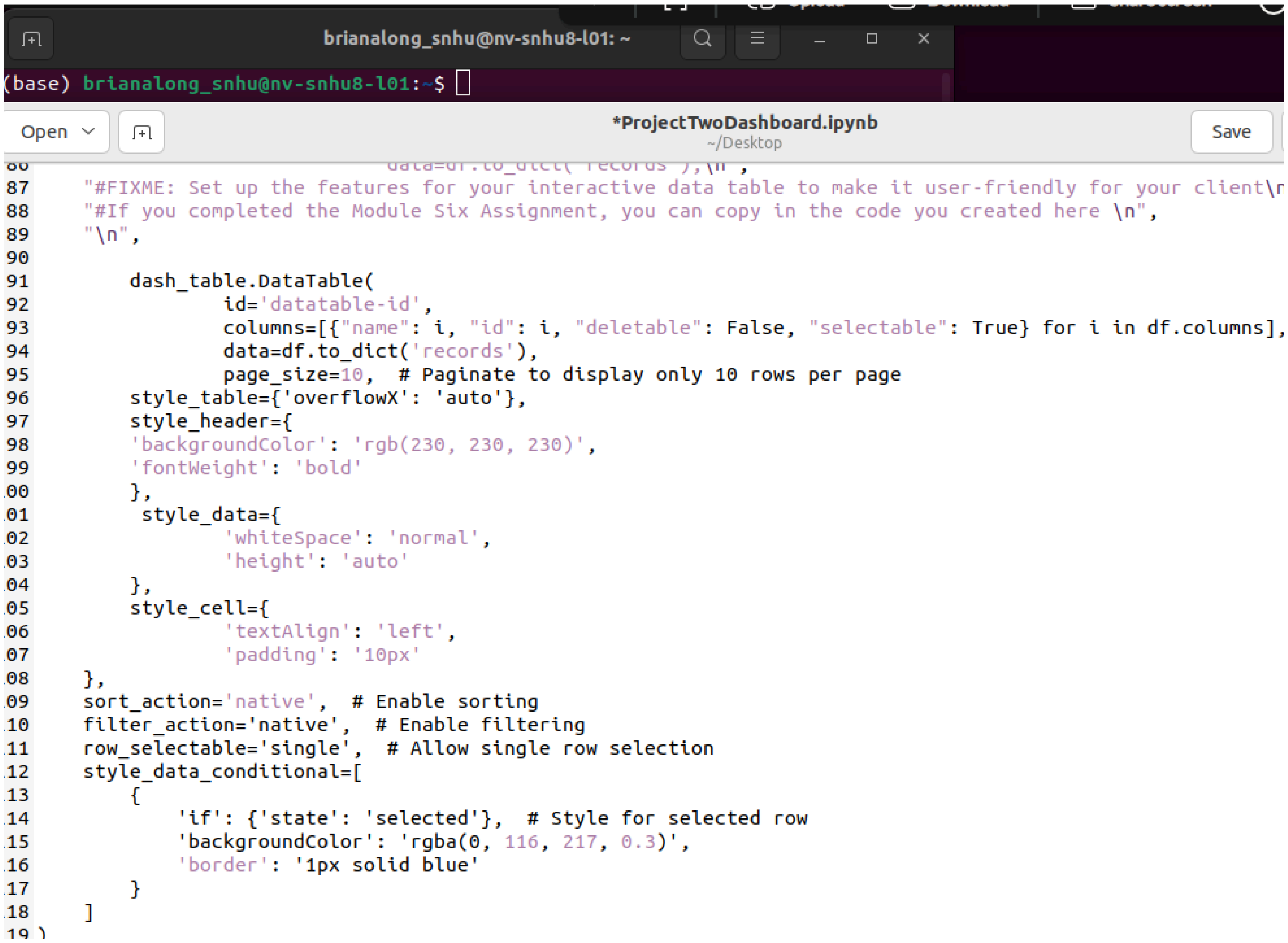
Briana Long

CS 340

Project Two

Describe the required functionality of the project.

The dynamic data table displays all the filtered data. The actions created for the table are sorting, filtering, and row selection. This allows users to see the data highlighted.

