# CS 340 README Template

## About the Project/Project Title

This project generates an easy-to-use data analysis dashboard for Grazioso Salvare to help identify good candidates for animal search-and-rescue training, all entries of which are stored in a MongoDB database. The dashboard allows users to view the entire contents of the associated database, filter results, view a specific entry (animal) on an interactive map, and display other useful data with custom filters. The project follows the Model-View-Controller pattern, where the MongoDB server is the model, the dashboard is the view, and the controller is the python\_crud module that acts as a driver for interaction between the dashboard and the MongoDB database.

MongoDB was used as the server solution because it is highly dynamic, scalable, and widely supported by languages like Python. The NoSQL structure is well-equipped to handle the types of queries being sent to the database by the animal shelter dashboard. Similarly, the Plotly Dash framework was used to generate the dashboard because it is one of the most convenient, easy-to-use, and well-documented frontend Python modules. It allows for easy instantiation and manipulation of the dashboard and its data, combined with data visualization modules like those provided by the Pandas module.

## Motivation

Manually selecting good candidates for search-and-rescue without the help of a convenient dashboard is a slow and tedious process. This dashboard dramatically eases the process of finding these good candidates by providing easy-to-use data tables and data visualization features. No longer will users have to access the database’s entries using the MongoDB shell; instead, this dashboard performs CRUD operations on the database without a single command being run by the user.

## Getting Started

Creating the dashboard requires creating a Python project to generate the dashboard, running a MongoDB server with the required animal shelter entries, and using the python\_crud module as a driver for communication between the dashboard and the server. Developers using the python\_crud module must define the environment variables required to connect to the server: username, password, host, port, database, and collection. Once the appropriate class object is instantiated and a connection is made to the MongoDB server, the instance can be used to perform CRUD operations on the server. These CRUD operations can be used to visualize data from the server in a user-friendly way, as shown below.

## Installation

The python\_crud module uses Python 3 and PyMongo, which is the officially supported Python driver for MongoDB. Additionally, the ‘pprint’ module is used to produce pretty JSON formatting. To use this Python module in another Python project, first copy the .py file to the other project’s working directory. Then, import the module in the project code. Finally, instantiate an object of the module’s class (AnimalShelter) and the CRUD functions provided by this software will become available.

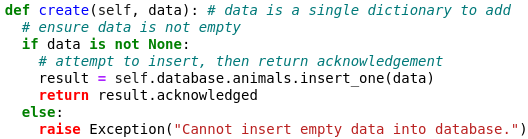
The dashboard shown below was built with a Jupyter Notebook and uses Plotly Dash, Numpy, and Pandas to generate the dashboard and populate it with data. This software requires first instantiating an object of python\_crud’s AnimalShelter class with the necessary environment variables to access the MongoDB server. Then, the dashboard can be generated using the Dash framework with data visualization features (using Pandas) that are populated with data from the server.

## Usage

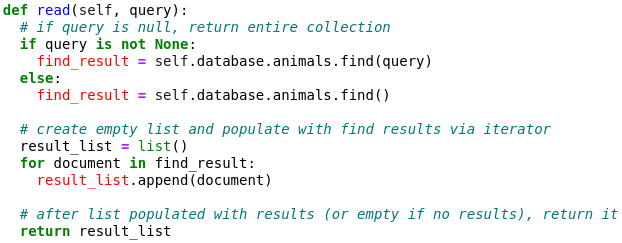
### Code Examples

Examples of the create, read, update, and delete functions of the python\_crud module are as follows.

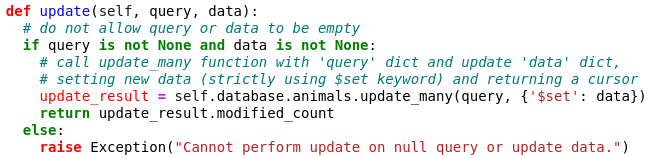
Create:



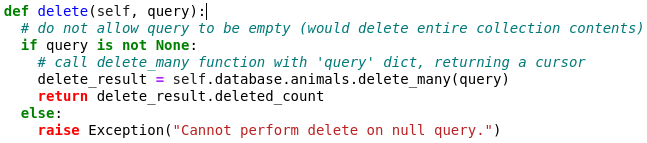
Read:



Update:

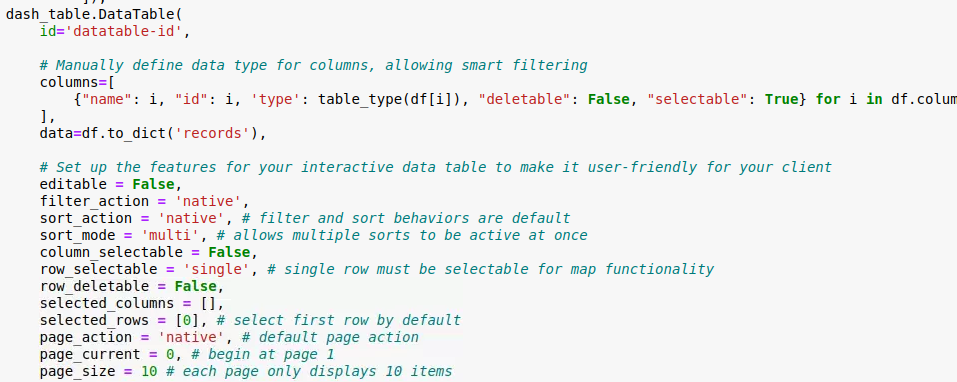


Delete:

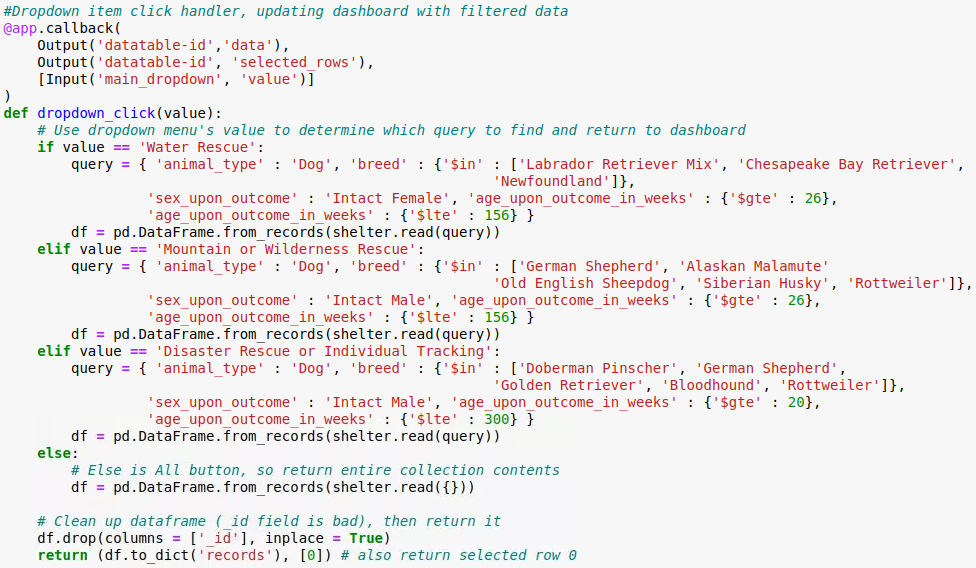


### Examples of dashboard implementation using Dash are shown below.

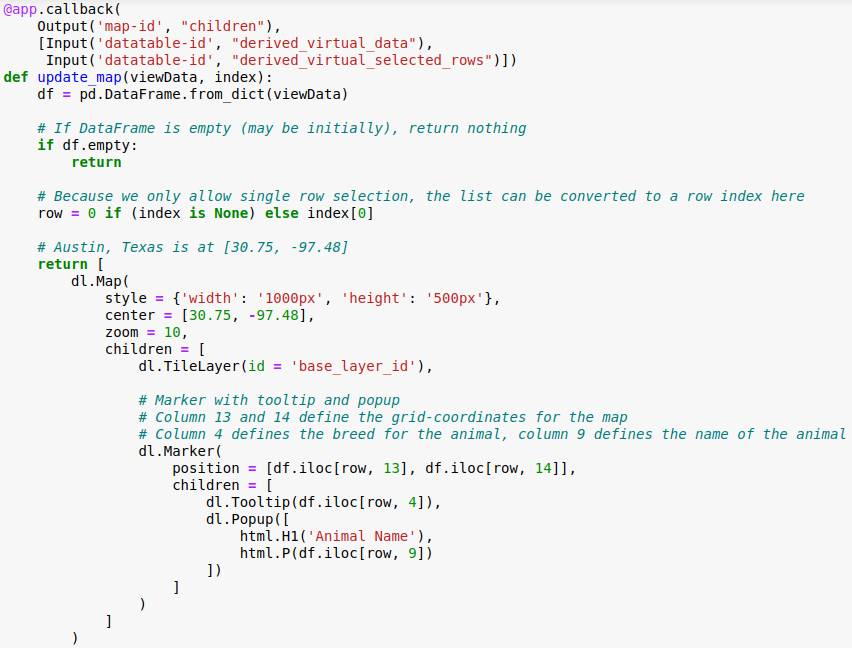
Data table:



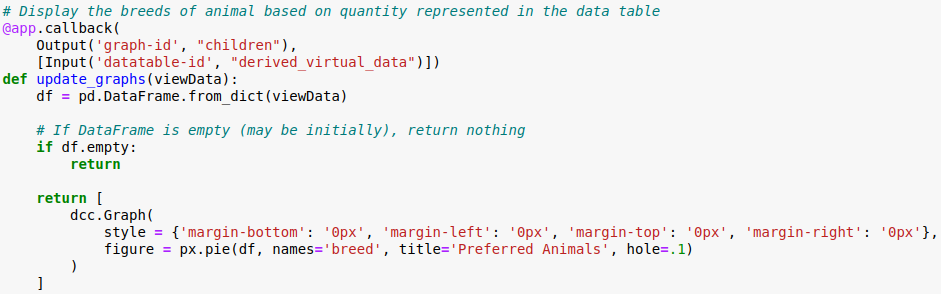
Main filter dropdown click with queries:



Interactive map update:



Pie chart update:



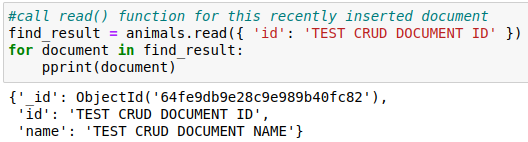
### Tests

To test the CRUD functions in the python\_crud module, first follow the ‘installation’ steps above. Then, the following code can be used to test the CRUD functions in another Python project. Note that the ‘pprint’ module is imported to produce pretty JSON formatting.

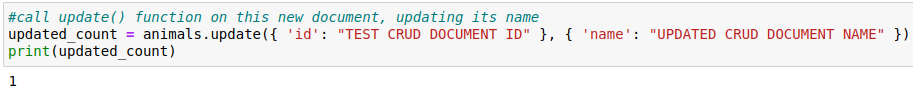
Create:

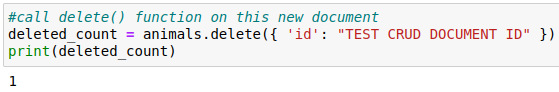


Read:



Update:

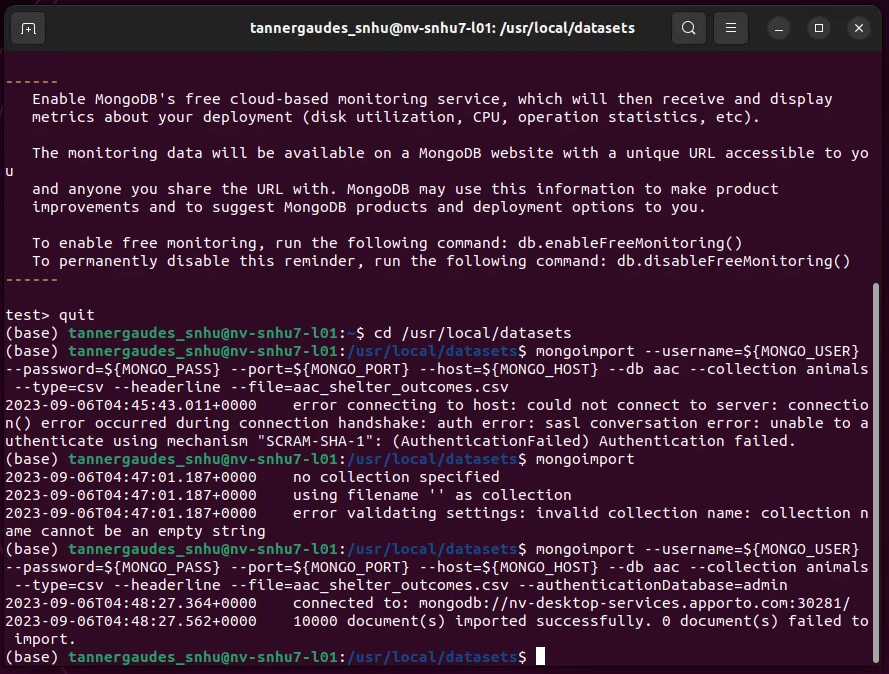
Delete:



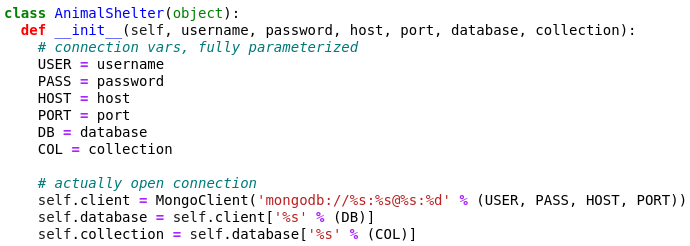
### Screenshots

### Below are screenshots of functional operations of this software.

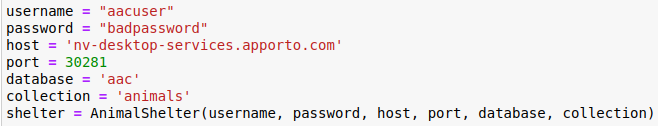
### CSV import to the MongoDB server:



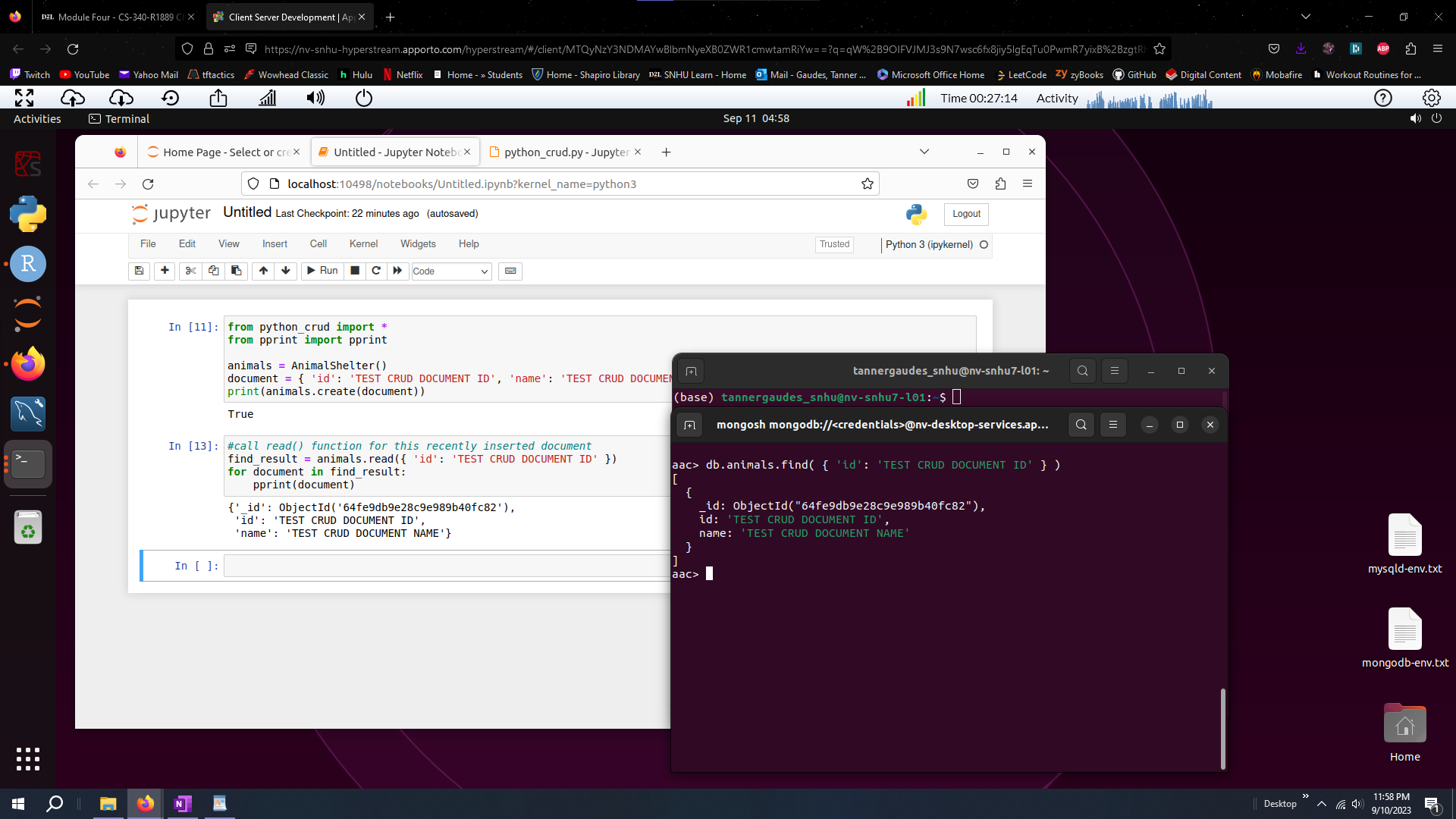
### Python\_crud user authentication and database connection creation:



Python\_crud module usage with constructor arguments:



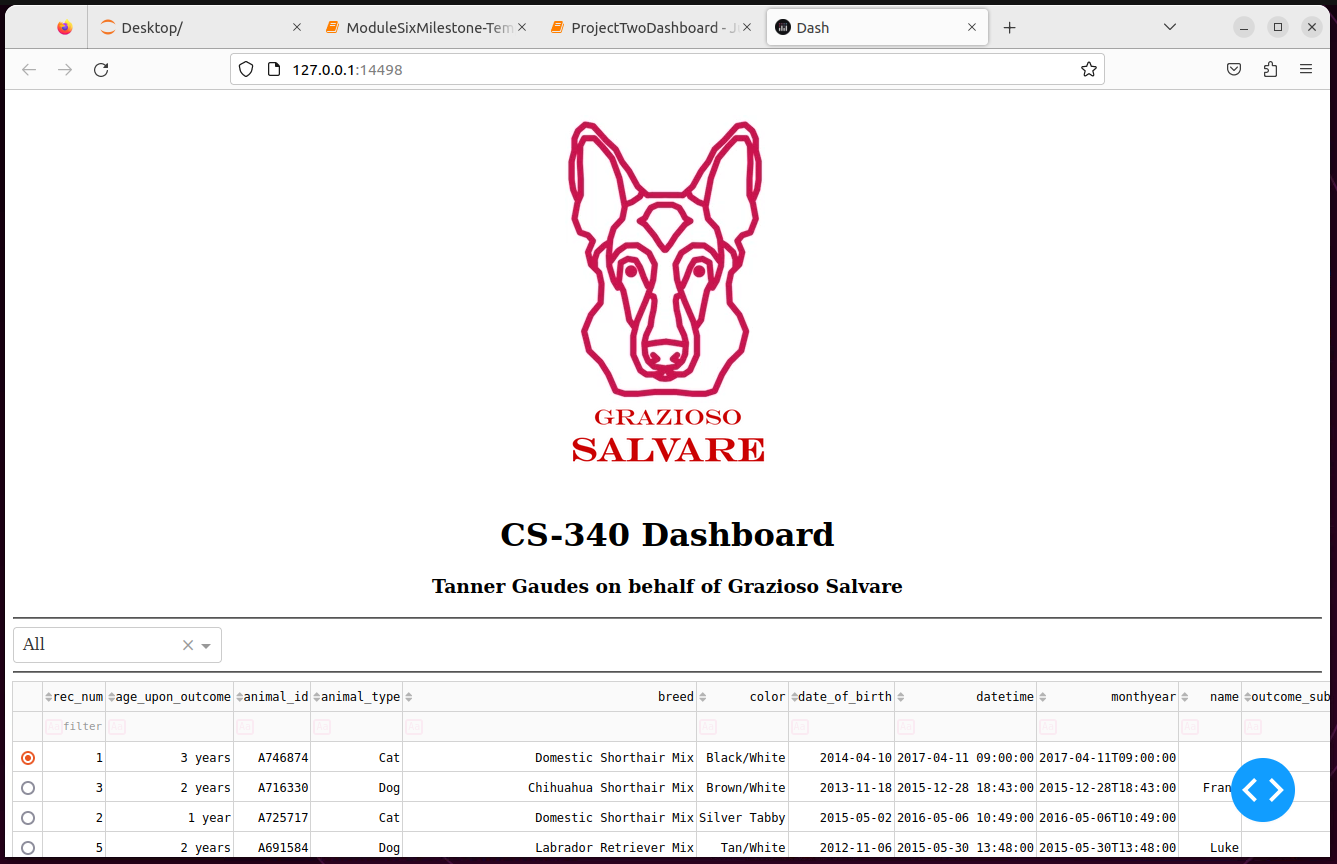
### Python\_crud Create & Read functionality testing:

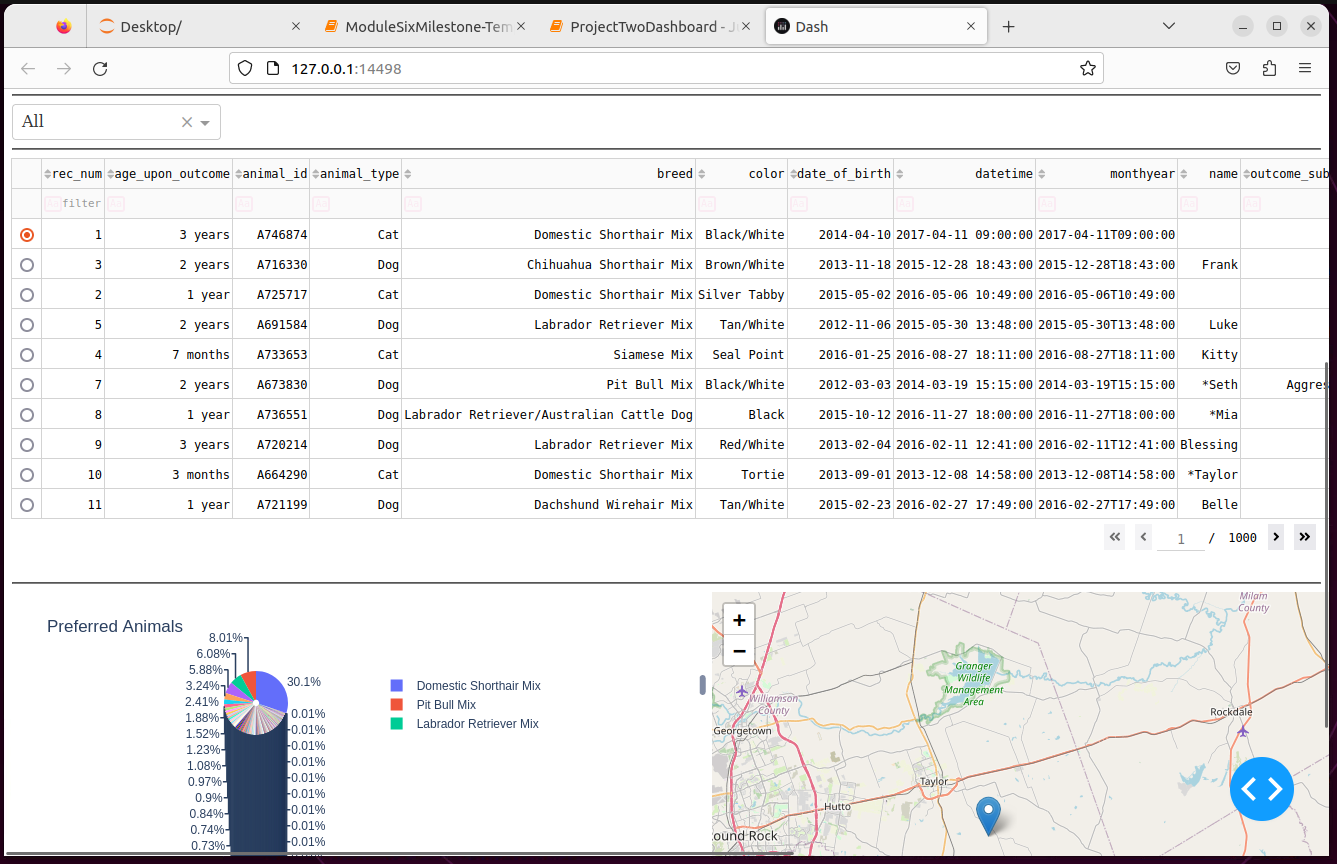


### Python\_crud Update & Delete functionality testing:

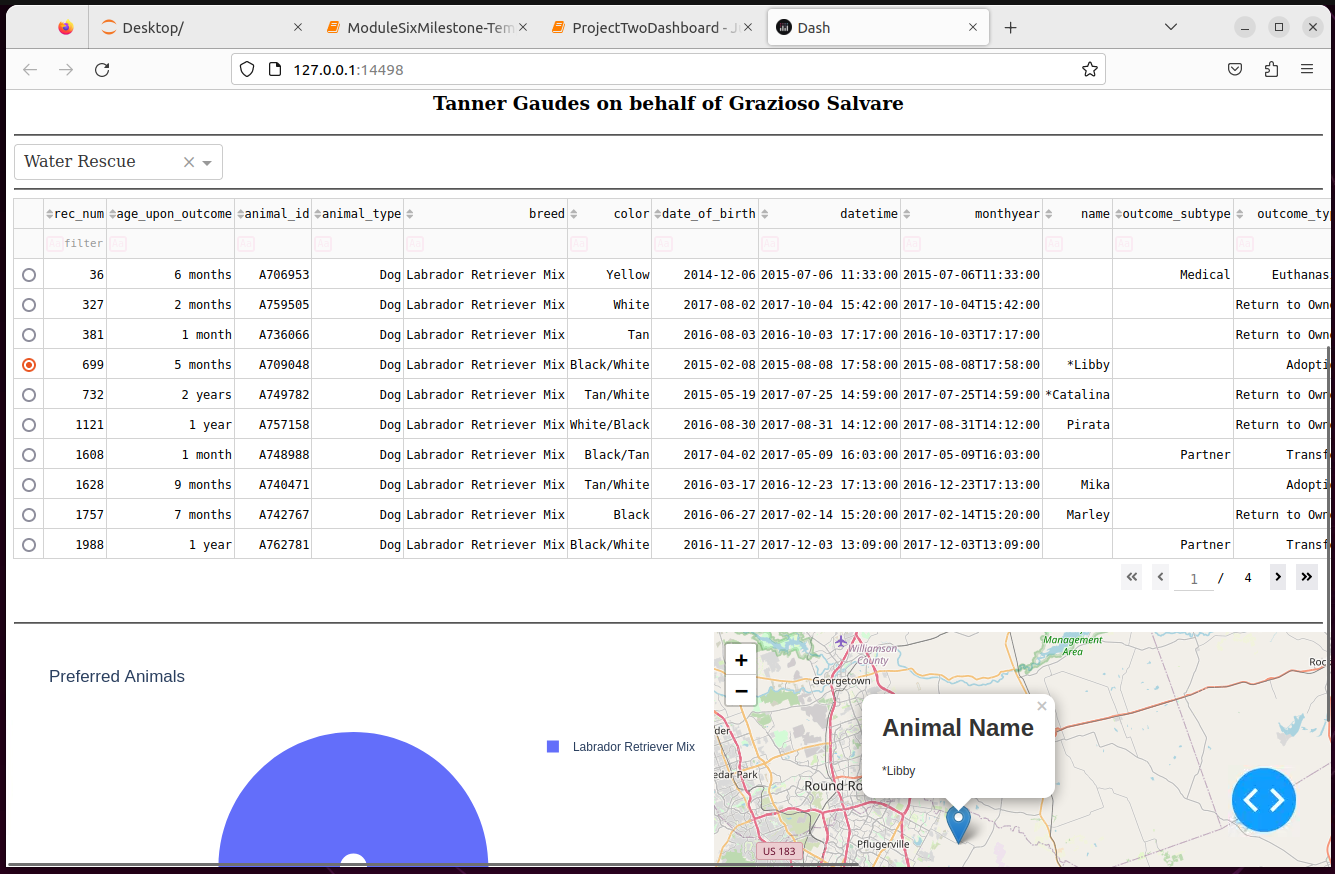
Below are screenshots of the functional dashboard.

