CS 340 README – Creating an Interactive Dashboard

About the Project/Project Title

In this project a dashboard will be created using imported Austin Animal Center data retrieved from MongoDB database and manipulated by the CRUD operations that were designed in a PY file. An IPYNB file will be created to import the PY file so that functions can be written within the that file to manipulate the data and design a user- friendly interactive dashboard web browser.

Motivation

My personal motivation to create this project was to gain a better understanding of how to use the MongoDB Database and all the tools that it has to offer to handle data sets. It is possible to simply use a MongoDB shell to handle and interpret data, but a more efficient way to do so is through another IDE that is compatible with MongoDB. Not only does this project offer more experience with MongoDB but with Jupyter Notebook and PyMongo as well. By using PyMongo, I have been able to learn about how to create a user-friendly web browser full of nifty, functional, widgets to manipulate data with ease. Creating a module with PyMongo gives someone with little to no computer experience or no experience with MongoDB a better opportunity to manipulate a desired dataset as well. The creation of the dashboard has taught me much more about python syntax, python style coding, coding structure, proper dashboard designs, debugging practices, functional code logic, algorithmic design and so much more.

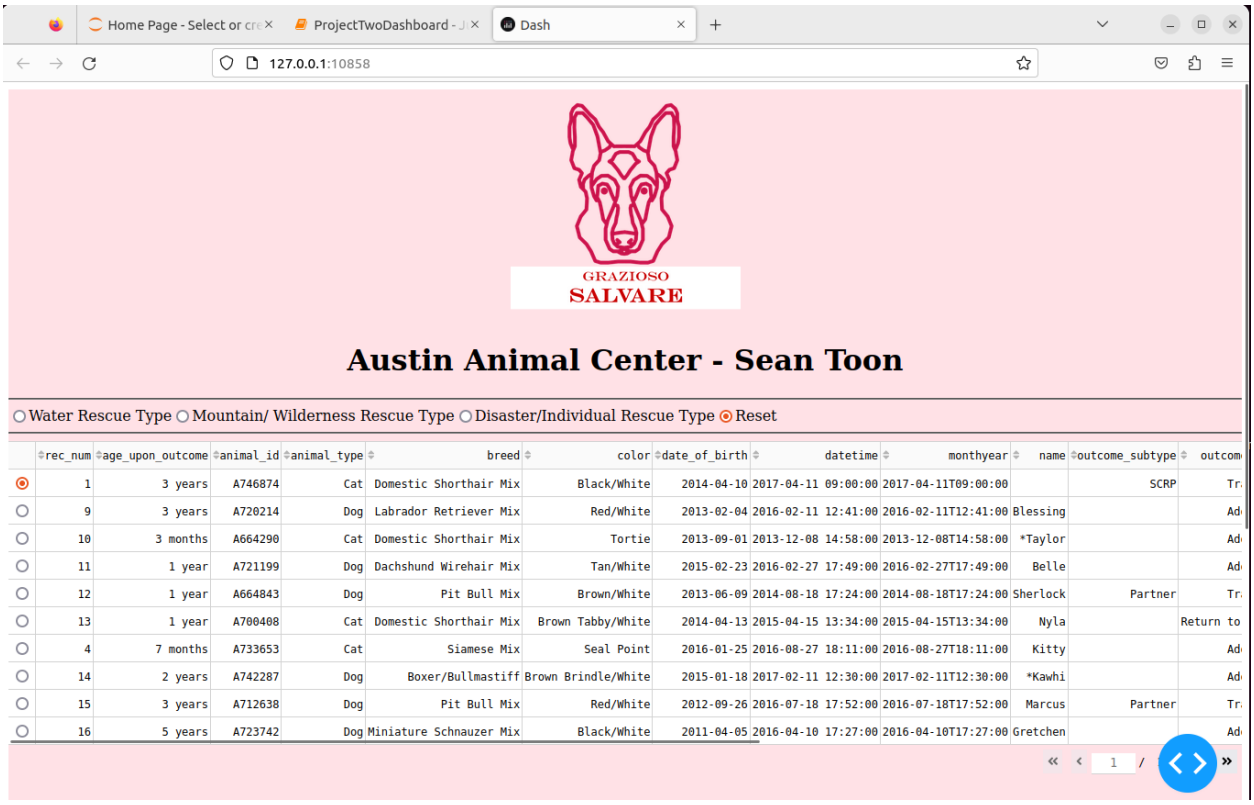
Installation

Setting up the tools for the project is simple. Instructions to download the free community MongoDB package can be accessed at https://www.mongodb.com/try/download/community. There are many ways to access Jupyter Notebook and there are many resources online to get instructions to do so. One way is at https://www.anaconda.com/download. This documentation of this particular section of the project only covers the creation of the IPYNB file so these instructions assume that a PY file is created, functional and ready to use!