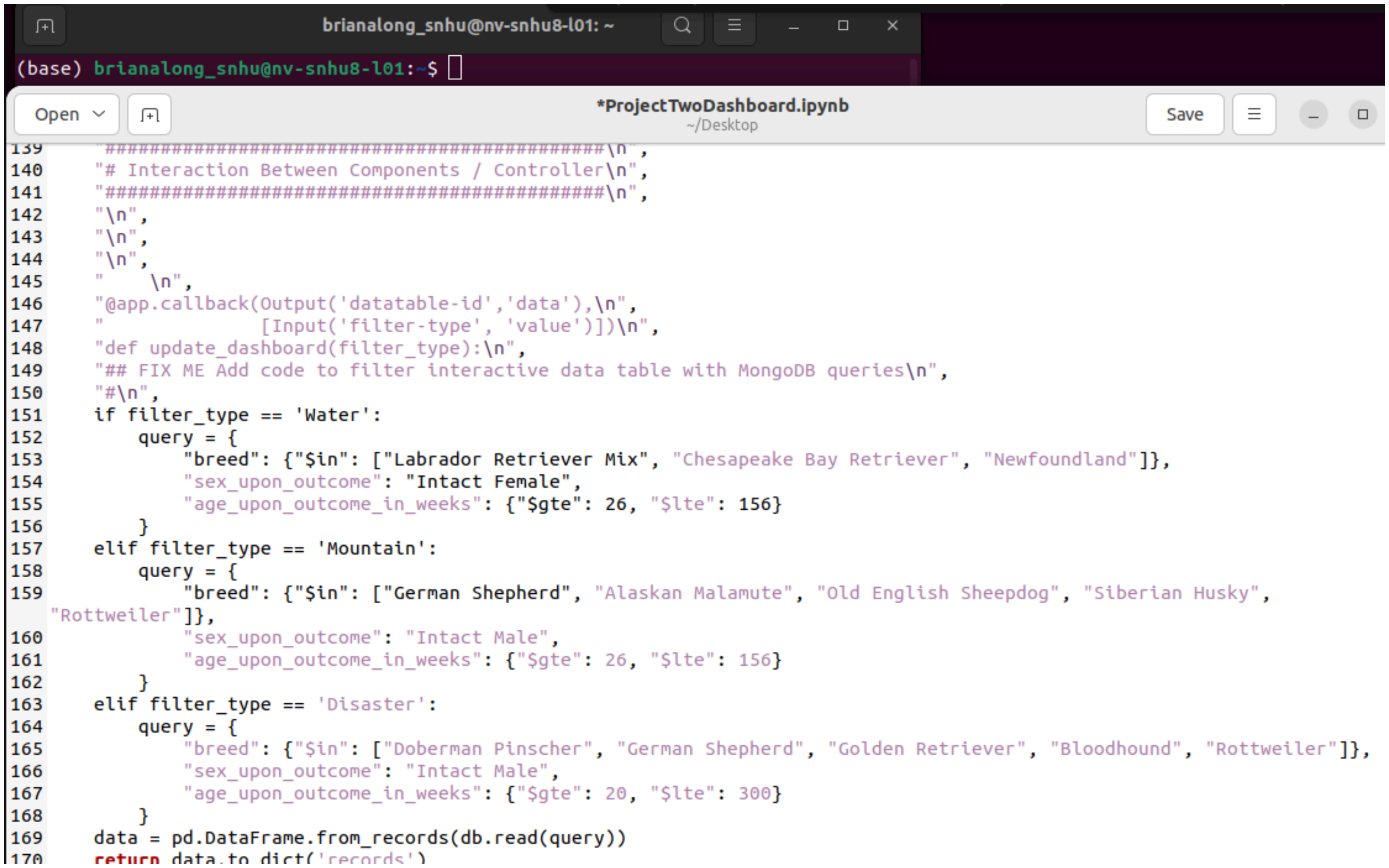


```
(base) brianalong_snhu@nv-snhu8-l01:~$
```

```
    "#FIXME: Set up the features for your interactive data table to make it user-friendly for your client\r
    "#If you completed the Module Six Assignment, you can copy in the code you created here \n",
    "\n",

        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'),
                page_size=10,  # Paginate to display only 10 rows per page
        style_table={'overflowX': 'auto'},
        style_header={
        'backgroundColor': 'rgb(230, 230, 230)',
        'fontWeight': 'bold'
        },
         style_data={
                'whiteSpace': 'normal',
                'height': 'auto'
        },
        style_cell={
                'textAlign': 'left',
                'padding': '10px'
    },
    sort_action='native',  # Enable sorting
    filter_action='native',  # Enable filtering
    row_selectable='single',  # Allow single row selection
    style_data_conditional=[
        {
            'if': {'state': 'selected'},  # Style for selected row
            'backgroundColor': 'rgba(0, 116, 217, 0.3)',
            'border': '1px solid blue'
        }
    ]
)
```

