# CS 340 README Ethan Cook

## About the Project/Project Title

Final Project

## Motivation

*We are creating a Python file and a IPYNB file as a testing script.*

## Getting Started

*To run this program, upload into Jupyter Notebook, then create a notebook and execute the calls and methods to ensure database interactions*

## Installation

*To run this, the OS needs MongoDB, Pymongo, and a web browser to access Jupyter Notebook.*

## Usage

*The usage of this code is intended to be manipulating databases through a python file. It also supports web based Dash application of the filtering of output data. One can search in any column for specific data and it will filter as such. I created the Rainbow color to show the filtering capabilities.*

### Code Example

*This code example is brief but I was able to get both create and delete working correctly. We initialized the mongoclient, and defined CRUD methods. For example,*

*def \_\_init(self, username, password):*

*self.client = MongoClient(‘mongodb//%s:%s@localhost:51019’ % (aacuser, Password))*

*self.database = self.client[‘AAC’]*

*This connects the python file to the mongo database through port 51019.*

*def create(self, insertdata):*

*if insertdata is not None:*

*self.database.animals.insert(insertdata)*

*return True*

*Would create the create method. Delete method below.*

*#Delete in Crud*

*def delete(self, animalRecord):*

*# Get deletion record*

*if animalRecord is not None:*

*deleteRecord = self.database.animals.delete\_one(animalRecord)*

*return deleteRecord*

*else:*

*return("Deletion failed")*

*To run the tests you use the methods created starting with the class. admin.create(insertdata) should add the data provided you initialized insertdata.*

*Admin.delete(animalRecord) deleted the first one created.*

### ## Dashboard Code

app.layout = html.Div([\n",

" html.Div(id='hidden-div', style={'display':'none'}),\n",

" html.Center(html.B(html.H1('Grazioso AAC Data Ethan Cook'))),\n",

" html.Hr(),\n",

" html.Img(id='customer-image',src='data:image/png;base64,{}'.format(encoded\_image.decode()),alt='customer image', style={'height':'10%', 'width':'10%'}),\n",

