

Ihab Elrayah  
12/2/23  
CS-340

## About the Project/ CS 340 README

### 5-1 Project One Submission

#### Motivation

This Project program was great practice as a test to improve my database skills for future career work, specifically in data manipulation. I selected Python for its easy sync with MongoDB. Python's compatibility with MongoDB and its many tutorials and easy syntax, particularly in Jupyter Notebook with its built-in compiler, made it an ideal choice for this project.

#### Getting Started

- In Mongo import the csv file aac\_shelter\_outcome.csv.
- Create a index to parse the data stored within the document.
- Authenticate a user would want to create both an Admin account and a aacuser account to access the database.
- Finally, a user would need to have access or install python and run the program out of a notebook.

#### Installation

- Python
- MongoDB

#### Usage

#### Code Examples

The command is used to import data into a MongoDB database using mongoimport. It specifies the credentials, database (enron), collection (emails), and imports data from the enron.json file, dropping the existing collection before the import.

```
(base) ihabelrayah_snhu@nv-snhu7-102:/usr/local/datasets$ mongoimport --username="${MONGO_USERNAME}" --password="${MONGO_PASS}" --port=${MONGO_PORT} --host=${MONGO_HOST} --db enron --collection emails --authenticationDatabase admin --drop ./enron.json
2023-12-16T21:31:57.205+0000 connected to: mongodb://nv-desktop-services.apporto.com:32237/
2023-12-16T21:31:57.206+0000 dropping: enron.emails
2023-12-16T21:31:57.575+0000 5929 document(s) imported successfully. 0 document(s) failed to import.
(base) ihabelrayah_snhu@nv-snhu7-102:/usr/local/datasets$
```

Entered mongosh then used the command AAC and created a simple index on the key "breed." with commands:

```
test> use AAC
switched to db AAC
AAC> db.animals.createIndex({breed: 1})
breed_1
AAC>
AAC> db.animals.getIndexes()
[ { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { breed: 1 }, name: 'breed_1' } ]
AAC> db.animals.find({breed: "Dog"}).explain("executionStats")
{
  explainVersion: '1',
  queryPlanner: {
    namespaces: [ 'AAC.animals' ],
    indexFilterSet: false,
    parsedQuery: { breed: { '$eq': 'Dog' } },
    queryHash: '1725800',
    planCacheKey: 'BF4F070',
    maxIndexedSolutionsReached: false,
    maxIndexedAndSolutionsReached: false,
    maxScansToExploredReached: false,
    winningPlan: {
      stage: 'FETCH',
      inputStages: {
        stage: 'IASCAN',
        keyPatterns: { breed: 1 },
        indexName: 'breed_1',
        isMultikey: false,
        multikeyPaths: [ { breed: [] } ],
        isUnique: false,
        isSparse: false
      }
    }
  }
}
```

Create the compound index on the "breed" and "outcome\_type" keys in your collection, Perform a query filtering by "breed" and "outcome\_type" of "Transfer" with commands

```

}
AAC> db.animals.createIndex({breed: 1, outcome_type:1})
breed_1_outcome_type_1
AAC> db.animals.getIndex()
TypeError: db.animals.getIndex is not a function
AAC> db.animals.getIndexes()
[
  { v: 2, key: { '_id': 1 }, name: '_id_' },
  { v: 2, key: { breed: 1 }, name: 'breed_1' },
  {
    v: 2,
    key: { breed: 1, outcome_type: 1 },
    name: 'breed_1_outcome_type_1'
  }
]

```

A method designed to efficiently insert and retrieve documents from a specified MongoDB database and collection.

```

In [21]: from crud_operations import CRUDOperations

username = "aacuser"
password = "your_password"
database_name = "AAC"
collection_name = "animals"

crud_instance = CRUDOperations()

data_to_insert = {"name": "Mr.White", "type": "Cat", "age": 3}
create_result = crud_instance.create_document(data_to_insert)
print(f"Create Operation Result: {create_result}")

read_query = {"type": "Cat"}
read_result = crud_instance.read_documents(read_query)
print(f"Read Operation Result: {read_result}")

Create Operation Result: Simulated create success
Read Operation Result: Simulated read result

```

An authentication method

```

from animalShelter import AnimalShelter
from bson.objectid import ObjectId

animals = AnimalShelter ("aacuser", "aacuser")

```

Created new entries, complete with a boolean indicator for successful outcomes.

```

print (animals.create ( {
    "age_upon_outcome" : "4 years",
    "animal_id" : "test",
    "animal_type" : "Cat",
    "breed" : "Domestic Shorthair Mix",
    "color" : "Black/White",
    "date_of_birth" : "2021-12-19",
    "datetime" : "2022-04-03 12:00:00",
    "monthyear" : "2022-04-03 12:00:00",
    "name" : "Binx",
    "outcome_subtype" : "SCRIP",
    "outcome_type" : "Transfer",
    "sex_upon_outcome" : "Neutered Male",
    "location_lat" : 15.2215,
    "location_long" : -80.300,
    "age_upon_outcome_in_weeks" : 15
}))
True

```

Contact  
Ihab Elrayah