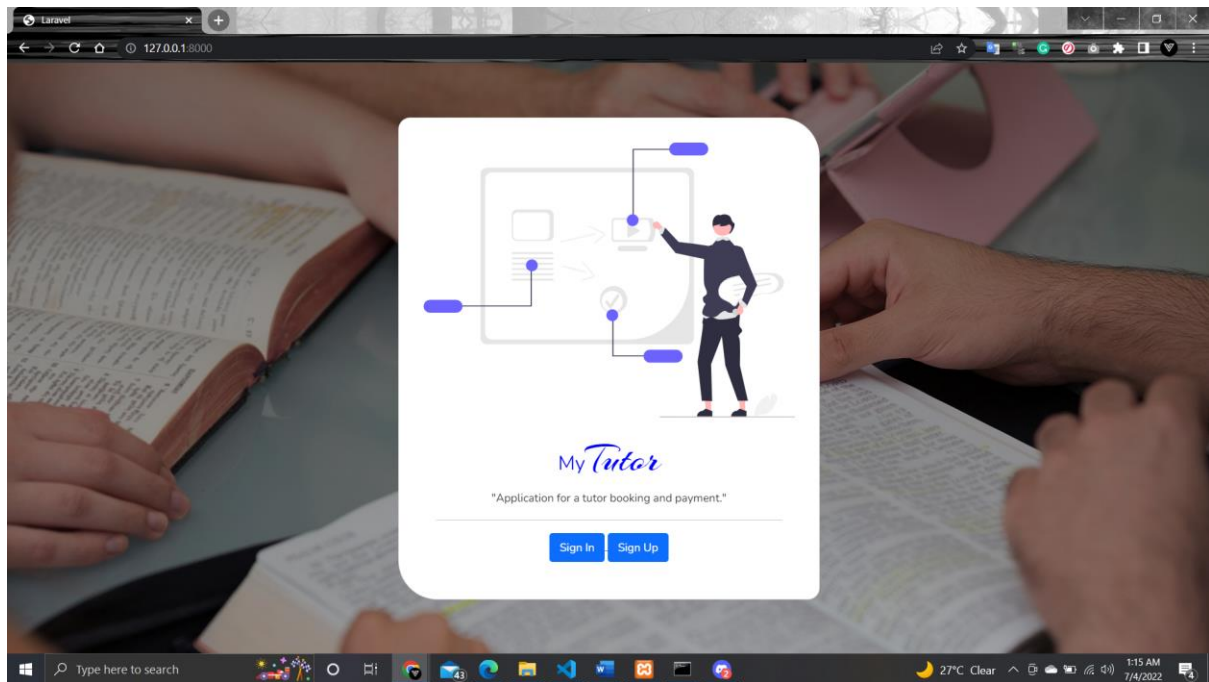
namespace App\Http\Controllers;

use Illuminate\Http\Request;

class LandingController extends Controller
{
    public function landing()
    {
        return view('welcome');
    }
}
```

c. Show your landing page screenshot.

3 Marks



2. The New tutor registration module allows any new tutor to register with the “MyTutor” web application. The following information will be required for a tutor registration page: full name, email, phone number, mailing address, state (Malaysia), and password.

(30 Marks)

I use Laravel AUTH Installer so user can automatically go to the login blade or register blade by just including the Auth class and put it on the web.php. to do this, we have to run some command on our command prompt as follow:

- **composer require laravel/ui**
- **php artisan ui bootstrap --auth**
- **npm install**
- **npm run dev**

There will be blade file generated so we just need to modify the blade. All of the route handling have already been solved to us so we don't need to worry and the security is much more secure then we are making it ourself. By installing these, we can utilize the routing they give by using `Auth::Routes()` on the web.php file.

- a. Create a new form for the new tutor registration based on the information. Show your **HTML form** script from your blade file.

10 Marks

```
@extends('layouts.register_login_template')
```

```

@section('content')

