Ganpat University | Institute of Computer Technology

॥ विद्यया समाजोत्कर्षः ॥

**Name: Tushar Panchal**

**En.No: 21162101014**

**Sub: EADC (Enterprise Application Development for Cloud)**

**Branch: CBA**

**Batch:61**

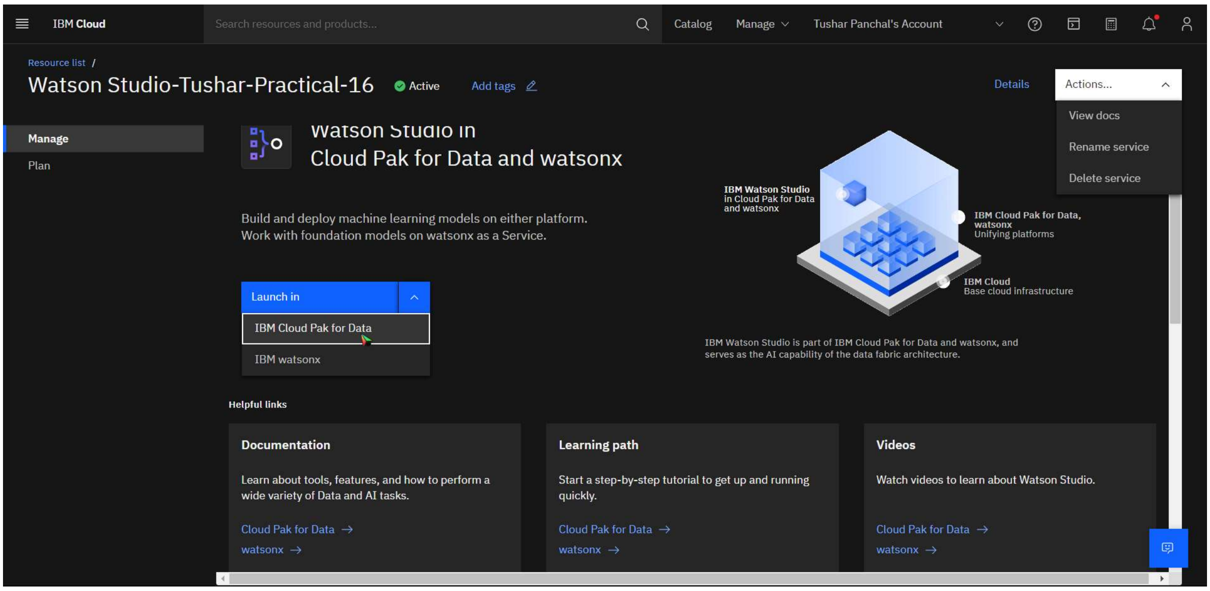
## ---------------------------PRACTICAL 18----------------------------

**Assume you are working in a company where you need to extract useful insights from the data collected by organization. Demonstrate how to analyze large datasets with Python data science packages. We'll provide an example use case of analysing hourly air quality data provided by the EPA.**

Perform the following Tasks:
1. Create a Juypter notebook in Watson Studio.
2. Extract patterns from datasets using pandas.
3. Visualize data trends via matplotlib graphs.

## Task 1 : Create a Juypter notebook in Watson Studio.

Navigate watson studio that we created in practical - 16 than click on launch in IBM cloud Pak for Data



Watson Studio in Cloud Pak for Data and watsonx

Build and deploy machine learning models on either platform. Work with foundation models on watsonx as a Service.

IBM Watson Studio is part of IBM Cloud Pak for Data and watsonx, and serves as the AI capability of the data fabric architecture.

**Documentation**

Learn about tools, features, and how to perform a wide variety of Data and AI tasks.

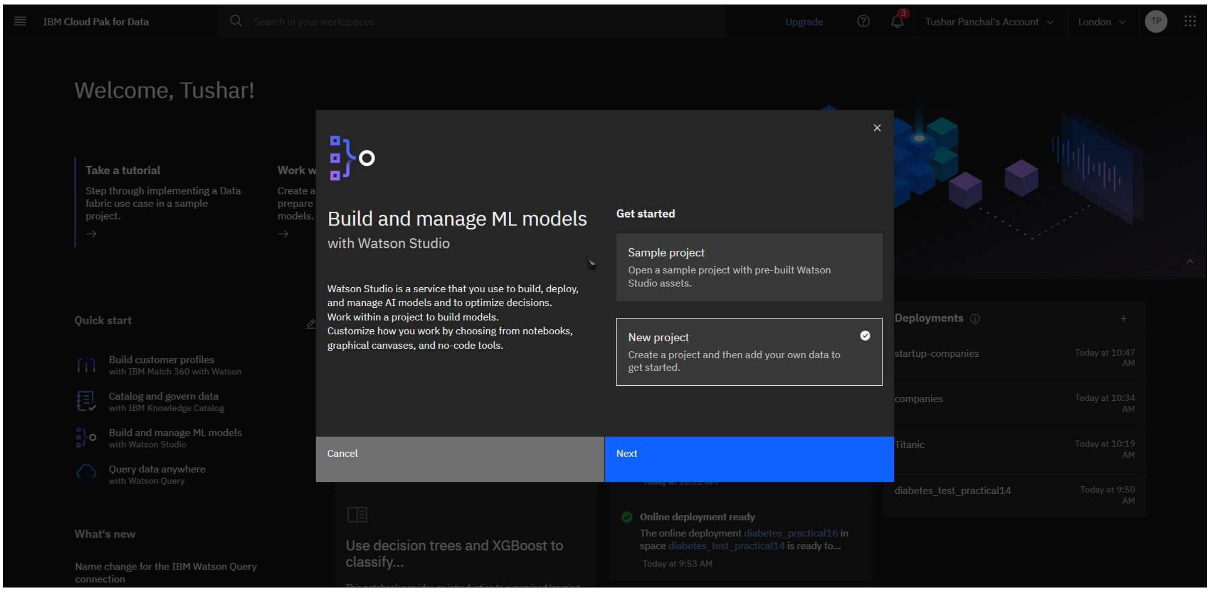
**Learning path**

Start a step-by-step tutorial to get up and running quickly.

**Videos**

Watch videos to learn about Watson Studio.

