Custom Auth In Laravel 6

1. Users have to register with giving name, email, phone number and password.

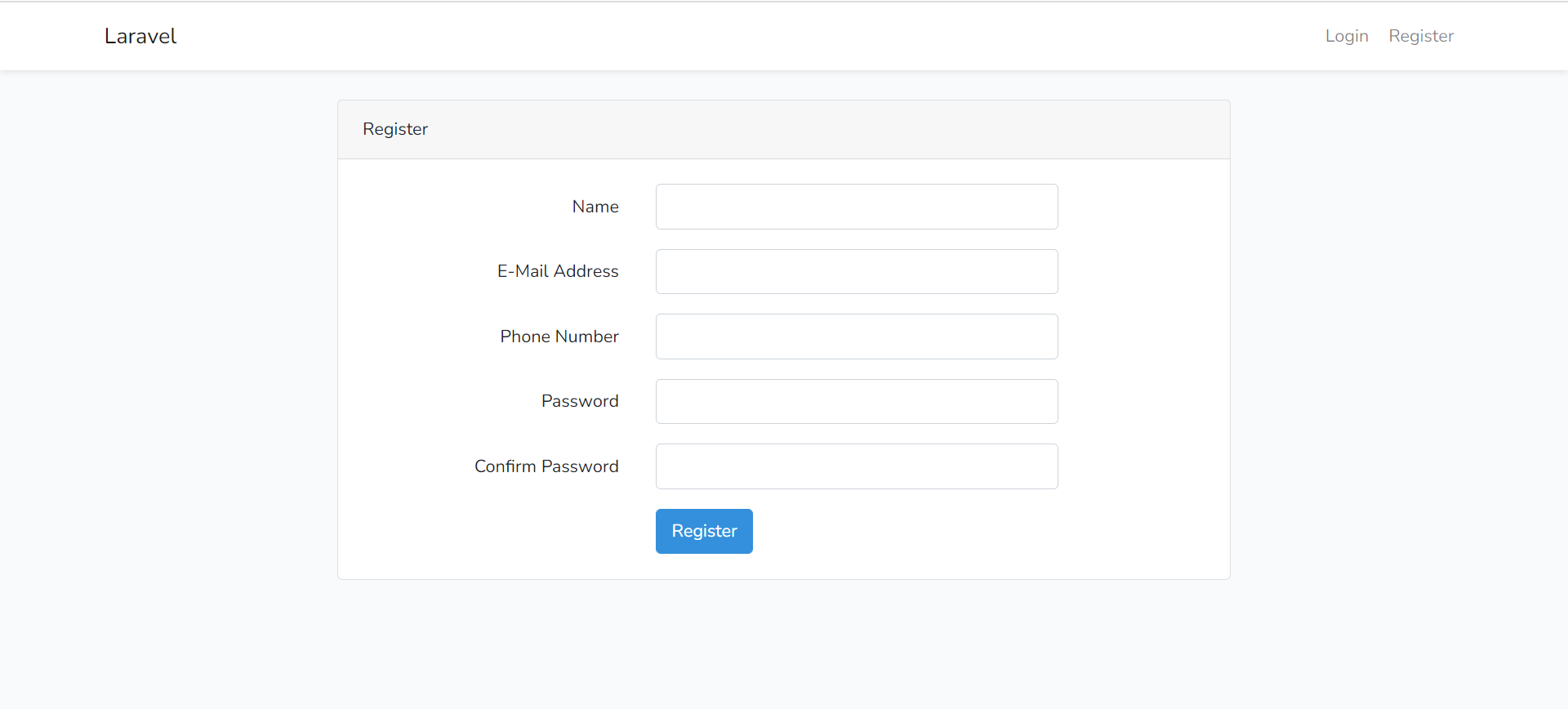


Fig 0.1: Registration form for user

Email id must be email type, phone number must be given with country code and password must be 8 in length and unique.

