# CS 340 README Template

## About the Project

Grazioso Salvare is an innovative international rescue-animal training company that identifies dogs suitable for search-and-rescue training. This software application can work with existing data from animal shelters to identify and categorize available dogs for training. The application includes a database.

## Motivation

Grazioso Salvare needs to identify dogs with specific profiles suitable for different types of rescue missions, such as water rescue, mountain rescue, disaster recovery, and scent tracking. The project aims to improve this process by using data from local animal shelters.

## Getting Started

Installation prerequisites:

* **MongoDB**: We need MongoDB to store and manage our data in a document-oriented database. You can download it from the [official MongoDB website](https://www.mongodb.com/docs/manual/installation/).
* **Python 3+**: Programming language for writing and executing our data processing scripts. You can download it from the [official Python website](https://www.python.org/downloads/).
* **Jupyter Notebook**: Use Jupyter Notebook to write and run our Python code, visualize data, and much more. You can install it using pip (after installing Python) by following the steps in the [official Jupyter website](https://jupyter.org/install).

Follow the Installation steps to setup the project.

## Installation

1. Clone the repository or download the files. This repository includes the following files:

* animal\_shelter.py
* ProjectTwoDashboard.ipynb
* Grazioso Salvare Logo.png

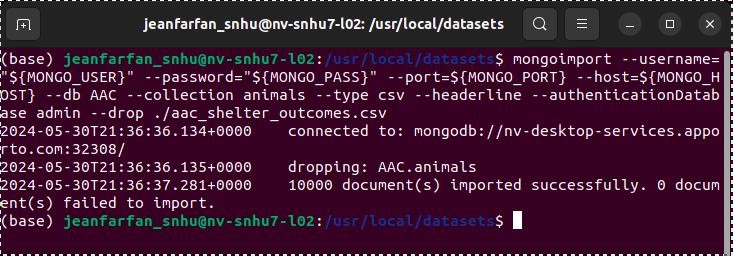
1. Install **pymongo** to connect our Python code to the MongoDB database and perform CRUD database operations from our scripts:

pip install pymongo

1. Import the ***aac\_shelter\_outcomes.csv*** file into MongoDB:

mongoimport –username=”${MONGO\_USER}” --password=”${MONGO\_PASS}” --port=${MONGO\_PORT} --host=${MONGO\_HOST} --db AAC --collection animals --type csv --headerline –authenticationDatabase admin --file path/to/aac\_shelter\_outcomes.csv

(Note: If the command throws an error, make sure that the .csv file directory is correct and that you are using the correct MongoDB user).



1. Edit the connection variables in ***animal\_shelter.py*** file to match your MongoDB:

USER = 'your\_user'

PASS = 'your\_password'

HOST = 'your\_host'

PORT = your\_port

DB = 'AAC'

COL = 'animals'

**Create a Mongo user with read and write privileges for our database (optional)**

1. Open the terminal and enter the MongoDB shell with the following command:

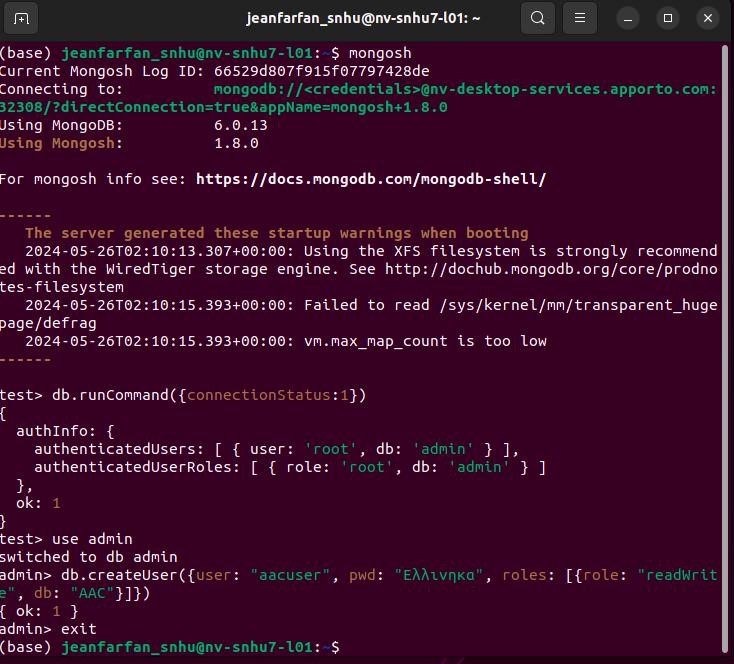
mongosh

1. Use admin database:

use admin

1. Create user:

db.createUser({user: “your\_username”, pwd: “your\_password”, roles: [{role: “readWrite”, db: “AAC”}]})



