Assignment Overview

- Story
- Components of the report items
- Expected layout
- Requirements to create the dashboard
- What is new in this exercise compared to other labs?
- Review
- Hints to complete TODOs

Estimated time needed: 45 minutes

About Skills Network Cloud IDE

This Skills Network Labs Cloud IDE (Integrated Development Environment) provides a hands-on environment in your web browser for completing course and project related labs. It utilizes Theia, an open-source IDE platform, that can be run on desktop or on the cloud. So far in the course you have been using Jupyter notebooks to run your python code. This IDE provides an alternative for editing and running your Python code. In this lab you will be using this alternative Python runtime to create and launch your Dash applications.

Important Notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. When you launch the Cloud IDE, you are presented with a 'dedicated computer on the cloud' exclusively for you. This is available to you as long as you are actively working on the labs.

Once you close your session or it is timed out due to inactivity, you are logged off, and this 'dedicated computer on the cloud' is deleted along with any files you may have created, dowloaded or installed. The next time you launch this lab, a new environment is created for you.

If you finish only part of the lab and return later, you may have to start from the beginning. So, it is a good idea to plan to your time accordingly and finish your labs in a single session.

Story:

As a data analyst, you have been given a task to monitor and report US domestic airline flights performance. Goal is to analyze the performance of the reporting airline to improve flight reliability thereby improving customer reliability.

Below are the key report items,

- Yearly airline performance report
- Yearly average flight delay statistics

NOTE: Year range is between 2005 and 2020.

Components of the report items

1. Yearly airline performance report

   For the chosen year provide,

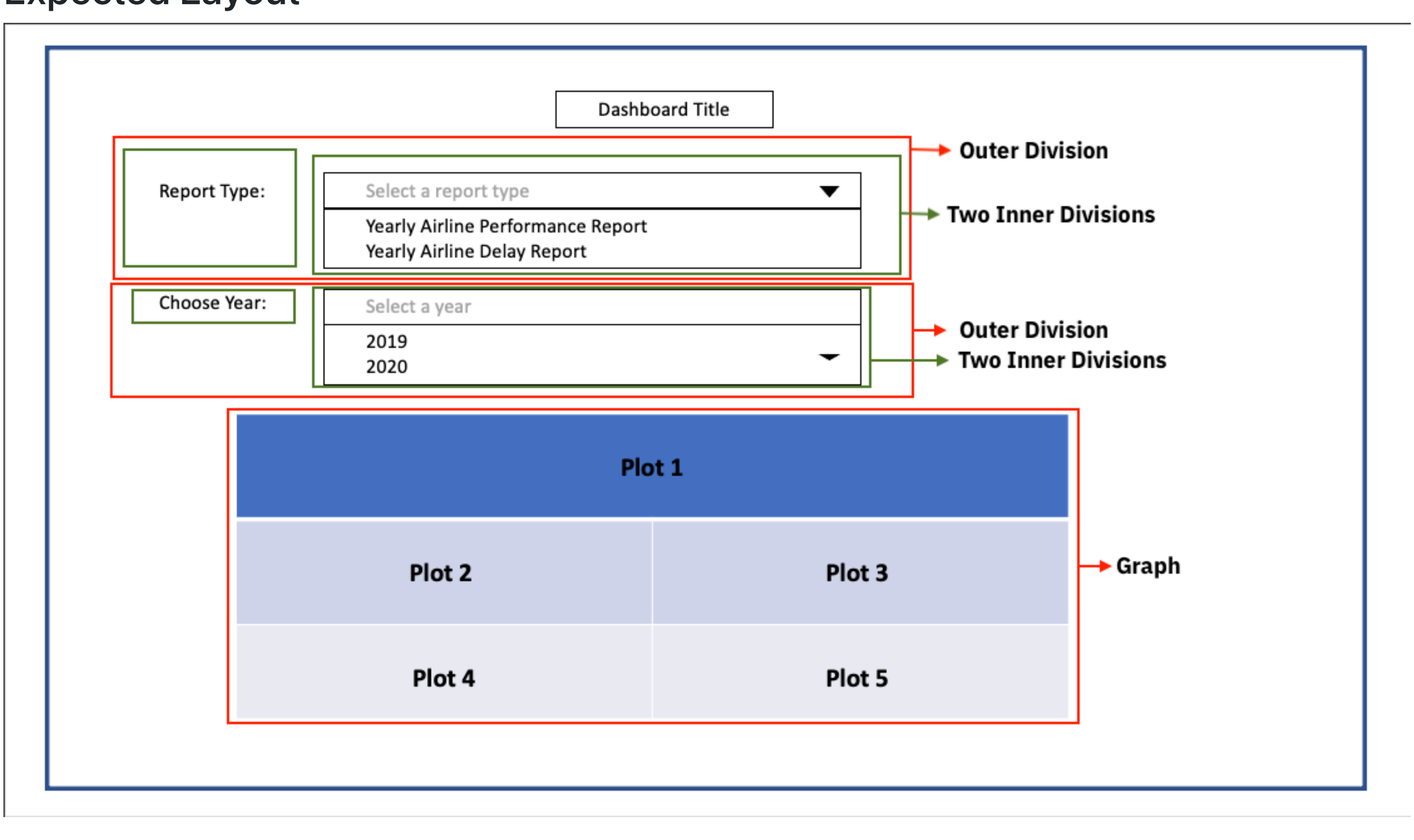
   - Number of flights under different cancellation categories using bar chart.
   - Average flight time by reporting airline using line chart.
   - Percentage of diverted airport landings per reporting airline using pie chart.
   - Number of flights flying from each state using choropleth map.
   - Number of flights flying to each state from each reporting airline using treemap chart.
2. Yearly average flight delay statistics

