CS 306 - Database Systems
Project Phase 4 Report
Due 31.05.2024

Student Names and IDs:
İpek Akkuş - 30800
Semih Doğan - 32560

1. Introduction

This project involves the implementation of a playlist service that enables users to manage their own music collections through a MongoDB Atlas database. Users can create collections for different bands, specifically Opeth and Soen, and store detailed information about their songs. Each song in the collections can have properties such as name, album name, insert date, duration, view count, release year, and rating. By creating their own databases using their user ID, users can perform several operations, including creating collections, reading all data, filtering data, inserting new songs, deleting songs, updating song details, and exiting the program. This service provides a user-friendly interface for music enthusiasts to organize and manage their favorite tracks efficiently. It aims to enhance the user's ability to keep track of their own music, rate songs, and maintain detailed records of their listening habits, offering a personalized and convenient experience.

2. System Description

The system offers several features and functionalities as explained below.

- **User-Specific Databases:** Each user can create their own database using their username, ensuring personalized and secure management of their music collections.
- **Collection Management**: Database may contain several collections, in the scope of this project, it is limited to 2, which are for Opeth and Soen

songs. User may do these with the collections that they created or already created.

- **Create a Collection:** Users can create new collections within their database, such as separate collections for Opeth and Soen songs.
- **Read All Data in a Collection:** Users can retrieve and display all the songs stored in a particular collection, allowing them to view their entire music library.
- **Read Data with Filtering:** Users can filter and retrieve songs based on specific attributes such as name, album name, rating, or release year, helping them find specific songs quickly.

- **Data Manipulation:** Users may contain their data, by performing several operations on their own collections. Our project offers these functionalities:
    - **Insert Data:** Users can add new songs to their collections by providing details such as the song name, album name, insert date, duration, view count, release year, and rating.
    - **Delete Data:** Users can delete songs from their collections by specifying the song name, ensuring their music library remains current and relevant.
    - **Update Data:** Users can update specific attributes of a song, such as the view count or rating, to reflect their latest listening habits and preferences.
- **User-Friendly Interface:** The service provides a menu-driven interface that guides users through the various operations they can perform, prompting them for necessary inputs clearly and providing clear feedback on the actions taken.
- **Error Handling:** The service includes error handling mechanisms to manage exceptions and provide meaningful error messages to users, ensuring a smooth and reliable user experience.

By integrating these functionalities, the playlist service offers a comprehensive and efficient tool for music enthusiasts to manage and enjoy their music collections.

3. Database Design

- **Reason for selecting NoSQL database (MongoDB Atlas)**

MongoDB Atlas was chosen for its flexibility, scalability, and ease of use. As a NoSQL database, it stores data in a flexible, JSON-like format, ideal for diverse song metadata. Being a fully-managed cloud service, MongoDB Atlas offers automated backups, robust security, and scalability, ensuring reliable performance and easy management. This allows users to handle large volumes of data efficiently without managing infrastructure.

- **Two created collections, purposes and structures included**

For this project, two collections have been created: one for Opeth songs and one for Soen songs. Each collection is designed to store detailed information about the songs, allowing users to organize and manage their music library effectively.

a. Opeth Songs Collection:

It is created to store detailed information about songs by the well-known band Opeth. Each document in this collection represents a song and includes the following properties:

name: The name of the song.

album_name: The name of the album the song belongs to.

insert_date: The date the song was added to the playlist.

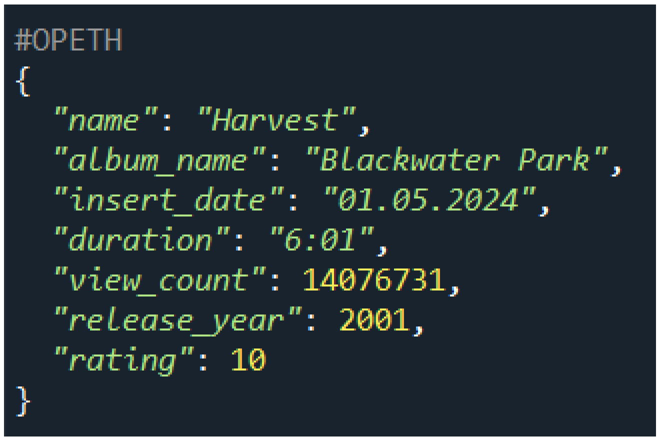
duration: The length of the song.

view_count: The number of times the song has been played.

release_year: The year the song was released.

rating: The user's rating for the song.

Figure 1 demonstrates a sample for Opeth Playlist collection:



```
#OPETH
{
  "name": "Harvest",
  "album_name": "Blackwater Park",
  "insert_date": "01.05.2024",
  "duration": "6:01",
  "view_count": 14076731,
  "release_year": 2001,
  "rating": 10
}
```

Figure 1. Sample Opeth Playlist entry

b. Soen Songs Collection:

It is created to store detailed information about songs by the well-known band Soen. Each document in this collection represents a song and includes the same properties as the Opeth collection:

name: The name of the song.

album_name: The name of the album the song belongs to.

insert_date: The date the song was added to the playlist.

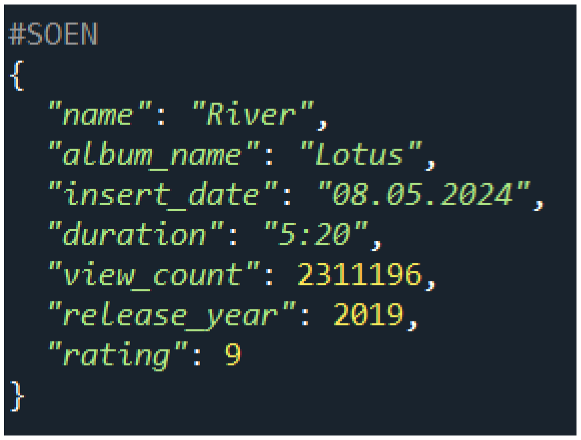
duration: The length of the song.

view_count: The number of times the song has been played.

release_year: The year the song was released.

rating: The user's rating for the song.

Figure 2 demonstrates a sample for Soen Playlist collection:



```
#SOEN
{
  "name": "River",
  "album_name": "Lotus",
  "insert_date": "08.05.2024",
  "duration": "5:20",
  "view_count": 2311196,
  "release_year": 2019,
  "rating": 9
}
```

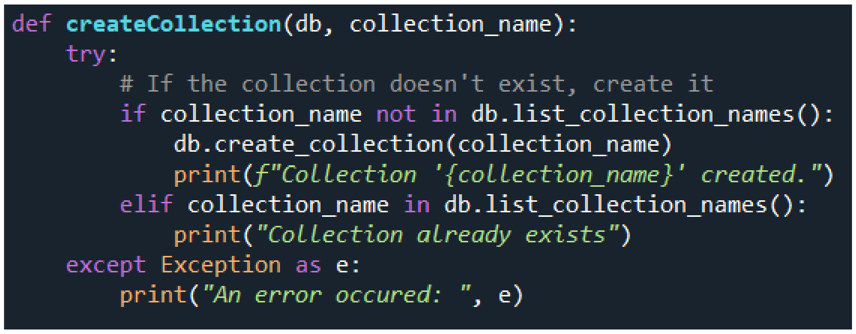
Figure 2. Sample SoenPlaylist entry

4. Implementation

The implementation of the playlist service involves connecting to a MongoDB Atlas database, creating and managing collections, and performing CRUD (Create, Read, Update, Delete) operations. The project is structured to provide a menu-driven interface for users to interact with their music collections. Below are descriptions of the functions created for each CRUD operation along with code snippets and explanations.

Functions:

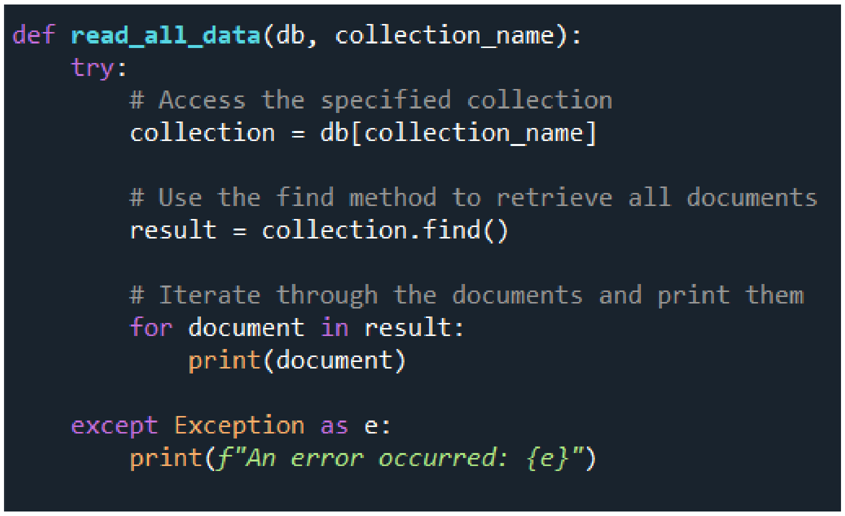
1. **Create a Collection:** The createCollection function first checks if the collection name exists in the database. If the collection does not exist, it creates a new collection and confirms its creation to the user. Otherwise, it notifies the user that the collection already exists. This function ensures that users can organize their songs under distinct collections, which helps in maintaining a structured and organized database. Figure 3 shows the implementation of createCollection function.



```python
def createCollection(db, collection_name):
    try:
        # If the collection doesn't exist, create it
        if collection_name not in db.list_collection_names():
            db.create_collection(collection_name)
            print(f"Collection '{collection_name}' created.")
        elif collection_name in db.list_collection_names():
            print("Collection already exists")
    except Exception as e:
        print("An error occured: ", e)
```