1. Exit MongoDB shell and change Mongo user in the terminal:

MONGO\_USER=your\_username

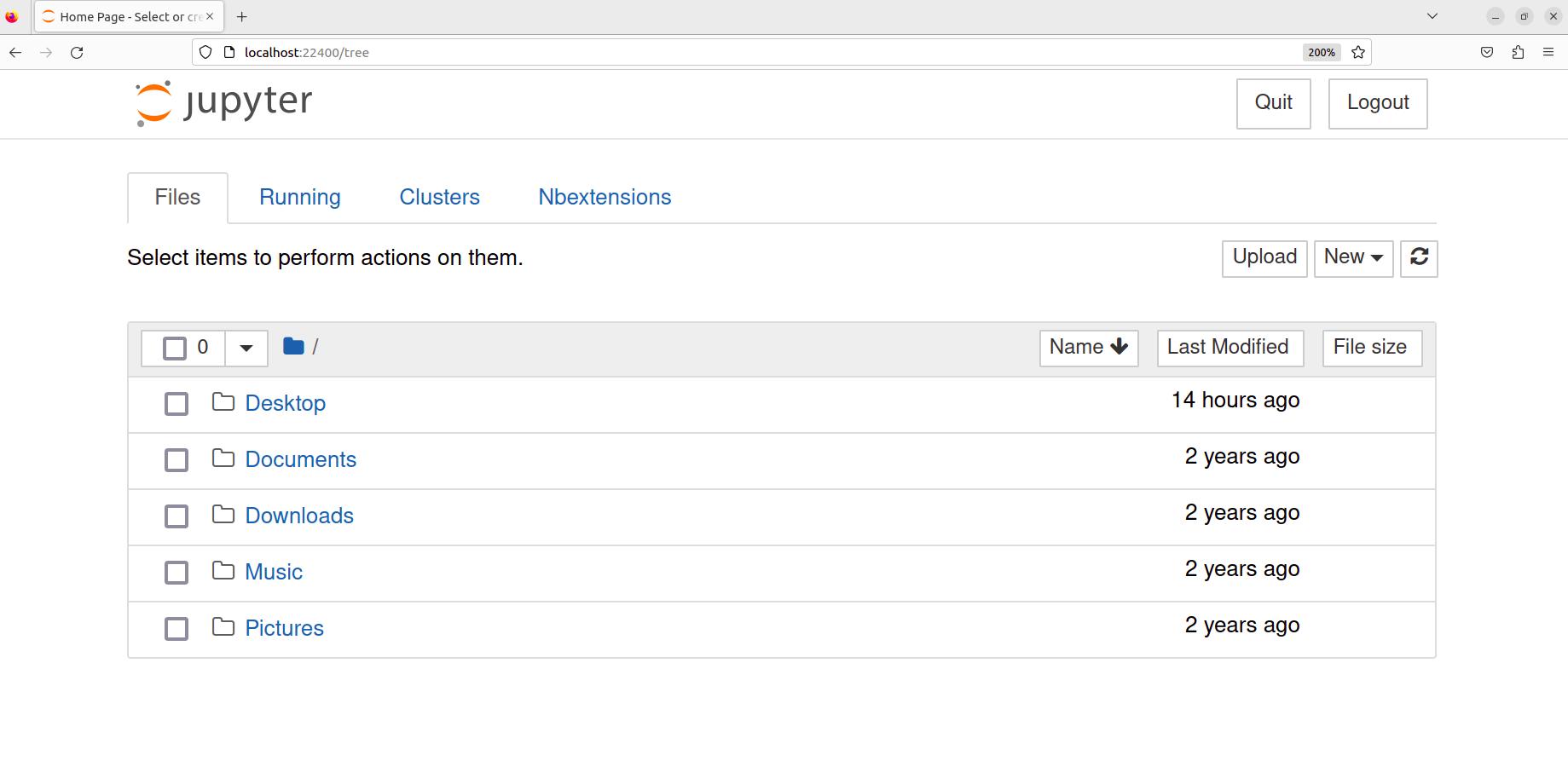
MONGO\_PASS=your\_password

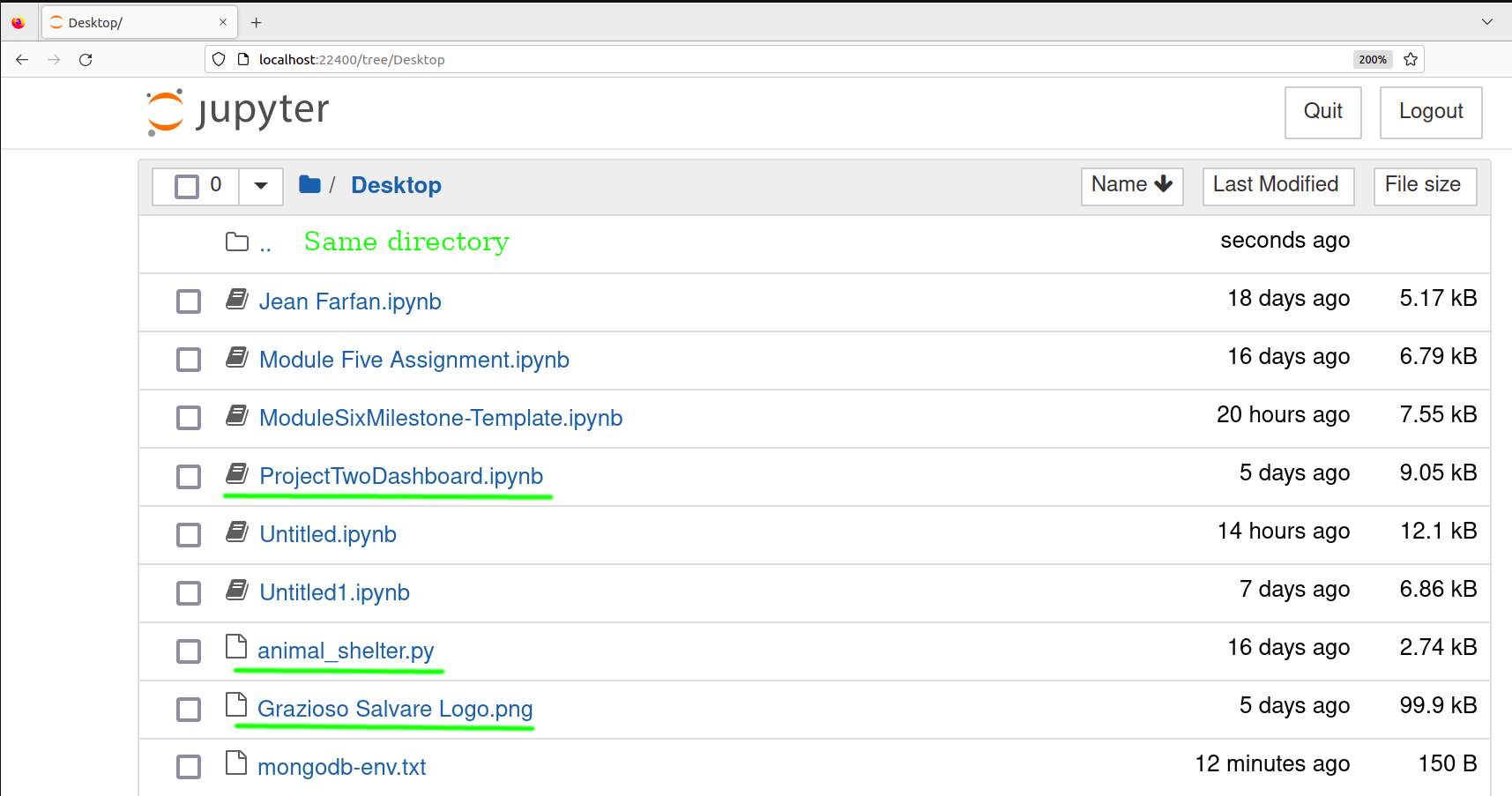