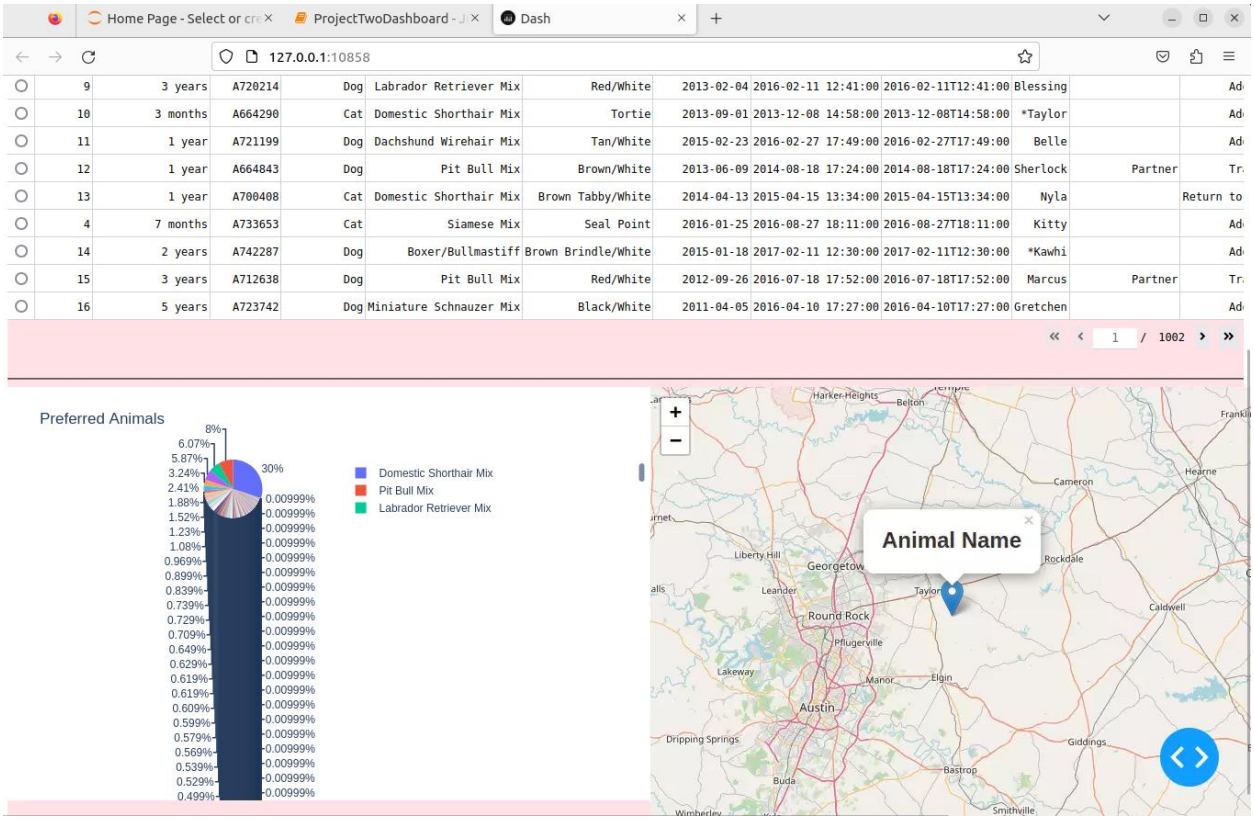
Usage

To start the project, a PY file should be created and should contain all the CRUD operations needed to successfully Create, Read, Update and Delete any of the data that is being handled within MongoDB. In this case the data is a collection named “animals”, found in the “AAC” database. It is good practice to make sure that the database exists and that it can be properly imported into the Mongosh shell.

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.



	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcom
◉	1	3 years	A746874	Cat	Domestic Shorthair Mix	Black/White	2014-04-10	2017-04-11 09:00:00	2017-04-11T09:00:00		SCRP	Tr
○	9	3 years	A720214	Dog	Labrador Retriever Mix	Red/White	2013-02-04	2016-02-11 12:41:00	2016-02-11T12:41:00	Blessing		Ad
○	10	3 months	A664290	Cat	Domestic Shorthair Mix	Tortie	2013-09-01	2013-12-08 14:58:00	2013-12-08T14:58:00	*Taylor		Ad
○	11	1 year	A721199	Dog	Dachshund Wirehair Mix	Tan/White	2015-02-23	2016-02-27 17:49:00	2016-02-27T17:49:00	Belle		Ad
○	12	1 year	A664843	Dog	Pit Bull Mix	Brown/White	2013-06-09	2014-08-18 17:24:00	2014-08-18T17:24:00	Sherlock	Partner	Tr
○	13	1 year	A700408	Cat	Domestic Shorthair Mix	Brown Tabby/White	2014-04-13	2015-04-15 13:34:00	2015-04-15T13:34:00	Nyla		Return to
○	4	7 months	A733653	Cat	Siamese Mix	Seal Point	2016-01-25	2016-08-27 18:11:00	2016-08-27T18:11:00	Kitty		Ad
○	14	2 years	A742287	Dog	Boxer/Bullmastiff	Brown Brindle/White	2015-01-18	2017-02-11 12:30:00	2017-02-11T12:30:00	*Kawhi		Ad
○	15	3 years	A712638	Dog	Pit Bull Mix	Red/White	2012-09-26	2016-07-18 17:52:00	2016-07-18T17:52:00	Marcus	Partner	Tr
○	16	5 years	A723742	Dog	Miniature Schnauzer Mix	Black/White	2011-04-05	2016-04-10 17:27:00	2016-04-10T17:27:00	Gretchen		Ad



	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype	outcom
○	9	3 years	A720214	Dog	Labrador Retriever Mix	Red/White	2013-02-04	2016-02-11 12:41:00	2016-02-11T12:41:00	Blessing		Ad
○	10	3 months	A664290	Cat	Domestic Shorthair Mix	Tortie	2013-09-01	2013-12-08 14:58:00	2013-12-08T14:58:00	*Taylor		Ad
○	11	1 year	A721199	Dog	Dachshund Wirehair Mix	Tan/White	2015-02-23	2016-02-27 17:49:00	2016-02-27T17:49:00	Belle		Ad
○	12	1 year	A664843	Dog	Pit Bull Mix	Brown/White	2013-06-09	2014-08-18 17:24:00	2014-08-18T17:24:00	Sherlock	Partner	Tr
○	13	1 year	A700408	Cat	Domestic Shorthair Mix	Brown Tabby/White	2014-04-13	2015-04-15 13:34:00	2015-04-15T13:34:00	Nyla		Return to
○	4	7 months	A733653	Cat	Siamese Mix	Seal Point	2016-01-25	2016-08-27 18:11:00	2016-08-27T18:11:00	Kitty		Ad
○	14	2 years	A742287	Dog	Boxer/Bullmastiff	Brown Brindle/White	2015-01-18	2017-02-11 12:30:00	2017-02-11T12:30:00	*Kawhi		Ad
○	15	3 years	A712638	Dog	Pit Bull Mix	Red/White	2012-09-26	2016-07-18 17:52:00	2016-07-18T17:52:00	Marcus	Partner	Tr
○	16	5 years	A723742	Dog	Miniature Schnauzer Mix	Black/White	2011-04-05	2016-04-10 17:27:00	2016-04-10T17:27:00	Gretchen		Ad

Note: This template has been adapted from the following sample templates: [Make a README](), [Best README Template](), and [A Beginners Guide to Writing a Kickass README]().

Above are pictures of how the web browser should look after all the interactive options to filter data are added. The first interactive widget that was created is the Grazioso Salvare Logo located at the top of the dashboard. If the logo is clicked the user is directed to www.snhu.edu. I added four different radio buttons that the user can click on to filter the data within each of those separate parameters. There are also buttons on the sides of each row that when clicked, display the location and animal name of the data in the row on the geolocation chart below the data table. Below the data table is a pie chart and a geolocation chart to display data in response to user updates.

Code Example

To create a standard dashboard with no interactive filter options except for radio buttons to display data like any code you need the correct imports:

The PY file being used in this project is titled aac_lib.

```
# Setup the Jupyter version of Dash
from jupyter_dash import JupyterDash

# Configure the necessary Python module imports
import dash_leaflet as dl
from dash import dcc
from dash import html
import plotly.express as px
from dash import dash_table
from dash.dependencies import Input, Output


# Configure the plotting routines
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt


#### FIX ME #####
# change animal_shelter and AnimalShelter to match your CRUD Python module file name and clas
from aac_lib import AnimalShelter
```

User authentication is needed for any of the functions within the CRUD module (aac_lib) to work. In this case the user info is coded in aac_lib.