Figure 3. Implementation of createCollection function

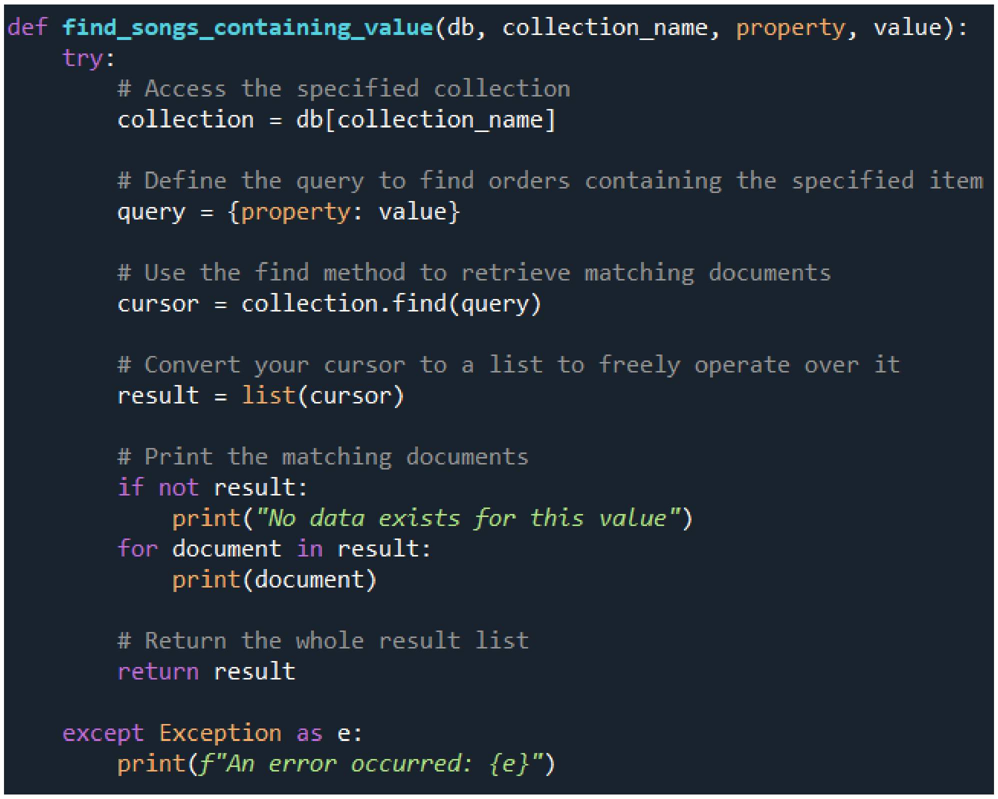
2. **Read All Data in a Collection:** The read_all_data function retrieves and displays all documents in a specified collection. This function accesses the specified collection and retrieves all documents using the find method. It then iterates through the retrieved documents and prints each one, providing a comprehensive view of the collection's contents. Figure 4 shows the implementation of the read_all_data function.



```python
def read_all_data(db, collection_name):
    try:
        # Access the specified collection
        collection = db[collection_name]

        # Use the find method to retrieve all documents
        result = collection.find()

        # Iterate through the documents and print them
        for document in result:
            print(document)

    except Exception as e:
        print(f"An error occurred: {e}")
```

Figure 4. Implementation of read_all_data function

3. **Read Data with Filtering:** The find_songs_containing_value function retrieves documents that match a specific property value. This function constructs a query based on a given property and value, retrieves matching documents, and prints them. It converts the cursor to a list and checks if any documents were found, providing appropriate feedback to the user. Figure 5 shows the implementation of the find_songs_containing_value function.



```python
def find_songs_containing_value(db, collection_name, property, value):
    try:
        # Access the specified collection
        collection = db[collection_name]

        # Define the query to find orders containing the specified item
        query = {property: value}

        # Use the find method to retrieve matching documents
        cursor = collection.find(query)

        # Convert your cursor to a list to freely operate over it
        result = list(cursor)

        # Print the matching documents
        if not result:
            print("No data exists for this value")
        for document in result:
            print(document)

        # Return the whole result list
        return result

    except Exception as e:
        print(f"An error occurred: {e}")
```

Figure 5. Implementation of find_songs_containing_value function

4. **Insert Data:** The insert_into_collection function inserts a new document into a specified collection. This function inserts a new document into the specified collection and confirms the insertion by printing the inserted document's ID. It ensures that new songs can be added to the database with all necessary details. Figure 6 shows the implementation of the insert_into_collection function.

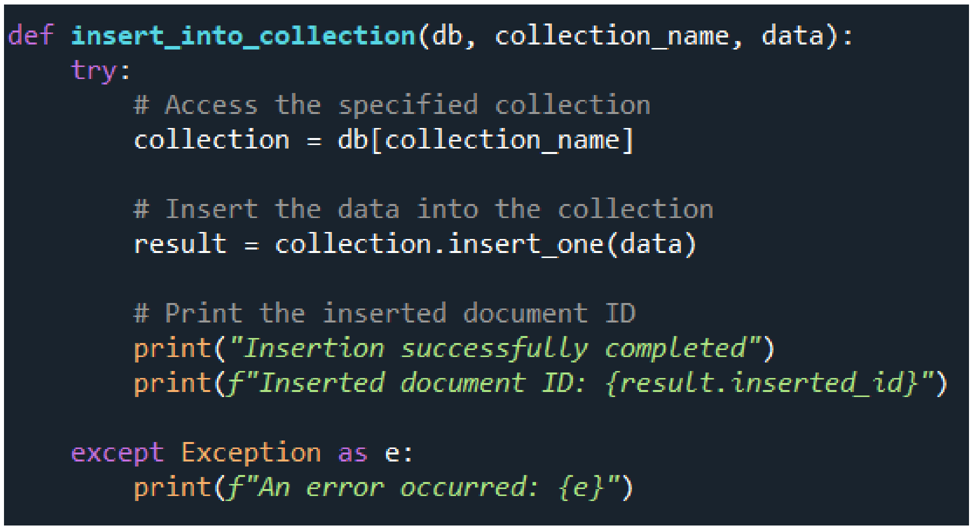

```python
def insert_into_collection(db, collection_name, data):
    try:
        # Access the specified collection
        collection = db[collection_name]

        # Insert the data into the collection
        result = collection.insert_one(data)

        # Print the inserted document ID
        print("Insertion successfully completed")
        print(f"Inserted document ID: {result.inserted_id}")

    except Exception as e:
        print(f"An error occurred: {e}")
```

Figure 6. Implementation of insert_into_collection function

5. **Delete Data:** The delete_record_by_name function deletes a document based on the song name. This function deletes a document from the collection based on the song name and confirms the deletion. It ensures that users can keep their collections up-to-date by removing unwanted songs. Figure 7 shows the implementation of the delete_record_by_name function.

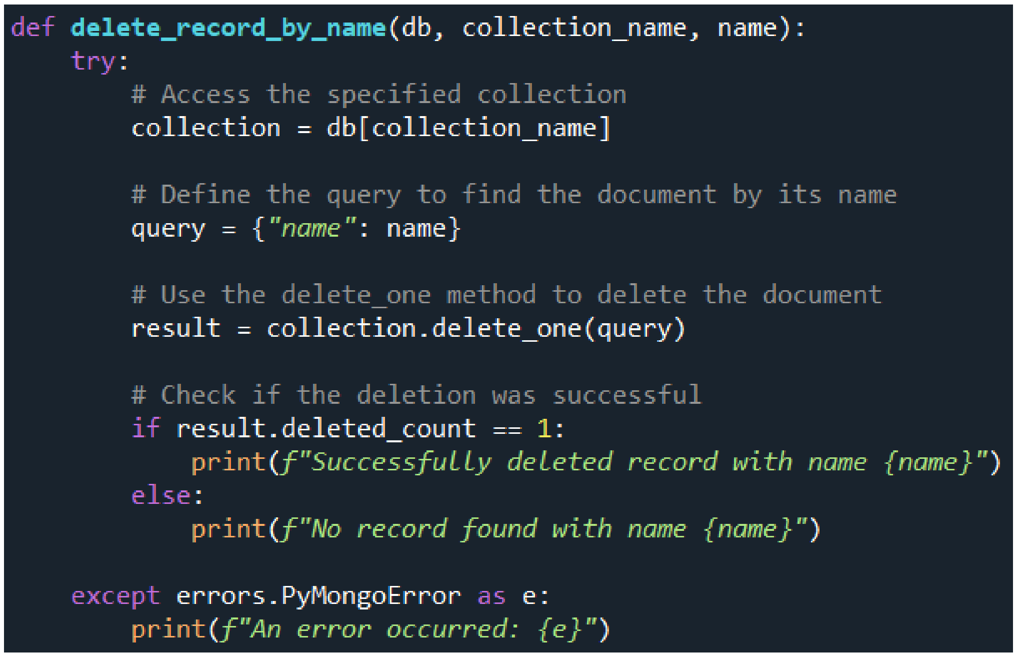
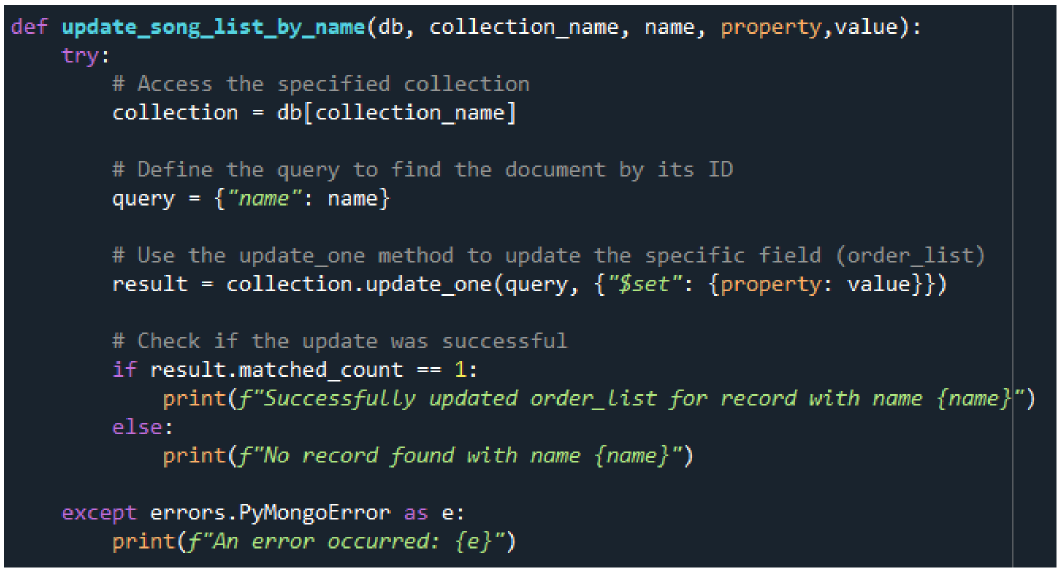

```python
def delete_record_by_name(db, collection_name, name):
    try:
        # Access the specified collection
        collection = db[collection_name]

        # Define the query to find the document by its name
        query = {"name": name}

        # Use the delete_one method to delete the document
        result = collection.delete_one(query)

        # Check if the deletion was successful
        if result.deleted_count == 1:
            print(f"Successfully deleted record with name {name}")
        else:
            print(f"No record found with name {name}")

    except errors.PyMongoError as e:
        print(f"An error occurred: {e}")
```