1. Try to open the MongoDB shell and execute the following command for verification:

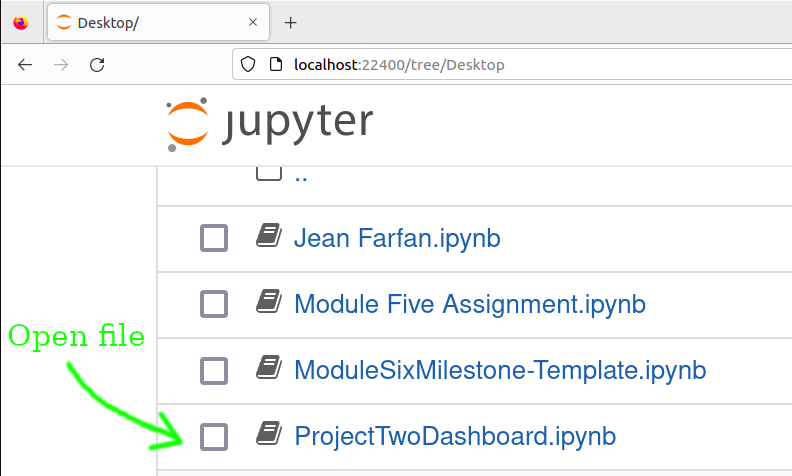
db.runCommand({connectionStatus:1})