<div class="container">
  <h5 style="color:blue">SIGN UP</h5>
  <p class="left">
    <form method="POST" action="{{ route('register') }}">
      @csrf
      <div class="row">
        <div class="col-sm">
          <div class="form-group">
            <label for="name" class="text-left">Full Name <span
style="color: red;">*</span></label>
            <div>
              <input id="name" type="text" class="form-control
@error('name') is-invalid @enderror" name="name" value="{{ old('name') }}"
autofocus>

              @error('name')
              <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
              </span>
              @enderror
            </div>
          </div>
          <br>
          <div class="form-group">
            <label for="emailInput" class="text-left">Email Address
<span style="color: red;">*</span></label>
            <div>
              <input id="emailInput" type="text" class="form-control
@error('email') is-invalid @enderror" name="email" value="{{ old('email') }}"
autofocus>

              @error('email')
              <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
              </span>
              @enderror
            </div>
          </div>
          <br>
          <div class="form-group">
            <label for="passwordInput">Password <span style="color:
red;">*</span></label>
            <div>
              <input id="passwordInput" type="password" class="form-
control @error('password') is-invalid @enderror" name="password"
autocomplete="current-password">

```

```

        @error('password')
        <span class="invalid-feedback" role="alert">
            <strong>{{ $message }}</strong>
        </span>
    @enderror
</div>
</div>
<br>
</div>
<div class="col-sm">
    <div class="form-group">
        <label for="phoneInput" class="text-left">Phone Number
<span style="color: red; ">*</span></label>
        <div>
            <input id="phoneInput" type="text" class="form-control
@error('phone') is-invalid @enderror" name="phone" value="{{ old('phone') }}"
autofocus>

            @error('phone')
            <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
            </span>
            @enderror
        </div>
    </div>
    <br>
    <div class="form-group">
        <label for="addressInput" class="text-left">Mailing
Address <span style="color: red; ">*</span></label>
        <div>
            <textarea id="addressInput" type="text" class="form-
control @error('address') is-invalid @enderror" name="address" autofocus>{{
old('address') }}</textarea>
            @error('address')
            <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
            </span>
            @enderror
        </div>
    </div>
    <br>
    <div class="form-group">
        <label for="state" class="text-left">State <span
style="color: red; ">*</span></label></label>
        <div class="col-sm-10">
            <select class="form-control form-select" id="state"
name="state">

```

```

        <option value="0" {{ (old("state") == "") ?
"selected":"" }}>N/A</option>
        <option value="Johor" {{ (old("state") == "Johor")
? "selected":"" }}>Johor</option>
        <option value="Kedah" {{ (old("state") == "Kedah")
? "selected":"" }}>Kedah</option>
        <option value="Kelantan" {{ (old("state") ==
"Kelantan") ? "selected":"" }}>Kelantan</option>
        <option value="Kuala Lumpur" {{ (old("state") ==
"Kuala Lumpur") ? "selected":"" }}>Kuala Lumpur</option>
        <option value="Labuan" {{ (old("state") ==
"Labuan") ? "selected":"" }}>Labuan</option>
        <option value="Malacca" {{ (old("state") ==
"Malacca") ? "selected":"" }}>Malacca</option>
        <option value="Negeri Sembilan" {{ (old("state")
== "Negeri Sembilan") ? "selected":"" }}>Negeri Sembilan</option>
        <option value="Pahang" {{ (old("state") ==
"Pahang") ? "selected":"" }}>Pahang</option>
        <option value="Perak" {{ (old("state") == "Perak")
? "selected":"" }}>Perak</option>
        <option value="Perlis" {{ (old("state") ==
"Perlis") ? "selected":"" }}>Perlis</option>
        <option value="Penang" {{ (old("state") ==
"Penang") ? "selected":"" }}>Penang</option>
        <option value="Sabah" {{ (old("state") == "Sabah")
? "selected":"" }}>Sabah</option>
        <option value="Sarawak" {{ (old("state") ==
"Sarawak") ? "selected":"" }}>Sarawak</option>
        <option value="Selangor" {{ (old("state") ==
"Selangor") ? "selected":"" }}>Selangor</option>
        <option value="Terengganu" {{ (old("state") ==
"Terengganu") ? "selected":"" }}>Terengganu</option>
    </select>
</div>
    @error('state')
    <span class="invalid-feedback d-block" role="alert">
        <strong>{{ $message }}</strong>
    </span>
    @enderror
</div>
<br>
</div>
</div>

    <button type="submit" class="btn btn-primary">Submit</button>
</form>
</p>
@endsection

```


- b. Create a route entry for the page accessible from the landing page. Show the script required to enable the routing for the registration page.

2 Marks

Here, I am using the Auth model after installing the auth before.

