# CS 340 README Template

## About the Project/Project Title

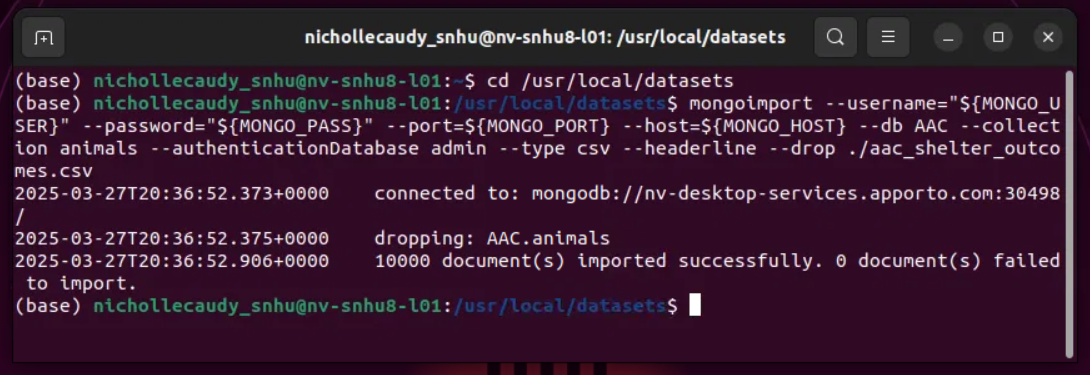
This project is to expand the knowledge and experience of using CRUD functionality in Python by implementing the create, read, update, and delete functions from the Austin Animal Center (AAC) Shelter Outcomes database. This project also includes a user-friendly interactive dashboard that uses filters to present desired information on a data table, pie chart, and a geolocation map. Grazioso Salvare required filter options for Water Rescue, Mountain or Wilderness Rescue, and Disaster Rescue or Individual Tracking to easily identify the dogs available for training in search and rescue.

## Motivation

This project is used to create a new record to the AAC database and gives the ability to read from the database by querying for a document in the AAC database. This project also allows us to update and delete information in the AAC database. Grazioso Salvare will use the existing data from animal shelters to identify and categorize available dogs that can be trained for search and rescue assignments. This means we need a way to filter through the database for specific breeds, age, and sex that have been proven most proficient in different search and rescue scenarios.

## Getting Started

* Open the command terminal, navigate to the usr/local/datasets folder, and upload the Austin Animal Center (AAC) Outcomes dataset. This will give you the host and port information that you will need to complete your code.



* Create a user account and password for the individual you wish to access the database and give permissions that will allow them to do nothing more than their job function.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

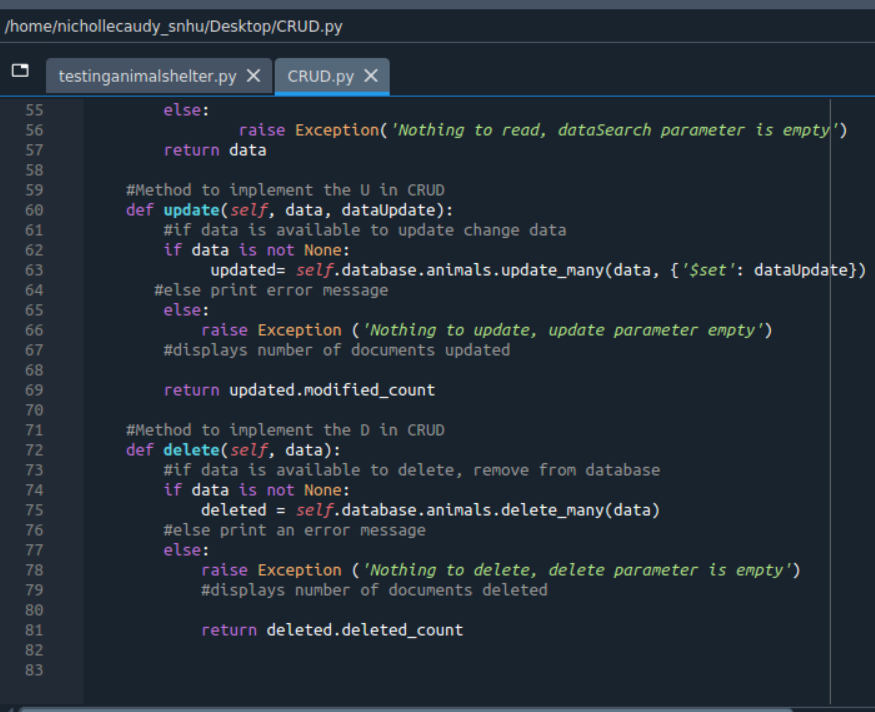
AI-generated content may be incorrect.