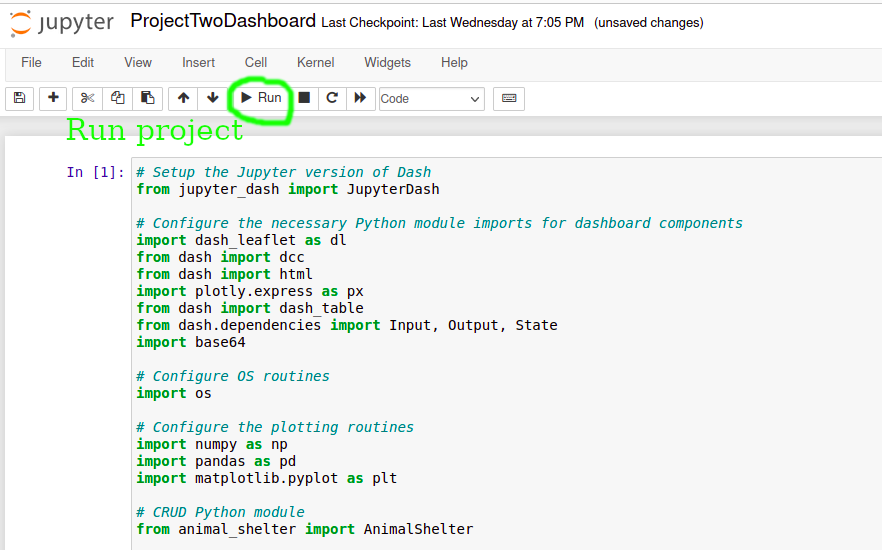
**Open dashboard**

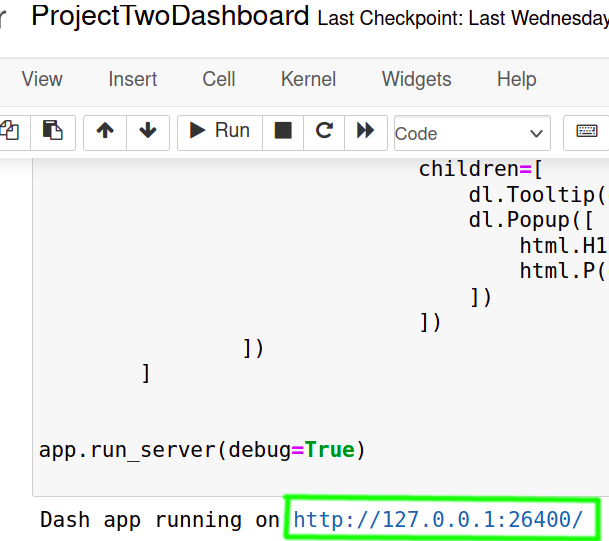
1. To open the dashboard, open Jupyter Notebook.
2. Search and open the ***ProjectTwoDashboard.ipynb*** file. Make sure that all the files (***ProjectTwoDashboard.ipynb***, ***Grazioso Salvare Logo.png***, and ***animal\_shelter.py***) are in the same directory.

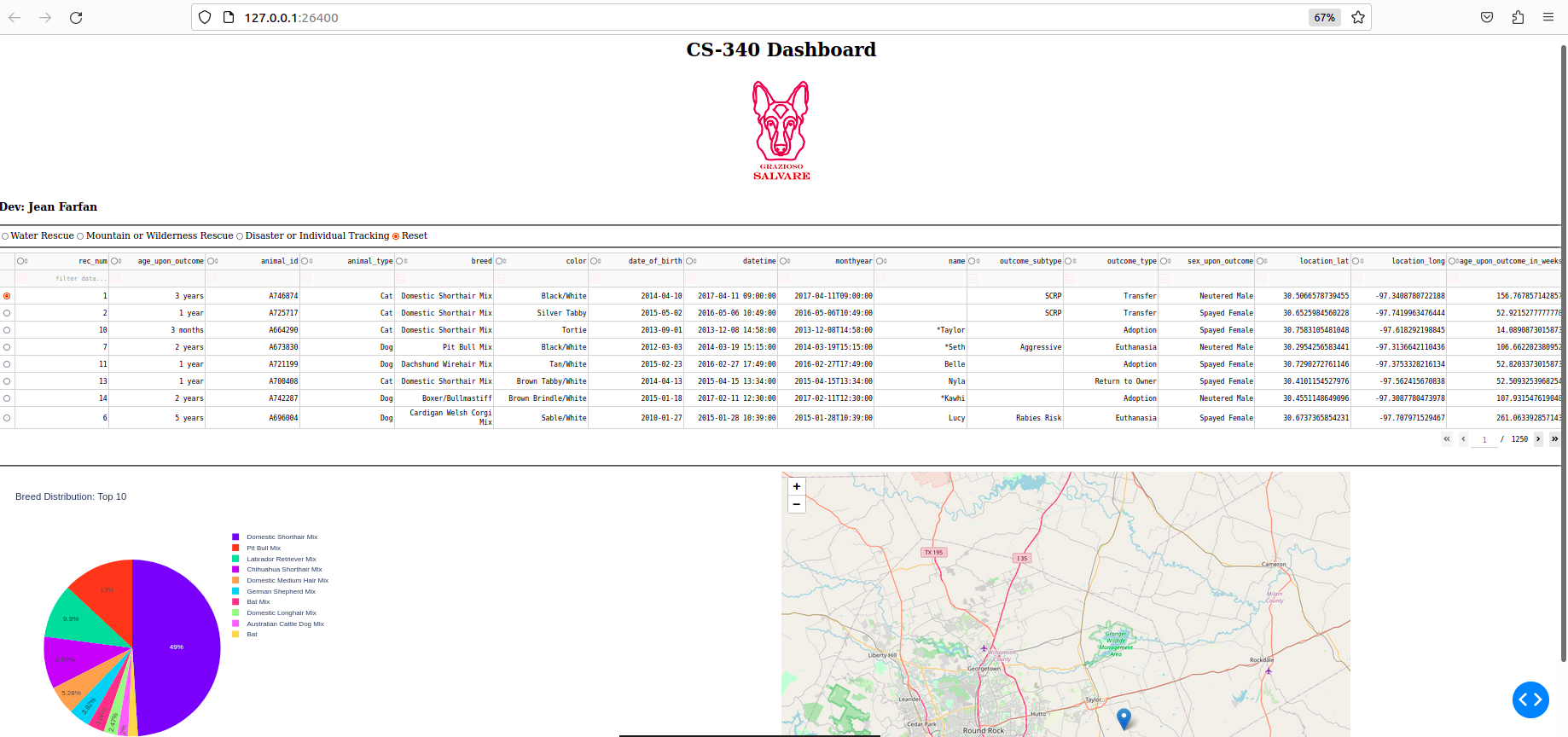




1. Run the file to show the dashboard. This will launch the dashboard on a local server at <http://127.0.0.1:8050/>.







## Usage

The dashboard developed for Grazioso Salvare uses data from local animal shelters to help identify dogs suitable for various rescue missions. This dashboard provides an interactive interface to filter and visualize the data. This dashboard allows users to filter dogs based on the type of rescue mission. Also, it displays the filtered data in an interactive data table and a pie chart for better analysis and decision-making. Finally, it supports CRUD (Create, Read, Update, and Delete) operations on the dog data stored in MongoDB.

