# CS 340 README Template

## About the Project/Project Title

This project is a Python-based module that implements full Create, Read, Update, and Delete (CRUD) operations for interacting with a MongoDB database. It serves as the backend data interface for Grazioso Salvare, an organization that identifies and trains dogs for rescue missions. The module allows developers to interact with animal shelter data by inserting, querying, updating, and removing records from the MongoDB collection.

Additionally, a Python dashboard was created using Dash and Dash Leaflet to provide a client-facing, interactive interface for visualizing and filtering the Austin Animal Center Outcomes data. This dashboard allows users to:

* View an unfiltered data table of shelter animals.
* Filter data by rescue type (Water Rescue, Mountain/Wilderness Rescue, Disaster/Individual Tracking).
* See a geolocation chart of selected animals.
* Display a secondary chart (optional) for additional insights.
* View a unique identifier and the Grazioso Salvare logo for branding.

## Motivation

The goal of this project is to provide a reusable, object-oriented Python interface to MongoDB for animal shelter data while also creating a user-friendly dashboard. This simplifies database access for developers, reduces errors for end users, and allows Grazioso Salvare to quickly identify and categorize animals suitable for search-and-rescue training.

## Getting Started

To get a local copy of this project up and running, follow these steps:

1. Install Python 3.x if it’s not already installed.
2. Set up a MongoDB database named AAC.
3. Create an authenticated user account (aacuser) with read/write permissions on the AAC database.
4. Clone or download the project folder (includes animal\_shelter.py and ProjectTwoDashboard.ipynb).
5. Install the required Python libraries:

pip install pymongo pandas numpy dash dash-leaflet plotly matplotlib jupyter-dash

1. Update the MongoDB credentials in animal\_shelter.py and in the dashboard notebook (ProjectTwoDashboard.ipynb).
2. Run ProjectTwoDashboard.ipynb in Jupyter Notebook to start the dashboard.

## Installation

Required Tools:

* Python 3.x
* MongoDB (local or cloud instance)
* Required Python packages (pymongo, pandas, dash, dash-leaflet, plotly, matplotlib, jupyter-dash)

## Usage

Here’s how you can use the module to perform Create, Read, Update, and Delete operations on the MongoDB database used by Grazioso Salvare.

### Code Example

from animal\_shelter import AnimalShelter

# instantiate the class with your MongoDB credentials

crud = AnimalShelter(“aacuser”, yourPassword)

# Create a new animal document

New\_animal = {

"name": "Jane Smith",  
 "breed": "Labrador Retriever",  
 "age\_upon\_outcome": "1 year",  
 "animal\_type": "Dog",  
 "outcome\_type": "Adoption"

}

insert\_result = crud.create(new\_animal)

print("Insert successful:", insert\_result)

# Read documents where age is 1 year

query\_filter = {"age\_upon\_outcome": "1 year"}

results = crud.read(query\_filter)

for document in results:

print(document)

**Dashboard Usage**

1. Open ProjectTwoDashboard.ipynb in Jupyter Notebook.
2. Run all cells to start the dashboard.
3. The dashboard will display:
   1. **Interactive data table** of all animals.
   2. **Geolocation map** showing the location of the selected animal.
   3. **Secondary chart** (optional).
   4. Footer with unique identifier: Jeremy Brown | SNHU CS-340 MongoDB Dashboard.

### Tests

from animal\_shelter import AnimalShelter

# Instatiate the class

crud = AnimalShelter()

# Test Create

new\_animal = {

"name": "Buddy",

"breed": "Golden Retriever",

"age\_upon\_outcome": "2 years",

"animal\_type": "Dog",

"outcome\_type": "Adoption"

}

insert\_result = crud.create(new\_animal)

print("Insert successful:", insert\_result)

# Test READ

query\_result = crud.read({"breed": "Golden Retriever"})

print("Number of records found:", len(query\_result))

for doc in query\_result[:2]: # Print first two for brevity

print(doc)

