PUBLIC TRANSPORTATION EFFICIENCY ANALYSIS

TEAM MATES:

C.DEEPAKKUMAR	(952421104017)
E.S JEBIN	(952421104026)
J ABISHEK PAUL	(952421104004)
S K. STANLY	(952421104053)

INTRODUCTION REPORT FOR DATA ANALYSIS:

The primary objective of this analysis is to assess and enhance the efficiency of public transportation systems. By examining key performance indicators, identifying areas for improvement, and proposing data-driven solutions, we aim to contribute to the optimization of public transportation services, resulting in improved accessibility, reduced environmental impact, and enhanced overall urban mobility.

DATA ACCURACY:

Use automated data collection methods whenever possible to minimize manual errors.Implement GPS tracking, automated passenger counters, and other sensor technologies to collect real-time data on routes, stops, and passenger counts.

Regularly update data collection devices and ensure their proper functioning to avoid hardware-related inaccuracies.

Establish data validation protocols to identify and rectify outliers, anomalies, or inconsistencies in the collected data.

Regularly clean and preprocess data to eliminate errors, missing values, and duplicate entries.

Integrate data from multiple sources to create a comprehensive dataset. Ensure compatibility and consistency between different datasets.

Verify that data from various sources, such as ticketing systems, scheduling software, and traffic sensors, align coherently.

DATA QUALITIES :

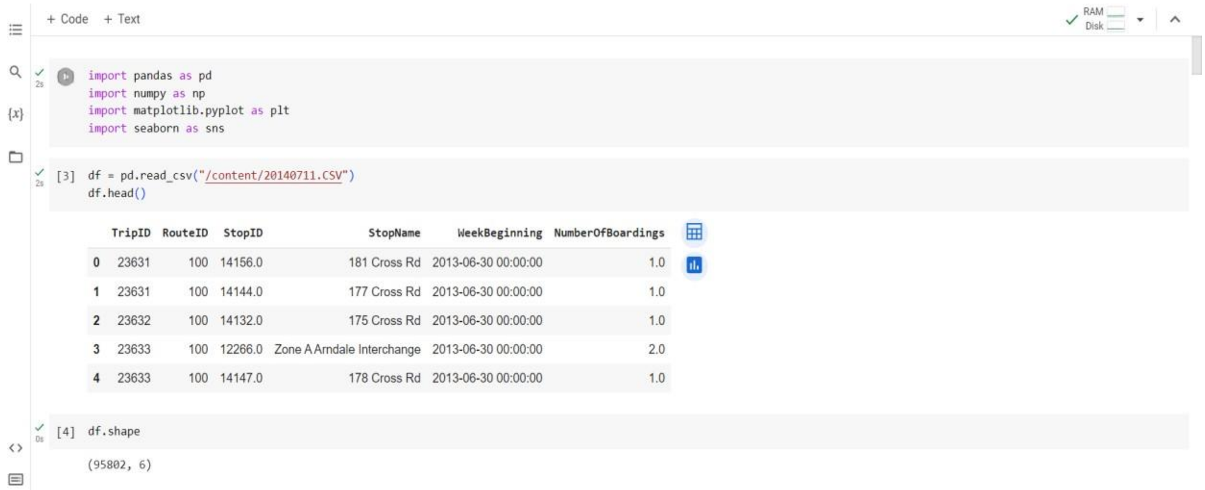
In the context of our public transportation efficiency analysis project, data preprocessing is the critical phase that ensures our dataset is transformed into a clean, organized, and analytically valuable resource. This section outlines the steps taken to prepare our public bus transport data for in-depth analysis and insights.

Data preprocessing involves a series of tasks, including data cleaning, handling missing values, and data formatting. These actions aim to enhance the quality and integrity of our dataset, making it suitable for statistical analysis, modeling, and visualization. Additionally, any transformations or conversions applied to the data will be documented, ensuring transparency in our data preparation process.

Implement a robust quality assurance process to verify the accuracy of the collected data.

Conduct periodic audits to ensure that the data accurately represents the current state of the public transportation system.