Use interactive radio items to filter the data based on the type of rescue mission. The data table and charts will update dynamically to reflect the filtered data.

To perform Create, Read, Update, and Delete operations on the data, you need to write code in ***ProjectTwoDashboard.ipynb*** or in a new Python file. Ensure the MongoDB connection variables in animal\_shelter.py are correctly configured to match your MongoDB setup. Below is an example of how to use CRUD methods using the provided ***animal\_shelter.py***.

### Code Example

from animal\_shelter import AnimalShelter

shelter = AnimalShelter() # Create an instance of AnimalShelter

# Example animal document

animal = {

"age\_upon\_outcome": ‘99 years’,

"animal\_id": 'B666777',

"animal\_type": 'Cuy',

"breed": 'Domestic Shorthair Mix',

"color": 'Orange',

"date\_of\_birth": '1925-06-13',

"datetime": '2018-04-11 09:00:00',

"monthyear": '2019-04-11T09:00:00',

"name": 'Gamo',

"outcome\_subtype": 'SCRP',

"outcome\_type": 'Transfer',

"sex\_upon\_outcome": 'Neutered Male',

"location\_lat": 30.5066578739454,

"location\_long": -97.3408780722187,

"age\_upon\_outcome\_in\_weeks": 156.767857142856

}

# Create

# Create a new animal

print(shelter.create(animal))

# Find

# Read an animal

results = shelter.read({"animal\_id": "B666777"})

for k in results:

print(k)

# Update

# First, key/value to find

criteria = {"animal\_id" : "B666777"}

# Then, key/value to modify

updateData = {"age\_upon\_outcome": '77 years'}

# print modified count

print(shelter.update(criteria, updateData))

# Find the updated animals to see results

results = shelter.read({"animal\_id" : "B666777"})

# Print all animals with animal\_id B666777

for k in results:

print (k)

# Delete

# Delete animal with animal\_id B666777

print(shelter.delete({"animal\_id" : "B666777"}))

# Find deleted animals to see results

results = shelter.read({"animal\_id" : "B666777"})

# Print all animals with animal\_id B666777

for k in results:

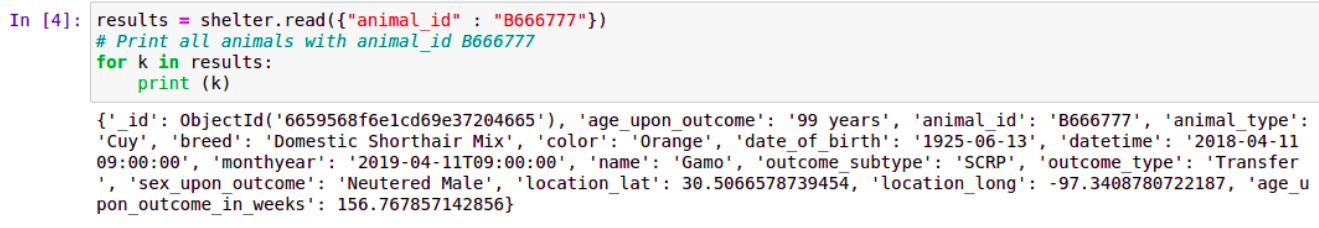
print (k)