* To create the C (create) in CRUD I used the def call to define the function to create a new document into the Mongo DB. If the information is created successfully the code prints a message that the data was inserted correctly and the data added for the user to see. If there was an error in creating the document the user gets a message stating it was unsuccessful.
* To create the R (read) in CRUD I used the def call to define the function to read from the documents in the AAC database by using find() to search for a specific animal. Again, a message is printed with successful or unsuccessful information and the information that was found with the search.

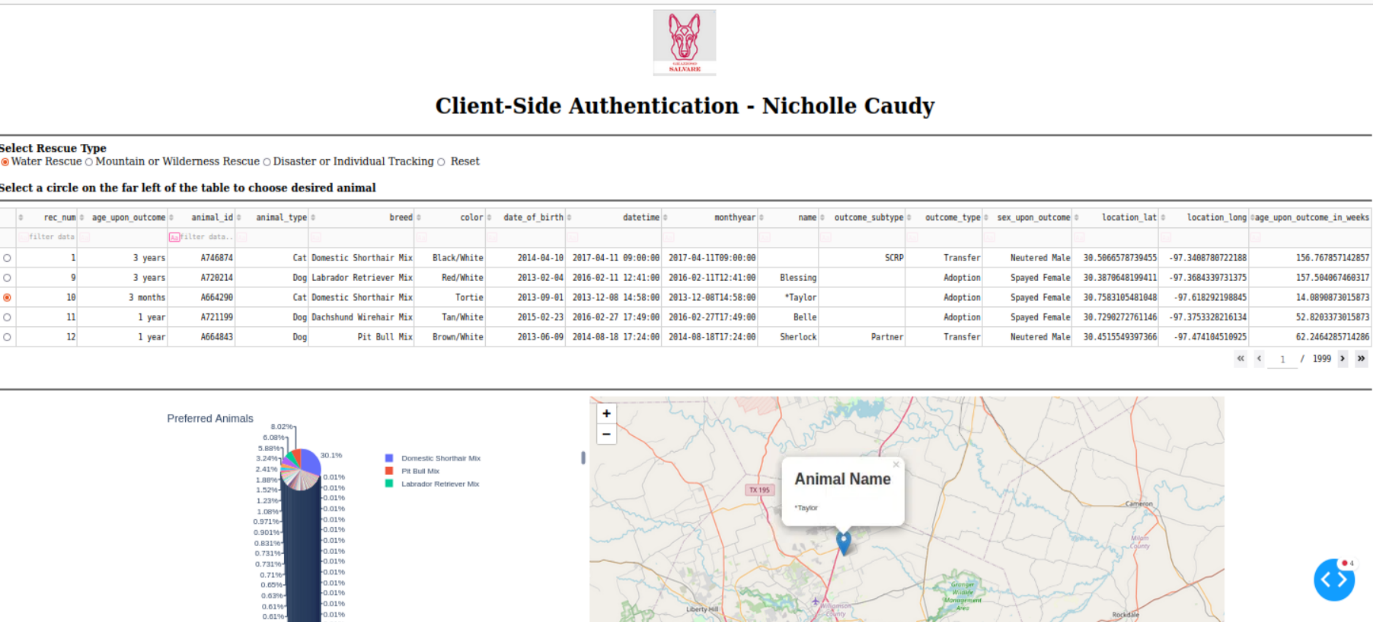
A screenshot of a computer program

AI-generated content may be incorrect.

