# CS 340 README

## Describe the required functionality

The project aims to create a web dashboard for Grazioso Salvare. This dashboard will help identify and categorize rescue dogs from local shelters. It will use a MongoDB database to allow users to create, read, update, and delete dog profiles.

The user-friendly dashboard makes it easy for staff to search for and manage dogs. Users can filter dogs by age, breed, and suitability for rescue. The dashboard will also show useful statistics through charts. This project will be open source on GitHub and include a README file with setup instructions.

### ****Data Table with Unfiltered View****

#### The dashboard features a dynamic data table displaying all animal shelter records. Users can scroll, sort, and search for specific entries. The table updates in real time based on filter selections.

#### **Default Data Table Without Filters**

A screenshot of a computer

AI-generated content may be incorrect.

### 2. Interactive filters allow users to select dogs based on rescue type:

* + Water Rescue
  + Mountain or Wilderness Rescue
  + Disaster or Individual Tracking
* The filtering options dynamically update the table and charts.

#### **Applying 'Water Rescue' Filter**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

#### **Applying 'Mountain or Wilderness Rescue' Filter**

#### A screenshot of a computer AI-generated content may be incorrect.

#### A screenshot of a computer AI-generated content may be incorrect.📷

#### **Applying 'Disaster or Individual Tracking' Filter**

A screenshot of a computer

AI-generated content may be incorrect.

