Laravel Sanctum API Activity

Creating a Laravel 11 application with Laravel Sanctum for API authentication involves several steps. Below, this will guide you through the process of setting up a Laravel 11 project, creating models with relationships, and configuring API endpoints.

Step 1: Set Up Laravel

First, ensure you have Composer installed. Then, create a new Laravel project:

```
composer create-project laravel/laravel laravel-sanctum-api --prefer-dist

cd laravel-sanctum-api
```

Step 2: Migrate the auth models

This is an initial migration. You will need to migrate again later.

```
php artisan migrate
```

Step 3: Install Laravel Sanctum

Install Laravel Sanctum for API authentication:

```
php artisan install:api
```

Step 4: Create Models and Migrations

We'll create three models: User, Post, and Comment. Here are their relationships:

- A User has many Posts.
- A Post has many Comments.
- A Comment belongs to a Post and a User.

Create the models and migrations:

```
php artisan make:model Post -m
php artisan make:model Comment -m
```

Update the Post migration file:

```
// database/migrations/xxxx_xx_xx_create_posts_table.php

public function up()
{
    Schema::create('posts', function (Blueprint $table) {
        $table->id();
```

```
        $table->foreignId('user_id')->constrained()->onDelete('cascade');
        $table->string('title');
        $table->text('body');
        $table->timestamps();
    });
}
```

Update the Comment migration file:

```
// database/migrations/xxxx_xx_xx_create_comments_table.php

public function up()
{
    Schema::create('comments', function (Blueprint $table) {
        $table->id();
        $table->foreignId('post_id')->constrained()->onDelete('cascade');
        $table->foreignId('user_id')->constrained()->onDelete('cascade');
        $table->text('body');
        $table->timestamps();
    });
}
```

Run the migrations:

```
php artisan migrate
```

Step 5: Define Model Relationships

Define the relationships in the models.

User.php:

```
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory;

    public function posts()
    {
        return $this->hasMany(Post::class);
    }

    public function comments()
    {
        return $this->hasMany(Comment::class);
    }
}
```

Post.php:

```
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Post extends Model
{
    use HasFactory;

    protected $fillable = ['user_id', 'title', 'body'];

    public function user()
    {
        return $this->belongsTo(User::class);
    }

    public function comments()
    {
        return $this->hasMany(Comment::class);
    }
}
```

Comment.php:

```
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Comment extends Model
{
    use HasFactory;

    protected $fillable = ['post_id', 'user_id', 'body'];

    public function user()
    {
        return $this->belongsTo(User::class);
    }

    public function post()
    {
        return $this->belongsTo(Post::class);
    }
}
```

Step 6: Create API Controllers and Routes

Create controllers for handling API requests:

```
php artisan make:controller Api\PostController
php artisan make:controller Api\CommentController
```

PostController.php:

```
namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use App\Models\Post;
use Illuminate\Http\Request;

class PostController extends Controller
{
    public function index()
    {
        return Post::with('user', 'comments')->get();
    }

    public function store(Request $request)
    {
        $request->validate([
            'title' => 'required|string|max:255',
            'body' => 'required|string',
        ]);

        $post = Post::create([
            'user_id' => auth()->id(),
            'title' => $request->title,
            'body' => $request->body,
        ]);

        return response()->json($post, 201);
    }

    public function show(Post $post)
    {
        return $post->load('user', 'comments');
    }

    public function update(Request $request, Post $post)
    {
        $this->authorize('update', $post);

        $request->validate([
            'title' => 'sometimes|string|max:255',
            'body' => 'sometimes|string',
        ]);

        $post->update($request->only('title', 'body'));

        return response()->json($post);
    }

    public function destroy(Post $post)
    {
        $this->authorize('delete', $post);

        $post->delete();
```

```
        return response()->json(null, 204);
    }
}
```

CommentController.php:

```
namespace App\Http\Controllers\Api;

use App\Http\Controllers\Controller;
use App\Models\Comment;
use Illuminate\Http\Request;

class CommentController extends Controller
{
    public function store(Request $request)
    {
        $request->validate([
            'post_id' => 'required|exists:posts,id',
            'body' => 'required|string',
        ]);

        $comment = Comment::create([
            'post_id' => $request->post_id,
            'user_id' => auth()->id(),
            'body' => $request->body,
        ]);

        return response()->json($comment, 201);
    }

    public function update(Request $request, Comment $comment)
    {
        $this->authorize('update', $comment);

        $request->validate([
            'body' => 'sometimes|string',
        ]);

        $comment->update($request->only('body'));

        return response()->json($comment);
    }

    public function destroy(Comment $comment)
    {
        $this->authorize('delete', $comment);

        $comment->delete();

        return response()->json(null, 204);
    }
}
```

Add routes in routes\api.php:

```
use App\Http\Controllers\Api\PostController;
use App\Http\Controllers\Api\CommentController;
use Illuminate\Support\Facades\Route;

Route::middleware('auth:sanctum')->group(function () {
    Route::apiResource('posts', PostController::class);
    Route::post('comments', [CommentController::class, 'store']);
    Route::put('comments/{comment}', [CommentController::class, 'update']);
    Route::delete('comments/{comment}', [CommentController::class,
'destroy']);
});
```

Step 7: Test Your API

Ensure you have a way to create users and log in to get tokens. You can use Laravel's built-in authentication scaffolding or manually create user registration and login endpoints.

Since there is no available views necessary you may use Postman as the client to test you API access. You may download Postman through this url: https://www.postman.com/. Register an account it is free anyway.

Alternatively you may use the curl command line utility program that can be downloaded from multiple websites on the internet.

If you might require **Administrator** access to install Postman please ask for assistance from the attending Laboratory assistant.

Conclusion

You now have a basic Laravel 11 API with Laravel Sanctum, three models (User, Post, Comment) with relationships, and endpoints for managing posts and comments. You can further enhance this setup by adding request validation, policies, and more robust error handling.