**TASK 1:**

**Video Streaming Application**

**Overview**

This Django-based video streaming application allows users to manage video content, perform CRUD operations on videos, and access video streaming functionalities. The application provides a RESTful API for interacting with video resources.

**Usage**

**1. Project Setup**

Assuming you have Python and Django installed, follow these steps to set up the project:

# Create a new Django project

django-admin startproject myproject

cd myproject

# Create a new Django app named 'videos'

python manage.py startapp videos

**2. Code Overview**

The provided code implements a Django app named **videos** that includes:

* Models (**Video** and **Profile**) representing video uploads and user profiles.
* Serializers for converting model instances to JSON data and vice versa.
* API views (**VideoListCreateAPIView**, **VideoRetrieveUpdateDestroyAPIView**, **ProfileRetrieveAPIView**) using Django REST Framework for CRUD operations on videos and profile retrieval.
* URL configuration (**urls.py**) for routing API endpoints.

**3. Dependencies Installation**

Ensure you have Django and Django REST Framework installed:

pip install django djangorestframework

**4. Database Migration**

Set up the database and apply migrations:

python manage.py migrate

**5. Running the Development Server**

Start the Django development server:

python manage.py runserver

**6. Accessing the APIs**

Once the development server is running, you can access the following endpoints:

* **Video List/Create API**: **http://localhost:8000/api/videos/**
  + **GET**: Retrieve a list of videos.
  + **POST**: Create a new video (authentication required).
* **Video Retrieve/Update/Delete API**: **http://localhost:8000/api/videos/<video\_id>/**
  + **GET**: Retrieve details of a specific video.
  + **PUT**: Update details of a specific video.
  + **DELETE**: Delete a specific video (authentication required).
* **Profile Retrieve API**: **http://localhost:8000/api/profile/**
  + **GET**: Retrieve the profile of the authenticated user.

**Additional Notes**

* Ensure that you have a valid Django **User** model instance for authentication.
* Use tools like **curl** or a REST client (e.g., Postman) to interact with the APIs.
* Customize the authentication and permissions based on your project's requirements.
* Update the **settings.py** file with necessary configurations (e.g., **INSTALLED\_APPS**, **REST\_FRAMEWORK** settings) as per your project needs.

By following these steps and accessing the provided API endpoints, you should be able to interact with the Django application implementing video uploads and user profiles.