1. After registration, otp will be generated and otp will be sent to users mobile phone number.

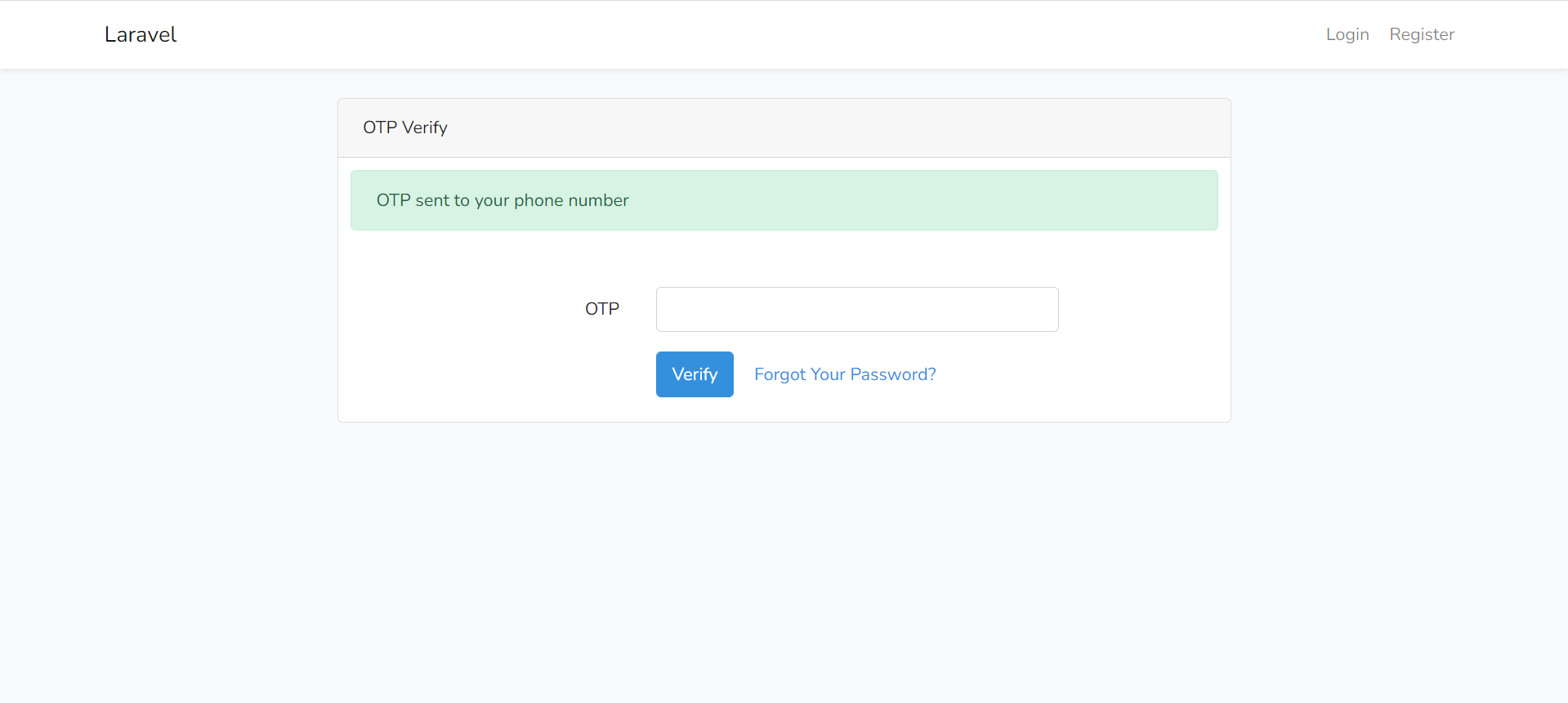


Fig 0.2: Otp sent to users phone number

