Enhancements in Database Management

Zaffar Shiekh

CS-499-10941-M01 Computer Science Capstone 2024 C-4

Federico Bermudez

08/18/2024

**1. Introduction**

This document details the database management enhancements made to the Grazioso Salvare Dashboard. These changes focus on optimizing data retrieval, implementing security measures, and ensuring data integrity through advanced MongoDB features.

**2. Software Requirements**

Ensure you have the following software installed:

1. **Python 3.x**
2. **Jupyter Lab** or **Visual Studio Code (VSCode)** with Python extensions
   * **Jupyter Lab**: Install using pip:

pip install jupyterlab

* + **VSCode**: Install from [Visual Studio Code's official website](https://code.visualstudio.com/) and add the Python extension.

1. **MongoDB and MongoDB Compass**
   * Install MongoDB Community Edition and MongoDB Compass from [MongoDB's official website](https://www.mongodb.com/try/download/community).
2. **Python Packages**
   * Install via pip:

pip install pandas dash dash\_table pymongo

**3. File Setup**

Ensure the following files are in place:

1. **Jupyter Notebook (example\_notebook.ipynb) or Python script (example\_script.py)**
   * Store this file in the main project directory.
2. **Data File (AAC.animals.json)**
   * Store this file in the same directory as the Jupyter Notebook or Python script.

**Directory Structure Example**:

GraziosoSalvareProject/

│

├── example\_notebook.ipynb # or example\_script.py

└── AAC.animals.json

**4. Implementing Database Management Enhancements**

**4.1 Old Code**

**Original MongoDB Connection and Query Logic:**

class AnimalShelter(object):

def \_\_init\_\_(self, username, password, host='nv-desktop-services.apporto.com', port=30043, db='AAC', collection='animals'):

self.client = MongoClient(f'mongodb://{username}:{password}@{host}:{port}/')

self.database = self.client[db]

self.collection = self.database[collection]

**Issues:**

* The connection string was hardcoded to a specific host, limiting flexibility.
* The query logic was not optimized for complex queries or indexing.

**4.2 New Code**

**Updated MongoDB Connection and Query Logic:**

from pymongo import MongoClient

from bson.objectid import ObjectId

class AnimalShelter(object):

"""CRUD operations for Animal collection in MongoDB"""

def \_\_init\_\_(self, username=None, password=None, host='localhost', port=27017, db='AAC', collection='animals'):

# Initialize Connection

if username and password:

self.client = MongoClient(f'mongodb://{username}:{password}@{host}:{port}/')

else:

self.client = MongoClient(f'mongodb://{host}:{port}/')

self.database = self.client[db]

self.collection = self.database[collection]

def create(self, data):

"""Inserts a document into the MongoDB collection."""

if data is not None:

try:

result = self.collection.insert\_one(data) # Insert the document

return True # If insertion was successful

except Exception as e:

print(f"An error occurred: {e}")

return False # If unsuccessful

else:

raise Exception("Nothing to save, because data parameter is empty")

def read(self, criteria):

"""Queries for documents and returns them as a list from MongoDB."""

try:

documents = list(self.collection.find(criteria))

return documents # Return the list of documents

except Exception as e:

print(f"An error occurred: {e}")

return [] # Return an empty list if an error occurs

def update(self, criteria, update\_values):

"""Updates documents matching criteria with new values."""

try:

result = self.collection.update\_many(criteria, update\_values)

return result.modified\_count # Return the number of documents modified

except Exception as e:

print(f"An error occurred during update: {e}")

return 0 # Return 0 if no documents were updated

def delete(self, criteria):

"""Deletes documents matching criteria."""

try:

result = self.collection.delete\_many(criteria)

return result.deleted\_count # Return the number of documents deleted

except Exception as e:

print(f"An error occurred during delete: {e}")

return 0 # Return 0 if no documents were deleted

**Changes and Reasons:**

* **Removed Hardcoded Host:** The connection string now defaults to localhost, with the option to specify a custom host, port, username, and password if needed. This increases flexibility, allowing the application to connect to different MongoDB instances without modification.
* **Improved Error Handling:** Error handling has been added to all CRUD operations to ensure that any issues during database interactions are caught and reported to the user.
* **CRUD Operations:** The create, read, update, and delete methods are now robustly handled, with error checking and feedback for better reliability.

**5. Running the Dashboard**

1. **Open Jupyter Lab or VSCode**:
   * Run Jupyter Lab by executing:

jupyter lab

* + Or open the Python script in VSCode.

1. **Load the Notebook or Script**:
   * Open example\_notebook.ipynb in Jupyter Lab or example\_script.py in VSCode.
2. **Run the Notebook or Script**:
   * Execute all cells in the notebook or run the Python script to launch the dashboard. In VSCode, you can run the script by pressing F5 or using the "Run Python File" command.
3. **Access the Dashboard**:
   * Once the dashboard runs, open your web browser and navigate to http://127.0.0.1:8050/ to view the dashboard interface.
4. **Running Jupyter as Administrator (Windows Users)**:
   * When running Jupyter Notebook or Jupyter Lab through the Command Prompt or Anaconda, it is recommended to open the Command Prompt or Anaconda as an administrator to avoid potential permission issues.

**6. Testing the Enhancements**

* **Advanced Query Techniques**: Test the aggregation queries to ensure they retrieve the correct data efficiently.
* **Database Indexing**: Verify that the indexes on frequently queried fields (e.g., rescue\_type, location) improve query performance.
* **Data Security Enhancements**: Confirm that role-based access control (RBAC) works correctly, ensuring only authorized users can perform specific actions.
* **Backup and Recovery**: To ensure the backup and recovery process is functional, backup the database using mongodump and restore it using mongorestore.