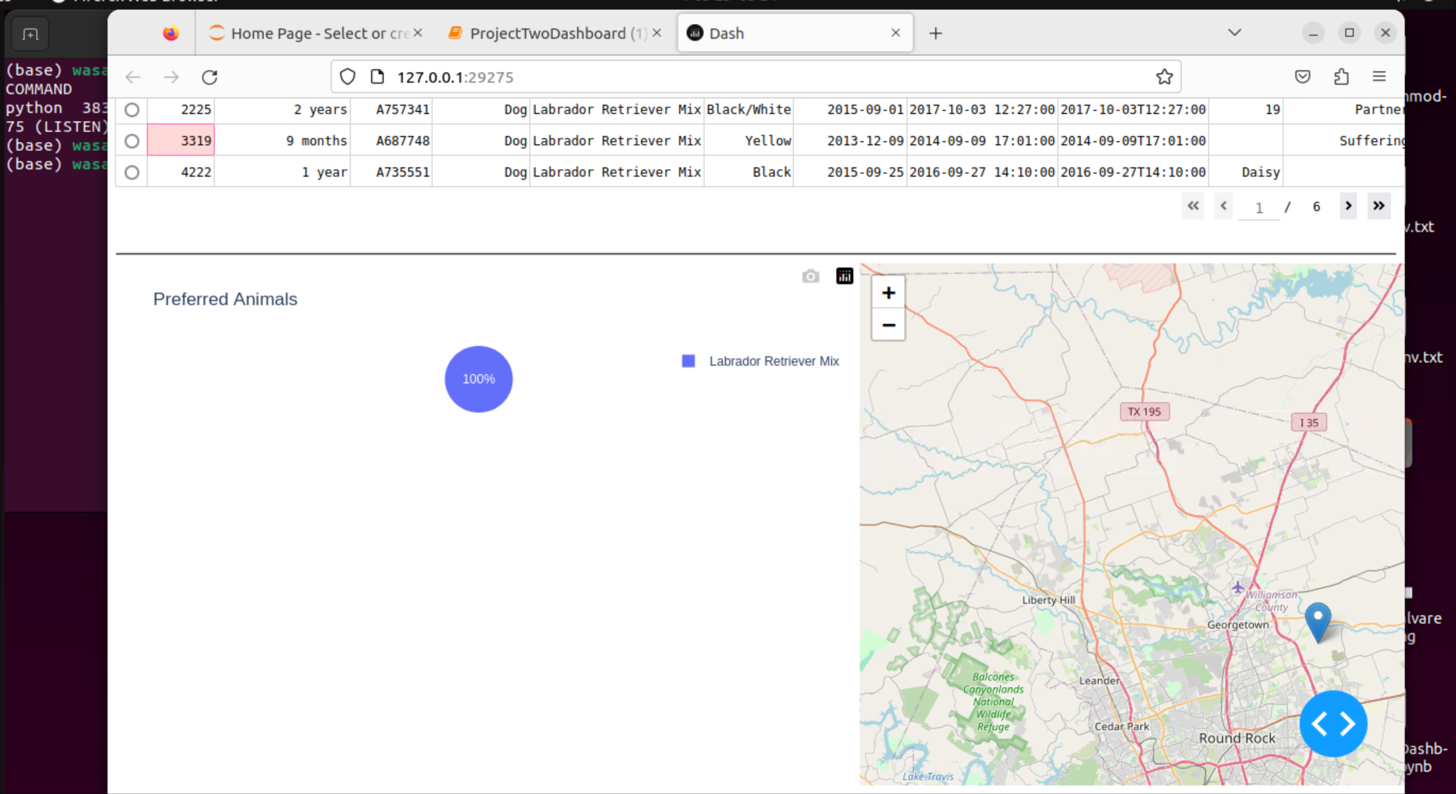
A screenshot of a computer

AI-generated content may be incorrect.

### 3. ****Geolocation Chart****

* The dashboard features an interactive map that plots dogs' locations based on their shelter’s location.
* The map updates dynamically when users apply filters.

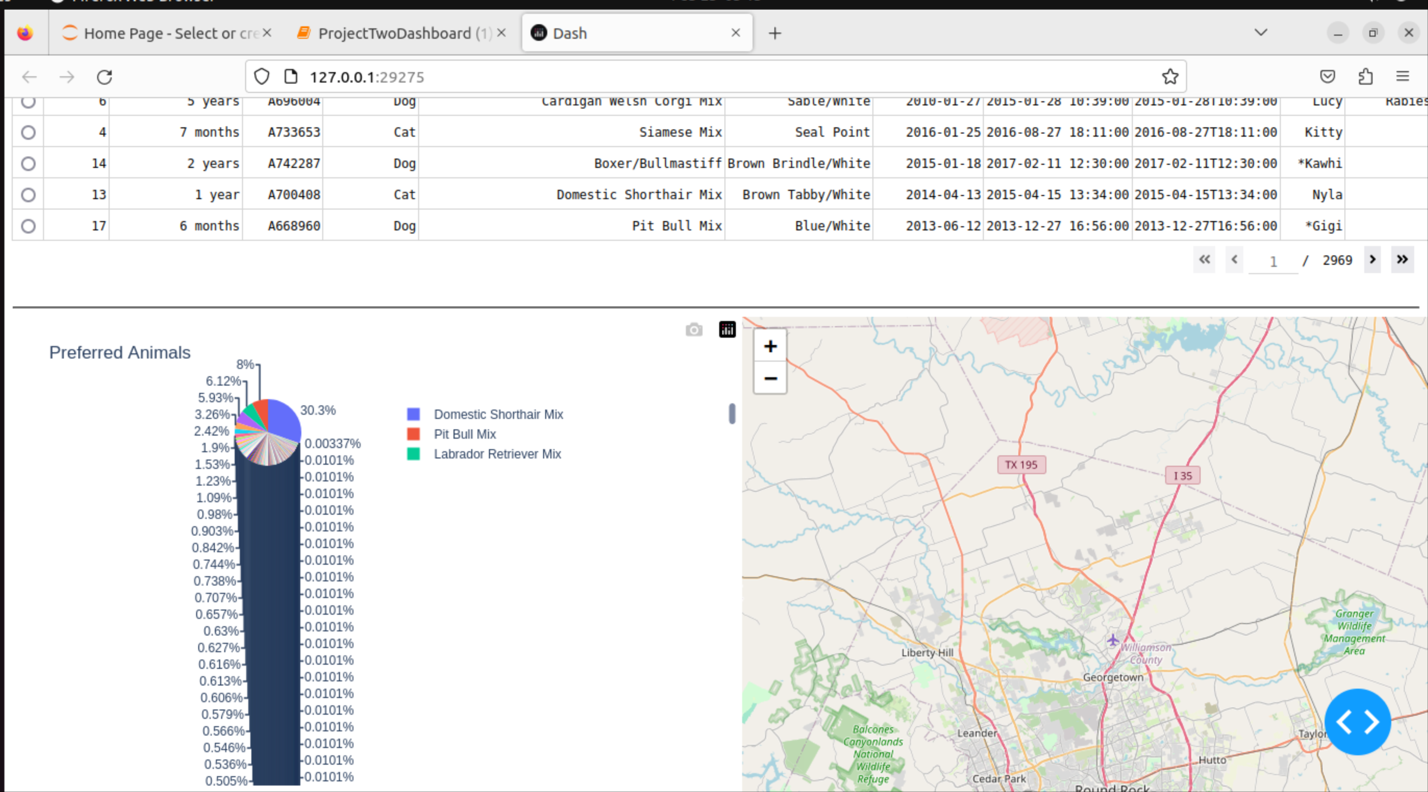
#### 📷 **Screenshot: Geolocation Chart with Filter Applied**



