# **Project Overview**

This project uses the **YouTube Data API** to fetch the latest videos for a specific search query (like "cricket") and stores them in a **MySQL** database. The project also includes a **FastAPI-based REST API** to access the saved videos. Optionally, you can connect it with a basic HTML dashboard to view the videos.

### **How It Works**

### 1. Background Video Fetcher

- Periodically connects to the YouTube API using your API key.
- Retrieves new videos based on the configured search term.
- Filters out duplicates and saves the latest videos into the database.

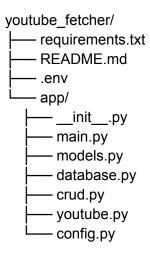
## 2. REST API (FastAPI)

- Exposes an endpoint (/videos) to get a list of videos.
- Supports filtering, sorting, and pagination.

#### 3. HTML Dashboard

A simple HTML/CSS frontend that consumes the API and displays the video list.

## **Folder Structure**



# **File Descriptions**

#### requirements.txt

List of required Python libraries:

• fastapi, uvicorn, sqlalchemy, aiohttp, python-dotenv, pydantic, pymysql

#### .env

Keeps all sensitive configuration data:

```
YOUTUBE_API_KEYS=your_api_key_here

SEARCH_QUERY=cricket

FETCH_INTERVAL=10

DATABASE_URL=mysql+pymysql://root:password@localhost:3306/youtube_db
```

#### app/config.py

Loads settings from . env and makes them available to the app.

### app/database.py

Sets up the MySQL database connection using SQLAlchemy.

### app/models.py

Defines a Video table schema with fields like video\_id, title, description, etc.

### app/crud.py

Contains helper functions for:

- Adding a new video
- Getting a video by ID
- Listing videos with pagination

### app/youtube.py

Implements logic to:

- Connect to YouTube API
- Rotate API keys if limit is reached

Save new videos to DB

### app/main.py

Main FastAPI app:

- Sets up the /videos API endpoint
- Initializes database
- Starts the background fetcher

# **Step-by-Step Usage**

1. Install dependencies

pip install -r requirements.txt

- 2. **Configure environment variables** Update the .env file with your API key and DB details.
- 3. Create the database

CREATE DATABASE youtube\_db;

4. Run the FastAPI server

uvicorn app.main:app --reload

5. (Optional) Open the dashboard Open your HTML dashboard (if available) in a browser.

# **Customization Options**

- Change the search term in .env to fetch different topics.
- Adjust FETCH\_INTERVAL for frequency control.
- Add more endpoints or filters in the FastAPI app.
- Build a more advanced frontend using React or Vue.

# **Summary**

This project provides a simple and clean backend system for fetching, storing, and displaying YouTube videos. It's ideal for learning how to work with APIs, databases, and RESTful services using FastAPI.