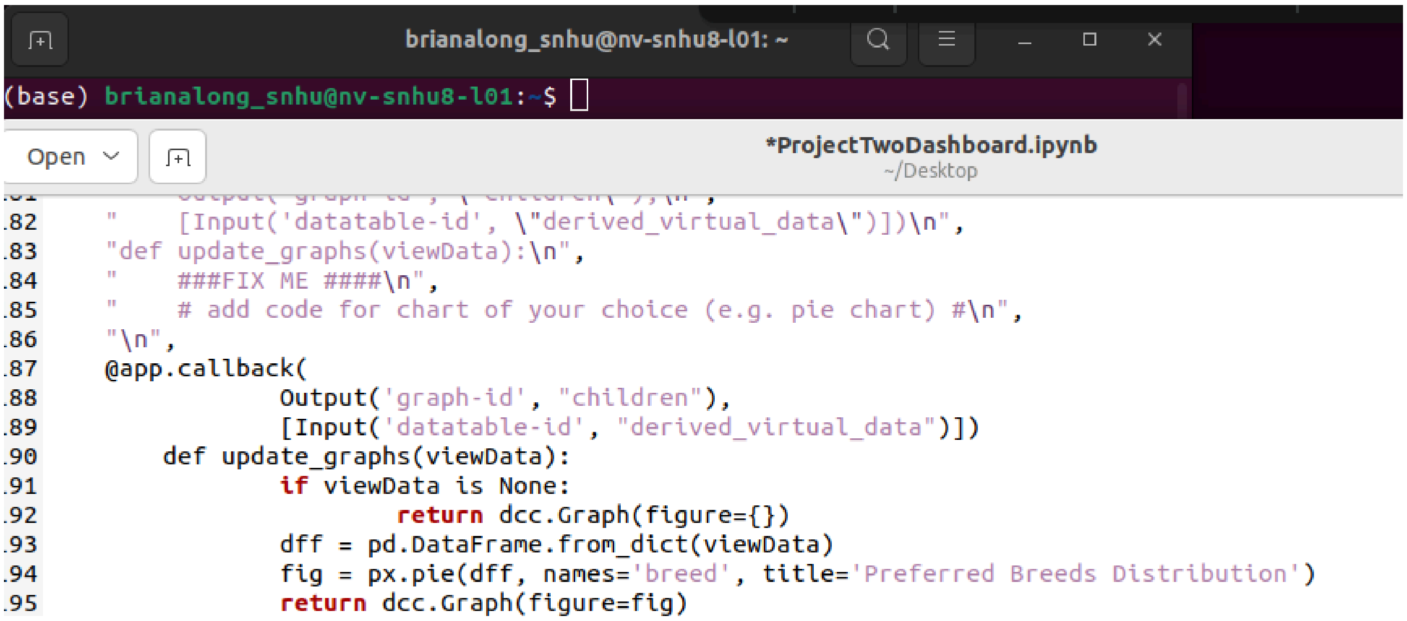
Interactive filtering is used to filter the database based on the rescue types of the animals in the shelter. This includes water rescues, mountain/wilderness, and disaster tracking.



```
(base) brianalong_snhu@nv-snhu8-l01:~$
```

```
    "#############################################\n",
    "# Interaction Between Components / Controller\n",
    "#############################################\n",
    "\n",
    "\n",
    "\n",
    "    \n",
    "@app.callback(Output('datatable-id','data'),\n",
    "               [Input('filter-type', 'value')])\n",
    "def update_dashboard(filter_type):\n",
    "## FIX ME Add code to filter interactive data table with MongoDB queries\n",
    "#\n",
    if filter_type == 'Water':
        query = {
            "breed": {"$in": ["Labrador Retriever Mix", "Chesapeake Bay Retriever", "Newfoundland"]},
            "sex_upon_outcome": "Intact Female",
            "age_upon_outcome_in_weeks": {"$gte": 26, "$lte": 156}
        }
    elif filter_type == 'Mountain':
        query = {
            "breed": {"$in": ["German Shepherd", "Alaskan Malamute", "Old English Sheepdog", "Siberian Husky",
"Rottweiler"]},
            "sex_upon_outcome": "Intact Male",
            "age_upon_outcome_in_weeks": {"$gte": 26, "$lte": 156}
        }
    elif filter_type == 'Disaster':
        query = {
            "breed": {"$in": ["Doberman Pinscher", "German Shepherd", "Golden Retriever", "Bloodhound", "Rottweiler"]},
            "sex_upon_outcome": "Intact Male",
            "age_upon_outcome_in_weeks": {"$gte": 20, "$lte": 300}
        }
    data = pd.DataFrame.from_records(db.read(query))
    return data.to_dict('records')
```

