Project Management System API Documentation

Overview

This API allows users to manage projects, tasks, and comments. The main entities in this API are Users, Projects, Tasks, and Comments. Authentication is handled using JWT tokens.

Entities

1. **Users**
2. **Projects**
3. **Project_members**
4. **Tasks**
5. **Comments**

Setup

1. **Clone the Repository.**
2.
3. **Install Dependencies**

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

4. **Run Migrations**

   ```
   python manage.py migrate
   ```

5. **Create a Superuser**

   ```
   python manage.py createsuperuser
   ```

6. **Run the Server**

   ```
   python manage.py runserver
   ```

Endpoints

Users

- **Register User**

- **POST** `/api/users/register/`
- **Request Body:**

```
{
    "username": "new",
    "email": "new@gmail.com",
    "password": "123456789",
    "first_name": "New1",
    "last_name": "User1"
}
```

- **Response**: 201 Created

- **Login User**

- **POST** `/api/users/login/`
- **Request Body:**

```
{
    "email": "new@gmail.com",
    "password": "123456789"
}
```

- **Response:** 200 OK.

```
{
    "refresh": "your-refresh-token",
    "access": "your-access-token"
}
```

- **Get User Details**

- **GET** `/api/users/{id}/`
- **Headers:** `Authorization: Bearer your-access-token`
- **Response:** 200 OK

- **Update User**

- **PATCH** `/api/users/{id}/`
- **Headers:** `Authorization: Bearer your-access-token`
- **Request Body (example):**

```
{
    "first_name": "Updated",
    "last_name": "Name"
}
```

- Response: 200 OK

- **Delete User**

- **DELETE** `/api/users/{id}/`
- **Headers:** `Authorization: Bearer your-access-token`
- **Response:** 204 No Content

Projects

- **List Projects**
  - **GET** `/api/projects/`
  - **Headers:** `Authorization: Bearer your-access-token`
  - **Response**: 200 OK

- **Create Project**
  - **POST** `/api/projects/`
  - **Headers:** `Authorization: Bearer your-access-token`
  - **Request Body**:

```
{
    "name": "New Project",
    "description": "This is a new project",
    "owner": 1,  // Use the ID of the logged-in user
    "created_at": "2024-06-24T00:00:00Z"
}
```

  - Response: 201 Created

- **Retrieve Project**
  - **GET** `/api/projects/{id}/`
  - **Headers**: `Authorization: Bearer your-access-token`
  - **Response**: 200 OK

- **Update Project**
  - **PUT/PATCH** `/api/projects/{id}/`
  - **Headers:** `Authorization: Bearer your-access-token`
  - **Request Body (example):**

```
{
    "description": "Updated description"
}
```

  - Response: 200 OK

- **Delete Project**
  - **DELETE** `/api/projects/{id}/`
  - **Headers:** `Authorization: Bearer your-access-token`
  - **Response:** 204 No Content

Tasks

- **List Tasks in a Project**
  - **GET** `/api/projects/{project_id}/tasks/`
  - Headers: `Authorization: Bearer your-access-token`
  - Response: 200 OK

- **Create Task in a Project**
  - **POST** `/api/projects/{project_id}/tasks/`
  - **Headers**: `Authorization: Bearer your-access-token`
  - **Request Body:**

```
{
   "title": "New Task",
    "description": "This is a new task",
```

```
    "status": "To Do",
    "priority": "Low",
    "assigned_to": 1,  // Use the ID of the logged-
user
    "due_date": "2024-07-01T00:00:00Z"
}
```

- Response: 201 Created

- **Retrieve Task**
  - **GET** `/api/tasks/{id}/`
  - **Headers**: `Authorization: Bearer your-access-token`
  - **Response:** 200 OK

- **Update Task**
  - **PUT/PATCH** `/api/tasks/{id}/`
  - Headers: `Authorization: Bearer your-access-token`
  - **Request Body (example):**

```
{
    "status": "In Progress"
}
```

  - Response: 200 OK

- **Delete Task**
  - **DELETE** `/api/tasks/{id}/`
  - **Headers:** `Authorization: Bearer your-access-token`
  - **Response:** 204 No Content

Comments

- **List Comments on a Task**
  - **GET** `/api/tasks/{task_id}/comments/`
  - **Headers**: `Authorization: Bearer your-access-token`
  - **Response:** 200 OK

- **Create Comment on a Task**
  - **POST** `/api/tasks/{task_id}/comments/`
  - **Headers:** `Authorization: Bearer your-access-token`
  - **Request Body:**

```
{
    "content": "This is a comment on the task.",
    "user": 1  // Use the ID of the logged-in user
}
```

  - **Response:** 201 Created

Example Workflow in Postman

1. **Register a User**
   - Create a new request in Postman.
   - Set the method to `POST` and the URL to `http://localhost:8000/api/users/register/`.
   - Set the body to raw JSON:

```
{
    "username": "new",
    "email": "new@gmail.com",
    "password": "123456789",
    "first_name": "New1",
    "last_name": "User"
}
```

   - Click `Send`.

2. **Login User**
   - Create a new request in Postman.
   - Set the method to `POST` and the URL to `http://localhost:8000/api/users/login/`.
   - Set the body to raw JSON:

```
{
    "email": "new@gmail.com",
    "password": "123456789"
}
```

- Click `Send`. (Copy the `access` token from the response.)

3. **Create a Project**
   - Create a new request in Postman.
   - Set the method to `POST` and the URL to `http://localhost:8000/api/projects/`.
   - Go to the **`Headers`** tab and add `Authorization: Bearer your-access-token`.
   - Set the body to raw JSON:

```
{
    "name": "New Project",
    "description": "This is a new project",
    "owner": 1,  // Use the ID of the logged-in user
    "created_at": "2024-06-24T00:00:00Z"
}
```

   - Click `Send`.

4. **Create a Task in a Project**
   - Create a new request in Postman.
   - Set the method to `POST` and the URL to `http://localhost:8000/api/projects/1/tasks/`.
   - Go to the `Headers` tab and add `Authorization: Bearer your-access-token`.
   - Set the body to raw JSON:

```
{
    "title": "New Task",
    "description": "This is a new task",
    "status": "To Do",
    "priority": "Low",
    "assigned_to": 1,  // Use the ID of the logged-in
user
    "due_date": "2024-07-01T00:00:00Z"
}
```

   - Click `Send`.

5. **List Comments on a Task**
   - Create a new request in Postman.

- Set the method to `GET` and the URL to `http://localhost:8000/api/tasks/1/comments/`.
- Go to the **`Headers`** tab and add `Authorization: Bearer your-access-token`.
- Click `Send`.

6. **Create Comment on a Task**
   - Create a new request in Postman.
   - Set the method to `POST` and the URL to `http://localhost:8000/api/tasks/1/comments/`.
   - Go to the **`Headers`** tab and add `Authorization: Bearer your-access-token`.
   - Set the body to raw JSON:

```
{
    "content": "This is a comment on the task.",
    "user": 1  // Use the ID of the logged-in user
}
```

   - Click `Send`.

Conclusion

You should be able to use JWT authentication in your Django application to manage users, projects, tasks, and comments by following this documentation. Test the endpoints with Postman to make sure everything is operating as it should.