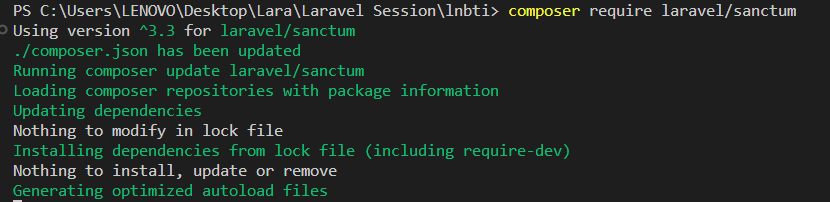
Laravel Authentication

overview of the steps you've taken:

1. Database Setup: Configured your database in the .env file.
2. Installation: Installed Laravel Sanctum via Composer.
3. Publishing Configuration: Published the Sanctum configuration and migration files.
4. Migration: Ran migrations to create necessary tables.
5. Middleware: Added Sanctum's middleware to the API middleware group.
6. Token Usage: Utilized API tokens for users by adding HasApiTokens trait to the User model.
7. Seeder Creation: Created a seeder for the User model.
8. Seeding Users: Inserted records into the users table using the seeder.
9. Controller Creation: Created a controller to handle user authentication.
10. Login Route: Defined a login route in the routes/api.php file.
11. Testing: Tested the login functionality using Postman.

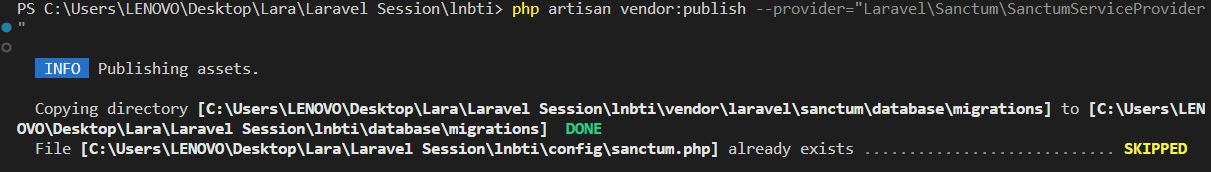
install Laravel Sanctum via the Composer package manager

composer require laravel/sanctum



publish the Sanctum configuration and migration files

php artisan vendor:publish --provider="Laravel\Sanctum\SanctumServiceProvider"



The sanctum configuration file would be placed in your application's config directory.

Run database migrations

php artisan migrate

If you plan to utilize Sanctum to authenticate a SPA, Add Sanctum's middleware to your api middleware group within your application's app/Http/Kernel.php file,

../app/Http/Kernel.php

use Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful;

...

protected $middlewareGroups = [

...

'api' => [

\Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,

            \Illuminate\Routing\Middleware\ThrottleRequests::class.':api',

            \Illuminate\Routing\Middleware\SubstituteBindings::class,

],

];

...

],

**Issuing API Tokens**

Sanctum allows you to issue API tokens / personal access tokens that may be used to authenticate API requests to your application.

Go to User model,

use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable

{

use HasApiTokens, HasFactory, Notifiable;

