# CS 340 Project Two

## About the Project/Project Title

*This project is a web application dashboard powered by MongoDB, designed for Grazioso Salvare, an international company specializing in rescue-animal training. The dashboard enables users to identify dogs that are suitable candidates for search-and-rescue training based on specific criteria. Key features of the application include interactive filtering, dynamic data tables, breed distribution charts, and geolocation mapping to visualize the locations of the dogs.*

## Motivation

*Grazioso Salvare collaborates with non-profit animal shelters to identify dogs suitable for specialized search-and-rescue training. The goal of this project is to simplify the process of identifying these dogs by offering an intuitive dashboard that filters and visualizes shelter data. This approach reduces manual effort and training time while enhancing the efficiency of selecting rescue candidates.*

## Getting Started

*To get a local copy of the project up and running, follow these steps.*

## Installation

***1. Prerequisites***

*Ensure you have the following tools installed:*

*Python 3.x*

*MongoDB (running on localhost:27017)*

*Dependencies:*

*Install the necessary libraries using pip:*

*pip install pymongo dash jupyter-dash dash-leaflet plotly pandas*

***2. Database Setup***

*Start your MongoDB server:*

*mongod*

*Import the sample dataset into MongoDB:*

*mongoimport --db AAC --collection animals --file aac\_shelter\_outcomes.json --jsonArray*

***3. Project Files***

*Ensure the following files are in the project directory:*

*AnimalShelter.py (CRUD module)*

*ProjectTwoDashboard.ipynb*

*Grazioso Salvare Logo.png (logo image)*

## Usage

*Follow these steps to run the project:*

***Run the Dashboard:***

*Open Jupyter Notebook and run the following in a cell:*

*app.run\_server(debug=True)*

***Access the Dashboard:***

*Open the following link in your web browser:*

*http://localhost:27023/*

### Code Example

***This class provides CRUD functionality for MongoDB.***

*from pymongo import MongoClient*

*class AnimalShelter:*

*def \_\_init\_\_(self):*

*self.client = MongoClient("mongodb://localhost:27023/AAC")*

*self.database = self.client['AAC']*

*def read(self, query):*

*return list(self.database.animals.find(query))*

***Filter Callback Example***

***This callback updates the data table based on the selected filter option.***

*@app.callback(*

*Output('datatable-id', 'data'),*

*[Input('filter-type', 'value')]*

*)*

*def update\_dashboard(filter\_type):*

*if filter\_type == 'Water':*

*query = {"breed": {"$in": ["Labrador Retriever Mix", "Chesapeake Bay Retriever", "Newfoundland"]}}*

*elif filter\_type == 'Mountain':*

*query = {"breed": {"$in": ["German Shepherd", "Alaskan Malamute", "Old English Sheepdog", "Siberian Husky", "Rottweiler"]}}*

*elif filter\_type == 'Disaster':*

*query = {"breed": {"$in": ["Doberman Pinscher", "German Shepherd", "Golden Retriever", "Bloodhound", "Rottweiler"]}}*