Figure 7. Implementation of delete_record_by_name function

6. **Update Data:** The update_order_list_by_name function updates a specific property of a document, identified by the song name. This function updates a specified property of a document, identified by the song name, and confirms the update. It allows users to keep their song information current and reflective of their latest listening habits and preferences. Figure 8 shows the implementation of the update_order_list_by_name function.



```python
def update_song_list_by_name(db, collection_name, name, property,value):
    try:
        # Access the specified collection
        collection = db[collection_name]

        # Define the query to find the document by its ID
        query = {"name": name}

        # Use the update_one method to update the specific field (order_list)
        result = collection.update_one(query, {"$set": {property: value}})

        # Check if the update was successful
        if result.matched_count == 1:
            print(f"Successfully updated order_list for record with name {name}")
        else:
            print(f"No record found with name {name}")

    except errors.PyMongoError as e:
        print(f"An error occurred: {e}")
```

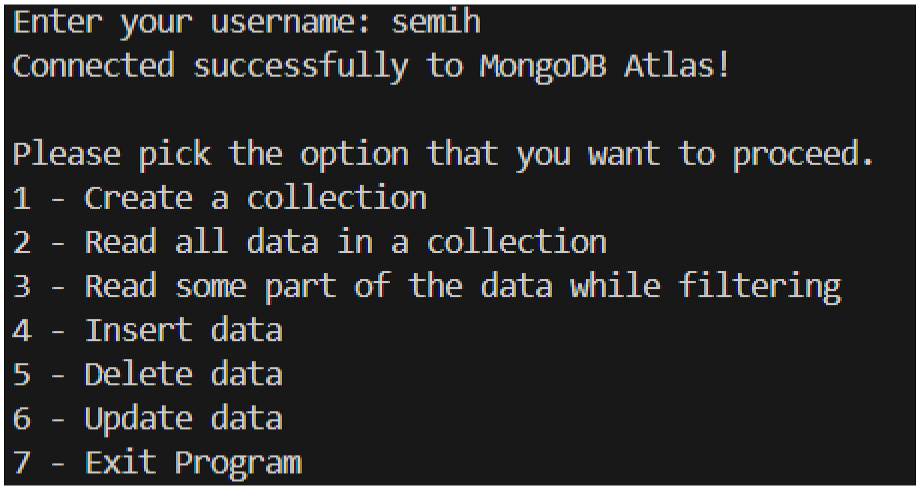
Figure 8. Implementation of update_order_list_by_name function

These code snippets and explanations illustrate the core functionalities of the playlist service, ensuring that users can effectively manage their music collections.

5. User Interaction

The playlist service features a menu-driven interface that allows users to interact with the system in a straightforward manner. Upon starting the program, users are prompted to enter their username, which is used to connect to their specific database. The menu then presents a series of options, each corresponding to a different CRUD operation. Users can select an option by entering the corresponding number. The system guides users through the process by prompting them for any necessary inputs, such as collection names or song details, and provides feedback on the success of each operation. Here are several example interactions with the system:

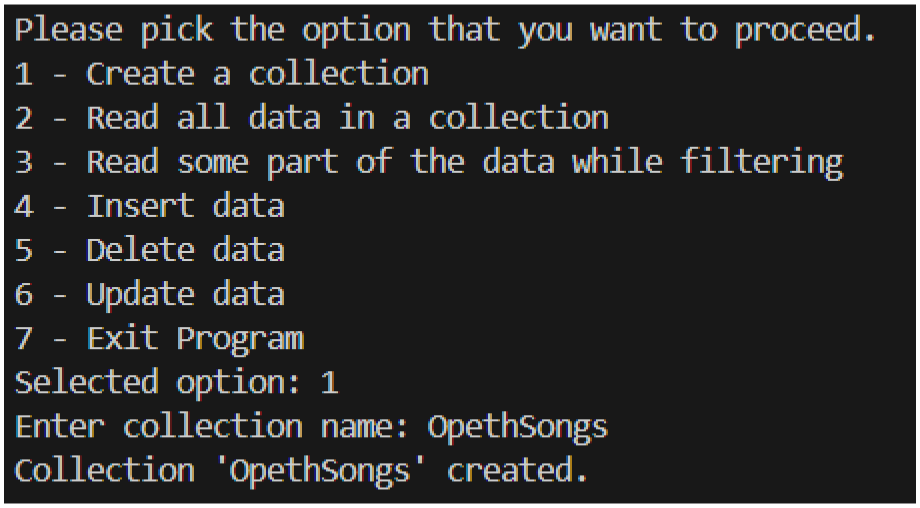
a. **Starting the Program and Connecting to the Database:** The program starts with asking username to user and showing a message indicating that it could connect to the MongoDB Atlas database.



```
Enter your username: semih
Connected successfully to MongoDB Atlas!

Please pick the option that you want to proceed.
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
```

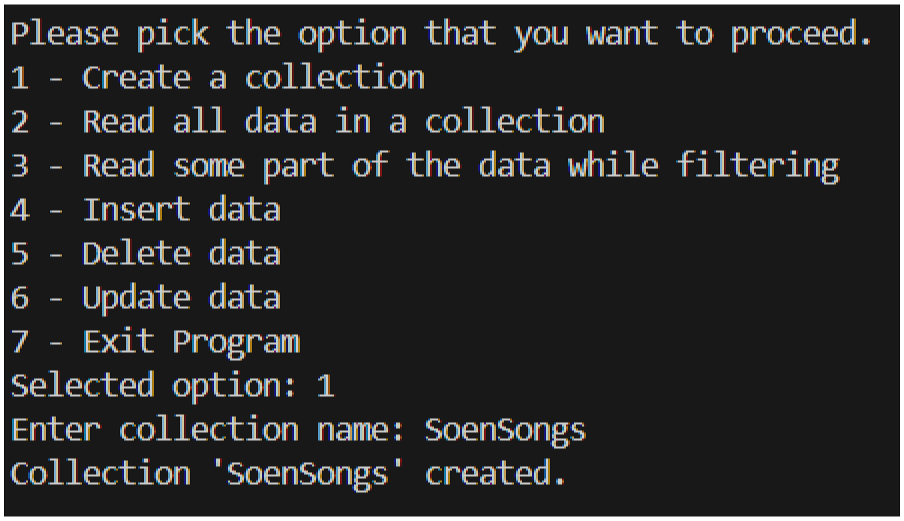
Figure 9. Starting the program and connecting to the database

b. **Creating a Collection:** In the scope of this project, users can create at most two collections: OpethSongs and SoenSongs. The createCollection function checks if the collection exists and creates it if it doesn't. Figure 10 shows the creation of the OpethSongs collection, while Figure 11 shows the creation of the SoenSongs collection. In these features, it should be noted that the user is remain as it is selected in the beginning.



```
Please pick the option that you want to proceed.
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 1
Enter collection name: OpethSongs
Collection 'OpethSongs' created.
```

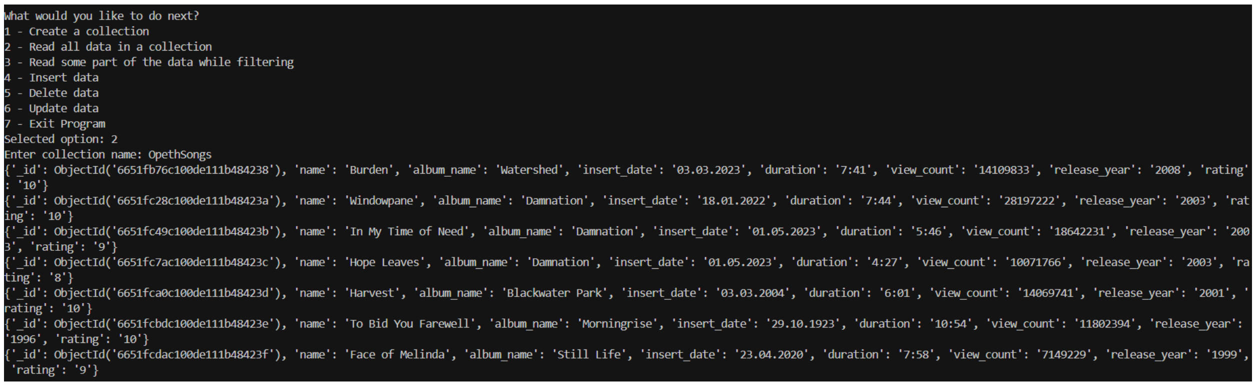
Figure 10. Creating OpethSongs collection for indicated user



```
Please pick the option that you want to proceed.
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 1
Enter collection name: SoenSongs
Collection 'SoenSongs' created.
```

