Bryce Jensen
CS-340-Q7708
5-1: README
2/7/2024

About the Project/Project Title

The Grazioso Salvare Canine Rescue Training Dashboard is a project developed by Global Rain to assist Grazioso Salvare, an innovative international rescue-animal training company. One goal of this project is to identify and categorize dogs that are capable of search-and-rescue training.

This project takes existing data from 5 non-profit animal shelters in the Austin, Texas region. This enable Grazioso Salvare to efficiently identify potential candidates for their training programs. Some key features of the project include filtering dogs based on their age, breed, and other characteristics that may indicate a suitability for different types of rescue missions such as water rescue, mountain rescue, disaster recovery, or scent tracking.

Motivation

This project was created to address the critical need to find dogs suitable for search-and-rescue training. The main motivation of this project is to improve the effectiveness and impact of Grazioso Salvare's training programs.

The decision to make the project open source is to allow collaboration and help the rescue animal training community. We are hoping to enable other similar organizations to adapt this software to their specific needs, enhancing the impact of canine rescue efforts around the world.

Getting Started

To get a local copy up and running, follow these simple example steps:

- Clone the Grazioso Salvare Canine Rescue Training Dashboard project from Github:
    - Open a command prompt.

- `git clone https://github.com/global_rain/grazioso-salvare-canine-rescue-dashboard.git`

Installation

- Install Python 3.12.1
- Install MongoDB (Community Edition)

Usage and Testing

Code

The code implements a CRUD (Create, Read, Update, Delete) to a MongoDB server.

`AnimalShelter()`

The constructor for the animalShelter Class Initializes the MongoDB server.

Example Constructor:

```
CRUD = AnimalShelter()
```

`create()`

Creates a new item in the database. Takes one argument for the data to be added. Must be in dictionary form.

Example Test:

```
newAnimalTestData = {
    'age_upon_outcome': '10 years',
    'animal_id': 'BGJ0715',
    'animal_type': 'Dog',
    'breed': 'Border Collie',
    'color': 'Black and White',
    'date_of_birth': '2013-11-19',
    'datetime': '2024-02-06 18:21:36',
    'name': 'Doug',
    'outcome_subtype': 'SCRP',
    'outcome_type': 'Adopted',
    'sex_upon_outcome': 'Neutered Male',
    'location_lat': 30.6525984560228,
    'location_long': -97.7419963476444,
    'age_upon_outcome_in_weeks': 533
}

created = CRUD.create(newAnimalTestData)
print(f"Item created: {created}")
```

`read()`	Finds and prints all documents of the search. Takes one argument as the search query. Can be as long as needed but must be in the correct Mongo format for queries.

Example Test:

```
read_count = CRUD.read(
    {"animal_id": "BGJ0715"})

print(f"There are {read_count} document(s) that fit the query.")
```

`update()`	Modifies a current document. Takes two arguments, one as the search to find the documents to be edited (must be in the correct Mongo format for queries), and the other is the data to be updated (must be in dictionary format).

Example Test:

```
updated_count = CRUD.update(
    {"animal_id": "BGJ0715"},
    {"outcome_type": "Adopted"})

print(CRUD.read({"animal_id": "BGJ0715"}))
print(f"There was {updated_count} document(s) changed.")
```

`delete()`	Deletes a current document. Takes one argument as the search query. Can be as long as needed but must be in the correct Mongo format for queries.

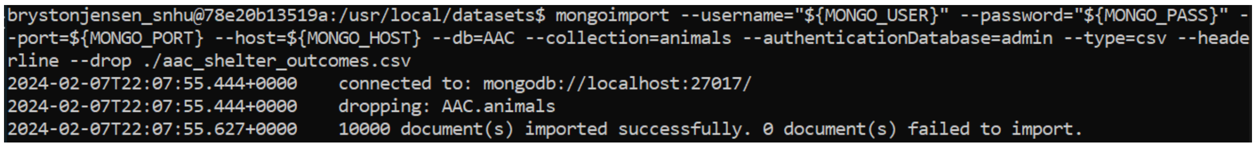
Example Test:

```
deleted_count = CRUD.delete(
    {'animal_id': 'BGJ0715'})
print(f"There was {deleted_count} document(s) deleted.")
```