   For the chosen year provide,

   - Monthly average carrier delay by reporting airline for the given year.
   - Monthly average weather delay by reporting airline for the given year.
   - Monthly average national air system delay by reporting airline for the given year.
   - Monthly average security delay by reporting airline for the given year.
   - Monthly average late aircraft delay by reporting airline for the given year.

   NOTE: You have worked created the same dashboard components in `Flight Delay Time Statistics Dashboard` section. We will be reusing the same.

Expected Layout



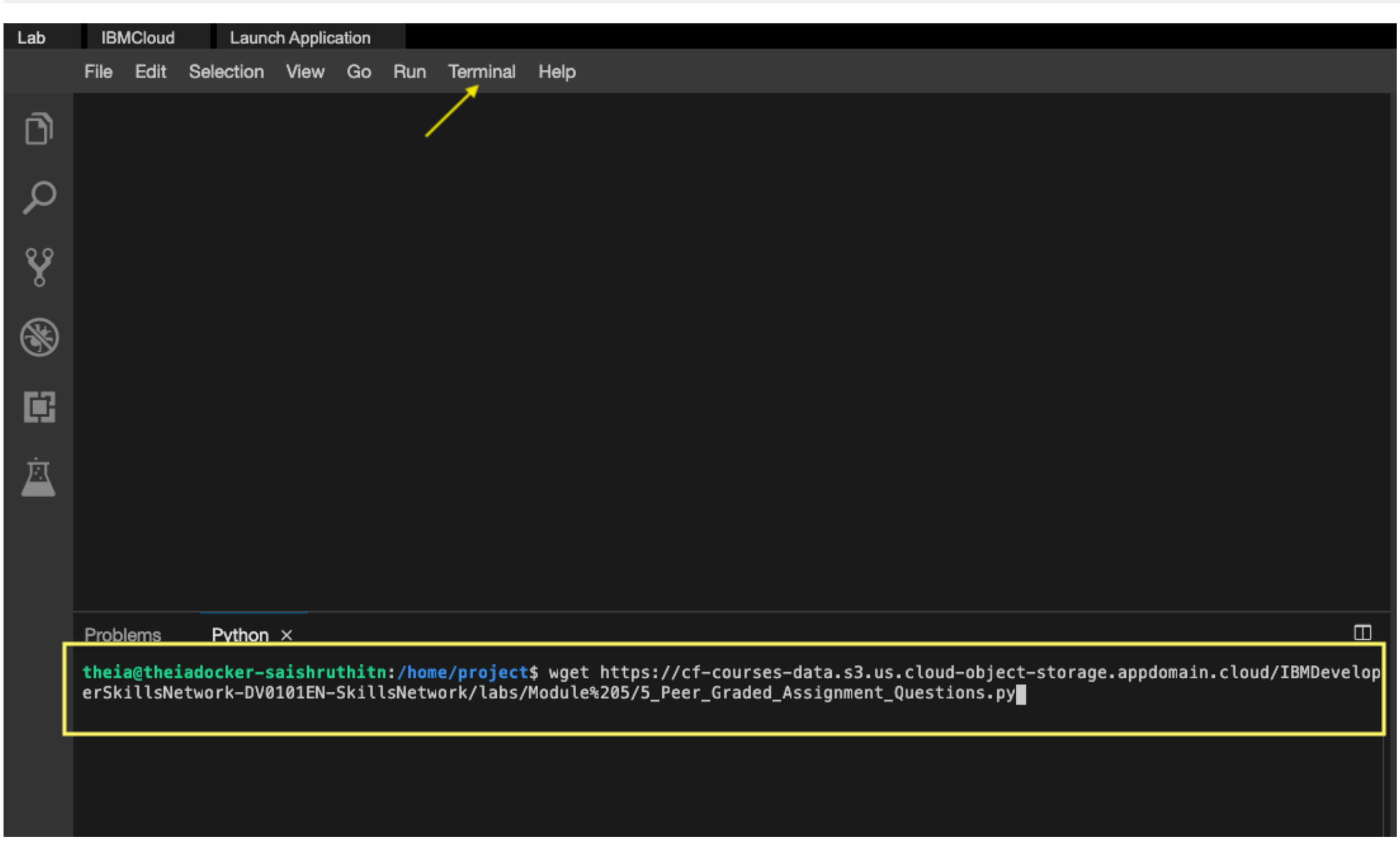
Requirements to create the expected result

