**🔐 Самый простой способ сделать авторизацию в Laravel + Vue + Vue Router**

Для минимальной системы аутентификации:  
✅ Используем **JWT (JSON Web Token)** для API-аутентификации в Laravel  
✅ **Axios** для запросов с токеном  
✅ **Vue Router** для защиты маршрутов

**1️⃣ Установка Laravel Sanctum**

Sanctum — самый простой способ аутентификации в Laravel API. Устанавливаем его:

bash

КопироватьРедактировать

composer require laravel/sanctum

Публикуем конфигурацию:

bash

КопироватьРедактировать

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

Добавляем Sanctum в api-middleware в app/Http/Kernel.php:

php

КопироватьРедактировать

use Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful;

protected $middlewareGroups = [

'api' => [

EnsureFrontendRequestsAreStateful::class,

'throttle:api',

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

],

];

Запускаем миграции:

bash

КопироватьРедактировать

php artisan migrate

**2️⃣ Добавляем аутентификацию в Laravel**

Создадим **маршруты API** в routes/api.php:

php

КопироватьРедактировать

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Route;

use App\Http\Controllers\AuthController;

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

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

Route::post('/logout', [AuthController::class, 'logout'])->middleware('auth:sanctum');

Route::get('/user', [AuthController::class, 'user'])->middleware('auth:sanctum');

**Создаём AuthController.php**:

bash

КопироватьРедактировать

php artisan make:controller AuthController

Добавляем в него код:

php

КопироватьРедактировать

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\Hash;

use App\Models\User;

class AuthController extends Controller

{

public function register(Request $request)

{

$request->validate([

'name' => 'required|string',

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

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

]);

$user = User::create([

'name' => $request->name,

'email' => $request->email,

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

]);

return response()->json(['token' => $user->createToken('token')->plainTextToken]);

}

public function login(Request $request)

{

$request->validate([

'email' => 'required|string|email',

'password' => 'required|string'

]);

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

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

return response()->json(['message' => 'Неверные данные'], 401);

}

return response()->json(['token' => $user->createToken('token')->plainTextToken]);

}

public function user(Request $request)

{

return response()->json($request->user());

}

public function logout(Request $request)

{

$request->user()->tokens()->delete();

return response()->json(['message' => 'Вы вышли']);

}

}

💡 **Sanctum работает с createToken()**, который выдаёт токен для API-запросов.

**3️⃣ Настраиваем Vue (Frontend)**

**Устанавливаем Vue Router и Axios**

bash

КопироватьРедактировать

npm install vue-router axios

**📌 Настроим маршруты (router/index.js)**

js

КопироватьРедактировать

import { createRouter, createWebHistory } from 'vue-router';

import Login from '../views/Login.vue';

import Register from '../views/Register.vue';

import Dashboard from '../views/Dashboard.vue';

const routes = [

{ path: '/login', component: Login },

{ path: '/register', component: Register },

{ path: '/dashboard', component: Dashboard, meta: { requiresAuth: true } }

];

const router = createRouter({

history: createWebHistory(),

routes

});

// Защита маршрутов (если нет токена - редирект на логин)

router.beforeEach((to, from, next) => {

const isAuthenticated = localStorage.getItem('token');

if (to.meta.requiresAuth && !isAuthenticated) {

next('/login');

} else {

next();

}

});

export default router;

💡 🔐 **Защита маршрутов:**  
Если у страницы стоит meta: { requiresAuth: true }, но нет токена – **редирект на /login**.

**4️⃣ Добавляем API-запросы в Vue**

**📌 Форма логина (views/Login.vue)**

vue

КопироватьРедактировать

<template>

<div>

<h2>Вход</h2>

<input v-model="email" placeholder="Email">

<input v-model="password" type="password" placeholder="Пароль">

<button @click="login">Войти</button>

</div>

</template>

<script>

import axios from 'axios';

export default {

data() {

return { email: '', password: '' };

},

methods: {

async login() {

try {

const response = await axios.post('/api/login', {

email: this.email,

password: this.password

});

localStorage.setItem('token', response.data.token);

axios.defaults.headers.common['Authorization'] = `Bearer ${response.data.token}`;

this.$router.push('/dashboard');

} catch (error) {

alert('Ошибка входа');

}

}

}

};

</script>

**📌 Личный кабинет (views/Dashboard.vue)**

vue

КопироватьРедактировать

<template>

<div>

<h2>Привет, {{ user.name }}</h2>

<button @click="logout">Выйти</button>

</div>

</template>

<script>

import axios from 'axios';

export default {

data() {

return { user: {} };

},

async mounted() {

try {

const response = await axios.get('/api/user', {

headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }

});

this.user = response.data;

} catch {

this.$router.push('/login');

}

},

methods: {

logout() {

axios.post('/api/logout', {}, {

headers: { Authorization: `Bearer ${localStorage.getItem('token')}` }

}).then(() => {

localStorage.removeItem('token');

this.$router.push('/login');

});

}

}

};

</script>

**5️⃣ Готово! 🎉**

🔥 **Как работает авторизация?**  
✅ Login.vue делает POST /api/login, получает token, сохраняет его в localStorage.  
✅ Все защищённые маршруты проверяют токен перед загрузкой (beforeEach в router/index.js).  
✅ Dashboard.vue делает GET /api/user, если токен действителен – показывает имя пользователя.

🎯 **Теперь у вас есть простая система авторизации в Laravel + Vue!** 🚀