### 4. ****Reset Functionality****

* Users can reset all filters to restore the original dataset.
* The data table, geolocation chart, and pie chart revert to their default state.

#### **Resetting Dashboard to Default State**

****

* **Describe the tools used to achieve this functionality and a rationale for why these tools were used**

MongoDB was chosen as the database component due to its flexibility, scalability, and efficiency in handling large amounts of unstructured data. Its document-based structure allows for easy storage and retrieval of complex animal shelter records, making it ideal for dynamically filtering and querying animal profiles based on rescue training criteria. Additionally, MongoDB integrates seamlessly with Python through libraries like PyMongo, which facilitates efficient CRUD operations that power the application's interactive features.

The Dash framework was selected to provide the view and controller structure for the web application because of its ability to create interactive and dynamic dashboards using Python. Dash follows a model-view-controller (MVC) pattern in which MongoDB acts as the model, Dash components (such as tables and graphs) serve as the view, and callbacks function as the controller, managing user interactions. Dash’s built-in support for interactive visualizations, data tables, and user input elements makes it an excellent choice for developing a user-friendly dashboard that allows Grazioso Salvare to easily filter and analyze animal data.

### Resources Used:

* [PyMongo (MongoDB Python Driver):](https://pymongo.readthedocs.io/en/stable/tutorial.html)
* [Dash Framework](https://www.tutorialspoint.com/python_web_development_libraries/python_web_development_libraries_dash_framework.htm)
* [Python Official Documentation](https://docs.python.org/3/reference/index.html)
* **Explain the steps that were taken to complete the project.**

I developed a Python module to handle CRUD (Create, Read, Update, Delete) operations for my project. This module allows the system to connect to a MongoDB database for storing and retrieving data about the animal shelter.

Next, I created a web-based dashboard using Dash. This dashboard has an interactive data table and filter options so users can search by rescue type and breed. I also added visual features, like a geolocation map and a pie chart, to show the data clearly.

After testing the application to make sure everything works well, I took screenshots of the features in action. I then deployed the application and wrote a README file to explain how to set it up. Finally, I uploaded the project to GitHub so that others can use it as an open-source resource.

## Identify any challenges that were encountered and explain how those challenges were overcome.

One of the challenges I faced was connecting the data table to a pie chart. I had difficulty ensuring that the filtered data from the table synced properly with the chart’s input, as I wanted the pie chart to display the same data as the table and update automatically based on the user’s filter selections.

To resolve this issue, I utilized Dash callbacks to create a link between the table and the pie chart. I ensured that the filtered data was processed before being passed to the pie chart. To verify that the correct data was being used, I added print statements in the callback function. This allowed me to confirm that the filtered data was being accurately updated in the pie chart, ensuring a seamless user experience without compromising performance.

## Contact

**Wasan Alabed**