# Erik Wilhelm CS 340 README – Module 7 – Project 2

## About the Project/Project Title

Web Application Dashboard using Python code for CRUD and filter operations

This project provides CRUD (Create, Read, Update, Delete) functions that are written in Python 3 for MongoDB 4.4. The project uses MongoDB with Python programming (Jupyter Notebook) for dashboard and CRUD features. MongoDB and Python have been chosen for this project because both offer flexible and dynamic schemas for relational programming MongoDB is provided with a native Python driver as well as a development team dedicated to ensuring that these two software offerings work together flawlessly (https://www.mongodb.com/languages/python).

## Motivation

This project has been created for the benefit of learning to use Python and MongoDB for use in interacting with MongoDB. It has been an interest of mine to better understand database manipulation and exploring the uses of Python in everyday work to see how it can increase efficiency.

## Getting Started

To get started using this project and to explore and/or expand on it would require a current and properly installed Python IDE and access to MongoDB. Additionally, it is important to have a sample database to perform CRUD operations on. The project was created and developed using Jupyter Notebook with Python 3 and MongoDB 4.4. The examples are created using the Austin Animal Center (AAC) database and has been included with my release of MongoDB. (if the AAC database is not included with your current release you can use any sample database or create one). The project has been created using MR. Grazioso Salvare’s Dashboard Specification document and has been built to support those needs and recommendations.

## Installation

The work that I have performed has been created on software that is provided by my organization so installation has not played a role in the development of this project. However, Python can be downloaded and installed from <https://www.python.org/downloads/> . Python version 3.10.5 and earlier is available for most operating systems such as Windows, Linux/Unix, macOS. Select the download appropriate for your system and install. MongoDB is the NoSQL database shell used for this project and is release 4.4. MongoDB is a paid subscription software that offers a free trial. IF interested in using a trial use this link: <https://www.mongodb.com/cloud/atlas/register> and is available on most OS. MongoDB uses a Linux terminal so it is recommended to be familiar with terminal commands and navigation as well as Linux commands. Most initial commands are available through searches and there are many “cheat sheet” resources openly available on the web as this one provided to me by my Professor Dr. Kellogg: <https://www.opentechguides.com/how-to/article/mongodb/118/mongodb-cheatsheat.html>

## Dash Framework

For the user interface and dashboard experience I have used Dash. Dash is self-described as a low-code framework for rapidly building data apps in Python (https://dash.plotly.com/introduction). Dash has been incorporated to provide filtering options via selection buttons and background filtering to provide animals based on breeds, sex, age, and preferred breed specific traits such as swimming, tracking, and endurance. Dash has been employed to show tables, maps, and graphs within the program to provide additional user efficiency and comfort of manipulation of data.

**Steps for completion and lessons learned**

The dashboard project has been incrementally developed over the course of many weeks. I started initially with understanding the MongoDB software and learning how to manipulate data from the Mongo terminal. I progressed to creating python code to perform CRUD operations as well as test scripts to run the code on a database from Jupyter notebook using a Python terminal. Finally, I worked with learning to incorporate Dash framework to make CRUD operations more user friendly for members of Mr. Salvare staff and Members of the Austin Animal Center to ensure that animal data is entered properly on initial animal entry so that searching using the database works efficiently from the dashboard. I have learned a considerable amount on manipulating data within MongoDB and Python> I have also learned the initial setup of creating a dashboard to perform functions and show how efficiency can be improved using additional software tools.

The greatest challenge for me has been understanding the starter code that has been provided by management to make modifications that positively impacts my personal work. I have spent many hours watching videos on YouTube to learn MongoDB data manipulation and Jupyter code interface. I have found a lot of great information using stack overflow, and information from contributors to the online collaboration site of GitHub. Much of my work has been developed using starter code provided to me, research and trial and error. In future developments I would intend to use a python IDE that has a more robust testing and debugging feature as Jupyter while providing insight did not always lead me to a proper answer. Another challenge that I faced was understanding the callback input and output functions, specifically for the data table. Initially, I was not able to get the data table to update with the filtering after many iterations and finally consulting with Dr. Kellogg I found that I was output columns in general and instead should be outputting only the selected\_columns. I found this to be a very difficult problem to overcome because I was not receiving error messages since the code was operating correctly to the interpreter. I read the dash documentation and watched videos and finally asked for help. This led me to this area of my code, and I tried various outputs until I got the current version.

## Usage

*Use this space to show useful examples of how your project works and how it can be used. Be sure to include examples of your code, tests, and screenshots.*

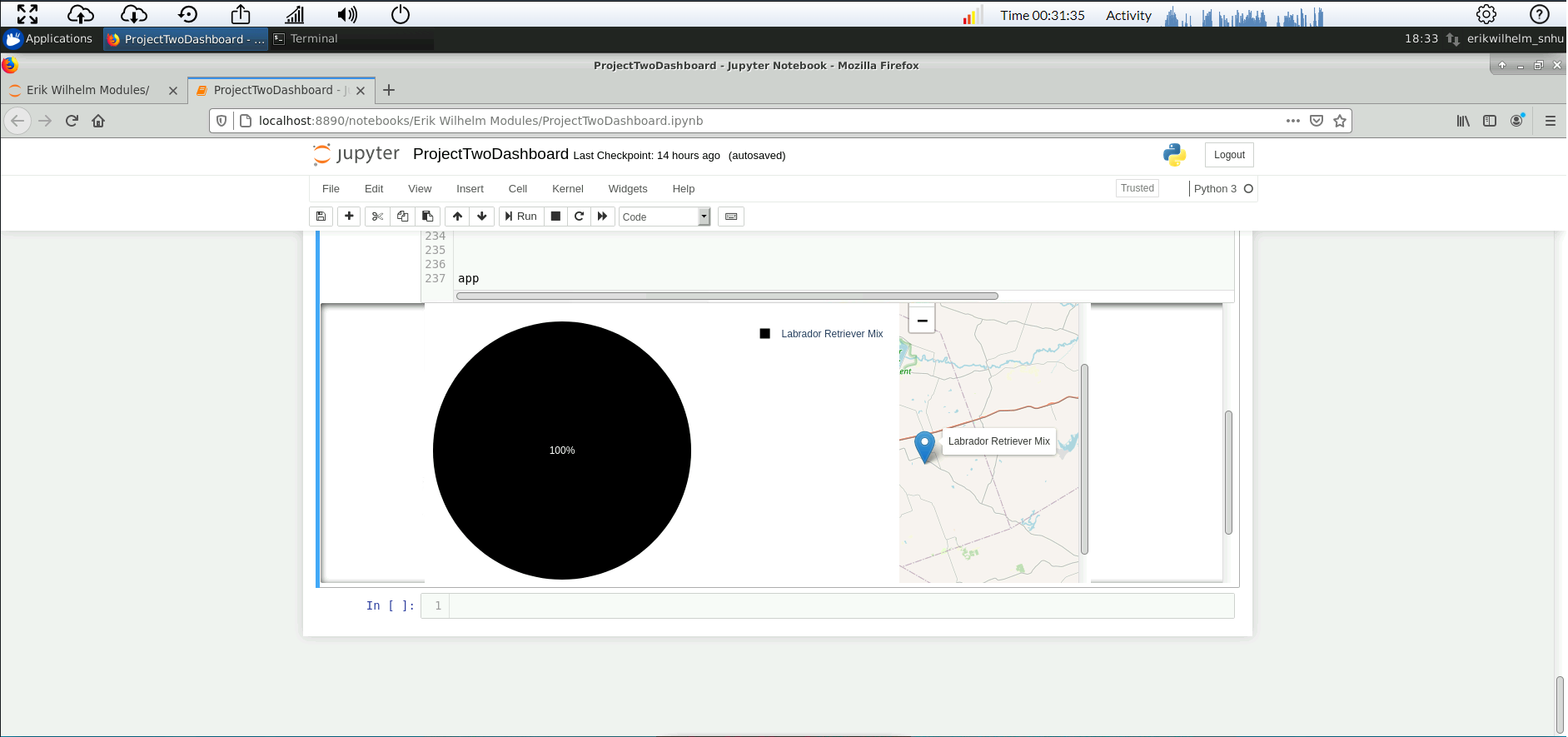
### Dropdown for Filtering by use

Dropdown for Filtering for water rescue and results

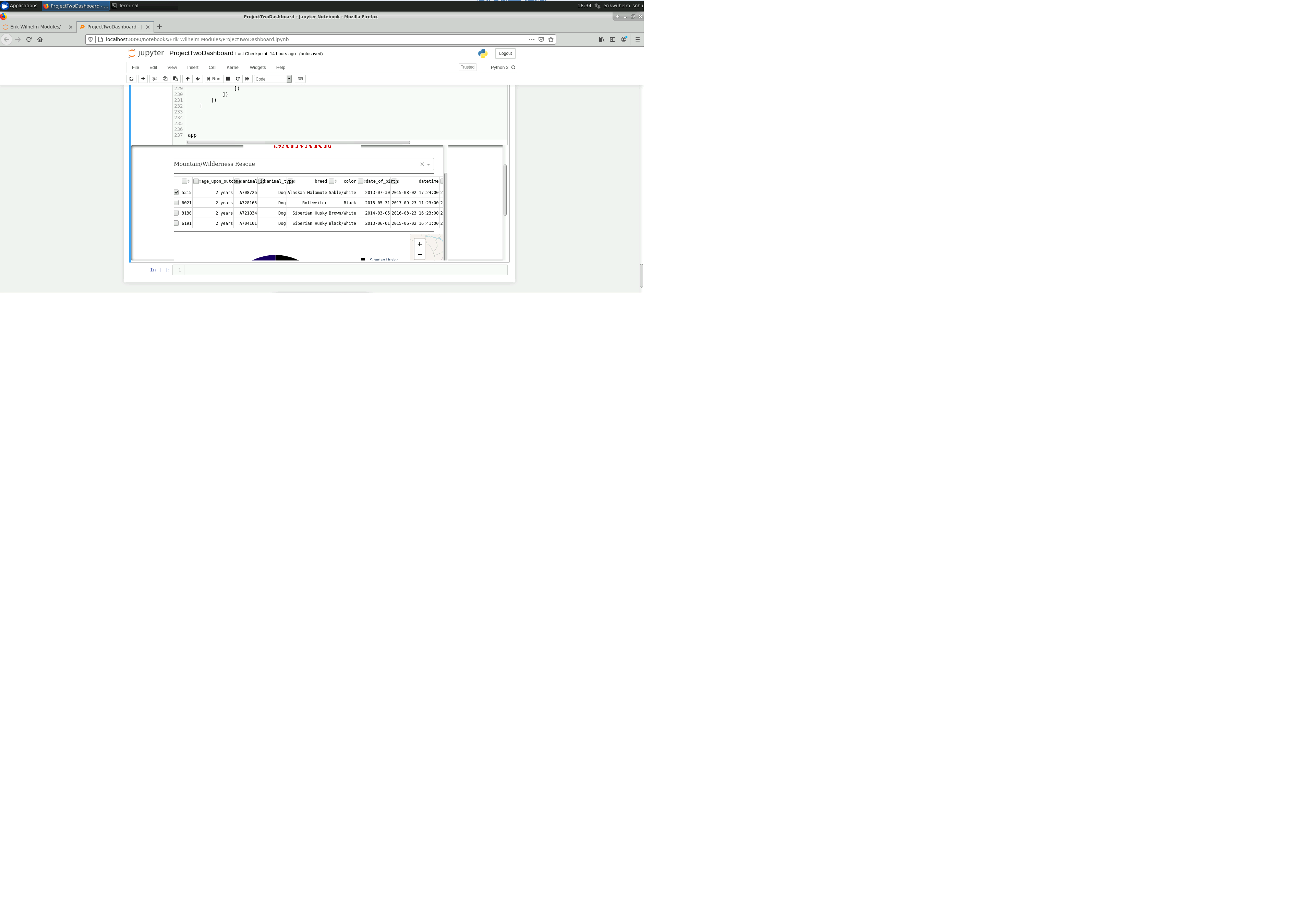
Graphical user interface, application

Description automatically generated

Graph and tooltip for water rescue and results



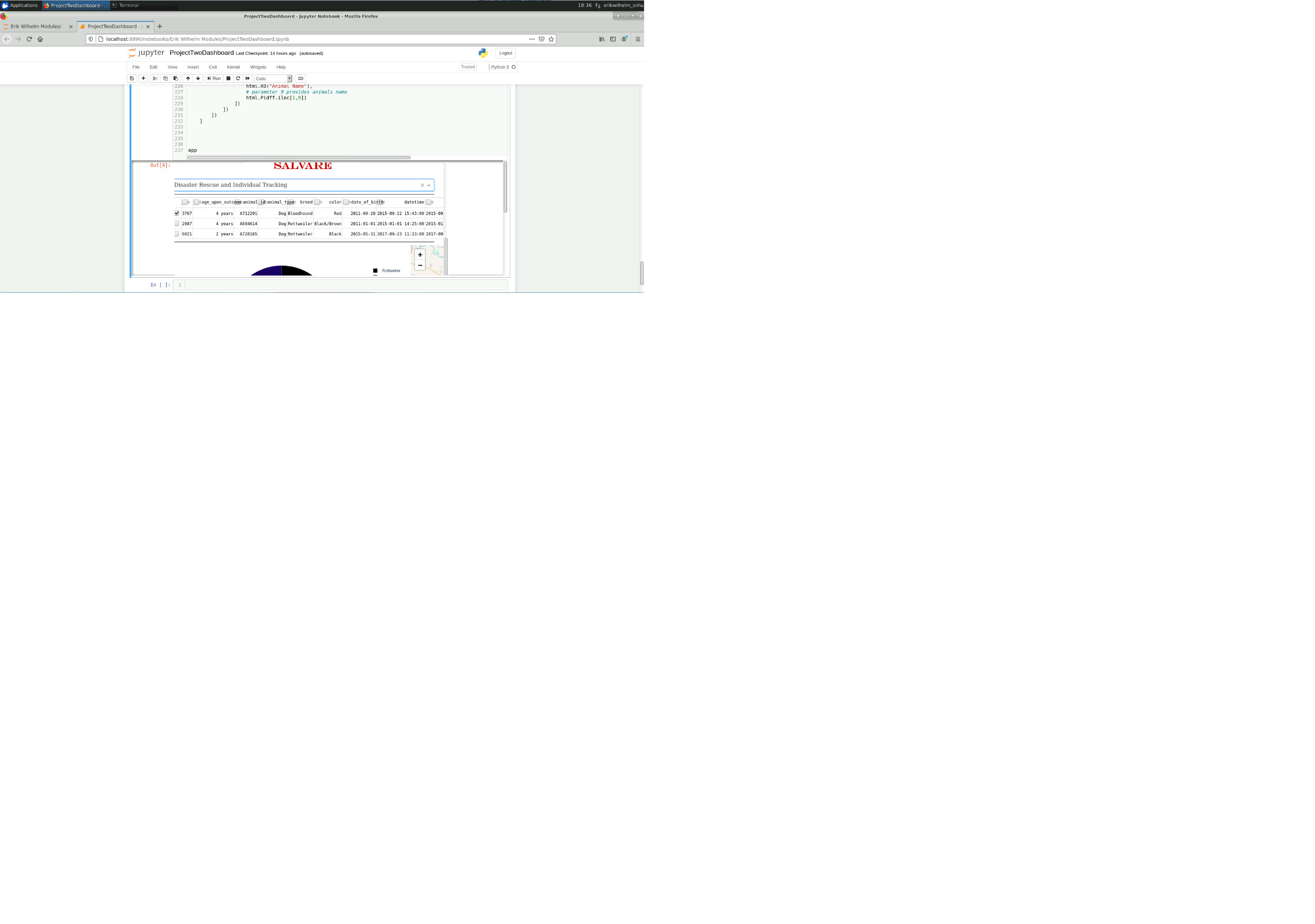
Filter for mountain rescue and results

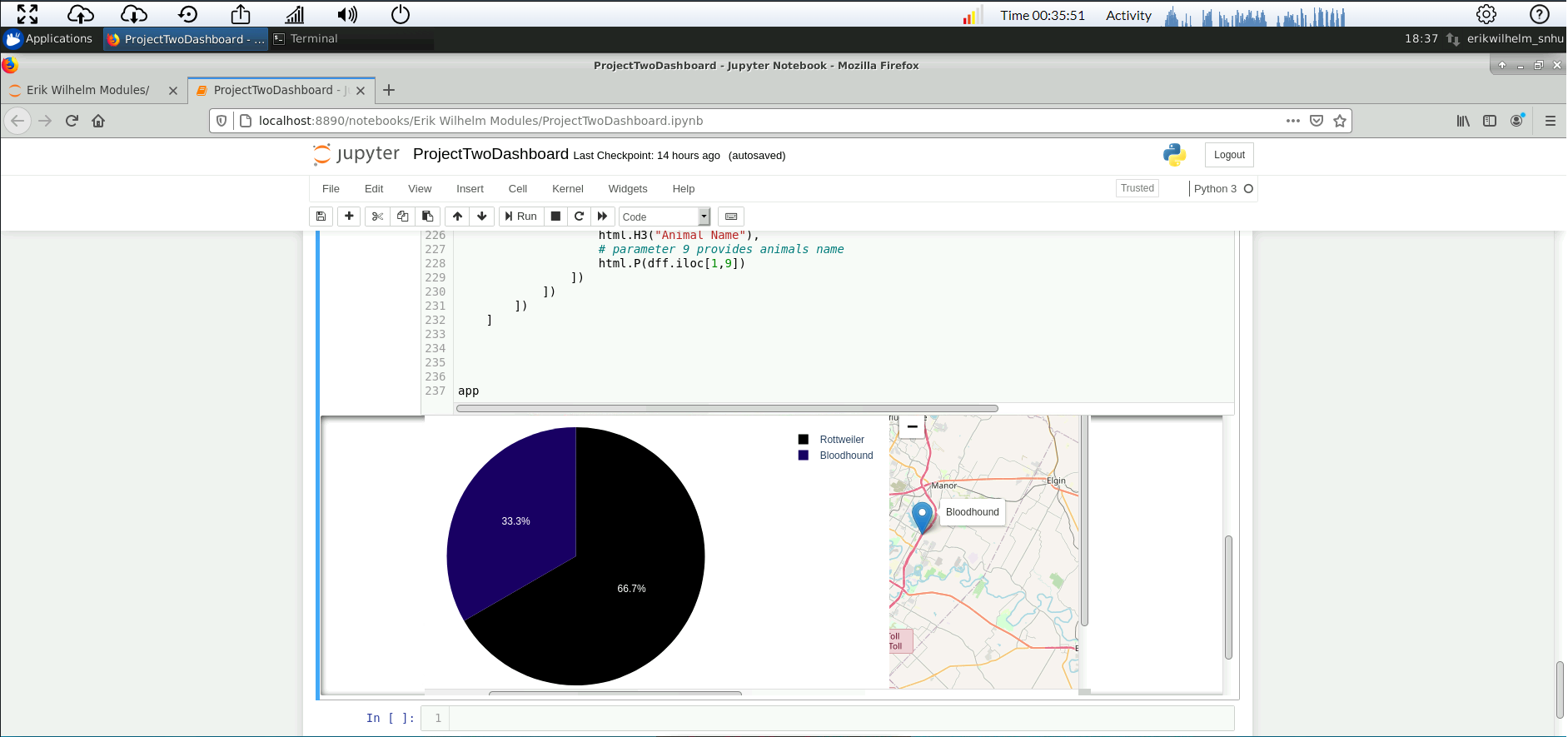


Graph and tooltip for Mountain Rescue and results

Graphical user interface, application

Description automatically generated

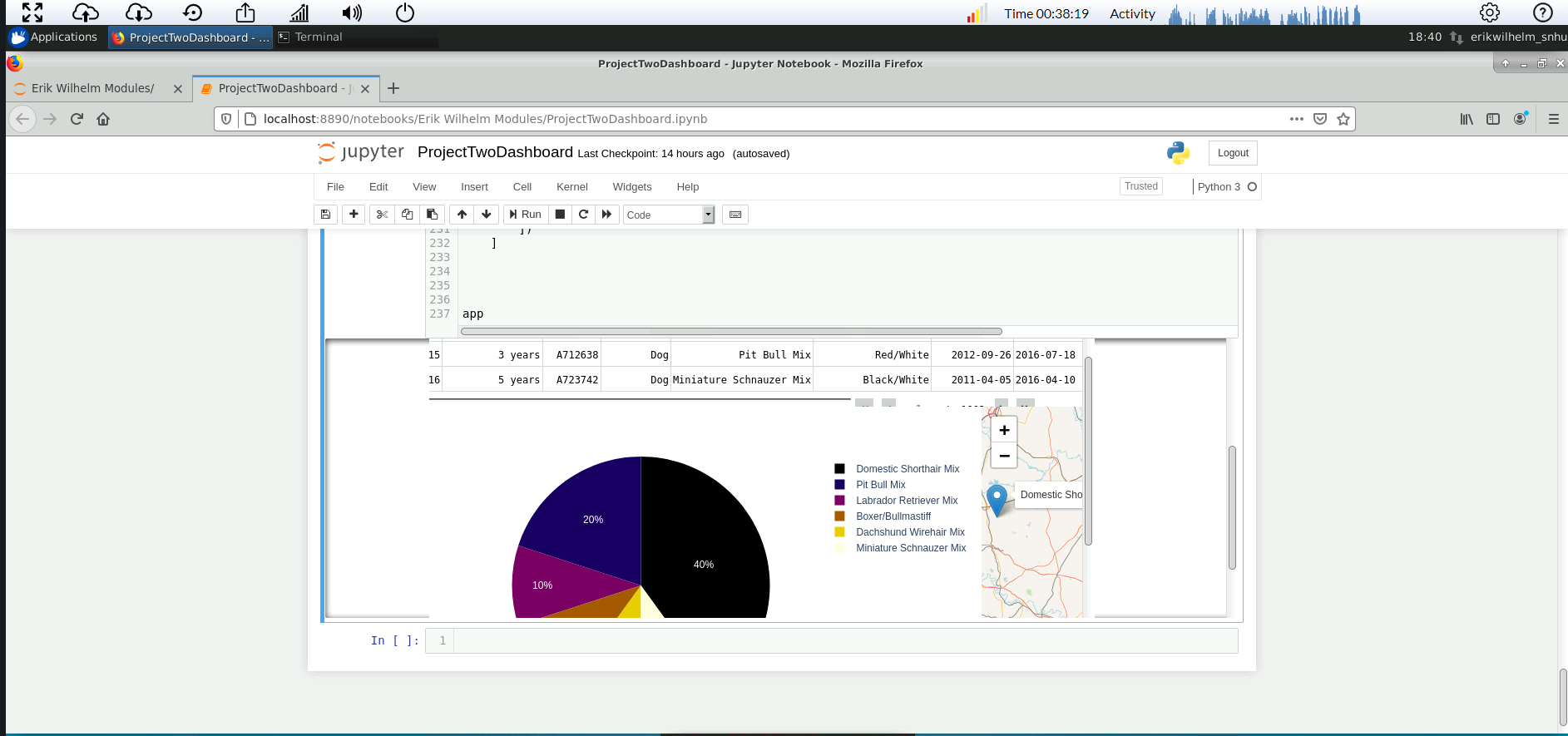
Dropdown and table for Disaster Rescue

Graph and tooltip for Disaster Rescue and tooltip

Reset to overall table, graph and tooltip

Graphical user interface, application

Description automatically generated



## Tests

## Throughout the development I exercised my application and debugged after adding each new feature. For example, after creating the table and app callback I would run the code and ensure the creation of tables. This process was repeated for each feature. The drop-down menu started as radio buttons and evolved into the dropdown seen in the final version. I have also incrementally modified from a simple bar graph/histogram to a pie chart and finally to the “electric” color scheme in the current version.

## Roadmap/Features (Optional)

*Additional versions to add animals as required by the client as well as updates to page layout and interface.*

## References

<https://stackoverflow.com/questions/66298470/mongodb-dash-and-python>

<https://dash.plotly.com/datatable>

<https://dash.plotly.com/datatable/dropdowns>

<https://dash.plotly.com/basic-callbacks>

<https://www.youtube.com/watch?v=-KLtU_t5bXs>

<https://www.youtube.com/watch?v=mTsZL-VmRVE>

<https://plotly.com/python/pie-charts/>

<https://towardsdatascience.com/are-you-still-creating-boring-static-plots-its-time-to-move-on-384f49b60d16>

<https://www.angela1c.com/projects/dash/basic_callbacks1/>