Now click on new project and click next


IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Tushar Panchal's Account
London
TP
Welcome, Tushar!
Take a tutorial
Step through implementing a Data fabric use case in a sample project.
Quick start
Build customer profiles with IBM Match 360 with Watson
Catalog and govern data with IBM Knowledge Catalog
Build and manage ML models with Watson Studio
Query data anywhere with Watson Query
What's new
Name change for the IBM Watson Query connection
Use decision trees and XGBoost to classify...
Online deployment ready
Deployments
startup-companies
companies
Titanic
diabetes_test_practical14
Cancel
Next

Build and manage ML models

with Watson Studio

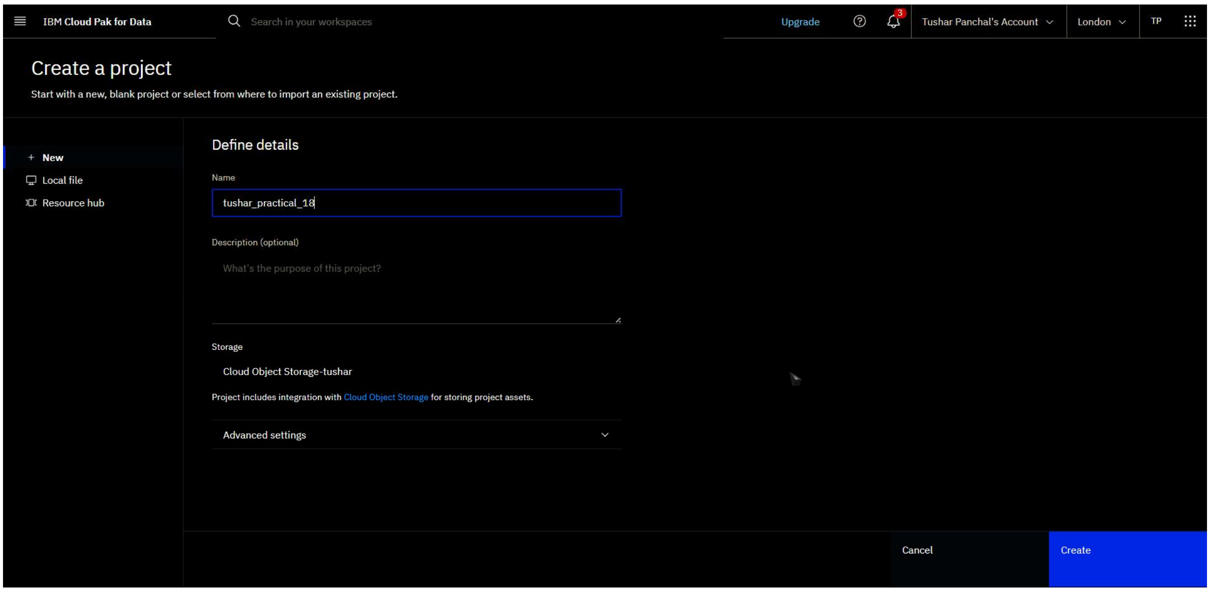
Watson Studio is a service that you use to build, deploy, and manage AI models and to optimize decisions. Work within a project to build models. Customize how you work by choosing from notebooks, graphical canvases, and no-code tools.

**Get started**

Sample project
Open a sample project with pre-built Watson Studio assets.

New project
Create a project and then add your own data to get started.

## Give name of the new project and click on create



Create a project

Start with a new, blank project or select from where to import an existing project.

Define details

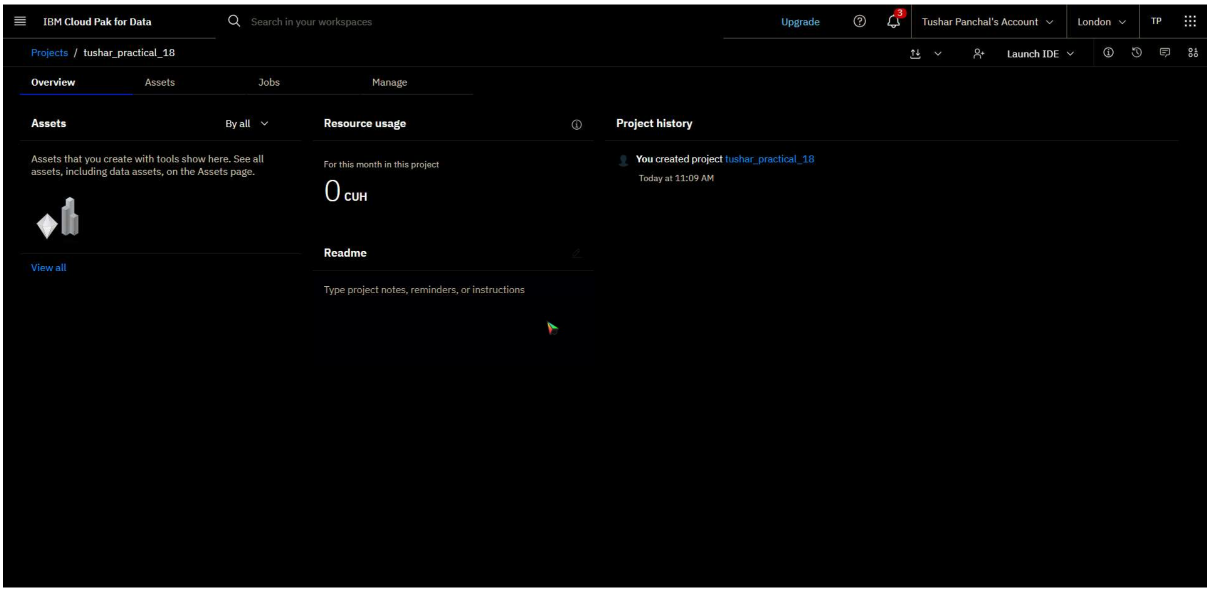
Name: tushar_practical_18

Description (optional): What's the purpose of this project?

Storage: Cloud Object Storage-tushar

Project includes integration with Cloud Object Storage for storing project assets.

## Its open this interface



Assets that you create with tools show here. See all assets, including data assets, on the Assets page.