- Two dropdown menus: For choosing report type and year
- Each dropdown will be designed as follows:
  - An outer division with two inner divisions (as shown in the expected layout)
  - One of the inner divisions will have information about the dropdown and the other one is dropdown.
- Layout for adding graphs.
- Callback function to compute data, create graph and return to the layout.

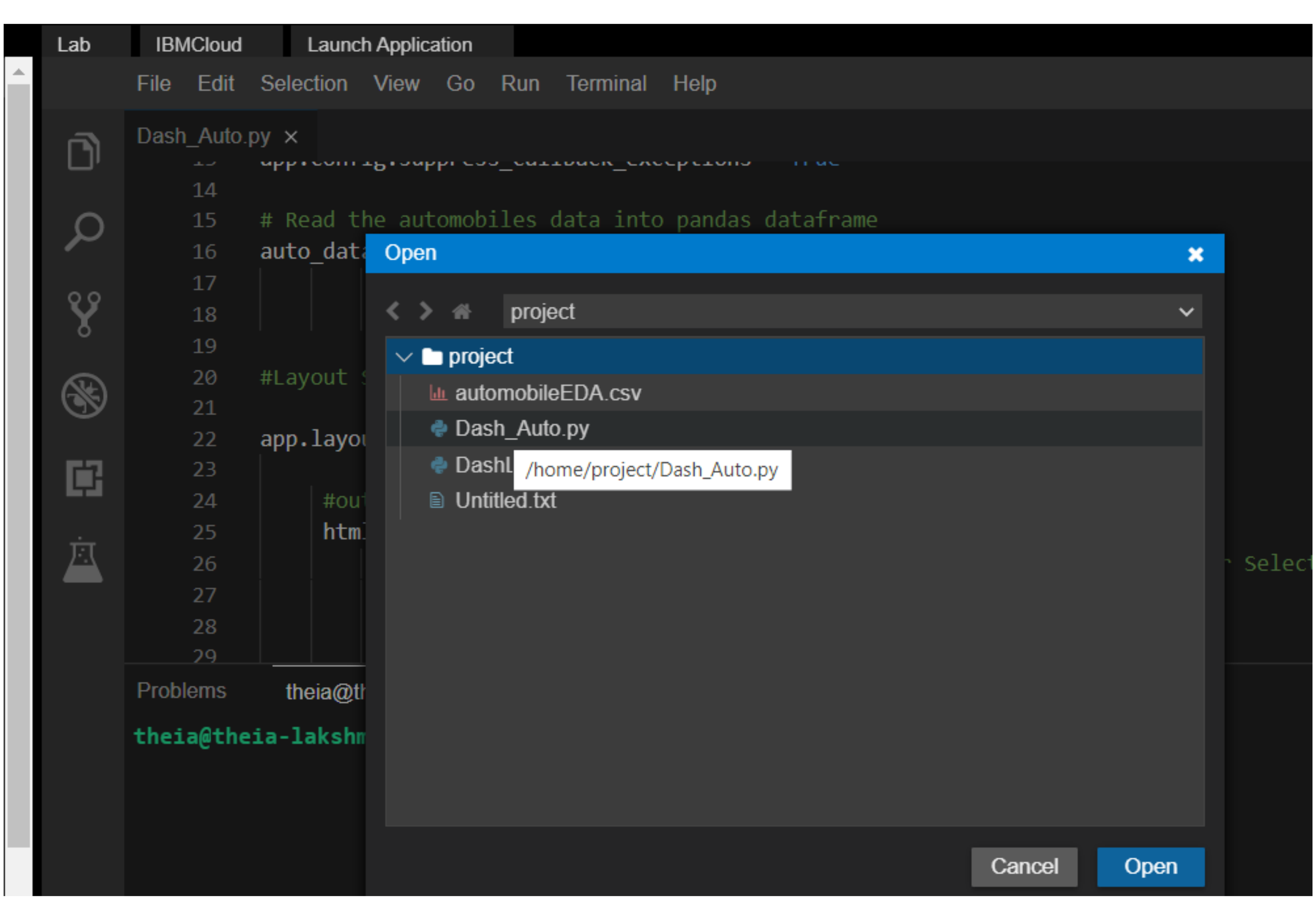
Get the application skeleton

- Copy and paste the below command in the terminal to download the skeleton.

```
wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-SkillsNetwork/labs/Module%205/5_Pe
```



- Open `5_Peer_Graded_Assignment_Questions.py`



You can use this as a base code to complete the task below.

What's new in this exercise compared to other labs?

- Make sure the layout is clean without any default graphs or graph layouts. We will do this by 3 changes:
  1. Having empty html.Div and use the callback to Output the dcc.graph as the Children of that Div.
  2. Add a state variable in addition to callback decorator input and output parameter. This will allow us to pass extra values without firing the callbacks. Here, we need to pass two inputs `chart type` and `year`. Input is read only after user entering all the information.
- Use new html display style `flex` to arrange the dropdown menu with description.
- Update app run step to avoid getting error message before initiating callback.

NOTE: These steps are only for review.

Let's create the application

Review

Search/Look for `Review` word in the script to learn how commands are used and computations are carried out. There are 7 review items.

- REVIEW1: Clear the layout and do not display exception till callback gets executed.
- REVIEW2: Dropdown creation.
- REVIEW3: Observe how we add an empty division and providing an id that will be updated during callback.
- REVIEW4: Holding output state till user enters all the form information. In this case, it will be chart type and year.
- REVIEW5: Number of flights flying from each state using choropleth
- REVIEW6: Return dcc.Graph component to the empty division