" html.Div(className='row',\n",

" style={'display' : 'flex'},\n",

" children=[\n",

" html.Button(id='submit-button-one', n\_clicks=0, children='Water Rescue'),\n",

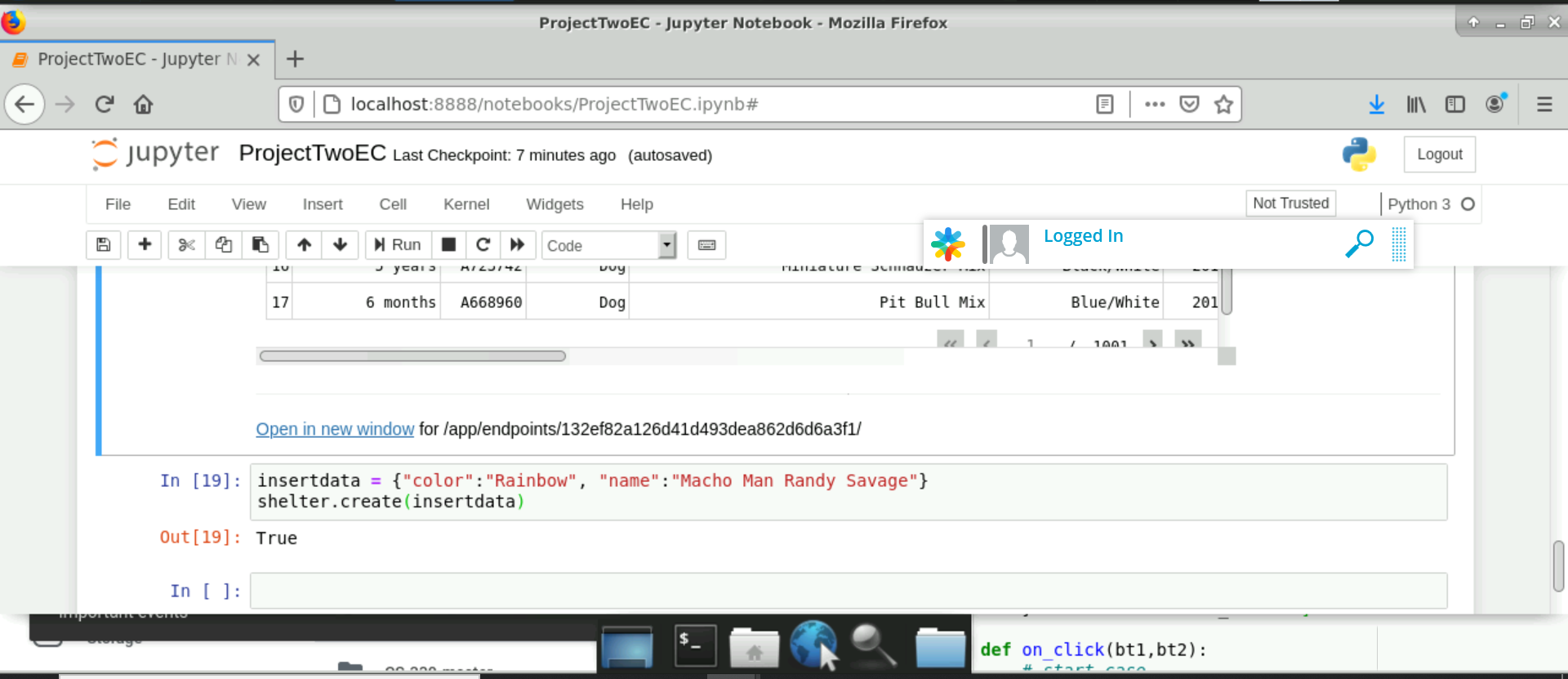
" html.Button(id='submit-button-two', n\_clicks=0, children='Mountain or Wilderness'),\n",

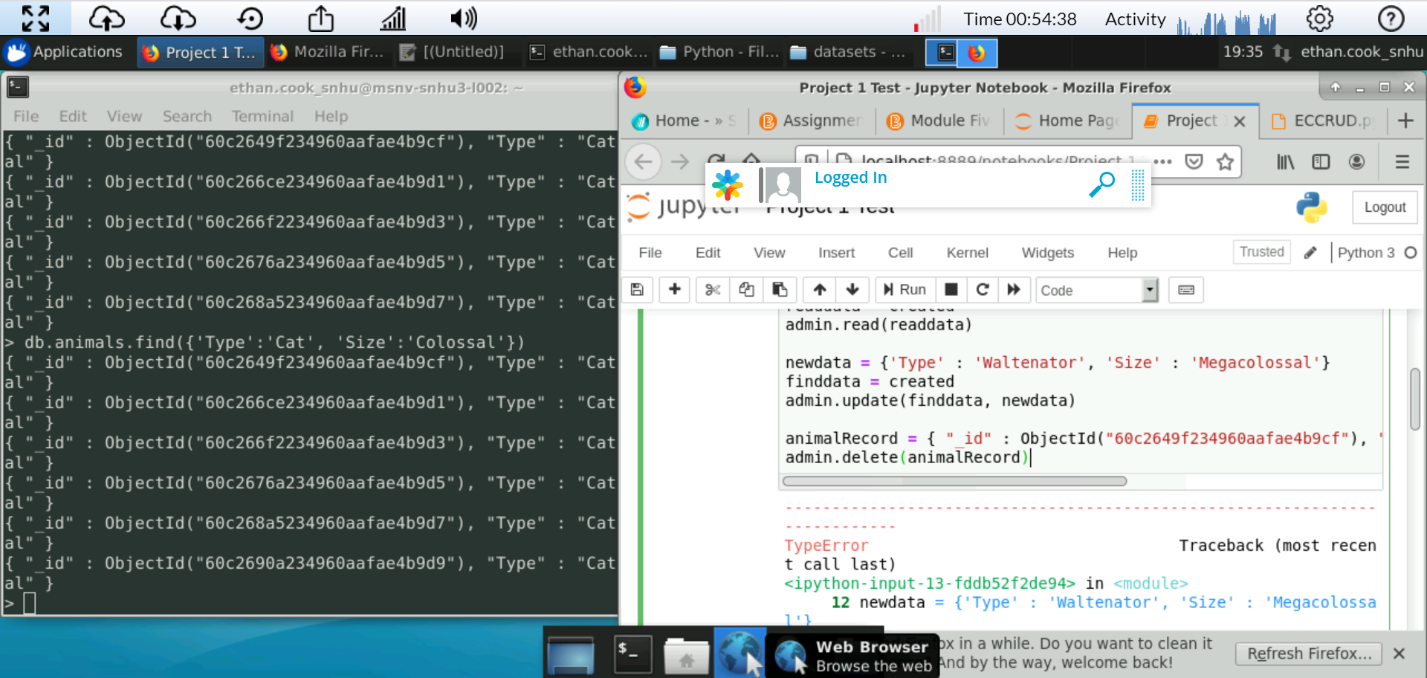
" html.Button(id='submit-button-three', n\_clicks=0, children='Disaster or Individual Tracking'),\n",