For this month in this project

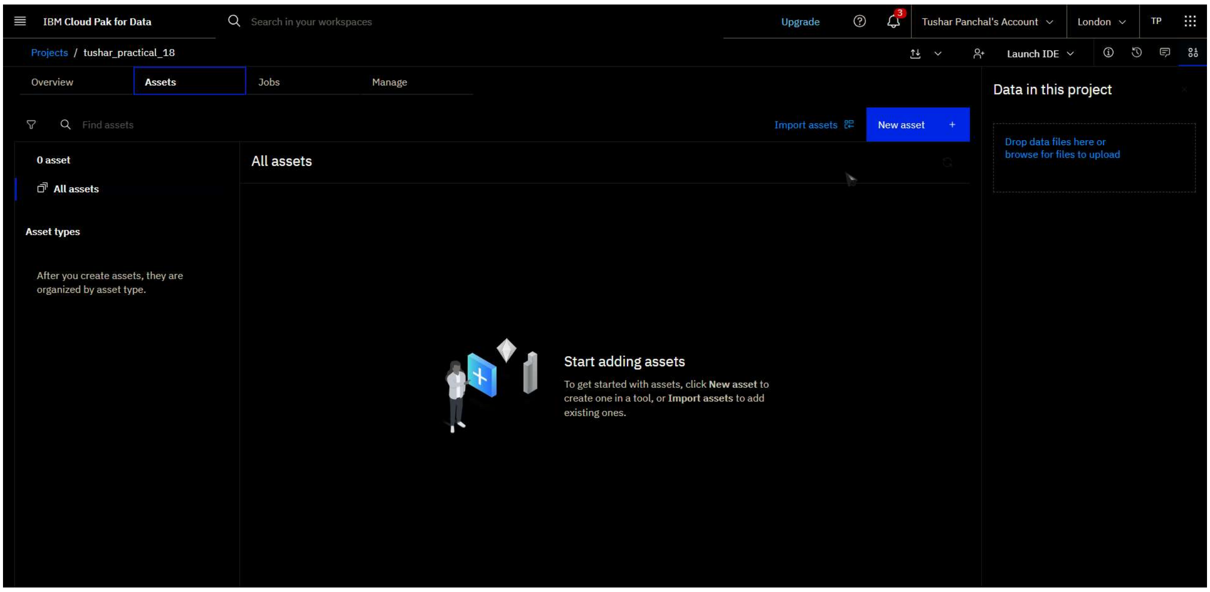
0 CUH

Type project notes, reminders, or instructions

You created project tushar_practical_18

Today at 11:09 AM

Select assets then click on new asset



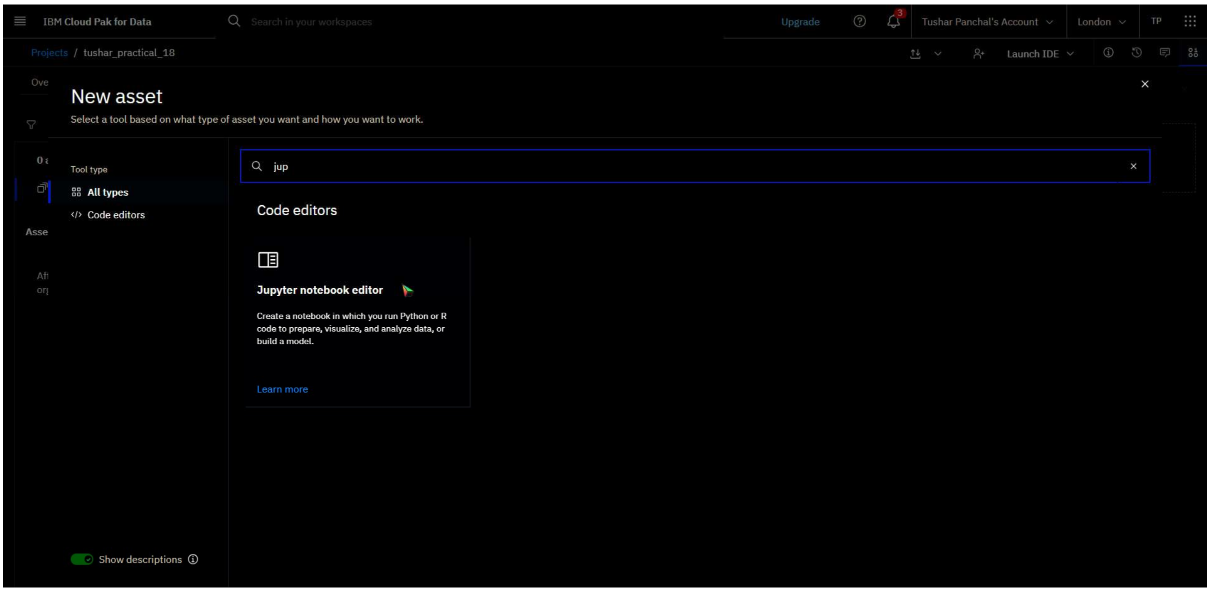
After you create assets, they are organized by asset type.

Start adding assets

To get started with assets, click **New asset** to create one in a tool, or **Import assets** to add existing ones.

Drop data files here or browse for files to upload

Now search jupyter than click on “jupyter notebook editor”



New asset

Select a tool based on what type of asset you want and how you want to work.

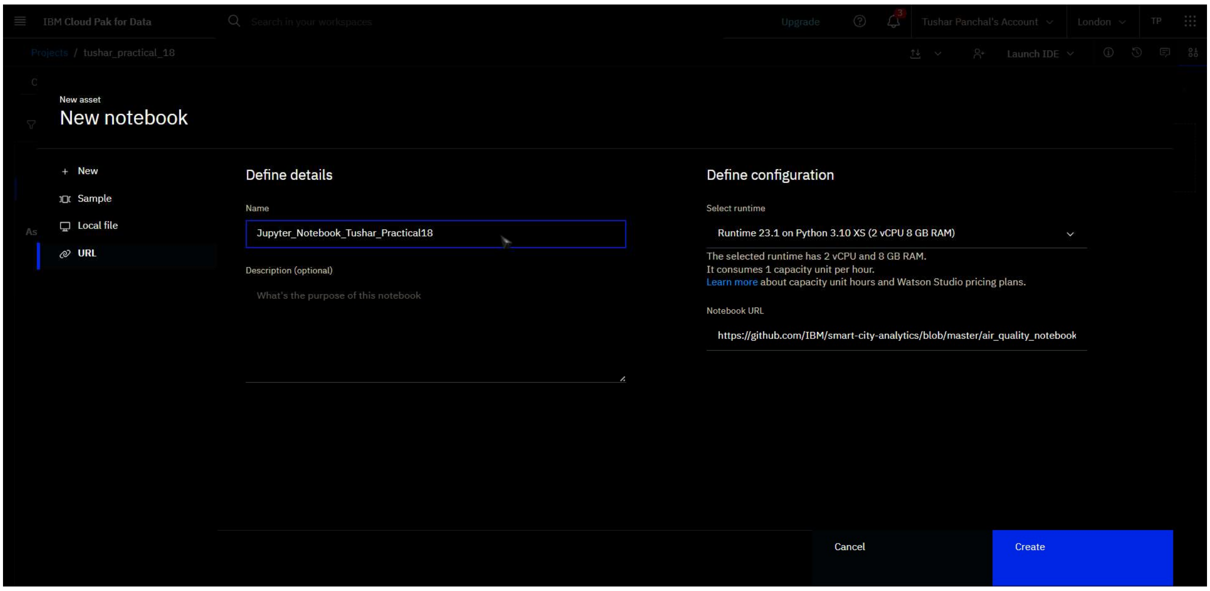
jup

**Jupyter notebook editor**

Create a notebook in which you run Python or R code to prepare, visualize, and analyze data, or build a model.