CODE SNIPPETS:



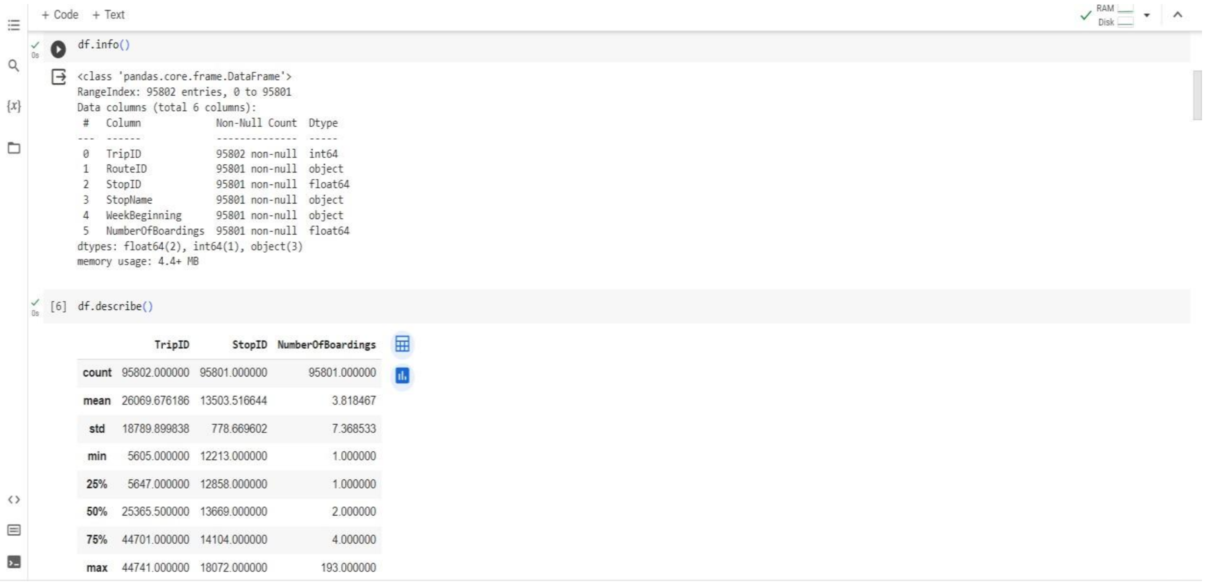
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
[3] df = pd.read_csv("/content/20140711.CSV")
    df.head()
```

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings
0	23631	100	14156.0	181 Cross Rd	2013-06-30 00:00:00	1.0
1	23631	100	14144.0	177 Cross Rd	2013-06-30 00:00:00	1.0
2	23632	100	14132.0	175 Cross Rd	2013-06-30 00:00:00	1.0
3	23633	100	12266.0	Zone A Arndale Interchange	2013-06-30 00:00:00	2.0
4	23633	100	14147.0	178 Cross Rd	2013-06-30 00:00:00	1.0