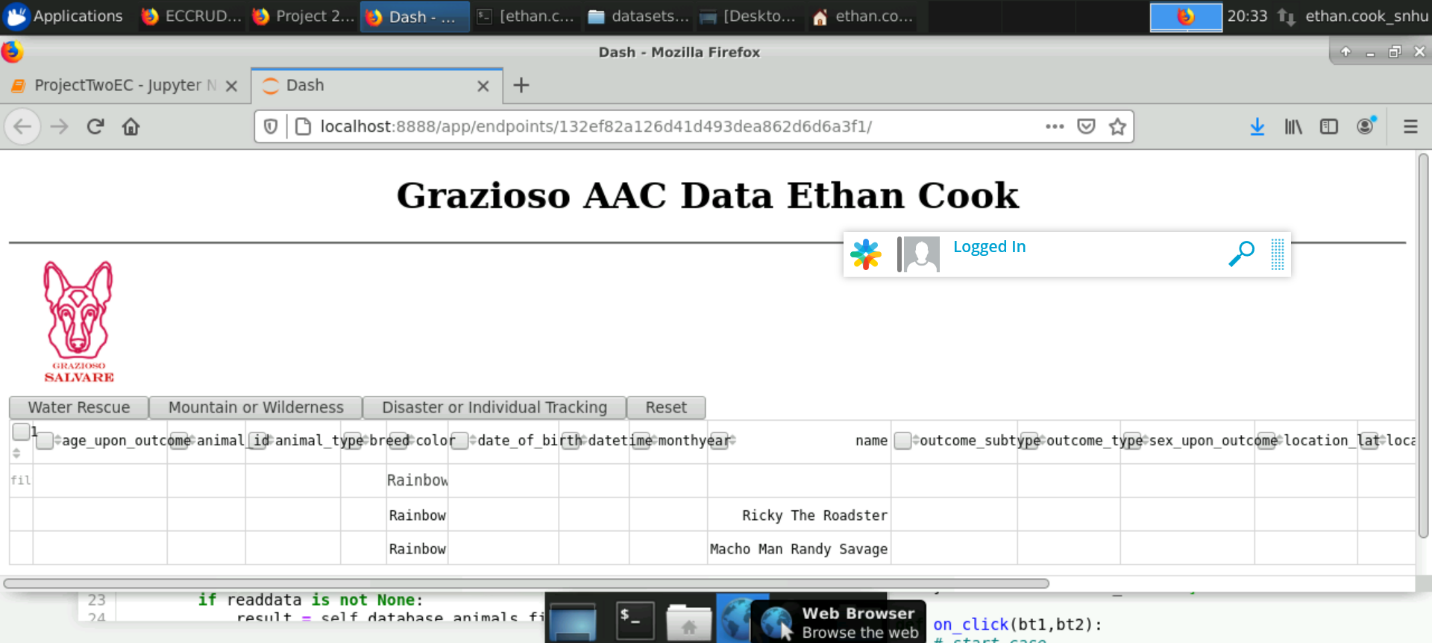
" html.Button(id='submit-button-four', n\_clicks=0, children='Reset')\n",