Data manipulation:

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

```
shelter = AnimalShelter()


# class read method must support return of list object and accept projection json input
# sending the read method an empty document requests all documents be returned
df = pd.DataFrame.from_records(shelter.read({}))

# MongoDB v5+ is going to return the '_id' column and that is going to have an
# invlaid object type of 'ObjectID' - which will cause the data_table to crash - so we remove
# it in the dataframe here. The df.drop command allows us to drop the column. If we do not se
# inplace=True - it will reeturn a new dataframe that does not contain the dropped column(s)
df.drop(columns=['_id'],inplace=True)
```

Next is the layout of the Dashboard:

```
#########################
# Dashboard Layout / View
#########################
app = JupyterDash('SimpleExample')
#unique identifier from mod 5
background_color = "#F0F8FF"

app.layout = html.Div(
    style={'backgroundColor': background_color},
    children=[
        html.Center(html.B(html.H1('Austin Animal Center - Sean Toon'))),
        html.Hr(),
        dash_table.DataTable(
            id='datatable-id',
            columns=[
                {"name": i, "id": i, "deletable": False, "selectable": True} for i in df.columns
        ],
        data=df.to_dict('records'),
        #FIXME: Set up the features for your interactive data table to make it user-friendly for your client
        page_size=10, #set number of rows per page
        sort_action='native', #enable sorting
        style_table={'overflowX': 'scroll'}, #add scroll to list
        row_selectable='single', #single-row selection
        selected_rows = [0]
    ),
    html.Br(),
    html.Hr(),
    html.Div(
        id='map-id',
        className='col s12 m6'),
        ]
)
```

Then to update the map and run the server:

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

```
# one value in the list.
# The iloc method allows for a row, column notation to pull data from the datatable
@app.callback(
    Output('map-id', "children"),
    [Input('datatable-id', "derived_virtual_data"),
     Input('datatable-id', "derived_virtual_selected_rows")]
)

def update_map(viewData, index):
#FIXME Add in the code for your geolocation chart

    dff = pd.DataFrame.from_dict(viewData)

    if dff.empty: #check to see if dff is empty
        return []

    if index is None:
        row = 0
    else:
        row = index[0]

    return[
        dl.Map(style={'width': '1000px', 'height': '500px'},
               center=[30.75, -97.48], zoom=10, children=[
                    dl.TileLayer(id="base_layer_id"),
                    dl.Marker(position=[dff.iloc[row,13], dff.iloc[row,14]],
                              children=[
                                  dl.Tooltip(dff.iloc[row,4]),
                                  dl.Popup([
                                      html.H1("Animal Name"),
                                      html.P(dff.iloc[row,9])
                                  ])
                              ])
                ])
    ]

app.run_server(debug=True)
```

Once a simple dashboard is created, it is easy to edit the functions within the code to accept interactive filtering options. New functions will also be created to update/create the pie chart.

Before creating any filtering options, it is important to know what values label the columns of the data that is being read. For example, an “id” is created for each data entry in the collection, but the “id” is not needed for the purposes of the filtering options that are going to be created. So, before an filtering is applied, it is necessary to drop the column “id” like so:

```
# drop column['_id'] to avoid invalid object crash
df.drop(columns=['_id'],inplace=True)
```

To import the Grazioso Salvare Logo, link the SNHU website to the logo and display the radio buttons that will update the data, the layout code needs to be updated:

(Downloadable Grazioso Salvare Logo)

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

```
########################
# Dashboard Layout / View
########################
app = JupyterDash(__name__)

#Add in Grazioso Salvare's logo
image_filename = '/home/seantoon_snhu/Desktop/Grazioso Salvare Logo2.png'
encoded_image = base64.b64encode(open(image_filename, 'rb').read())

#Unique Identifier and attach www.snhu.edu to photo
app.layout = html.Div(
    style={'backgroundColor': '#FFE1E6'},  # Set background color here
    children=[
        html.A([
            html.Center(html.Img(
                src='{}'.format(encoded_image.decode()),
                height=250, width=251))], href='https://www.snhu.edu', target="_blank"),
        html.Center(html.B(html.H1('Austin Animal Center - Sean Toon'))),
        html.Hr(),
        dcc.RadioItems(
            id='filter-type',
            options=[
                {'label': 'Water Rescue Type', 'value': 'Water Rescue Type'},
                {'label': 'Mountain/ Wilderness Rescue Type', 'value': 'Mountain/ Wilderness Rescue Type'},
                {'label': 'Disaster/Individual Rescue Type', 'value': 'Disaster/Individual Rescue Type'},
                {'label': 'Reset', 'value': 'Reset'}
            ],
            value='Reset'
    ),
```

```
    ),
    html.Hr(),
    dash_table.DataTable(
        id='datatable-id',
        columns=[{"name": i, "id": i, "deletable": False, "selectable": True} for i in df.columns],
        data=df.to_dict('records'),
        #Set up the features for interactive data table
        page_size=10, #set number of rows per page
        page_action="native",
        sort_action="native", #enable sorting
        style_table={'overflowX': 'scroll'}, #add scroll to list
        row_selectable='single', #single-row selection
        selected_rows = [0]

    ),
    html.Br(),
    html.Hr(),

    #Set up pie chart and geolocation chart side-by-side
    html.Div(className='row',
         style={'display' : 'flex'},
             children=[
        html.Div(
            id='graph-id',
            className='col s12 m6',

            ),
        html.Div(
            id='map-id',
            className='col s12 m6',
            )
        ])
])
```

A function to update the data table based on the filtering options needs to be created to meet the following criteria:

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

Rescue Type	Preferred Breeds	Preferred Sex	Training Age*
Water	Labrador Retriever Mix, Chesapeake Bay Retriever, Newfoundland	Intact Female	26 weeks to 156 weeks
Mountain or Wilderness	German Shepherd, Alaskan Malamute, Old English Sheepdog, Siberian Husky, Rottweiler	Intact Male	26 weeks to 156 weeks
Disaster or Individual Tracking	Doberman Pinscher, German Shepherd, Golden Retriever, Bloodhound, Rottweiler	Intact Male	20 weeks to 300 weeks

