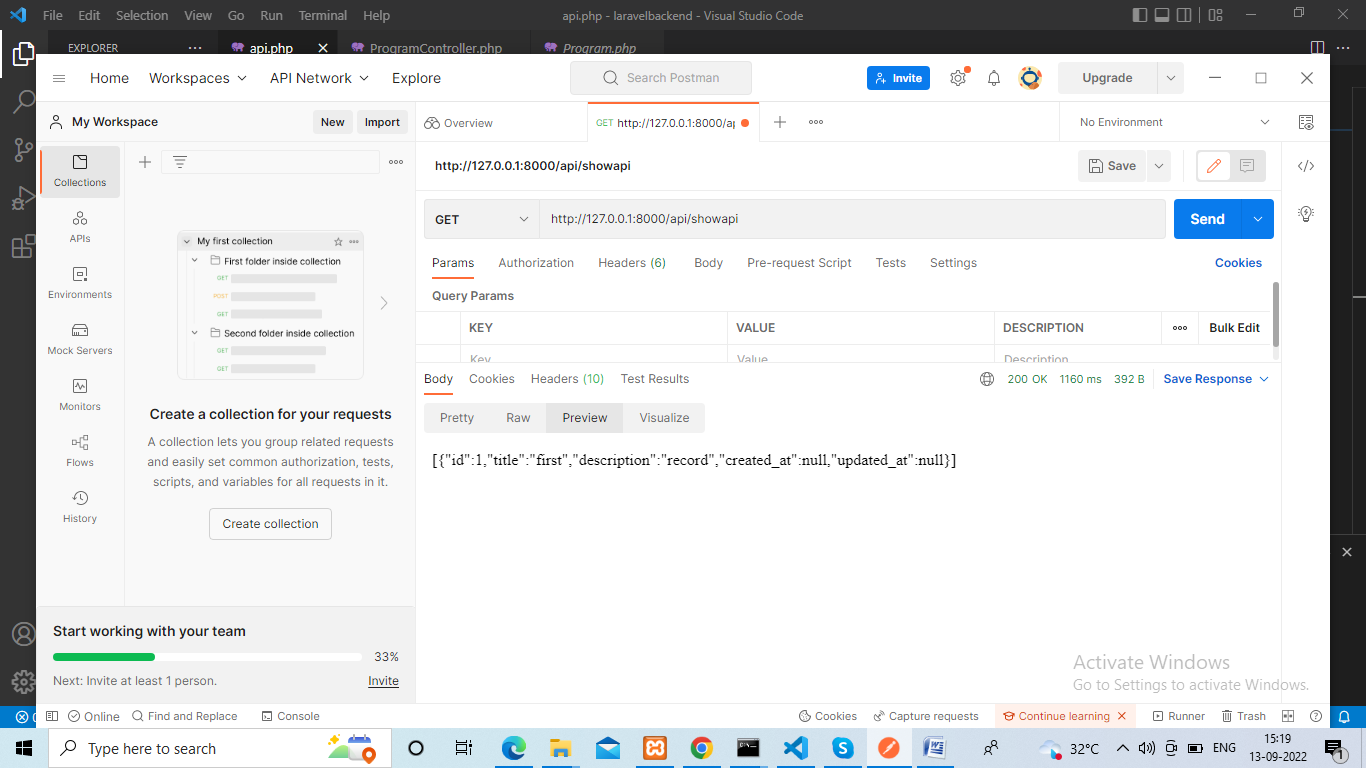
**Rest API**

**For detail :-** [**https://www.youtube.com/watch?v=xvqPEEpRBJ4**](https://www.youtube.com/watch?v=xvqPEEpRBJ4) **code with dary**

https://www.youtube.com/watch?v=k7eyOZwUIKg&list=PLd5VPTWezLeu7tTzVj0AvU9PQE0YfmEjS&index=4

1. Download postman
2. To run api firstly run php artisan serve
3. Copy url localhost:8000 and past in postman with \api\routeURL

Eg. Route::get('showapi',[ProgramController::class,'showdata']);

1. 