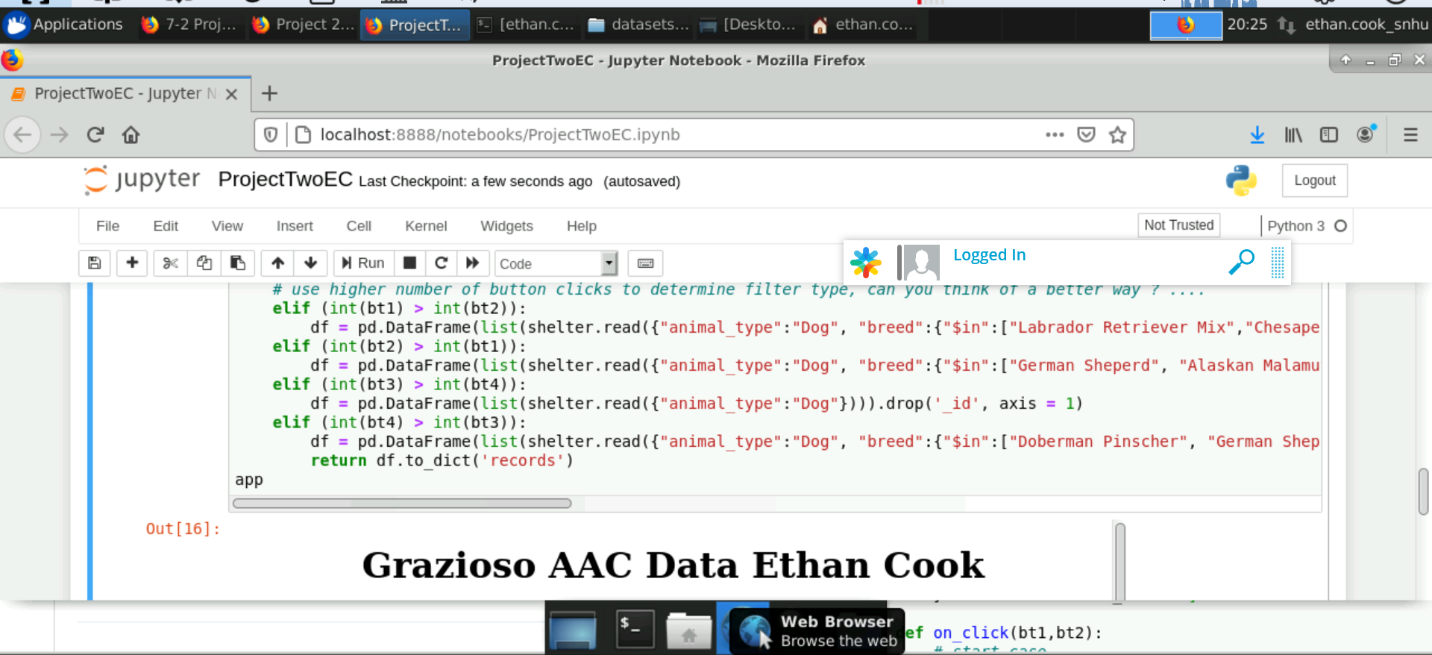
This bit of code helps to create the dash application that can be seen in the screenshots below.

### Screenshots









## Roadmap/Features (Optional)

*The most important roadmap/feature is to get the code connected to the Mongoclient like we did, and then call the methods from the Animal Shelter class. We have successful filtering, but unable to filter per the specific rescue definitions. We will continue to work to make the application tailored to your needs.*

## Contact

Ethan Cook