The code that matches the criteria:

```
#############################################
# Interaction Between Components / Controller
#############################################

@app.callback(
    [Output('datatable-id', 'data'),
     Output('datatable-id', 'columns')],
    [Input('filter-type', 'value')]
)
def update_dashboard(filter_type):
    # Filter interactions
    if filter_type == 'Water Rescue Type':
        df = pd.DataFrame.from_records(db.read({
            "animal_type": "Dog",
            "breed": {"$in": ["Labrador Retriever Mix","Chesapeake Bay Retriever", "Newfoundland"
                              ]},
            "sex_upon_outcome": "Intact Female",
            "age_upon_outcome_in_weeks": {"$gte":26.0, "$lte":156.0}
        }))

    elif filter_type == 'Mountain/ Wilderness Rescue Type':
        df = pd.DataFrame.from_records(db.read({
            "animal_type": "Dog",
            "breed": {"$in": ["German Shepard","Alaskan Malamute","Old English Sheepdog",
                              "Siberian Husky", "Rottweiler"
                              ]},
            "sex_upon_outcome": "Intact Male",
            "age_upon_outcome_in_weeks": {"$gte":26.0, "$lte":156.0}
        }))

    elif filter_type == 'Disaster/Individual Rescue Type':
        df = pd.DataFrame.from_records(db.read({
            "animal_type": "Dog",
            "breed": {"$in": ["Doberman Pinscher","German Shepard","Golden Retriever",
                              "Bloodhound","Rottweiler"
                              ]},
            "sex_upon_outcome": "Intact Male",
            "age_upon_outcome_in_weeks": {"$gte":20.0, "$lte":300.0}
        }))
```

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

```
# Filter reset
else:
    df = pd.DataFrame.from_records(db.read({}))
df.drop(columns=['_id'],inplace=True)
columns=[{"name": i, "id": i, "deletable": False, "selectable": True} for i in df.columns]
data=df.to_dict('records')

    #Debug
    #print(df.columns)
    #print(df['breed'].unique())
    #print(df[df['breed'] == 'Chesapeake Bay Retriever'])

return (data,columns)
```

Next, a function to update the pie graph needs to be created to pass the new filters:

```
# Display the breeds of animal based on quantity represented in
# the data table
@app.callback(
    Output('graph-id', "children"),
    [Input('datatable-id', "derived_virtual_data")])

def update_graphs(viewData):
    dff= pd.DataFrame.from_dict(viewData)
    return [

        dcc.Graph(
            figure = px.pie(dff, names='breed', title='Preferred Animals')
        )
    ]
```

Tests

Most of the tests done on the code created were to debug errors that arose when I would run the dash app. The data was challenging to filter because the data in the data frame was so messy when read. It was a tedious job to match the exact spelling and capitalization of each data identification value. Some of the statements that are still commented within my code (most I deleted), show my efforts to gain insight about indexing and values within post and the updated data_frame. Some examples:

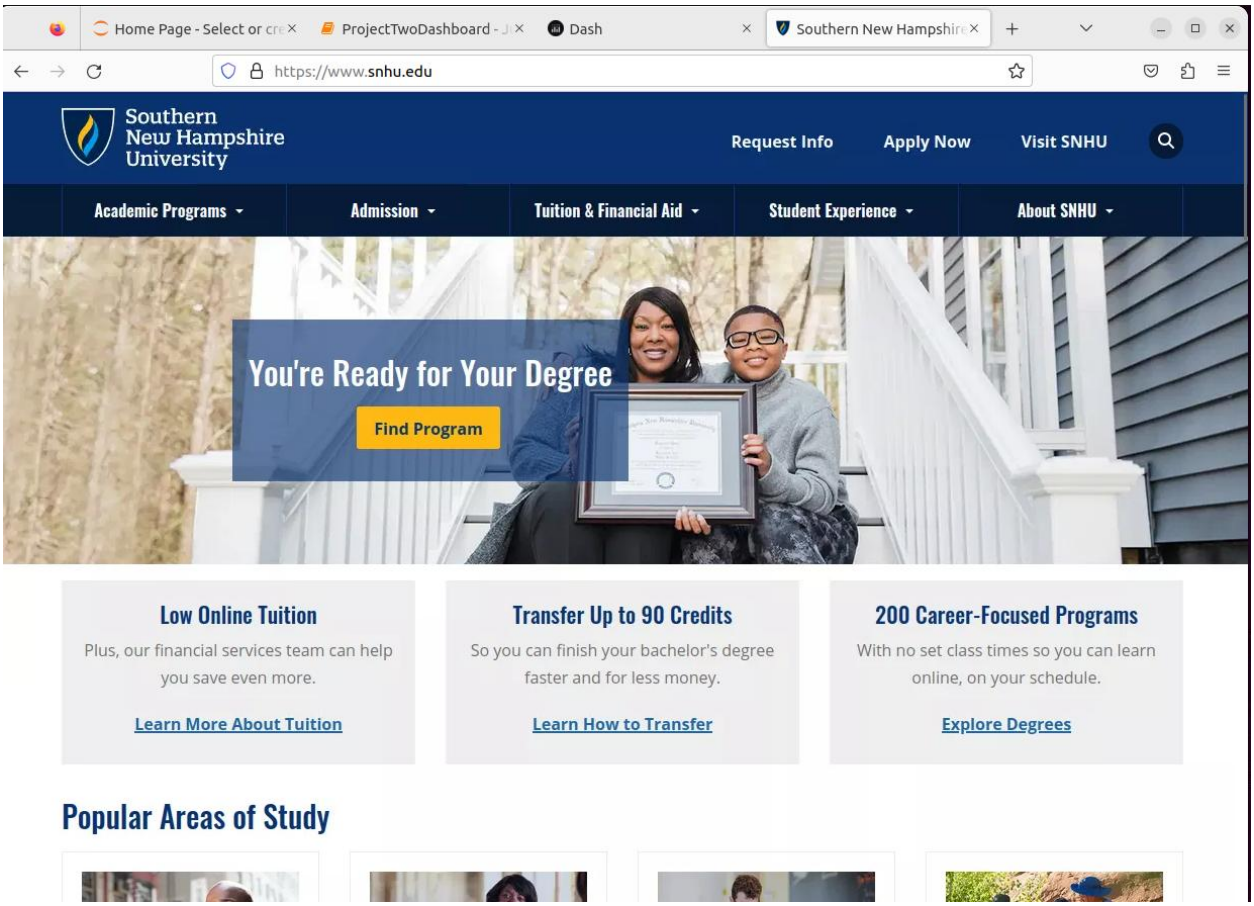
```
#Debug
#print(df.columns)
#print(df['breed'].unique())
#print(df[df['breed'] == 'Chesapeake Bay Retriever'])
```

```
## Debug
#print(len(df.to_dict(orient='records')))
#print(df.info())
#print(df.head())
```

Screenshots

Link attached to the Grazioso Salvare Logo:

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

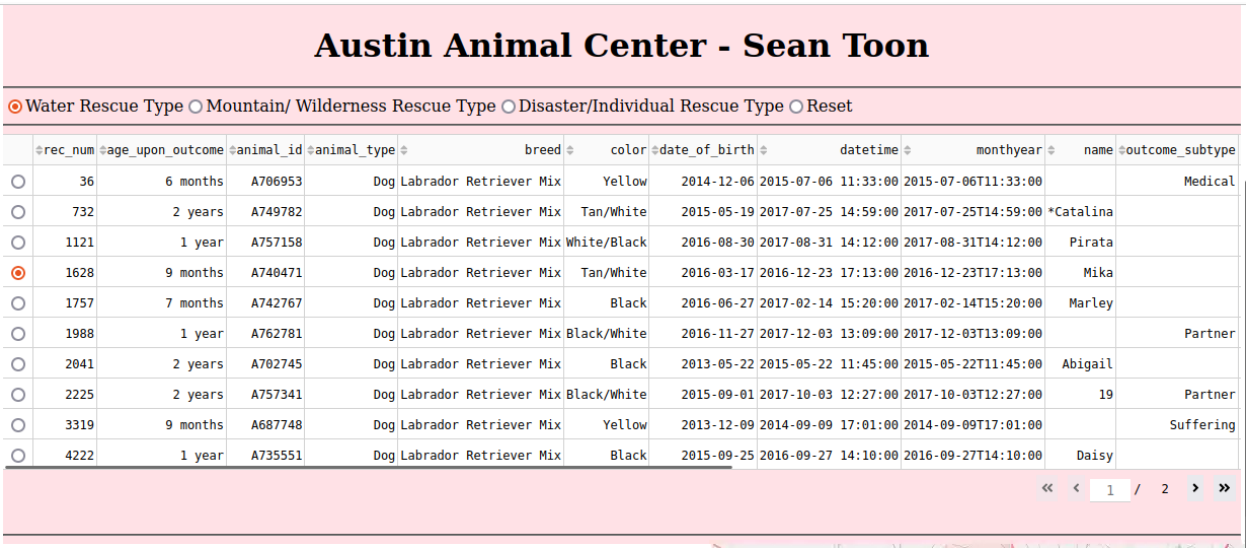


Dashboard data and charts after clicking the “Water Rescue Type” filtering radio button:

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

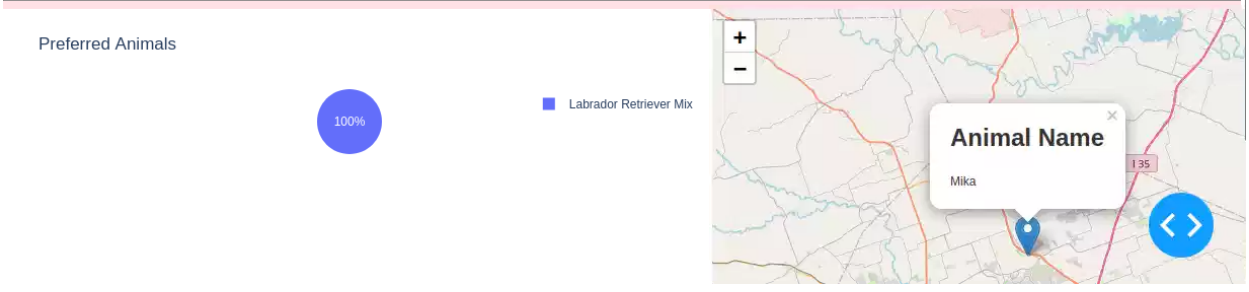
	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype
○	36	6 months	A706953	Dog	Labrador Retriever Mix	Yellow	2014-12-06	2015-07-06 11:33:00	2015-07-06T11:33:00		Medical
○	732	2 years	A749782	Dog	Labrador Retriever Mix	Tan/White	2015-05-19	2017-07-25 14:59:00	2017-07-25T14:59:00	*Catalina	
○	1121	1 year	A757158	Dog	Labrador Retriever Mix	White/Black	2016-08-30	2017-08-31 14:12:00	2017-08-31T14:12:00	Pirata	
○	1628	9 months	A740471	Dog	Labrador Retriever Mix	Tan/White	2016-03-17	2016-12-23 17:13:00	2016-12-23T17:13:00	Mika	
○	1757	7 months	A742767	Dog	Labrador Retriever Mix	Black	2016-06-27	2017-02-14 15:20:00	2017-02-14T15:20:00	Marley	
◉	1988	1 year	A762781	Dog	Labrador Retriever Mix	Black/White	2016-11-27	2017-12-03 13:09:00	2017-12-03T13:09:00		Partner
○	2041	2 years	A702745	Dog	Labrador Retriever Mix	Black	2013-05-22	2015-05-22 11:45:00	2015-05-22T11:45:00	Abigail	
○	2225	2 years	A757341	Dog	Labrador Retriever Mix	Black/White	2015-09-01	2017-10-03 12:27:00	2017-10-03T12:27:00	19	Partner
○	3319	9 months	A687748	Dog	Labrador Retriever Mix	Yellow	2013-12-09	2014-09-09 17:01:00	2014-09-09T17:01:00		Suffering
○	4222	1 year	A735551	Dog	Labrador Retriever Mix	Black	2015-09-25	2016-09-27 14:10:00	2016-09-27T14:10:00	Daisy	

« ‹ 1 / »