```
Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');
```

In here the installed home controller, will give us a middleware that checking whether it is a guest or a logged in user before going to index() function. If it is a guest it will automatically redirect to login page

```
class HomeController extends Controller
{
    public function __construct()
    {
        $this->middleware('auth');
    }

    public function index()
    {
        return view('home');
    }
}
```

Here is the auth class on middleware folder

```
class Authenticate extends Middleware
{
    protected function redirectTo($request)
    {
        if (! $request->expectsJson()) {
            return route('login');
        }
    }
}
```

On my web.php there is

```
Auth::routes();
```

This is a helper class that helps generate all the routes required for user authentication. It can be said equivalent to

```
// Authentication Routes...
$this->get('login', 'Auth\LoginController@showLoginForm')->name('login');
$this->post('login', 'Auth\LoginController@login');
$this->post('logout', 'Auth\LoginController@logout')->name('logout');

// Registration Routes...
$this->get('register', 'Auth\RegisterController@showRegistrationForm')->name('register');
$this->post('register', 'Auth\RegisterController@register');

// Password Reset Routes...
$this->get('password/reset', 'Auth\ForgotPasswordController@showLinkRequestForm');
$this->post('password/email', 'Auth\ForgotPasswordController@sendResetLinkEmail');
$this->get('password/reset/{token}', 'Auth\ResetPasswordController@showResetForm');
$this->post('password/reset', 'Auth\ResetPasswordController@reset');
```

This is why we can use button that call the route to go to the blade:

```
<a href="{{ route('register') }}">
```

- c. Create the “tutor” data model and show the command to create the model.

3 Marks

For this, I will use user model as the tutor to be able to have a large access to the AUTH model.

```
<?php

namespace App\Models;

use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable;
    protected $fillable = [
        'name',
        'email',
        'phone',
    ];
}
```

```

        'address',
        'state',
        'password',
    ];
    protected $hidden = [
        'password',
        'remember_token',
    ];
    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
}

```

COMMAND LINE IF WE ARE GOING TO MAKE A NEW TUTOR MODEL

php artisan make:model Tutor -m -s -c

-m : to create a migration file

-s : to create a seeder file

-c : to create a controller file

- d. Based on the information given, complete the data model definition from the function “up”. Show the function.

5 Marks

```

public function up()
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('name');
        $table->string('email')->unique();
        $table->string('phone');
        $table->string('address');
        $table->string('state');
        $table->timestamp('email_verified_at')->nullable();
        $table->string('password');
        $table->rememberToken();
        $table->timestamps();
    });
}

```

- e. Complete the tutor controller registration logic to register new tutors into the table. Show the controller script required to perform the tutor registration.

10 Marks

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use App\Models\User;
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;

class RegisterController extends Controller
{
    use RegistersUsers;

    protected $redirectTo = RouteServiceProvider::HOME;

    public function __construct()
    {
        $this->middleware('guest');
    }

    protected function validator(array $data)
    {
        return Validator::make($data, [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255',
'unique:users'],
            'password' => ['required', 'string', 'min:8'],
            'state' => ['required', 'not_in:0'],
            'address' => ['required', 'string'],
            'phone' => ['required', 'integer'],
        ]);
    }

    protected function create(array $data)
    {
        return User::create([
            'name' => $data['name'],
            'email' => $data['email'],
            'password' => Hash::make($data['password']),
            'address' => $data['address'],
            'phone' => $data['phone'],
            'state' => $data['state'],
        ]);
    }
}
```

