🌐 Organization Management System

📚 Table of Contents

🚀 Introduction

The **Organization Management System** is an API built with **FastAPI** and **SQLAlchemy** that allows users to manage organizations, user memberships, and handle authentication with JWT tokens. This system provides a structured way to create and manage organizations, invite users, and maintain a database of users and their associations.

🛠 Prerequisites

To use this project, ensure you have the following installed:

- Python 3.7 or higher 🐍
- pip (Python package installer) 📦
- Git 👩‍💻

🏗 Installation

Clone the Repository

Clone the project repository to your local machine:

```
git clone https://github.com/tree-1917/interview-task
cd interview-task
```

Set Up a Virtual Environment

Create and activate a virtual environment:

```
python -m venv env
source env/bin/activate  # On Windows use `env\Scripts\activate`
```

Install Dependencies

Install the necessary dependencies:

```
pip install -r requirements.txt
```

⚙️ Usage

To run the FastAPI application, execute:

```
uvicorn app.main:app --reload
```

You can access the application at `http://localhost:8000` 🌐.

📡 API Endpoints

The following API endpoints are available:

- **GET** `/organization/`: Retrieve all organizations 📋.
- **POST** `/organization/`: Create a new organization ➕.
- **GET** `/organization/{organization_id}`: Retrieve details of a specific organization 🔍.
- **PUT** `/organization/{organization_id}`: Update details of a specific organization ✏️.
- **DELETE** `/organization/{organization_id}`: Delete an organization ❌.
- **POST** `/organization/{org_id}/invite`: Invite a user to join an organization 📧.

🗄️ Database Schema

The application uses SQLAlchemy for database interactions. Below is an overview of the database schema:

Users Table

- **id**: Integer (Primary Key) 🔑
- **username**: String 🧑
- **email**: String (Unique) 📧
- **password**: String 🔒

Organizations Table

- **id**: Integer (Primary Key) 🔑
- **name**: String 🏢
- **description**: String 📜
- **owner_id**: Integer (Foreign Key) 👤

Token Table

- **user_id**: Integer (Foreign Key) 🔑
- **access_token**: String (Primary Key) 🔑
- **refresh_token**: String 🔄

- **status**: Boolean ✅
- **created_date**: DateTime 📅

Organization Membership Table

- **user_id**: Integer (Foreign Key) 🔑
- **organization_id**: Integer (Foreign Key) 🔑
- **member_at**: DateTime ⏰

🖼️ Database Schema Diagram

```
erDiagram
    USERS {
        Integer id PK
        String username
        String email
        String password
    }

    ORGANIZATIONS {
        Integer id PK
        String name
        String description
        Integer owner_id FK
    }

    TOKEN_TABLE {
        Integer user_id FK
        String access_token PK
        String refresh_token
        Boolean status
        DateTime created_date
    }

    ORGANIZATION_MEMBERSHIP {
        Integer user_id FK
        Integer organization_id FK
        DateTime member_at
    }

    USERS ||--o{ ORGANIZATIONS : owns
    USERS ||--o{ ORGANIZATION_MEMBERSHIP : members
    ORGANIZATIONS ||--o{ ORGANIZATION_MEMBERSHIP : has
    USERS ||--o{ TOKEN_TABLE : uses
```

📝 License

This project is licensed under the MIT License - see the LICENSE file for details 📄.
