API Design to Upload the Files [PDF]

Project Overview:

The goal of the task is to create an API endpoint that enable the user to upload the PDF file upto 100 MB size. The PDF file that is uploaded by the user at the endpoint should be download at my project directory in my local machine. Once the file get download, the metadata of the PDF file should be sent to Postgresql database.

Level:

Easy to Medium

Type of Project:

API Development, API endpoint

Skills Required:

- Python
- API Development & design
- FastAPI Framework
- PostgreSQL
- Rate-Limiting for API performance
- Async and Sync

Key Features

Feature 1: Design for docker-compose.yml: [Executed]

- I am using Windows operating system
- Use this image postgres:14-trixie
- Use this image for pgadmin dpage/pgadmin4:snapshot
- Make sure that you include Volume to make data persistent if container restart or delete in future
- Use docker bind mount. Mount a directory in the Project directory

Feature 2: Pydantic models

- There exist file api_models.py at src/models/api_models.py
- This file should contain pydantic models API response

Feature 2: API Design for Performance:

- Implement Leaky Bucket Rate Limiting to manage the number of request at the API endpoint.
- Set the capacity of the bucket as 2 and the leak rate as 0.016 req/sec

- If the user request exceed the capacity of the bucket then give the error “Server is Busy” with appropriate HTTP status code for it.

Feature 3: Upload File upto a limited size

- The PDF File to be upload by the user at the endpoint should be maximum size 100 MB.
- If the PDF file exceed 100MB, Give error “Cannot be Upload! Max File size is 100 MB”
- The endpoint must accept only PDF file only.
- Don’t use UploadFile class for file upload.
- Use request.stream()
- The following Metadata should be fetch. Note: User will not give these metadata explicitly
  - Unique id
  - File name
  - Storage Path
  - Uploaded at (time and date)
- These Metadata should be save at: “C:\Users\Admin\Desktop\rag-hootone\data\uploads\Metadata” for each file uploaded by the User.
- The file for Metadata should be name same as ‘File name’ and file format should be .txt
- Include mechanism to check that the same files is not repeated. Check this by there “File name” to avoid duplicacy. Make sure File name is Case-sensitive
- If the same file already exist then give this error “File already Exist!”.
- The PDF file should be download at “C:\Users\Admin\Desktop\rag-hootone\data\uploads\PDF”

Feature 4: Storing MetaData to PostgreSQL

- This feature is a part of the every request that leaks from the bucket in Leaky bucket algorithm for processing.
- The Metadata stored at a aprticular directory in the project directory
- There exist a table named as ‘Metadata’ in PostgreSQL
- Fetch Metadata from the given location and store it to PostgreSQL
- There exist a table named as ‘Metadata’. Don’t create any table explicitly
- This is the schema of the table

column_name	data_type	is_nullable	column_default
unique_id	uuid	NO	gen_random_uuid()
uploaded_at	timestamp without time zone	NO	CURRENT_TIMESTAMP
file_name	character varying	NO	NULL
storage_path	text	NO	NULL

Feature 4: code practice

- **Follow Modular Design**
- **Use logging module and file to maintain logs**