- REVIEW7: This covers chart type 2 and we have completed this exercise under Flight Delay Time Statistics Dashboard section

Hints to complete TASKS

Search/Look for `TASK` word in the script to identify places where you need to complete the code.

TASK1: Add title to the dashboard

- Provide title of the dash application title as `US Domestic Airline Flights Performance`.
- Make the heading center aligned, set color as `#503D36`, and font size as `24`. Sample: style={'textAlign': 'left', 'color': '#000000', 'font-size': 0}

Reference link

TASK2: Add a dropdown menu

Create a dropdown menu and add two chart options to it. Below is the skeleton:

```
dcc.Dropdown(id='....',
                options=[
                        {'label': '....', 'value': '...'},
                        {'label': '....', 'value': '...'}
                        ],
                placeholder='....',
                style={....})
```

Parameters to be updated in `dcc.Dropdown`:

- Set `id` as `input-type`.
- Set `options` to list containing dictionaries with key as `label` and user provided value for labels in `value`.

  1st dictionary

  - label: Yearly Airline Performance Report
  - value: OPT1

  2nd dictionary

  - label: Yearly Airline Delay Report
  - value: OPT2
- Set placeholder to `Select a report type`.
- Set width as `80%`, padding as `3px`, font size as `20px`, text-align-last as `center` inside style parameter dictionary.

Reference link

TASK3: Add a division with two empty divisions inside.

Add a division with two empty divisions inside. For reference, observe how code under `REVIEW3` has been structured.

Provide division ids as `plot4` and `plot5`. Display style as `flex`.

TASK4: Add 5 ouput components

Our layout has 5 outputs so we need to create 5 output components. Review how input components are constructed to fill in for output component.

It is a list with 5 output parameters with component id and property. Here, the component property will be `children` as we have created empty division and passing in `dcc.Graph` (figure) after computation.