1. Otp must be verified, otherwise user cannot be entered into dashboard via login.

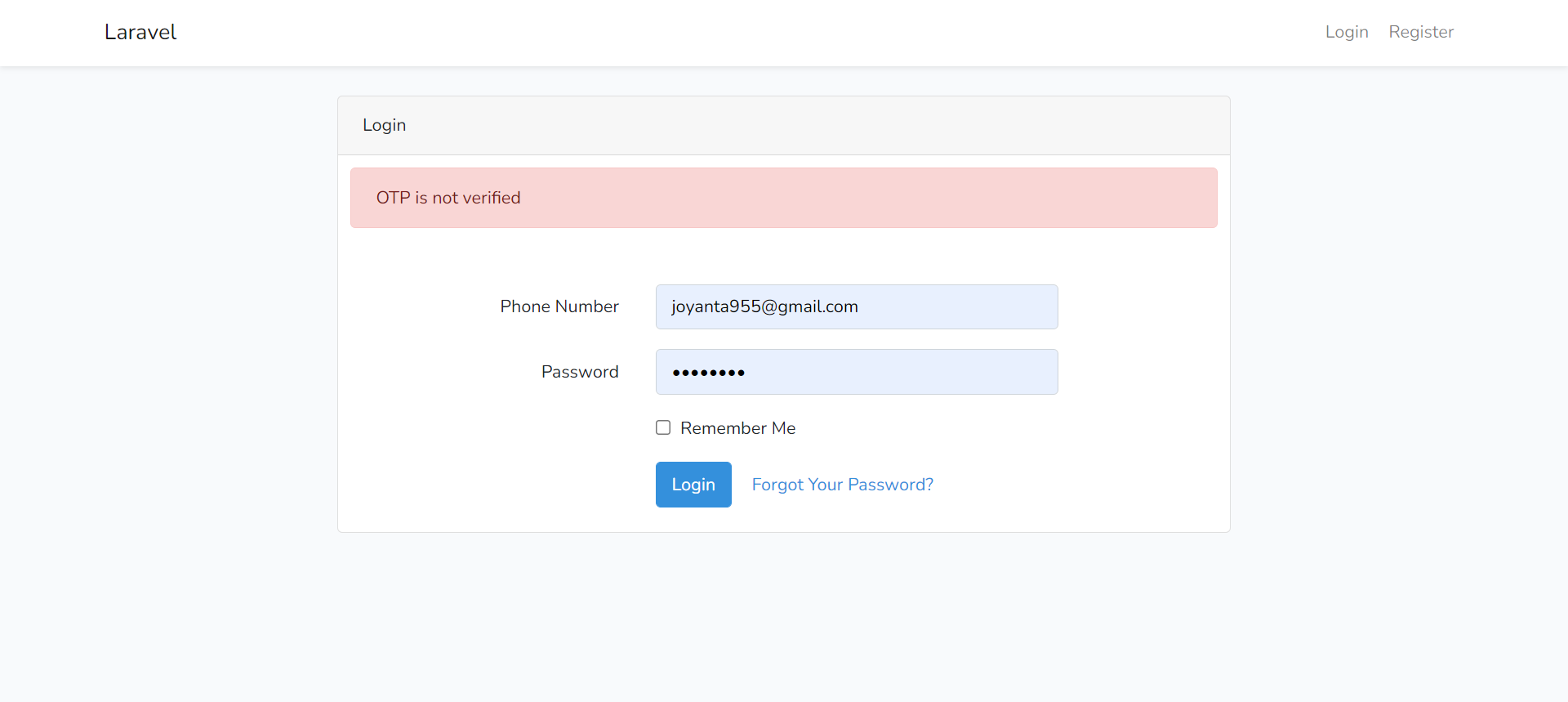


Fig 0.3: Otp must be verified.

1. Otp must be valid to verify.

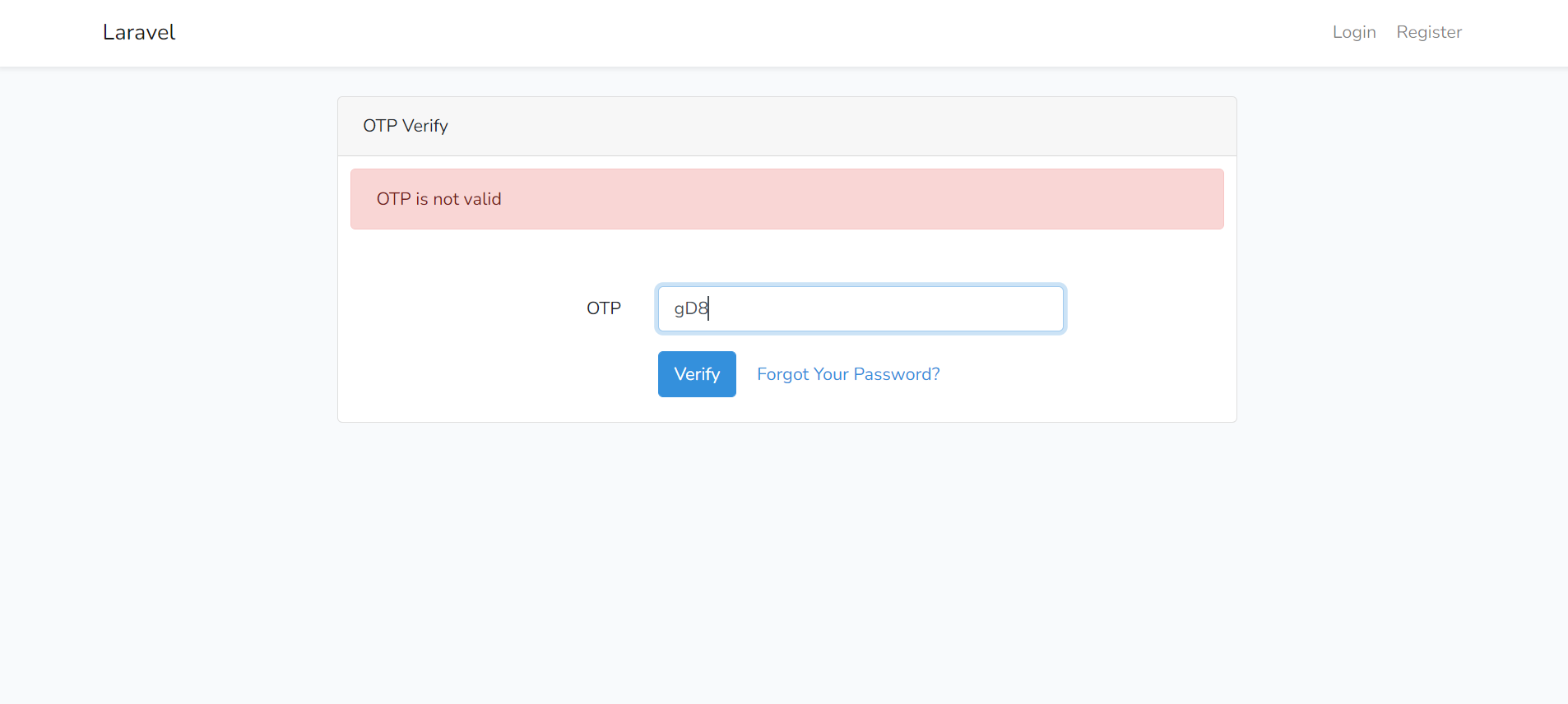


Fig 0.4: Otp must be valid

1. After otp verification, user will be gotten into dashboard.

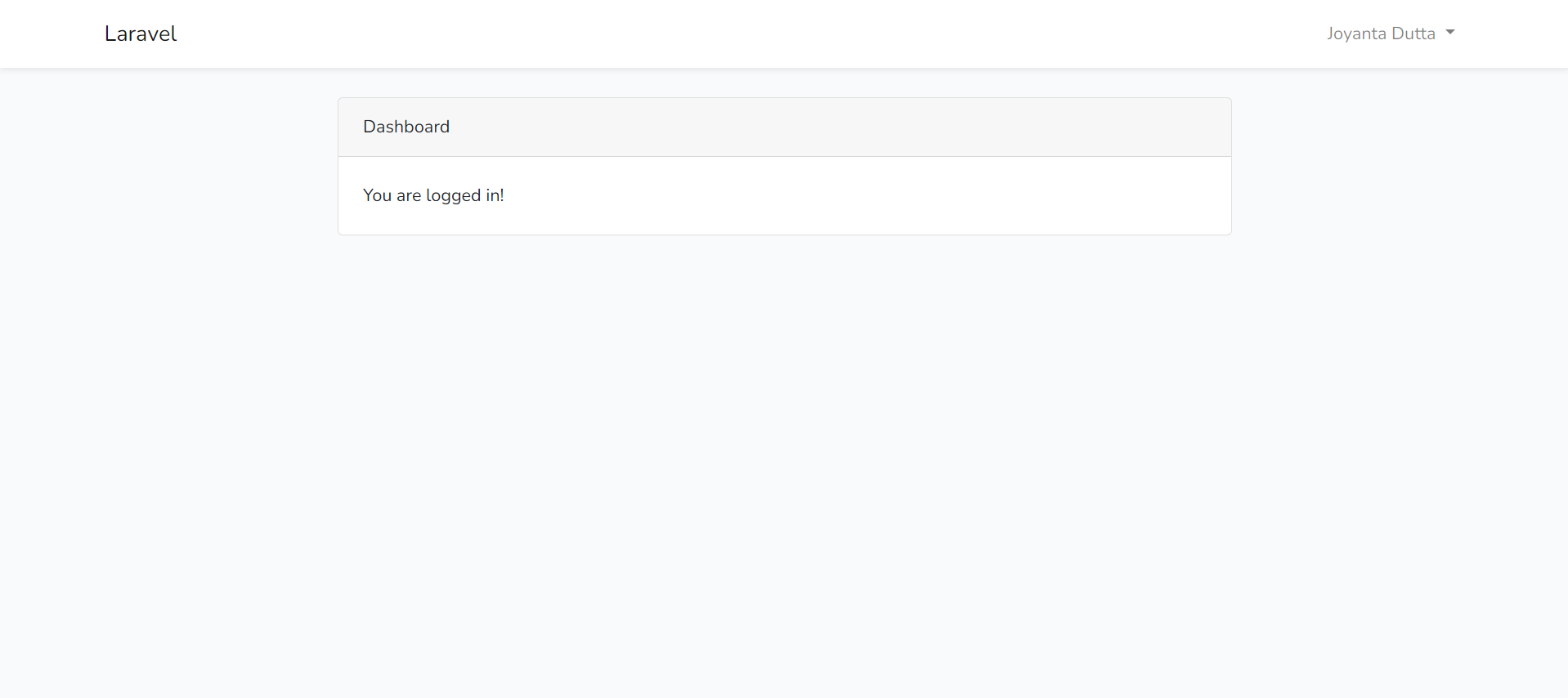


Fig 0.5: After otp verification, user dashboard

1. Phone number must be valid to log in.

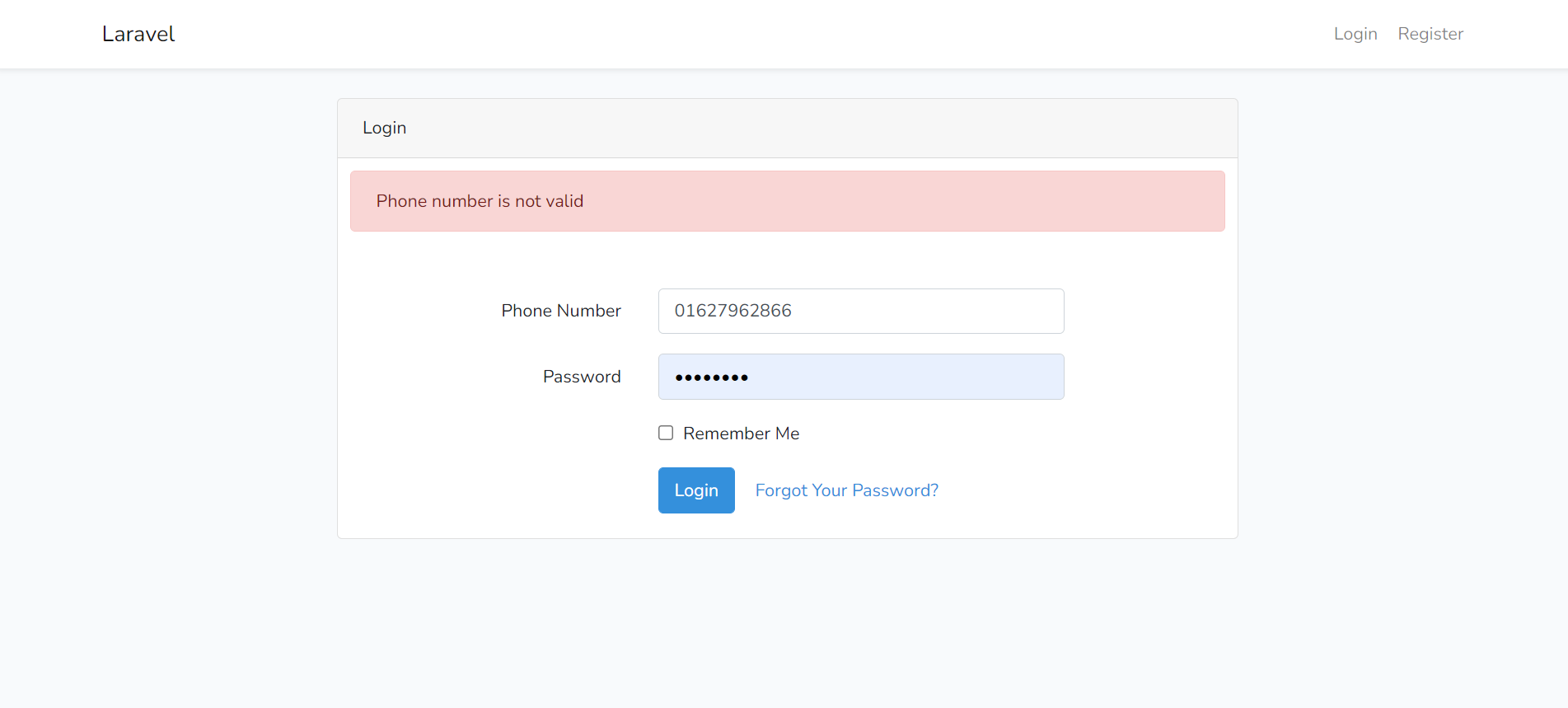


Fig 0.6: Phone number must be valid

1. Credentials must be matched with record.

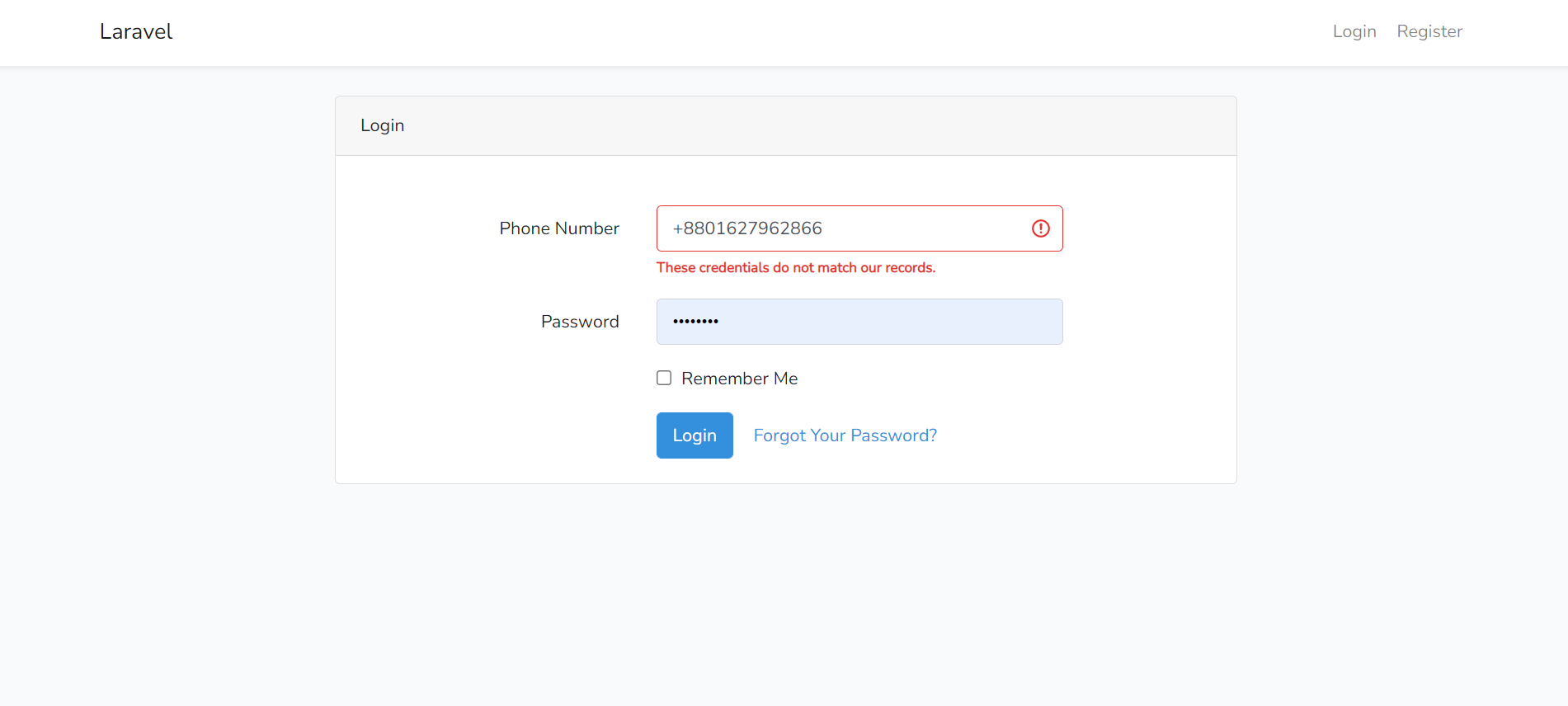


Fig 0.7: Credentials must be matched with record

1. After successful authentication, user will be redirected to dashboard.

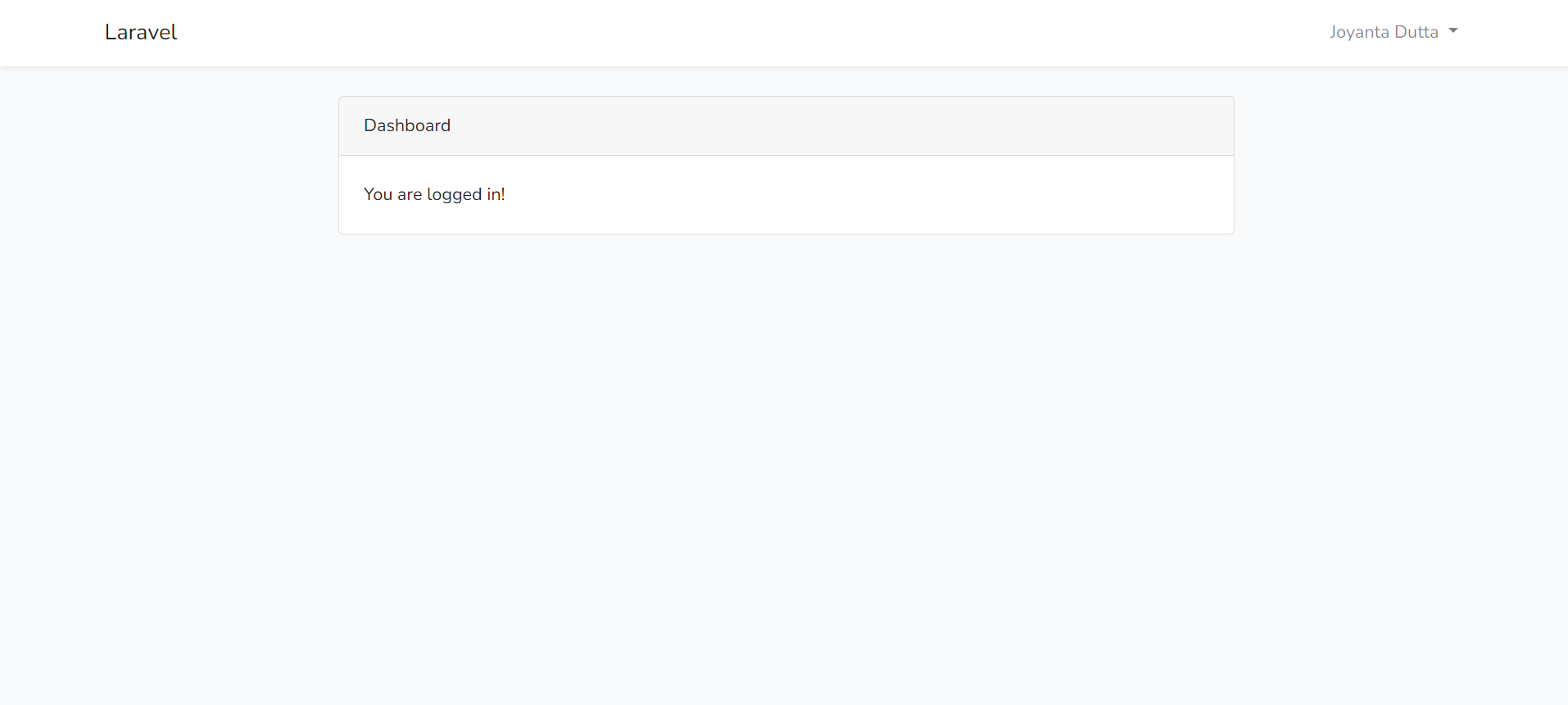


Fig 0.8: User dashboard after successful login

**Where I have changed:**

In users table

Schema::create('users', function (Blueprint $table) {

            $table->bigIncrements('id');

            $table->string('name');

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

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

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

            $table->boolean('is\_otp')->default(false);

            $table->timestamp('email\_verified\_at')->nullable();

            $table->string('password');

            $table->rememberToken();

            $table->timestamps();

        });

I included otp and is\_otp attribute to user model

protected $fillable = [

        'name', 'email', 'phonenumber', 'otp', 'is\_otp', 'password',

    ];

I used otp generator, by using these steps, given below:

1. composer require seshac/otp-generator
2. php artisan vendor:publish --provider="Seshac\Otp\OtpServiceProvider" --tag="migrations"
3. php artisan migrate
4. php artisan vendor:publish --provider="Seshac\Otp\OtpServiceProvider" --tag="config"

Created a otp migration table

Schema::create('otps', function (Blueprint $table) {

            $table->bigIncrements('id');

            $table->string('identifier');

            $table->string('token');

            $table->integer('validity');

            $table->boolean('expired')->default(false);

            $table->integer('no\_times\_generated')->default(0);

            $table->integer('no\_times\_attempted')->default(0);

            $table->timestamp('generated\_at');

            $table->timestamps();

        });

I included this

use Seshac\Otp\Otp;

in the RegisterController

Included validation operation for phone number also,

return Validator::make($data, [

            'name' => ['required', 'string', 'max:255'],

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

            'phonenumber' => ['required', 'regex:/^([0-9\s\-\+\(\)]\*)$/', 'min:10', 'unique:users'],

            'password' => ['required', 'string', 'min:8', 'confirmed'],

        ]);

User will be registered with otp generating and with creating a record also in otp table with some conditional attributes.

$otp =  Otp::setValidity(30)  // otp validity time in mins

            ->setLength(4)  // Lenght of the generated otp

            ->setMaximumOtpsAllowed(10) // Number of times allowed to regenerate otps

            ->setOnlyDigits(false)  // generated otp contains mixed characters ex:ad2312

            ->setUseSameToken(true) // if you re-generate OTP, you will get same token

            ->generate($identifier = $data['phonenumber']);

        return User::create([

            'name' => $data['name'],

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

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

            'otp' => $otp->token,

            'password' => Hash::make($data['password']),

        ]);

I changed the register function of RegistersUsers.php (trait) to redirect to the page of otp verification

public function register(Request $request)

    {

        $this->validator($request->all())->validate();

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

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

        // return $this->registered($request, $user)

        //                 ?: redirect($this->redirectPath());

        return redirect()->route('auth.otp\_verify')->with('success', 'OTP sent to your phone number');

    }

For the otp verification, I declared the new function in register controller

public function verify\_at(Request $request)

    {

        $is\_exist = User::where('otp', $request->otp)->first();

        if (!empty($is\_exist)) {

            User::where('otp', $is\_exist->otp)->update(['is\_otp' => 1]);

            return $this->getIn($is\_exist);

        } else {

            return redirect()->route('auth.otp\_verify')->with('error', 'OTP is not valid');

        }

    }

If otp exists, users tables is\_otp attributes record will be changed to 1. and users info will be returned by getin() function to the RegistersUsers trait.

Then, via

public function getIn($user)

    {

        $this->guard()->login($user);

        return redirect($this->redirectPath());

    }

this method, users all data will stored in auth guard with authentication, and user will be redirected to the users dashboard.

In AuthenticatesUsers.php (trait), I changed the username functions declared returning attribute email to phone number to help user login by using phone number.

public function username()

    {

        return 'phonenumber';

    }

I changed the login functions code

public function login(Request $request)

    {

        $this->validateLogin($request);

        // If the class is using the ThrottlesLogins trait, we can automatically throttle

        // the login attempts for this application. We'll key this by the username and

        // the IP address of the client making these requests into this application.

        $is = User::where('phonenumber', $request->phonenumber)->first();

        if($is){

            if($is->is\_otp == 0){

                return redirect()->back()->with('error', 'OTP is not verified');

            }

        }

        else{

            return redirect()->back()->with('error', 'Phone number is not valid');

        }

        if (method\_exists($this, 'hasTooManyLoginAttempts') &&

            $this->hasTooManyLoginAttempts($request)) {

            $this->fireLockoutEvent($request);

            return $this->sendLockoutResponse($request);

        }

        if ($this->attemptLogin($request)) {

            return $this->sendLoginResponse($request);

        }

        // If the login attempt was unsuccessful we will increment the number of attempts

        // to login and redirect the user back to the login form. Of course, when this

        // user surpasses their maximum number of attempts they will get locked out.

        $this->incrementLoginAttempts($request);

        return $this->sendFailedLoginResponse($request);

    }

Included these,

$is = User::where('phonenumber', $request->phonenumber)->first();

        if($is){

            if($is->is\_otp == 0){

                return redirect()->back()->with('error', 'OTP is not verified');

            }

        }

        else{

            return redirect()->back()->with('error', 'Phone number is not valid');

        }

to check otp is verified or not, and phone number is valid or not.

If all ok, then user can get into dashboard with a new session.

**Project Link:**

1. <https://github.com/Joyanta1000/TaskInLaravel>

**Resources:**

1. <https://github.com/seshac/otp-generator>
2. <https://codebrisk.com/blog/generate-otp-with-expiry-in-laravel-without-using-database>

**General Information:**

Developer: Joyanta Dutta