# Test UPDATE

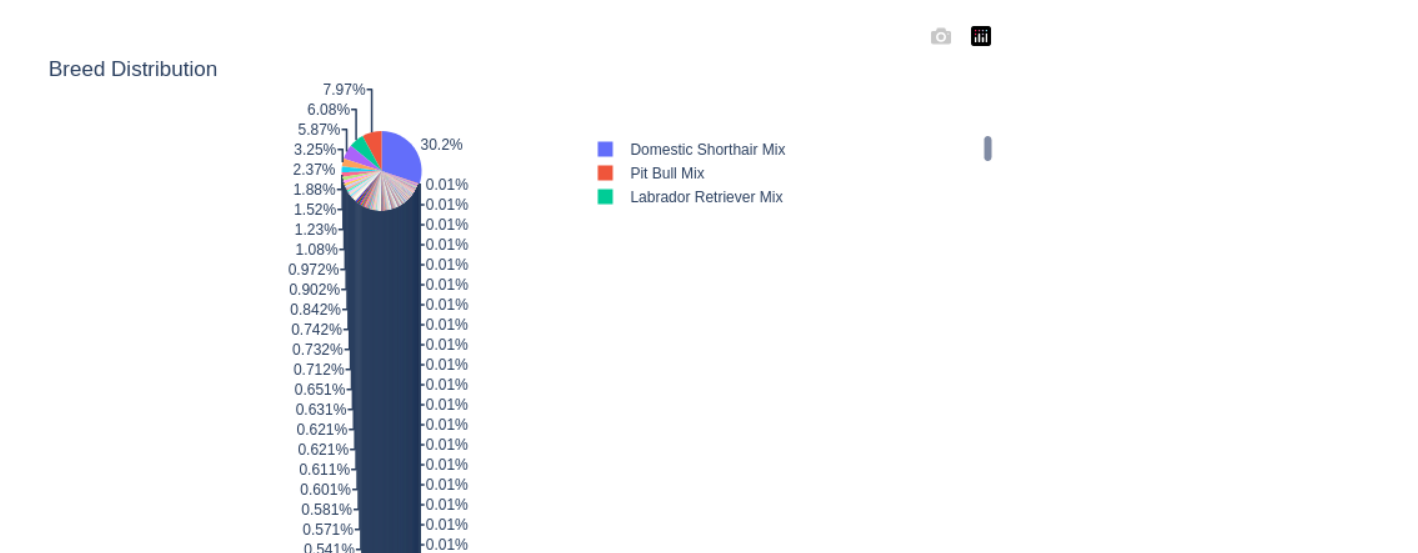
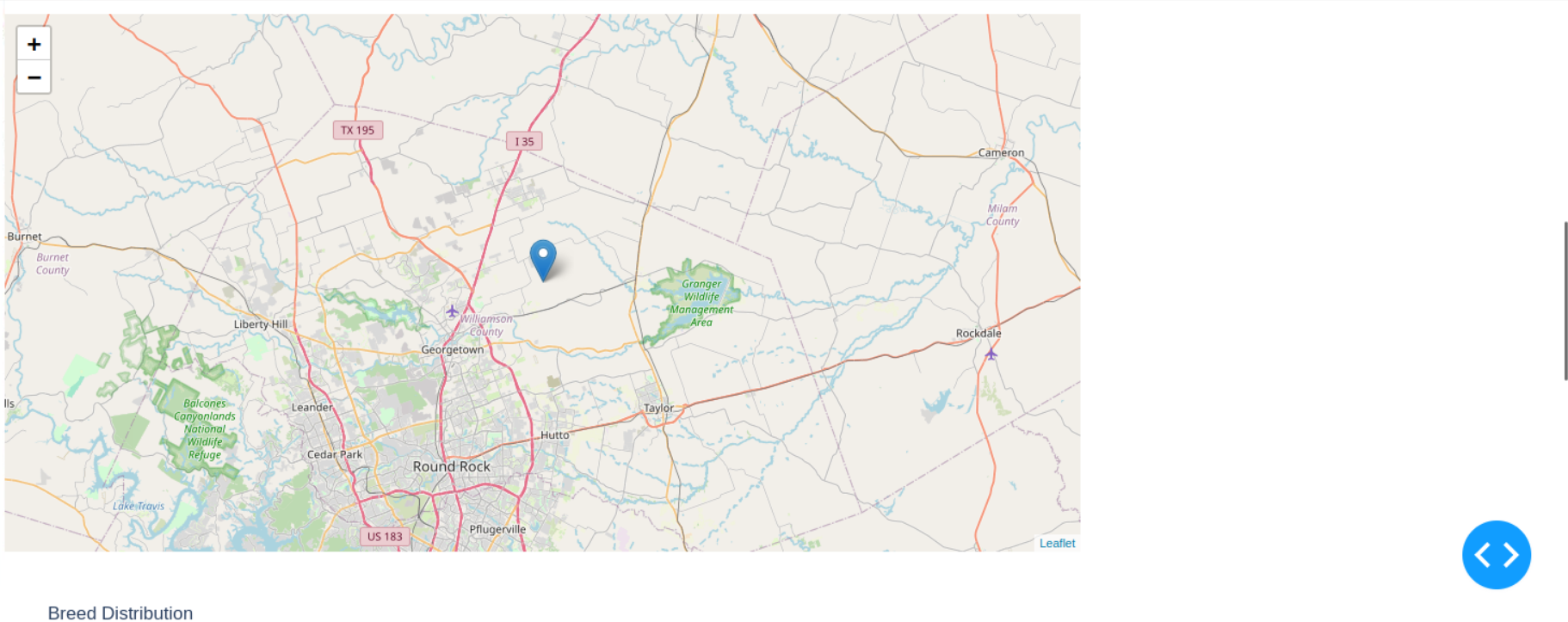
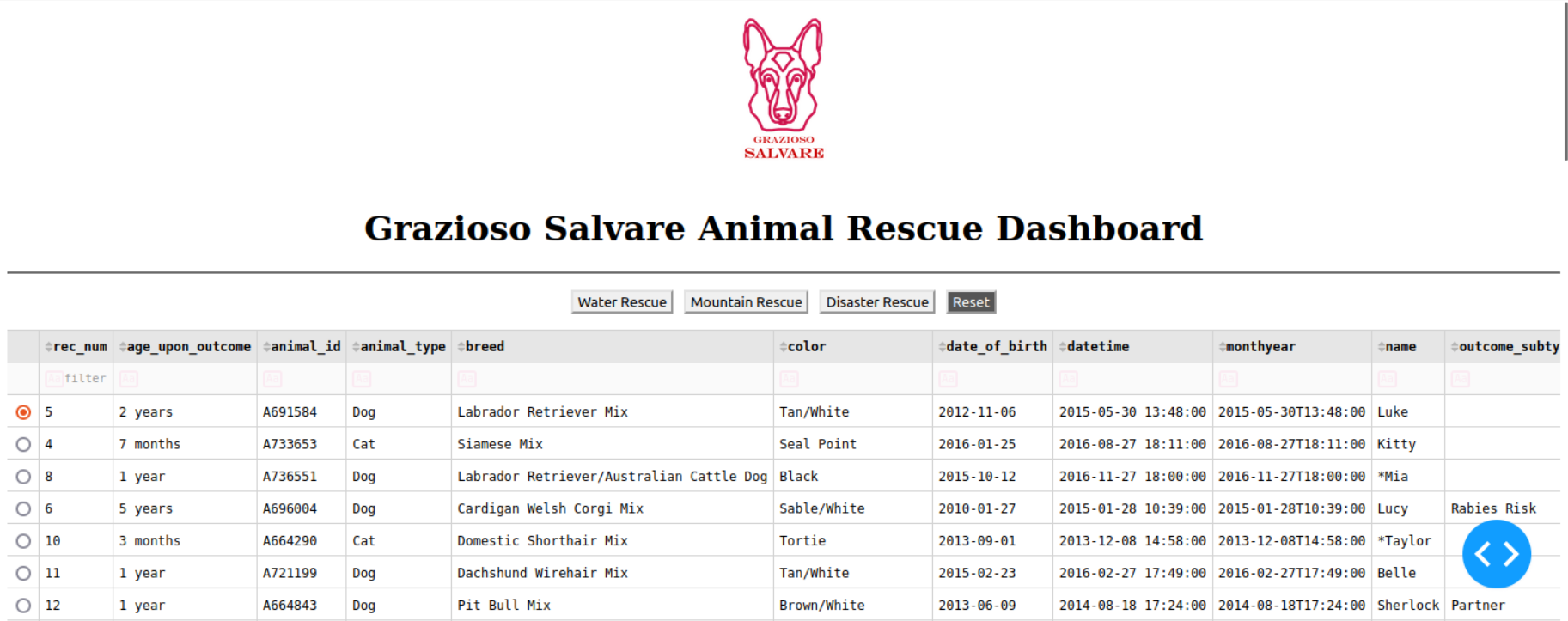
update\_result = crud.update({"name": "Buddy"}, {"age\_upon\_outcome": "3 years"}) print("Number of records updated:", update\_result)

# Test DELETE

delete\_result = crud.delete({"name": "Buddy"})

print("Number of records deleted:", delete\_result)

### Screenshots



## Contact

Jeremy Brown