### Tests

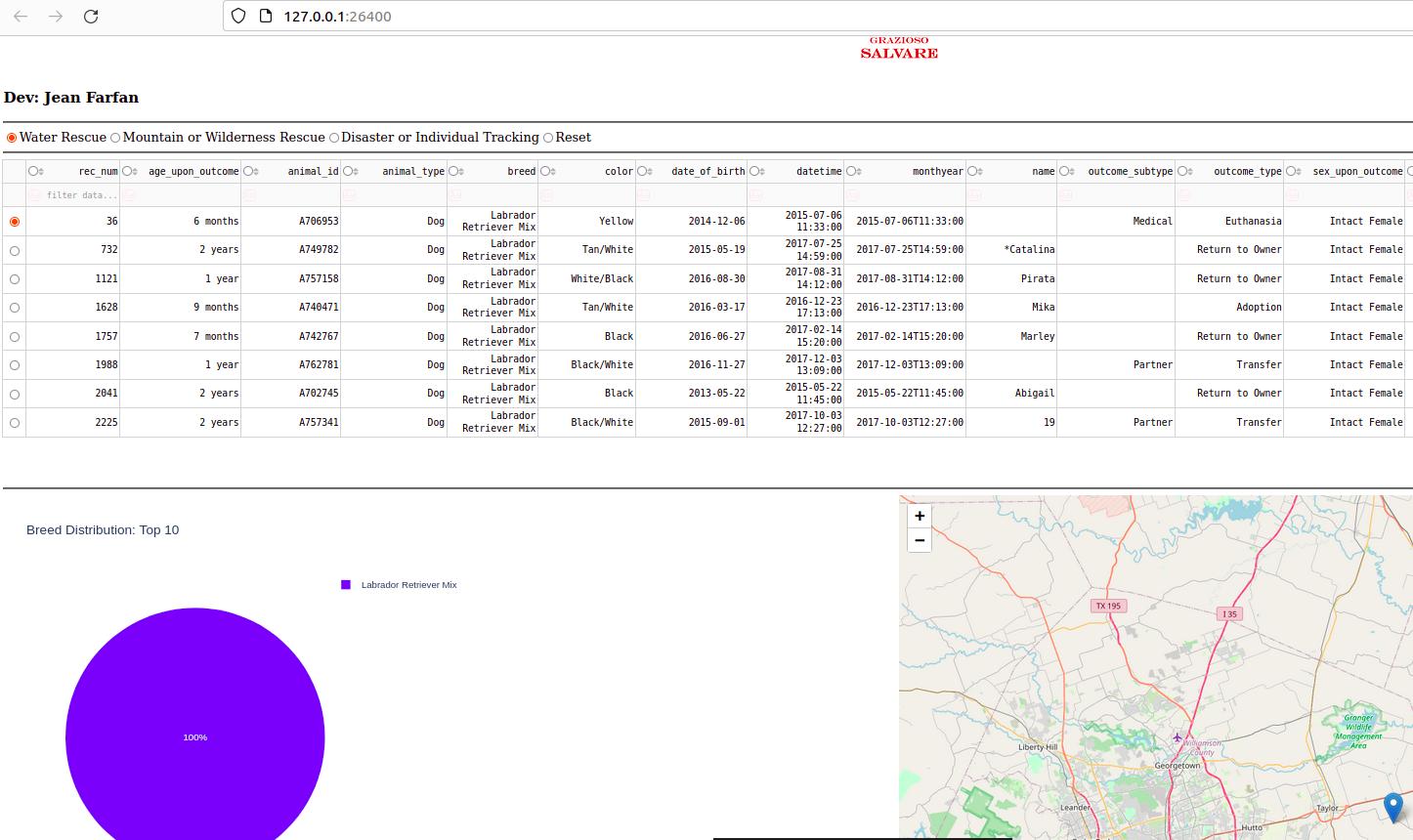
To validate the functionality, launch the dashboard and use the interactive widgets to filter data, and perform CRUD operations via the provided interface. This will ensure that the application meets the requirements for identifying and categorizing dogs suitable for various rescue missions.

