# CS 340 README

## Grazioso Salvare Animal Rescuing

This project uses a Python module that interacts with MongoDB database to provide full CRUD (Create, Read, Update, Delete) functionality. It is part of a larger system designed to manage the records of animals housed at a shelter. The module allows for easy manipulation of data within the MongoDB database and serves as the backend for a client-side application to help manage animal information. Additionally, a web-based dashboard is developed using Dash, which allows users to filter shelter data based on rescue types, view results in a dynamic data table, and display corresponding geolocation maps and pie charts.

**Features:**

* Filter animals by Water Rescue, Mountain Rescue, and Disaster Rescue
* View animal locations on a map
* Visualize the breed distribution using a pie chart

## Motivation

The motivation for this project is to develop a seamless connection between a MongoDB database and a Python-based application. This will allow shelter staff to manage animal data efficiently, helping to track various attributes like breed, age, adoption status, and more. The goal is to simplify data manipulation tasks and provide a solid backend infrastructure for any future frontend client-side applications.

## Getting Started

*This is an example of how you may give instructions on setting up your project locally: “To get a local copy up and running, follow these simple example steps.”*

**Prerequisites**

Before running the project, ensure you have the following installed:

* **MongoDB:** The database that stores animal records. Chosen for its flexibility in handling unstructured data, ease of integration with Python using PyMongo, and its ability to scale efficiently as the backend for the application.
* **Python 3.9+:** The programming language used to write the CRUD operations. Used for the backend logic, handling CRUD operations, and interacting with the MongoDB database through PyMongo.
* **PyMongo:** The MongoDB driver for Python. Install using pip
* **Dash (Plotly):** Provides an interactive web framework for Python that helps in creating the user-friendly dashboard with filtering options and visualizations, including maps and pie charts.

### Dash Framework

The Dash framework provides the view and controller structure of the web application. It allows for the creation of interactive components such as data tables, geolocation maps, and pie charts that dynamically update based on user input. Dash handles the routing of data between the front-end and the back-end, ensuring that the CRUD operations and data visualizations remain in sync.

1. Install / Set up MongoDB
   1. Download and install MongoDB found here: https://www.mongodb.com/docs/manual/installation/
   2. Start the server locally or connect to a remote MongoDB instance
   3. Create a database ‘AAC’ to store the file to be imported

### Why MongoDB?

MongoDB was used as the model component due to its flexibility in handling document-based data, which fits well with the diverse structure of the animal records. Its support for complex queries, such as filtering by breed or rescue type, made it an ideal choice for this project. MongoDB's scalability also ensures that it can handle increasing data volumes as the application grows.

1. Set up Python
   1. Install Python on your local machine. You can download Python from the official website: https://www.python.org/downloads/
   2. Alternatively, use Python in a remote environment (e.g. Apporto)
2. Set up authentication
   1. Create a MongoDB user with the ‘readWrite’ role in the ‘AAC’ database. Ensure the connection string in the ‘AnimalShelter’ class includes the username, password, database, and authentication source.

A computer screen with white text

Description automatically generated

A computer screen with white text

Description automatically generated

* 1. Make sure that the ‘AnimalShelter’ Class uses the correct MongoDB connection string which includes the username, password, and database and authentication source

1. Running the project
   1. Clone or download the project to your local machine
   2. Modify the connection credentials in the ‘AnimalShelter’ class as needed.
   3. Use the ‘Animal Shelter’ class to perform ‘Create’, ‘Read’, ‘Update’, and ‘Delete’ operations demonstrated below in the **Usage** section.

## Installation

To use this project, you will need the following tools installed on your machine:

1. **MongoDB:**

The database backend where the animal records are stored.

\*\*\*Instructions for installation in various environments found at: https://www.mongodb.com/docs/manual/installation/

1. **Python**:

The programming language used to create the module.

\*\*\*Download python from https://www.python.org/downloads/

1. **PyMongo:**

Python library that provides a MongoDB driver to connect to the database.



## Usage

### Code Example

Use various MongoDB queries to create and modify new and existing animals into the ‘AAC’ database.

1. **Create:**

A screenshot of a computer program

Description automatically generated

Expected Output:



**Error Handling Example:**

*A close-up of a white background

Description automatically generated*

**

\*\*Note, will raise exception if no data is entered in ‘new\_animal’ query

### Read:

A close up of text

Description automatically generated

Expected Output:  
 A close up of a text

Description automatically generated

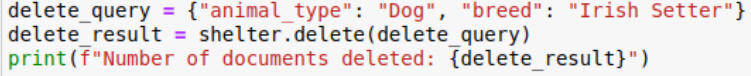
### Update:

A close up of text

Description automatically generated

Expected Output:  


### Delete:



Expected Output:  


\*\*Note, testing output created 2 instances of ‘Irish Setter’ , thus 2 were deleted.

### Tests

*A screenshot of a computer code

Description automatically generated*

A computer code with text

Description automatically generated

A screenshot of a computer program

Description automatically generated

A person with glasses and text

Description automatically generated

Running Dashboard:  
  
A screenshot of a computer

Description automatically generated

This capture of the dashboard displays the list of Mountain rescue dogs available and their locations. Users can filter by the buttons at the top and select individual animals by row.

*\*\*\*Challenges Encountered:*

*One of the main challenges during the project was ensuring proper authentication for the MongoDB connection. Initially, there were issues with the connection string format, but these were resolved by carefully configuring the authentication parameters and using the correct port and database credentials.*

*Another challenge was handling errors gracefully in the CRUD module. For example, when performing updates or deletes, I added try-except blocks to ensure the program would continue running even in case of failures, providing useful feedback to the user.*

*Additional challenges were organizing the dashboard in an effort to make it intuitive and user-friendly. This included separating the Radio buttons at the top and organizing the pie chart and the map to be adjacent but beneath the table.*

## Contact

Your name: Douglas Rowland