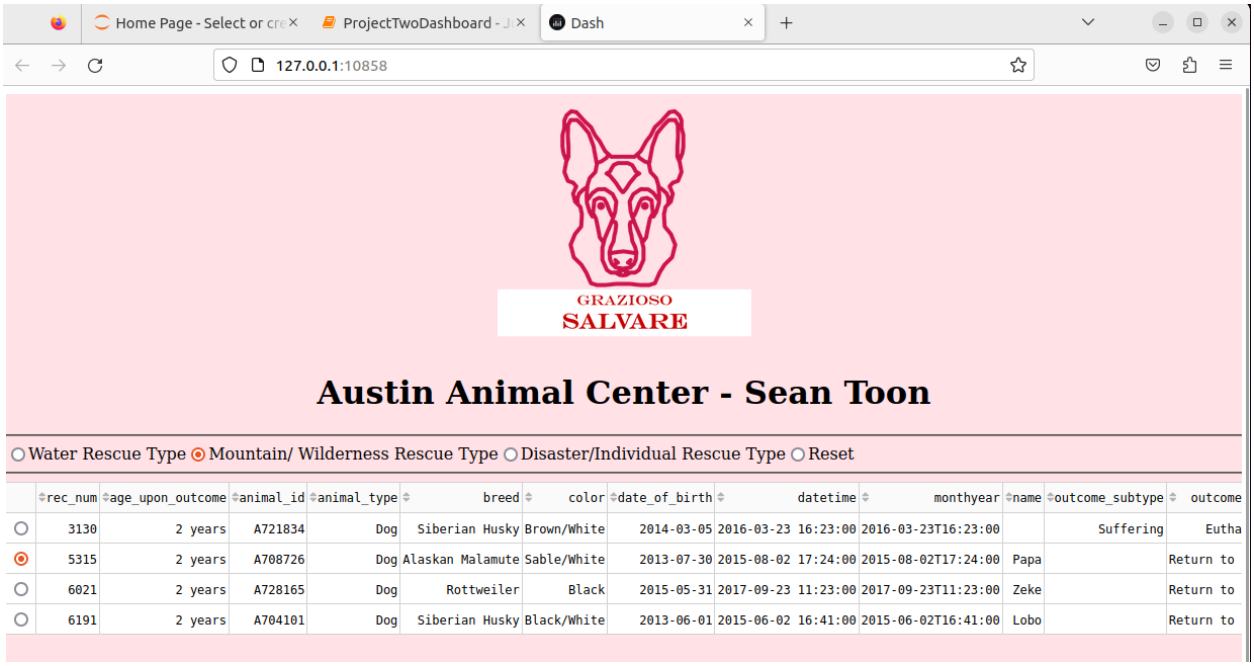
	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_subtype
○	36	6 months	A706953	Dog	Labrador Retriever Mix	Yellow	2014-12-06	2015-07-06 11:33:00	2015-07-06T11:33:00		Medical
○	732	2 years	A749782	Dog	Labrador Retriever Mix	Tan/White	2015-05-19	2017-07-25 14:59:00	2017-07-25T14:59:00	*Catalina	
○	1121	1 year	A757158	Dog	Labrador Retriever Mix	White/Black	2016-08-30	2017-08-31 14:12:00	2017-08-31T14:12:00	Pirata	
◉	1628	9 months	A740471	Dog	Labrador Retriever Mix	Tan/White	2016-03-17	2016-12-23 17:13:00	2016-12-23T17:13:00	Mika	
○	1757	7 months	A742767	Dog	Labrador Retriever Mix	Black	2016-06-27	2017-02-14 15:20:00	2017-02-14T15:20:00	Marley	
○	1988	1 year	A762781	Dog	Labrador Retriever Mix	Black/White	2016-11-27	2017-12-03 13:09:00	2017-12-03T13:09:00		Partner
○	2041	2 years	A702745	Dog	Labrador Retriever Mix	Black	2013-05-22	2015-05-22 11:45:00	2015-05-22T11:45:00	Abigail	
○	2225	2 years	A757341	Dog	Labrador Retriever Mix	Black/White	2015-09-01	2017-10-03 12:27:00	2017-10-03T12:27:00	19	Partner
○	3319	9 months	A687748	Dog	Labrador Retriever Mix	Yellow	2013-12-09	2014-09-09 17:01:00	2014-09-09T17:01:00		Suffering
○	4222	1 year	A735551	Dog	Labrador Retriever Mix	Black	2015-09-25	2016-09-27 14:10:00	2016-09-27T14:10:00	Daisy	

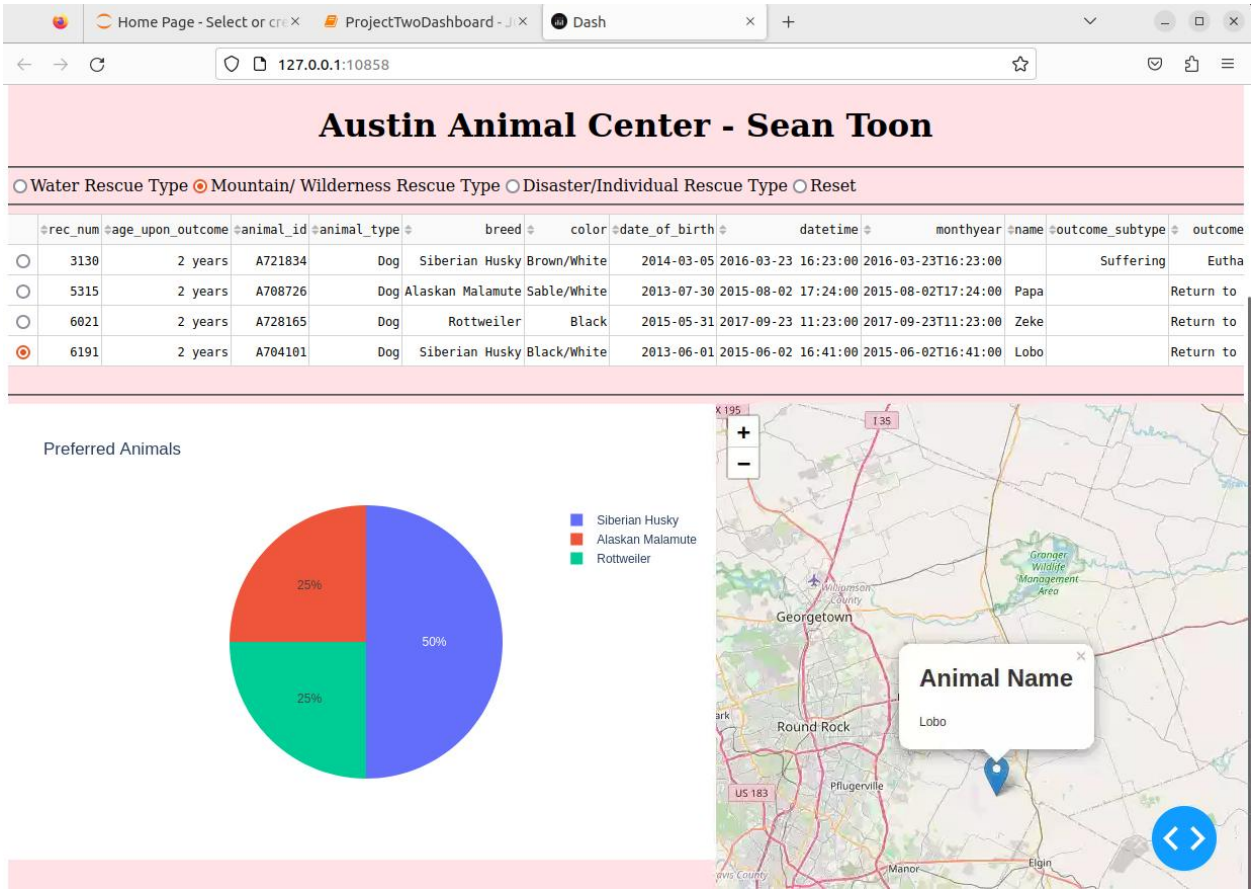
« ‹ 1 / 2 › »



Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

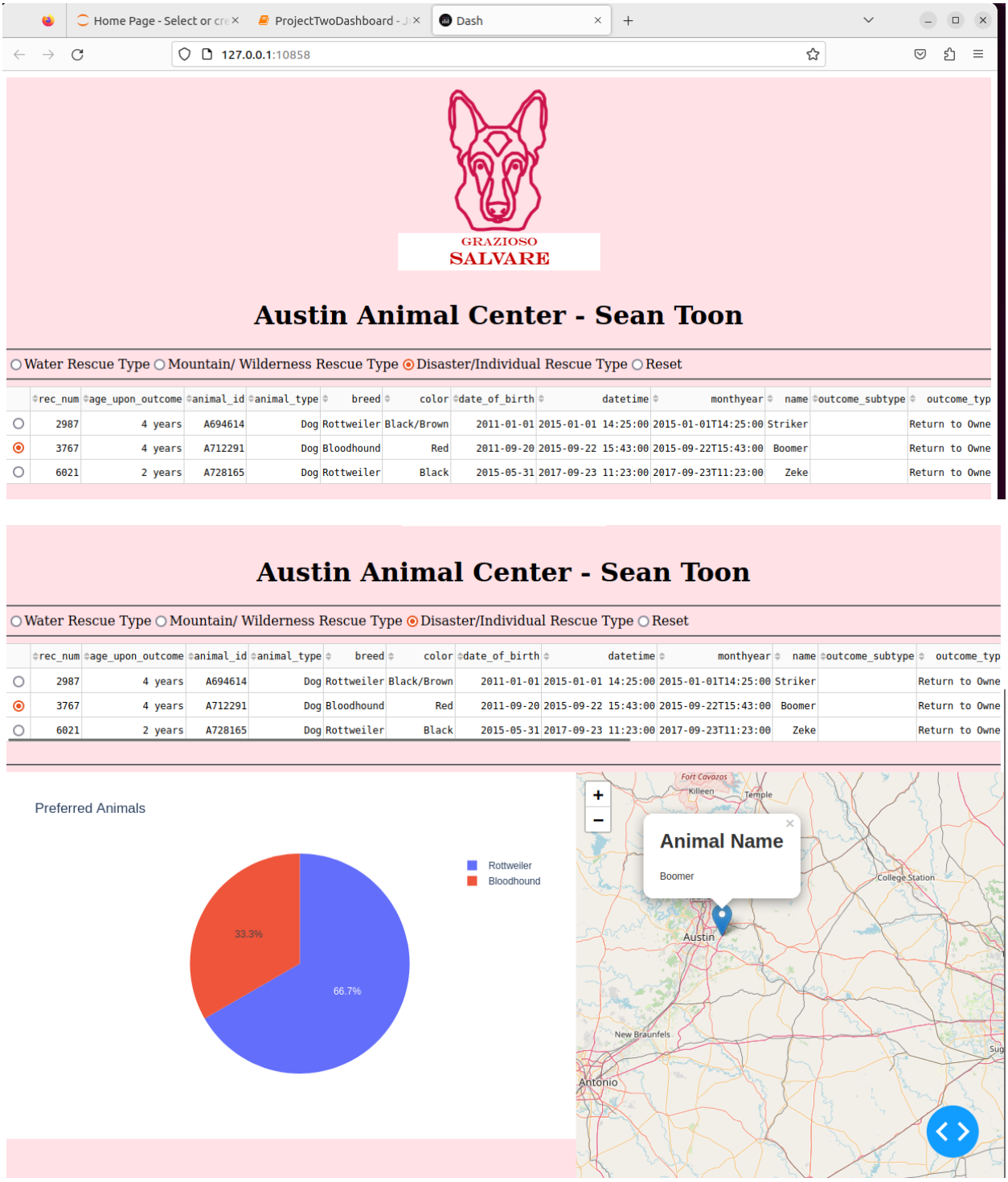
Dashboard after clicking the " Mountain/ Wilderness Rescue Type" radio button:





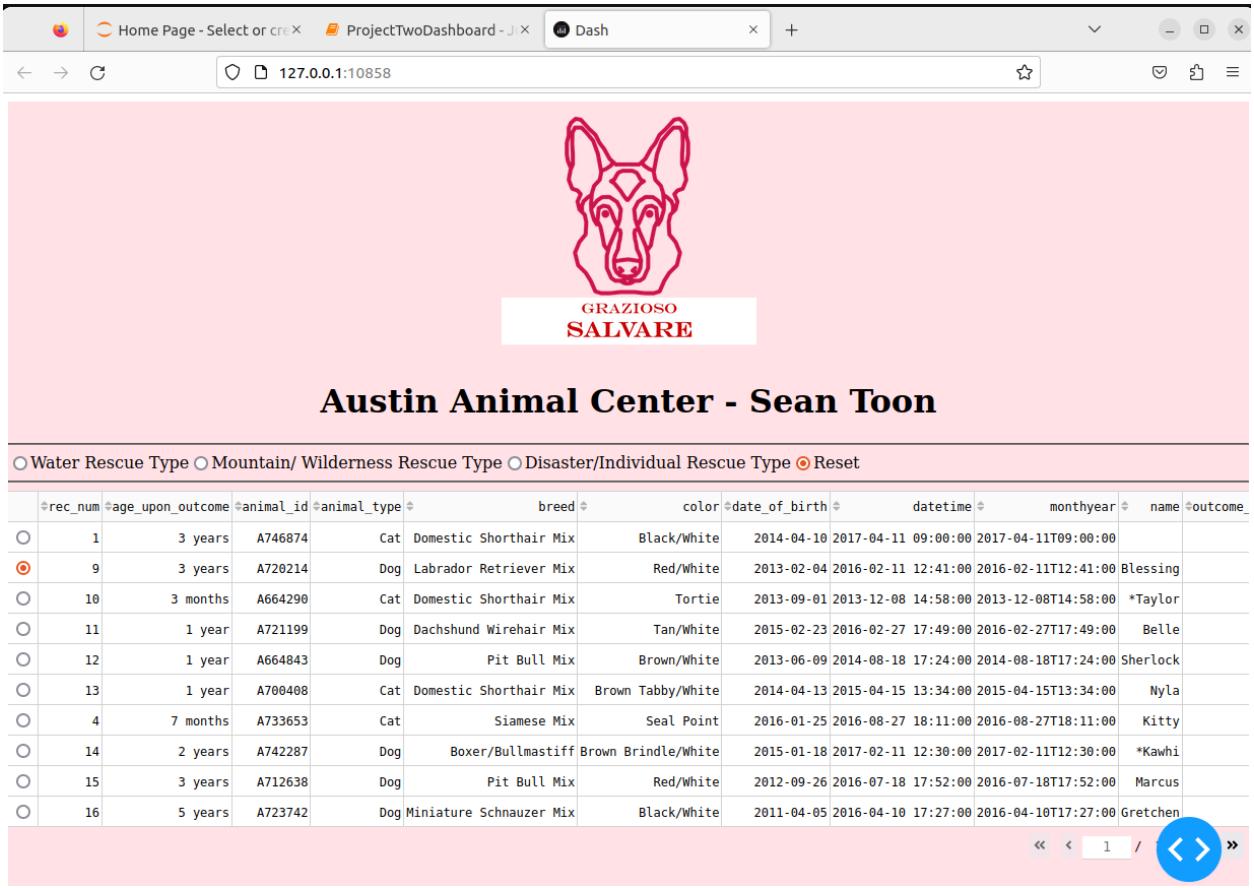
Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.