Bryce Jensen
CS-340-Q7708
5-1: README
2/7/2024

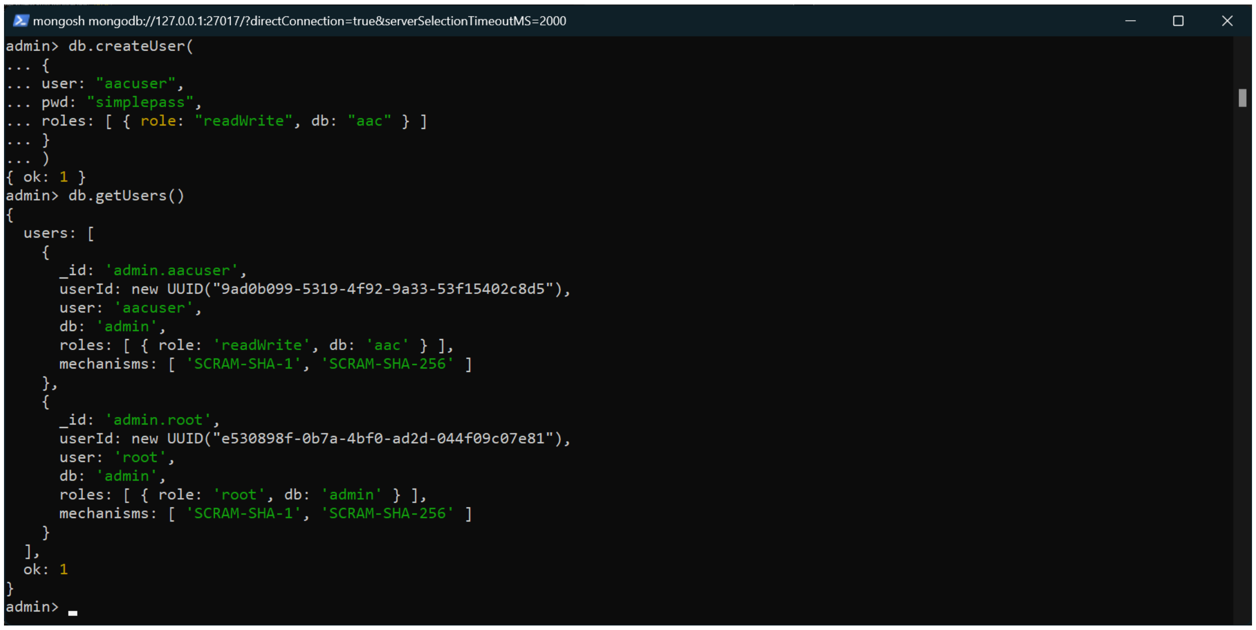
Screenshots

- Import the data:



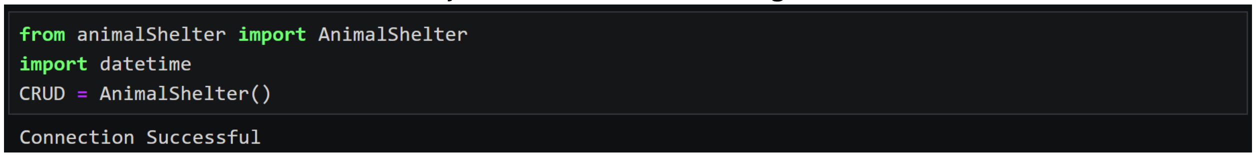
```
brystonjensen_snhu@78e20b13519a:/usr/local/datasets$ mongoimport --username="${MONGO_USER}" --password="${MONGO_PASS}" -
-port=${MONGO_PORT} --host=${MONGO_HOST} --db=AAC --collection=animals --authenticationDatabase=admin --type=csv --heade
rline --drop ./aac_shelter_outcomes.csv
2024-02-07T22:07:55.444+0000    connected to: mongodb://localhost:27017/
2024-02-07T22:07:55.444+0000    dropping: AAC.animals
2024-02-07T22:07:55.627+0000    10000 document(s) imported successfully. 0 document(s) failed to import.
```