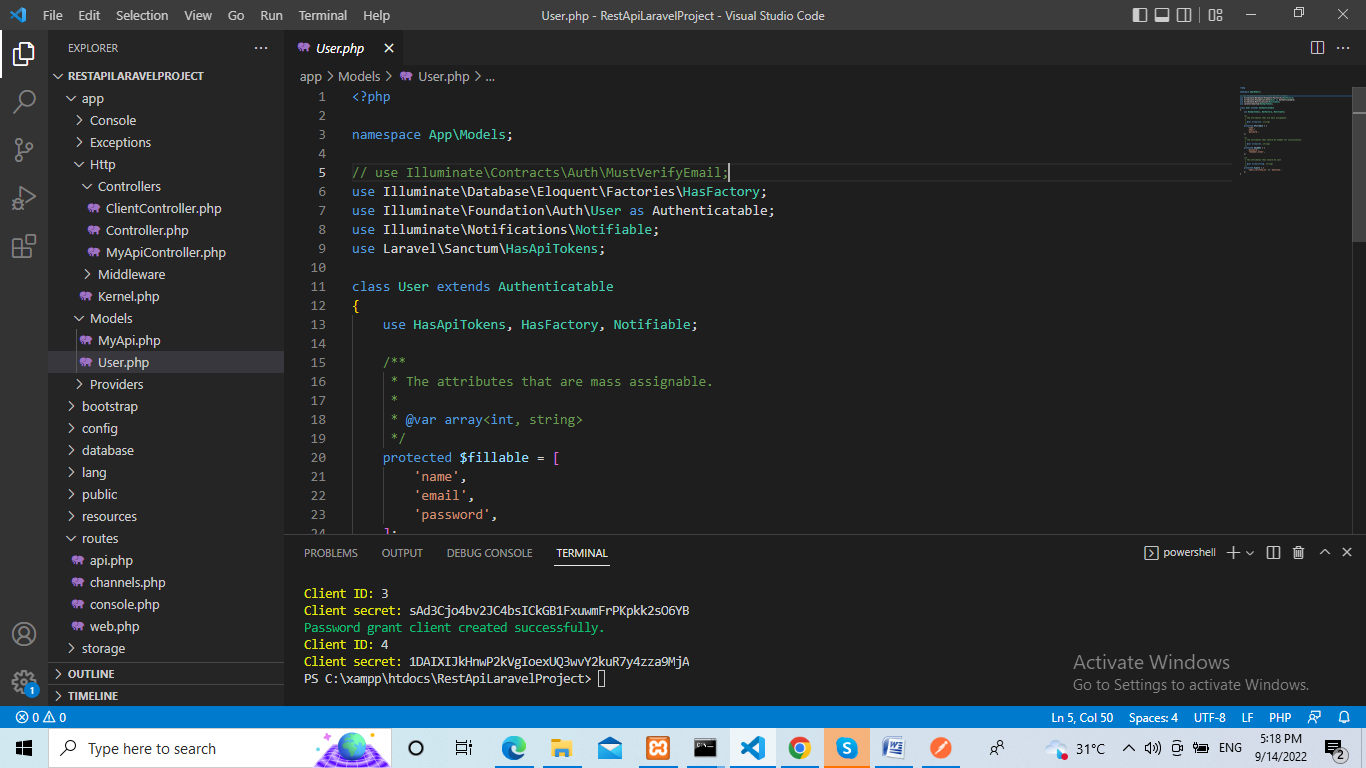
* **To get particular id and all data when we want**
* **Rest API for Register/login/Logout using tokens**

**Follow this official doc for installation**

* [Laravel Passport - Laravel - The PHP Framework For Web Artisans](https://laravel.com/docs/9.x/passport)

Laravel passport used to generate tokens for user login logout

Follow these steps in sequence

1. Composer require laravel/passport
2. Php artisan migrate
3. Php artisan passport:install
4. 

Change line no 9 with this line (change Sanctrum with Passport)

use Laravel\Passport\HasApiTokens;