vendor\laravel\ui\auth-backend\RegistersUsers.php

```
<?php

namespace Illuminate\Foundation\Auth;

use Illuminate\Auth\Events\Registered;
use Illuminate\Http\JsonResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

trait RegistersUsers
{
    use RedirectsUsers;

    /**
     * Show the application registration form.
     *
     * @return \Illuminate\View\View
     */
    public function showRegistrationForm()
    {
        return view('auth.register');
    }

    /**
     * Handle a registration request for the application.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\RedirectResponse|\Illuminate\Http\JsonResponse
     */
    public function register(Request $request)
    {
        $this->validator($request->all())->validate();

        event(new Registered($user = $this->create($request->all())));

        $this->guard()->login($user);

        if ($response = $this->registered($request, $user)) {
            return $response;
        }

        return $request->wantsJson()
            ? new JsonResponse([], 201)
            : redirect($this->redirectPath());
    }

    /**
```

```

    * Get the guard to be used during registration.
    *
    * @return \Illuminate\Contracts\Auth\StatefulGuard
    */
protected function guard()
{
    return Auth::guard();
}

/**
 * The user has been registered.
 *
 * @param \Illuminate\Http\Request $request
 * @param mixed $user
 * @return mixed
 */
protected function registered(Request $request, $user)
{
    //
}
}

```

3. Once the tutor registration module is completed, the tutor will be allowed to log in. The following submodule will require the implementation of the tutor's login.

(20 Marks)

- a. Create a new form for the tutor login page. Show your **HTML form** script from your blade file.

5 Marks

```

<h5 style="color:blue">SIGN IN</h5>
<p class="left">
  <form method="POST" action="{{ route('login') }}">
    @csrf
    <div class="form-group">
      <label for="EmailInput" class="text-left">Email address <span
style="color: red;">*</span></label>
      <div>
        <input id="email" type="text" class="form-control
@error('email') is-invalid @enderror" name="email" value="{{ old('email') }}"
autofocus>

        @error('email')

```

```

        <span class="invalid-feedback" role="alert">
            <strong>{{ $message }}</strong>
        </span>
    @enderror
</div>
</div>
<br>
<div class="form-group">
    <label for="PasswordInput">Password <span style="color:
red;">*</span></label>
    <div>
        <input id="password" type="password" class="form-control
@error('password') is-invalid @enderror" name="password"
autocomplete="current-password">

        @error('password')
        <span class="invalid-feedback" role="alert">
            <strong>{{ $message }}</strong>
        </span>
    @enderror
    </div>
</div>
<br>
<div class="row mb-3">
    <div class="col-md-6">
        <div class="form-check">
            <input class="form-check-input" type="checkbox"
name="remember" id="remember" {{ old('remember') ? 'checked' : '' }}>

            <label class="form-check-label" for="remember">
                {{ __( 'Remember Me' ) }}
            </label>
        </div>
    </div>
</div>
    <button type="submit" class="btn btn-primary">Submit</button>
</form>

</p>

```

- b. Create a route entry accessible from the landing page. Show the script required to enable the routing for the login page.

2 Marks

On my web.php there is

```
Auth::routes();
```

This is a helper class that helps generate all the routes required for user authentication. It can be said equivalent to

```
// Authentication Routes...
$this->get('login', 'Auth\LoginController@showLoginForm')->name('login');
$this->post('login', 'Auth\LoginController@login');
$this->post('logout', 'Auth\LoginController@logout')->name('logout');

// Registration Routes...
$this->get('register', 'Auth\RegisterController@showRegistrationForm')->name('register');
$this->post('register', 'Auth\RegisterController@register');

// Password Reset Routes...
$this->get('password/reset', 'Auth\ForgotPasswordController@showLinkRequestForm');
$this->post('password/email', 'Auth\ForgotPasswordController@sendResetLinkEmail');
$this->get('password/reset/{token}', 'Auth\ResetPasswordController@showResetForm');
$this->post('password/reset', 'Auth\ResetPasswordController@reset');
```

This is why we can use button that call the route to go to the blade:

```
<a href="{{ route('login') }}">
```

- c. Create a simple main page UI with its **route** entry. Show the blade file and its route.