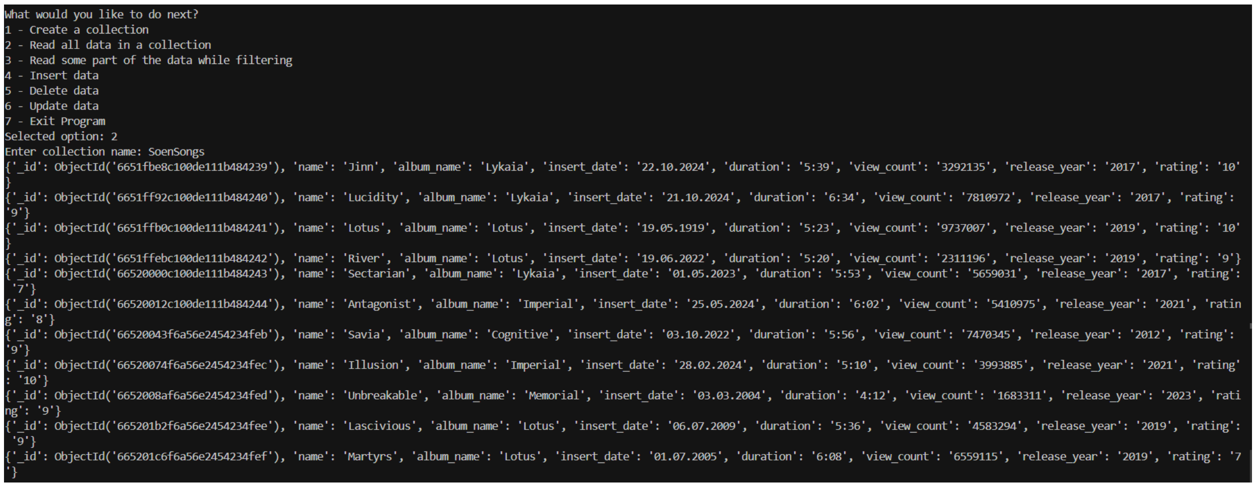
Figure 11. Creating SoenSongs collection for indicated user

c. **Reading All Data in a Collection:** The read_all_data function retrieves and displays all documents in a specified collection. This function provides a comprehensive view of the collection's contents. Figure 12 shows reading all data from the OpethSongs collection and Figure 13 shows reading all data from the SoenSongs collection.



```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 2
Enter collection name: OpethSongs
{'_id': ObjectId('6651fb76c100de111b484238'), 'name': 'Burden', 'album_name': 'Watershed', 'insert_date': '03.03.2023', 'duration': '7:41', 'view_count': '14109833', 'release_year': '2008', 'rating'
: '10'}
{'_id': ObjectId('6651fc28c100de111b48423a'), 'name': 'Windowpane', 'album_name': 'Damnation', 'insert_date': '18.01.2022', 'duration': '7:44', 'view_count': '28197222', 'release_year': '2003', 'rat
ing': '10'}
{'_id': ObjectId('6651fc49c100de111b48423b'), 'name': 'In My Time of Need', 'album_name': 'Damnation', 'insert_date': '01.05.2023', 'duration': '5:46', 'view_count': '18642231', 'release_year': '200
3', 'rating': '9'}
{'_id': ObjectId('6651fc7ac100de111b48423c'), 'name': 'Hope Leaves', 'album_name': 'Damnation', 'insert_date': '01.05.2023', 'duration': '4:27', 'view_count': '10071766', 'release_year': '2003', 'ra
ting': '8'}
{'_id': ObjectId('6651fca0c100de111b48423d'), 'name': 'Harvest', 'album_name': 'Blackwater Park', 'insert_date': '03.03.2004', 'duration': '6:01', 'view_count': '14069741', 'release_year': '2001', '
rating': '10'}
{'_id': ObjectId('6651fcbdc100de111b48423e'), 'name': 'To Bid You Farewell', 'album_name': 'Morningrise', 'insert_date': '29.10.1923', 'duration': '10:54', 'view_count': '11802394', 'release_year':
'1996', 'rating': '10'}
{'_id': ObjectId('6651fcdac100de111b48423f'), 'name': 'Face of Melinda', 'album_name': 'Still Life', 'insert_date': '23.04.2020', 'duration': '7:58', 'view_count': '7149229', 'release_year': '1999',
 'rating': '9'}
```

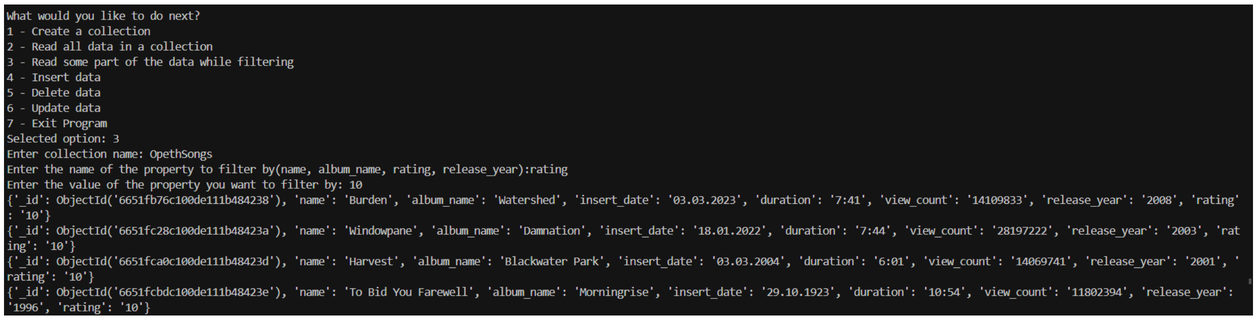
Figure 12. Reading all data in the OpethSongs collection



```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 2
Enter collection name: SoenSongs
{'_id': ObjectId('6651fbe8c100de111b484239'), 'name': 'Jinn', 'album_name': 'Lykaia', 'insert_date': '22.10.2024', 'duration': '5:39', 'view_count': '3292135', 'release_year': '2017', 'rating': '10'
}
{'_id': ObjectId('6651ff92c100de111b484240'), 'name': 'Lucidity', 'album_name': 'Lykaia', 'insert_date': '21.10.2024', 'duration': '6:34', 'view_count': '7810972', 'release_year': '2017', 'rating':
'9'}
{'_id': ObjectId('6651ffb0c100de111b484241'), 'name': 'Lotus', 'album_name': 'Lotus', 'insert_date': '19.05.1919', 'duration': '5:23', 'view_count': '9737007', 'release_year': '2019', 'rating': '10'
}
{'_id': ObjectId('6651ffebc100de111b484242'), 'name': 'River', 'album_name': 'Lotus', 'insert_date': '19.06.2022', 'duration': '5:20', 'view_count': '2311196', 'release_year': '2019', 'rating': '9'}
{'_id': ObjectId('66520000c100de111b484243'), 'name': 'Sectarian', 'album_name': 'Lykaia', 'insert_date': '01.05.2023', 'duration': '5:53', 'view_count': '5659031', 'release_year': '2017', 'rating':
 '7'}
{'_id': ObjectId('66520012c100de111b484244'), 'name': 'Antagonist', 'album_name': 'Imperial', 'insert_date': '25.05.2024', 'duration': '6:02', 'view_count': '5410975', 'release_year': '2021', 'ratin
g': '8'}
{'_id': ObjectId('66520043f6a56e2454234feb'), 'name': 'Savia', 'album_name': 'Cognitive', 'insert_date': '03.10.2022', 'duration': '5:56', 'view_count': '7470345', 'release_year': '2012', 'rating':
'9'}
{'_id': ObjectId('66520074f6a56e2454234fec'), 'name': 'Illusion', 'album_name': 'Imperial', 'insert_date': '28.02.2024', 'duration': '5:10', 'view_count': '3993885', 'release_year': '2021', 'rating'
: '10'}
{'_id': ObjectId('6652008af6a56e2454234fed'), 'name': 'Unbreakable', 'album_name': 'Memorial', 'insert_date': '03.03.2004', 'duration': '4:12', 'view_count': '1683311', 'release_year': '2023', 'rati
ng': '9'}
{'_id': ObjectId('665201b2f6a56e2454234fee'), 'name': 'Lascivious', 'album_name': 'Lotus', 'insert_date': '06.07.2009', 'duration': '5:36', 'view_count': '4583294', 'release_year': '2019', 'rating':
 '9'}
{'_id': ObjectId('665201c6f6a56e2454234fef'), 'name': 'Martyrs', 'album_name': 'Lotus', 'insert_date': '01.07.2005', 'duration': '6:08', 'view_count': '6559115', 'release_year': '2019', 'rating': '7
'}
```

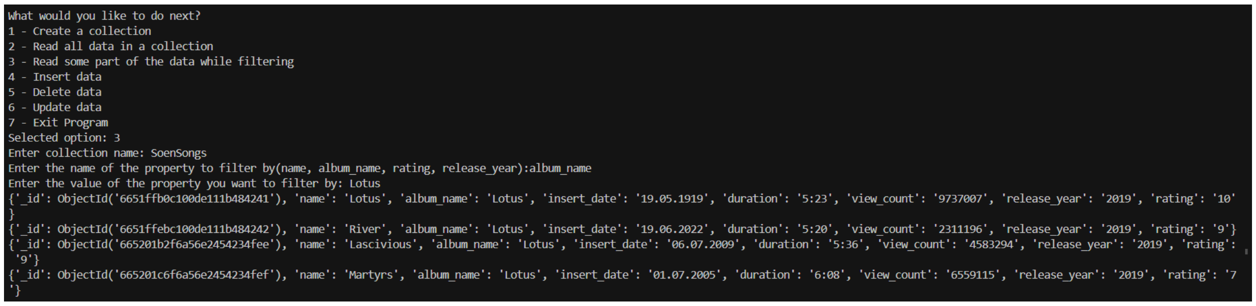
Figure 13. Reading all data in the SoenSongs collection

d. **Reading Data with Filtering:** The find_songs_containing_value function retrieves documents that match a specific property value. This allows users to filter songs based on certain criteria. Figure 14 shows reading data from the OpethSongs collection filtered by the song name "Harvest". Figure 15 shows reading specifically filtered data from the SoenSongs table.



```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 3
Enter collection name: OpethSongs
Enter the name of the property to filter by(name, album_name, rating, release_year):rating
Enter the value of the property you want to filter by: 10
{'_id': ObjectId('6651fb76c100de111b484238'), 'name': 'Burden', 'album_name': 'Watershed', 'insert_date': '03.03.2023', 'duration': '7:41', 'view_count': '14109833', 'release_year': '2008', 'rating'
: '10'}
{'_id': ObjectId('6651fc28c100de111b48423a'), 'name': 'Windowpane', 'album_name': 'Damnation', 'insert_date': '18.01.2022', 'duration': '7:44', 'view_count': '28197222', 'release_year': '2003', 'rat
ing': '10'}
{'_id': ObjectId('6651fca0c100de111b48423d'), 'name': 'Harvest', 'album_name': 'Blackwater Park', 'insert_date': '03.03.2004', 'duration': '6:01', 'view_count': '14069741', 'release_year': '2001', '
rating': '10'}
{'_id': ObjectId('6651fcbdc100de111b48423e'), 'name': 'To Bid You Farewell', 'album_name': 'Morningrise', 'insert_date': '29.10.1923', 'duration': '10:54', 'view_count': '11802394', 'release_year':
'1996', 'rating': '10'}
```

