Social Media API

.Description :

This API facilitates core social media platform features such as user authentication, profile management, and post creation and retrieval. Users can follow others, view personalized post feeds, and manage their profiles securely.

.Core features and functionality :

1. Authentication System:

Handled in the `auth` app

Features:

Registration: Users can register with a valid email, username, and password.
Login: Validates user credentials and returns an authentication token.
Logout: Invalidates the user's token to log them out securely.
Token Management: Generates and manages tokens using Django's token authentication

2. User Profile Management :

Handled in the `API` package under the `users` endpoint

Features:

- **Update Profile**: Users can modify their profile details (e.g., name, email, bio).
- **Delete Account**: Users can delete their account permanently.
- **Follow/Unfollow Users**: Allows users to follow/unfollow others, establishing connections between profiles.
- **Followers and Following Management**: Tracks the number of followers and following for each user.

3. Posts Management

Handled in the `posts` app and exposed through the `API` package

Features:

- **Create Posts:** Users can create posts with text and optional media (images, videos).
- **Read Posts:** Retrieve individual posts or all posts by a specific user.
- **Update Posts:** Modify the content of existing posts.
- **Delete Posts:** Remove posts from the platform.

4. Feed Feature

Custom action as part of the `posts` view in the API

Features:

- **Retrieve Personalized Feed**: Fetch posts created by users the logged-in user is following.
- **Pagination**: Supports paginated responses to manage large datasets efficiently.

. API endpoints to implement :

Authentication endpoints:

```
POST /auth/register/
POST /auth/login/
POST /auth/logout/
```

Users endpoints:

`PUT /api/users/<username>/` (Update Profile)

`DELETE /api/users/<username>/` (Delete Account)

`POST /api/users/<username>/follow/` (Follow a user)

`POST /api/users/<username>/unfollow/` (Unfollow a user)

Posts endpoints:

`POST /api/posts/` (Create Post)

`GET /api/posts/<title>/` (Retrieve a specific post)

`GET /api/posts/` (Retrieve all posts by a user)

`PUT /api/posts/<title>/` (Update Post)

`DELETE /api/posts/<title>/` (Delete Post)

Feed Posts endpoints:

`GET /api/posts/feed/` (Fetch personalized feed)

.Tools and libraries you plan to use :

Framework: Django & Django REST Framework (DRF)

Database: MySQL

Data Model Design :

Link

In the social media platform, users are connected through a **many-to-many relationship** facilitated by the following system. Each user can follow multiple users and, in turn, be followed by others. This relationship is managed via a **UserFollow** table that links a user (`follower_id`) to another user they follow (`followed_id`). For example, if **User A** follows **User B** and **User C**, the table records these relationships, and if **User D** follows **User A**, it tracks this as well. Similarly, there is a **one-to-many relationship** between users and posts, where each user can create multiple posts. For instance, **User A** can create posts like "Post 1" and "Post 2", while **User B** can create their own posts. Together, these relationships allow users to build networks and interact with content shared by the people they follow.