5 Marks

```
Route::get('/home', [App\Http\Controllers\HomeController::class, 'index'])->name('home');
```

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller
```



```
{

    public function __construct()
    {
        $this->middleware('auth');
    }

    public function index()
    {
        return view('home');
    }
}
```

Template blade

```
<!doctype html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- CSRF Token -->
    <meta name="csrf-token" content="{{ csrf_token() }}">

    <title>MyTutor</title>

    <!-- Scripts -->
    <script src="{{ asset('js/app.js') }}" defer></script>

    <!-- Fonts -->
    <link rel="dns-prefetch" href="//fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css?family=Nunito"
rel="stylesheet">

    <!-- Styles -->
    <link href="{{ asset('css/app.css') }}" rel="stylesheet">
        <link href="{{ asset('css/landing.css') }}" rel="stylesheet">
        <link href="{{ asset('css/style.css') }}" rel="stylesheet">

</head>
<body class="bg-secondary">
    <div id="app ">
        <nav class="navbar navbar-expand-md navbar-dark bg-dark shadow-sm">
            <div class="container">
```

```

        <a class="navbar-brand" href="{{ url('/') }}" style="font-size: 17px;">
        <div class="container"><span
style="color:orange">My</span><span class="tutor-text" style="font-size: 30px;
color:orange">Tutor</span></div>
        </a>
        <button class="navbar-toggler" type="button" data-bs-
toggle="collapse" data-bs-target="#navbarSupportedContent" aria-
controls="navbarSupportedContent" aria-expanded="false" aria-label="{{
__('Toggle navigation') }}">
        <span class="navbar-toggler-icon"></span>
        </button>

        <div class="collapse navbar-collapse"
id="navbarSupportedContent">
        <ul class="navbar-nav me-auto">
        </ul>
        <ul class="navbar-nav ms-auto">
            @guest
                @if (Route::has('login'))
                    <li class="nav-item">
                        <a class="nav-link" href="{{
route('login') }}">{{ __('Login') }}</a>
                    </li>
                @endif

                @if (Route::has('register'))
                    <li class="nav-item">
                        <a class="nav-link" href="{{
route('register') }}">{{ __('Register') }}</a>
                    </li>
                @endif
            @else

                <li class="nav-item center align-items-center
justify-content-center" style="margin-right:20px">
                    <div class="nav-link">Hi, {{ Auth::user()->name
}}</div>
                </li>
                <li class="nav-item center align-items-center
justify-content-center">
                    <form id="logout-form" action="{{ route('logout')
}}" method="POST" class="d-block">
                        @csrf
                        <button class="btn btn-danger"
type="submit">Logout</button>
                    </form>
                </li>

```

```
                @endguest
            </ul>
        </div>
    </div>
</nav>

    <main class="py-4">
        @yield('content')
    </main>
</div>
</body>
</html>
```

- d. Implement the tutor controller's registration logic to log in users using the authentication method provided by Laravel. Show the controller scripts required to perform the login operation. Once authenticated, the user will be forwarded to the main page.

8 Marks

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Foundation\Auth\AuthenticatesUsers;

class LoginController extends Controller
{
    use AuthenticatesUsers;

    protected $redirectTo = RouteServiceProvider::HOME;

    public function __construct()
    {
        $this->middleware('guest')->except('logout');
    }
}
```

