CRUD operation in Flask with Neo4j Database

Introduction

This project implements a CRUD (Create, Read, Update, Delete) API using Flask and Neo4j. It allows creating, retrieving, updating, and deleting users and posts, as well as establishing relationships between them.

Libraries Used

- Flask: a lightweight WSGI web application framework in Python.
- Neo4j: a graph database management system.

CRUD Operations

Create User:

Path: `/user`

Method: POST

Request Payload:**

```
{
   "username": "jack001 ",
   "email": "jack@test.com"
}
```

Create Post:

- Path: `/post`
- Method: POST
- Request Body:

```
{
   "user_id": 1,
   "title": "My First Post",
   "content": "This is the content of my post."
}
```

Update User:

- Path: `/user/{user_id}`
- Method: PUT
- Request Body:

```
{
   "new_username": "jack_02",
   "new_email": "jack002@test.com"
}
```

Update Post

- Path: `/post/{post_id}`
- Method: PUT
- Request Body:

```
{
   "new_title": "Updated Post Title",
   "new_content": "This is the updated content of my post."
}
```

Delete User:

- Path: `/user/{user_id}`
- Method: DELETE

Delete Post:

- Path: `/post/{post_id}`
- Method: DELETE

Get All Users:

- Path: `/users`
- Method: GET

Get User by ID:

- Path: `/user/{user_id}`
- Method: GET

Get All Posts:

- Path: `/posts`
- Method: GET

Get Post by ID:

- Path: `/post/{post_id}`

- Method: GET