PUBLIC TRANSPORTATION EFFICIENCY ANALYSIS

TEAM LEADER: RASIKA S (310121104084)

TEAM MEMBERS: SARUMATHI S (310121104091)

PRIYADHARSHINI R (310121104077)

SEBASTINRAJAN A (310121104093)

MONISHA M (310121104064)

INTRODUCTION:

Public transportation plays a vital role in modern urban life, offering an eco-friendly and efficient means of commuting. However, the challenges faced by public transportation systems are multifaceted, ranging from overcrowded buses and trains to delays and suboptimal routes. In response to these challenges, this project aims to delve into the heart of public transportation efficiency, using data-driven analysis to identify areas for improvement. By focusing on enhancing the reliability, accessibility, and overall performance of public transportation, we seek to not only improve the daily lives of commuters but also contribute to a more sustainable and connected urban environment.

This project undertakes a systematic investigation of public transportation systems, encompassing various factors, from punctuality and frequency to route optimization. Through rigorous data analysis and modeling, we aspire to provide valuable insights and recommendations to stakeholders and decision-makers within the public transportation sector. As we navigate through this project, we will explore the hurdles, innovations, and potential solutions that influence the efficiency and effectiveness of public transportation, ultimately working toward a more accessible and user-friendly transportation network for everyone.

OBJECTIVES:

The primary objectives of our "Public Transportation Efficiency Analysis" project are as follows:

Identify Efficiency Gaps: Conduct a thorough analysis of existing public transportation systems to pinpoint areas where efficiency gaps exist. These gaps may relate to overcrowding, delays, suboptimal routes, or other factors that hinder the effectiveness of the service.

Data-Driven Insights: Utilize data-driven insights and analysis techniques to gain a comprehensive understanding of the challenges faced by public transportation systems. This includes examining historical ridership data, traffic patterns, and other relevant data sources.

Recommendations for Improvement: Based on the findings, develop actionable recommendations and solutions to enhance the efficiency of public transportation. These recommendations may encompass route optimization, scheduling improvements, and strategies to reduce overcrowding.

Enhanced User Experience: Strive to improve the overall user experience for commuters by addressing common pain points such as punctuality, accessibility, and the availability of real-time information.

Sustainability: Promote the sustainability of public transportation as a green alternative to private car ownership by making it a more efficient and attractive choice for urban commuters.

APPROACH:

Data Collection: Gather relevant data sources that include historical ridership data, traffic patterns, schedules, and geographic information. This data will serve as the foundation for our analysis.

Data Preprocessing: Clean, transform, and preprocess the collected data to ensure its quality and consistency. This step is critical for accurate analysis.

Data Analysis: Utilize various data analysis techniques, such as descriptive statistics, data visualization, and machine learning algorithms, to extract meaningful insights from the data.

Route Optimization: Employ optimization algorithms to assess and potentially redesign public transportation routes for greater efficiency and alignment with commuter needs.

Scheduling Improvements: Analyze historical schedules and propose scheduling enhancements to minimize delays and improve punctuality.

User Experience Enhancement: Consider strategies to enhance the overall user experience, including real-time information dissemination, accessibility improvements, and technology integration.

Evaluation Metrics: Develop and apply key performance indicators (KPIs) and evaluation metrics to measure the impact of proposed changes and identify areas of improvement.

Stakeholder Engagement: Engage with public transportation authorities, urban planners, and other stakeholders to ensure that our analysis aligns with the practical needs of the transportation system.

Recommendations: Summarize the insights and findings and present actionable recommendations for improving public transportation efficiency.

DATA COLLECTION:

Data collection is a foundational step in our project, as it forms the basis for our public transportation efficiency analysis. We obtained data from various sources relevant to public transportation, including:

Ridership Data: Collected historical ridership data, including passenger counts, travel patterns, and ticketing information.

Traffic Patterns: Gathered data on traffic patterns, road congestion, and other external factors that influence public transportation efficiency.

Schedules and Timetables: Acquired schedules and timetables of public transportation services, including bus and train routes, stop locations, and departure times.

Geographic Information: Utilized geographic information systems (GIS) data to map transportation routes, infrastructure, and service coverage.

Real-time Data: Incorporated real-time data sources to monitor and analyze current transportation conditions and delays.

DATA LOADING:

Data loading is the process of importing and integrating the collected datasets into the project environment. This step involves the following key activities:

Data Source Integration: Gather data from various sources, which may include databases, spreadsheets, APIs, and other data repositories.

Data Format Conversion: Ensure that data is in a format compatible with the analysis tools and software used in the project.

Data Validation: Verify the integrity and consistency of the loaded data to identify and rectify any issues, such as missing values or inconsistencies.

Data Storage: Store the data in a secure and organized manner, making it readily accessible for analysis and modeling.

DATA CLEANING:

Data cleaning is a crucial process in preparing your collected data for analysis. It involves the following key activities:

Handling Missing Data: Identify and address missing values in the dataset, which can impact the accuracy of your analysis. This may involve imputation or data removal, depending on the nature of the missing data.

Outlier Detection: Identify and handle outliers—data points significantly different from the majority of the data. Outliers can skew analysis results and should be addressed appropriately.

Data Transformation: Perform transformations on the data as needed. This could involve standardizing units, scaling, or encoding categorical variables for analysis.

Data Consistency: Ensure data consistency by resolving inconsistencies in data formats, naming conventions, and other discrepancies.

Data Quality Assurance: Implement measures to maintain data quality, which may include data validation checks, removing duplicates, and addressing data entry errors.

Data Privacy and Security: Implement measures to protect sensitive data, ensuring compliance with privacy and security standards.

DATA QUALITY:

Data quality refers to the reliability, accuracy, completeness, and consistency of the data used in your analysis. Ensuring high data quality is critical because it directly impacts the validity and credibility of your findings. Key aspects of data quality include:

Accuracy: Data should be free from errors, inaccuracies, or inconsistencies. Accurate data leads to reliable analysis results.

Completeness: Ensure that your dataset contains all the necessary information required for your analysis. Missing data can introduce bias and affect results.

Consistency: Data should follow a consistent format and structure. Inconsistent data can lead to misinterpretations and challenges in analysis.

Relevance: The data should be relevant to your analysis objectives. Irrelevant data can introduce noise and complexity.

Timeliness: Data should be up-to-date to reflect the current state of the subject matter. Outdated data may not be representative of the present situation.

Validity: Data should accurately represent the concepts it intends to measure. Valid data is essential for meaningful analysis and conclusions.

Data Privacy and Security: Ensure that sensitive data is appropriately protected to comply with privacy regulations and maintain security standards.

DATA VISUALIZATION:

Data visualization is the process of representing data in a visual format, such as charts, graphs, maps, and dashboards. It is a critical component of your public transportation efficiency analysis as it allows for the clear and intuitive presentation of insights and findings. Key aspects of data visualization include:

Graphical Representation: Use graphs, charts, and maps to visually represent data, making complex information more accessible and understandable.

Exploratory Data Analysis (EDA): Data visualization is a crucial part of EDA, helping you to identify patterns, trends, and anomalies in the data. Scatter plots, histograms, and box plots are common tools for EDA.

Time Series Analysis: Visualize time series data to understand how public transportation parameters change over time. Line charts and heatmaps can be useful for this purpose.

Geospatial Analysis: Utilize maps and geospatial visualizations to assess transportation routes, coverage, and congestion. Geographic Information System (GIS) tools can be helpful.

Interactive Dashboards: Create interactive dashboards that allow users to explore and interact with the data, enabling real-time insights and scenario analysis.

Visual Storytelling: Use data visualization to tell a compelling story about public transportation efficiency. Visualization helps convey the impact of your findings to stakeholders and the general public.

Insight Communication: Visualization helps in the effective communication of complex data-driven insights to both technical and non-technical audiences.

MACHINE LEARNING ALGORITHM:

Machine learning algorithms are a subset of complex analysis models that play a significant role in your public transportation efficiency analysis. These algorithms are designed to enable computers to learn from data and make predictions or decisions based on patterns and trends within the data. Key aspects of machine learning algorithms in your project include:

Supervised Learning: Utilize supervised learning algorithms when you have labeled data, such as historical ridership information. These algorithms learn to make predictions by associating inputs (features) with known outputs (labels).

Unsupervised Learning: Apply unsupervised learning when you have unlabeled data or want to discover hidden patterns within the data. Clustering algorithms, such as k-means, can help identify groups of similar data points.

Regression Analysis: Use regression algorithms for predicting continuous numerical values, which could be useful for forecasting passenger demand or travel times.

Classification Analysis: Employ classification algorithms when you need to categorize data into discrete classes, such as predicting delays or service interruptions.