Component ids will be `plot1` , `plot2`, `plot2`, `plot4`, and `plot5`. Skeleton is provided below:

```
[Output(component_id='plot1', component_property='children'),
 Output(....),
 Output(....),
 Output(....),
 Output(....)]
```

TASK5: Average flight time by reporting airline

Create a line plot using returned dataframe `line_data` from the above function `compute_data_choice` using `plotly.express`. Link for reference is here

Set:

- Figure name as `line_fig`
- Input data as `line_data`
- x as `Month`, y as `AirTime`, color as `Reporting_Airline` and `title` as `Average monthly flight time (minutes) by airline`.

Below is the skeleton:

```
figure_name = px.line(input_data, x='....', y='...', color='...', title='....')
```

TASK6: Number of flights flying to each state from each reporting airline

Create a treemap plot using returned dataframe `tree_data` from the above function `compute_data_choice` using `plotly.express`. Link for reference is here

Set

- Figure name as `tree_fig`
- Data as `tree_data`
- Path as `['DestState', 'Reporting_Airline']`
- Values as `Flights`
- Colors as `Flights` and color_continuous_scale as `'RdBu'`
- Title as `'Flight count by airline to destination state'`

Below is the skeleton:

```
tree_fig = px.treemap(data, path=['...', '...'],
                      values='...',
                      color='...',
                      color_continuous_scale='...',
                      title='...'
            )
```

Run the Application

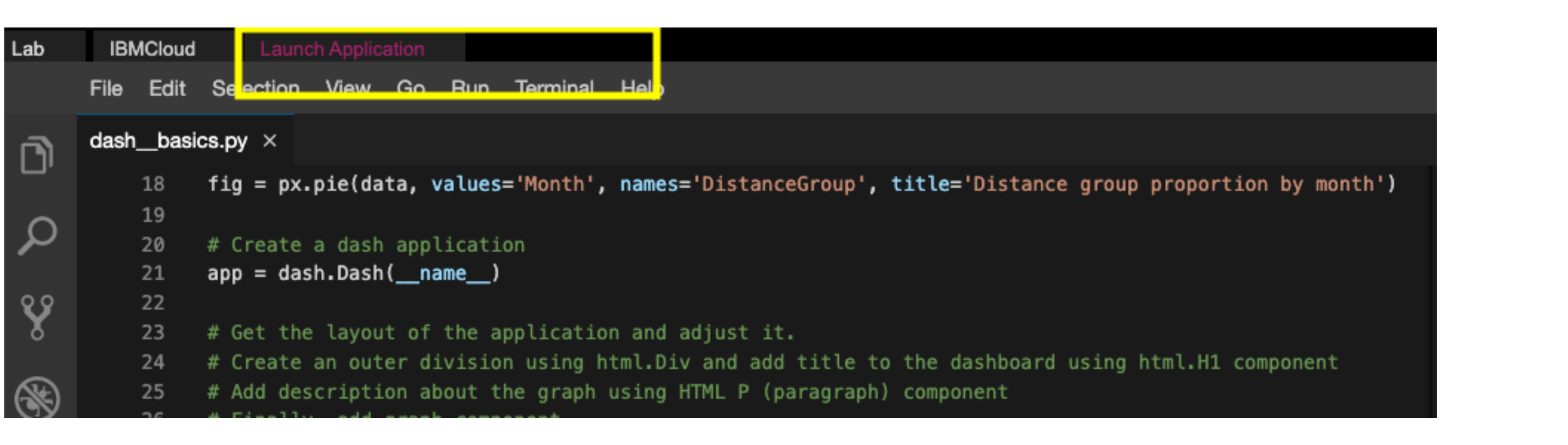
- Firstly, install pandas and dash using the following command

```
pip3 install pandas dash
```