*else:*

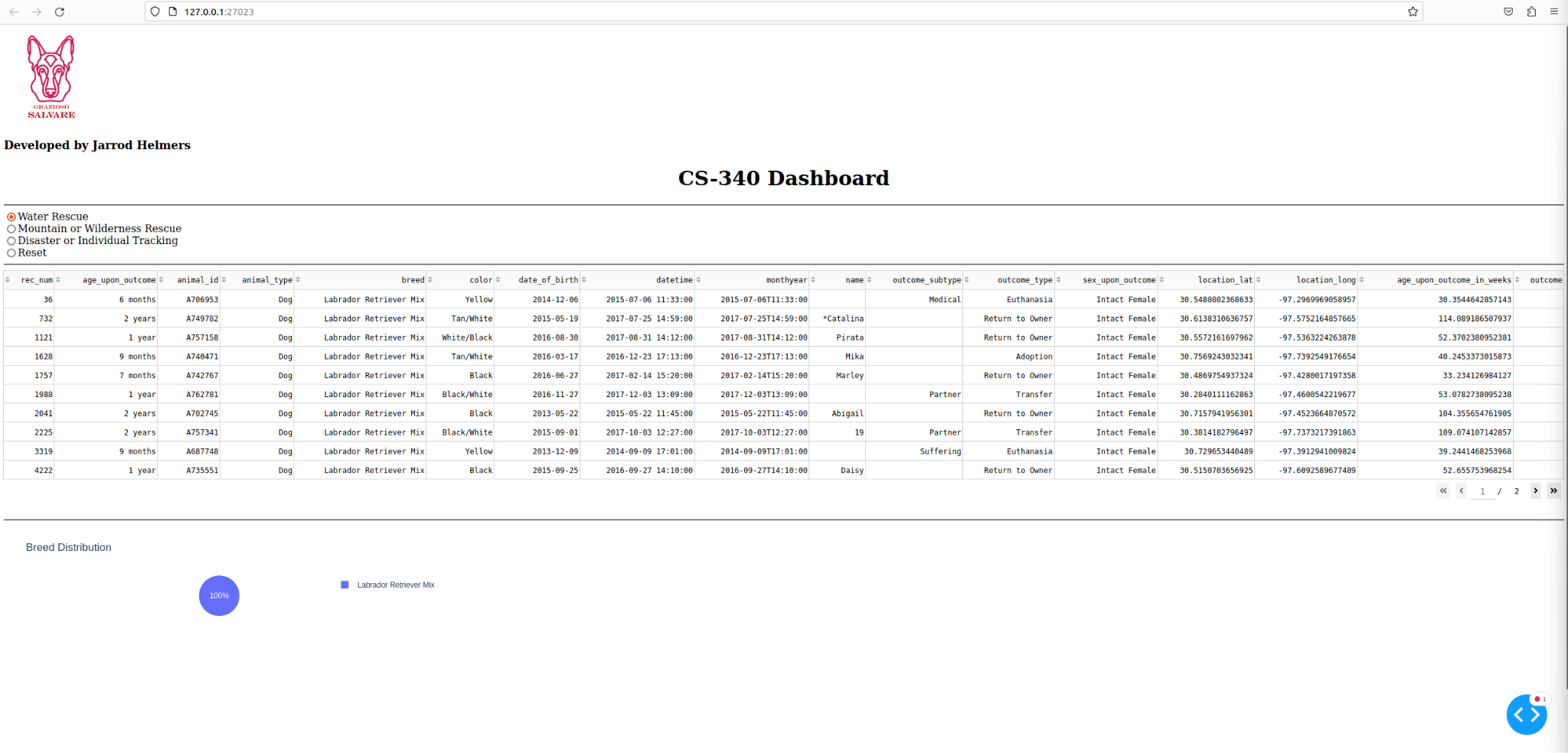
*query = {}*

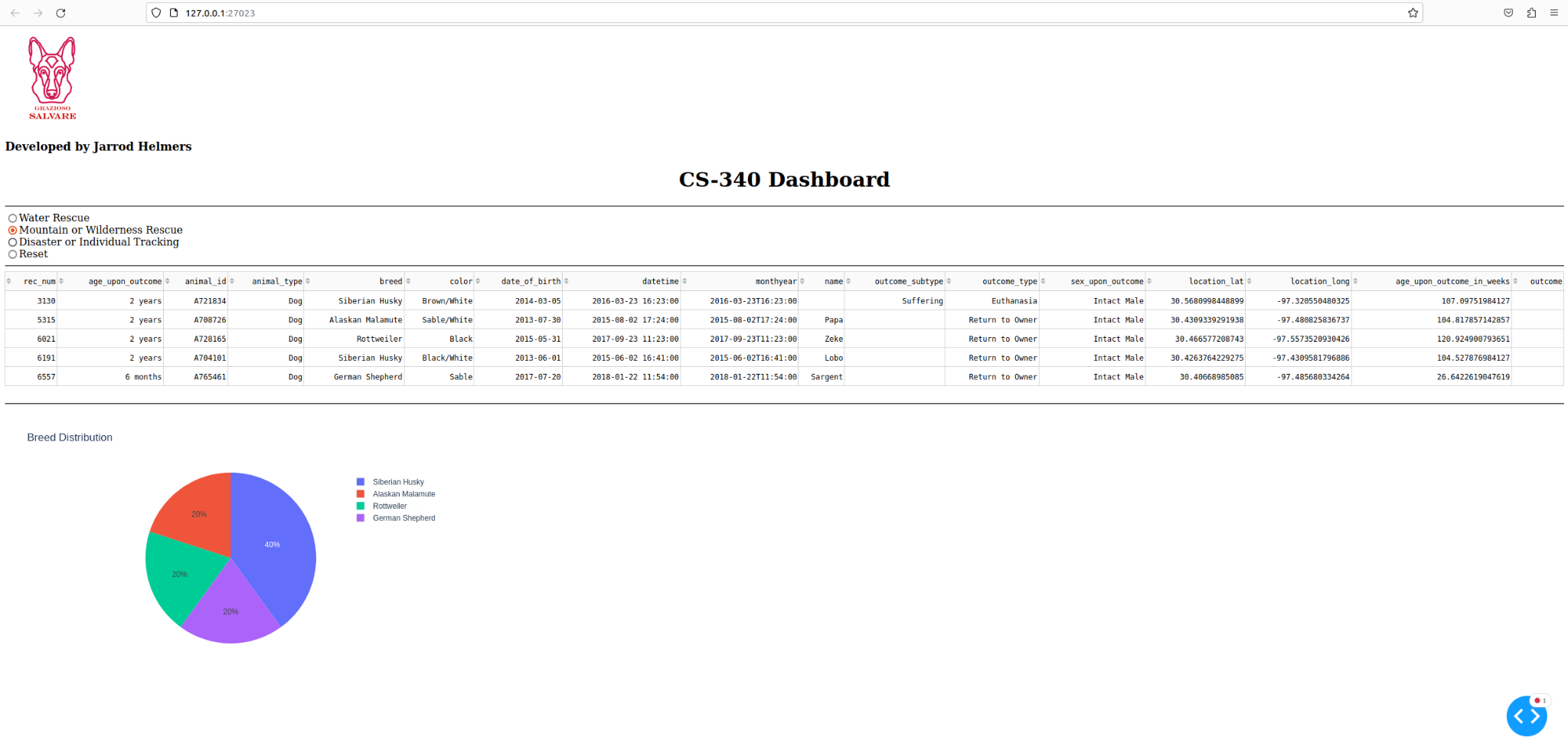
*return pd.DataFrame.from\_records(db.read(query)).drop(columns=['\_id']).to\_dict('records')*