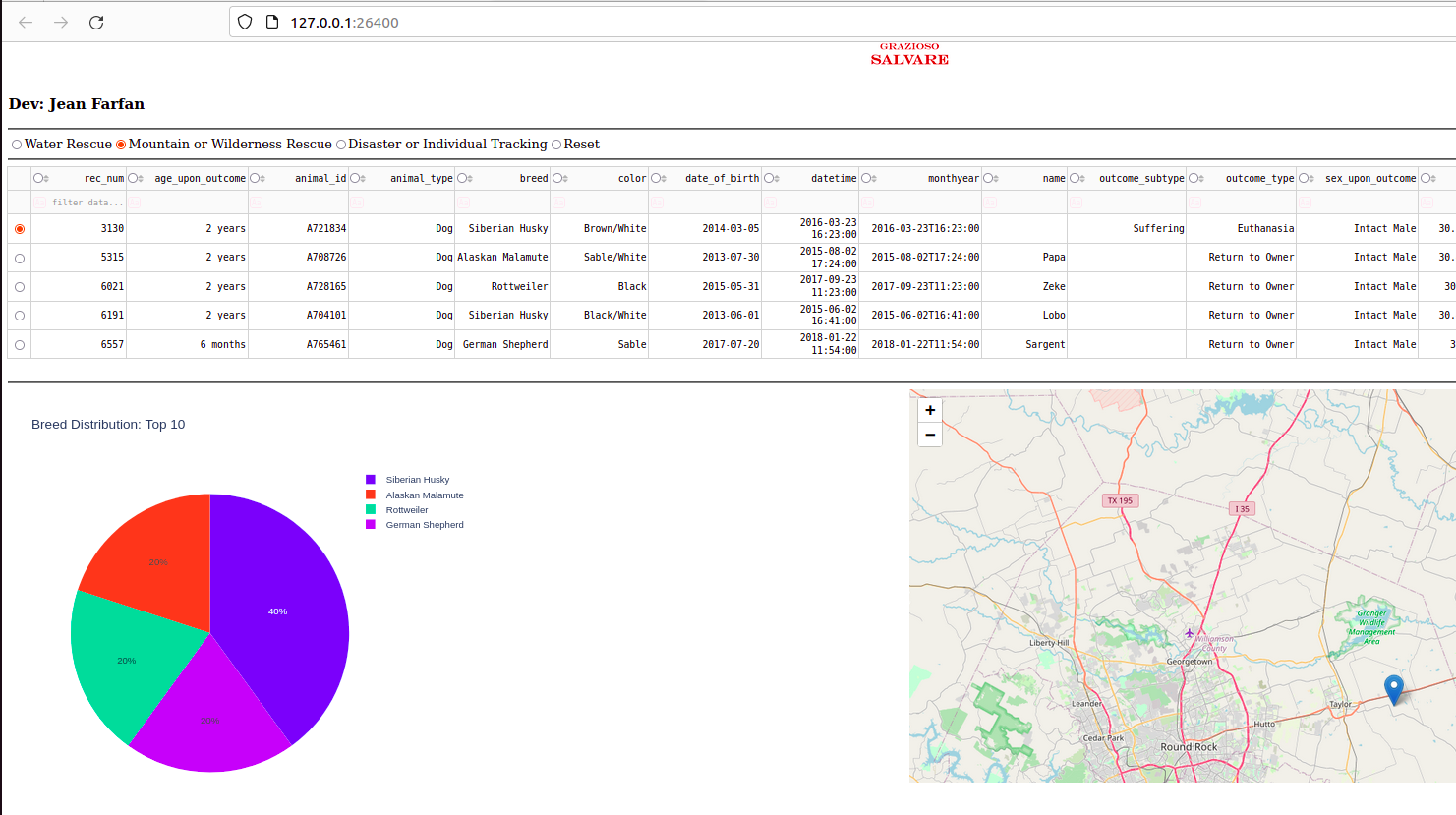
### Screenshots

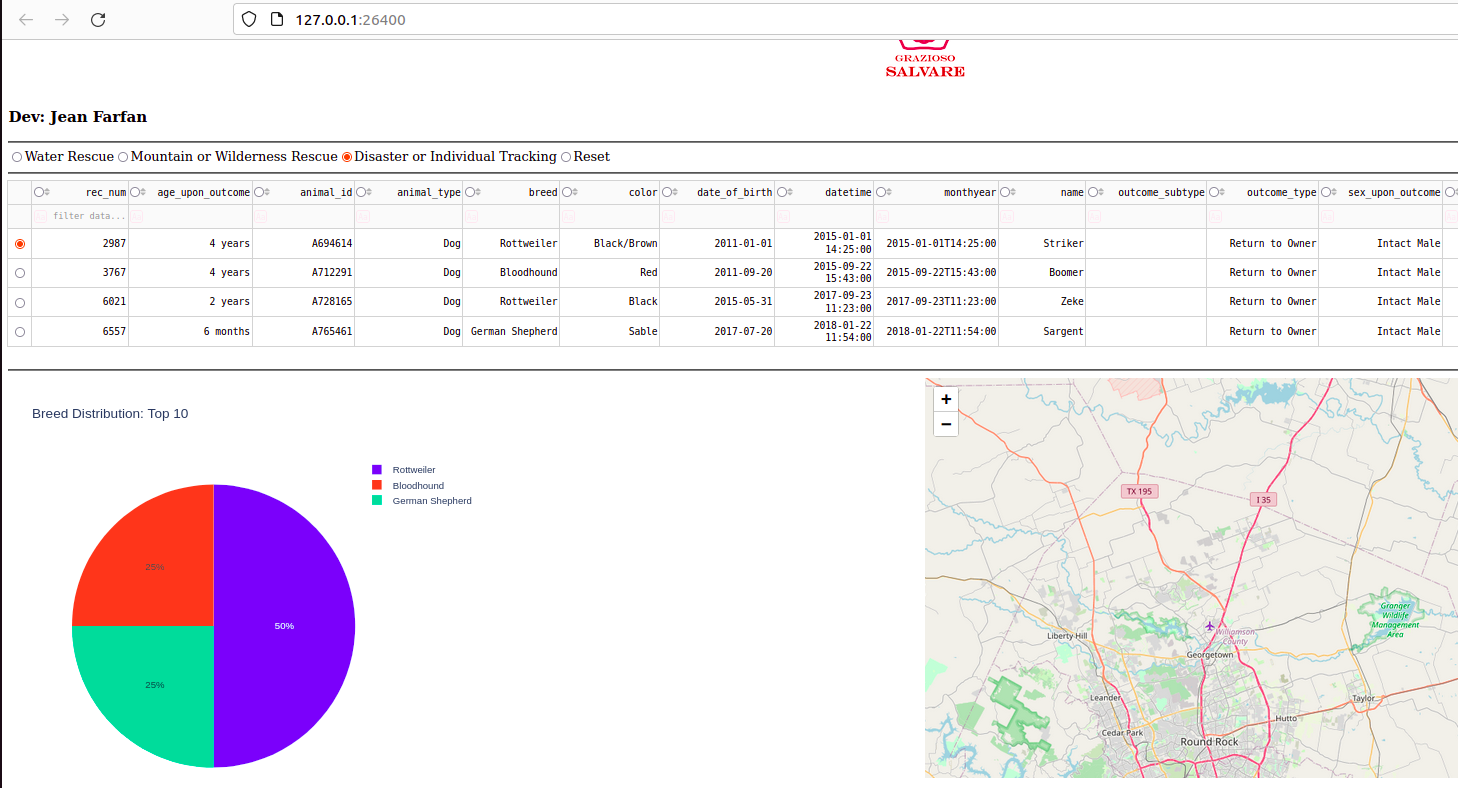