[vendor\laravel\ui\auth-backend\AuthenticatesUsers.php](#)

```
<?php

namespace Illuminate\Foundation\Auth;

use Illuminate\Http\JsonResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Validation\ValidationException;

trait AuthenticatesUsers
{
    use RedirectsUsers, ThrottlesLogins;

    public function showLoginForm()
    {
        return view('auth.login');
    }

    public function login(Request $request)
    {
        $this->validateLogin($request);
        if (method_exists($this, 'hasTooManyLoginAttempts') &&
            $this->hasTooManyLoginAttempts($request)) {
            $this->fireLockoutEvent($request);

            return $this->sendLockoutResponse($request);
        }

        if ($this->attemptLogin($request)) {
            if ($request->hasSession()) {
                $request->session()->put('auth.password_confirmed_at',
time());
            }

            return $this->sendLoginResponse($request);
        }

        $this->incrementLoginAttempts($request);

        return $this->sendFailedLoginResponse($request);
    }

    protected function validateLogin(Request $request)
    {
        $request->validate([
            $this->username() => 'required|string',
            'password' => 'required|string',
        ]);
    }
}
```

```
protected function attemptLogin(Request $request)
{
    return $this->guard()->attempt(
        $this->credentials($request), $request->boolean('remember')
    );
}

protected function credentials(Request $request)
{
    return $request->only($this->username(), 'password');
}

protected function sendLoginResponse(Request $request)
{
    $request->session()->regenerate();

    $this->clearLoginAttempts($request);

    if ($response = $this->authenticated($request, $this->guard()-
>user())) {
        return $response;
    }

    return $request->wantsJson()
        ? new JsonResponse([], 204)
        : redirect()->intended($this->redirectPath());
}

protected function authenticated(Request $request, $user)
{
    //
}

protected function sendFailedLoginResponse(Request $request)
{
    throw ValidationException::withMessages([
        $this->username() => [trans('auth.failed')],
    ]);
}

public function username()
{
    return 'email';
}

public function logout(Request $request)
{
    $this->guard()->logout();
}
```

```

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        if ($response = $this->loggedOut($request)) {
            return $response;
        }

        return $request->wantsJson()
            ? new JsonResponse([], 204)
            : redirect('/');
    }

    protected function loggedOut(Request $request)
    {
        //
    }

    /**
     * Get the guard to be used during authentication.
     *
     * @return \Illuminate\Contracts\Auth\StatefulGuard
     */
    protected function guard()
    {
        return Auth::guard();
    }
}

```

vendor\laravel\ui\auth-backend\RedirectsUsers.php

```

<?php

namespace Illuminate\Foundation\Auth;

trait RedirectsUsers
{
    public function redirectPath()
    {
        if (method_exists($this, 'redirectTo')) {
            return $this->redirectTo();
        }

        return property_exists($this, 'redirectTo') ? $this->redirectTo :
        '/home';
    }
}

```

```
}
```

4. The following submodule will require the application to display the main page once the user successfully login. The main page is responsible for listing the tutor's current registered subjects. However, since there is no data yet available for subject listing, create a subject registration form using the following information: subject id, title, description, price, and total learning hours.

(25 Marks)

- a. Create a new link from the main page for the new subject registration. Show the link and route script required.

2 Marks

Home blade

```
<a href="{{url('/add-subject')}}" class="btn btn-warning ">Add
Subject</a>
```

Web.php route

```
Route::get('/add-subject', [App\Http\Controllers\SubjectController::class,
'showAddSubject']);
```

Subject controller

```
public function showAddSubject()
{
    return view('add_subject');
}
```

- a. Create a new form for the new subject registration. Show your **HTML form** script from the HTML blade file.

5 Marks

```
@extends('layouts.app')

@section('content')

<div class="parent-landing-card " style="height: 80vh;">
```

```

<div class="card landing-card" style="width: 70%;">
  <h3 class="card-title "><span class="tutor-text text-center">Add New
Subject</span></h3>
  <div class="card-body text-dark">
    <div class="container" style="width: 100%;">
      <p class="left">
        <form method="POST" action="{{ url('/add-subject') }}">
          @csrf
          <div class="row">
            <div class="col-md">
              <div class="form-group">
                <label for="titleInput" class="text-
left">Subject Title<span style="color: red; ">*</span></label>
                <div>
                  <input id="titleInput" type="text"
class="form-control @error('title') is-invalid @enderror" name="title"
value="{{ old('title') }}" autofocus>

                  @error('title')
                  <span class="invalid-feedback"
role="alert">

                    <strong>{{ $message }}</strong>
                  </span>
                  @enderror
                </div>
              </div>
            </div>
            <div class="col-md">
              <div class="row">
                <div class="col-md">
                  <div class="form-group">
                    <label for="priceInput" class="text-
left">Subject price<span style="color: red; ">*</span></label>
                    <div>
                      <input id="priceInput" type="text"
class="form-control @error('price') is-invalid @enderror" name="price"
value="{{ old('price') }}" autofocus>

                      @error('price')
                      <span class="invalid-feedback"
role="alert">

                        <strong>{{ $message
}}</strong>

                      </span>
                      @enderror
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </form>
      </p>
    </div>
  </div>
</div>