Feature Engineering: Feature selection and engineering are crucial in machine learning. Identify relevant features (variables) and transform them to improve the model's performance.

Model Training: Train machine learning models on historical data to learn from patterns and relationships within the data. Common algorithms include decision trees, random forests, support vector machines, and neural networks.

Evaluation Metrics: Assess the performance of machine learning models using evaluation metrics like accuracy, precision, recall, and F1-score, depending on the specific problem you're addressing.

DATA PREPROCESSING:

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.

By presenting this section, we provide a comprehensive view of the procedures undertaken to refine our data, positioning us for more accurate and meaningful analysis in the subsequent phases of the project. Properly preprocessed data is the key to uncovering patterns and trends that can help us optimize public transportation efficiency

CODE:

In [1]:
```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import datetime
```

In [2]:
```
df=pd.read_csv("D:\saru.csv")
```

```
C:\Users\dhars\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3444: Dty
peWarning: Columns (1) have mixed types.Specify dtype option on import or set low_mem
ory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

In [3]:
```
out_geo = pd.read_csv("D:\output_geo.csv")
```

In [4]:
```
df.shape
```

Out[4]:
```
(1048575, 6)
```

	accuracy	formatted_address	google_place_id	input_string	latitude	l
1	ROOFTOP	177 Cross Rd, Westbourne Park SA 5041, Australia	ChIJ-VFZ87bPsGoRyfVgC5qbPpE	177 Cross Rd	-34.966607	1:
2	ROOFTOP	175 Cross Rd, Westbourne Park SA 5041, Australia	ChIJIztlirbPsGoR38KRk76kPFI	175 Cross Rd	-34.966758	1:
3	GEOMETRIC_CENTER	Zone A Arndale Interchange - South side, Kilke...	ChIJn0C1hCPGsGoRIWvCdhF1RIg	Zone A Arndale Interchange	-34.875160	1:
4	ROOFTOP	178 Cross Rd, Malvern SA 5061, Australia	ChIJycNiylvOsGoRdhfq9GKnpq0	178 Cross Rd	-34.964960	1:

In [8]:
```
from math import sin, cos, sqrt, atan2, radians
def calc_dist(lat1,lon1):
    ## approximate radius of earth in km
    R = 6373.0
    dlon = radians(138.604801) - radians(lon1)
    dlat = radians(-34.921247) - radians(lat1)
    a = sin(dlat / 2)**2 + cos(radians(lat1)) * cos(radians(-34.921247)) * sin(dlon
    c = 2 * atan2(sqrt(a), sqrt(1 - a))
    return R * c
```

In [9]:
```
out_geo['dist_from_centre'] = out_geo[['latitude','longitude']].apply(lambda x: calc
```

In [10]:

```
out_geo.head()
```

Out[10]:

	accuracy	formatted_address	google_place_id	input_string	latitude	l
0	ROOFTOP	181 Cross Rd, Westbourne Park SA 5041, Australia	ChIJKT7I9rbPsGoRVHMHkly-Oyk	181 Cross Rd	-34.966656	1:
1	ROOFTOP	177 Cross Rd, Westbourne Park SA 5041, Australia	ChIJ-VFZ87bPsGoRyfVgC5qbPpE	177 Cross Rd	-34.966607	1:
2	ROOFTOP	175 Cross Rd, Westbourne Park SA 5041, Australia	ChIJIztlirbPsGoR38KRk76kPFI	175 Cross Rd	-34.966758	1:
3	GEOMETRIC_CENTER	Zone A Arndale Interchange - South side, Kilke...	ChIJn0C1hCPGsGoRIWvCdhF1RIg	Zone A Arndale Interchange	-34.875160	1:
4	ROOFTOP	178 Cross Rd, Malvern SA 5061, Australia	ChIJycNiylvOsGoRdhfq9GKnpq0	178 Cross Rd	-34.964960	1:

In [11]:

```
out_geo['type'].fillna('street_address',inplace=True)
out_geo['type'] = out_geo['type'].apply(lambda x: str(x).split(',')[-1])
```

In [12]:

```
out_geo['type'].unique()
```

Out[12]:

```
array(['street_address', 'transit_station', 'premise', 'political',
       'school', 'route', 'intersection', 'point_of_interest',
       'subpremise', 'real_estate_agency', 'university', 'travel_agency',
       'restaurant', 'supermarket', 'store', 'post_office'], dtype=object)
```

In [13]:

```
df['WeekBeginning'] = pd.to_datetime(df['WeekBeginning']).dt.date
df['WeekBeginning'][1]
```

Out[13]:

```
datetime.date(2013, 6, 30)
```

In [14]:

```
df= pd.merge(df,out_geo,how='left',left_on = 'StopName',right_on = 'input_string')
df.head(5)
```

Out[14]:

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings	accuracy	f
0	23631	100	14156	181 Cross Rd	2013-06-30	1	ROOFTOP	
1	23631	100	14144	177 Cross Rd	2013-06-30	1	ROOFTOP	
2	23632	100	14132	175 Cross Rd	2013-06-30	1	ROOFTOP	
3	23633	100	12266	Zone A Arndale Interchange	2013-06-30	2	GEOMETRIC_CENTER	I
4	23633	100	14147	178 Cross Rd	2013-06-30	1	ROOFTOP	

In [15]:
```
df.shape
```
Out[15]:
```
(1048575, 17)
```
In [16]:
```
col = ['TripID', 'RouteID', 'StopID', 'StopName', 'WeekBeginning','NumberOfBoardings
      'latitude', 'longitude','postcode','type','dist_from_centre']
df= df[col]
```
In [17]:
```
grouped = df.groupby(['StopName','WeekBeginning','type'])
```
In [18]:
```
grouped = df.groupby(['StopName','WeekBeginning','type']).agg({'NumberOfBoardings':
grouped.columns = ["_".join(x) for x in grouped.columns.ravel()]
```
```
C:\Users\dhars\AppData\Local\Temp/ipykernel_12660/3162596695.py:2: FutureWarning: Ind
ex.ravel returning ndarray is deprecated; in a future version this will return a view
on self.
  grouped.columns = ["_".join(x) for x in grouped.columns.ravel()]
```
In [19]:
```
grouped.head(10)
```
Out[19]:

StopName	WeekBeginning	type	NumberOfBoardings_sum	NumberOfBoardings_count	Nu
1 Anzac Hwy	2013-01-09	street_address	89	42	
	2013-01-12	street_address	81	41	
	2013-03-11	street_address	50	30	
	2013-04-08	street_address	74	33	
	2013-06-10	street_address	47	22	
	2013-06-30	street_address	131	41	
	2013-07-07	street_address	118	34	
	2013-07-14	street_address	91	29	
	2013-07-21	street_address	71	33	
	2013-07-28	street_address	88	38	

In [20]:
```
grouped.columns
```
Out[20]:
```
Index(['NumberOfBoardings_sum', 'NumberOfBoardings_count',
       'NumberOfBoardings_max'],
      dtype='object')
```
In [21]:
```
st_week_grp = pd.DataFrame(grouped).reset_index()
st_week_grp.shape
```
Out[21]:
```
(28084, 6)
```

In [22]:

```
st_week_grp.head()
```

Out[22]:

	StopName	WeekBeginning	type	NumberOfBoardings_sum	NumberOfBoardings_count
0	1 Anzac Hwy	2013-01-09	street_address	89	42
1	1 Anzac Hwy	2013-01-12	street_address	81	41
2	1 Anzac Hwy	2013-03-11	street_address	50	30
3	1 Anzac Hwy	2013-04-08	street_address	74	33
4	1 Anzac Hwy	2013-06-10	street_address	47	22

In [23]:

```
st_week_grp1 = pd.DataFrame(st_week_grp.groupby('StopName')["WeekBeginning"].count()
```

In [24]:

```
st_week_grp1.head()
```

Out[24]:

	StopName	WeekBeginning
0	1 Anzac Hwy	54
1	1 Fullarton Rd	54
2	1 George St	53
3	1 Glen Osmond Rd	14
4	1 Henley Beach Rd	54

In [25]:

```
aa = list(st_week_grp1[st_week_grp1['WeekBeginning'] == 54]['StopName'])
aa[1:10]
```

Out[25]:

```
['1 Fullarton Rd',
 '1 Henley Beach Rd',
 '1 Kensington Rd',
 '1 Port Rd',
 '10  Holbrooks Rd',
 '10  Marion Rd',
 '10 Greenhill Rd',
 '10 Harvey Av',
 '10 Kensington Rd']
```

In [26]:
```
bb = st_week_grp[st_week_grp['StopName'].isin(aa)]
bb.head()
```

Out[26]:

	StopName	WeekBeginning	type	NumberOfBoardings_sum	NumberOfBoardings_count	N
0	1 Anzac Hwy	2013-01-09	street_address	89	42	
1	1 Anzac Hwy	2013-01-12	street_address	81	41	
2	1 Anzac Hwy	2013-03-11	street_address	50	30	
3	1 Anzac Hwy	2013-04-08	street_address	74	33	
4	1 Anzac Hwy	2013-06-10	street_address	47	22	

In [27]:
```
bb.shape
```

Out[27]:
```
(23166, 6)
```

In [28]:
```
type(bb)
```

Out[28]:
```
pandas.core.frame.DataFrame
```

In [29]:
```
new_df = df[df['StopName'].isin(aa)]
new_df.shape
print("data without stopage removing: ", df.shape)
print("data, after removing stoppage not having the data of whole 54 weeks: ", new_d
```

```
data without stopage removing:  (1048575, 11)
data, after removing stoppage not having the data of whole 54 weeks:  (1004048, 11)
```

In [30]:
```
new_df.head(2)
filtered_data = new_df[new_df['dist_from_centre'] <= 100]
filtered_data.shape
```

Out[30]:
```
(987243, 11)
```

In [31]:
```
df = filtered_data.copy()
df.shape
```

Out[31]:
```
(987243, 11)
```

In [32]:
```
stopageName_with_boarding = bb.groupby(['StopName']).agg({'NumberOfBoardings_sum': [
```

In [33]:
```
stopageName_with_boarding = pd.DataFrame(stopageName_with_boarding.reset_index())
```

In [34]:
```
stopageName_with_boarding.columns = ["StopName", "Total_boarding_on_the_stopage"]
```

In [35]:
```
stopageName_with_boarding.head()
```

In [35]:

```
stopageName_with_boarding.head()
```

Out[35]:

	StopName	Total_boarding_on_the_stopage
0	1 Anzac Hwy	3631
1	1 Fullarton Rd	585
2	1 Henley Beach Rd	1157
3	1 Kensington Rd	11346
4	1 Port Rd	3098

In [36]:

```
df.nunique()
```

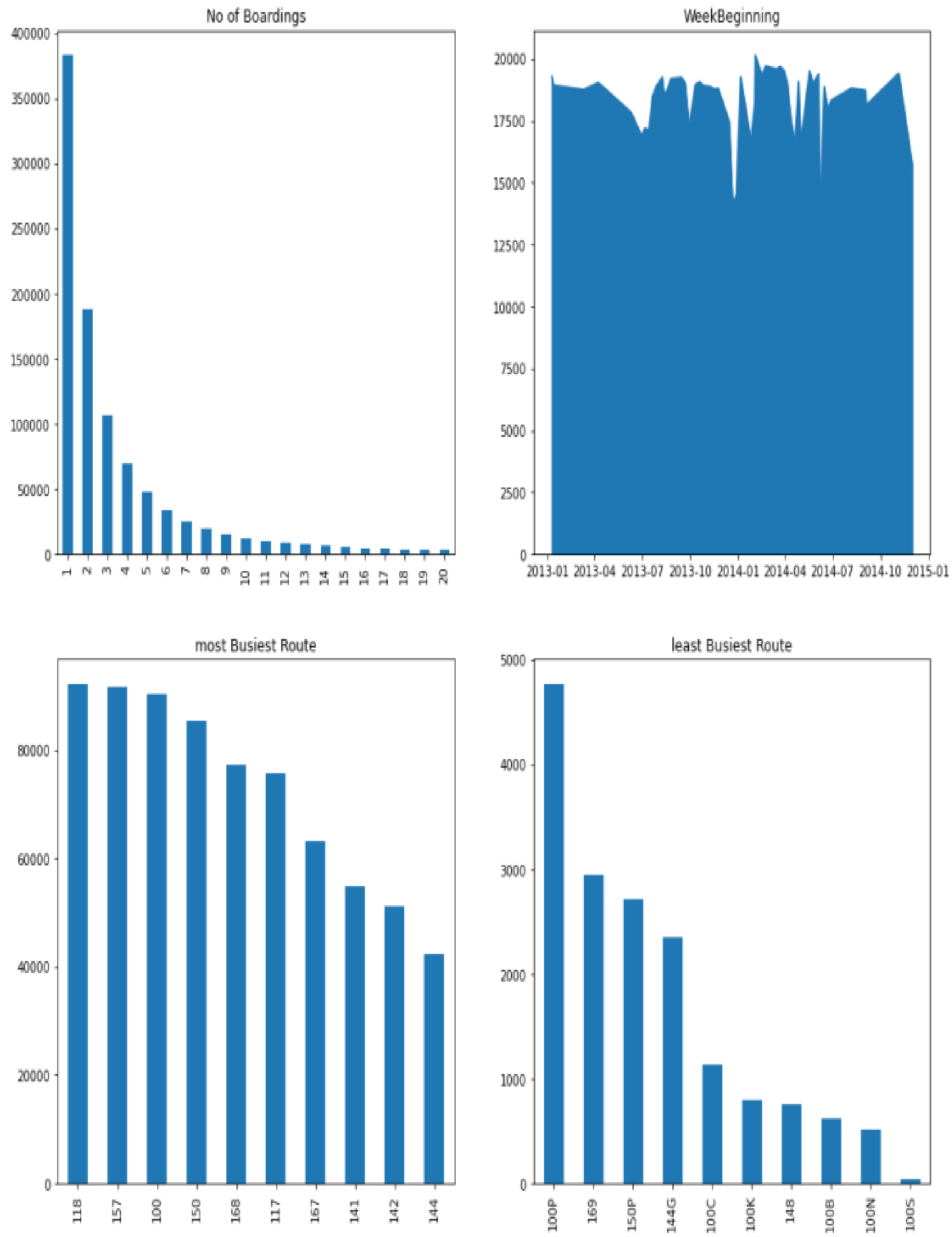
Out[36]:

```
TripID              3294
RouteID               36
StopID               738
StopName             421
WeekBeginning         54
NumberOfBoardings    156
latitude             324
longitude            324
postcode              54
type                   7
dist_from_centre     325
dtype: int64
```

In [37]:

```
fig,axrr=plt.subplots(2,2,figsize=(15,15))

ax=axrr[0][0]
ax.set_title("No of Boardings")
df['NumberOfBoardings'].value_counts().sort_index().head(20).plot.bar(ax=axrr[0][0])

ax=axrr[0][1]
ax.set_title("WeekBeginning")
df['WeekBeginning'].value_counts().plot.area(ax=axrr[0][1])

ax=axrr[1][0]
ax.set_title("most Busiest Route")
df['RouteID'].value_counts().head(10).plot.bar(ax=axrr[1][0])

ax=axrr[1][1]
ax.set_title("least Busiest Route")
df['RouteID'].value_counts().tail(10).plot.bar(ax=axrr[1][1])
```

Out[37]: <AxesSubplot:title={'center':'least Busiest Route'}>



In [38]:

```
stopageName_with_boarding = stopageName_with_boarding.sort_values('Total_boarding_on
```

In [39]:

```
stopageName_with_boarding.head(10)
```

Out[39]:

	StopName	Total_boarding_on_the_stopage
410	W1 North Tce	122028
387	I1 North Tce	116468
409	V2 Currie St	95443
376	E2 Currie St	89985
407	V1 Currie St	81356
374	D North Tce	78903
381	G2 North Tce	77547
396	R2 North Tce	74813
382	G3 Grenfell St	72887
406	U1 North Tce	71947

In [40]:

```
stopageName_with_boarding.tail(10)
```

Out[40]:

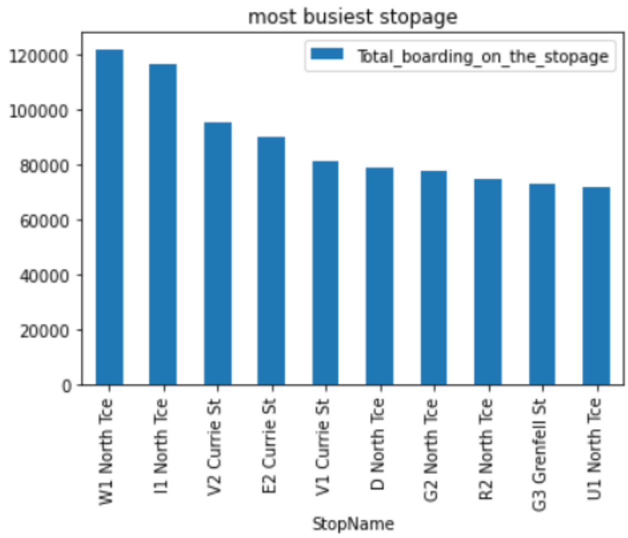
	StopName	Total_boarding_on_the_stopage
408	V1 Hutt St	557
294	5 The Parade	526
249	36C Sansom Rd	492
267	42D Semaphore Rd	481
297	53 Victoria Rd	454
201	3 Rundle St	437
171	24 Marion Rd	428
38	12A Streeters Rd	376
202	3 Unley Rd	361
115	19 Gilles Rd	215

In [41]:

```
ax = stopageName_with_boarding.head(10).plot.bar(x='StopName', y='Total_boarding_on_
ax.set_title("most busiest stopage")
```

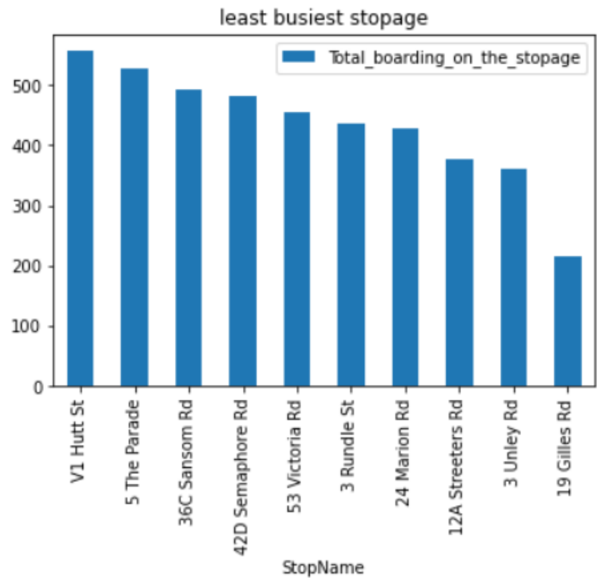
Out[41]:

```
Text(0.5, 1.0, 'most busiest stopage')
```



In [42]:
```
ax = stopageName_with_boarding.tail(10).plot.bar(x='StopName', y='Total_boarding_on_
ax.set_title("least busiest stopage")
```

Out[42]:
```
Text(0.5, 1.0, 'least busiest stopage')
```



In [43]:
```
df['WeekBeginning'].value_counts().mean()
```

Out[43]:
```
18282.277777777777
```

In [44]:
```
bb_grp = df.groupby(['dist_from_centre']).agg({'NumberOfBoardings': ['sum']}).reset_
bb_grp.columns = bb_grp.columns.get_level_values(0)
bb_grp.head()
```

Out[44]:

	dist_from_centre	NumberOfBoardings
0	0.000018	499639

	dist_from_centre	NumberOfBoardings
1	0.131368	17560
2	0.309089	72887
3	0.314937	41409
4	0.343642	20879

In [45]:

```
bb_grp.columns
```

Out[45]:

```
Index(['dist_from_centre', 'NumberOfBoardings'], dtype='object')
```

In [46]:

```
bb_grp.tail()
```

Out[46]:

	dist_from_centre	NumberOfBoardings
320	39.908350	10551
321	39.946258	4655
322	39.950945	4712
323	45.706398	9041
324	99.665190	4080

In [47]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 987243 entries, 0 to 1048566
Data columns (total 11 columns):
 #   Column             Non-Null Count   Dtype  
---  ------             --------------   -----  
 0   TripID             987243 non-null  int64  
 1   RouteID            987243 non-null  object 
 2   StopID             987243 non-null  int64  
 3   StopName           987243 non-null  object 
 4   WeekBeginning      987243 non-null  object 
 5   NumberOfBoardings  987243 non-null  int64  
 6   latitude           987243 non-null  float64
 7   longitude          987243 non-null  float64
 8   postcode           920887 non-null  object 
 9   type               987243 non-null  object 
 10  dist_from_centre   987243 non-null  float64
dtypes: float64(3), int64(3), object(5)
memory usage: 90.4+ MB
```

In [48]:

```
import seaborn as sns
```

In [49]:

```
df["StopID"].value_counts()
```

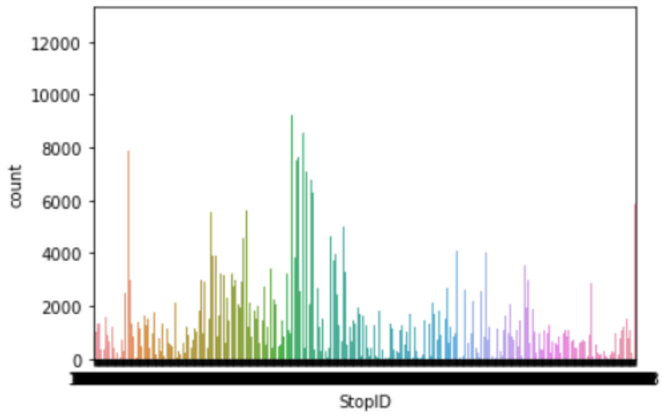
Out[49]:

```
13278    12678
13279     9221
13308     8557
13364     8313
13280     8096
         ...
```

```
13587        3
13562        2
13599        1
13688        1
13746        1
Name: StopID, Length: 738, dtype: int64
```

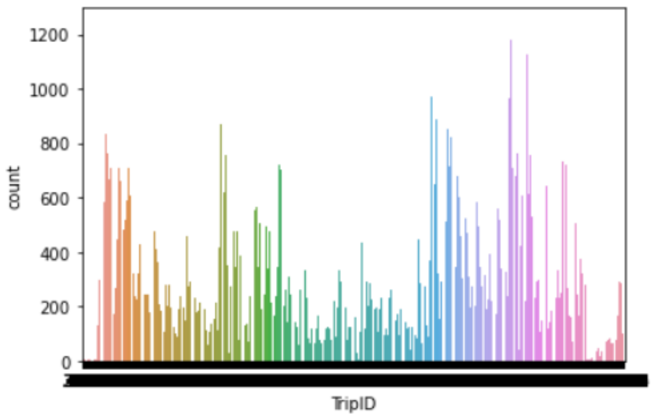
In [50]:
```
sns.countplot(x='StopID',data=df)
```

Out[50]:
```
<AxesSubplot:xlabel='StopID', ylabel='count'>
```



In [51]:
```
sns.countplot(x='TripID',data=df)
```

Out[51]:
```
<AxesSubplot:xlabel='TripID', ylabel='count'>
```



In [52]:
```
df.isnull().sum()
```

Out[52]:
```
TripID                  0
RouteID                 0
StopID                  0
StopName                0
WeekBeginning           0
NumberOfBoardings       0
latitude                0
longitude               0
postcode            66356
type                    0
```

```
dist_from_centre        0
dtype: int64
```

In [53]:

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

```
postcode : 6.72%
```

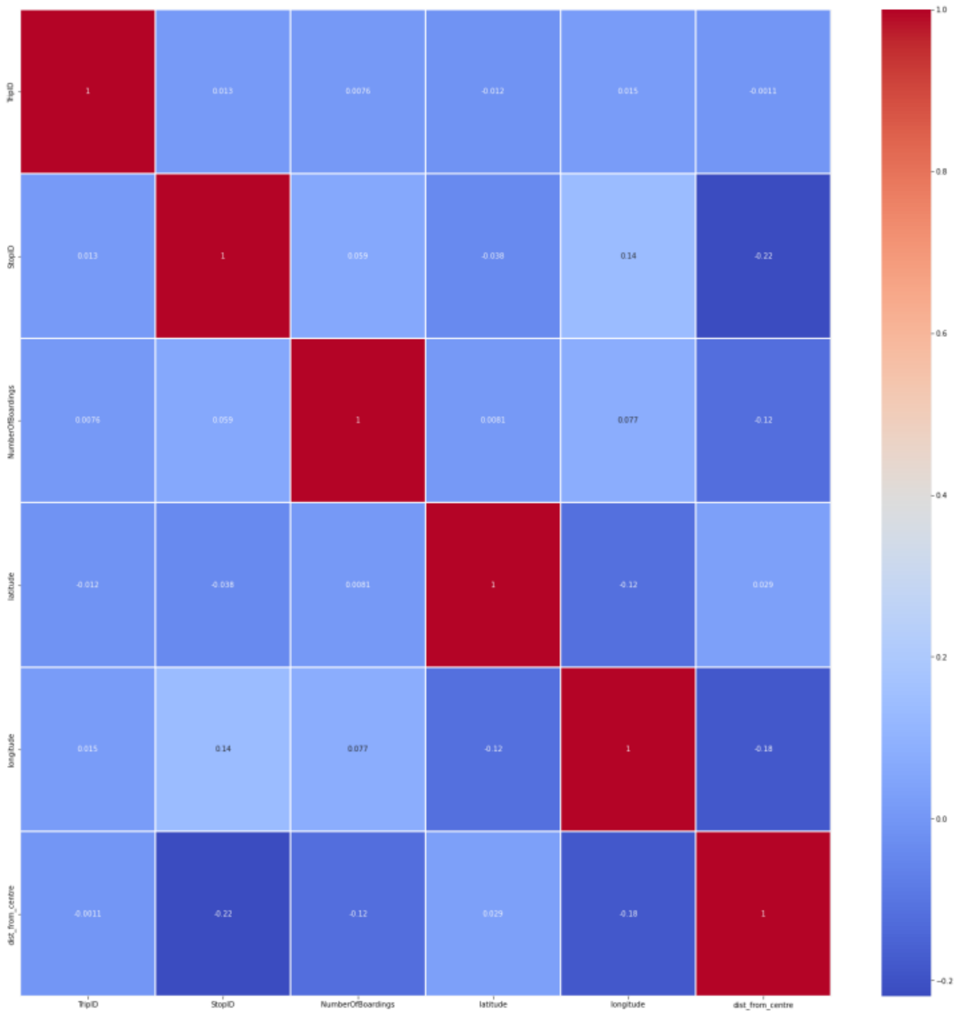
In [54]:

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

```
 TripID  :  3294
 RouteID  :  36
 StopID  :  738
 StopName  :  421
 WeekBeginning  :  54
 NumberOfBoardings  :  156
 latitude  :  324
 longitude  :  324
 postcode  :  55
 type  :  7
 dist_from_centre  :  325
```

In [55]:

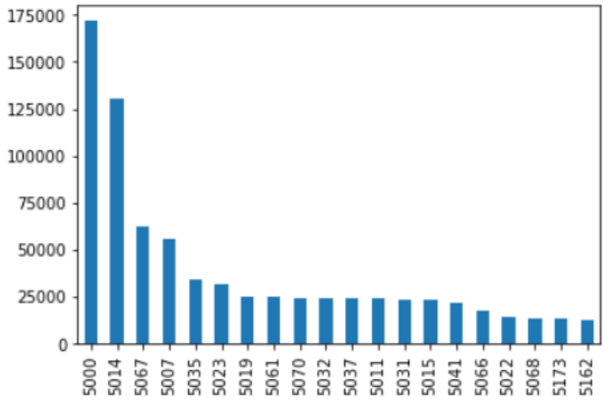
```python
plt.figure(figsize=(25,25))
ax = sns.heatmap(df.corr(), cmap = "coolwarm", annot=True, linewidth=2)
```



In [64]:

```
df['postcode'].value_counts().head(20).plot.bar()
```

Out[64]: <AxesSubplot:>



In [67]:

```
bb_grp = df.groupby(['dist_from_centre']).agg({'NumberOfBoardings': ['sum']}).reset_
bb_grp.columns = bb_grp.columns.get_level_values(0)
bb_grp.head()
```

Out[67]:

	dist_from_centre	NumberOfBoardings
0	0.000018	499639
1	0.131368	17560
2	0.309089	72887
3	0.314937	41409
4	0.343642	20879

In [68]:

```
bb_grp.columns
```

Out[68]: Index(['dist_from_centre', 'NumberOfBoardings'], dtype='object')

In [72]:

```
bb_grp = bb.groupby(['StopName']).agg({'NumberOfBoardings_sum': ['sum']}).reset_inde
```

In [78]:

```
bb_grp[1000:1005]
bb.groupby(['StopName']).agg({'NumberOfBoardings_sum': ['sum']}).reset_index().iloc[
```

Out[78]:

	StopName	NumberOfBoardings_sum
		sum
135	20 Pine Av	3650
28	12 Portrush Rd	17967
54	14 Devereux Rd	4542
103	18 Devereux Rd	8863
145	21 Pine Av	4272

In [81]:
```
bb1=bb.copy()
```

In [83]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn import metrics
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score
from sklearn.metrics import accuracy_score,confusion_matrix
```

In [84]:
```
d=[]
for i in bb['StopName'].unique():
    d.append({'StopName': i,'Boarding_sum':np.sum(bb[bb['StopName'] == i]['NumberOfB
             'Boarding_count':np.sum(bb[bb['StopName'] == i]['NumberOfBoardings_coun
             'Boarding_max':np.sum(bb[bb['StopName'] == i]['NumberOfBoardings_max'].
pct_chng = pd.DataFrame(d)
```

In [87]:
```
pct_chng['Boarding_sum'].nlargest(5)
```

Out[87]:
```
80     3.275757
417    2.430625
84     2.107047
82     1.925259
404    1.830294
Name: Boarding_sum, dtype: float64
```

In [92]:
```
pct_chng['Boarding_sum'].nsmallest(5)
```

Out[92]:
```
74     0.004324
172    0.006087
21     0.009635
424    0.009892
7      0.010404
Name: Boarding_sum, dtype: float64
```

In [89]:
```
pct_chng[pct_chng['Boarding_sum']<0].shape
```

Out[89]:
```
(0, 4)
```

In [91]:
```
pct_chng.iloc[[311,214,114,153,129]]
```

Out[91]:

	StopName	Boarding_sum	Boarding_count	Boarding_max
311	6 Grove Av	0.056369	0.039387	0.125375
214	33A Tapleys Hill Rd	0.020153	0.005316	0.091696
114	19 Portrush Rd	0.232944	0.020618	0.692598
153	21G Gordon St	0.136070	0.026715	0.532690
129	2 Richmond Rd	0.039069	0.008527	0.109963

In [93]:

```
bb1 = pd.merge(bb, out_geo, how='left', left_on = 'StopName', right_on = 'input_stri
```

In [95]:

```
'''Holidays--
2013-09-01,Father's Day
2013-10-07,Labour day
2013-12-25,Christmas day
2013-12-26,Proclamation Day
2014-01-01,New Year
2014-01-27,Australia Day
2014-03-10,March Public Holiday
2014-04-18,Good Friday
2014-04-19,Easter Saturday
2014-04-21,Easter Monday
2014-04-25,Anzac Day
2014-06-09,Queen's Birthday'''
```

Out[95]:

```
"Holidays--\n2013-09-01,Father's Day\n2013-10-07,Labour day\n2013-12-25,Christmas day
\n2013-12-26,Proclamation Day\n2014-01-01,New Year\n2014-01-27,Australia Day\n2014-03
-10,March Public Holiday\n2014-04-18,Good Friday\n2014-04-19,Easter Saturday\n2014-04
-21,Easter Monday\n2014-04-25,Anzac Day\n2014-06-09,Queen's Birthday"
```

In [96]:

```
def holiday_label (row):
    if row == datetime.date(2013, 9, 1) :
          return '1'
    if row == datetime.date(2013, 10, 6) :
          return '1'
    if row == datetime.date(2013, 12, 22) :
          return '2'
    if row == datetime.date(2013, 12, 29):
          return '1'
    if row  == datetime.date(2014, 1, 26):
          return '1'
    if row == datetime.date(2014, 3, 9):
          return '1'
    if row == datetime.date(2014, 4, 13) :
          return '2'
    if row == datetime.date(2014, 4, 20):
          return '2'
    if row == datetime.date(2014, 6, 8):
          return '1'
    return '0'
```

In [97]:

```
df['WeekBeginning'] = pd.to_datetime(df['WeekBeginning']).dt.date
```

In [98]:

```
df['holiday_label'] = df['WeekBeginning'].apply (lambda row: holiday_label(row))
```

In [99]:

```
df= pd.merge(df,out_geo,how='left',left_on = 'StopName',right_on = 'input_string')
```

In [100…

```
df
```

Out[100…

	TripID	RouteID	StopID	StopName	WeekBeginning	NumberOfBoardings	latitude_x	long
0	23631	100	14156	181 Cross Rd	2013-06-30	1	-34.966656	138
1	23631	100	14144	177 Cross Rd	2013-06-30	1	-34.966607	138
2	23632	100	14132	175 Cross Rd	2013-06-30	1	-34.966758	138
3	23633	100	12266	Zone A Arndale Interchange	2013-06-30	2	-34.875160	138
4	23633	100	14147	178 Cross Rd	2013-06-30	1	-34.964960	138
...	...	...	...	...	...	...	...	
987238	45679	171	13536	Q1 Hutt St	2013-09-29	4	-34.930028	138
987239	45680	171	13391	V1 Hutt St	2013-09-29	1	-34.930028	138
987240	45680	171	13536	Q1 Hutt St	2013-09-29	10	-34.930028	138
987241	45680	171	13594	O3 Hutt Rd	2013-09-29	1	-34.935505	138
987242	45680	171	13484	S1 Hutt St	2013-09-29	6	-34.930028	138

987243 rows × 23 columns

In [102…

```
bb1 = pd.merge(bb, out_geo, how='left', left_on = 'StopName', right_on = 'input_stri
```

In [103…

```
bb1
```

Out[103…

	StopName	WeekBeginning	type_x	NumberOfBoardings_sum	NumberOfBoardings_cou
0	1 Anzac Hwy	2013-01-09	street_address	89	
1	1 Anzac Hwy	2013-01-12	street_address	81	
2	1 Anzac Hwy	2013-03-11	street_address	50	
3	1 Anzac Hwy	2013-04-08	street_address	74	
4	1 Anzac Hwy	2013-06-10	street_address	47	
...	...	...	...	...	
23161	Zone D Port Adelaide Interchan	2014-08-06	transit_station	1248	1
23162	Zone D Port Adelaide Interchan	2014-09-02	transit_station	1233	1
23163	Zone D Port Adelaide Interchan	2014-09-03	transit_station	961	1
23164	Zone D Port Adelaide Interchan	2014-11-05	transit_station	1479	1
23165	Zone D Port Adelaide Interchan	2014-12-01	transit_station	908	1

23166 rows × 17 columns

In [104…

```
bb1['holiday_label'] = bb1['WeekBeginning'].apply (lambda row: holiday_label(row))
```

In [106…

```
cols = ['StopName','WeekBeginning','type_x','NumberOfBoardings_sum','NumberOfBoardin
bb1=bb1[cols]
```

In [107…

```
bb1.shape
```

Out[107…

```
(23166, 11)
```

In [108…

```
bb1.head()
```

Out[108...

	StopName	WeekBeginning	type_x	NumberOfBoardings_sum	NumberOfBoardings_count	N
0	1 Anzac Hwy	2013-01-09	street_address	89	42	
1	1 Anzac Hwy	2013-01-12	street_address	81	41	
2	1 Anzac Hwy	2013-03-11	street_address	50	30	
3	1 Anzac Hwy	2013-04-08	street_address	74	33	
4	1 Anzac Hwy	2013-06-10	street_address	47	22	

In [109...

```
for i in bb1.columns:
    bb1[i].fillna(bb1[i].mode()[0], inplace=True)
bb1[["postcode", "holiday_label"]] = bb1[["postcode", "holiday_label"]].apply(pd.to_
```

In [110...

```
le = LabelEncoder()
bb1['StopName'] = le.fit_transform(bb1['StopName'])
bb1['type_x'] = le.fit_transform(bb1['type_x'])
```

In [111...

```
train = bb1[bb1['WeekBeginning'] < datetime.date(2014, 6, 1)]
test = bb1[bb1['WeekBeginning'] >= datetime.date(2014, 6, 1)]
train.shape
```

Out[111...

```
(18876, 11)
```

In [112...

```
test.shape
```

Out[112...

```
(4290, 11)
```

In [114...

```
le = LabelEncoder()
train['WeekBeginning'] = le.fit_transform(train['WeekBeginning'])
test['WeekBeginning'] = le.fit_transform(test['WeekBeginning'])
```

```
C:\Users\dhars\AppData\Local\Temp/ipykernel_12660/3357953768.py:2: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  train['WeekBeginning'] = le.fit_transform(train['WeekBeginning'])
C:\Users\dhars\AppData\Local\Temp/ipykernel_12660/3357953768.py:3: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/us
er_guide/indexing.html#returning-a-view-versus-a-copy
  test['WeekBeginning'] = le.fit_transform(test['WeekBeginning'])
```

In [115...

```
tr_col = ['StopName', 'WeekBeginning', 'type_x', 'latitude',
       'longitude', 'postcode', 'dist_from_centre', 'holiday_label']
train_sum_y = train[['StopName','NumberOfBoardings_sum']]
train_count_y = train[['StopName','NumberOfBoardings_count']]
train_max_y = train[['StopName','NumberOfBoardings_max']]
train_x = train[tr_col]
test_x = test[tr_col]

test_sum_y = test[['StopName','NumberOfBoardings_sum']]
test_count_y = test[['StopName','NumberOfBoardings_count']]
test_max_y = test[['StopName','NumberOfBoardings_max']]
```

In [117…
```python
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators=700, min_samples_leaf=3, max_features=0.5
model.fit(train_x.values,train_sum_y['NumberOfBoardings_sum'].values)
preds = model.predict(test_x.values)
```

In [118…
```python
preds
```

Out[118…
```
array([  75.47143217,   75.47143217,   75.14697469, ..., 1135.70426014,
       1152.81469804, 1162.29256653])
```

In [119…
```python
model
```

Out[119…
```
RandomForestRegressor(max_features=0.5, min_samples_leaf=3, n_estimators=700,
                      n_jobs=-1)
```

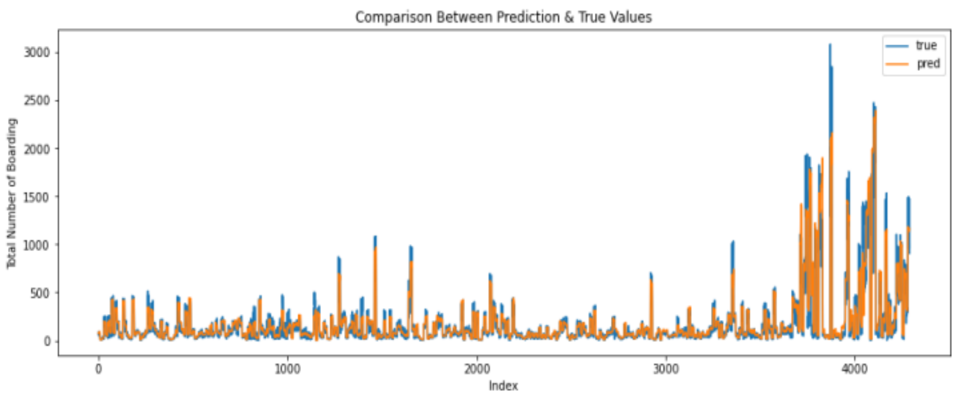
In [120…
```python
rms = sqrt(mean_squared_error(test_sum_y['NumberOfBoardings_sum'].values, preds))
rms
```

Out[120…
```
100.3075140622033
```

In [121…
```python
test_sum_y.values[:15]
preds[:15]
```

Out[121…
```
array([75.47143217, 75.47143217, 75.14697469, 75.61671958, 76.79906188,
       89.50985106, 92.11640315, 91.56273753, 84.2618574 , 81.36238239,
        7.36146436,  7.40676425,  7.35613134,  6.82066622,  6.86647858])
```

In [123…
```python
plt.figure(figsize=(15,5))
plt.plot(test_sum_y['NumberOfBoardings_sum'].values, label='true')
plt.plot(preds, label='pred')
plt.ylabel("Total Number of Boarding")
plt.xlabel("Index")
plt.title("Comparison Between Prediction & True Values")
plt.legend()
plt.show()
```



In [124…
```
bb1['WeekBeginning'] = le.fit_transform(bb1['WeekBeginning'])
```

In [125…
```
df = bb1.sort_values(['WeekBeginning','StopName'])
```

In [126…
```
for i in df.columns:
    df[i].fillna(df[i].mode()[0], inplace=True)
df[["postcode", "holiday_label"]] = df[["postcode", "holiday_label"]].apply(pd.to_nu
```

In [127…
```
target_names = ['NumberOfBoardings_sum', 'NumberOfBoardings_count', 'NumberOfBoardin
train_col = ['StopName','WeekBeginning','type_x','latitude','longitude','postcode','
##want to predict 1 day in future.
shift_days = 6
shift_steps = shift_days * 3249
```

In [128…
```
df_targets = df[target_names].shift(-shift_steps)
x_data = df.iloc[:,1:].values[0:-shift_steps]
y_data = df_targets.values[:-shift_steps]
print(type(y_data))
print("Shape:", y_data.shape)
```

```
<class 'numpy.ndarray'>
Shape: (3672, 3)
```

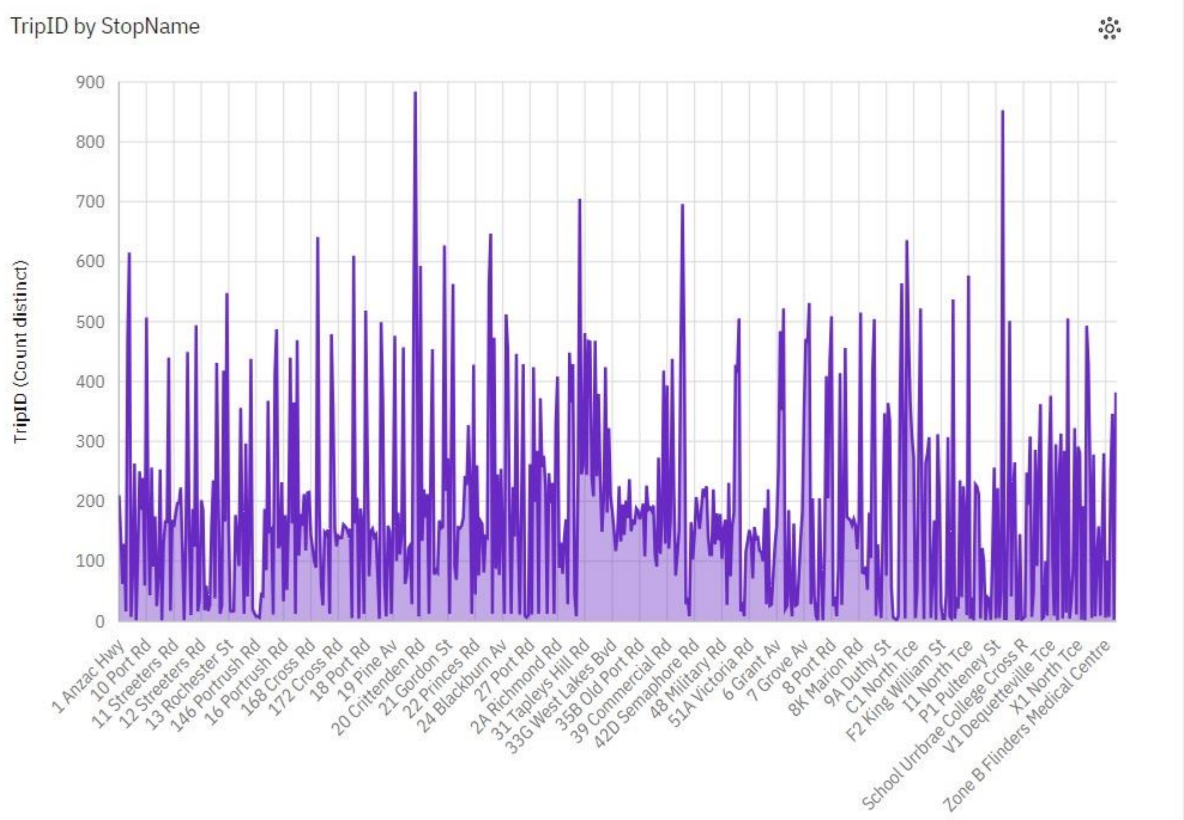
In [129…
```
##data split into 90% training and 10% testing
num_data = len(x_data)
train_split = 0.9
num_train = int(train_split * num_data)
x_train = x_data[0:num_train]
x_test = x_data[num_train:]
print(len(x_train) + len(x_test))
```

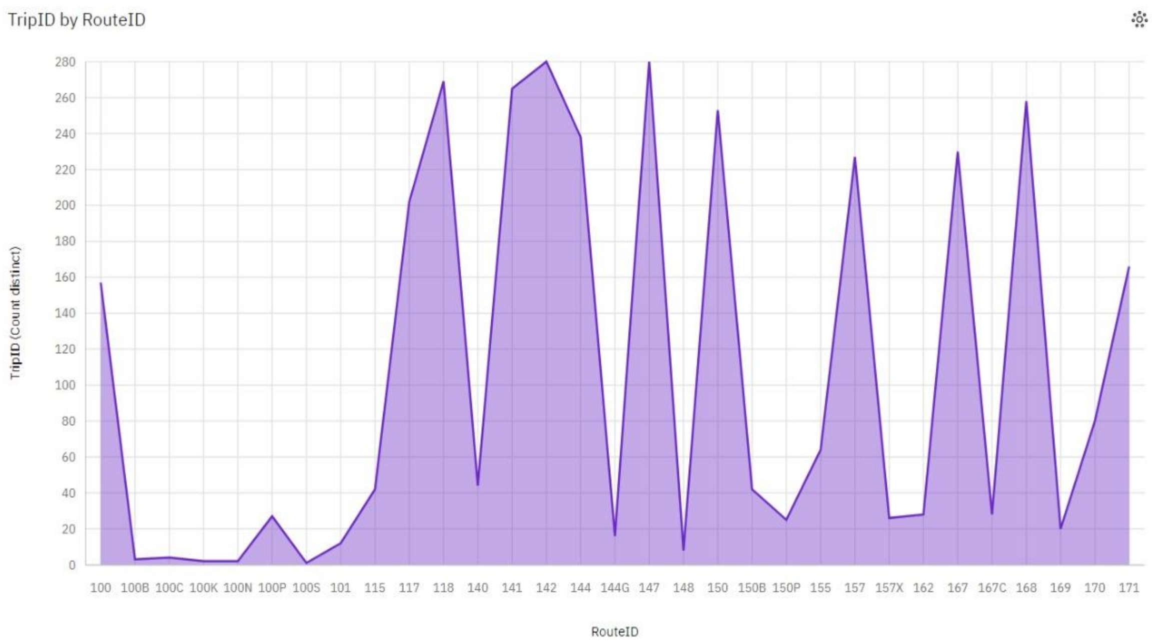
```
3672
```

In [130…
```
##target values for test and train
y_train = y_data[0:num_train]
y_test = y_data[num_train:]
print(len(y_train) + len(y_test))
##input dimension and output dimension
num_x_signals = x_data.shape[1]
print(num_x_signals)
num_y_signals = y_data.shape[1]
print(num_y_signals)
```

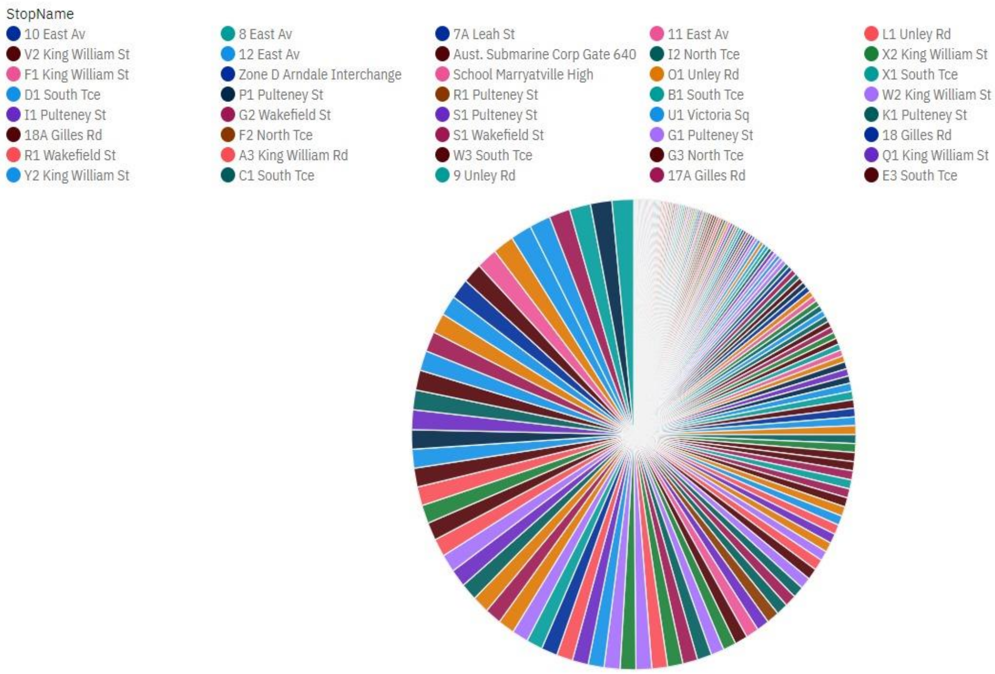
```
3672
10
3
```

DATA VISUVALIZATION USING COGNOS:

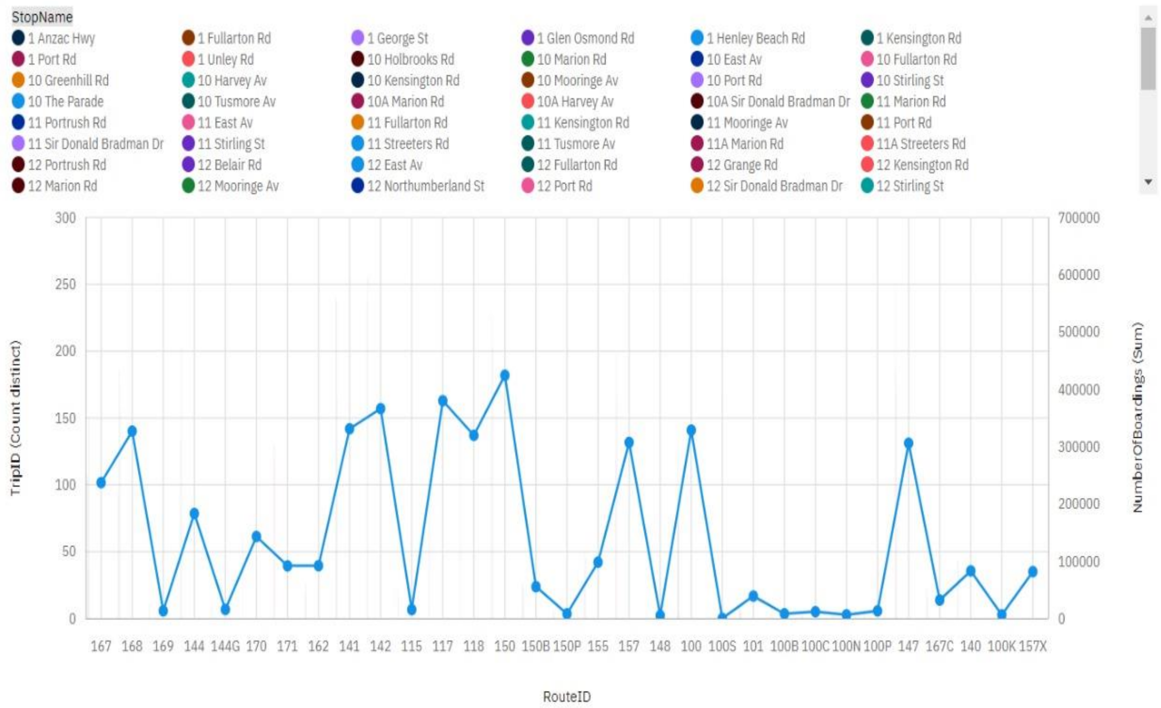




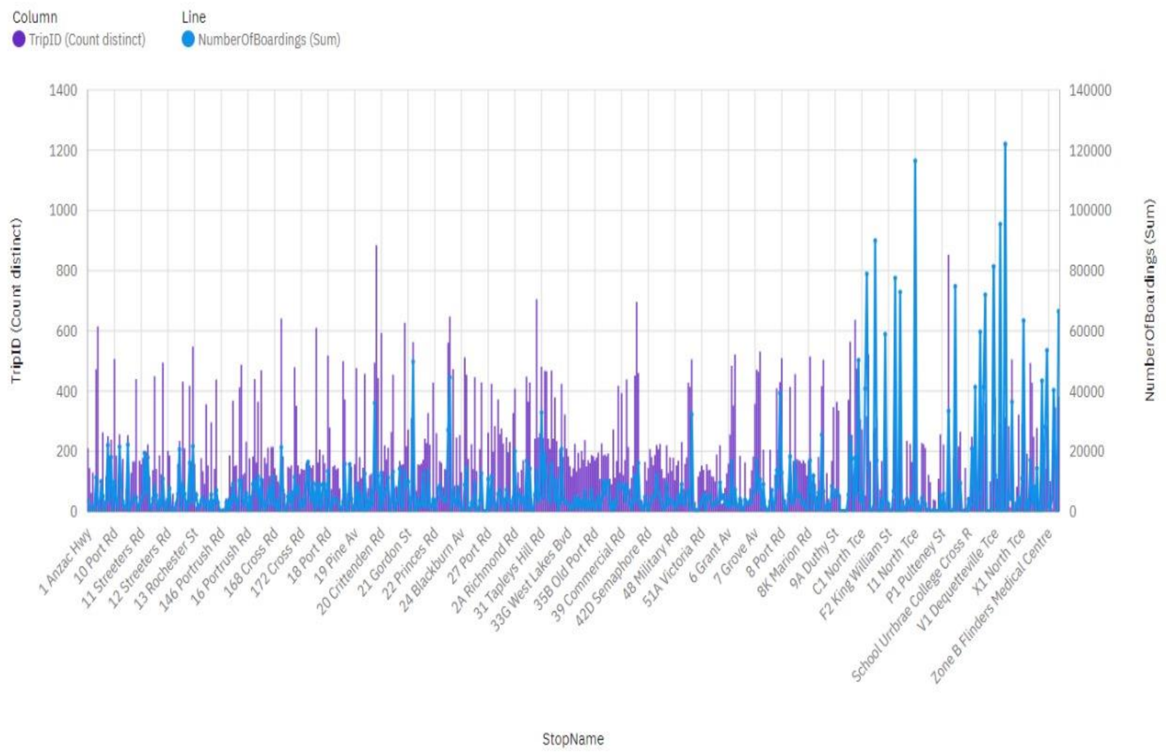
TripID by StopName



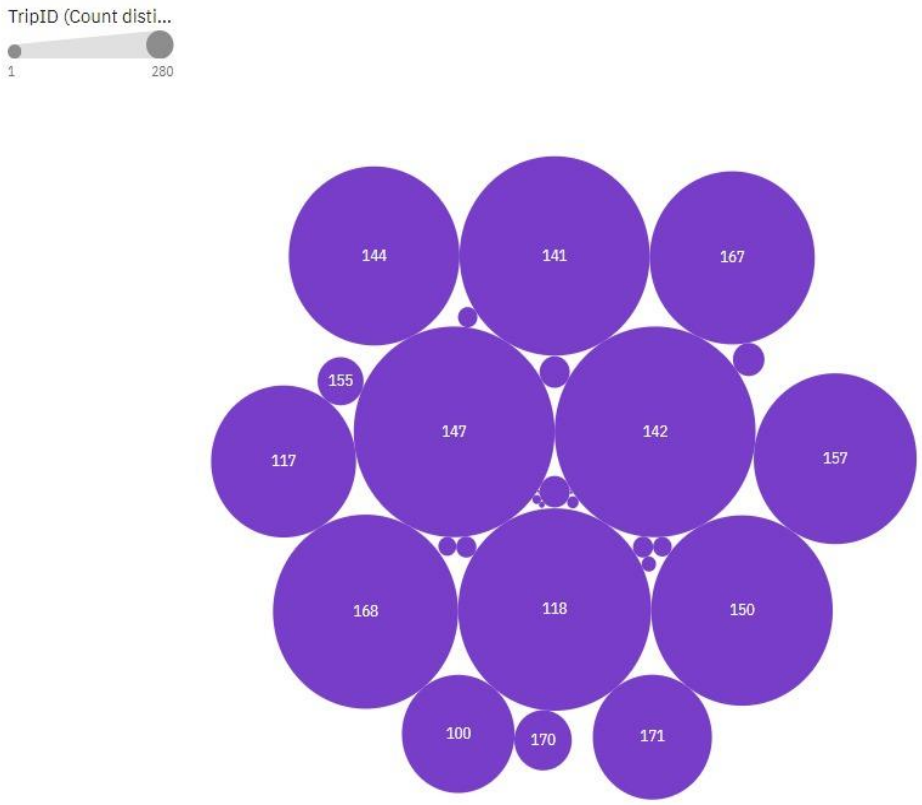
NumberOfBoardings and TripID for RouteID colored by StopName



NumberOfBoardings and TripID by StopName



RouteID sized by TripID



CONCLUSION:

The "Public Transportation Efficiency Analysis" project is an endeavor to address the challenges and enhance the efficiency of public transportation systems in our urban environments. Through a rigorous and data-driven approach, we have delved into the core issues affecting public transportation and strived to identify actionable solutions.

Our journey through this project has revealed valuable insights:

Data-Driven Insights: We have harnessed data analytics, complex analysis models, and machine learning algorithms to extract meaningful insights from vast and diverse datasets.

Challenges and Solutions: Throughout the project, we encountered challenges related to data collection, data quality, complex analysis, and the need for stakeholder engagement. We overcame these challenges through innovative solutions, interdisciplinary collaboration, and effective data governance.

Recommendations for Improvement: Our analysis has led to actionable recommendations for enhancing public transportation efficiency. These recommendations encompass route optimization, scheduling improvements, user experience enhancements, and technology integration.

Impact on Sustainability: By making public transportation more efficient, we have contributed to a more sustainable and environmentally friendly urban environment, reducing congestion, energy consumption, and carbon emissions.

As we conclude this project, we emphasize the significance of continuous improvement in public transportation systems. Our findings are a call to action for transportation authorities, urban planners, and stakeholders to implement the recommendations and enhancements proposed in this analysis.

This project is not an endpoint but a stepping stone towards a more efficient, accessible, and sustainable future for public transportation. We look forward to the adoption of these insights and the further development of public transportation systems in our cities.

Thank you for joining us on this journey to enhance public transportation efficiency. Together, we can create a brighter, greener, and more connected urban future.