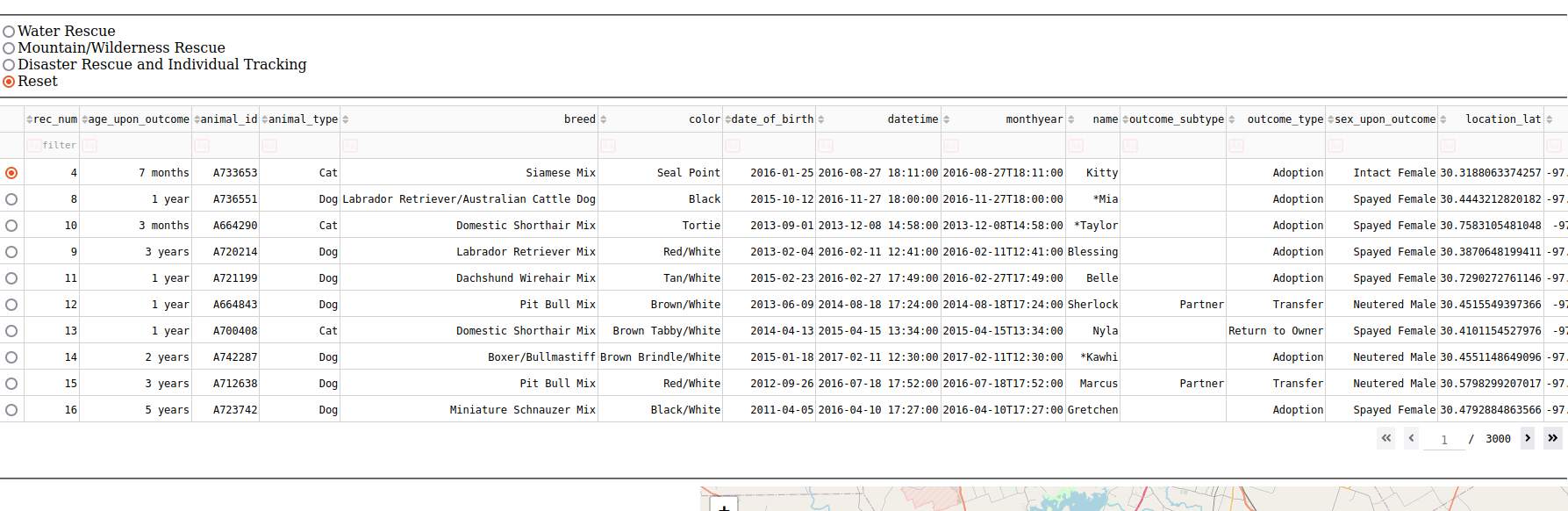
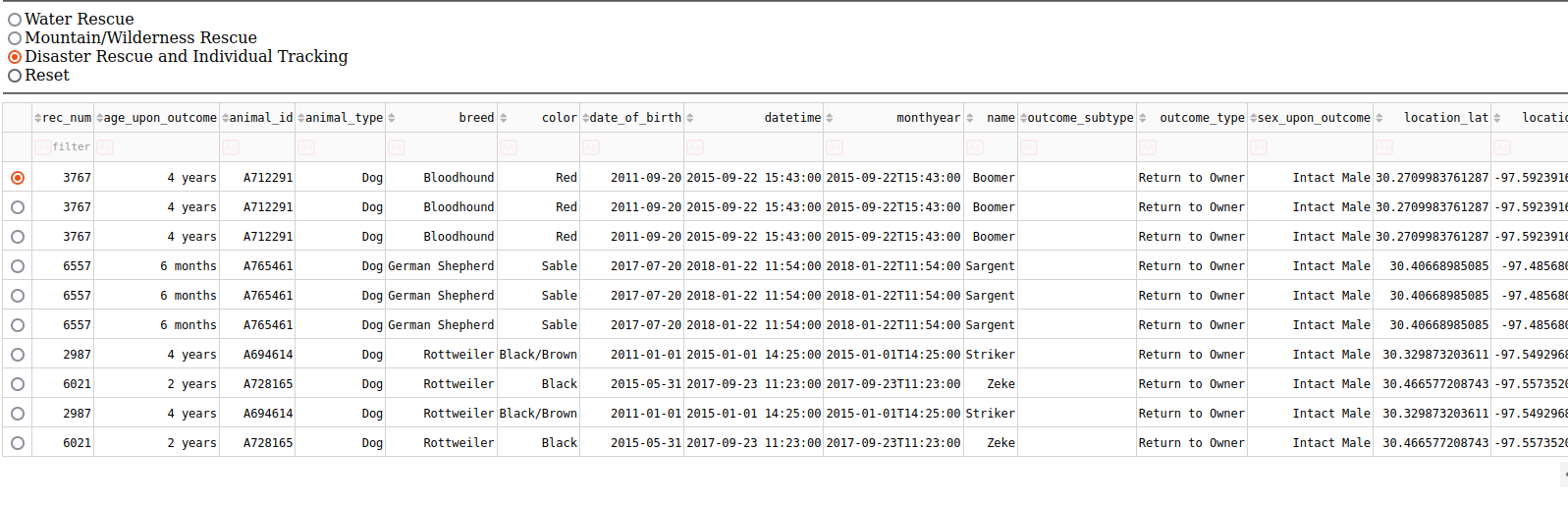
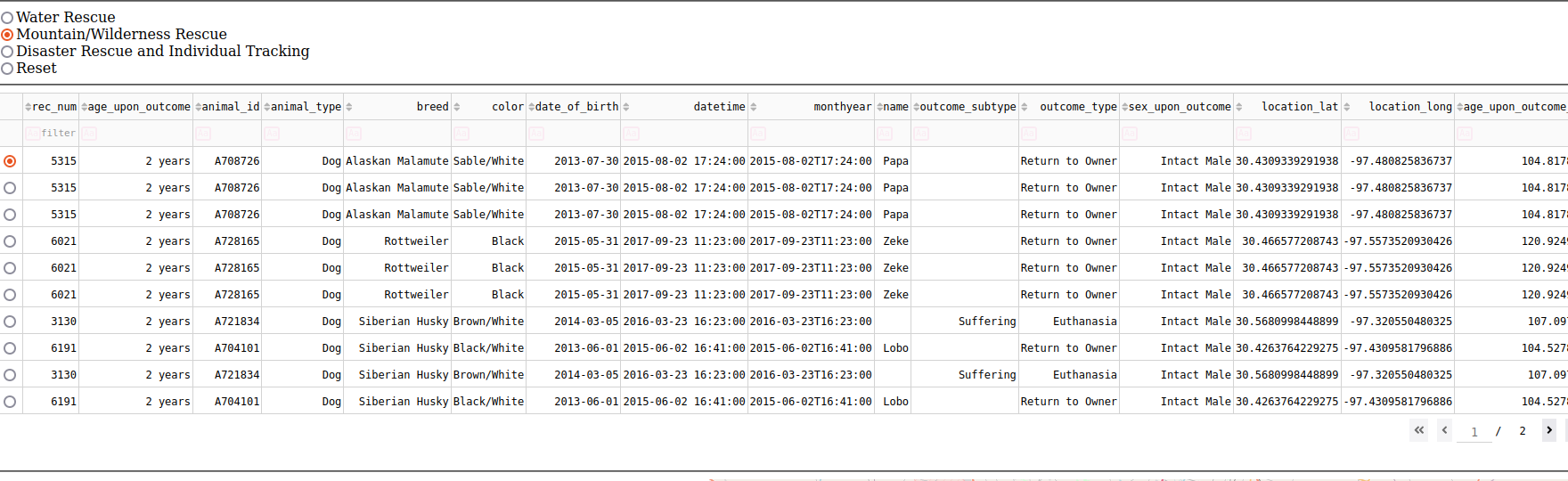
# Graviosos Salvares dashboard README

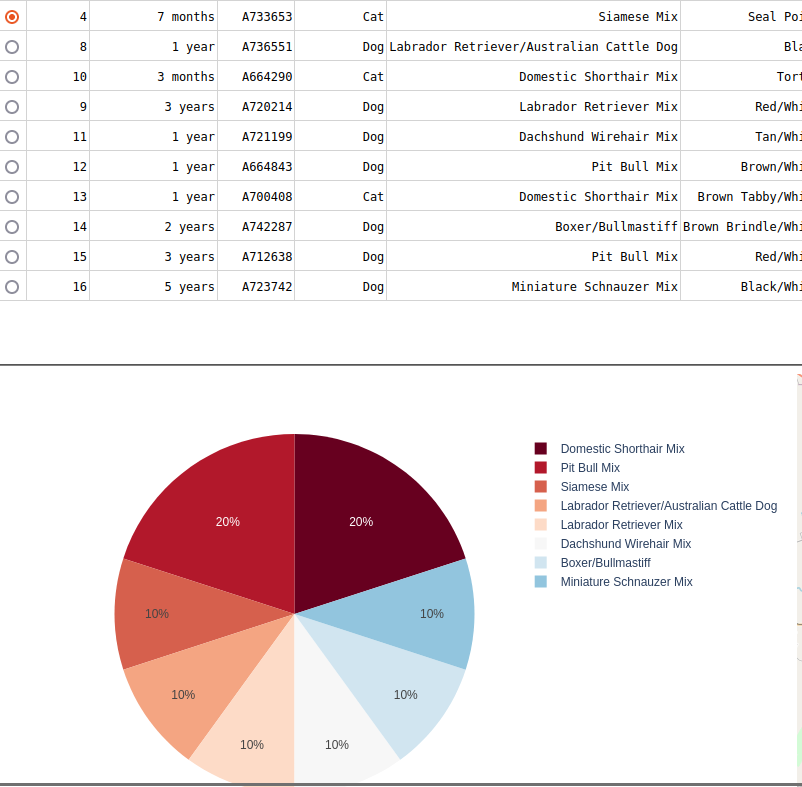
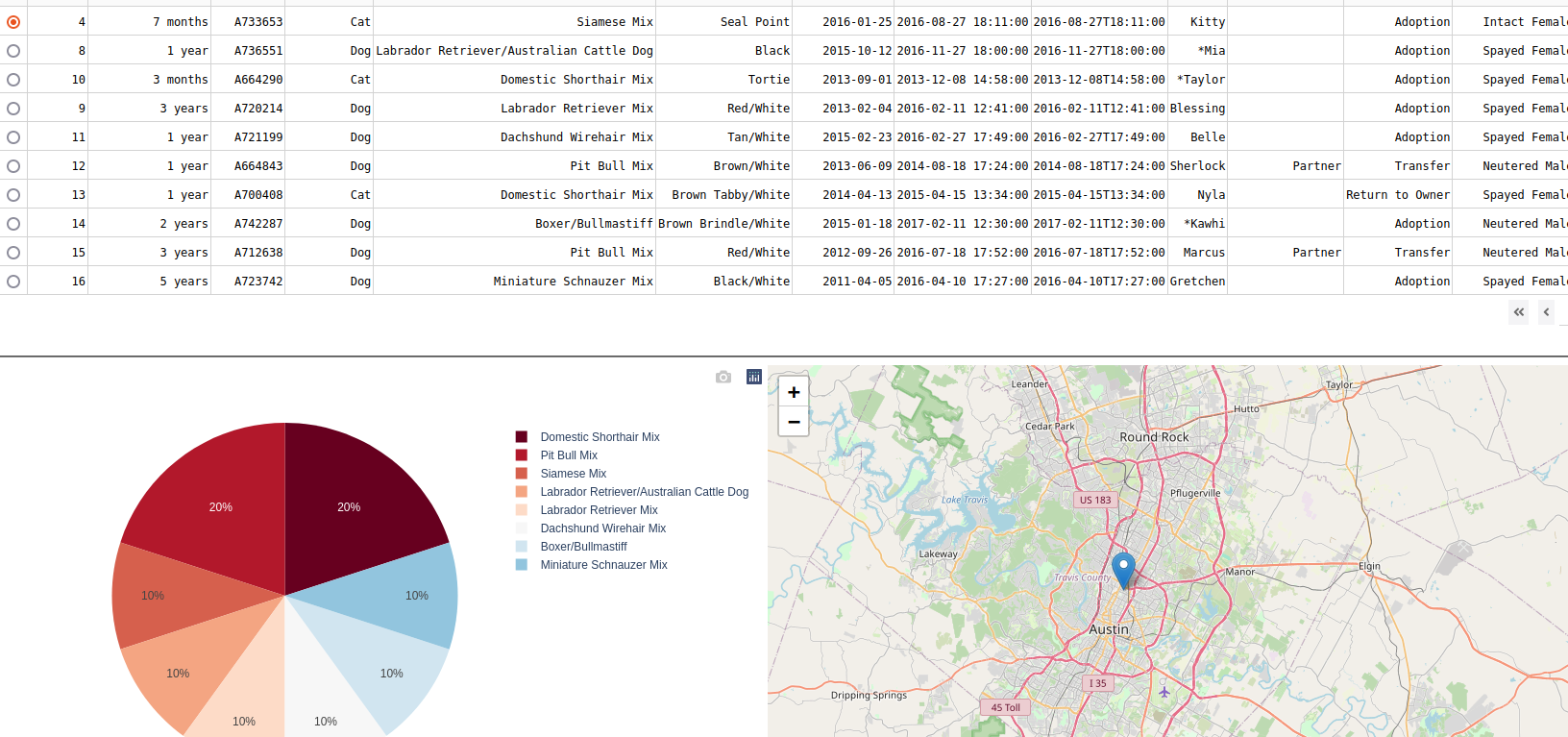
## Functionality:

The Graviosos Salvares dashboard provides users with detailed information about rescue animal records in the shelter's database. It categorizes data into specific types such as water rescues, mountain or wilderness rescues, and disaster or individual tracking. This allows users to easily filter and focus on the exact data they need. Additionally, the dashboard includes visual aids like graphs, making it easier for users to understand the data at a glance rather than sifting through numbers. The pie chart specifically illustrates aspects such as breed, geolocation, and age of the animals.









**Tools Used:**

* MongoDB: MongoDB was employed to store documents and collections, facilitating organized data management and efficient execution of CRUD (Create, Read, Update, Delete) operations.
* Dash: We utilized the Dash framework to integrate specific features that enable interaction with web-based functions.
* Python: This project was developed using Python, requiring numerous variables and precise syntax to ensure proper functionality.
* Jupyter Notebook: The Python code was written and tested in Jupyter Notebook, which served as the Integrated Development Environment (IDE) for developing and executing main functions and methods.

## Steps Taken:

* Importing Documents into MongoDB: The initial step was importing the necessary documents into the MongoDB platform. This is crucial because Python needs to reference these documents to implement CRUD (Create, Read, Update, Delete) operations effectively.
* Implementing CRUD Operations: I then developed the CRUD methods in Python. These methods allow MongoDB to create, read, update, and delete records as required.
* Authentication and User Credentials: Ensuring proper authentication was vital. I verified that user credentials were securely and effectively managed within the MongoDB platform.
* Developing the Front-End Application: The next phase involved creating the front-end interface using Dash. This included adding buttons and other UI elements to facilitate easy navigation and filtering through the databases.

## Challenges:

One of the main challenges I faced during this project was integrating Dash to create a responsive and functional web application interface. Getting Dash to interact correctly with the MongoDB database was especially tricky. Initially, understanding how to link the Dash components with the backend data seemed daunting. I had to delve into extensive documentation and experiment with various configurations to establish a seamless connection. However, after persistent efforts and troubleshooting, I successfully developed a smooth-running web application.

Another significant challenge was implementing the CRUD operations, particularly the delete function. CRUD methods are fundamental for managing database operations—creating, reading, updating, and deleting records. While the create, read, and update functions were relatively straightforward, the delete function posed unexpected complexities. I often had to revisit and thoroughly review the MongoDB and Python documentation to ensure proper implementation. This process required meticulous attention to detail to prevent unintentional data loss or errors. Through persistence and a deeper understanding of the underlying principles, I eventually mastered the delete function and integrated it effectively.

These challenges taught me invaluable lessons in problem-solving and resilience. Working through the intricacies of Dash and MongoDB expanded my technical skills and improved my ability to troubleshoot and adapt to new tools and technologies. This experience reinforced the importance of patience and persistence when tackling complex programming tasks.

## Contact

Mitchel Harmon