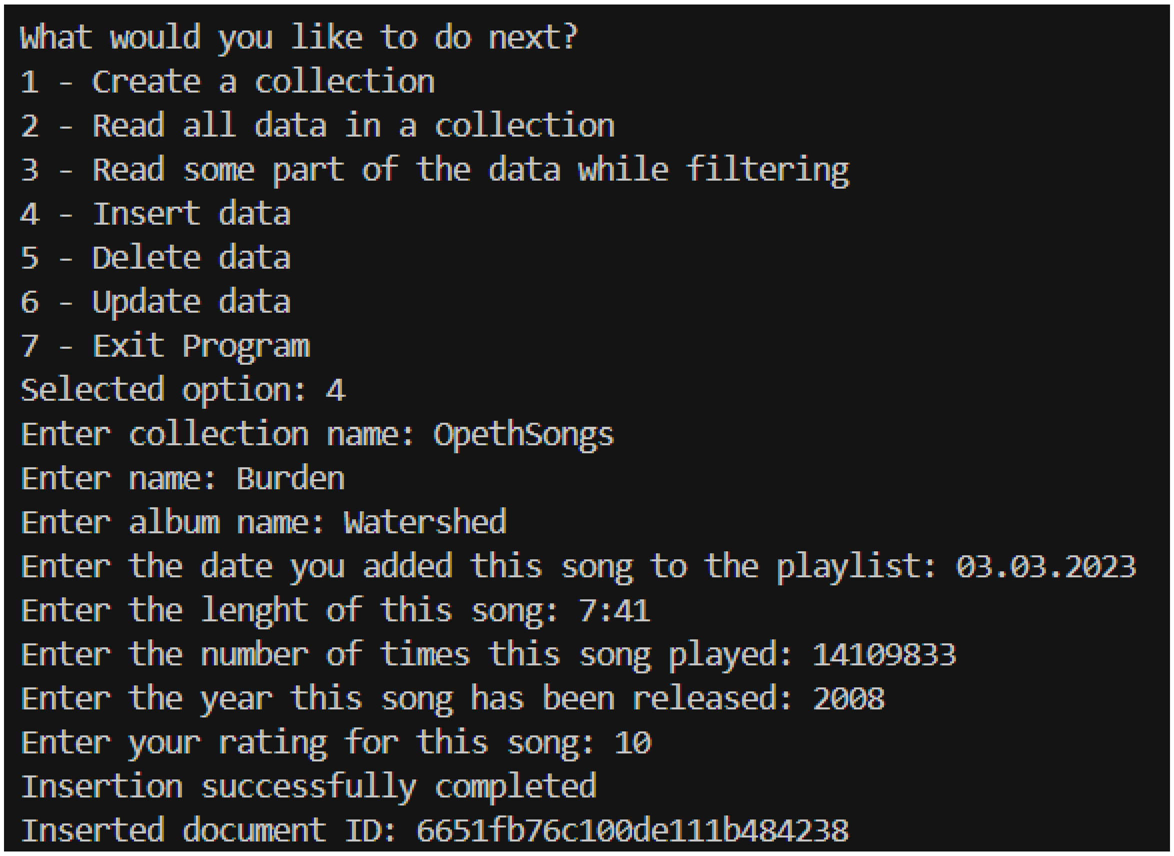
Figure 14. Reading specific data in the OpethSongs collection



```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 3
Enter collection name: SoenSongs
Enter the name of the property to filter by(name, album_name, rating, release_year):album_name
Enter the value of the property you want to filter by: Lotus
{'_id': ObjectId('6651ffb0c100de111b484241'), 'name': 'Lotus', 'album_name': 'Lotus', 'insert_date': '19.05.1919', 'duration': '5:23', 'view_count': '9737007', 'release_year': '2019', 'rating': '10'
}
{'_id': ObjectId('6651ffebc100de111b484242'), 'name': 'River', 'album_name': 'Lotus', 'insert_date': '19.06.2022', 'duration': '5:20', 'view_count': '2311196', 'release_year': '2019', 'rating': '9'}
{'_id': ObjectId('665201b2f6a56e2454234fee'), 'name': 'Lascivious', 'album_name': 'Lotus', 'insert_date': '06.07.2009', 'duration': '5:36', 'view_count': '4583294', 'release_year': '2019', 'rating':
 '9'}
{'_id': ObjectId('665201c6f6a56e2454234fef'), 'name': 'Martyrs', 'album_name': 'Lotus', 'insert_date': '01.07.2005', 'duration': '6:08', 'view_count': '6559115', 'release_year': '2019', 'rating': '7
'}
```

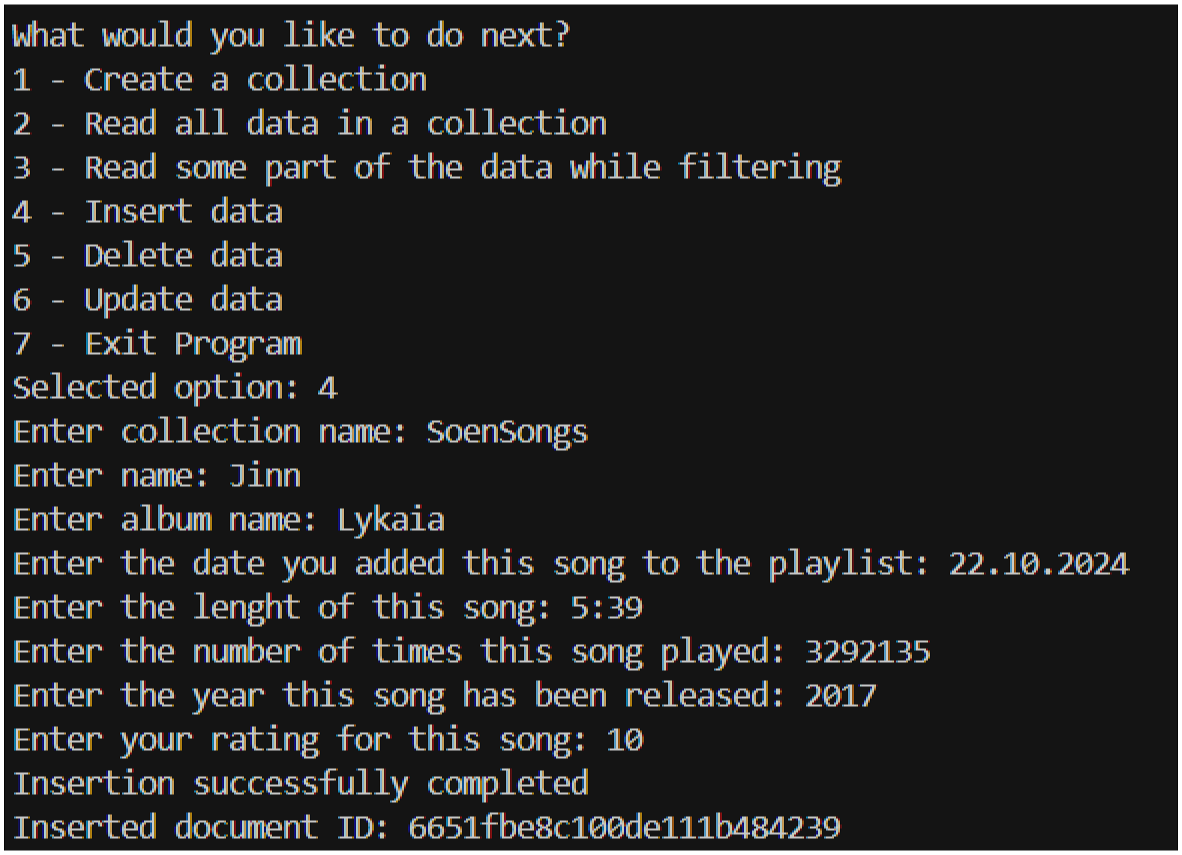
Figure 15. Reading specific data in the SoenSongs collection

e. **Inserting Data:** The insert_into_collection function allows users to add new songs to their collections by providing detailed information about each song. Figure 16 shows inserting a new song into the OpethSongs collection. Figure 17 shows inserting a new song into the SoenSongs collection.



```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 4
Enter collection name: OpethSongs
Enter name: Burden
Enter album name: Watershed
Enter the date you added this song to the playlist: 03.03.2023
Enter the lenght of this song: 7:41
Enter the number of times this song played: 14109833
Enter the year this song has been released: 2008
Enter your rating for this song: 10
Insertion successfully completed
Inserted document ID: 6651fb76c100de111b484238
```

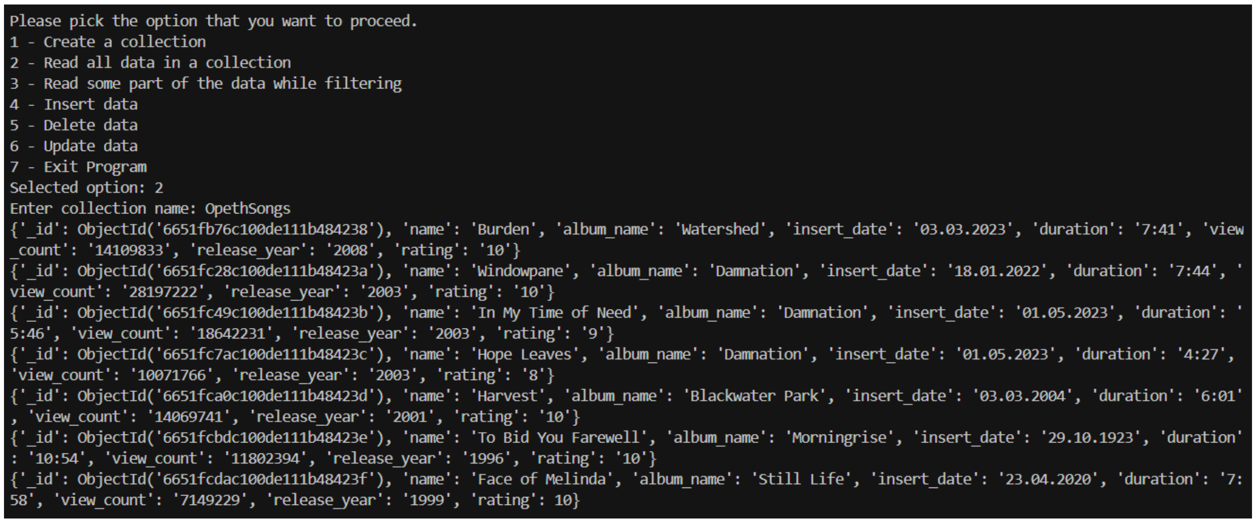
Figure 16. Inserting a new song into the OpethSongs collection



