-- MongoDB Project Documentation --

YouTube Manager App

Author: Prathamesh PC

This document provides an overview of a simple YouTube video manager application using MongoDB and Python. The purpose of this project is to demonstrate my knowledge of MongoDB, including setup, connection, and CRUD operations. The project is implemented using MongoDB Atlas, the cloud-based service for MongoDB.

MongoDB Setup

```
from pymongo import MongoClient
client =
MongoClient("mongodb+srv://<username>:<password>@cluster0.wntvop7.mongodb.net/?retry
Writes=true&w=majority&appName=Cluster0", tlsAllowInvalidCertificates=True)
db = client["ytmanager"]
video_collection = db["videos"]
```

Description

MongoClient: Establishes a connection to the MongoDB cluster.
db: Accesses the ytmanager database.
video_collection: Accesses the videos collection within the database.

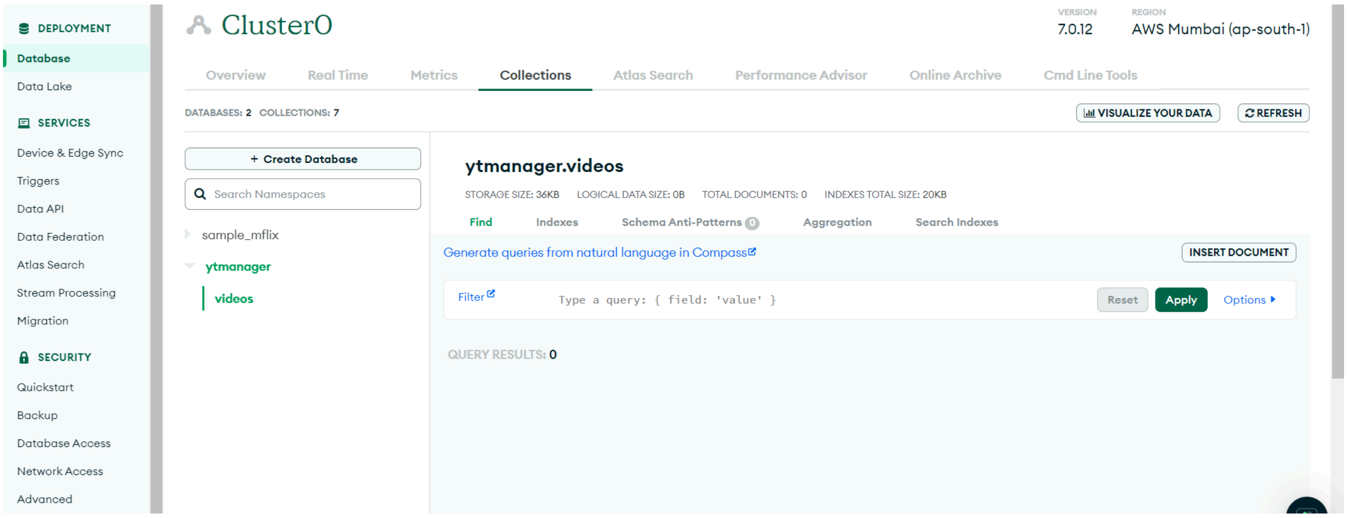
Project Code

Here is the complete code for the YouTube Manager App (or refer youtube_manager_mongodb_py1.py file):

```
from pymongo import MongoClient
from bson import ObjectId

client =
MongoClient("mongodb+srv://<username>:<password>@cluster0.wntvop7.mongodb.net/?retryWrites=tru
e&w=majority&appName=Cluster0", tlsAllowInvalidCertificates=True)
db = client["ytmanager"]
video_collection = db["videos"]

def add_video(name, time):
    video_collection.insert_one({"name": name, "time": time})

def list_videos():
    for video in video_collection.find():
        print(f"ID: {video['_id']}, Name: {video['name']} and Time: {video['time']}")

def update_video(video_id, new_name, new_time):
    video_collection.update_one({'_id': ObjectId(video_id)}, {"$set": {"name": new_name, "time":
new_time}})

def delete_video(video_id):
    video_collection.delete_one({"_id": ObjectId(video_id)})
```

```
def main():
    while True:
        print("\n YouTube Manager App")
        print("1. List all videos")
        print("2. Add a new video")
        print("3. Update a video")
        print("4. Delete a video")
        print("5. Exit the app")
        choice = input("Enter your choice: ")

        if choice == '1':
            list_videos()
        elif choice == '2':
            name = input("Enter the video name: ")
            time = input("Enter the video time: ")
            add_video(name, time)
        elif choice == '3':
            video_id = input("Enter the video id to update: ")
            name = input("Enter the updated video name: ")
            time = input("Enter the updated video time: ")
            update_video(video_id, name, time)
        elif choice == '4':
            video_id = input("Enter the video id to delete: ")
            delete_video(video_id)
        elif choice == '5':
            break
        else:
            print("Invalid choice")

if __name__ == "__main__":
    main()
```