Now add name and select “URL” give below URl Then click on create

https://github.com/IBM/smart-city-analytics/blob/master/air_quality_notebook.ipynb



Define details

Name: Jupyter_Notebook_Tushar_Practical18

Description (optional): What's the purpose of this notebook

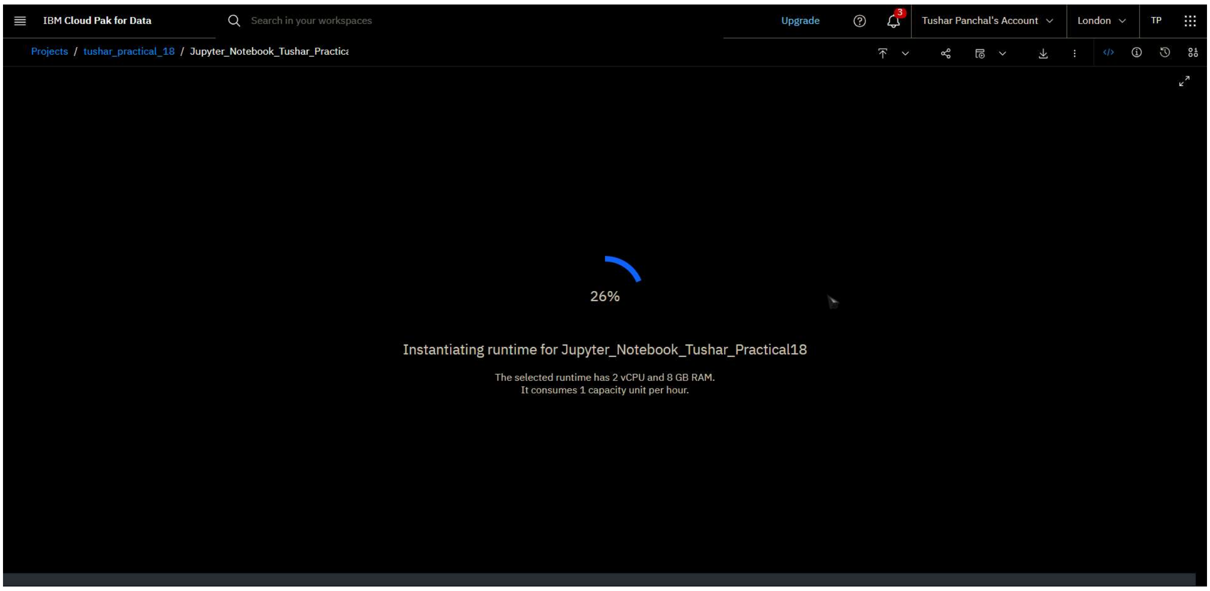
Define configuration

Select runtime: Runtime 23.1 on Python 3.10 XS (2 vCPU 8 GB RAM)

The selected runtime has 2 vCPU and 8 GB RAM.
It consumes 1 capacity unit per hour.
Learn more about capacity unit hours and Watson Studio pricing plans.

Notebook URL: https://github.com/IBM/smart-city-analytics/blob/master/air_quality_notebook

After create its loading the interface

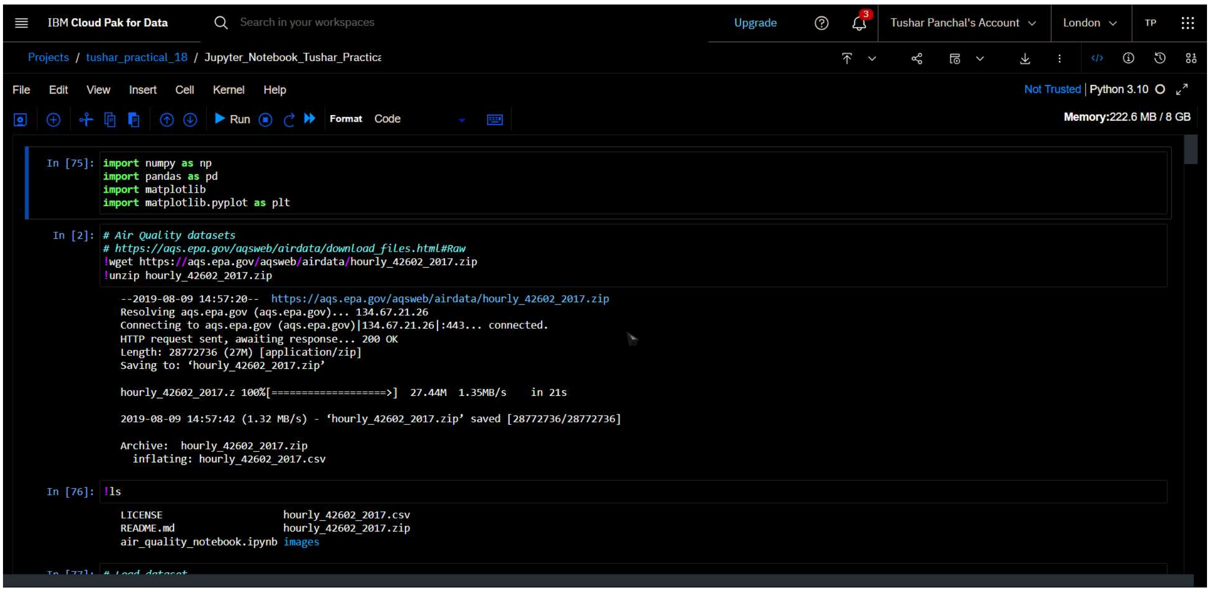

IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Tushar Panchal's Account
London
TP
Projects / tushar_practical_18 / Jupyter_Notebook_Tushar_Practica

26%

Instantiating runtime for Jupyter_Notebook_Tushar_Practical18

The selected runtime has 2 vCPU and 8 GB RAM.
It consumes 1 capacity unit per hour.

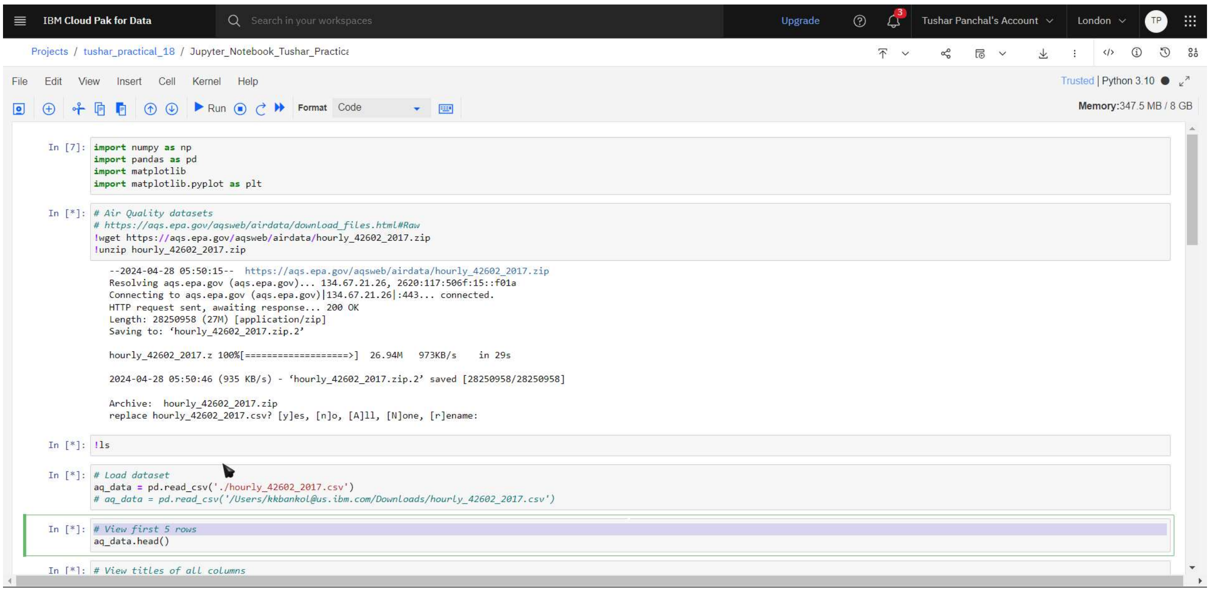
As we can see in below screenshot all query


IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Tushar Panchal's Account
London
TP
Projects / tushar_practical_18 / Jupyter_Notebook_Tushar_Practica
File Edit View Insert Cell Kernel Help
Not Trusted | Python 3.10
Run
Format Code
Memory:222.6 MB / 8 GB

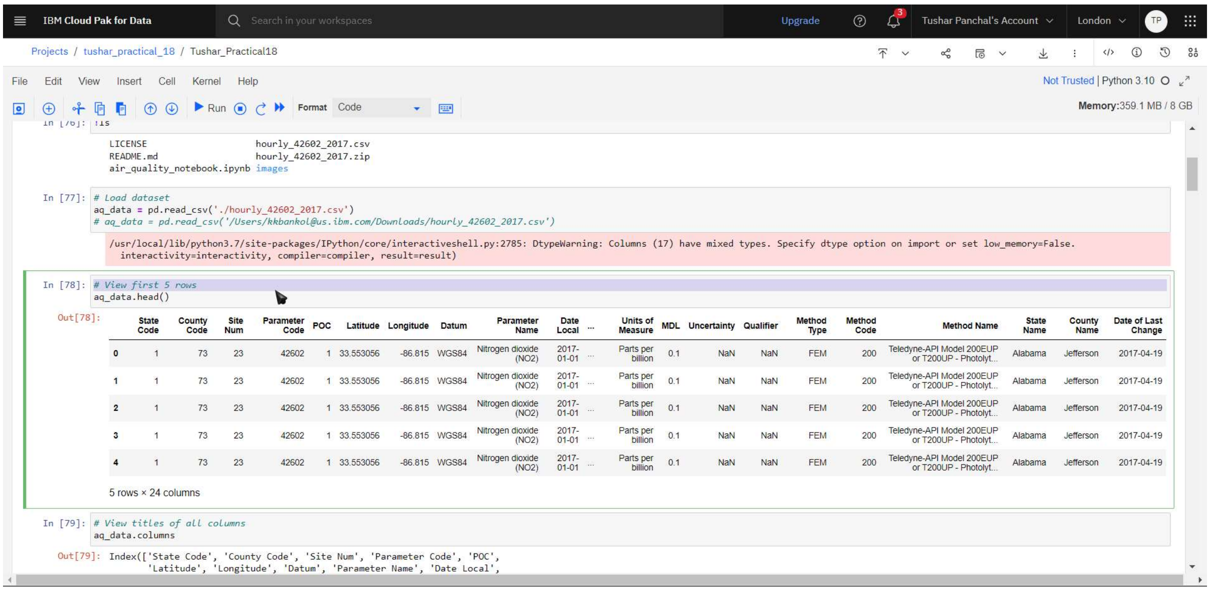
```
In [75]: import numpy as np
         import pandas as pd
         import matplotlib
         import matplotlib.pyplot as plt

In [2]:  # Air Quality datasets
         # https://aqs.epa.gov/aqsweb/airdata/download_files.html#Raw
         !wget https://aqs.epa.gov/aqsweb/airdata/hourly_42602_2017.zip
         !unzip hourly_42602_2017.zip

           --2019-08-09 14:57:20--  https://aqs.epa.gov/aqsweb/airdata/hourly_42602_2017.zip
           Resolving aqs.epa.gov (aqs.epa.gov)... 134.67.21.26
           Connecting to aqs.epa.gov (aqs.epa.gov)|134.67.21.26|:443... connected.
           HTTP request sent, awaiting response... 200 OK
           Length: 28772736 (27M) [application/zip]
           Saving to: 'hourly_42602_2017.zip'

           hourly_42602_2017.z 100%[===================>]  27.44M  1.35MB/s    in 21s

           2019-08-09 14:57:42 (1.32 MB/s) - 'hourly_42602_2017.zip' saved [28772736/28772736]

           Archive:  hourly_42602_2017.zip
             inflating: hourly_42602_2017.csv

In [76]: !ls

           LICENSE                    hourly_42602_2017.csv
           README.md                  hourly_42602_2017.zip
           air_quality_notebook.ipynb images
```

Now run one by one all codes

## Task 2 : Extract patterns from datasets using pandas.


IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Tushar Panchal's Account
London
TP
Projects / tushar_practical_18 / Jupyter_Notebook_Tushar_Practica
File Edit View Insert Cell Kernel Help
Trusted | Python 3.10
Run
Format Code
Memory:347.5 MB / 8 GB

```
In [7]: import numpy as np
        import pandas as pd
        import matplotlib
        import matplotlib.pyplot as plt

In [*]: # Air Quality datasets
        # https://aqs.epa.gov/aqsweb/airdata/download_files.html#Raw
        !wget https://aqs.epa.gov/aqsweb/airdata/hourly_42602_2017.zip
        !unzip hourly_42602_2017.zip

          --2024-04-28 05:50:15--  https://aqs.epa.gov/aqsweb/airdata/hourly_42602_2017.zip
          Resolving aqs.epa.gov (aqs.epa.gov)... 134.67.21.26, 2620:117:506f:15::f01a
          Connecting to aqs.epa.gov (aqs.epa.gov)|134.67.21.26|:443... connected.
          HTTP request sent, awaiting response... 200 OK
          Length: 28250958 (27M) [application/zip]
          Saving to: 'hourly_42602_2017.zip.2'

          hourly_42602_2017.z 100%[===================>]  26.94M   973KB/s    in 29s

          2024-04-28 05:50:46 (935 KB/s) - 'hourly_42602_2017.zip.2' saved [28250958/28250958]

          Archive:  hourly_42602_2017.zip
          replace hourly_42602_2017.csv? [y]es, [n]o, [A]ll, [N]one, [r]ename:

In [*]: !ls

In [*]: # Load dataset
        aq_data = pd.read_csv('./hourly_42602_2017.csv')
        # aq_data = pd.read_csv('/Users/kkbankol@us.ibm.com/Downloads/hourly_42602_2017.csv')

In [*]: # View first 5 rows
        aq_data.head()

In [*]: # View titles of all columns
```



```
In [76]: !ls

         LICENSE                   hourly_42602_2017.csv
         README.md                 hourly_42602_2017.zip
         air_quality_notebook.ipynb images
```

```
In [77]: # Load dataset
         aq_data = pd.read_csv('./hourly_42602_2017.csv')
         # aq_data = pd.read_csv('/Users/kkbankol@us.ibm.com/Downloads/hourly_42602_2017.csv')

         /usr/local/lib/python3.7/site-packages/IPython/core/interactiveshell.py:2785: DtypeWarning: Columns (17) have mixed types. Specify dtype option on import or set low_memory=False.
           interactivity=interactivity, compiler=compiler, result=result)
```