}

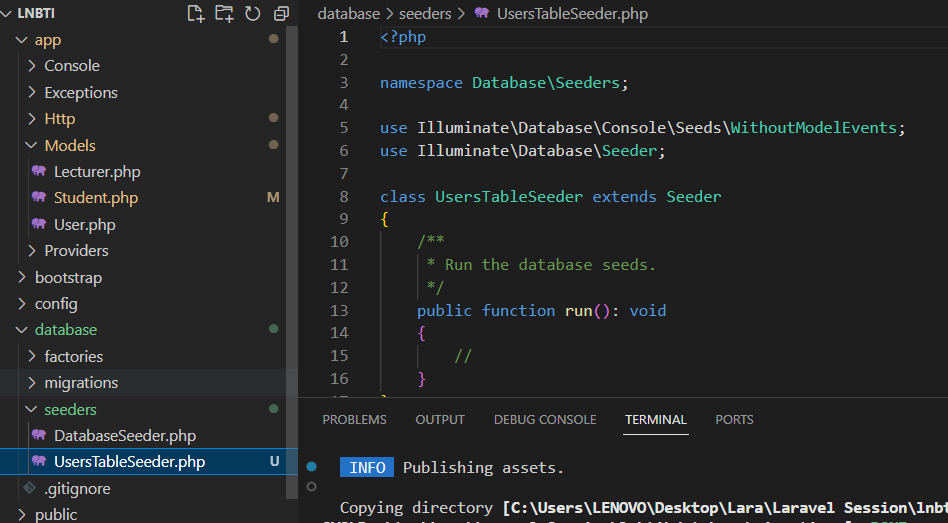
If User model is not created create it

php artisan make:model User

Let's create the seeder for the User model

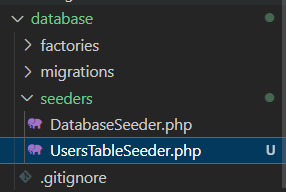
php artisan make:seeder UsersTableSeeder

Check whether UserTableSeeder is created?



Now let's insert as record

Go to



Add dependencies and add records inside the run function

use Illuminate\Support\Facades\DB;

use Illuminate\Support\Facades\Hash;

...

...

DB::table('users')->insert([

'name' => 'John Doe',

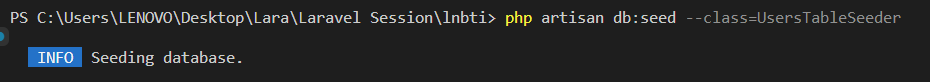
'email' => 'john@doe.com',

'password' => Hash::make('password')

]);

To seed users table with user

php artisan db:seed --class=UsersTableSeeder



Create a controller nad /login route in the routes/api.php file:

php artisan make:controller UserController

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\User;

use Illuminate\Support\Facades\Hash;

class UserController extends Controller

{

//

function index(Request $request)

{

$user= User::where('email', $request->email)->first();

// print\_r($data);

if (!$user || !Hash::check($request->password, $user->password)) {

return response([

'message' => ['These credentials do not match our records.']

], 404);

}

$token = $user->createToken('my-app-token')->plainTextToken;

$response = [

'user' => $user,

'token' => $token

];

return response($response, 201);

}

}

Make Details API or any other with secure route

use App\Http\controllers\UserController;

Route::group(['middleware' => 'auth:sanctum'], function(){

//All secure URL's

Route::resource('/lecturer', LecturerController::class);

Route::resource('/student', StudentController::class);

});

Route::post("login",[UserController::class,'index']);

Test through Postman

Create blade.php file for login

Create balde.php file for register

Add register for user controller

Update UserController

Add Logout function to the UserController

    public function logout(Request $request)

    {

        Auth::logout();

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

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

        return redirect('/login');

    }

Updated UserController

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Models\User;

use Illuminate\Support\Facades\Hash;

use Illuminate\Support\Facades\Auth;

class UserController extends Controller

{

    public function index(Request $request)

    {

        // Validate user credentials

        $credentials = $request->only('email', 'password');

        if (Auth::attempt($credentials)) {

            // Authentication successful, redirect to /student

            return redirect('/student');

        }

        // Authentication failed, redirect back to login with error message

        return redirect('/login')->with('error', 'Invalid credentials.');

    }

    public function register(Request $request)

    {

        // Validate the request data

        $request->validate([

            'name' => 'required|string|max:255',

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

            'password' => 'required|string|min:8|confirmed',

        ]);

        // Create a new user

        $user = User::create([

            'name' => $request->name,

            'email' => $request->email,

            'password' => Hash::make($request->password),

        ]);

        // Assuming you want to automatically log in the user after registration,

        // you can generate a token for the user

        $token = $user->createToken('my-app-token')->plainTextToken;

        // return response()->json([

        //     'user' => $user,

        //     'token' => $token,

        // ], 201);

        return redirect('/login');

    }

    public function logout(Request $request)

    {

        Auth::logout();

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

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

        return redirect('/login');

    }

}

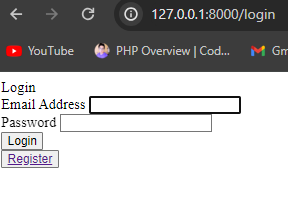
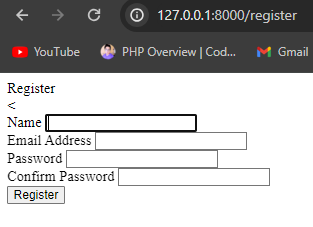
Add Logout Button for layout Page