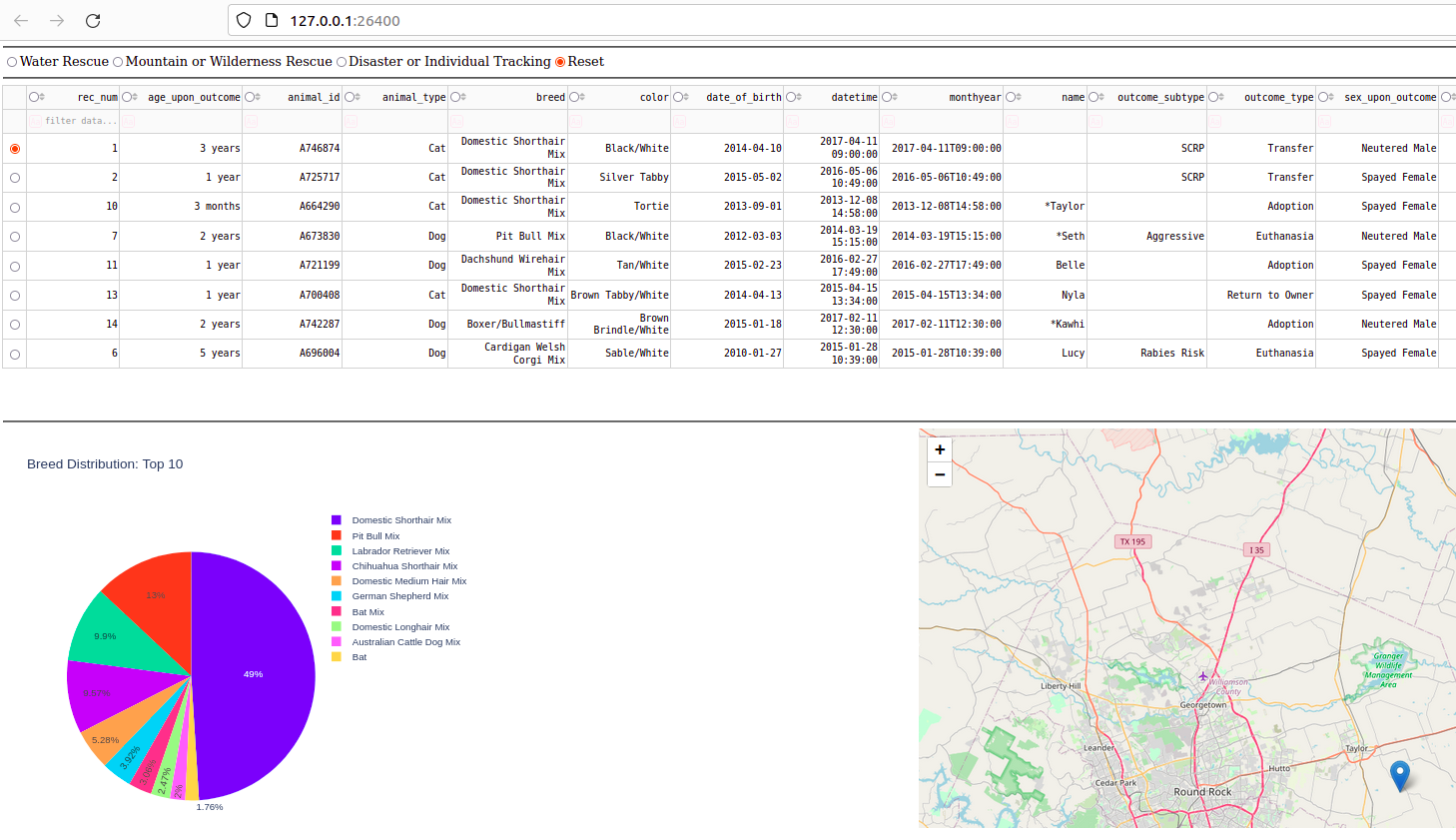


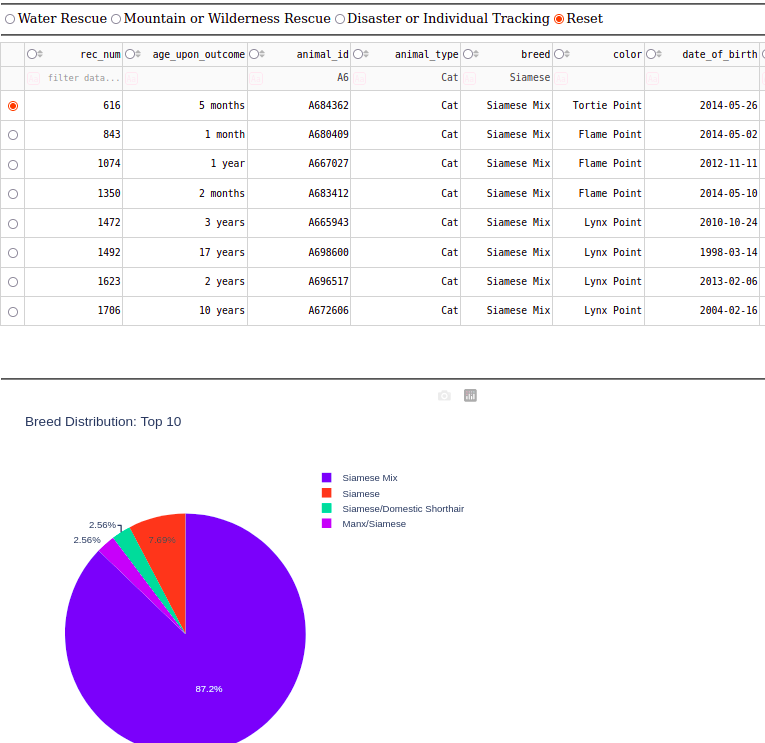












## Contact

Your name: Jean Farfan