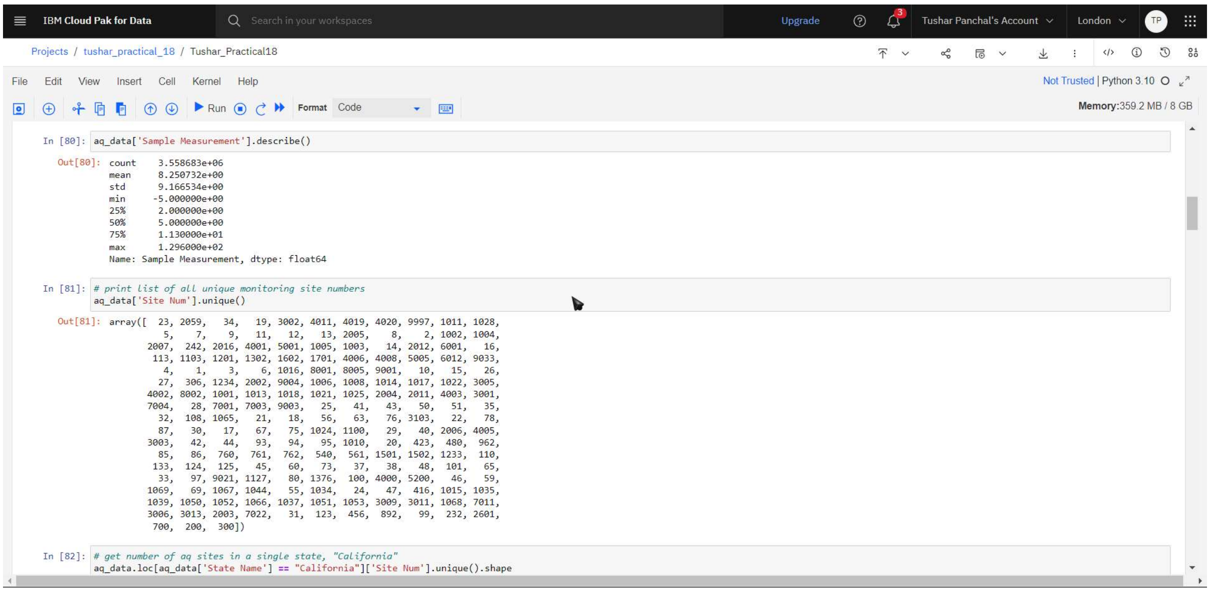
```
In [78]: # View first 5 rows
         aq_data.head()
```

Out[78]:

| | State Code | County Code | Site Num | Parameter Code | POC | Latitude | Longitude | Datum | Parameter Name | Date Local | ... | Units of Measure | MDL | Uncertainty | Qualifier | Method Type | Method Code | Method Name | State Name | County Name | Date of Last Change |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 73 | 23 | 42602 | 1 | 33.553056 | -86.815 | WGS84 | Nitrogen dioxide (NO2) | 2017-01-01 | ... | Parts per billion | 0.1 | NaN | NaN | FEM | 200 | Teledyne-API Model 200EUP or T200UP - Photolyt... | Alabama | Jefferson | 2017-04-19 |
| 1 | 1 | 73 | 23 | 42602 | 1 | 33.553056 | -86.815 | WGS84 | Nitrogen dioxide (NO2) | 2017-01-01 | ... | Parts per billion | 0.1 | NaN | NaN | FEM | 200 | Teledyne-API Model 200EUP or T200UP - Photolyt... | Alabama | Jefferson | 2017-04-19 |
| 2 | 1 | 73 | 23 | 42602 | 1 | 33.553056 | -86.815 | WGS84 | Nitrogen dioxide (NO2) | 2017-01-01 | ... | Parts per billion | 0.1 | NaN | NaN | FEM | 200 | Teledyne-API Model 200EUP or T200UP - Photolyt... | Alabama | Jefferson | 2017-04-19 |
| 3 | 1 | 73 | 23 | 42602 | 1 | 33.553056 | -86.815 | WGS84 | Nitrogen dioxide (NO2) | 2017-01-01 | ... | Parts per billion | 0.1 | NaN | NaN | FEM | 200 | Teledyne-API Model 200EUP or T200UP - Photolyt... | Alabama | Jefferson | 2017-04-19 |
| 4 | 1 | 73 | 23 | 42602 | 1 | 33.553056 | -86.815 | WGS84 | Nitrogen dioxide (NO2) | 2017-01-01 | ... | Parts per billion | 0.1 | NaN | NaN | FEM | 200 | Teledyne-API Model 200EUP or T200UP - Photolyt... | Alabama | Jefferson | 2017-04-19 |

5 rows × 24 columns

```
In [79]: # View titles of all columns
         aq_data.columns

Out[79]: Index(['State Code', 'County Code', 'Site Num', 'Parameter Code', 'POC',
                'Latitude', 'Longitude', 'Datum', 'Parameter Name', 'Date Local',
```