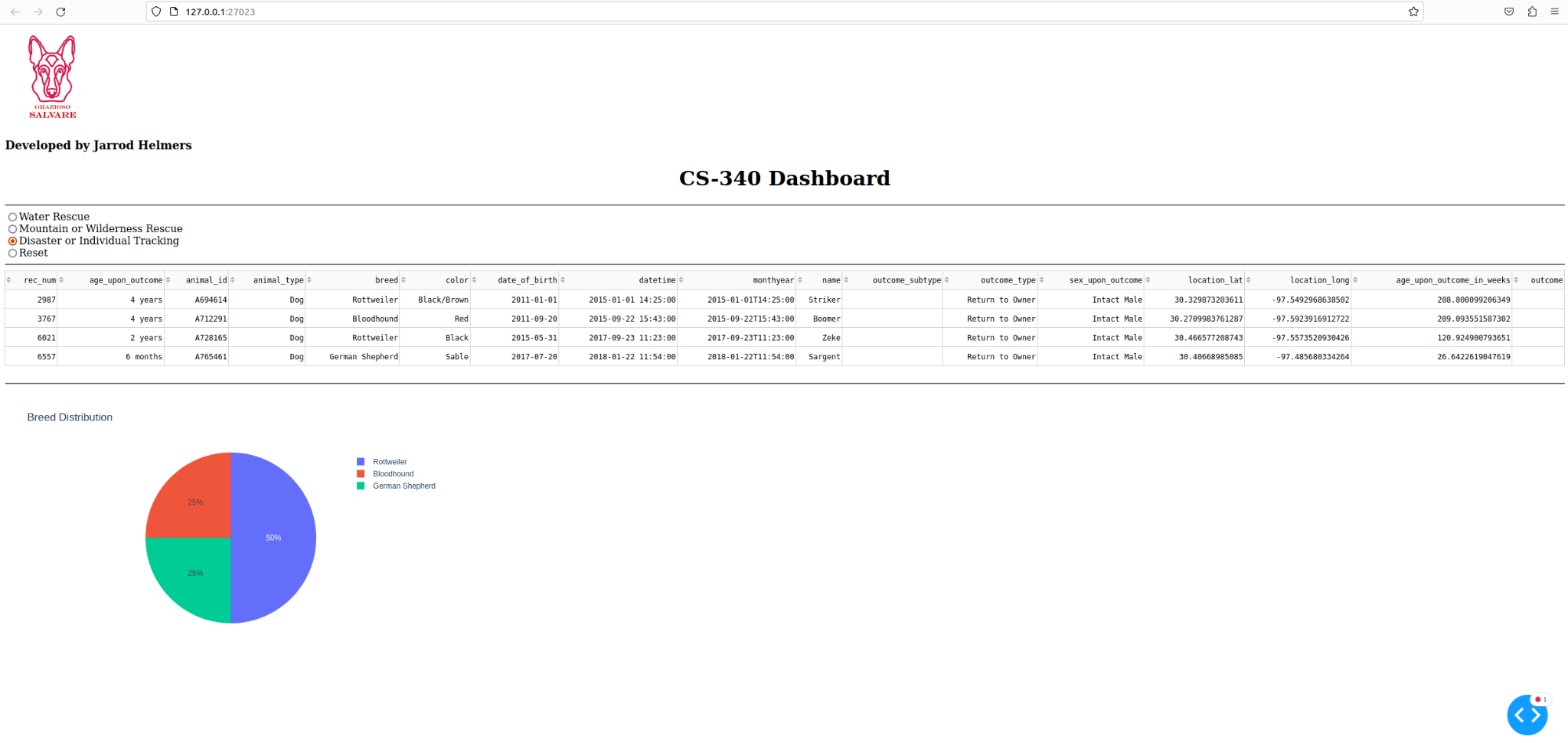
### Tests

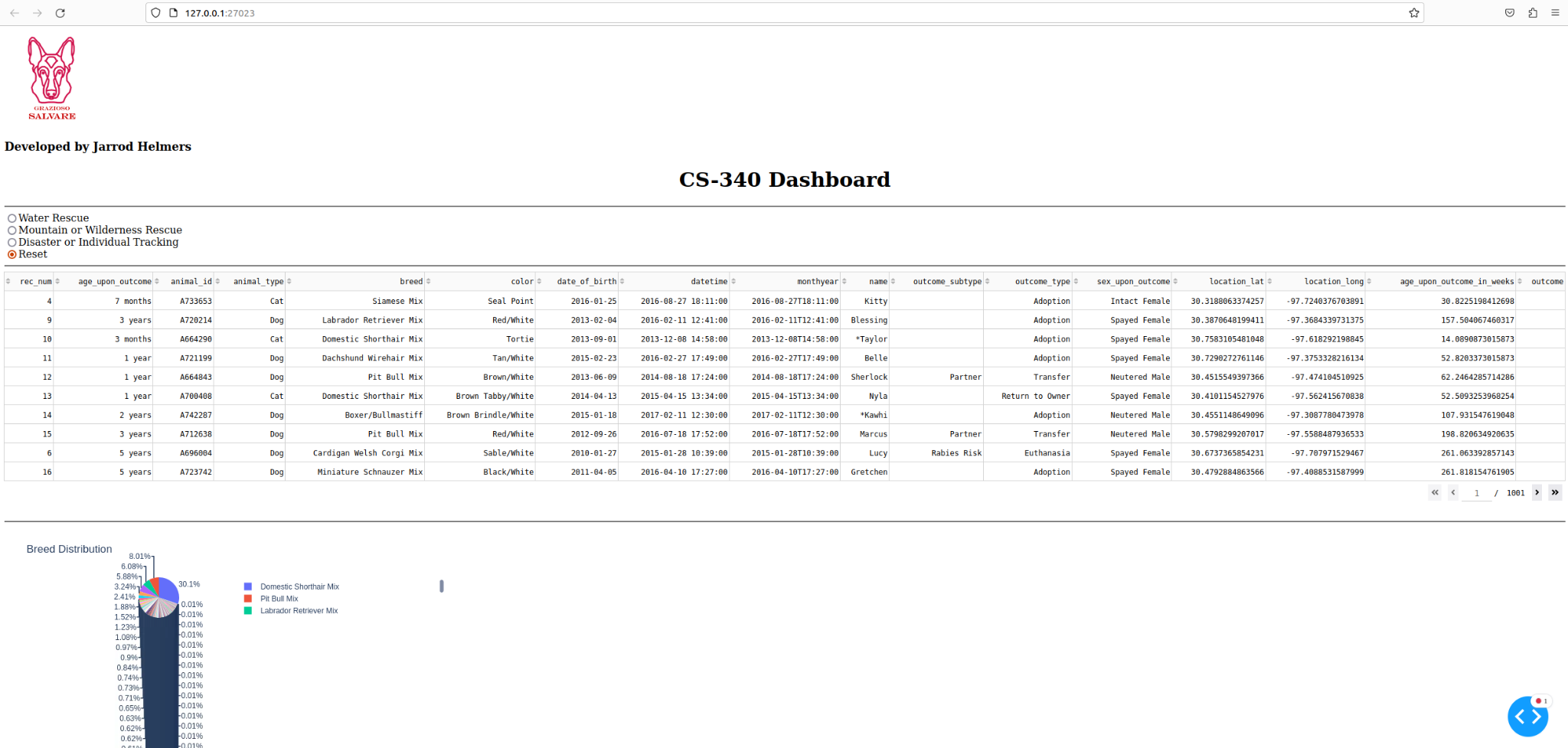
*You can test the filtering feature by selecting various options on the dashboard and confirming that the data table, pie chart, and map update accordingly.*

### Screenshots

**

**

**

**

### *Planned Features*

* ***Additional Charts****: Display more insights, such as age distribution and shelter statistics.*
* ***User Authentication****: Add login functionality for enhanced security.*
* ***Export Data****: Allow users to export filtered data as CSV or Excel.*

## Contact

Jarrod Helmers  
github: <https://github.com/JarrodHelmers>

email: jarrodhelmers@gmail.com