* The functionality for the U (update) in CRUD finds the data that needs to be updated and changes the information from the user input.
* The D (delete) in CRUD finds the document that needs to be deleted and removes it from the database.



* Use Jupyter Notebook to test the code created to ensure all the functions run properly. Screenshots provided below in the Tests section.
* Create a user-friendly interactive web application dashboard using Jupyter Notebook. You will need to import your CRUD Python module and hard code the username and password you created in MongoDB for the AAC database. Using the HTML layout options for the desired look of the dashboard and appropriate ids for the data table, geolocation map, and pie chart.
* Create a filtering type, in this case I used radio items, and program the queries based on the specifications of the client (Water Rescue, Mountain or Wilderness Rescue, and Disaster Rescue or Individual Tracking).
* Create application callbacks to update the dashboard based on the user’s selections. This was challenging as I continuously received the same error of ‘UnboundLocalError: local variable 'df' referenced before assignment’ and I was unable to find the solution. The data is there but the filters do not work to update the dashboard when options are selected.



## Installation

You will need the following tools:

Python: [Download Python | Python.org](https://www.python.org/downloads/) download and install (you can use a search engine by simply typing python in the search box).

Jupyter Notebook: [Project Jupyter | Home](https://jupyter.org/) download and install (you can use a search engine to find this website, navigate to download and install or try for free).

Mongo DB: [Download MongoDB Community Server | MongoDB](https://www.mongodb.com/try/download/community?msockid=3784723d78bd659a39576763790a6431) download and install (same as above use your choice of search engine, search for Mongo DB and follow the prompts to use).

## Usage

For this project I have created code in Python for the C and R in CRUD. This can be used to create or insert a new animal into the AAC database as animals come into the shelter. You will also be able to find and read the information about the desired animal. The U and D in CRUD were developed to update and delete information in the AAC database.

### Code Example

from pymongo import MongoClient

from bson.objectid import ObjectId

class AnimalShelter(object):

"""CRUD operations for Animal collection in MongoDB"""

def \_\_init\_\_(self, AACUSER, AACPASS):  
**#Method to implement the C in CRUD**

def create (self, data):

if data is not None:

#Insert data for new animal

self.database.animals.insert(data) #data should be dictionary

print("animal successfully added") #print when the animal is added

return True

#else print error message

else:

return False

raise Exception('Nothing to save, data parameter is empty')

**#Method to implement the R in CRUD**

def read(self, data):

#using find() to search DB from an animal

if data is not None:

data = self.database.animals.find(data, {'\_id': False})

#else print error message

else:

raise Exception('Nothing to read, searchData parameter is empty')

return data

**#Method to implement the U in CRUD**

def update(self, data, dataUpdate):

#if data is available to update change data

if data is not None:

updated= self.database.animals.update\_many(data, {'$set': dataUpdate})

#else print error message

else:

raise Exception ('Nothing to update, update parameter empty')

#displays number of documents updated

return updated.modified\_count

**#Method to implement the D in CRUD**

def delete(self, data):

#if data is available to delete, remove from database

if data is not None:

deleted = self.database.animals.remove(data)

#else print an error message

else:

raise Exception ('Nothing to delete, delete parameter is empty')

#displays number of documents deleted

return deleted.deleted\_count

This is the code from the CRUD.py file to show the methods for creating and reading from the AAC database and animal collection. I had previously created an AAC user and password in Mongo Shell to allow access to the database. This code will create a new record with the animal’s information into the AAC database and allow you to find and read from a document. This code is also used to update and delete documents in the AAC database.

