Daily Public Transport Passenger Boardings by Ticket Type

Overview of the dataset:

```
print(df.head())

        Date  MyWay  Paper Ticket
0 2019-07-01  66215          4325
1 2023-09-15  63800          7349
2 2021-12-28   9994          1882
3 2023-01-11  43769          3991
4 2021-09-11   3810           685

print(df.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1778 entries, 0 to 1777
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Date           1778 non-null   datetime64[ns]
 1   MyWay          1778 non-null   int64
 2   Paper Ticket   1778 non-null   int64
dtypes: datetime64[ns](1), int64(2)
memory usage: 41.8 KB
None
```

TASK 1:

- ✓ To filter a last one year data from the given dataset.

Filtered the data by 1 Year (2023-04-01 to 2024-04-01):

```
[161] df['Date'] = pd.to_datetime(df['Date'], infer_datetime_format=True)

<ipython-input-161-45a55d54a14d>:1: UserWarning: The argument 'infer_datetime_format' is deprecated and will be removed in a future version. A strict version c
  df['Date'] = pd.to_datetime(df['Date'], infer_datetime_format=True)

[162]
df_filtered = df[df['Date'].between('2023-04-01', '2024-04-01')]

Filtered 1 year data

df_filtered
```

	Date	MyWay	Paper Ticket
1	2023-09-15	63800	7349
10	2023-08-29	69155	6507
19	2023-04-11	50246	4306
27	2023-09-17	19207	2368
33	2023-10-07	24249	3089
...	...	...	...
1766	2023-11-25	23664	4617
1767	2024-03-20	76782	1346
1768	2023-07-13	49311	4309
1774	2023-08-16	70115	6955
1777	2023-07-26	69067	6811

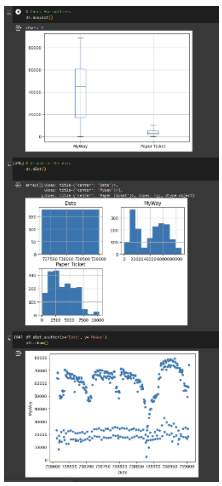
367 rows × 3 columns

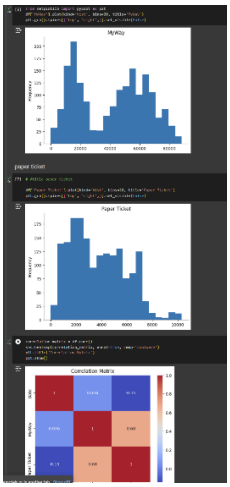
EDA Techniques:

- ✓ Undertake Exploratory Data Analysis
- ✓ Undertake any preliminary data analysis as required.

```
[154] df= df.drop_duplicates()

[155] # Check for missing values
      df.isnull().sum()

Date            0
MyWay           0
Paper Ticket    0
dtype: int64

df.describe()
```

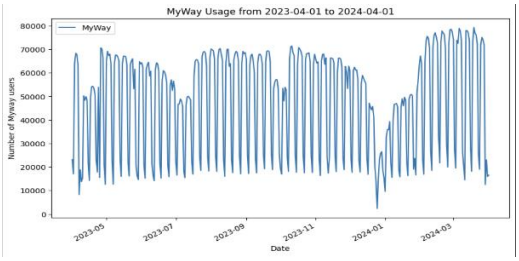
	MyWay	Paper Ticket
count	1778.000000	1778.000000
mean	40985.889201	3744.153543
std	23275.162569	2153.562760
min	0.000000	13.000000
25%	17207.750000	1991.500000
50%	45079.500000	3374.500000
75%	60946.500000	5383.000000
max	88313.000000	10310.000000



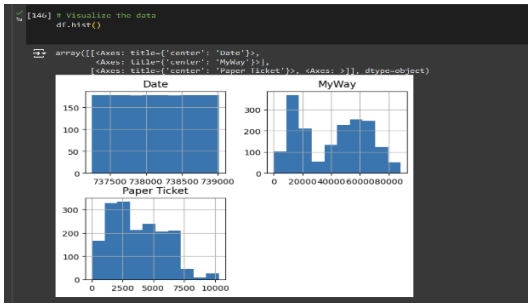


INFERENCES:

Data of my usage from last 1 year:



HISTOGRAM:



- ✓ The distribution of passenger boardings is skewed to the right, with a long tail on the right side. This indicates that there are a few days with very high passenger boardings, while most days have relatively low passenger boardings.
- ✓ The most frequent passenger boardings is around 50,000, which occurs on about 10 days.

TASK 2:

Use only MyWay column to predict patronage for next 7 days (1st May 2024 – 7th May 2024)

Predicted output:

```
print("Date\t\tPredicted")

for date, prediction in zip(next_week, next_week_predictions):
    print(f"{date.strftime('%Y-%m-%d')}:\t{int(prediction)}")
```

```
Date            Predicted
2024-05-01:     75575
2024-05-02:     75606
2024-05-03:     75638
2024-05-04:     75669
2024-05-05:     75700
2024-05-06:     75732
2024-05-07:     75763
```

way 2

```
import pandas as pd
from sklearn.linear_model import LinearRegression

next_7_days = pd.date_range(df_filtered['Date'].max() + pd.Timedelta(days=1), periods=7)

# Convert dates to timestamps (numeric representation)
next_7_days_numeric = next_7_days.astype(int).values.reshape(-1, 1)

# Predict the MyWay values for the next 7 days
predicted_myway = model.predict(next_7_days_numeric)

predicted_df = pd.DataFrame({'Date': next_7_days, 'Predicted_MyWay': predicted_myway})

print(predicted_df)
```

```
        Date  Predicted_MyWay
0 2024-04-02     5.572943e+18
1 2024-04-03     5.573224e+18
2 2024-04-04     5.573505e+18
3 2024-04-05     5.573787e+18
4 2024-04-06     5.574068e+18
5 2024-04-07     5.574349e+18
6 2024-04-08     5.574630e+18
```