```
[4] df.shape

(95802, 6)
```

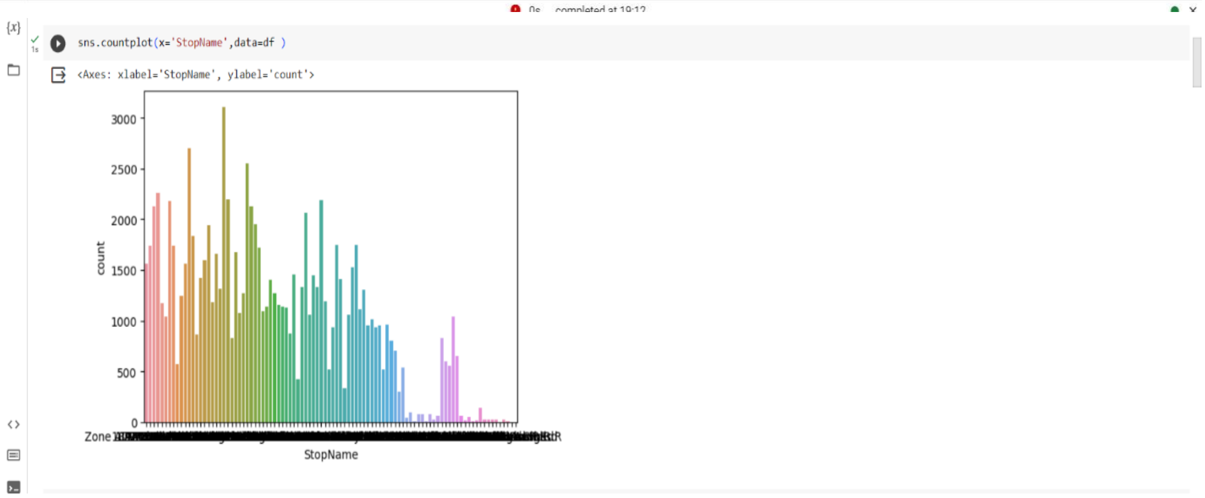


```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 95802 entries, 0 to 95801
Data columns (total 6 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   TripID             95802 non-null  int64
 1   RouteID            95801 non-null  object
 2   StopID             95801 non-null  float64
 3   StopName           95801 non-null  object
 4   WeekBeginning      95801 non-null  object
 5   NumberOfBoardings  95801 non-null  float64
dtypes: float64(2), int64(1), object(3)
memory usage: 4.4+ MB
```

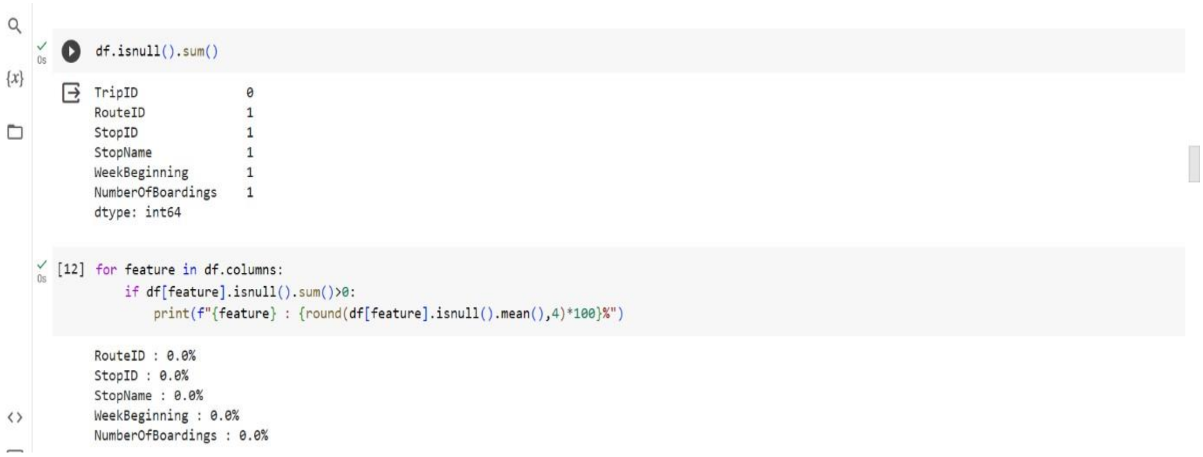
```
[6] df.describe()
```

	TripID	StopID	NumberOfBoardings
count	95802.000000	95801.000000	95801.000000
mean	26069.676186	13503.516644	3.818467
std	18789.899838	778.669602	7.368533
min	5605.000000	12213.000000	1.000000
25%	5647.000000	12858.000000	1.000000
50%	25365.500000	13669.000000	2.000000
75%	44701.000000	14104.000000	4.000000
max	44741.000000	18072.000000	193.000000



```
sns.countplot(x='StopName',data=df )

<Axes: xlabel='StopName', ylabel='count'>
```

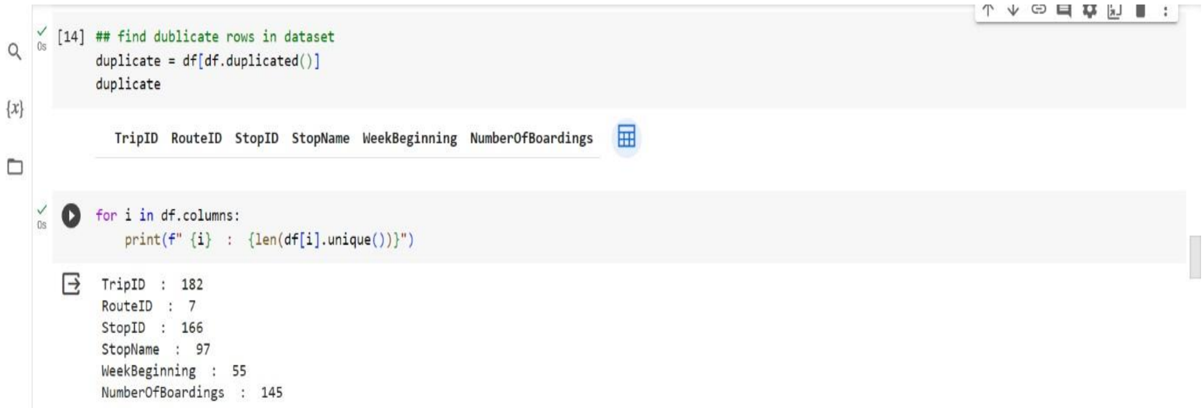


```
df.isnull().sum()
```

```
TripID              0
RouteID             1
StopID              1
StopName            1
WeekBeginning       1
NumberOfBoardings   1
dtype: int64
```

```
[12] for feature in df.columns:
         if df[feature].isnull().sum()>0:
             print(f"{feature} : {round(df[feature].isnull().mean(),4)*100}%")
```

```
RouteID : 0.0%
StopID : 0.0%
StopName : 0.0%
WeekBeginning : 0.0%
NumberOfBoardings : 0.0%
```

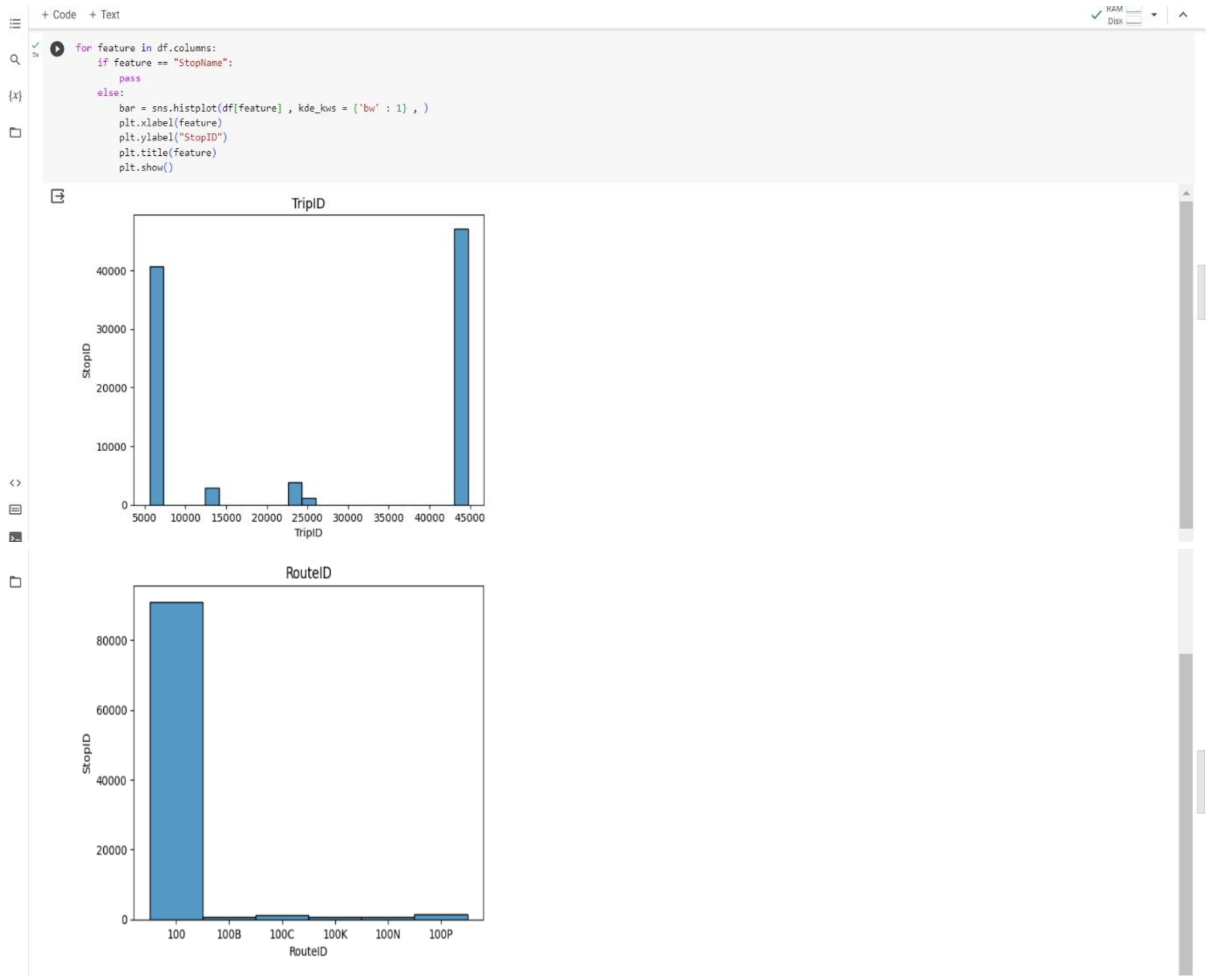


```
[14] ## find dublicate rows in dataset
     duplicate = df[df.duplicated()]
     duplicate
```

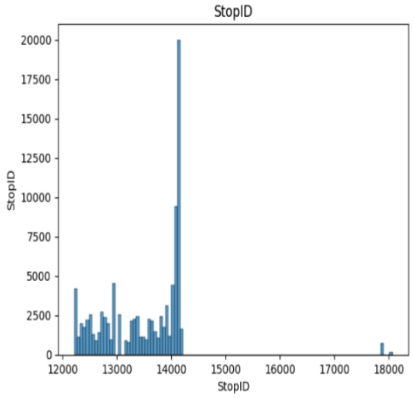
```
  TripID  RouteID  StopID  StopName  WeekBeginning  NumberOfBoardings
```

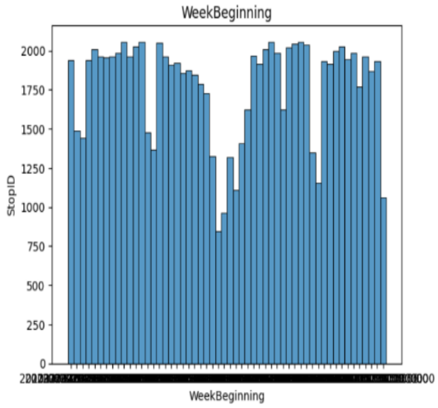
```
for i in df.columns:
    print(f" {i}  :  {len(df[i].unique())}")
```

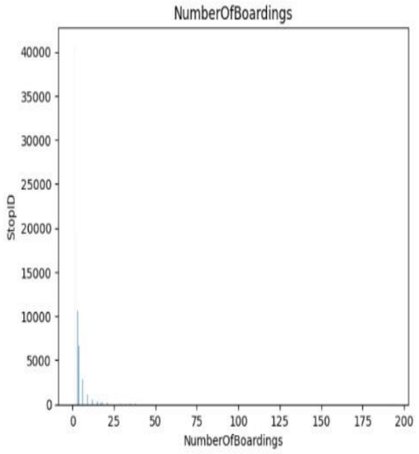
```
 TripID  :  182
 RouteID  :  7
 StopID  :  166
 StopName  :  97
 WeekBeginning  :  55
 NumberOfBoardings  :  145
```

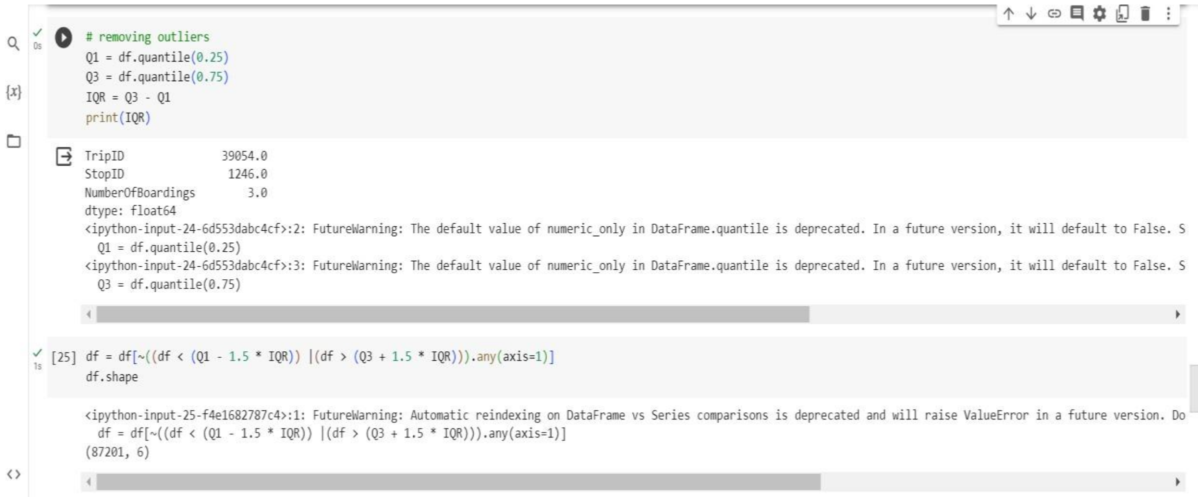


```
for feature in df.columns:
    if feature == "StopName":
        pass
    else:
        bar = sns.histplot(df[feature] , kde_kws = {'bw' : 1} , )
        plt.xlabel(feature)
        plt.ylabel("StopID")
        plt.title(feature)
        plt.show()
```







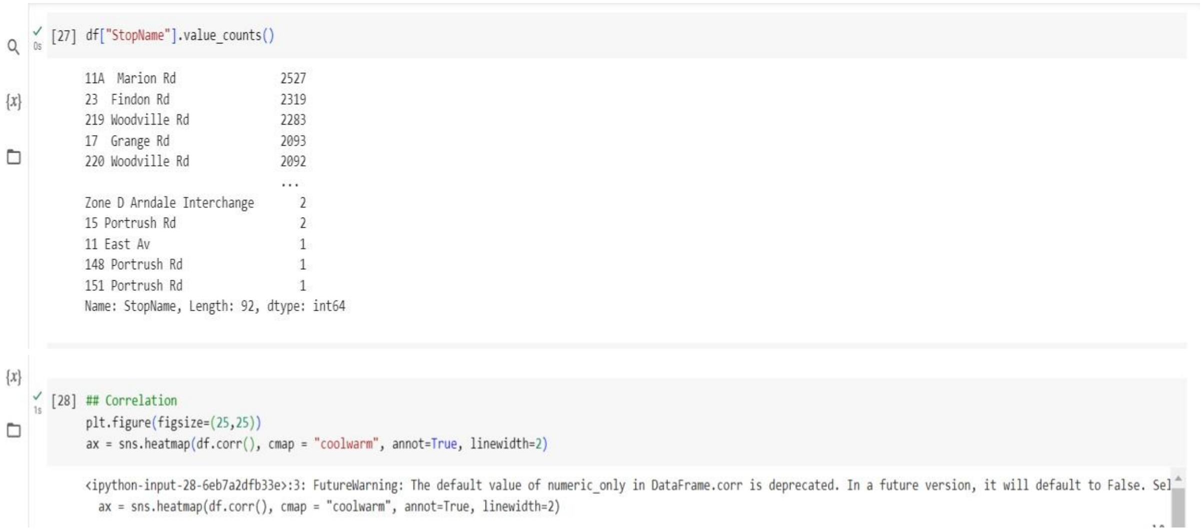


```
# removing outliers
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
TripID               39054.0
StopID                1246.0
NumberOfBoardings        3.0
dtype: float64
<ipython-input-24-6d553dabc4cf>:2: FutureWarning: The default value of numeric_only in DataFrame.quantile is deprecated. In a future version, it will default to False. S
  Q1 = df.quantile(0.25)
<ipython-input-24-6d553dabc4cf>:3: FutureWarning: The default value of numeric_only in DataFrame.quantile is deprecated. In a future version, it will default to False. S
  Q3 = df.quantile(0.75)
```

```
[25] df = df[~((df < (Q1 - 1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
     df.shape
```

```
<ipython-input-25-f4e1682787c4>:1: FutureWarning: Automatic reindexing on DataFrame vs Series comparisons is deprecated and will raise ValueError in a future version. Do
  df = df[~((df < (Q1 - 1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
(87201, 6)
```

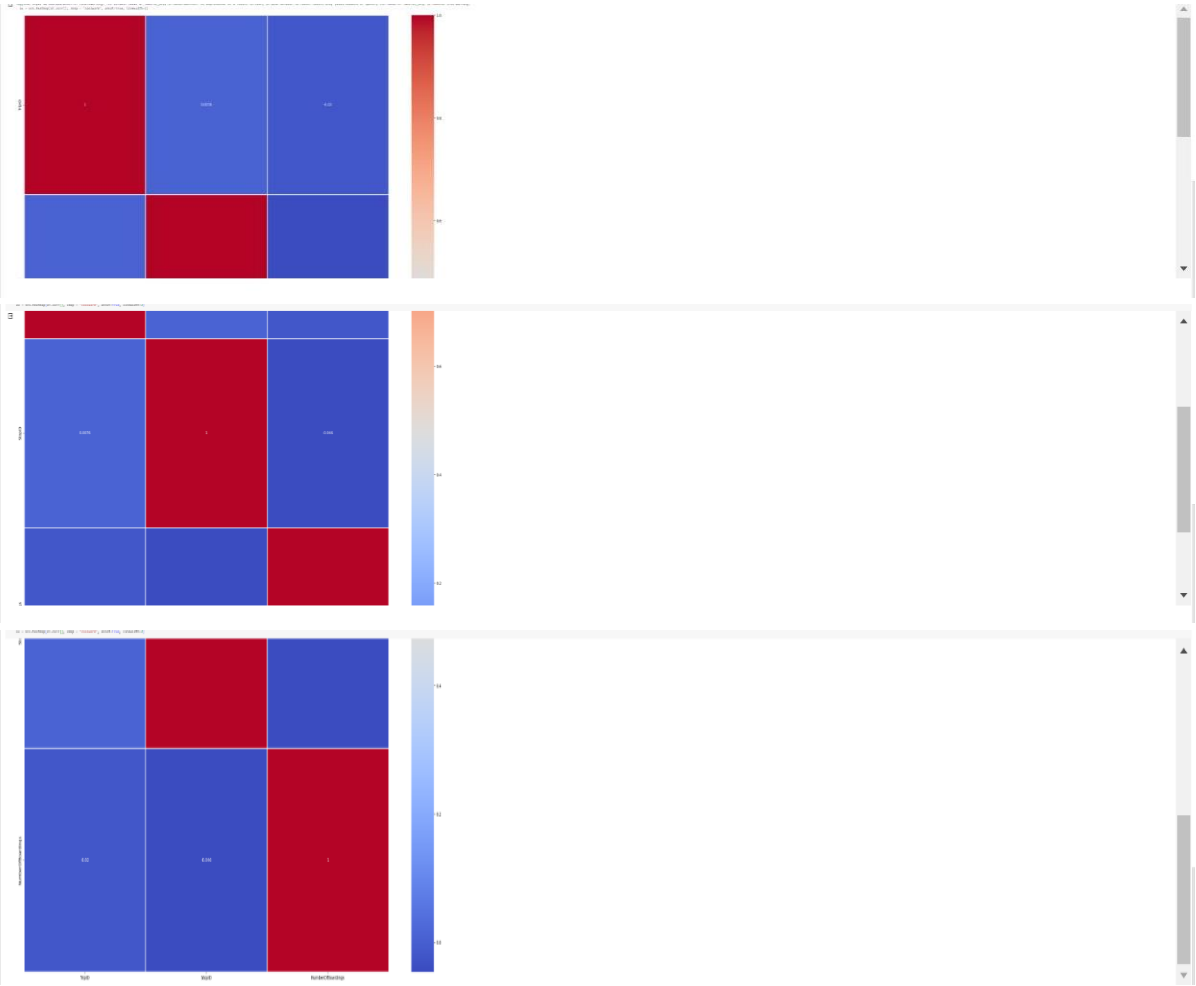


```
[27] df["StopName"].value_counts()
```

```
11A  Marion Rd               2527
23  Findon Rd                2319
219 Woodville Rd             2283
17  Grange Rd                2093
220 Woodville Rd             2092
                             ...
Zone D Arndale Interchange      2
15 Portrush Rd                  2
11 East Av                      1
148 Portrush Rd                 1
151 Portrush Rd                 1
Name: StopName, Length: 92, dtype: int64
```

```
[28] ## Correlation
     plt.figure(figsize=(25,25))
     ax = sns.heatmap(df.corr(), cmap = "coolwarm", annot=True, linewidth=2)
```

```
<ipython-input-28-6eb7a2dfb33e>:3: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Sel
  ax = sns.heatmap(df.corr(), cmap = "coolwarm", annot=True, linewidth=2)
```



THANK YOU