Dashboard default with header:

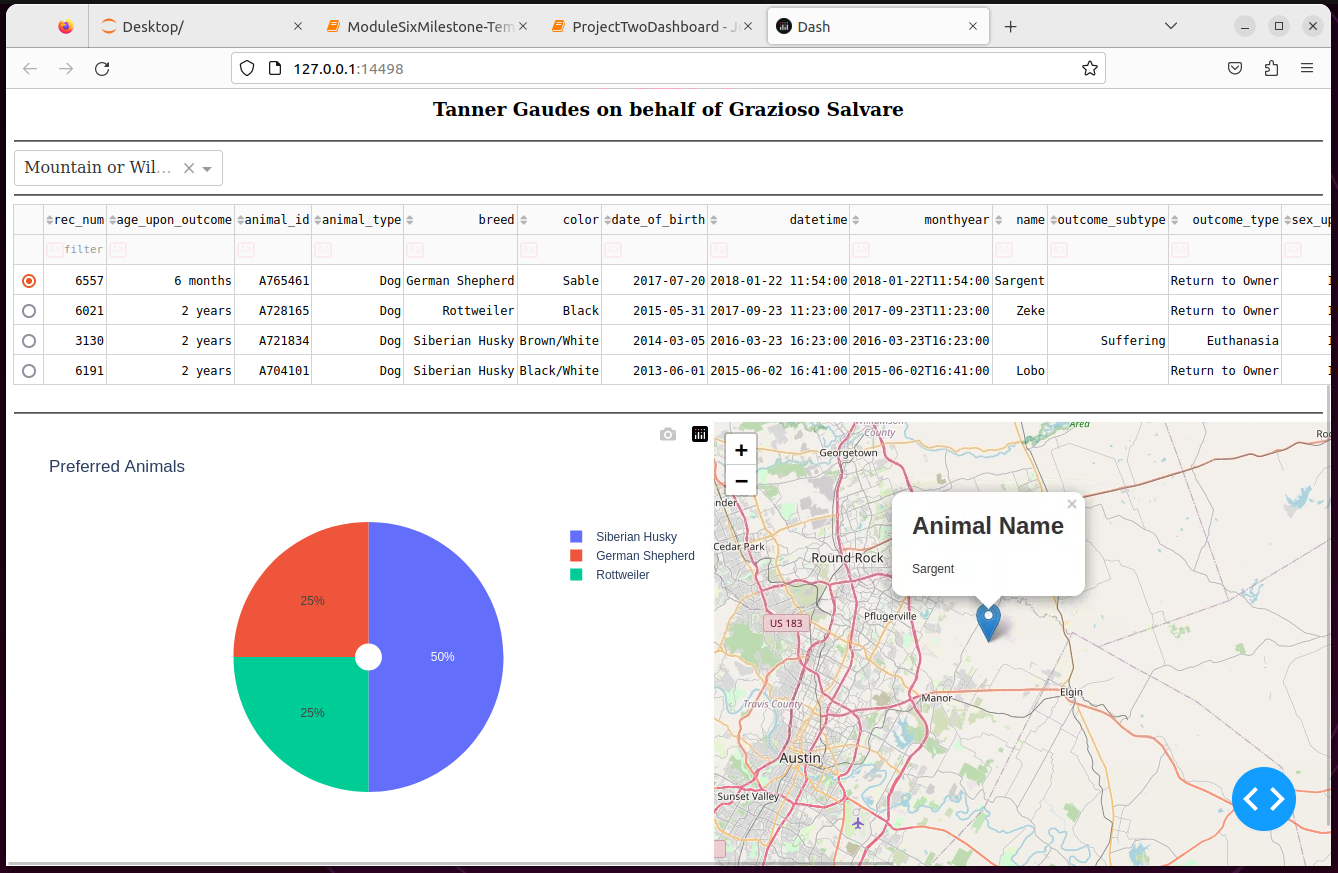
All filter option selected (default/reset):