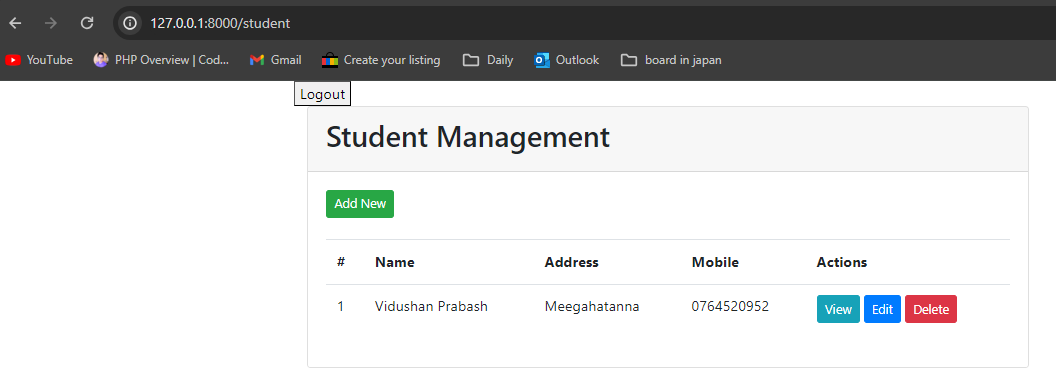
<form method="POST" action="{{ route('logout') }}">

        @csrf

        <button type="submit">Logout</button>

</form>

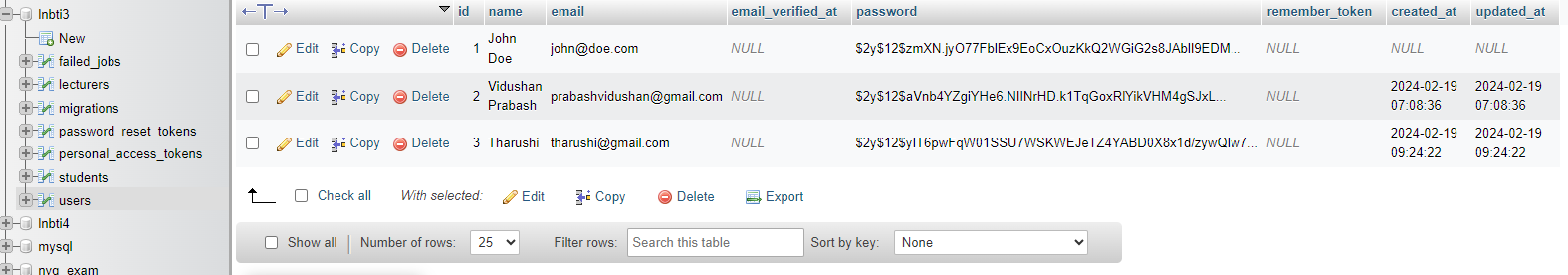
 



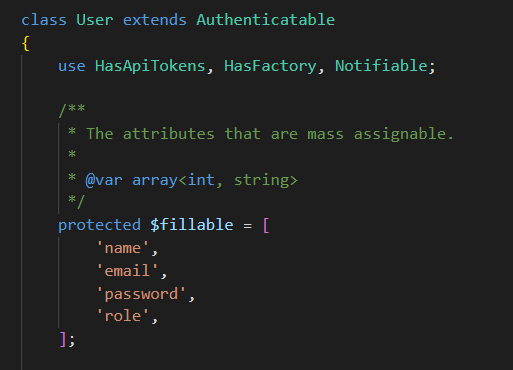
Add authorizations

integrate authorization

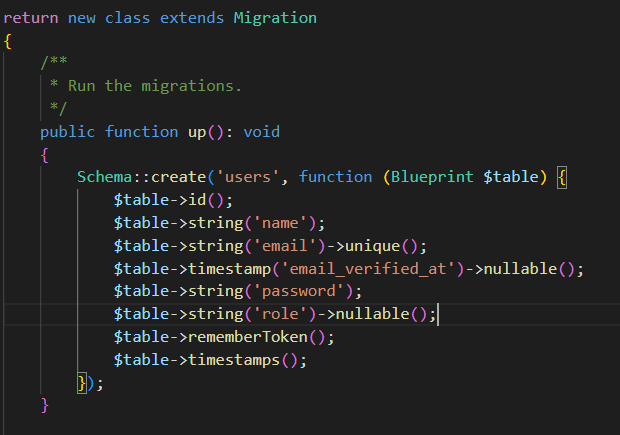
* If registered role as a nullable he can access index student, show student, update student.
* If registered role as a registrar he can access index student, show student, update student, create student.
* If registered role as an admin he can access index student, show student, update student, create student, delete student.