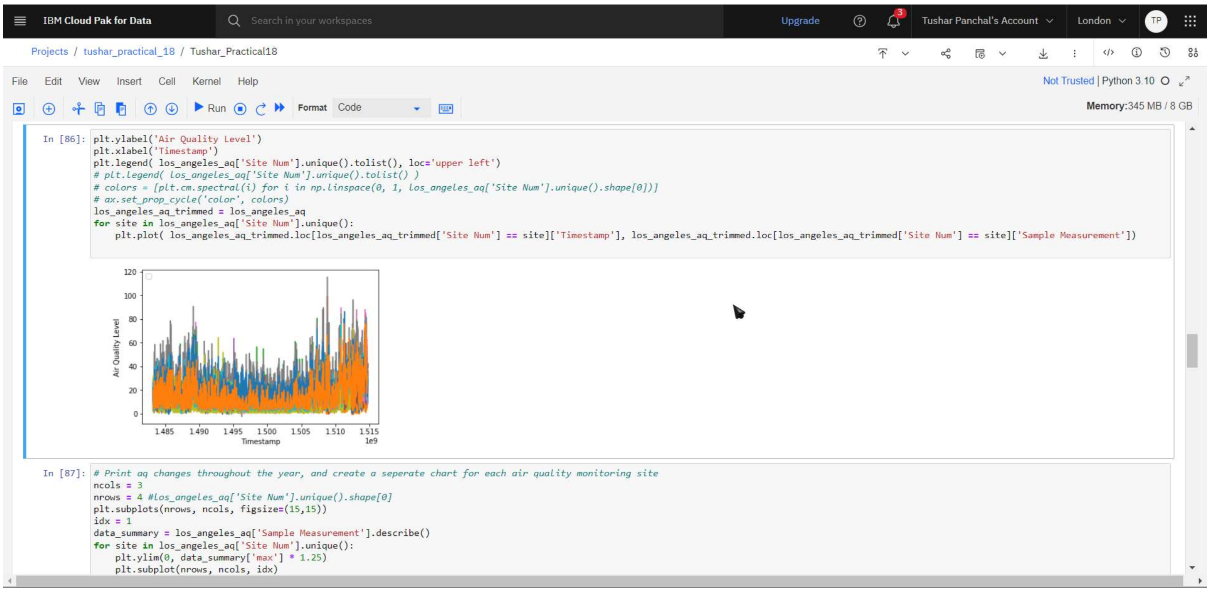


```
In [80]: aq_data['Sample Measurement'].describe()

Out[80]: count    3.558683e+06
         mean     8.250732e+00
         std      9.166534e+00
         min     -5.000000e+00
         25%      2.000000e+00
         50%      5.000000e+00
         75%      1.130000e+01
         max      1.296000e+02
         Name: Sample Measurement, dtype: float64
```

```
In [81]: # print list of all unique monitoring site numbers
         aq_data['Site Num'].unique()

Out[81]: array([  23, 2059,   34,   19, 3002, 4011, 4019, 4020, 9997, 1011, 1028,
                  5,    7,    9,   11,   12,   13, 2005,    8,    2, 1002, 1004,
               2007,  242, 2016, 4001, 5001, 1005, 1003,   14, 2012, 6001,   16,
                113, 1103, 1201, 1302, 1602, 1701, 4006, 4008, 5005, 6012, 9033,
                  4,    1,    3,    6, 1016, 8001, 8005, 9001,   10,   15,   26,
                 27,  306, 1234, 2002, 9004, 1006, 1008, 1014, 1017, 1022, 3005,
               4002, 8002, 1001, 1013, 1018, 1021, 1025, 2004, 2011, 4003, 3001,
               7004,   28, 7001, 7003, 9003,   25,   41,   43,   50,   51,   35,
                 32,  108, 1065,   21,   18,   56,   63,   76, 3103,   22,   78,
                 87,   30,   17,   67,   75, 1024, 1100,   29,   40, 2006, 4005,
               3003,   42,   44,   93,   94,   95, 1010,   20,  423,  480,  962,
                 85,   86,  760,  761,  762,  540,  561, 1501, 1502, 1233,  110,
                133,  124,  125,   45,   60,   73,   37,   38,   48,  101,   65,
                 33,   97, 9021, 1127,   80, 1376,  100, 4000, 5200,   46,   59,
               1069,   69, 1067, 1044,   55, 1034,   24,   47,  416, 1015, 1035,
               1039, 1050, 1052, 1066, 1037, 1051, 1053, 3009, 3011, 1068, 7011,
               3006, 3013, 2003, 7022,   31,  123,  456,  892,   99,  232, 2601,
                700,  200,  300])
```

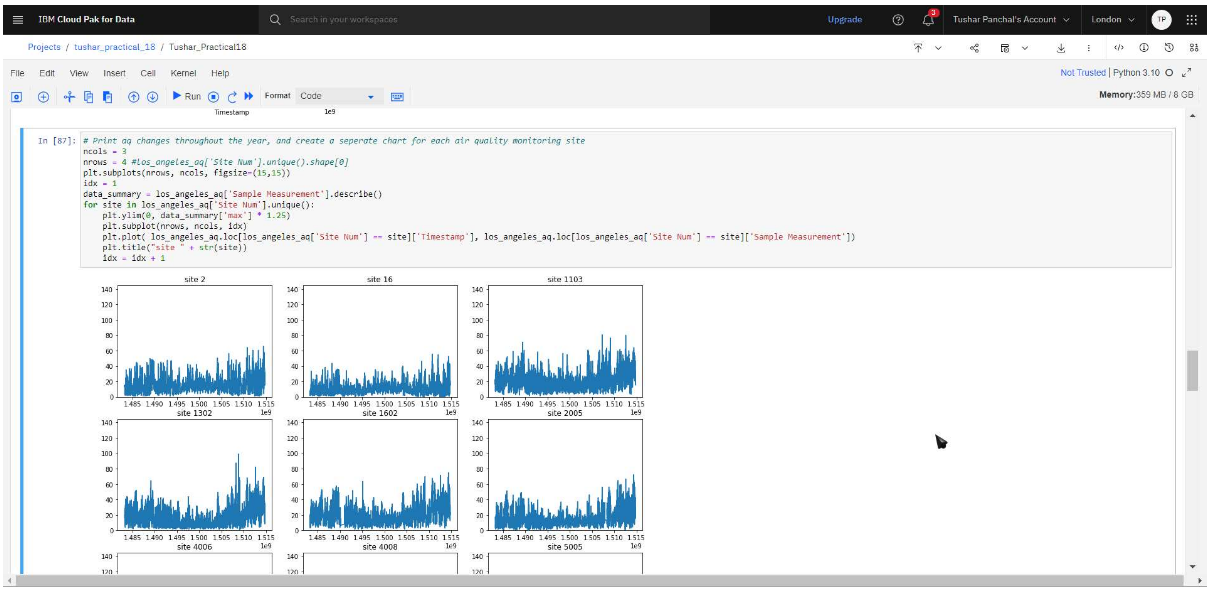
```
In [82]: # get number of aq sites in a single state, "California"
         aq_data.loc[aq_data['State Name'] == "California"]['Site Num'].unique().shape
```