- Add a user to ensure user authentication to the database and collection:



```
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
admin> db.createUser(
... {
... user: "aacuser",
... pwd: "simplepass",
... roles: [ { role: "readWrite", db: "aac" } ]
... }
... )
{ ok: 1 }
admin> db.getUsers()
{
  users: [
    {
      _id: 'admin.aacuser',
      userId: new UUID("9ad0b099-5319-4f92-9a33-53f15402c8d5"),
      user: 'aacuser',
      db: 'admin',
      roles: [ { role: 'readWrite', db: 'aac' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    },
    {
      _id: 'admin.root',
      userId: new UUID("e530898f-0b7a-4bf0-ad2d-044f09c07e81"),
      user: 'root',
      db: 'admin',
      roles: [ { role: 'root', db: 'admin' } ],
      mechanisms: [ 'SCRAM-SHA-1', 'SCRAM-SHA-256' ]
    }
  ],
  ok: 1
}
admin> 
```

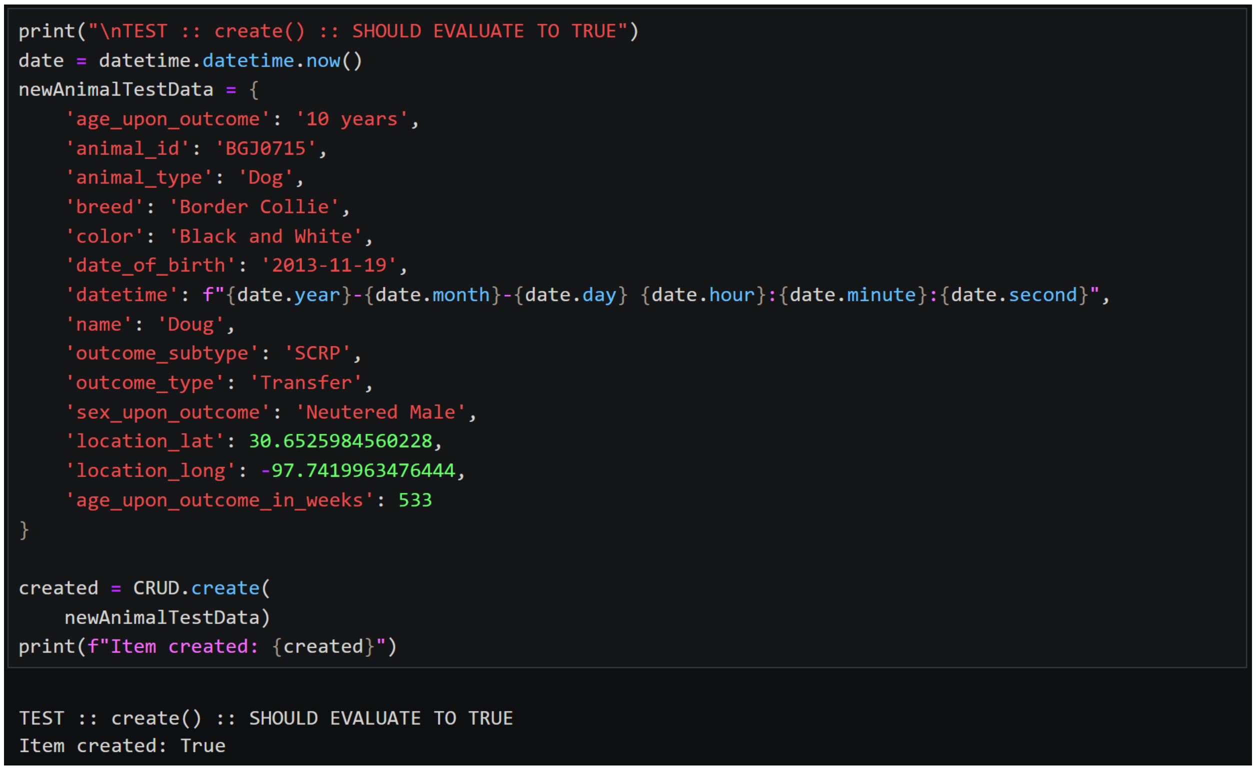
- Create an AnimalShelter object and Initialize the MongoDB server:



```
from animalShelter import AnimalShelter
import datetime
CRUD = AnimalShelter()

Connection Successful
```