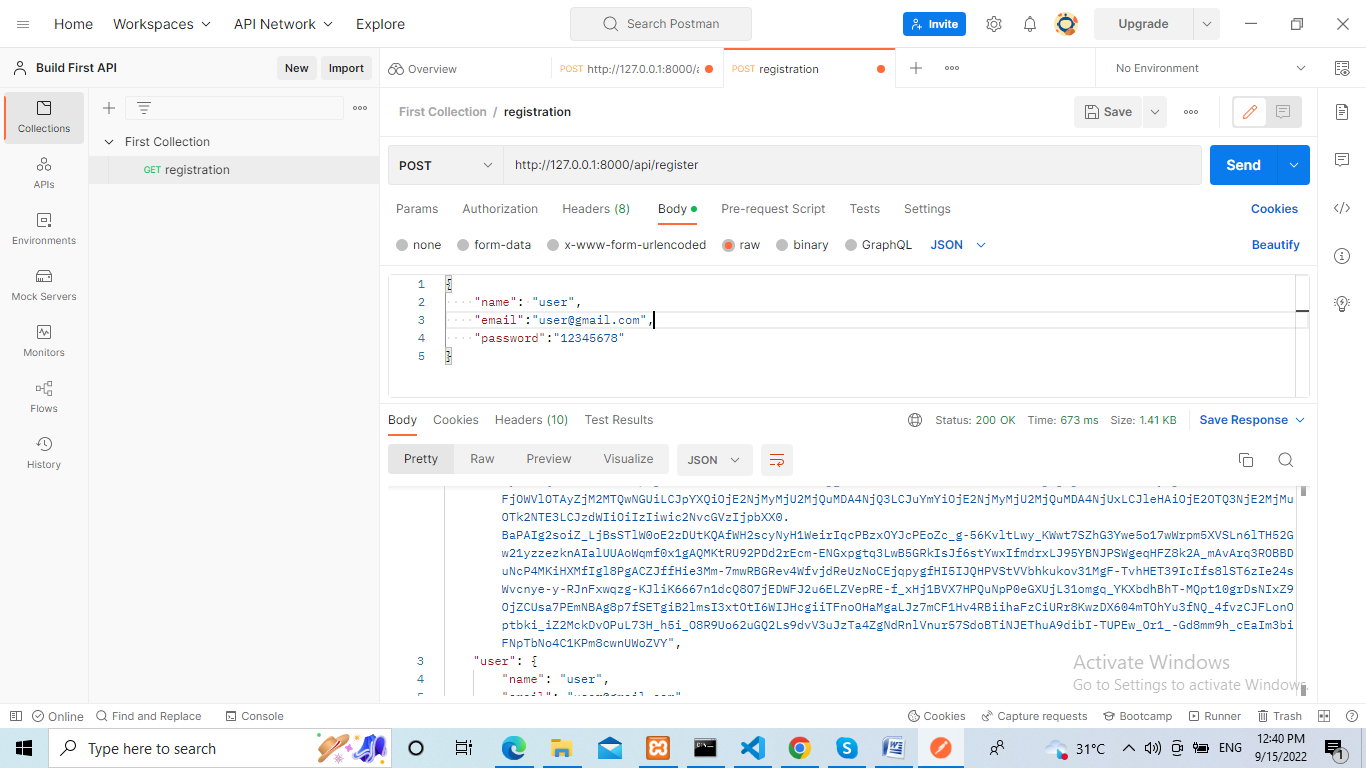
1. Providers AuthServiceProvider.php :- add this line

use Laravel\Passport\Passport;