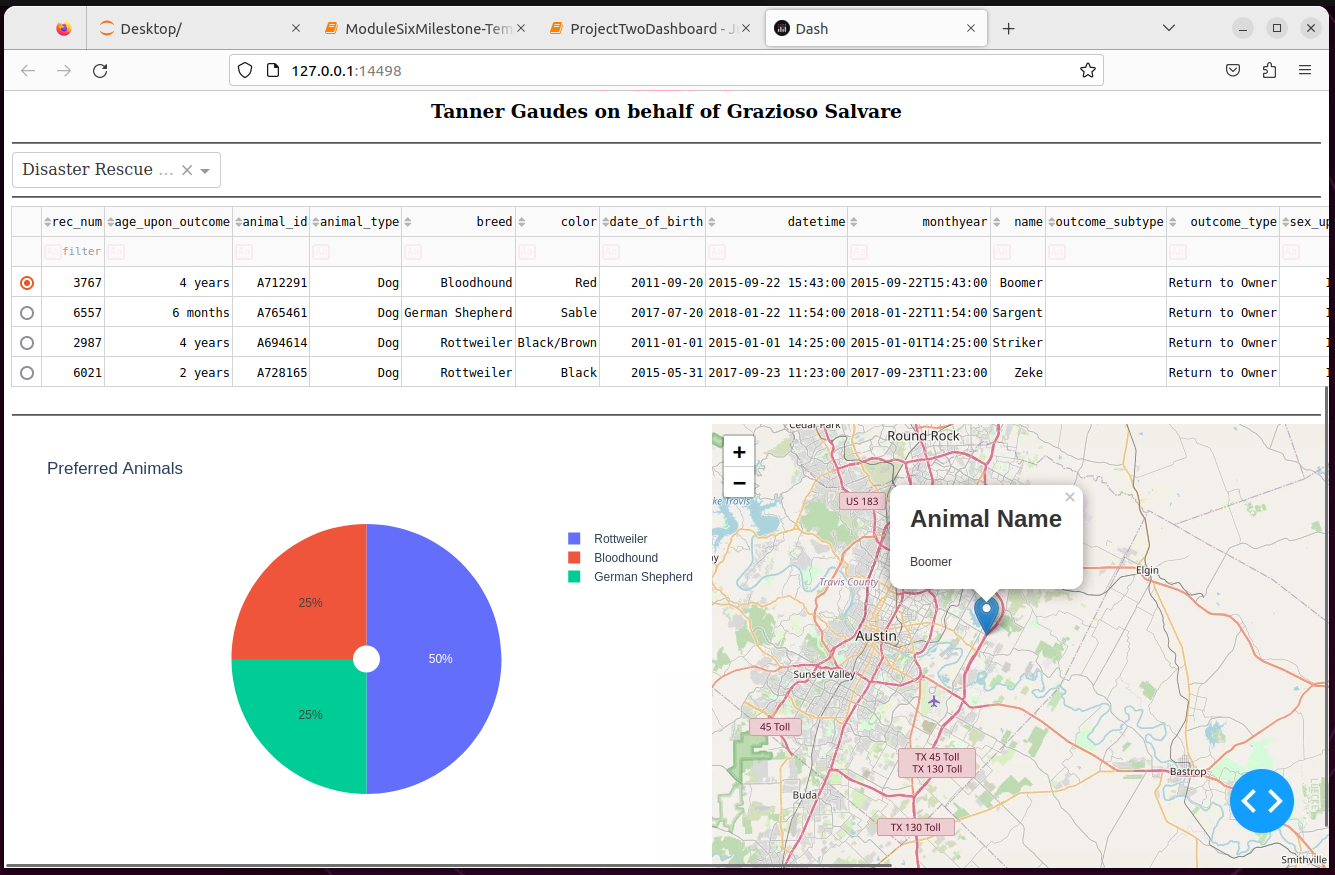


Water Rescue filter option selected:



Mountain or Wilderness Rescue filter option selected:

Disaster Rescue or Individual Tracking filter option selected:



## Roadmap/Features (Optional)

Future quality-of-life functions like those to view server status will be implemented later along the roadmap.

## Contact

Tanner Gaudes – tanner.gaudes@snhu.edu