Below is a screenshot of the MongoDB Atlas panel showing the ytmanager database and the videos collection:



Cluster0

VERSION 7.0.12

REGION AWS Mumbai (ap-south-1)

DATABASES: 2 COLLECTIONS: 7

ytmanager.videos

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 0B TOTAL DOCUMENTS: 0 INDEXES TOTAL SIZE: 20KB

Generate queries from natural language in Compass

Type a query: { field: 'value' }

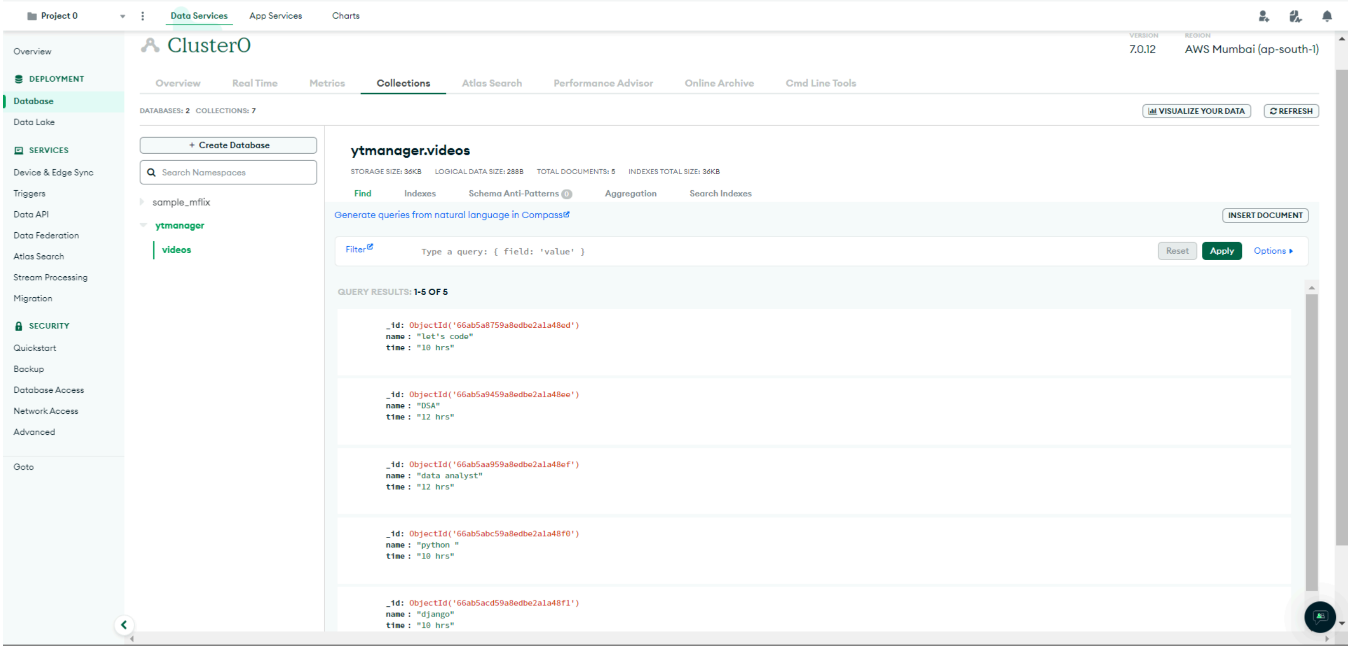
QUERY RESULTS: 0

Code Output

Here is an example output from running the YouTube Manager App (or refer output.txt and youtube_manager_mongodb_py 1.ipynb file):

```
YouTube Manager App
1. List all videos
2. Add a new video
3. Update a video
4. Delete a video
5. Exit the app
Enter your choice: 1
ID: 66ab5a8759a8edbe2a1a48ed, Name: "let's code", Time: "10 hrs"
ID: 66ab5a9459a8edbe2a1a48ee, Name: "DSA", Time: "12 hrs"
```

Below is a screenshot of the updated videos collection after running the project:



```
_id: ObjectId('66ab5a8759a8edbe2a1a48ed')
name : "let's code"
time : "10 hrs"

_id: ObjectId('66ab5a9459a8edbe2a1a48ee')
name : "DSA"
time : "12 hrs"

_id: ObjectId('66ab5aa959a8edbe2a1a48ef')
name : "data analyst"
time : "12 hrs"

_id: ObjectId('66ab5abc59a8edbe2a1a48f0')
name : "python "
time : "10 hrs"

_id: ObjectId('66ab5acd59a8edbe2a1a48f1')
name : "django"
time : "10 hrs"
```

This project demonstrates the basic CRUD operations in MongoDB using Python, showcasing my understanding of connecting to a MongoDB database, performing operations, and managing data collections. The project is implemented using MongoDB Atlas, highlighting the use of cloud-based MongoDB services.