Dashboard after clicking the “Disaster/Individual Rescue” radio button:



Dashboard after clicking the “Reset” radio button:

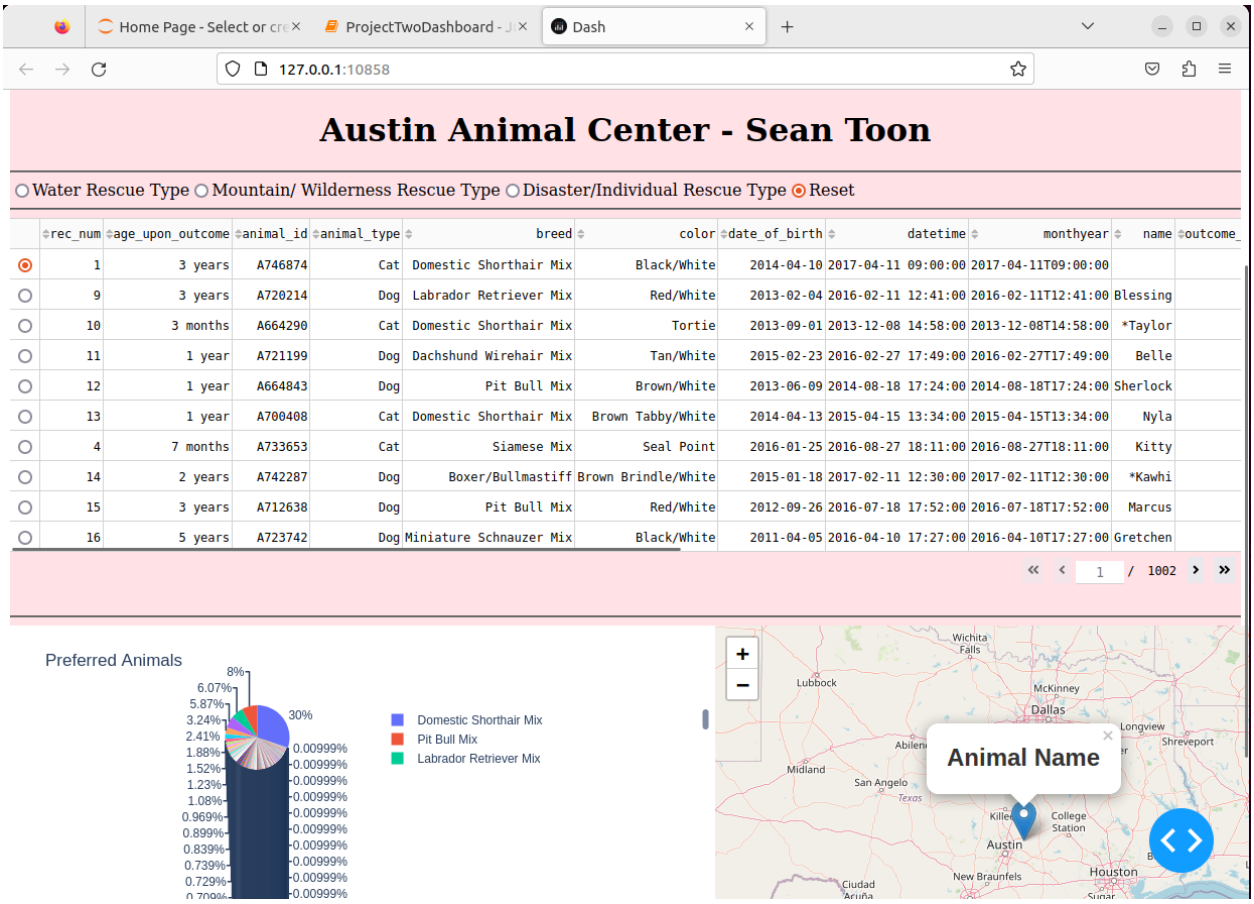
Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.



	rec_num	age_upon_outcome	animal_id	animal_type	breed	color	date_of_birth	datetime	monthyear	name	outcome_
○	1	3 years	A746874	Cat	Domestic Shorthair Mix	Black/White	2014-04-10	2017-04-11 09:00:00	2017-04-11T09:00:00		
◉	9	3 years	A720214	Dog	Labrador Retriever Mix	Red/White	2013-02-04	2016-02-11 12:41:00	2016-02-11T12:41:00	Blessing	
○	10	3 months	A664290	Cat	Domestic Shorthair Mix	Tortie	2013-09-01	2013-12-08 14:58:00	2013-12-08T14:58:00	*Taylor	
○	11	1 year	A721199	Dog	Dachshund Wirehair Mix	Tan/White	2015-02-23	2016-02-27 17:49:00	2016-02-27T17:49:00	Belle	
○	12	1 year	A664843	Dog	Pit Bull Mix	Brown/White	2013-06-09	2014-08-18 17:24:00	2014-08-18T17:24:00	Sherlock	
○	13	1 year	A700408	Cat	Domestic Shorthair Mix	Brown Tabby/White	2014-04-13	2015-04-15 13:34:00	2015-04-15T13:34:00	Nyla	
○	4	7 months	A733653	Cat	Siamese Mix	Seal Point	2016-01-25	2016-08-27 18:11:00	2016-08-27T18:11:00	Kitty	
○	14	2 years	A742287	Dog	Boxer/Bullmastiff	Brown Brindle/White	2015-01-18	2017-02-11 12:30:00	2017-02-11T12:30:00	*Kawhi	
○	15	3 years	A712638	Dog	Pit Bull Mix	Red/White	2012-09-26	2016-07-18 17:52:00	2016-07-18T17:52:00	Marcus	
○	16	5 years	A723742	Dog	Miniature Schnauzer Mix	Black/White	2011-04-05	2016-04-10 17:27:00	2016-04-10T17:27:00	Gretchen	

« ‹ 1 / ›

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.



Contact
Your name: Sean Toon

Note: This template has been adapted from the following sample templates: Make a README, Best README Template, and A Beginners Guide to Writing a Kickass README.