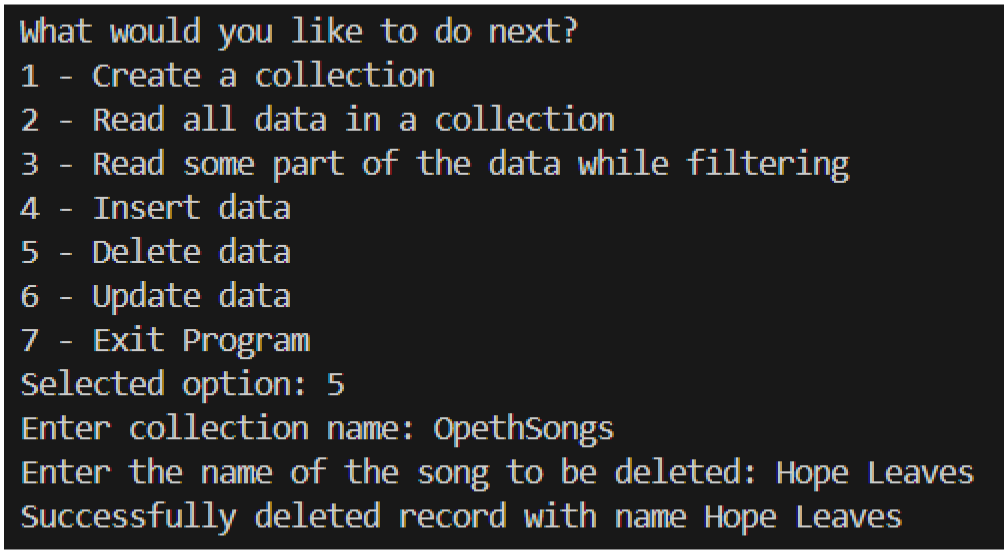
```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 4
Enter collection name: SoenSongs
Enter name: Jinn
Enter album name: Lykaia
Enter the date you added this song to the playlist: 22.10.2024
Enter the lenght of this song: 5:39
Enter the number of times this song played: 3292135
Enter the year this song has been released: 2017
Enter your rating for this song: 10
Insertion successfully completed
Inserted document ID: 6651fbe8c100de111b484239
```

Figure 17. Inserting a new song into the SoenSongs collection

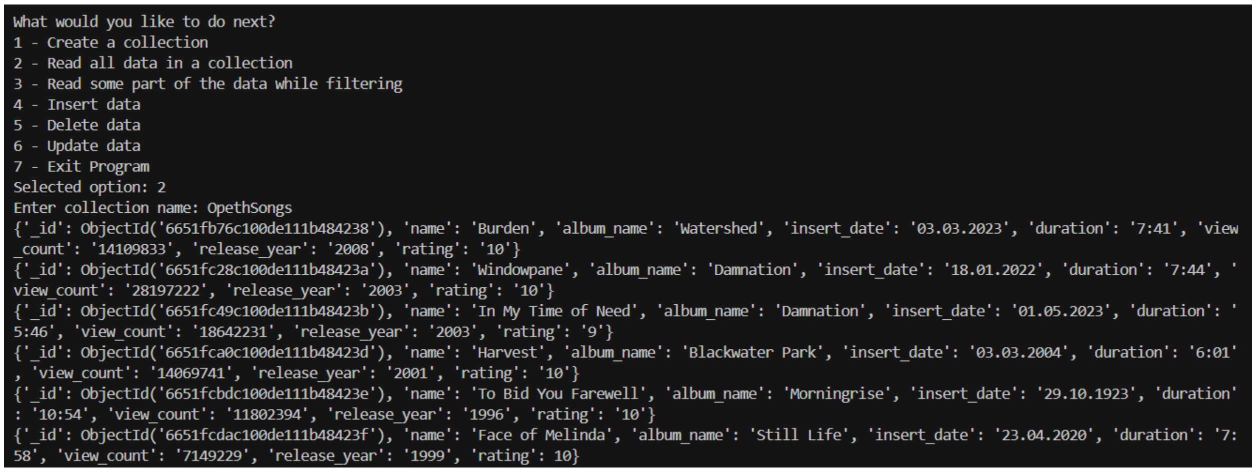
f. **Deleting Data:** The delete_record_by_name function allows users to delete songs from their collections by specifying the song name. Figure 18 shows deleting a song from the OpethSongs collection, and Figure 19 shows deleting a song from the SoenSongs collection. These figures shows the collection before and after performing deletion operations with the queried deletion operation, as well.



```
Please pick the option that you want to proceed.
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 2
Enter collection name: OpethSongs
{'_id': ObjectId('6651fb76c100de111b484238'), 'name': 'Burden', 'album_name': 'Watershed', 'insert_date': '03.03.2023', 'duration': '7:41', 'view
_count': '14109833', 'release_year': '2008', 'rating': '10'}
{'_id': ObjectId('6651fc28c100de111b48423a'), 'name': 'Windowpane', 'album_name': 'Damnation', 'insert_date': '18.01.2022', 'duration': '7:44', '
view_count': '28197222', 'release_year': '2003', 'rating': '10'}
{'_id': ObjectId('6651fc49c100de111b48423b'), 'name': 'In My Time of Need', 'album_name': 'Damnation', 'insert_date': '01.05.2023', 'duration': '
5:46', 'view_count': '18642231', 'release_year': '2003', 'rating': '9'}
{'_id': ObjectId('6651fc7ac100de111b48423c'), 'name': 'Hope Leaves', 'album_name': 'Damnation', 'insert_date': '01.05.2023', 'duration': '4:27',
'view_count': '10071766', 'release_year': '2003', 'rating': '8'}
{'_id': ObjectId('6651fca0c100de111b48423d'), 'name': 'Harvest', 'album_name': 'Blackwater Park', 'insert_date': '03.03.2004', 'duration': '6:01'
, 'view_count': '14069741', 'release_year': '2001', 'rating': '10'}
{'_id': ObjectId('6651fcbdc100de111b48423e'), 'name': 'To Bid You Farewell', 'album_name': 'Morningrise', 'insert_date': '29.10.1923', 'duration'
: '10:54', 'view_count': '11802394', 'release_year': '1996', 'rating': '10'}
{'_id': ObjectId('6651fcdac100de111b48423f'), 'name': 'Face of Melinda', 'album_name': 'Still Life', 'insert_date': '23.04.2020', 'duration': '7:
58', 'view_count': '7149229', 'release_year': '1999', 'rating': 10}
```

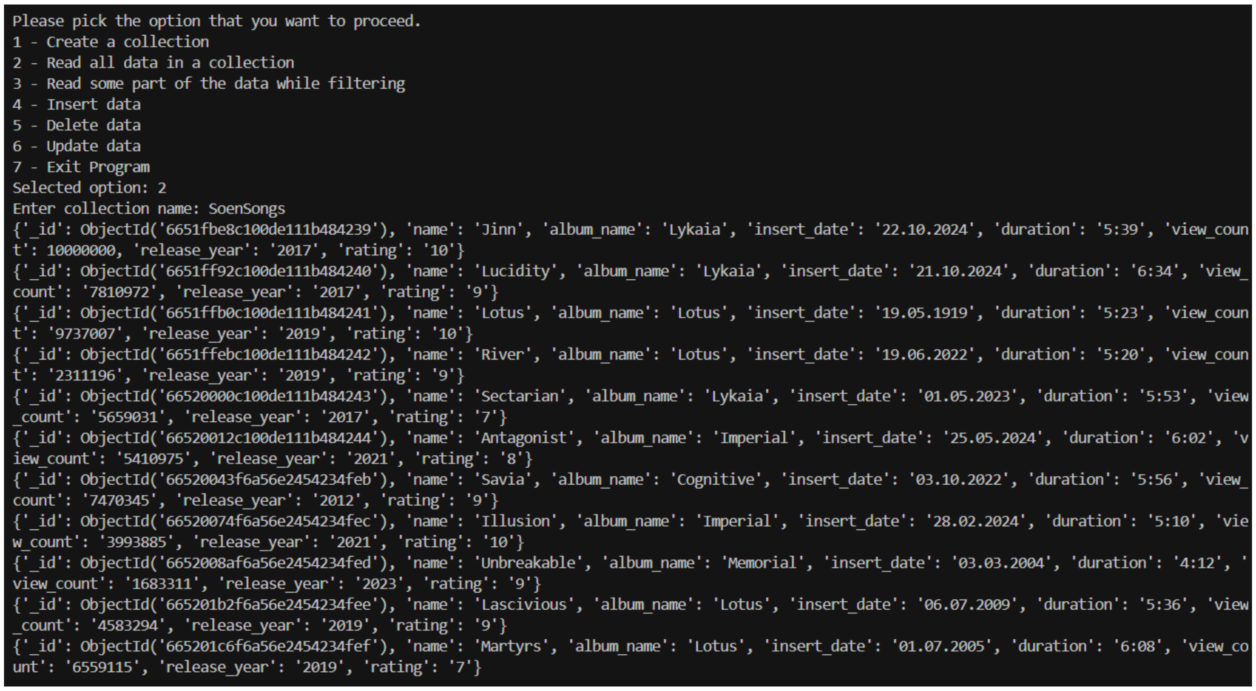


```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 5
Enter collection name: OpethSongs
Enter the name of the song to be deleted: Hope Leaves
Successfully deleted record with name Hope Leaves
```