```



```

        <div class="col-md">
            <div class="form-group">
                <label for="hourInput" class="text-
left">Learning hour<span style="color: red;">*</span></label>
                <div>
                    <input id="hourInput"
type="number" min="0" step="1" class="form-control @error('hour') is-invalid
@enderror" name="hour" value="{{ old('hour') }}" autofocus>

                    @error('hour')
                    <span class="invalid-feedback"
role="alert">

                        <strong>{{ $message
}}</strong>

                    </span>
                    @enderror
                </div>
            </div>
        </div>
    </div>
    <br>
    <div class="form-group">
        <label for="descriptionInput" class="text-
left">Subject Description <span style="color: red;">*</span></label>
        <div>
            <textarea id="descriptionInput" type="text"
class="form-control @error('description') is-invalid @enderror"
name="description" autofocus>{{ old('description') }}</textarea>
            @error('description')
            <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
            </span>
            @enderror
        </div>
    </div>
    <br>
    <button type="submit" class="btn btn-
primary">Submit</button>
    </form>
</p>
</div>
</div>
</div>
</div>
@endsection

```

- b. Create the route entry for the new subject registration. Show the route entry.

2 Marks

Home blade

```
<a href="{{url('/add-subject')}}" class="btn btn-warning ">Add  
Subject</a>
```

Web.php route

```
Route::get('/add-subject', [App\Http\Controllers\SubjectController::class,  
'showAddSubject']);
```

Subject controller

```
public function showAddSubject()  
{  
    return view('add_subject');  
}
```

- c. Create the subject data model and show the instruction required to create the model.

3 Marks

```
class Subject extends Model  
{  
    use HasFactory;  
  
    protected $fillable = [  
        'title',  
        'description',  
        'price',  
        'learning_hour',  
    ];  
}
```

COMMAND LINE IF WE ARE GOING TO MAKE A NEW SUBJECT MODEL

php artisan make:model Subject -m -s -c

-m : to create a migration file

-s : to create a seeder file

- c : to create a controller file

- d. Based on the information given, complete the data model definition from the function “up”. Show the function.

5 Marks

```
public function up()
{
    Schema::create('subjects', function (Blueprint $table) {
        $table->id();
        $table->string('title');
        $table->string('description');
        $table->double('price');
        $table->integer('learning_hour');
        $table->timestamps();
    });
}
```

- e. Complete the new subject controller registration logic to register a new subject into the database. Show the controller script required to perform the registration.

8 Marks

```
public function addSubject(Request $request)
{
    $rules = Validator::make($request->all(), [
        'title' => ['required', 'min:5', 'max:50'],
        'description' => ['required', 'min:10'],
        'price' => ['required', 'numeric', 'min:0'],
        'hour' => ['required'],
    ]);

    $rules->validate();

    $obj = new Subject;

    $obj->title = $request->title;
    $obj->description = $request->description;
    $obj->price = $request->price;
    $obj->learning_hour = $request->hour;

    $obj->save();
}
```

```
return redirect('home')->with('success','Item created successfully!');  
}
```

5. Create a video presentation that demonstrates each of the following and upload it to YouTube. Make sure not to set the link as private.

10 Marks

YouTube Link: → https://youtu.be/_QpVRH1tlg0

- a. Landing page
- b. Registration page
- c. Main page
- d. New subject registration