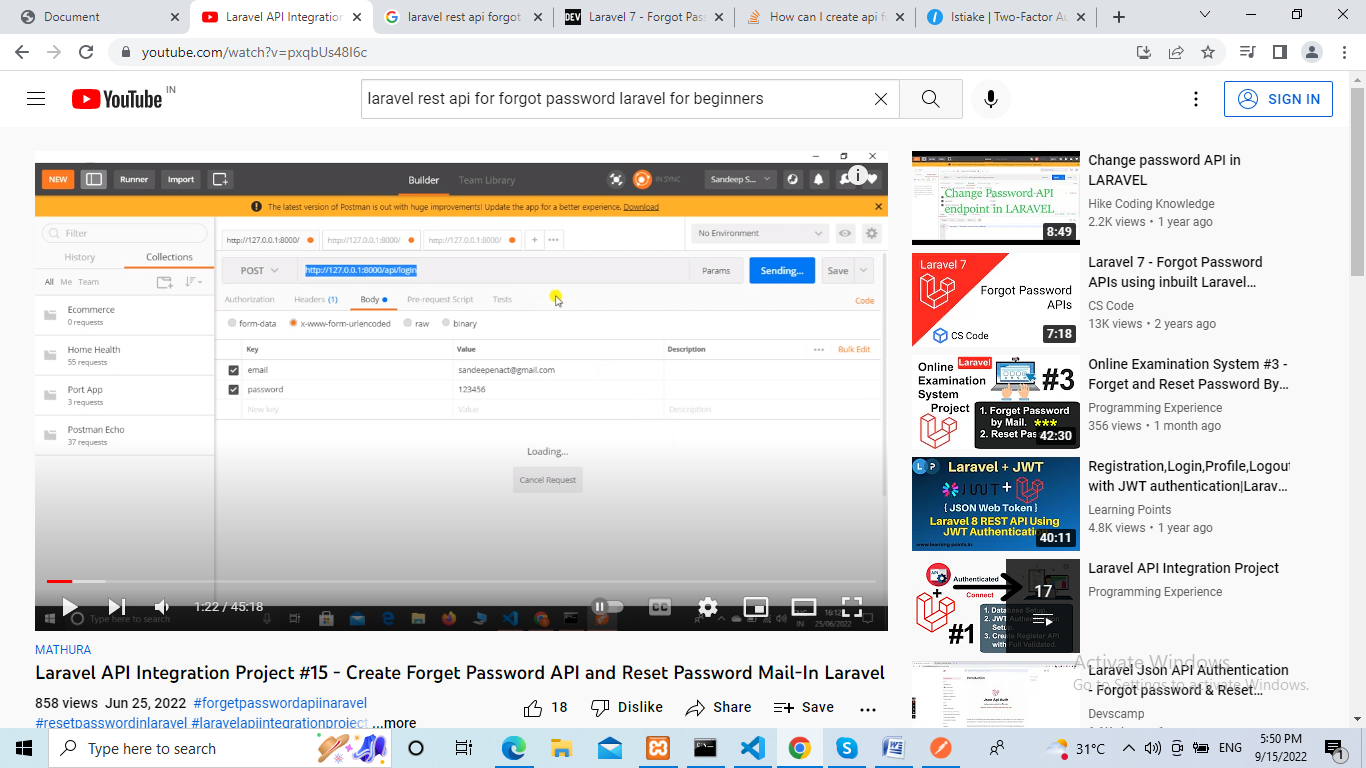
uncomment line no 17. Policiesline

1. Confide  auth.php
2. 'guards' => [
3. 'web' => [
4. 'driver' => 'session',
5. 'provider' => 'users',
6. ],
7. 'api' => [
8. 'driver' => 'passport',
9. 'provider' => 'users',
10. ],
11. ],

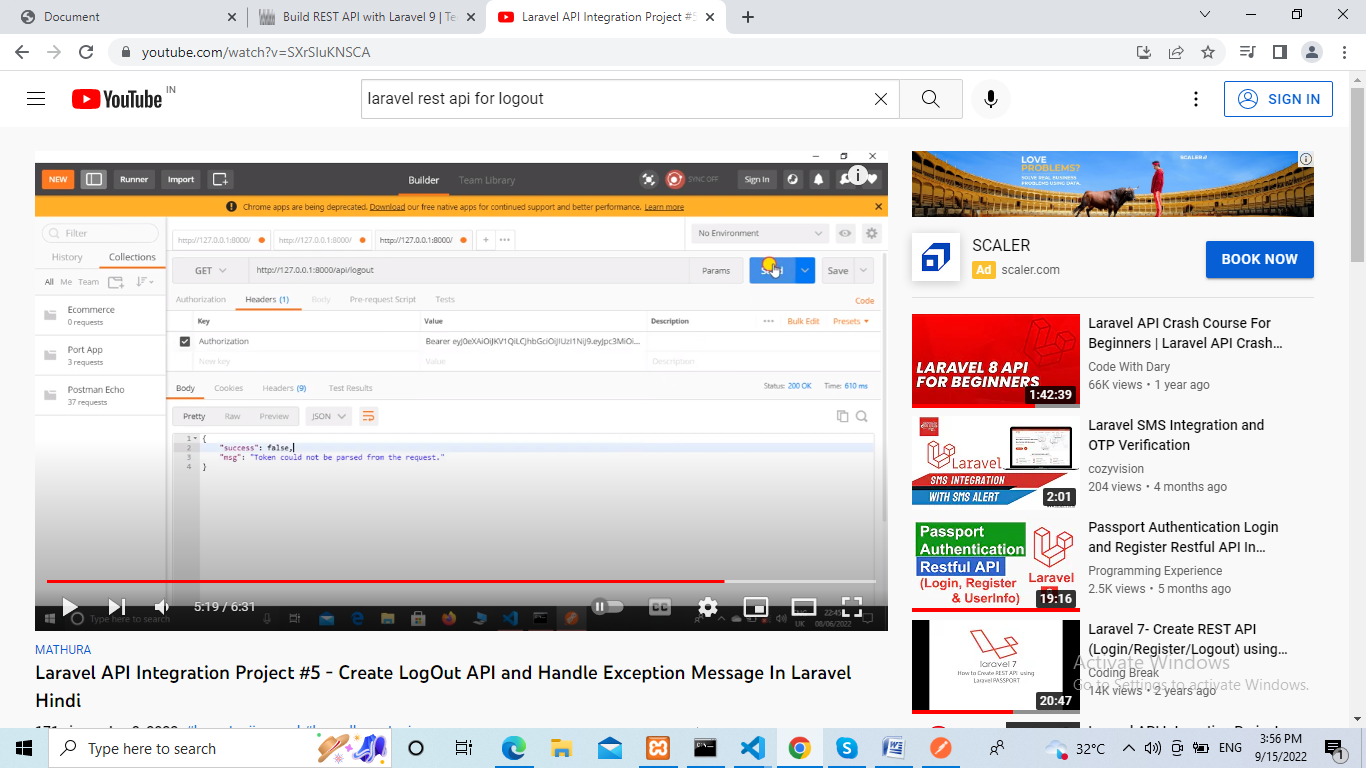
8.in postman setting and values for registration user



Code code is in controller of project :- RestApiLaravelProject



How to pass token for logout



Registration Api with validation

 public function register(Request $request)

    {

        try{

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

                'name'=> 'required|max:15',

                'email'=>'required|email|unique:users',

                'password'=>'required|min:6',

            ]);

            if($validator->fails()){

                return response(['errors'=>$validator->errors()->all()],401);

            }

            $user = new User;

            $user->name = $request->name;

            $user->email = $request->email;

            $user->password = Hash::make($request->password);

            $user->save();

            $token = $user->createToken('Token')->accessToken;

            return response()->json(['token' => $token, 'users' => $user], 200);

            }

        catch (Throwable $e) {

            return response()->json(['error'=>$e]);

        }

    }

Or second way to apply validation in registration

 public function validation(Request $request)

    {

        $rules = array(

            "title" => "required|min:3|max:8",

            "description" => "required",

        );

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

        if ($validator->fails()) {

            return response()->json($validator->errors(), 401);

        } else {

            $user = new MyApi;

            $user->title = $request->title;

            $user->description = $request->description;

            $result = $user->save();

            if ($result) {

                return ["return" => "data has been saved..."];

            } else {

                return ["return" => "data not saved.."];

            }

        }

    }

Route in api.php

Route::post('register', [AuthController::class,'register']);

Route::post('login', [AuthController::class,'login']);

Route::get('showuser/{id}', [AuthController::class,'show']);

CRUD

Route::get('/index',[ProductController::class,'index'])->name('index');

Route::post('/store',[ProductController::class,'store'])->name('store');

Route::delete('/delete/{id}',[ProductController::class,'delete'])->name('delete');

Route::post('/update/{id}',[ProductController::class,'update'])->name('update');

<?php

namespace App\Http\Controllers;

use App\Models\Product;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Validator;

use Illuminate\Support\Facades\Hash;

use app\Models\User;

class ProductController extends Controller

{

// public function register(Request $request)

// {

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

// 'name' =>'required',

// 'email' =>'required|email',

// 'password' =>'required',

// ]);

// if($validator->fails()){

// return response()->json($validator->errors()->toJson(), 400);

// }

// $input = $request->all();

// $input['password'] = Hash::make($input['password']);

// $user = User::create($input);

// $token = JWTAuth::fromUser($user);

// return response()->json(compact('user','token'), 201);

// }

public function store(Request $request)

{

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

'name' => 'required',

'details' => 'required',

]);

if ($validator->fails()) {

return response(['errors' => $validator->errors()->all()], 401);

}

$user = new Product;

$user->name = $request->name;

$user->details = $request->details;

$user->save();

return response()->json(['message' => 'Data stored successfully', 'users' => $user], 201);

}

public function index(Request $request)

{

$userDetails = Product::all();

// dd($userDetails);

return response()->json($userDetails);

}

public function delete($id)

{

$detail = Product::where('id', $id)->delete();

return response()->json(['message' => 'Data deleted successfully'], 201);

}

public function update(Request $request, $id)

{

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

'name' => 'required',

'details' => 'required',

]);

if ($validator->fails()) {

return response(['errors' => $validator->errors()->all()], 401);

}

$userDetail = Product::find($id);

// dd($userDetail);

$userDetail->name = $request->name;

$userDetail->details = $request->details;

$userDetail->save();

return response()->json(['message' => 'Record updated successfully'], 200);

}

}