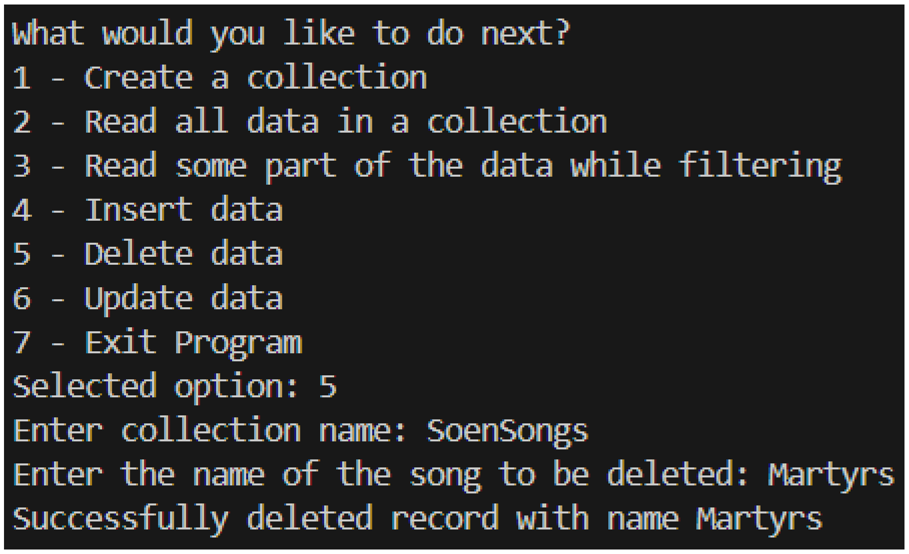


```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 2
Enter collection name: OpethSongs
{'_id': ObjectId('6651fb76c100de111b484238'), 'name': 'Burden', 'album_name': 'Watershed', 'insert_date': '03.03.2023', 'duration': '7:41', 'view
_count': '14109833', 'release_year': '2008', 'rating': '10'}
{'_id': ObjectId('6651fc28c100de111b48423a'), 'name': 'Windowpane', 'album_name': 'Damnation', 'insert_date': '18.01.2022', 'duration': '7:44', '
view_count': '28197222', 'release_year': '2003', 'rating': '10'}
{'_id': ObjectId('6651fc49c100de111b48423b'), 'name': 'In My Time of Need', 'album_name': 'Damnation', 'insert_date': '01.05.2023', 'duration': '
5:46', 'view_count': '18642231', 'release_year': '2003', 'rating': '9'}
{'_id': ObjectId('6651fca0c100de111b48423d'), 'name': 'Harvest', 'album_name': 'Blackwater Park', 'insert_date': '03.03.2004', 'duration': '6:01'
, 'view_count': '14069741', 'release_year': '2001', 'rating': '10'}
{'_id': ObjectId('6651fcbdc100de111b48423e'), 'name': 'To Bid You Farewell', 'album_name': 'Morningrise', 'insert_date': '29.10.1923', 'duration'
: '10:54', 'view_count': '11802394', 'release_year': '1996', 'rating': '10'}
{'_id': ObjectId('6651fcdac100de111b48423f'), 'name': 'Face of Melinda', 'album_name': 'Still Life', 'insert_date': '23.04.2020', 'duration': '7:
58', 'view_count': '7149229', 'release_year': '1999', 'rating': 10}
```

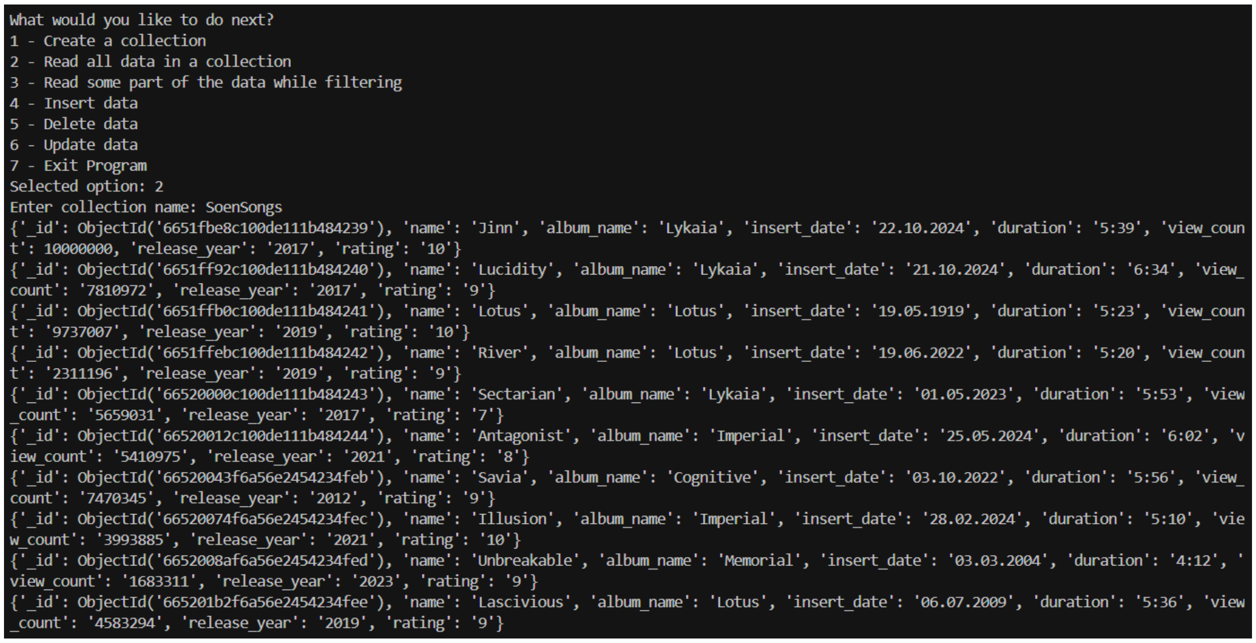
Figure 18. Deleting a song from the OpethSongs collection



```
Please pick the option that you want to proceed.
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 2
Enter collection name: SoenSongs
{'_id': ObjectId('6651fbe8c100de111b484239'), 'name': 'Jinn', 'album_name': 'Lykaia', 'insert_date': '22.10.2024', 'duration': '5:39', 'view_coun
t': 10000000, 'release_year': '2017', 'rating': '10'}
{'_id': ObjectId('6651ff92c100de111b484240'), 'name': 'Lucidity', 'album_name': 'Lykaia', 'insert_date': '21.10.2024', 'duration': '6:34', 'view_
count': '7810972', 'release_year': '2017', 'rating': '9'}
{'_id': ObjectId('6651ffb0c100de111b484241'), 'name': 'Lotus', 'album_name': 'Lotus', 'insert_date': '19.05.1919', 'duration': '5:23', 'view_coun
t': '9737007', 'release_year': '2019', 'rating': '10'}
{'_id': ObjectId('6651ffebc100de111b484242'), 'name': 'River', 'album_name': 'Lotus', 'insert_date': '19.06.2022', 'duration': '5:20', 'view_coun
t': '2311196', 'release_year': '2019', 'rating': '9'}
{'_id': ObjectId('66520000c100de111b484243'), 'name': 'Sectarian', 'album_name': 'Lykaia', 'insert_date': '01.05.2023', 'duration': '5:53', 'view
_count': '5659031', 'release_year': '2017', 'rating': '7'}
{'_id': ObjectId('66520012c100de111b484244'), 'name': 'Antagonist', 'album_name': 'Imperial', 'insert_date': '25.05.2024', 'duration': '6:02', 'v
iew_count': '5410975', 'release_year': '2021', 'rating': '8'}
{'_id': ObjectId('66520043f6a56e2454234feb'), 'name': 'Savia', 'album_name': 'Cognitive', 'insert_date': '03.10.2022', 'duration': '5:56', 'view_
count': '7470345', 'release_year': '2012', 'rating': '9'}
{'_id': ObjectId('66520074f6a56e2454234fec'), 'name': 'Illusion', 'album_name': 'Imperial', 'insert_date': '28.02.2024', 'duration': '5:10', 'vie
w_count': '3993885', 'release_year': '2021', 'rating': '10'}
{'_id': ObjectId('6652008af6a56e2454234fed'), 'name': 'Unbreakable', 'album_name': 'Memorial', 'insert_date': '03.03.2004', 'duration': '4:12', '
view_count': '1683311', 'release_year': '2023', 'rating': '9'}
{'_id': ObjectId('665201b2f6a56e2454234fee'), 'name': 'Lascivious', 'album_name': 'Lotus', 'insert_date': '06.07.2009', 'duration': '5:36', 'view
_count': '4583294', 'release_year': '2019', 'rating': '9'}
{'_id': ObjectId('665201c6f6a56e2454234fef'), 'name': 'Martyrs', 'album_name': 'Lotus', 'insert_date': '01.07.2005', 'duration': '6:08', 'view_co
unt': '6559115', 'release_year': '2019', 'rating': '7'}
```



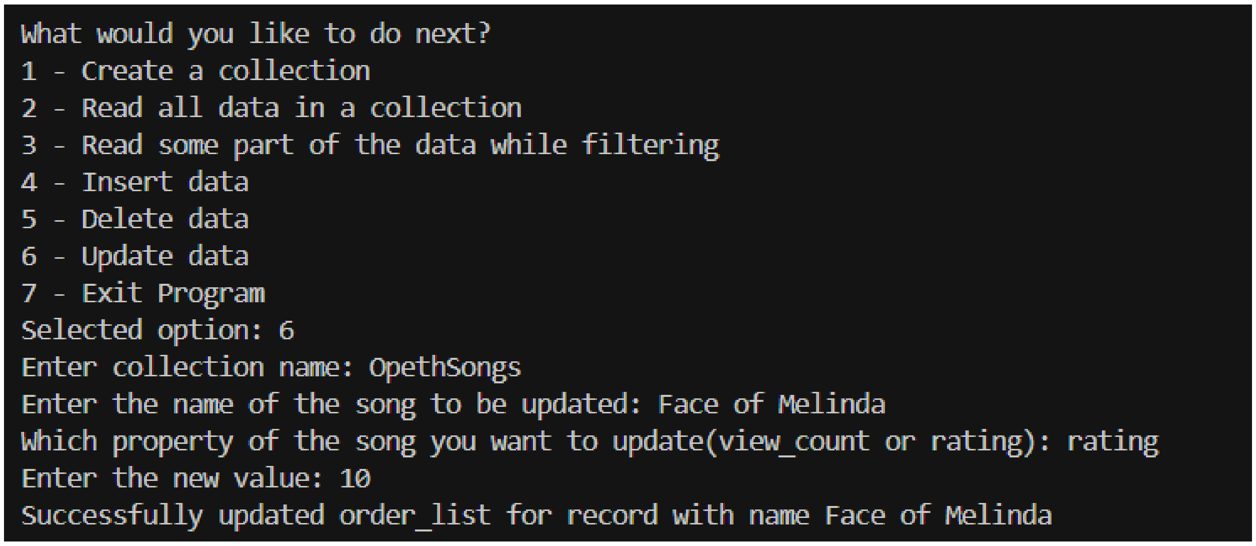
```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 5
Enter collection name: SoenSongs
Enter the name of the song to be deleted: Martyrs
Successfully deleted record with name Martyrs
```



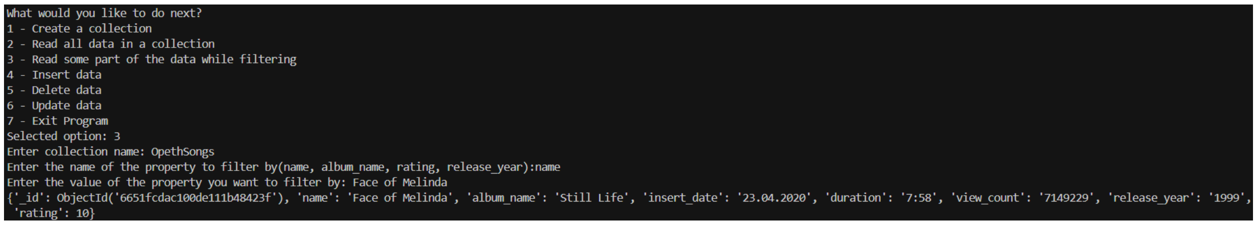
```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 2
Enter collection name: SoenSongs
{'_id': ObjectId('6651fbe8c100de111b484239'), 'name': 'Jinn', 'album_name': 'Lykaia', 'insert_date': '22.10.2024', 'duration': '5:39', 'view_coun
t': 10000000, 'release_year': '2017', 'rating': '10'}
{'_id': ObjectId('6651ff92c100de111b484240'), 'name': 'Lucidity', 'album_name': 'Lykaia', 'insert_date': '21.10.2024', 'duration': '6:34', 'view_
count': '7810972', 'release_year': '2017', 'rating': '9'}
{'_id': ObjectId('6651ffb0c100de111b484241'), 'name': 'Lotus', 'album_name': 'Lotus', 'insert_date': '19.05.1919', 'duration': '5:23', 'view_coun
t': '9737007', 'release_year': '2019', 'rating': '10'}
{'_id': ObjectId('6651ffebc100de111b484242'), 'name': 'River', 'album_name': 'Lotus', 'insert_date': '19.06.2022', 'duration': '5:20', 'view_coun
t': '2311196', 'release_year': '2019', 'rating': '9'}
{'_id': ObjectId('66520000c100de111b484243'), 'name': 'Sectarian', 'album_name': 'Lykaia', 'insert_date': '01.05.2023', 'duration': '5:53', 'view
_count': '5659031', 'release_year': '2017', 'rating': '7'}
{'_id': ObjectId('66520012c100de111b484244'), 'name': 'Antagonist', 'album_name': 'Imperial', 'insert_date': '25.05.2024', 'duration': '6:02', 'v
iew_count': '5410975', 'release_year': '2021', 'rating': '8'}
{'_id': ObjectId('66520043f6a56e2454234feb'), 'name': 'Savia', 'album_name': 'Cognitive', 'insert_date': '03.10.2022', 'duration': '5:56', 'view_
count': '7470345', 'release_year': '2012', 'rating': '9'}
{'_id': ObjectId('66520074f6a56e2454234fec'), 'name': 'Illusion', 'album_name': 'Imperial', 'insert_date': '28.02.2024', 'duration': '5:10', 'vie
w_count': '3993885', 'release_year': '2021', 'rating': '10'}
{'_id': ObjectId('6652008af6a56e2454234fed'), 'name': 'Unbreakable', 'album_name': 'Memorial', 'insert_date': '03.03.2004', 'duration': '4:12', '
view_count': '1683311', 'release_year': '2023', 'rating': '9'}
{'_id': ObjectId('665201b2f6a56e2454234fee'), 'name': 'Lascivious', 'album_name': 'Lotus', 'insert_date': '06.07.2009', 'duration': '5:36', 'view
_count': '4583294', 'release_year': '2019', 'rating': '9'}
```

Figure 19. Deleting a song from the SoenSongs collection

g. **Updating Data:** The update_order_list_by_name function allows users to update specific attributes of a song, such as view count or rating. Figure 20 shows updating the view count of a song in the OpethSongs collection. Figure 21 shows updating the view count of a song in the OpethSongs collection.

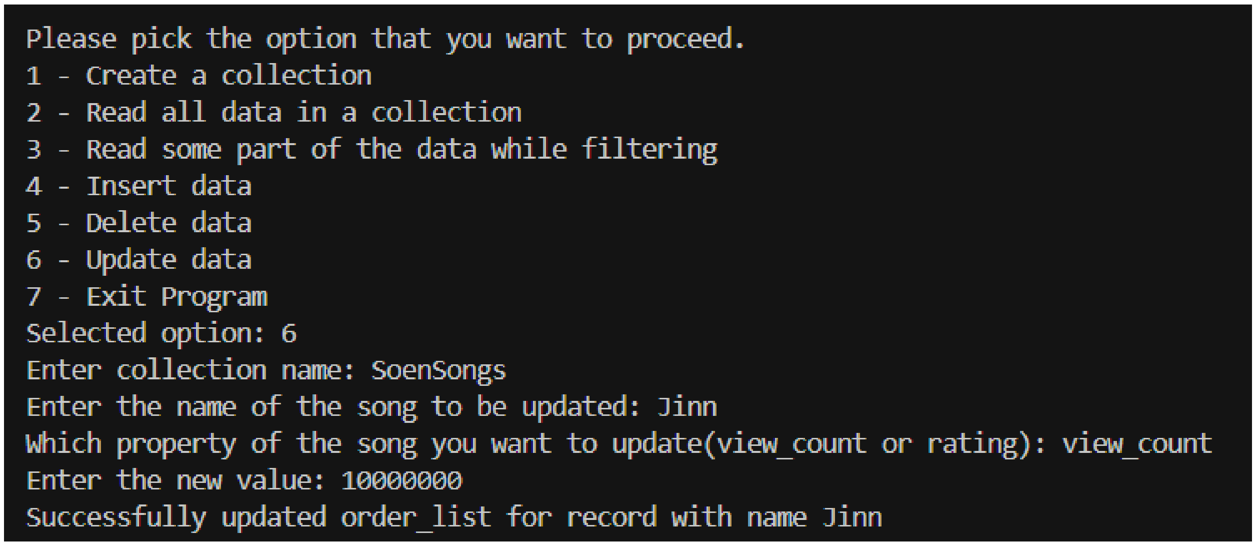


```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 6
Enter collection name: OpethSongs
Enter the name of the song to be updated: Face of Melinda
Which property of the song you want to update(view_count or rating): rating
Enter the new value: 10
Successfully updated order_list for record with name Face of Melinda
```

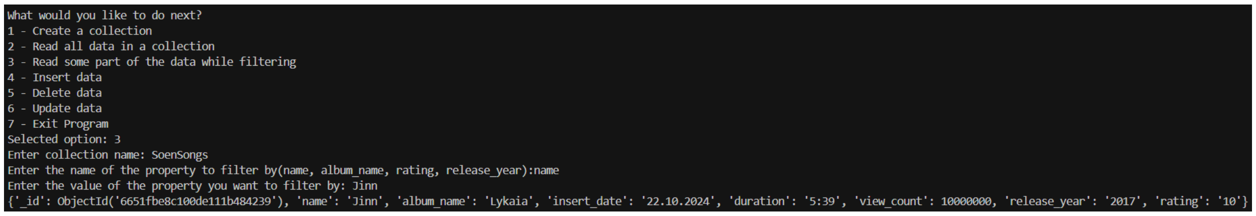


```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 3
Enter collection name: OpethSongs
Enter the name of the property to filter by(name, album_name, rating, release_year):name
Enter the value of the property you want to filter by: Face of Melinda
{'_id': ObjectId('6651fcdac100de111b48423f'), 'name': 'Face of Melinda', 'album_name': 'Still Life', 'insert_date': '23.04.2020', 'duration': '7:58', 'view_count': '7149229', 'release_year': '1999',
 'rating': 10}
```

Figure 20. Updating song details of the OpethSongs collection



```
Please pick the option that you want to proceed.
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 6
Enter collection name: SoenSongs
Enter the name of the song to be updated: Jinn
Which property of the song you want to update(view_count or rating): view_count
Enter the new value: 10000000
Successfully updated order_list for record with name Jinn
```



```
What would you like to do next?
1 - Create a collection
2 - Read all data in a collection
3 - Read some part of the data while filtering
4 - Insert data
5 - Delete data
6 - Update data
7 - Exit Program
Selected option: 3
Enter collection name: SoenSongs
Enter the name of the property to filter by(name, album_name, rating, release_year):name
Enter the value of the property you want to filter by: Jinn
{'_id': ObjectId('6651fbe8c100de111b484239'), 'name': 'Jinn', 'album_name': 'Lykaia', 'insert_date': '22.10.2024', 'duration': '5:39', 'view_count': 10000000, 'release_year': '2017', 'rating': '10'}
```

Figure 21. Updating song details of the SoenSongs collection

6. Conclusion

This project involved the implementation of a playlist service using a MongoDB Atlas database to manage users' music collections. The service allows users to create and interact with collections specifically for Opeth and Soen songs, storing detailed information about each track. Through a user-friendly, menu-driven interface, users can perform various CRUD operations, including creating collections, reading all data, filtering data, inserting new songs, deleting songs, and updating song details. The system ensures personalized database management through user-specific databases, providing an efficient and organized way for music enthusiasts to maintain their favorite tracks. Overall, the project successfully demonstrated the integration of MongoDB Atlas with Python to offer a robust solution for managing music collections.