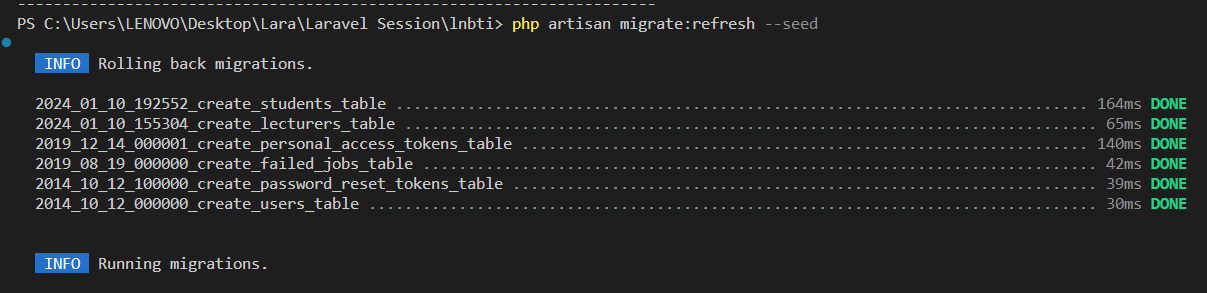
Add role(new attribute) to the User model

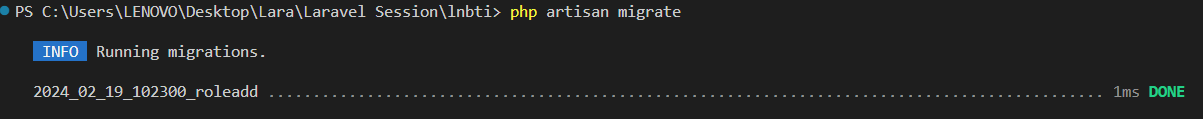


Add role to the migration file



**php artisan migrate:refresh –seed**





Create a middleware

php artisan make:middleware RoleMiddleware

<?php

namespace App\Http\Middleware;

use Closure;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class RoleMiddleware

{

    public function handle(Request $request, Closure $next, ...$roles)

    {

        $user = Auth::user();

        if ($user && in\_array($user->role, $roles)) {

            return $next($request);

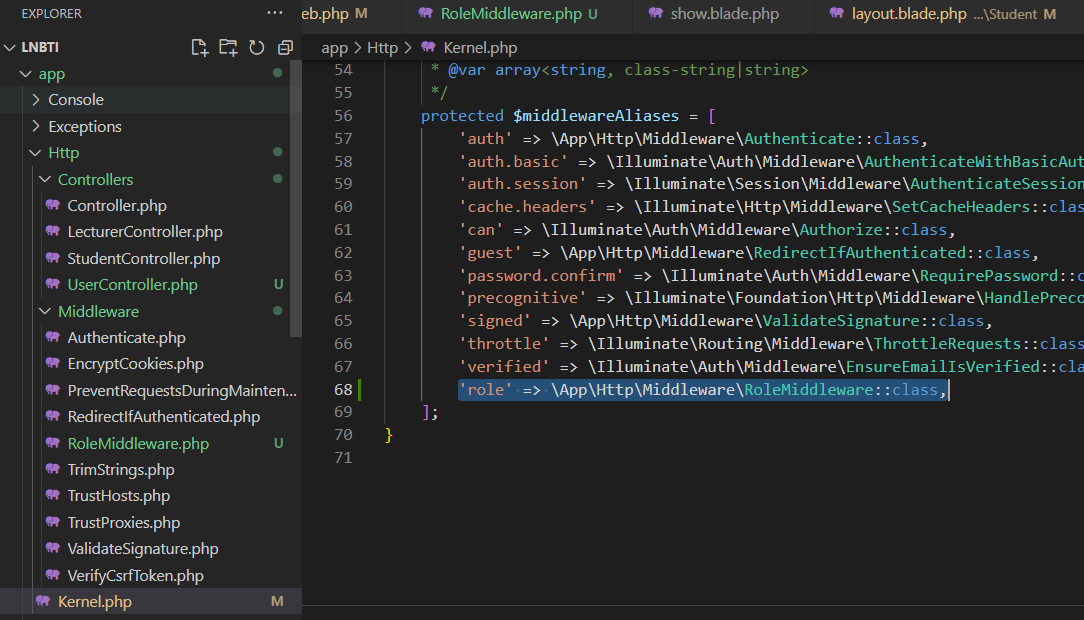
        }

        return redirect('/login')->with('error', 'Unauthorized access.');

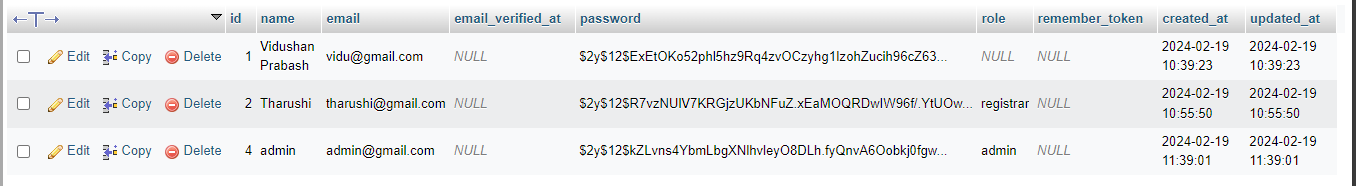
    }

}

Add register middleware



Note : Role should manually add on mysql database, All general registered users consider as a student.



StudentController

<?php

namespace App\Http\Controllers;

use App\Models\Student;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;

class StudentController extends Controller

{

    public function index()

    {

        $students = Student::all();

        return view('Student.index')->with('students', $students);

    }

    public function create()

    {

        if (Auth::user()->role == 'admin' || Auth::user()->role == 'registrar') {

            return view('Student.create');

        } else {

            return redirect()->route('student.index')->with('error', 'Unauthorized access');

        }

    }

    public function store(Request $request)

    {

        if (Auth::user()->role == 'admin' || Auth::user()->role == 'registrar') {

            $input = $request->all();

            Student::create($input);

            return redirect('student')->with('flash\_message', 'Student Added!');

        } else {

            return redirect()->route('student.index')->with('error', 'Unauthorized access');

        }

    }

    public function show(Student $student)

    {

        return view('Student.show')->with('student', $student);

    }

    public function edit(Student $student)

    {

        if (Auth::user()->role == 'admin' || Auth::user()->role == '') {

            return view('Student.edit')->with('student', $student);

        } else {

            return redirect()->route('student.index')->with('error', 'Unauthorized access');

        }

    }

    public function update(Request $request, Student $student)

    {

        if (Auth::user()->role == 'admin' || Auth::user()->role == '') {

            $input = $request->all();

            $student->update($input);

            return redirect('student')->with('flash\_message', 'Student Updated!');

        } else {

            return redirect()->route('student.index')->with('error', 'Unauthorized access');

        }

    }

    public function destroy(Student $student)

    {

        if (Auth::user()->role == 'admin') {

            $student->delete();

            return redirect('student')->with('flash\_message', 'Student deleted!');

        } else {

            return redirect()->route('student.index')->with('error', 'Unauthorized access');

        }

    }

}

Web.php

<?php

use Illuminate\Support\Facades\Route;

use App\Http\Controllers\LecturerController;

use App\Http\Controllers\StudentController;

use App\Http\Controllers\UserController;

Route::get('/', function () { return view('welcome'); });

Route::group(['middleware' => 'auth:sanctum'], function () {

    Route::resource('/lecturer', LecturerController::class);

    Route::resource('/student', StudentController::class);

});

Route::post('login', [UserController::class, 'index'])->name('login');

Route::get('login', function () { return view('login'); });

Route::get('/register', function () { return view('register'); })->name('register');

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

Route::post('/logout', [UserController::class, 'logout'])->name('logout');

// apply the auth:sanctum middleware to a group of routes using Route::group()

// Redirect authenticated users to the /student route

Route::middleware('auth')->group(function () {

    Route::get('/home', function () {

        return redirect('/student');

    });

});