Video : <https://www.youtube.com/watch?v=jIzPuM76-nI&list=LL&index=4&t=2s&ab_channel=ScalableScripts>

Step 0 : khali user table er migration thakbe jetar structure minimum level e erokom, (baki migration gula remove kore dite hobe),

Remember token tao remove kore dite hobe, ota Sanctum er shathe create hobe migration er time,

In user\_table\_migation,

<?php  
  
use Illuminate\Database\Migrations\Migration;  
use Illuminate\Database\Schema\Blueprint;  
use Illuminate\Support\Facades\Schema;  
  
class CreateUsersTable extends Migration  
{  
 /\*\*  
 \* Run the migrations.  
 \*  
 \* @return void  
 \*/  
 public function up()  
 {  
 Schema::create('users', function (Blueprint *$table*) {  
 *$table*->id();  
 *$table*->string('name');  
 *$table*->string('email')->unique();  
 *$table*->string('password');  
 *$table*->timestamps();  
 });  
 }  
  
 /\*\*  
 \* Reverse the migrations.  
 \*  
 \* @return void  
 \*/  
 public function down()  
 {  
 Schema::dropIfExists('users');  
 }  
}

Step 1 : composer require laravel/sanctum

Step 2 : php artisan vendor:publish --provider="Laravel\Sanctum\SanctumServiceProvider"

Step 3 : php artisan migrate

Step 4 : php artisan make:controller AuthController

Step 5 : adding all (use) in AuthController,

use Illuminate\Support\Facades\Cookie;  
use Illuminate\Support\Facades\Hash;  
use Illuminate\Support\Facades\Auth;  
use Symfony\Component\HttpFoundation\Response;

Full AuthCotroller Code,

<?php  
  
namespace App\Http\Controllers;  
  
use App\Models\User;  
use Illuminate\Http\Request;  
use Illuminate\Support\Facades\Cookie;  
use Illuminate\Support\Facades\Hash;  
use Illuminate\Support\Facades\Auth;  
use Symfony\Component\HttpFoundation\Response; //must notice  
  
class AuthController extends Controller  
{  
 public function user()  
 {  
 return Auth::user();  
 }  
  
 public function register(Request *$request*)  
 {  
 $user = User::create([  
 'name' => *$request*->input('name'),  
 'email' => *$request*->input('email'),  
 'password' => Hash::make(*$request*->input('password'))  
 ]);  
  
 return $user;  
 }  
  
 public function login(Request *$request*)  
 {  
  
 if (!Auth::attempt(*$request*->only('email', 'password'))) {  
 return response([  
 'message' => 'Inavalid Credentials'  
 ], Response::*HTTP\_UNAUTHORIZED*);  
 };  
 $user = Auth::user();  
 $token = $user->createToken('token')->plainTextToken;  
 $cookie = cookie('jwt',$token,60\*24); //1day  
 return response([  
 'message' => $token  
 ])->withCookie($cookie);  
  
 }

public function logout(){  
 $cookie = Cookie::forget('jwt');  
  
 return response([  
 'message' => 'success'  
 ])->withCookie($cookie);  
 }  
}

\*\*\*in config/cors.php,(mainly false thake, true kore dite hobe),

'supports\_credentials' => true,

**Then** User.php te use e eta add korte hobe,

use Laravel\Sanctum\HasApiTokens;

Then etake use korte hobe User.php te evabe,

use HasFactory, Notifiable,HasApiTokens;

Full User.php code,

<?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; //use kora hoise  
  
class User extends *Authenticatable*{  
 use HasFactory, Notifiable,HasApiTokens; //using HasApiTokens  
  
 /\*\*  
 \* The attributes that are mass assignable.  
 \*  
 \* @var array  
 \*/  
 protected $fillable = [  
 'name',  
 'email',  
 'password',  
 ];  
  
 /\*\*  
 \* The attributes that should be hidden for arrays.  
 \*  
 \* @var array  
 \*/  
 protected $hidden = [  
 'password',  
 ];  
}

\*\*\* middleware/Authenticate.php te evabe add korte hobe,

public function handle(*$request*, Closure *$next*, ...*$guards*)  
{  
 if ($jwt = *$request*->cookie('jwt')){  
 *$request*->headers->set('Authorization','Bearer '.$jwt);  
 }  
  
 $this->authenticate(*$request*, *$guards*);  
  
 return *$next*(*$request*);  
}

Full Authenticate.php(middleware) code,

In middleware->Authenticate.php,

<?php  
  
namespace App\Http\Middleware;  
  
use Closure;  
use Illuminate\Auth\Middleware\Authenticate as Middleware;  
  
class Authenticate extends *Middleware*{  
 /\*\*  
 \* Get the path the user should be redirected to when they are not authenticated.  
 \*  
 \* @param \Illuminate\Http\Request *$request* \* @return string|null  
 \*/  
 protected function redirectTo(*$request*)  
 {  
 if (! *$request*->expectsJson()) {  
 return route('login');  
 }  
 }  
  
 public function handle(*$request*, Closure *$next*, ...*$guards*)  
 {  
 if ($jwt = *$request*->cookie('jwt')){  
 *$request*->headers->set('Authorization','Bearer '.$jwt);  
 }  
  
 $this->authenticate(*$request*, *$guards*);  
  
 return *$next*(*$request*);  
 }  
}

\*\*\* lastly route ta evabe setup hobe,

Route::post('register',[AuthController::class,'register']);  
Route::post('login',[AuthController::class,'login']);  
  
Route::middleware('auth:sanctum')->group(function (){  
 Route::get('user',[AuthController::class,'user']);  
 Route::post('logout',[AuthController::class,'logout']);  
});

Update 2025, logout route,

From Api.php,

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

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

    Route::post('logout\_all\_devices', [AuthController::class, 'logout\_all\_devices']);

});

Now Code from Controller,

    public function logout(Request $request)

    {

        $token = $request->bearerToken();

        // Revoke the specific token

        $user = Auth::user();

        $user->tokens()->where('id', explode('|', $token)[0])->delete();

        $cookie = Cookie::forget('jwt');

        return response([

            'message' => 'successfully logged out'

        ])->withCookie($cookie);

    }

    public function logout\_all\_devices()

    {

        // ---------------------------------------- Revoke all the token that was used for authentication

        $user = Auth::user();

        $user->tokens()->delete(); // Delete all tokens for the user

        $cookie = Cookie::forget('jwt');

        return response([

            'message' => 'successfully logged out'

        ])->withCookie($cookie);

    }