### Tests

Running the tests, I needed to pull the information and code from the CRUD.py file using:

*from CRUD import AnimalShelter*

Then entering the data for a new animal to be added to the AAC database

data = {'rec\_num':'test number',

'age\_upon\_outcome':'3 years',

'animal\_id': 'test ID',

'animal\_type': 'Dog',

'breed': 'Boxer',

'color': 'brown',

'name': 'Tyson'}

I then used if and else statements to verify that the data was added successfully, or it failed.

#if the add is successful print the data added

if animals.create(data):

print (data)

#else print an error message

else:

print ('add unsuccessful')

To test the R in CRUD was a little shorter, I searched for the name of a dog, Sapphire. I knew this record existed from a previous assignment to check that the code worked and searched the database properly and returned the right information.

read = animals.read({'name': 'Sapphire'})

#print the doc

for animal in read:

print(animal)

To test the U in CRUD we need to find the data and then update the relevant information

Update = animals.update({‘name’ = ‘Sapphire’}, {‘name’: ‘Sophie’})

print(update)

Testing for the delete functionality:

delete = animals.delete({‘name’: ‘Tyson’})

print(delete)

### Screenshots

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

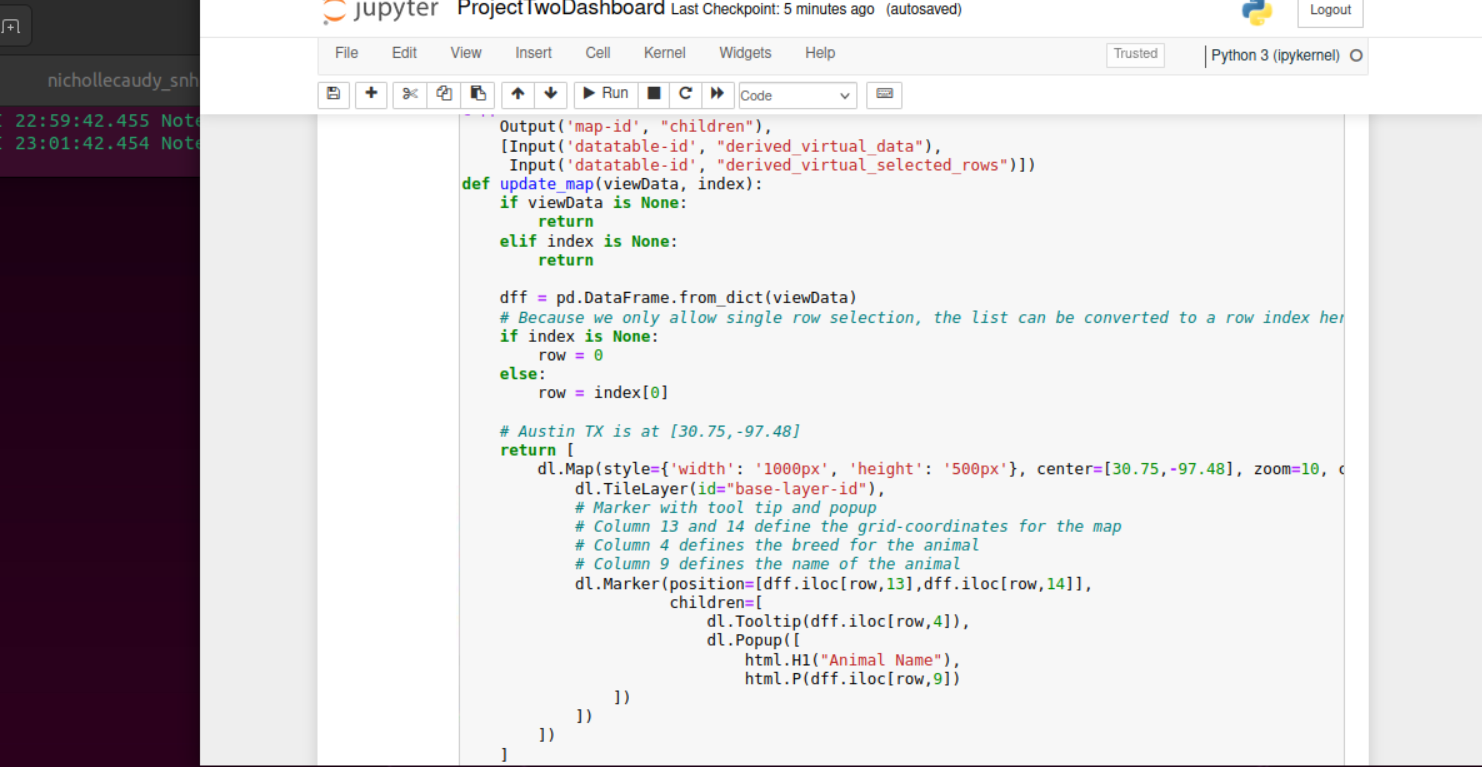
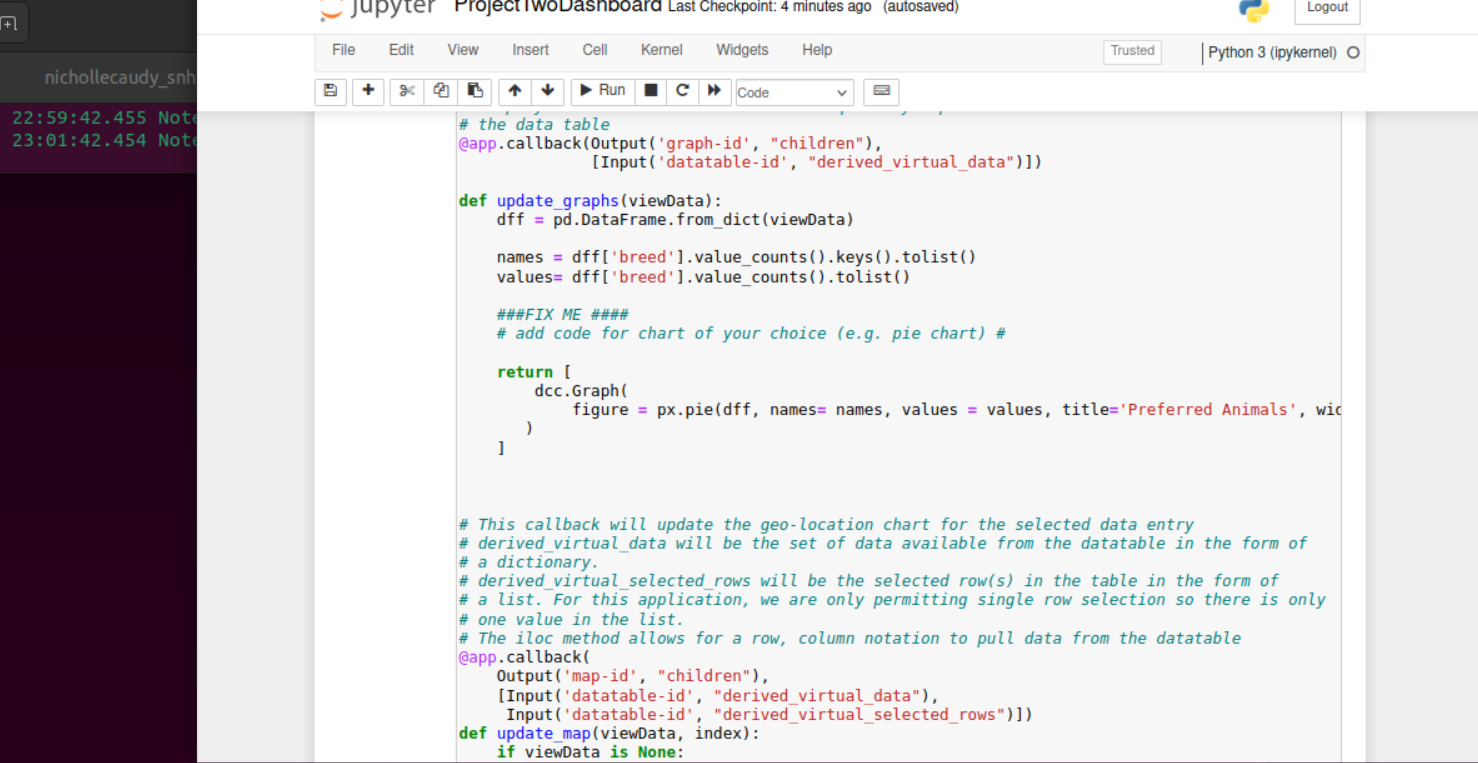
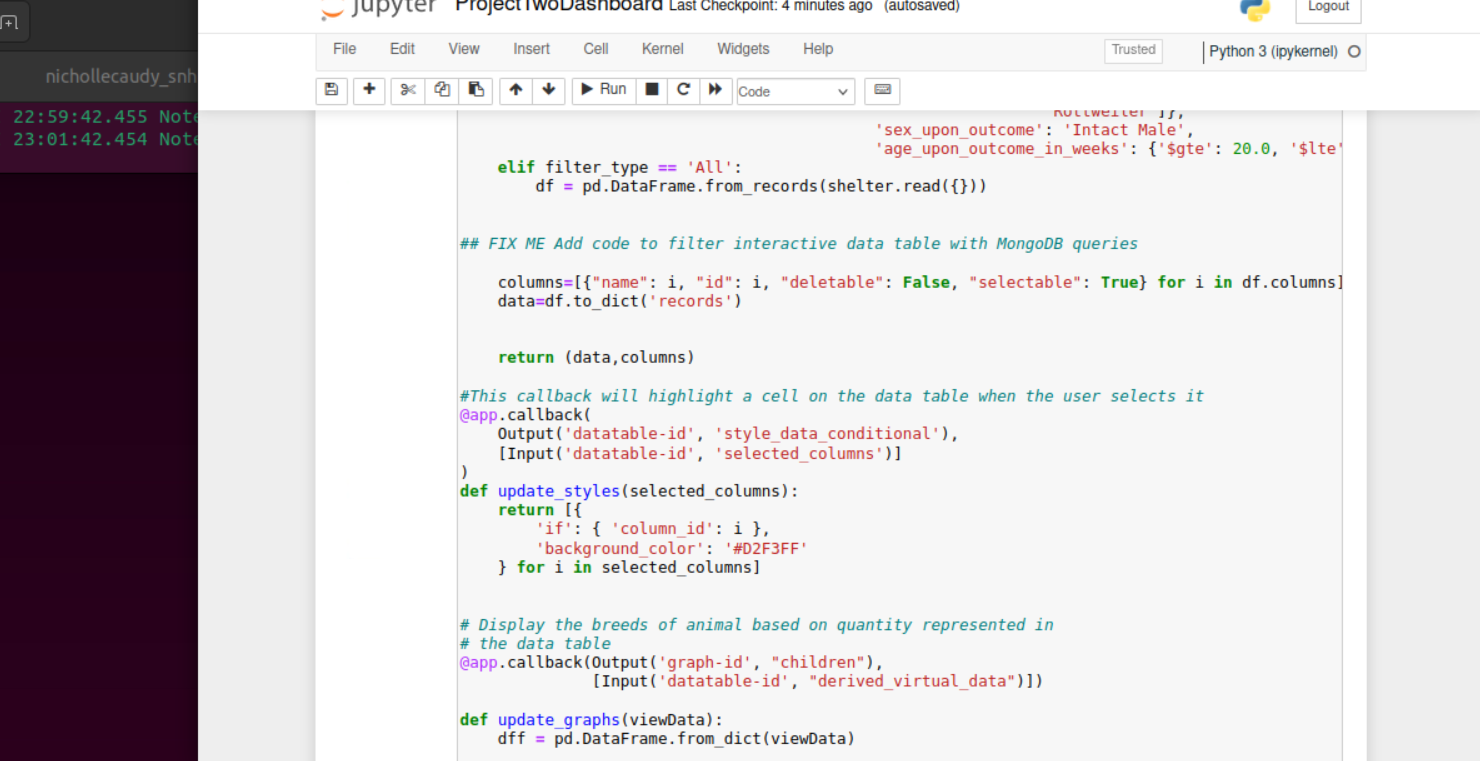
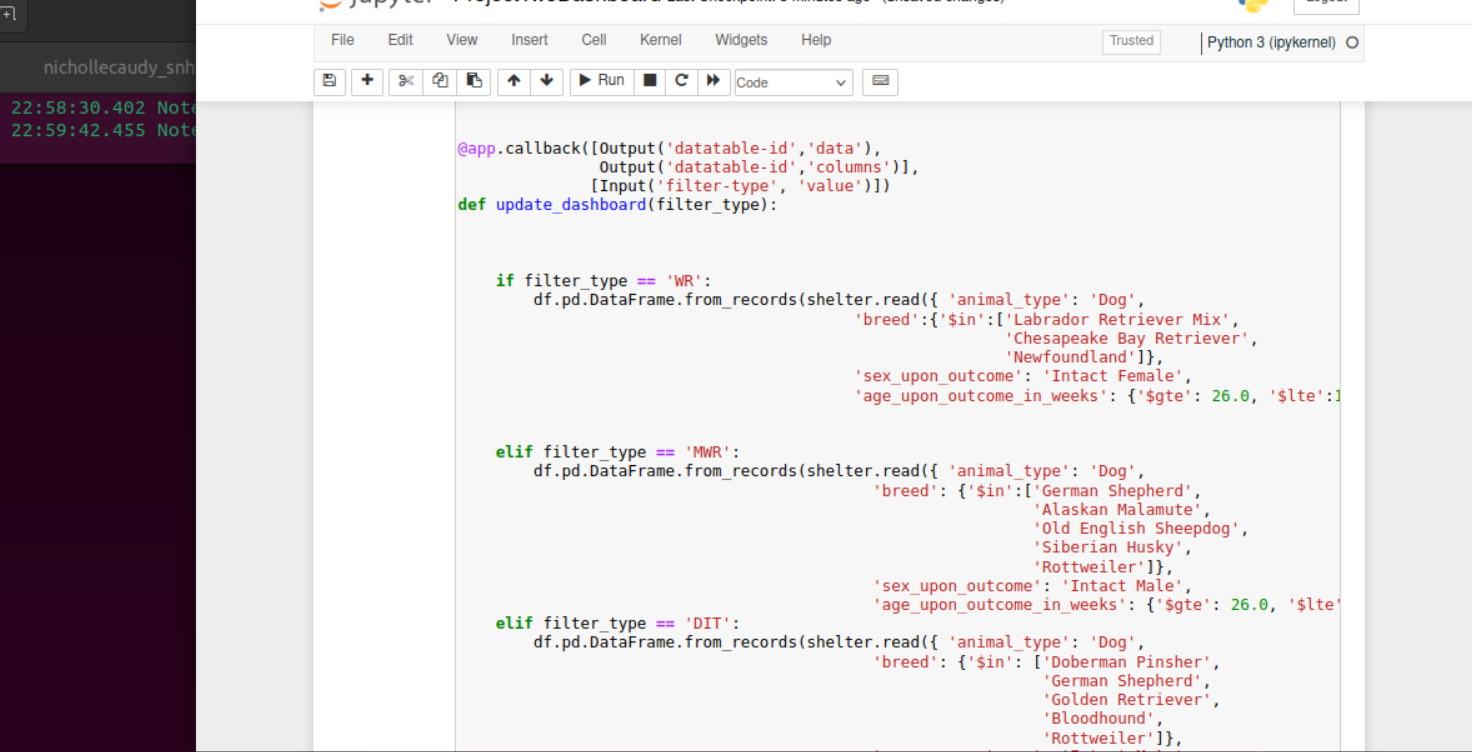
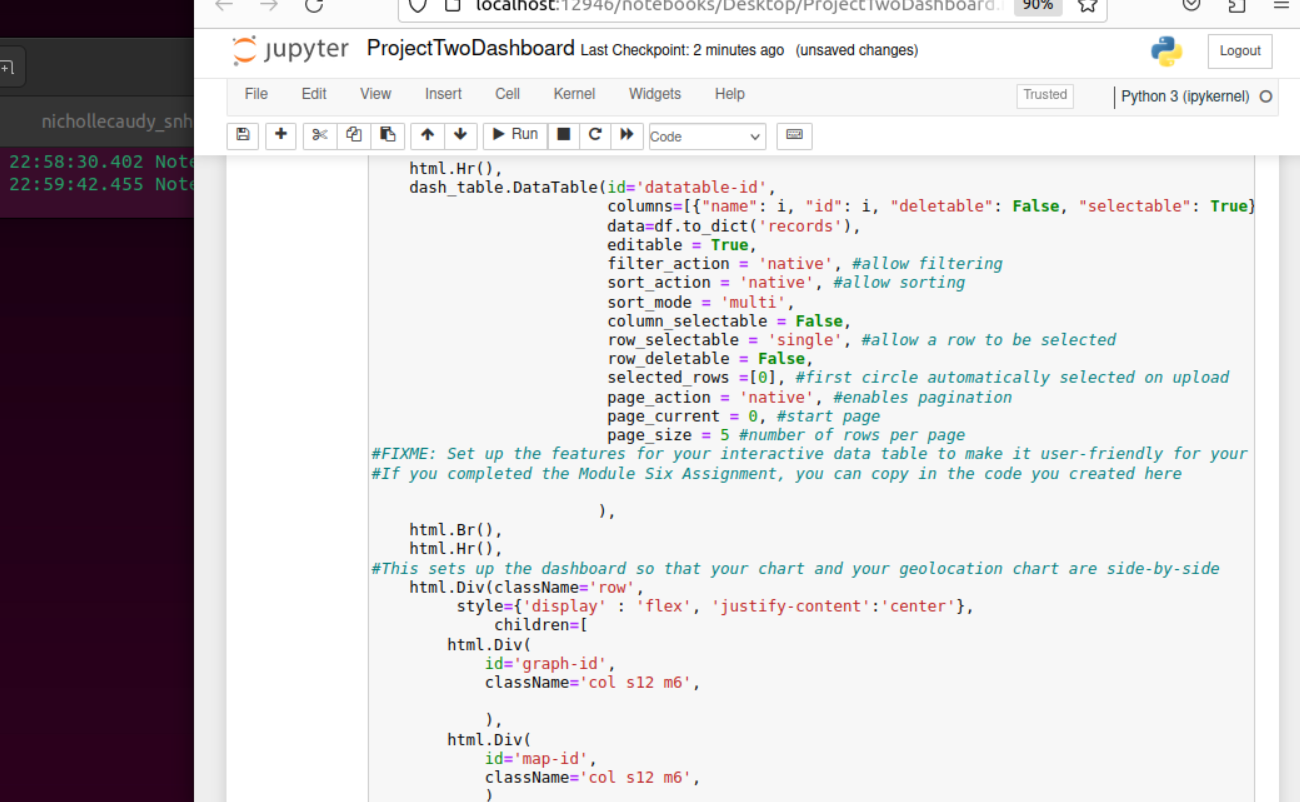
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

## Contact

Your name: Nicholle Caudy