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

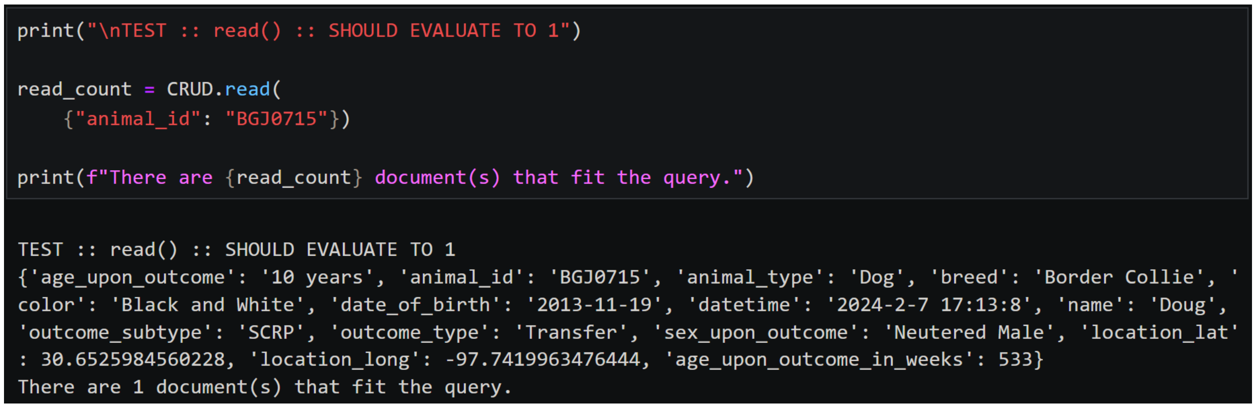
- Test the create method:



```
print("\nTEST :: create() :: SHOULD EVALUATE TO TRUE")
date = datetime.datetime.now()
newAnimalTestData = {
    'age_upon_outcome': '10 years',
    'animal_id': 'BGJ0715',
    'animal_type': 'Dog',
    'breed': 'Border Collie',
    'color': 'Black and White',
    'date_of_birth': '2013-11-19',
    'datetime': f"{date.year}-{date.month}-{date.day} {date.hour}:{date.minute}:{date.second}",
    'name': 'Doug',
    'outcome_subtype': 'SCRP',
    'outcome_type': 'Transfer',
    'sex_upon_outcome': 'Neutered Male',
    'location_lat': 30.6525984560228,
    'location_long': -97.7419963476444,
    'age_upon_outcome_in_weeks': 533
}

created = CRUD.create(
    newAnimalTestData)
print(f"Item created: {created}")
```

```
TEST :: create() :: SHOULD EVALUATE TO TRUE
Item created: True
```

- Test the read method:



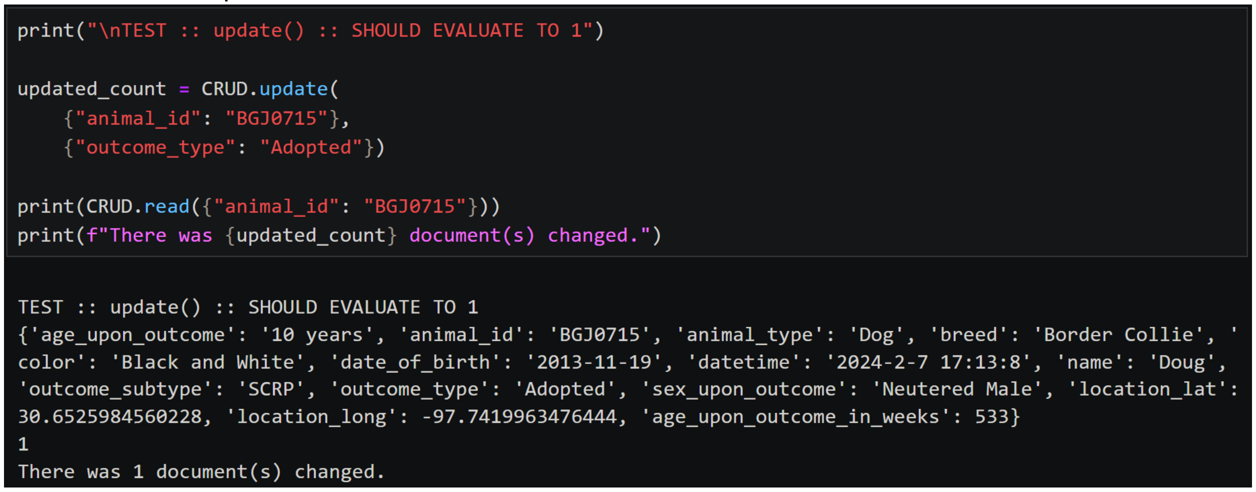
```
print("\nTEST :: read() :: SHOULD EVALUATE TO 1")

read_count = CRUD.read(
    {"animal_id": "BGJ0715"})

print(f"There are {read_count} document(s) that fit the query.")
```

```
TEST :: read() :: SHOULD EVALUATE TO 1
{'age_upon_outcome': '10 years', 'animal_id': 'BGJ0715', 'animal_type': 'Dog', 'breed': 'Border Collie', '
color': 'Black and White', 'date_of_birth': '2013-11-19', 'datetime': '2024-2-7 17:13:8', 'name': 'Doug',
'outcome_subtype': 'SCRP', 'outcome_type': 'Transfer', 'sex_upon_outcome': 'Neutered Male', 'location_lat'
: 30.6525984560228, 'location_long': -97.7419963476444, 'age_upon_outcome_in_weeks': 533}
There are 1 document(s) that fit the query.
```

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

Bryce Jensen
CS-340-Q7708
5-1: README
2/7/2024

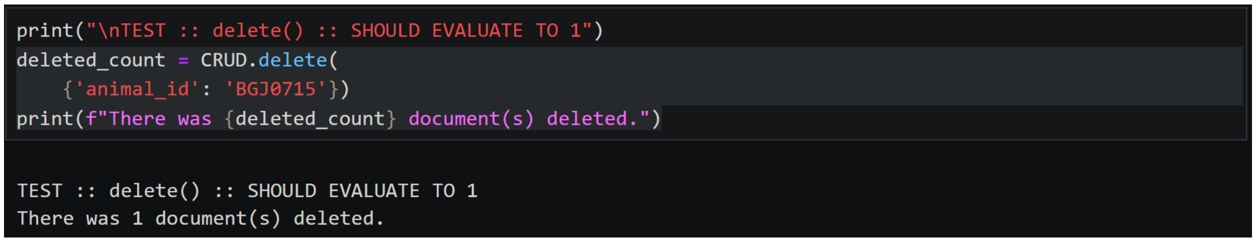
- Test the update method:



```
print("\nTEST :: update() :: SHOULD EVALUATE TO 1")

updated_count = CRUD.update(
    {"animal_id": "BGJ0715"},
    {"outcome_type": "Adopted"})

print(CRUD.read({"animal_id": "BGJ0715"}))
print(f"There was {updated_count} document(s) changed.")
```

```
TEST :: update() :: SHOULD EVALUATE TO 1
{'age_upon_outcome': '10 years', 'animal_id': 'BGJ0715', 'animal_type': 'Dog', 'breed': 'Border Collie', '
color': 'Black and White', 'date_of_birth': '2013-11-19', 'datetime': '2024-2-7 17:13:8', 'name': 'Doug',
'outcome_subtype': 'SCRP', 'outcome_type': 'Adopted', 'sex_upon_outcome': 'Neutered Male', 'location_lat':
30.6525984560228, 'location_long': -97.7419963476444, 'age_upon_outcome_in_weeks': 533}
1
There was 1 document(s) changed.
```

- Test the delete method:



```
print("\nTEST :: delete() :: SHOULD EVALUATE TO 1")
deleted_count = CRUD.delete(
    {'animal_id': 'BGJ0715'})
print(f"There was {deleted_count} document(s) deleted.")
```

```
TEST :: delete() :: SHOULD EVALUATE TO 1
There was 1 document(s) deleted.
```

Contact

Bryce Jensen

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.