# README

## Project Functionality

The dashboard displays animals from a database and allows filtering by animal type, breed, age, and outcome type. As the user changes filters, the data table updates automatically. A pie chart shows the top ten breeds in the filtered results, and a map displays where each selected animal is located. A reset button clears all filters so the user can see the full dataset again.

## Tools and Rationale

MongoDB was chosen because it stores flexible JSONlike data that works well with Python. Dash was used to build the interface since it’s great for data visualizations and lets you connect Python code directly to interactive elements. Dash Leaflet was used to show animal locations on a map, and Plotly Express was used to make simple charts that respond to user input.

## Steps to Complete the Project

1. I built the CRUD\_Python\_Module.py file to connect to MongoDB and handle basic operations like reading and writing data.  
2. I then created the ProjectTwoDashboard.ipynb file in JupyterLab and set up the Dash layout.  
3. The main data table was added first to show the full list of animals.  
4. Filter controls like dropdown menus and sliders were added next.  
5. The pie chart and map were added so the dashboard could display visual data.  
6. Finally, I tested each feature and made sure everything updated correctly when filters changed.

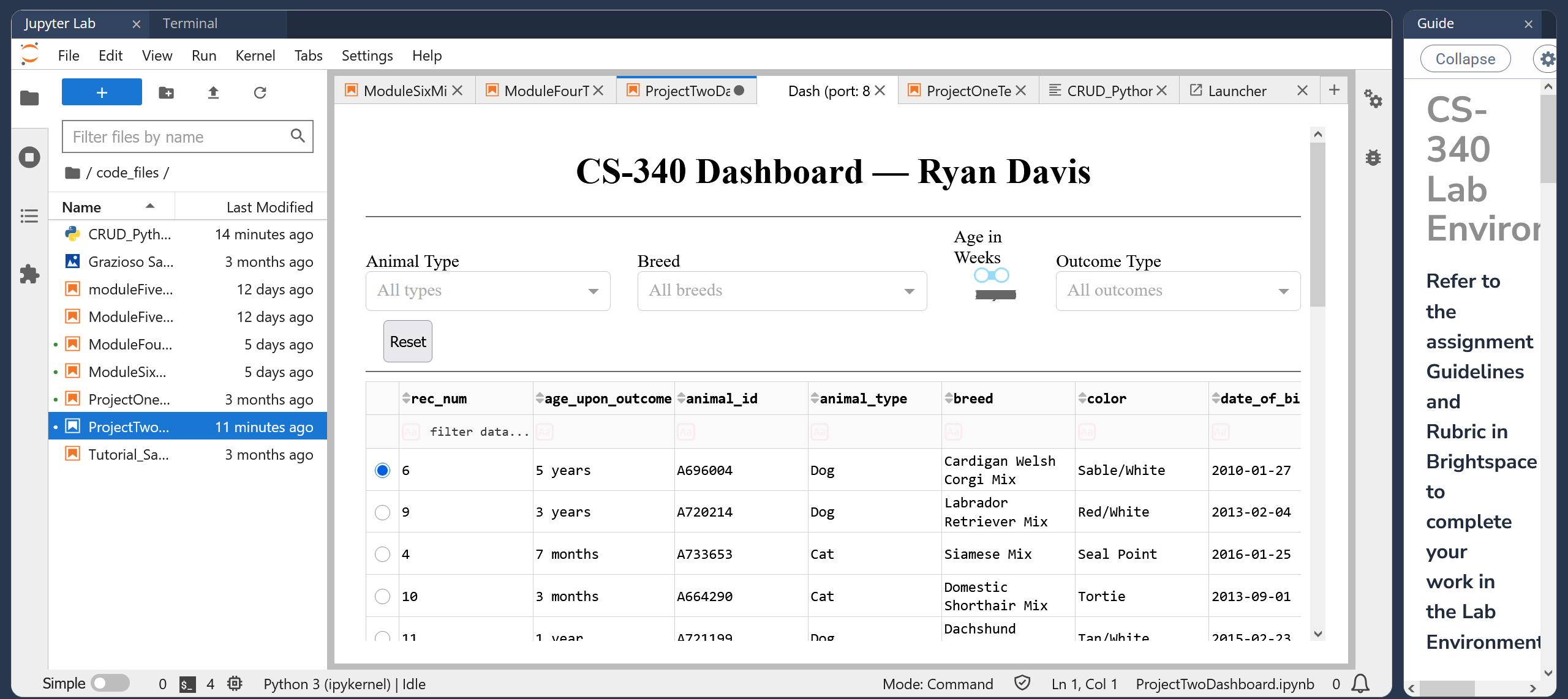
## Challenges and Solutions

One challenge was making all the filters update the dashboard at the same time. I solved this by linking each filter to the same Dash callback function. Another problem was when queries returned no results, which caused empty data tables. I fixed that by adding sample fallback data so the dashboard would still display something. Getting the MongoDB connection right also took a bit of troubleshooting with the username and password.

## Screenshots

The following screenshots should be included to show the dashboard in action:

• Starting view with all filters and the full data table.

• Water Rescue filter applied.

• Mountain or Wilderness Rescue filter applied.

• Disaster or Individual Tracking filter applied.

• Reset view showing all data again.

## Reproduction Instructions

1. Install Python and MongoDB.  
2. Load the Austin Animal Center dataset into a MongoDB database named “aac” and collection “animals.”  
3. Update the username and password in CRUD\_Python\_Module.py if needed.  
4. Run ProjectTwoDashboard.ipynb in JupyterLab.  
5. The dashboard will open in your browser or inside JupyterLab.  
6. Try out the filters and take the screenshots listed above.