The geolocation map displays the location data of the animals selected in the database map. This can be updated with the selected data and filters. The chart style I have chosen is a pie chart. It can show the favored breeds of dogs based on the dataset.



brianalong_snhu@nv-snhu8-l01: ~

```
(base) brianalong_snhu@nv-snhu8-l01:~$
```

Open

*ProjectTwoDashboard.ipynb
~/Desktop

```
    "    [Input('datatable-id', \"derived_virtual_data\")])\n",
    "def update_graphs(viewData):\n",
    "    ###FIX ME ####\n",
    "    # add code for chart of your choice (e.g. pie chart) #\n",
    "\n",
    @app.callback(
                Output('graph-id', "children"),
                [Input('datatable-id', "derived_virtual_data")])
        def update_graphs(viewData):
                if viewData is None:
                        return dcc.Graph(figure={})
                dff = pd.DataFrame.from_dict(viewData)
                fig = px.pie(dff, names='breed', title='Preferred Breeds Distribution')
                return dcc.Graph(figure=fig)
```

Describe the tools used to achieve this functionality and a rationale for why these tools were used.

Tools used:

MongoDB

MongoDB is used for databases that are NoSQL. This makes the structure easily changeable for dynamic models and chart changes. MongoDB uses the Python library pymongo for easier programming. This allows data to be found more easily with simple inquiries.

Dash Framework

Dash is used for creating interactive web applications in Python. This allows for backend functions such as searching to be combined with front-end data visualizations such as dynamic charts.

Plotly Express

Plotly is used with Dash for the ability to dynamically update charts. It is used for pie charts and bar graphs. I had chosen to use a pie chart for my data.

Dash Leaflet

Dash Leaflet is used for interactivity of the geolocation data map.

Pandas

Pandas is used for the conversion of the data between MongoDB and the Dash components

Links:

MongoDB: https://www.mongodb.com

Dash Documentation: https://dash.plotly.com

Plotly Express: https://plotly.com/python/plotly-express/

Dash Leaflet: https://dash-leaflet.herokuapp.com/

Python: https://www.python.org/

Explain the steps that were taken to complete the project.

The first step is to load the data. This is done by connecting MongoDB to the database. The files are then loaded and able to be retrieved. The dashboard layout is then designed. This started with adding the logo and the identifier of my name as the title. The interactive filters for the data were then added. This ensures that rescues by type can be searched. Next, the dynamic table is created. The Dash Datatable is created with filters based on the breeds. The visualization chart is made with a pie chart. This is an easy way to represent the data of the breed distributions of the animals in the shelter network. Each feature is tested individually to make sure each functions correctly.

Identify any challenges that were encountered and explain how those challenges were overcome.

The challenge I found was proper indentation. I was constantly fixing my indentation when writing the application. When my indentation was incorrect an error was displayed. I found creating the pie chart an interesting process as I have never done that before. I had to do extra research to make sure I understood the way to code the implementation. My graph chart was not functioning correctly so I made a pie chart instead.