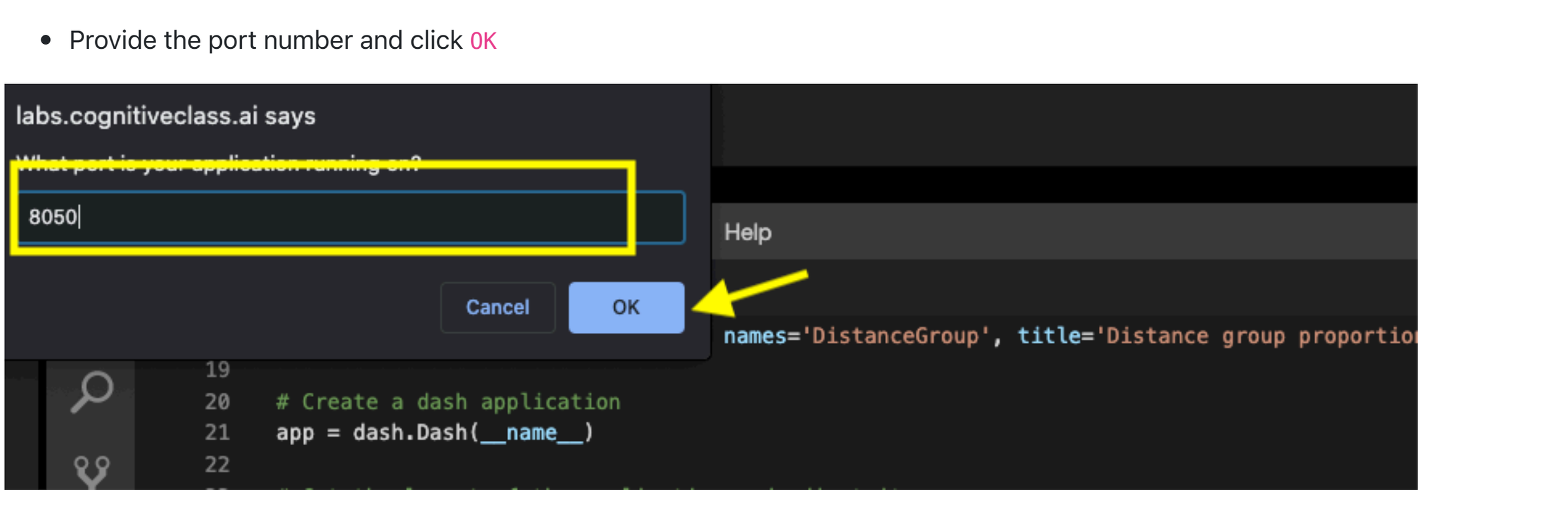
- Run the python file using the command

```
python3 5_Peer_Graded_Assignment_Questions.py
```

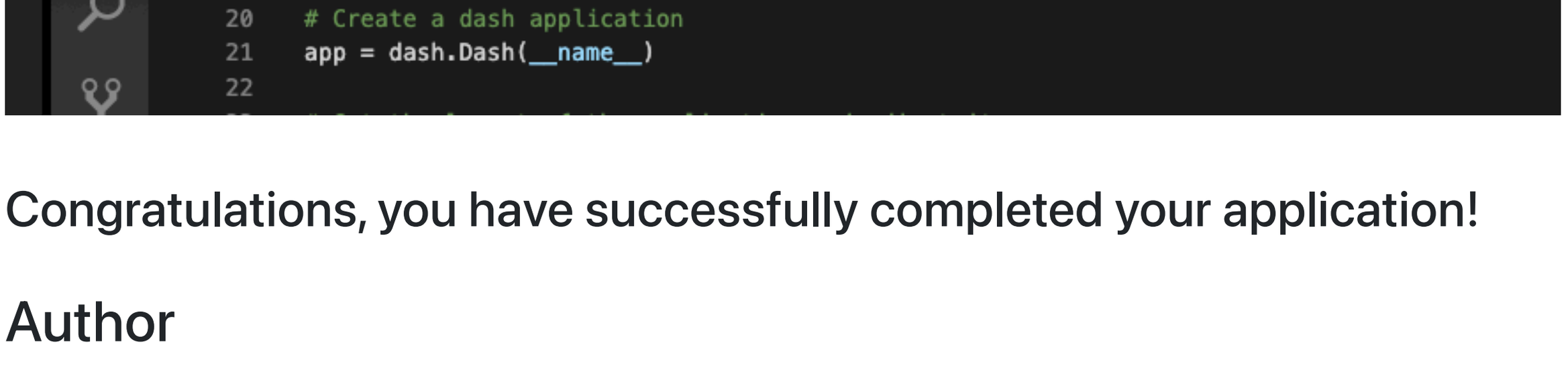
- Observe the port number shown in the terminal.



- Click on the `Launch Application` option from the menu bar.



- Provide the port number and click `OK`



Congratulations, you have successfully completed your application!

Author

Saishruthi Swaminathan

Changelog

Date	Version	Changed by	Change Description
05-10-2021	1.0	Saishruthi	Initial version created