## » Task 3 : Visualize data trends via matplotlib graphs.

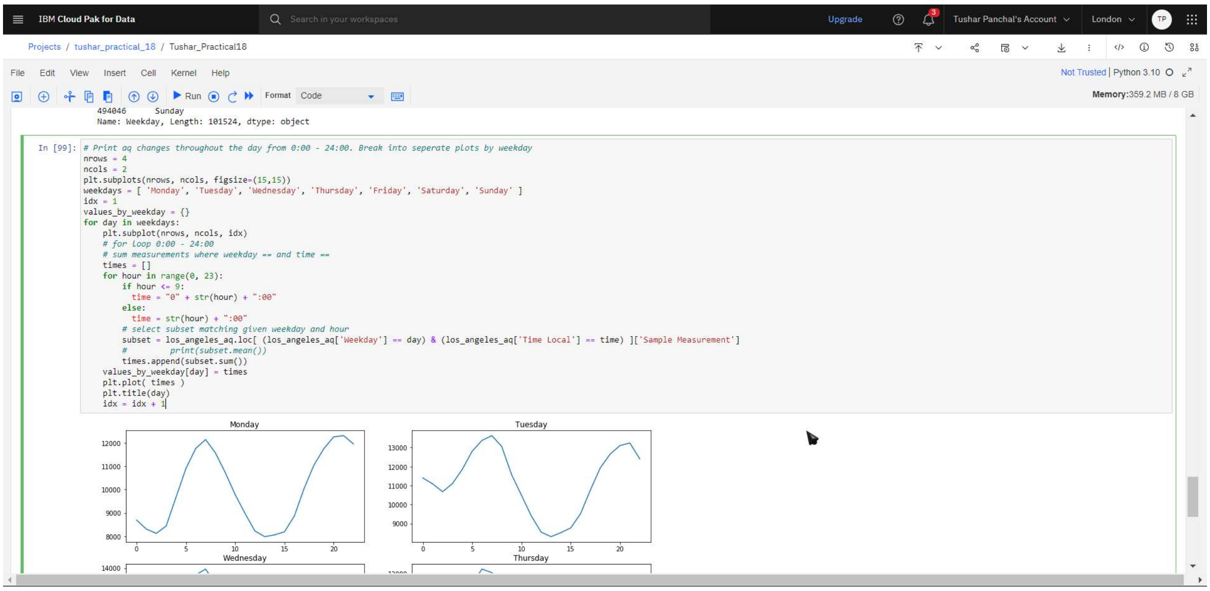


```
In [86]: plt.ylabel('Air Quality Level')
         plt.xlabel('Timestamp')
         plt.legend( los_angeles_aq['Site Num'].unique().tolist(), loc='upper left')
         # plt.legend( los_angeles_aq['Site Num'].unique().tolist() )
         # colors = [plt.cm.spectral(i) for i in np.linspace(0, 1, los_angeles_aq['Site Num'].unique().shape[0])]
         # ax.set_prop_cycle('color', colors)
         los_angeles_aq_trimmed = los_angeles_aq
         for site in los_angeles_aq['Site Num'].unique():
             plt.plot( los_angeles_aq_trimmed.loc[los_angeles_aq_trimmed['Site Num'] == site]['Timestamp'], los_angeles_aq_trimmed.loc[los_angeles_aq_trimmed['Site Num'] == site]['Sample Measurement'])
```

```
In [87]: # Print aq changes throughout the year, and create a seperate chart for each air quality monitoring site
         ncols = 3
         nrows = 4 #los_angeles_aq['Site Num'].unique().shape[0]
         plt.subplots(nrows, ncols, figsize=(15,15))
         idx = 1
         data_summary = los_angeles_aq['Sample Measurement'].describe()
         for site in los_angeles_aq['Site Num'].unique():
             plt.ylim(0, data_summary['max'] * 1.25)
             plt.subplot(nrows, ncols, idx)
```



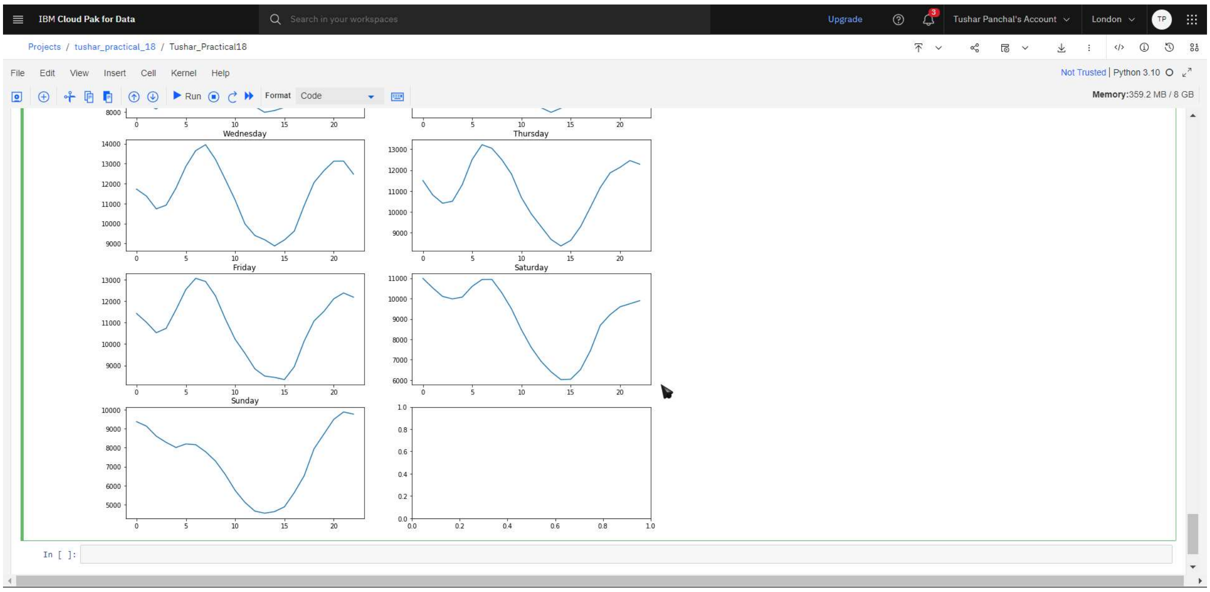
```
In [87]: # Print aq changes throughout the year, and create a seperate chart for each air quality monitoring site
         ncols = 3
         nrows = 4 #los_angeles_aq['Site Num'].unique().shape[0]
         plt.subplots(nrows, ncols, figsize=(15,15))
         idx = 1
         data_summary = los_angeles_aq['Sample Measurement'].describe()
         for site in los_angeles_aq['Site Num'].unique():
             plt.ylim(0, data_summary['max'] * 1.25)
             plt.subplot(nrows, ncols, idx)
             plt.plot( los_angeles_aq.loc[los_angeles_aq['Site Num'] == site]['Timestamp'], los_angeles_aq.loc[los_angeles_aq['Site Num'] == site]['Sample Measurement'])
             plt.title("site " + str(site))
             idx = idx + 1
```



```
494046      Sunday
Name: Weekday, Length: 101524, dtype: object
```

```
In [99]: # Print aq changes throughout the day from 0:00 - 24:00. Break into seperate plots by weekday
nrows = 4
ncols = 2
plt.subplots(nrows, ncols, figsize=(15,15))
weekdays = [ 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday' ]
idx = 1
values_by_weekday = {}
for day in weekdays:
    plt.subplot(nrows, ncols, idx)
    # for loop 0:00 - 24:00
    # sum measurements where weekday == and time ==
    times = []
    for hour in range(0, 23):
        if hour <= 9:
          time = "0" + str(hour) + ":00"
        else:
          time = str(hour) + ":00"
        # select subset matching given weekday and hour
        subset = los_angeles_aq.loc[ (los_angeles_aq['Weekday'] == day) & (los_angeles_aq['Time Local'] == time) ]['Sample Measurement']
        #         print(subset.mean())
        times.append(subset.sum())
    values_by_weekday[day] = times
    plt.plot( times )
    plt.title(day)
    idx = idx + 1
```

IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Tushar Panchal's Account
London
TP
Projects / tushar_practical_18 / Tushar_Practical18
File
Edit
View
Insert
Cell
Kernel
Help
Not Trusted | Python 3.10
Run
Format
Code
Memory:359.2 MB / 8 GB
Monday
Tuesday
Wednesday
Thursday



IBM Cloud Pak for Data
Search in your workspaces
Upgrade
Tushar Panchal's Account
London
TP
Projects / tushar_practical_18 / Tushar_Practical18
File
Edit
View
Insert
Cell
Kernel
Help
Not Trusted | Python 3.10
Run
Format
Code
Memory:359.2 MB / 8 GB
Wednesday
Thursday
Friday
Saturday
Sunday
In [ ]: