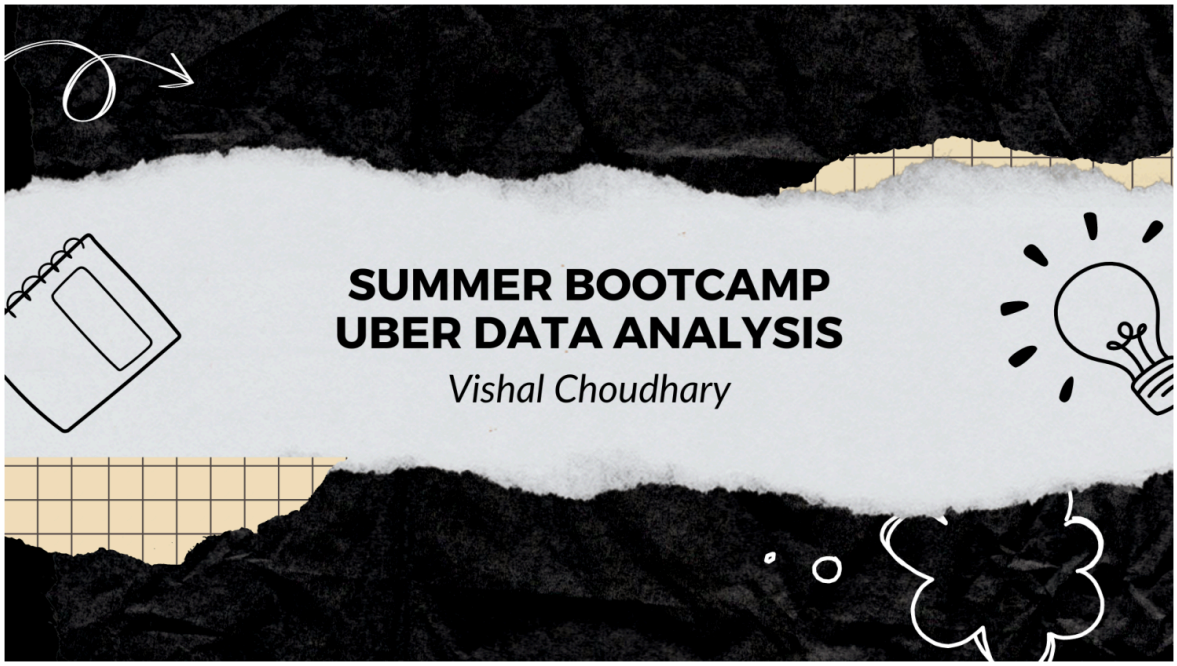
In [1]:
```
# Reading the image from a file
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
img = mpimg.imread('/Users/vishal/Desktop/Bootcamp.png')

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



Uber Data Analysis

In [2]:
```
import os
os.getcwd()
```

Out[2]:
```
'/Users/vishal'
```

Importing the necessary Libraries

In [3]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading the dataset

In [4]:
```
df = pd.read_csv('/Users/vishal/Desktop/3-Uber_Data_New.csv')
```

Basic Exploration

1- Display the top 5 rows.

In [5]:
```
df.head()
```

Out[5]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
0	1/1/2015 1:00	Bronx	152.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
1	1/1/2015 1:00	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	?
2	1/1/2015 1:00	EWR	0.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
3	1/1/2015 1:00	Manhattan	5258.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
4	1/1/2015 1:00	Queens	405.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y

In [6]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-19 at 11.27.49.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

'

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
0	1/1/2015 1:00	Bronx	152.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
1	1/1/2015 1:00	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	?
2	1/1/2015 1:00	EWR	0.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
3	1/1/2015 1:00	Manhattan	5258.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
4	1/1/2015 1:00	Queens	405.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y

Obervation :- on column 'temp' and 'hday' there is missing values on 2nd row Brooklyn temp = nan and hday = wrong entry(?)

2- Display the last 5 rows

In [7]:
```
df.tail()
```

Out[7]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
29096	30-06-2015 23:00	EWR	0.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29097	30-06-2015 23:00	Manhattan	3828.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29098	30-06-2015 23:00	Queens	580.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29099	30-06-2015 23:00	Staten Island	0.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29100	30-06-2015 23:00	NaN	3.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N

In [8]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-19 at 11.29.27.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
29096	30-06-2015 23:00	EWR	0.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29097	30-06-2015 23:00	Manhattan	3828.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29098	30-06-2015 23:00	Queens	580.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29099	30-06-2015 23:00	Staten Island	0.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N
29100	30-06-2015 23:00	NaN	3.0	7.0	10.0	75.0	65.0	1011.8	0.0	0.0	0.0	0.0	N

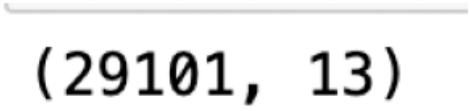
Obervation :- In column 'borough' and row 29100 value = NaN

3- Check the shape of dataset.

In [9]:
```
df.shape
```

Out[9]:
```
(29101, 13)
```

In [158]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.10.45.pn

# Displaying the image
plt.figure(dpi=60)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



Observation:- number of Rows = 29101 and number of column = 13

4- Check the info of dataset

In [36]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29101 entries, 0 to 29100
Data columns (total 13 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   pickup_dt  17559 non-null  object 
 1   borough    26058 non-null  object 
 2   pickups    29099 non-null  float64
 3   spd        29101 non-null  float64
 4   vsb        29101 non-null  float64
 5   temp       28742 non-null  float64
 6   dewp       29101 non-null  float64
 7   slp        29101 non-null  float64
 8   pcp01      29101 non-null  float64
 9   pcp06      29101 non-null  float64
 10  pcp24      29101 non-null  float64
 11  sd         29101 non-null  float64
 12  hday       29099 non-null  object 
dtypes: float64(10), object(3)
memory usage: 2.9+ MB
```

In [11]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 13.49.28.pn

# Displaying the image

plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 29101 entries, 0 to 29100
Data columns (total 13 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   pickup_dt  29101 non-null  object 
 1   borough    26058 non-null  object 
 2   pickups    29099 non-null  float64
 3   spd        29101 non-null  float64
 4   vsb        29101 non-null  float64
 5   temp       28742 non-null  float64
 6   dewp       29101 non-null  float64
 7   slp        29101 non-null  float64
 8   pcp01      29101 non-null  float64
 9   pcp06      29101 non-null  float64
 10  pcp24      29101 non-null  float64
 11  sd         29101 non-null  float64
 12  hday       29101 non-null  object 
dtypes: float64(10), object(3)
memory usage: 2.9+ MB
```

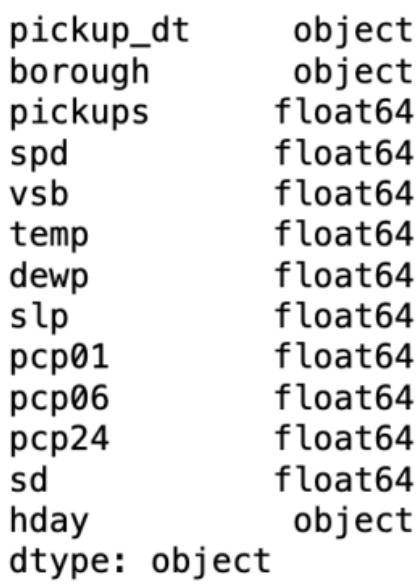
5- Check the datatypes of each feature.

In [12]:
```
df.dtypes
```

Out[12]:
```
pickup_dt     object
borough       object
pickups      float64
spd          float64
vsb          float64
temp         float64
dewp         float64
slp          float64
pcp01        float64
pcp06        float64
pcp24        float64
sd           float64
hday          object
dtype: object
```

Obeservation:- We seen that pickup_dt is in wrong data type we have to convert into date time format, and also we seen their is two more columns in our data set which is not showing here where names are weekday and is_weekend

In [13]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 13.50.15.pn

# Displaying the image

plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



Converting pickup_dt in date time format

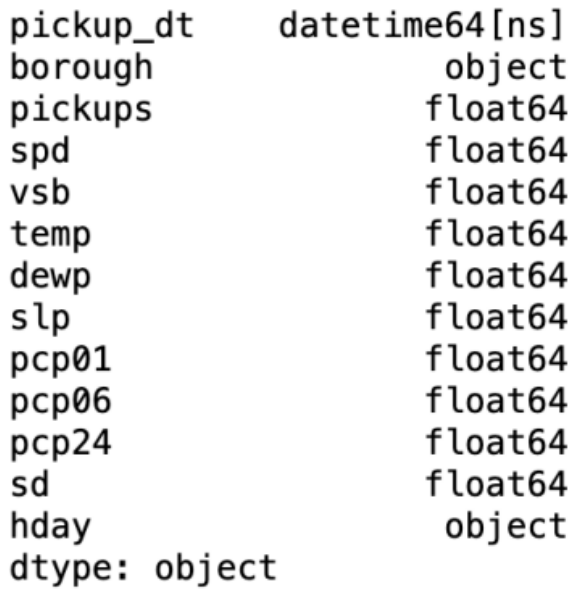
In [7]:
```
df['pickup_dt'] = pd.to_datetime(df['pickup_dt'], format='%d-%m-%Y %H:%M', errors='coerce'
```

After Converting pickup_dt in date time format

In [16]:
```
df.dtypes
```

Out[16]:
```
pickup_dt    datetime64[ns]
borough              object
pickups             float64
spd                 float64
vsb                 float64
temp                float64
dewp                float64
slp                 float64
pcp01               float64
pcp06               float64
pcp24               float64
sd                  float64
hday                 object
dtype: object
```

In [38]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 11.17.06.pn

# Displaying the image

plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



6- Check the Statistical summary

In [17]:
```
df.describe().T
```

Out[17]:

	count	mean	min	25%	50%	75%	max	std
pickup_dt	17559	2015-04-08 05:01:44.151717120	2015-01-13 00:00:00	2015-02-21 15:00:00	2015-04-14 00:00:00	2015-05-23 03:00:00	2015-06-30 23:00:00	NaN
pickups	29099.0	490.236022	0.0	1.0	54.0	449.0	7883.0	995.680628
spd	29101.0	5.984924	0.0	3.0	6.0	8.0	21.0	3.699007
vsb	29101.0	8.818125	0.0	9.1	10.0	10.0	10.0	2.442897
temp	28742.0	47.900262	0.0	32.0	46.5	65.0	89.0	19.800541
dewp	29101.0	30.823065	-16.0	14.0	30.0	50.0	73.0	21.283444
slp	29101.0	1052.633123	1.0	1012.5	1018.2	1022.9	1015200.0	5945.147362
pcp01	29101.0	0.00383	0.0	0.0	0.0	0.0	0.28	0.018933
pcp06	29101.0	0.026129	0.0	0.0	0.0	0.0	1.24	0.093125
pcp24	29101.0	0.090464	0.0	0.0	0.0	0.05	2.1	0.219402
sd	29101.0	2.529169	0.0	0.0	0.0	2.958333	19.0	4.520325

obervation:- minimum dew point = -16 and maximum sea level pressure = 1015200.00

In [39]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 11.18.21.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```


	count	mean	min	25%	50%	75%	max	std
pickup_dt	17559	2015-04-08 05:01:44.151717120	2015-01-13 00:00:00	2015-02-21 15:00:00	2015-04-14 00:00:00	2015-05-23 03:00:00	2015-06-30 23:00:00	NaN
pickups	29099.0	490.236022	0.0	1.0	54.0	449.0	7883.0	995.680628
spd	29101.0	5.984924	0.0	3.0	6.0	8.0	21.0	3.699007
vsb	29101.0	8.818125	0.0	9.1	10.0	10.0	10.0	2.442897
temp	28742.0	47.900262	0.0	32.0	46.5	65.0	89.0	19.800541
dewp	29101.0	30.823065	-16.0	14.0	30.0	50.0	73.0	21.283444
slp	29101.0	1052.633123	1.0	1012.5	1018.2	1022.9	1015200.0	5945.147362
pcp01	29101.0	0.00383	0.0	0.0	0.0	0.0	0.28	0.018933
pcp06	29101.0	0.026129	0.0	0.0	0.0	0.0	1.24	0.093125
pcp24	29101.0	0.090464	0.0	0.0	0.0	0.05	2.1	0.219402
sd	29101.0	2.529169	0.0	0.0	0.0	2.958333	19.0	4.520325

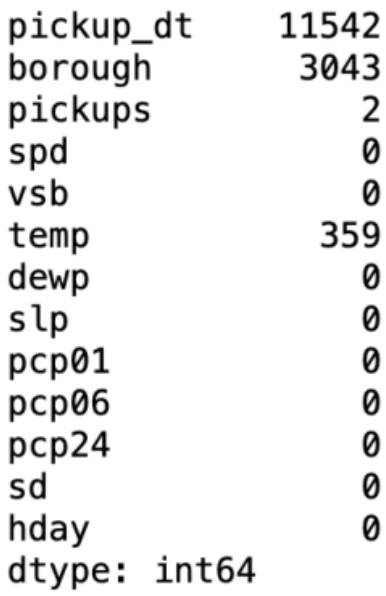
7- Check the null values

In [11]:
```
df.isnull().sum()
```

Out[11]:
```
pickup_dt    11542
borough       3043
pickups          2
spd              0
vsb              0
temp           359
dewp             0
slp              0
pcp01            0
pcp06            0
pcp24            0
sd               0
hday             0
dtype: int64
```

Observation:- column borough having "3043" null values, pickups having "2" null values and temp having "359" null values and pickup_dt having "11542" null values

In [20]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 09.10.53.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



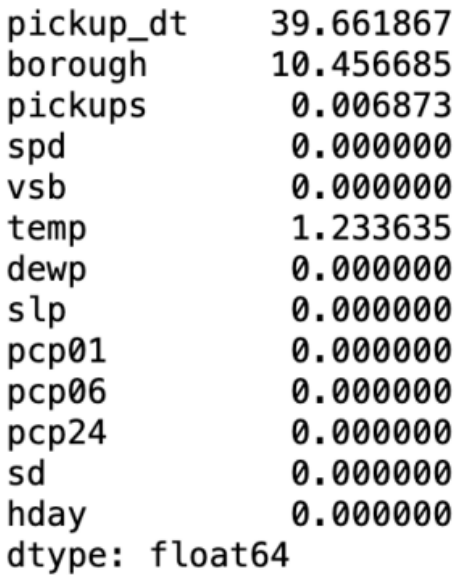
Null values in percentage

In [21]:
```
#Checking null vales in percentage
df.isnull().sum()/len(df)*100
```

Out[21]:
```
pickup_dt    39.661867
borough      10.456685
pickups       0.006873
spd           0.000000
vsb           0.000000
temp          1.233635
dewp          0.000000
slp           0.000000
pcp01         0.000000
pcp06         0.000000
pcp24         0.000000
sd            0.000000
hday          0.000000
dtype: float64
```

Observation :- Here hday is showing zero null value but by checking data set we seen there is two wrong value, we must have to replace wrong values from Null

In [22]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 09.11.59.pn

# Displaying the image
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



8. Check the duplicate values

In [9]:
```
df.duplicated().sum()
```

Out[9]:
```
0
```

In [10]:
```
df[df.duplicated(keep=False)]
```

Out[10]:

pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday

9- Check the anomalies or wrong entries.

Identifying Wrong Value

In [16]:
```
df[df['hday']=='?']
```

Out[16]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday	season	day_of_week
1	NaT	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	?	Fall	NaN
123	NaT	Queens	238.0	7.0	10.0	37.0	7.0	1016.2	0.0	0.0	0.0	0.0	?	Fall	NaN

In [17]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 02.12.40.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
1	2015-01-01 01:00:00	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	?
123	2015-01-01 19:00:00	Queens	238.0	7.0	10.0	37.0	7.0	1016.2	0.0	0.0	0.0	0.0	?

In [18]:
```
df["hday"]=df["hday"].replace('?', np.nan)
```

In [19]:
```
df[df['hday']=='?']
```

Out[19]:

pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday	season	day_of_week

In [20]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 02.19.01.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday

Observation in above we remove all wrong entries using Null values

After making some changes we are repeating step 7 again and again we check for NULL values

Top ten null values

In [21]:
```
#Checking top ten Null values
df[df.isnull().any(axis=1)].head(10)
```

Out[21]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday	season	day_of_week
0	NaT	Bronx	152.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
1	NaT	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	NaN	Fall	NaN
2	NaT	EWR	0.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
3	NaT	Manhattan	5258.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
4	NaT	Queens	405.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
5	NaT	Staten Island	6.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
6	NaT	NaN	4.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
7	NaT	Bronx	120.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
8	NaT	Brooklyn	1229.0	3.0	10.0	NaN	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
9	NaT	EWR	0.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN

In [22]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 12.32.20.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

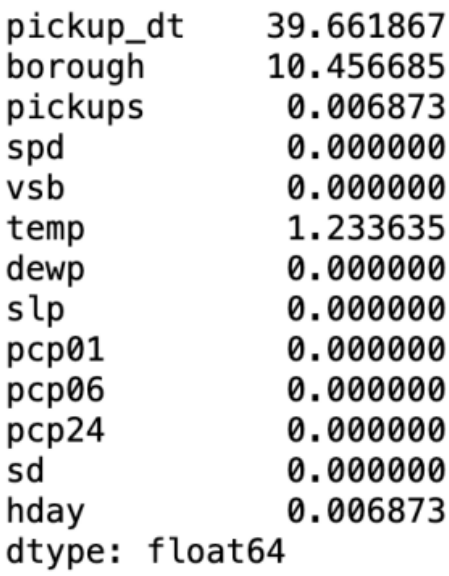
	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
1	1/1/2015 1:00	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	NaN
6	1/1/2015 1:00	NaN	4.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
8	1/1/2015 2:00	Brooklyn	1229.0	3.0	10.0	NaN	6.0	1023.0	0.0	0.0	0.0	0.0	Y
13	1/1/2015 2:00	NaN	11.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y
15	1/1/2015 3:00	Brooklyn	1601.0	5.0	10.0	NaN	8.0	1022.3	0.0	0.0	0.0	0.0	Y
20	1/1/2015 3:00	NaN	1.0	5.0	10.0	30.0	8.0	1022.3	0.0	0.0	0.0	0.0	Y
21	1/1/2015 4:00	Bronx	NaN	5.0	10.0	29.0	9.0	1022.0	0.0	0.0	0.0	0.0	Y
22	1/1/2015 4:00	Brooklyn	1390.0	5.0	10.0	NaN	9.0	1022.0	0.0	0.0	0.0	0.0	Y
27	1/1/2015 4:00	NaN	2.0	5.0	10.0	29.0	9.0	1022.0	0.0	0.0	0.0	0.0	Y
29	1/1/2015 5:00	Brooklyn	759.0	5.0	10.0	NaN	9.0	1021.8	0.0	0.0	0.0	0.0	Y

NULL Values in percentage

In [23]:
```
#Checking null vales in percentage
df.isnull().sum()/len(df)*100
```

Out[23]:
```
pickup_dt      39.661867
borough        10.456685
pickups         0.006873
spd             0.000000
vsb             0.000000
temp            1.233635
dewp            0.000000
slp             0.000000
pcp01           0.000000
pcp06           0.000000
pcp24           0.000000
sd              0.000000
hday            0.006873
season          0.000000
day_of_week    39.661867
dtype: float64
```

In [24]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 09.15.28.pn

# Displaying the image
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



Observation :- Now we can see in our data "hday" also having null values and after converting pickup_dt into date time fromat we can also see null values in pickup_dt

BEFORE Replacing all null values using median and mode because we also have objective type data type

In [25]:
```
df[df.isnull().any(axis=1)].head(10)
```

Out[25]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday	season	day_of_week
0	NaT	Bronx	152.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
1	NaT	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	NaN	Fall	NaN
2	NaT	EWR	0.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
3	NaT	Manhattan	5258.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
4	NaT	Queens	405.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
5	NaT	Staten Island	6.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
6	NaT	NaN	4.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
7	NaT	Bronx	120.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
8	NaT	Brooklyn	1229.0	3.0	10.0	NaN	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
9	NaT	EWR	0.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN

In [26]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 12.32.20.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday
1	1/1/2015 1:00	Brooklyn	1519.0	5.0	10.0	NaN	7.0	1023.5	0.0	0.0	0.0	0.0	NaN
6	1/1/2015 1:00	NaN	4.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y
8	1/1/2015 2:00	Brooklyn	1229.0	3.0	10.0	NaN	6.0	1023.0	0.0	0.0	0.0	0.0	Y
13	1/1/2015 2:00	NaN	11.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y
15	1/1/2015 3:00	Brooklyn	1601.0	5.0	10.0	NaN	8.0	1022.3	0.0	0.0	0.0	0.0	Y
20	1/1/2015 3:00	NaN	1.0	5.0	10.0	30.0	8.0	1022.3	0.0	0.0	0.0	0.0	Y
21	1/1/2015 4:00	Bronx	NaN	5.0	10.0	29.0	9.0	1022.0	0.0	0.0	0.0	0.0	Y
22	1/1/2015 4:00	Brooklyn	1390.0	5.0	10.0	NaN	9.0	1022.0	0.0	0.0	0.0	0.0	Y
27	1/1/2015 4:00	NaN	2.0	5.0	10.0	29.0	9.0	1022.0	0.0	0.0	0.0	0.0	Y
29	1/1/2015 5:00	Brooklyn	759.0	5.0	10.0	NaN	9.0	1021.8	0.0	0.0	0.0	0.0	Y

In [27]:
```
# Calculate the mode of the "city" column and "Holiday" column
mode_city = df["borough"].mode()[0]
mode_holiday = df["hday"].mode()[0]

df["borough"].fillna(mode_city, inplace=True)
df["hday"].fillna(mode_holiday, inplace=True)
```

In [28]:
```
median_pickups = df["pickups"].median()
median_temp = df["temp"].median()

# Replace NaN values with the calculated medians
df["pickups"].fillna(median_pickups, inplace=True)
df["temp"].fillna(median_temp, inplace=True)
```

After replacing all null values using median and mode because we also have objective type data type

In [29]:
```
df[df.isnull().any(axis=1)].head(10)
```

Out[29]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday	season	day_of_week
0	NaT	Bronx	152.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
1	NaT	Brooklyn	1519.0	5.0	10.0	46.5	7.0	1023.5	0.0	0.0	0.0	0.0	N	Fall	NaN
2	NaT	EWR	0.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
3	NaT	Manhattan	5258.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
4	NaT	Queens	405.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
5	NaT	Staten Island	6.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
6	NaT	Bronx	4.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
7	NaT	Bronx	120.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
8	NaT	Brooklyn	1229.0	3.0	10.0	46.5	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
9	NaT	EWR	0.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN

In [30]:
```
df.head(10)
```

Out[30]:

	pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday	season	day_of_week
0	NaT	Bronx	152.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
1	NaT	Brooklyn	1519.0	5.0	10.0	46.5	7.0	1023.5	0.0	0.0	0.0	0.0	N	Fall	NaN
2	NaT	EWR	0.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
3	NaT	Manhattan	5258.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
4	NaT	Queens	405.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
5	NaT	Staten Island	6.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
6	NaT	Bronx	4.0	5.0	10.0	30.0	7.0	1023.5	0.0	0.0	0.0	0.0	Y	Fall	NaN
7	NaT	Bronx	120.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
8	NaT	Brooklyn	1229.0	3.0	10.0	46.5	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN
9	NaT	EWR	0.0	3.0	10.0	30.0	6.0	1023.0	0.0	0.0	0.0	0.0	Y	Fall	NaN

In [31]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 11.23.23.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

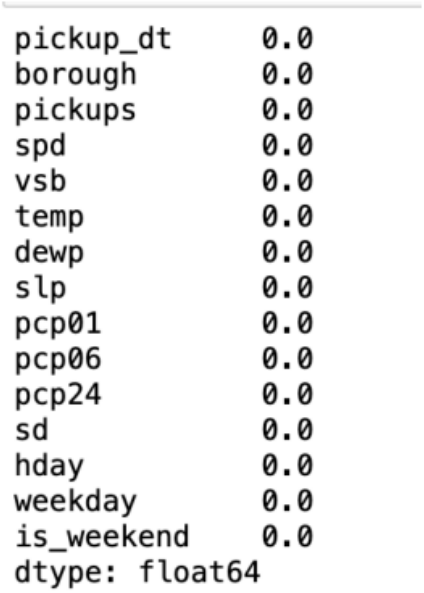
pickup_dt	borough	pickups	spd	vsb	temp	dewp	slp	pcp01	pcp06	pcp24	sd	hday

After Removing all NULL Values showing in percentage

In [32]:
```
df.isnull().sum()/len(df)*100
```

Out[32]:
```
pickup_dt      39.661867
borough         0.000000
pickups         0.000000
spd             0.000000
vsb             0.000000
temp            0.000000
dewp            0.000000
slp             0.000000
pcp01           0.000000
pcp06           0.000000
pcp24           0.000000
sd              0.000000
hday            0.000000
season          0.000000
day_of_week    39.661867
dtype: float64
```

In [33]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-20 at 11.36.56.pn

# Displaying the image
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



Solving pickup_dt issues

In [37]:
```
# Forward fill
df['pickup_dt'] = df['pickup_dt'].fillna(method='ffill')
```

```
/var/folders/cz/49clhtds01v79zh558p1fb3m0000gn/T/ipykernel_6205/2513405293.py:2: FutureWa
rning: Series.fillna with 'method' is deprecated and will raise in a future version. Use
obj.ffill() or obj.bfill() instead.
  df['pickup_dt'] = df['pickup_dt'].fillna(method='ffill')
```

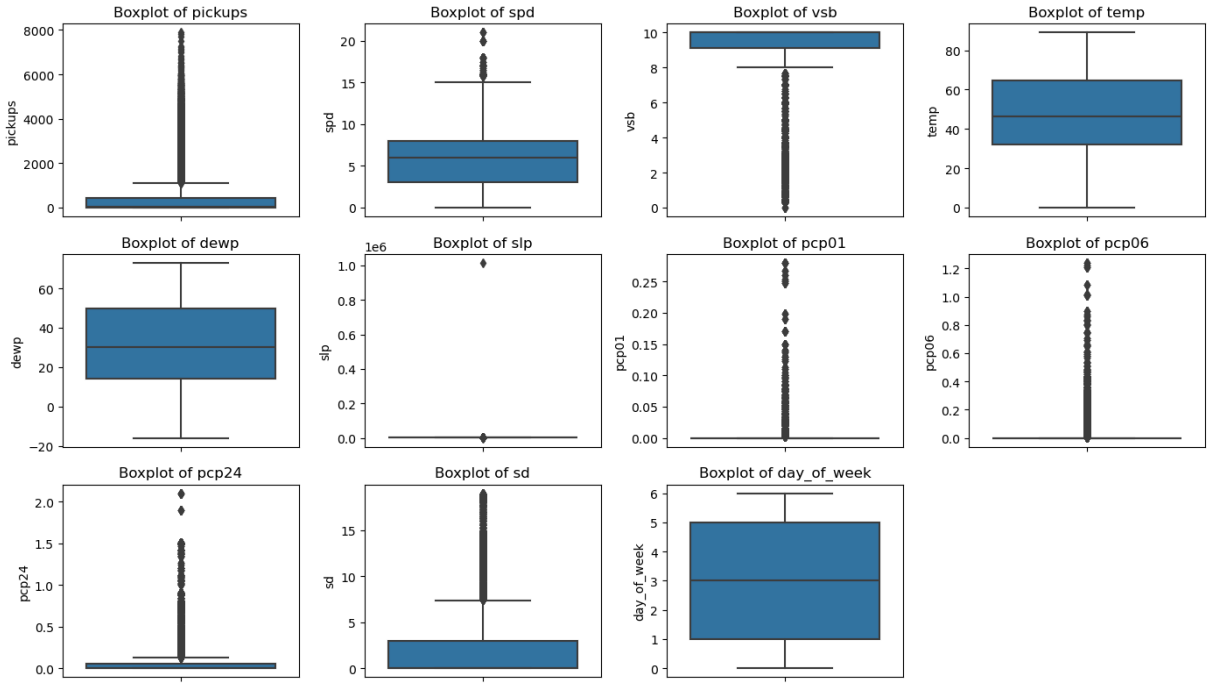
10- Do the necessary data cleaning steps like dropping duplicates, unnecessary columns, null value imputation, outliers treatment etc.

A. DATA Cleaning Steps

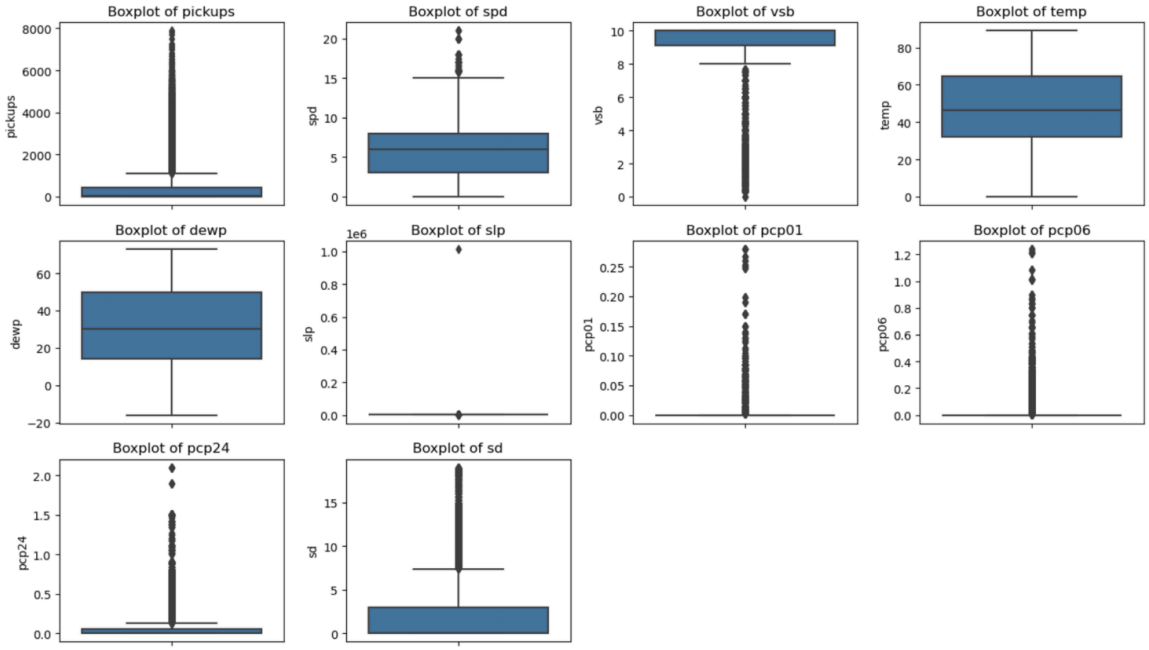
In [38]:
```
# Dropping duplicates
df= df.drop_duplicates()
```

B .Check for Outliers: Identify outliers in numerical columns.

In [40]:
```
plt.figure(figsize=(14, 8))
for i, column in enumerate(df.select_dtypes(include=[np.number]).columns, 1):
    plt.subplot(3, 4, i)
    sns.boxplot(y=df[column])
    plt.title(f'Boxplot of {column}')
plt.tight_layout()
plt.show()
```



In [87]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 13.32.03.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

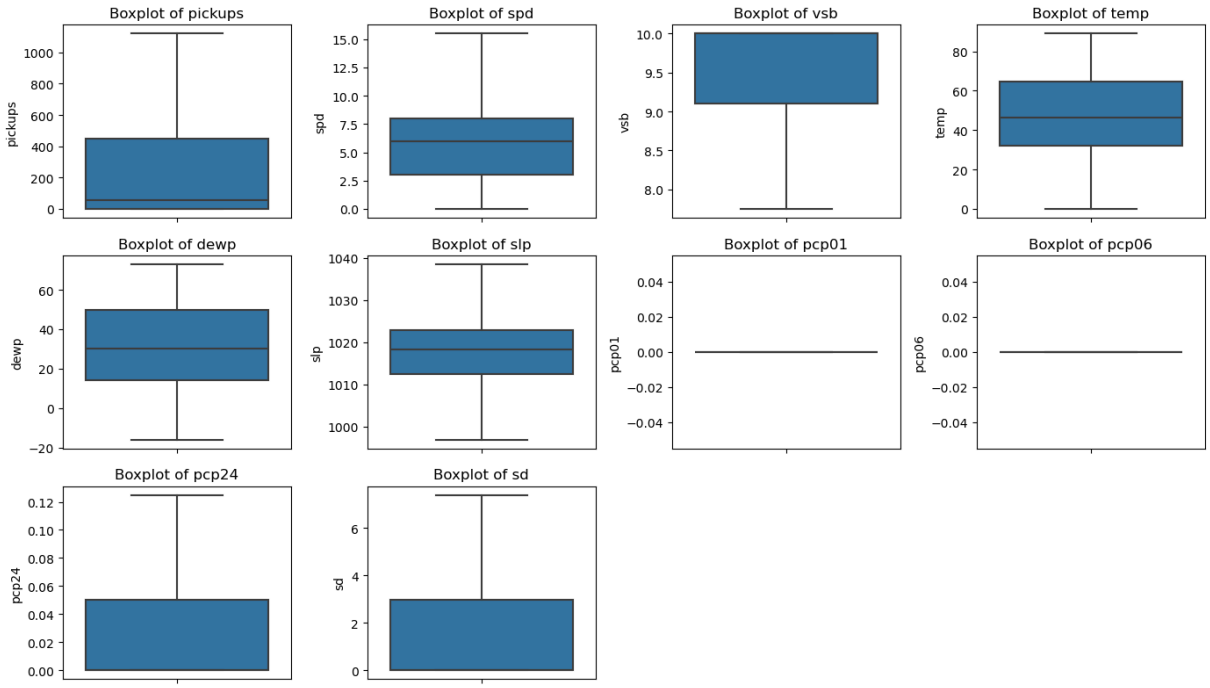


Treating Outliers

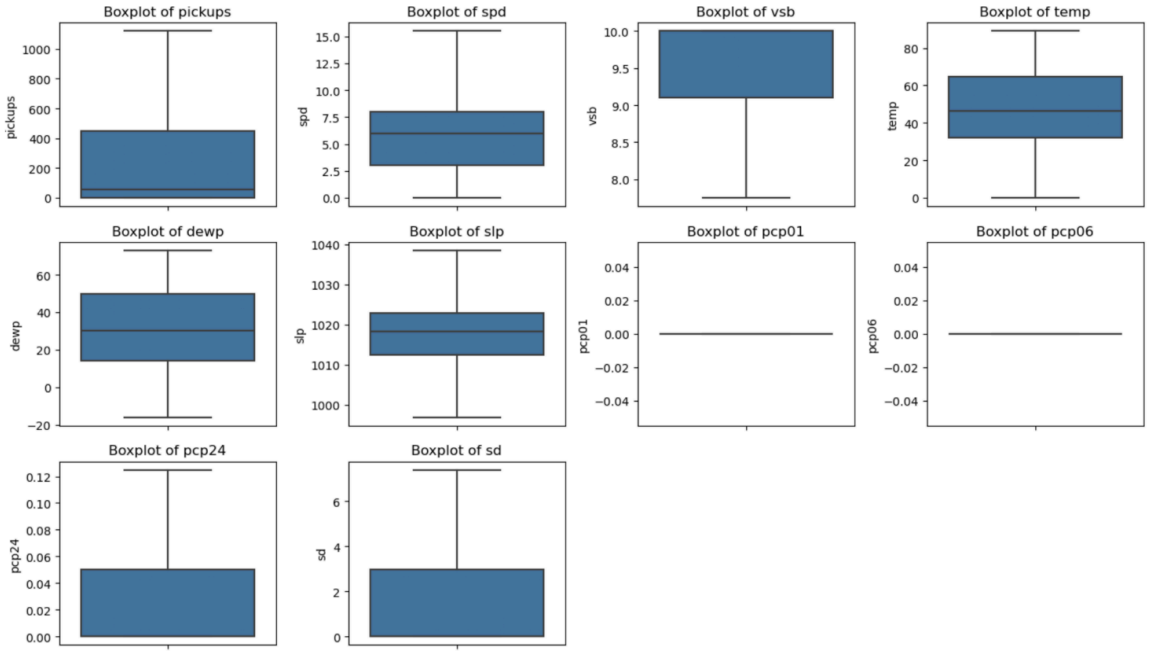
In [32]:
```
#Treating Outliers
def remove_outlier(col):
    sorted(col)
    Q1,Q3=col.quantile([0.25,0.75])
    IQR=Q3-Q1
    lower_range= Q1-(1.5 * IQR)
    upper_range= Q3+(1.5 * IQR)
    return lower_range, upper_range

for i in df.columns:
    if df[i].dtype !='object':
        lr,ur=remove_outlier(df[i])
        df[i]=np.where(df[i]>ur,ur,df[i])
        df[i]=np.where(df[i]<lr,lr,df[i])
```

After treating outliers

In [34]:
```
plt.figure(figsize=(14, 8))
for i, column in enumerate(df.select_dtypes(include=[np.number]).columns, 1):
    plt.subplot(3, 4, i)
    sns.boxplot(y=df[column])
    plt.title(f'Boxplot of {column}')
plt.tight_layout()
plt.show()
```



In [35]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-21 at 12.56.52.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



1. Pickup Analysis

1. What is the total number of Uber pickups across all boroughs?

Output

In [42]:
```
total_pickups = df['pickups'].sum()
print("Total number of Uber pickups across all boroughs : ", total_pickups)
```

```
Total number of Uber pickups across all boroughs :  14265486.0
```

In [155]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.08.16.pn

# Displaying the image
plt.figure(dpi=140)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

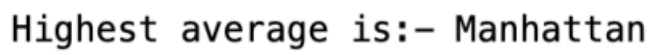
2. Which borough has the highest average number of hourly pickups?

Output

In [151]:
```
highest_avg = df.groupby('borough')['pickups'].mean().idxmax()
print("Highest average is:-",highest_avg)
```

```
Highest average is:- Manhattan
```

In [152]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.06.27.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



3. How do the number of pickups vary across different hours of the day?

Output

Numerical representation

In [44]:
```
pickups_per_hour = df.groupby(['hday'])['pickups'].sum()
print("Number of pickups across different hours of the day:")
print(pickups_per_hour)
```

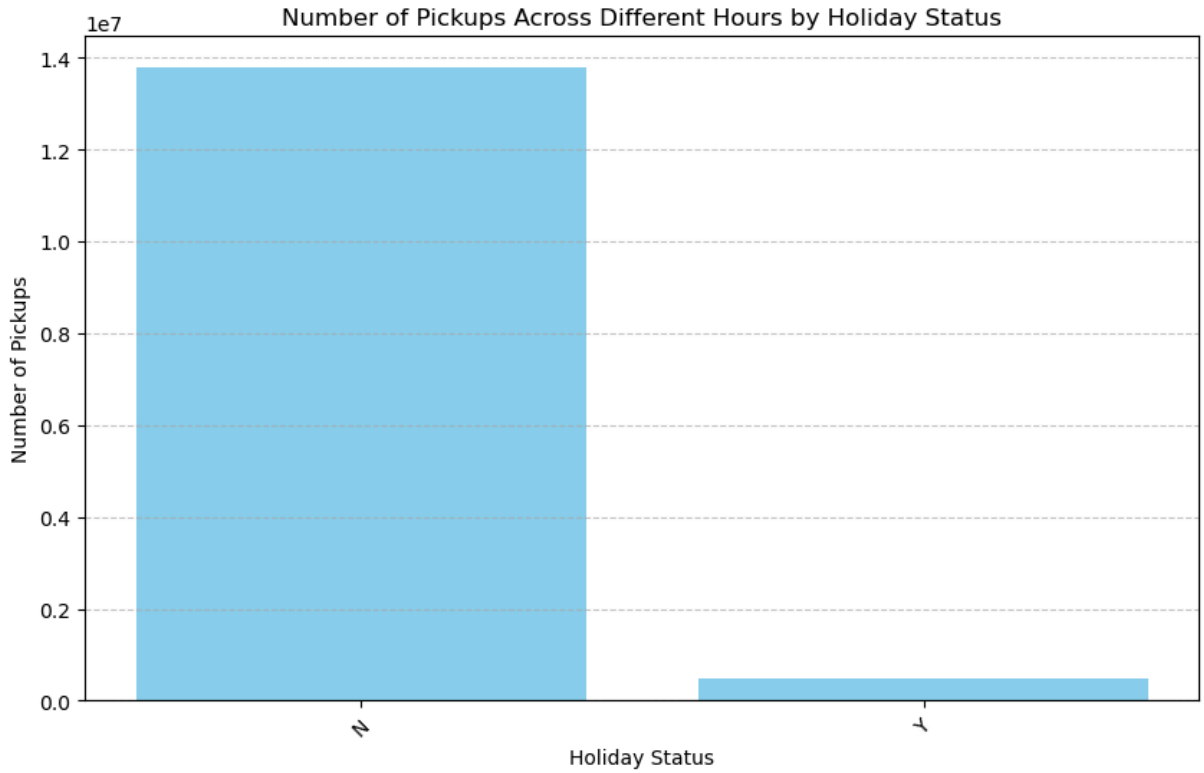
```
Number of pickups across different hours of the day:
hday
N    13777429.0
Y      488057.0
Name: pickups, dtype: float64
```

In [150]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.04.02.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

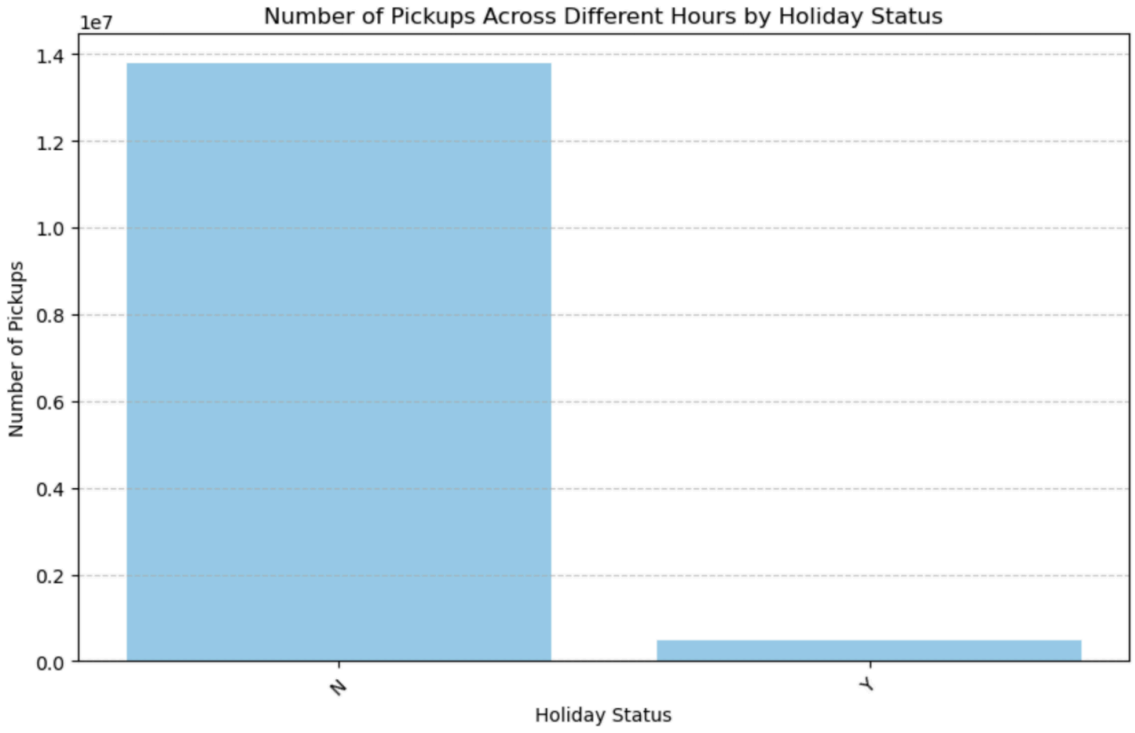
Number of pickups across different hours of the day:
hday
N   13777429.0
Y   488057.0
Name: pickups, dtype: float64

Graphical Represenatation

In [146]:
```
# Group by 'hday' and sum pickups
pickups_per_hour = df.groupby('hday')['pickups'].sum().reset_index()

# Sort by 'hday' to ensure proper ordering on the plot
pickups_per_hour = pickups_per_hour.sort_values(by='hday')

# Create a bar plot
plt.figure(figsize=(10, 6))
plt.bar(pickups_per_hour['hday'], pickups_per_hour['pickups'], color='skyblue')
plt.title('Number of Pickups Across Different Hours by Holiday Status')
plt.xlabel('Holiday Status')
plt.ylabel('Number of Pickups')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



In [148]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.03.25.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



4. Which day of the week has the highest number of pickups?

Output

In [46]:
```
df['pickup_dt'] = pd.to_datetime(df['pickup_dt'], format='%d-%m-%Y %H:%M', errors='coerce')
df['season'] = df['pickup_dt'].dt.month.map(
    lambda x: 'Winter' if x in [12, 1, 2] else 'Spring' if x in [3, 4, 5] else 'Summer' if
)
pickups_per_season = df.groupby('season')['pickups'].sum()
print(pickups_per_season)
```

```
season
Fall       700499.0
Spring    7507697.0
Summer    1700758.0
Winter    4356532.0
Name: pickups, dtype: float64
```

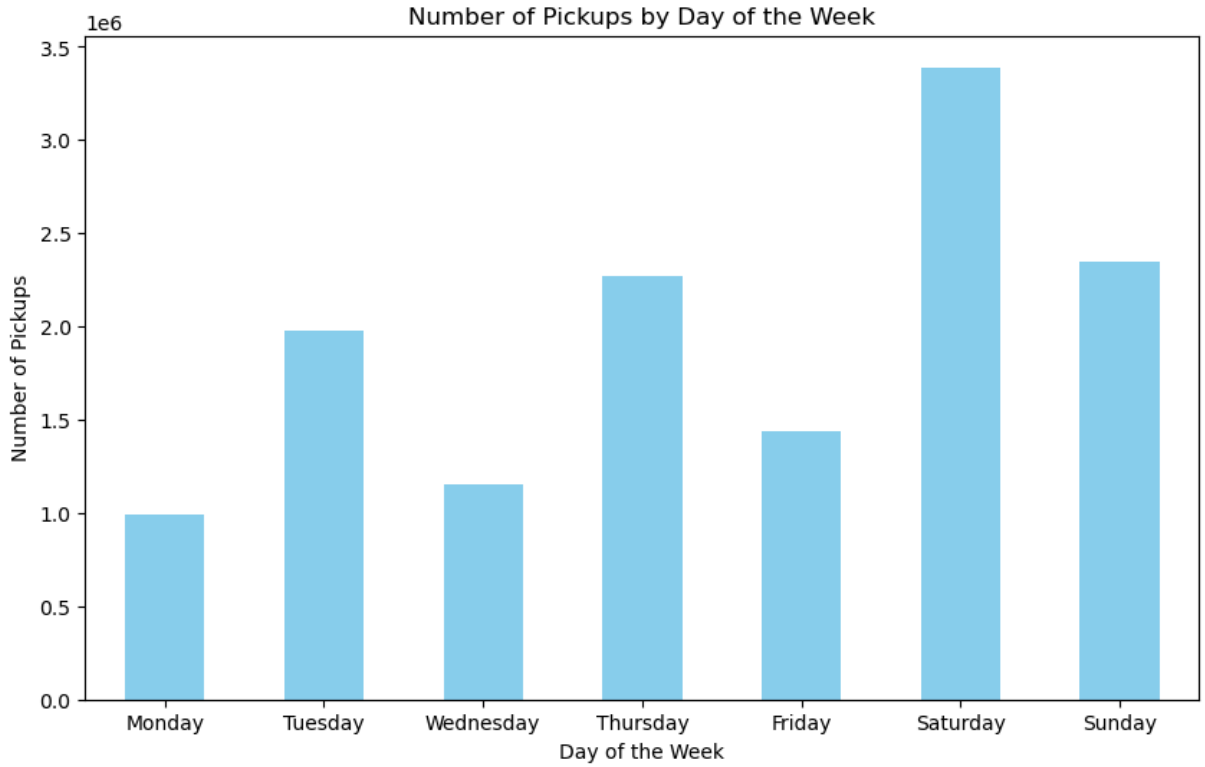
Numerical data represenation

In [145]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.58.38.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

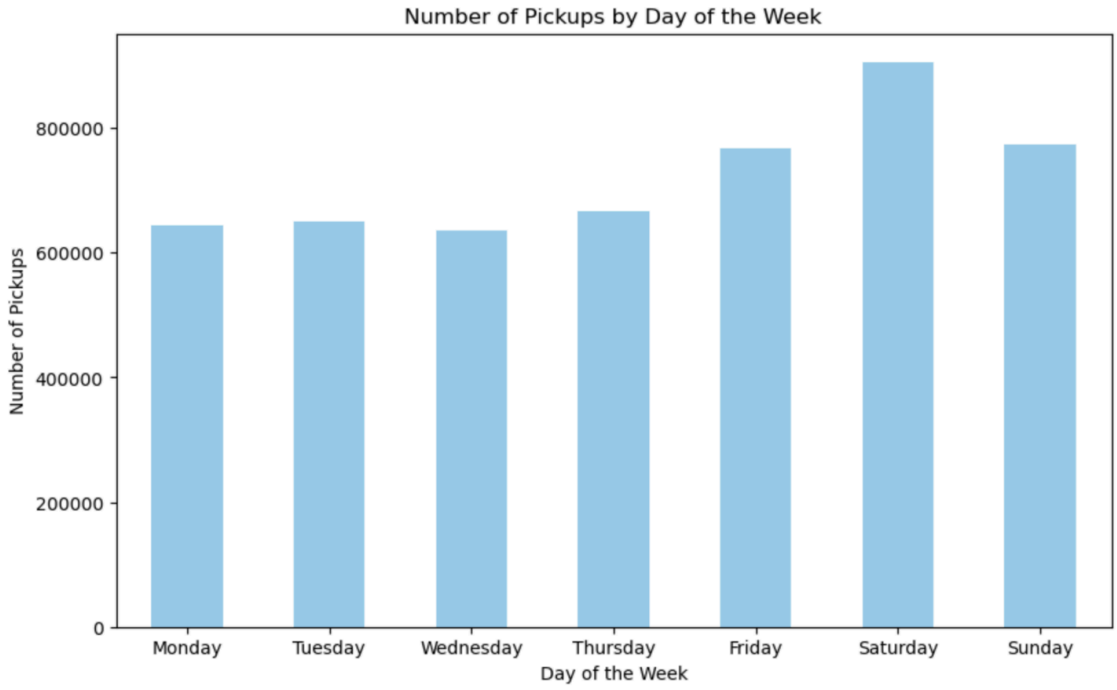
```
season
Fall       700499.0
Spring    7507697.0
Summer    1700758.0
Winter    4356532.0
Name: pickups, dtype: float64
```

Graphical Representation

In [47]:
```
ickup_dt'] = pd.to_datetime(df['pickup_dt'], errors='coerce')
ay_of_week'] = df['pickup_dt'].dt.dayofweek
ps_by_day = df.groupby('day_of_week')['pickups'].sum()
igure(figsize=(10, 6))
ps_by_day.plot(kind='bar', color='skyblue')
itle('Number of Pickups by Day of the Week')
label('Day of the Week')
label('Number of Pickups')
ticks(ticks=range(7), labels=['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Satu
how()
```



In [48]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 11.33.54.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



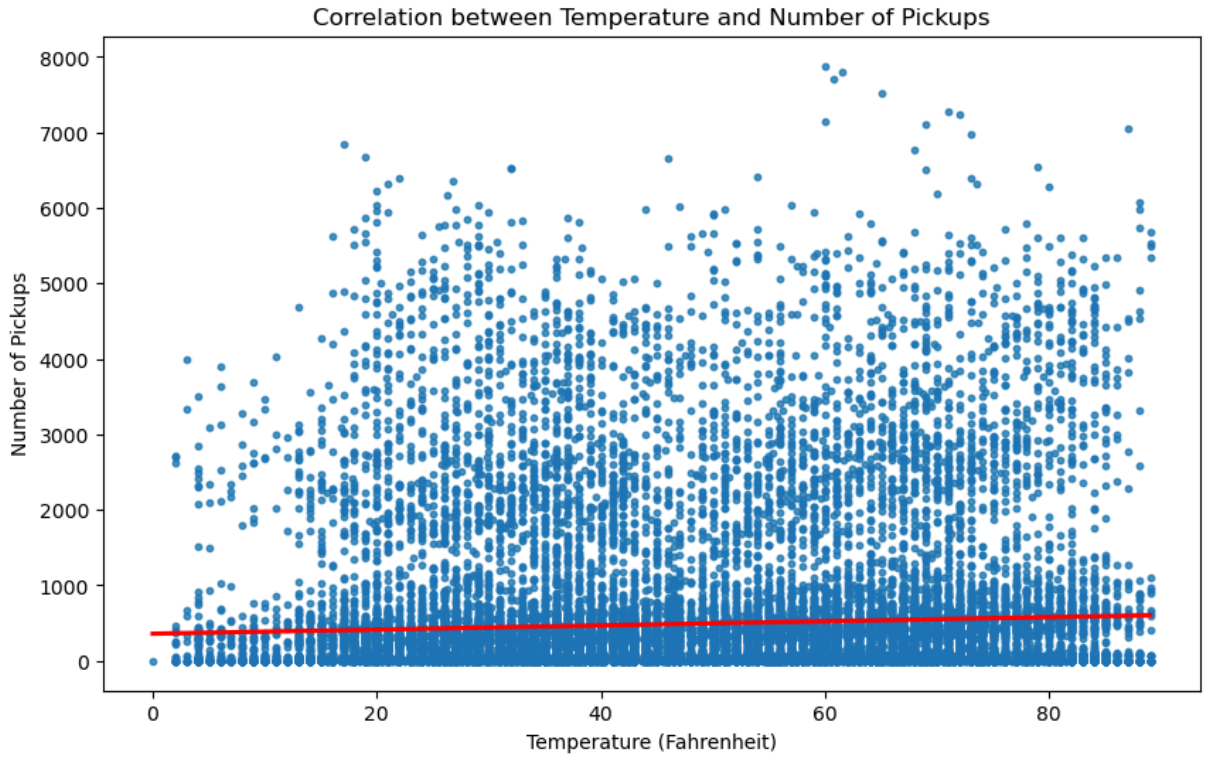
2. Weather Impact

1. What is the correlation between temperature and the number of pickups?

Output

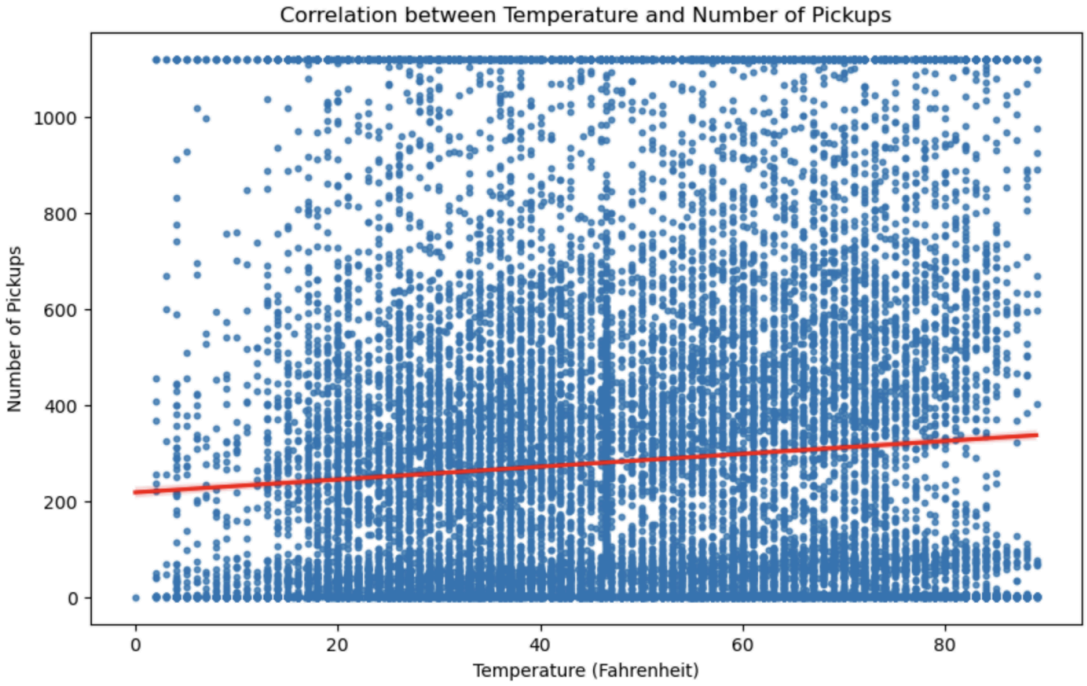
In [49]:
```
correlation_temp_pickups = df['temp'].corr(df['pickups'])
print("Correlation between temperature and number of pickups:", correlation_temp_pickups)
# Create a scatter plot with a regression line
plt.figure(figsize=(10, 6))
sns.regplot(x='temp', y='pickups', data=df, scatter_kws={'s':10}, line_kws={'color':'red'}
plt.title('Correlation between Temperature and Number of Pickups')
plt.xlabel('Temperature (Fahrenheit)')
plt.ylabel('Number of Pickups')
plt.show()
```

```
Correlation between temperature and number of pickups: 0.05422465020651789
```



In [50]:
```
# Reading the image from a file
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 11.31.37.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

```
Correlation between temperature and number of pickups: 0.06813480303893267
```



2. How does visibility impact the number of pickups?

Output

In [51]:
```
df.groupby('borough').apply(lambda x: x['temp'].corr(x['pickups']))
```

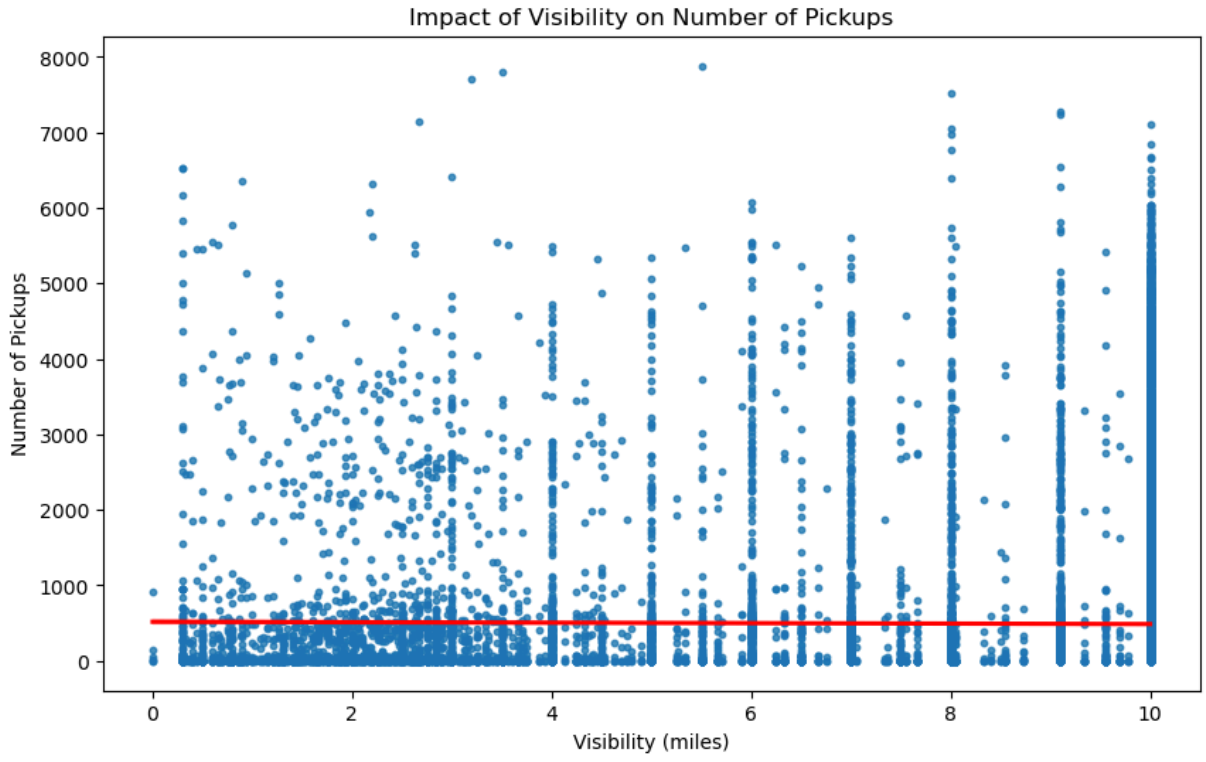
Out[51]:
```
borough
Bronx            0.200889
Brooklyn         0.291973
EWR              0.045346
Manhattan        0.159015
Queens           0.401791
Staten Island    0.243382
dtype: float64
```

Numerical representation

In [143]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.56.59.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

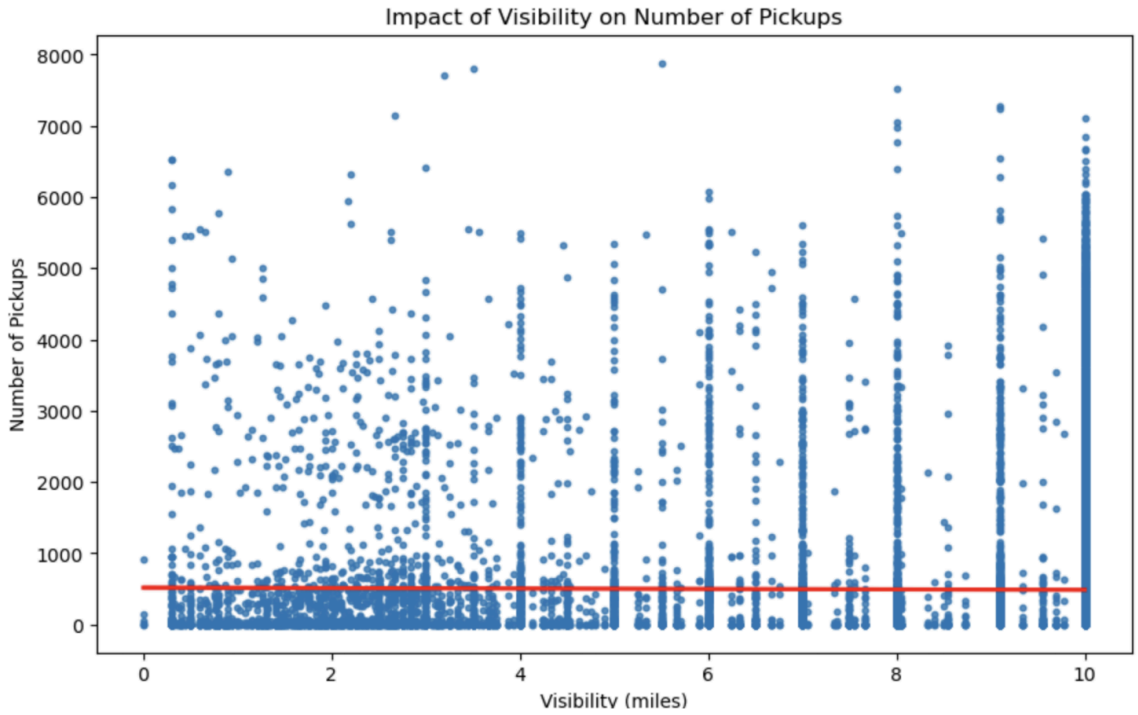
```
borough
Bronx            0.200889
Brooklyn         0.291973
EWR              0.045346
Manhattan        0.159015
Queens           0.401791
Staten Island    0.243382
dtype: float64
```

In [52]:
```
plt.figure(figsize=(10, 6))
sns.regplot(x='vsb', y='pickups', data=df, scatter_kws={'s':10}, line_kws={'color':'red'})
plt.title('Impact of Visibility on Number of Pickups')
plt.xlabel('Visibility (miles)')
plt.ylabel('Number of Pickups')
plt.show()
```



Graphical representation

In [141]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.55.01.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



3. Is there a relationship between wind speed and the number of pickups?

In [53]:
```
correlation_spd_pickups = df['spd'].corr(df['pickups'])
print("Correlation between wind speed and number of pickups:", correlation_spd_pickups)
```

```
Correlation between wind speed and number of pickups: 0.011075499985935629
```

Output

In [139]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.52.25.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

```
Correlation between wind speed and number of pickups: 0.011075499985935629
```

4. How does precipitation (1-hour, 6-hour, 24-hour) affect the number of pickups?

In [54]:
```
correlation_pcp01_pickups = df['pickups'].corr(df['pcp01'])
print("Correlation between 1-hour precipitation and number of pickups:", correlation_pcp01_
```

```
Correlation between 1-hour precipitation and number of pickups: 0.004398191771173067
```

Output

In [140]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.53.39.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

Correlation between 1-hour precipitation and number of pickups: 0.004398191771173067

3. Seasonal Trends

1. How do the number of pickups vary across different seasons (winter, spring, summer, fall)?

Output

In [57]:
```
pickups_per_season = df.assign(season=df['pickup_dt'].dt.month.map(lambda x: 'Winter' if x
print(pickups_per_season)
```

```
season
Fall       700499.0
Spring    7507697.0
Summer    1700758.0
Winter    4356532.0
Name: pickups, dtype: float64
```

In [137]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.50.36.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

season
Fall 700499.0
Spring 7507697.0
Summer 1700758.0
Winter 4356532.0
Name: pickups, dtype: float64

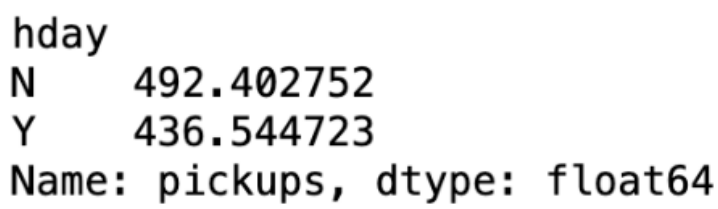
2. What is the average number of pickups during holidays compared to non-holidays?

In [58]:
```
df.groupby('hday')['pickups'].mean()
```

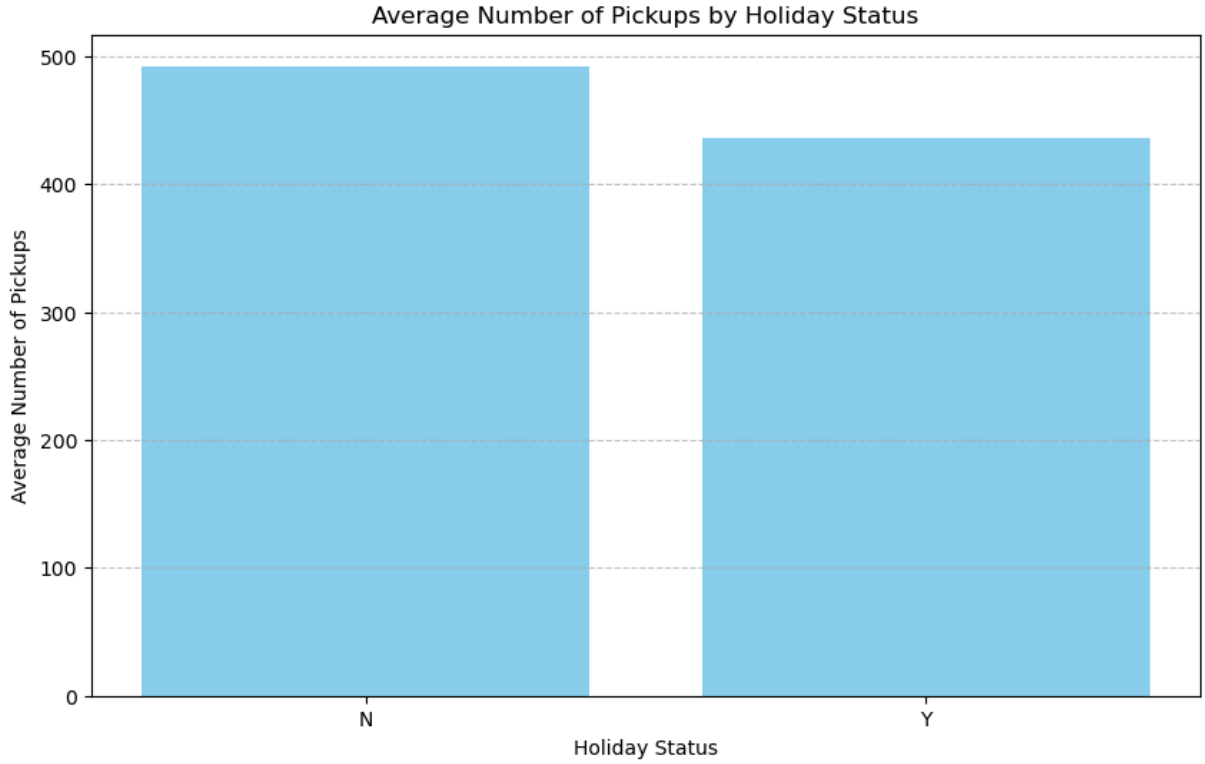
Out[58]:
```
hday
N    492.402752
Y    436.544723
Name: pickups, dtype: float64
```

Numerical representation

In [135]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.49.43.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

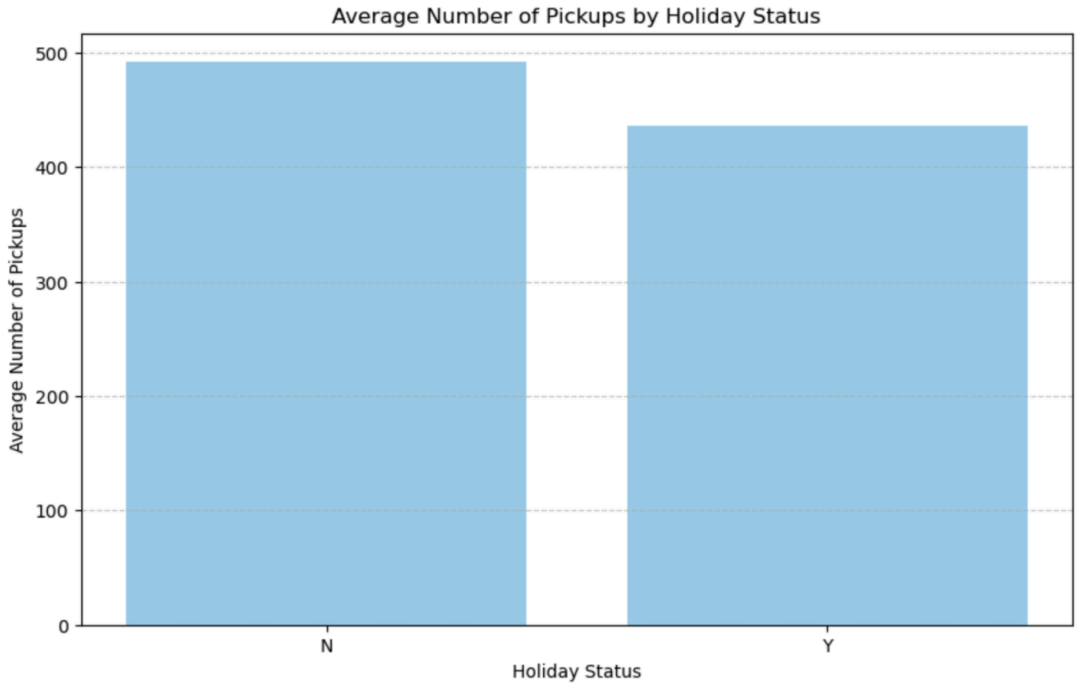


In [128]:
```
# Group by 'hday' and calculate mean pickups
avg_pickups_by_holiday = df.groupby('hday')['pickups'].mean().reset_index()

# Sort by holiday status to ensure proper ordering on the plot
avg_pickups_by_holiday = avg_pickups_by_holiday.sort_values(by='hday')

# Create a bar plot
plt.figure(figsize=(10, 6))
plt.bar(avg_pickups_by_holiday['hday'], avg_pickups_by_holiday['pickups'], color='skyblue'
plt.title('Average Number of Pickups by Holiday Status')
plt.xlabel('Holiday Status')
plt.ylabel('Average Number of Pickups')
plt.xticks(rotation=0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



Graphical representation

In [132]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.47.09.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



4. Hourly Trends

1. What are the peak hours for Uber pickups in each borough?

In [60]:
```
df.groupby(['borough', df['pickup_dt'].dt.hour])['pickups'].sum().idxmax()
```

Out[60]:
```
('Manhattan', 23.0)
```

2. How do the number of pickups change during rush hours (e.g., 7-9 AM, 5-7 PM)?

Output

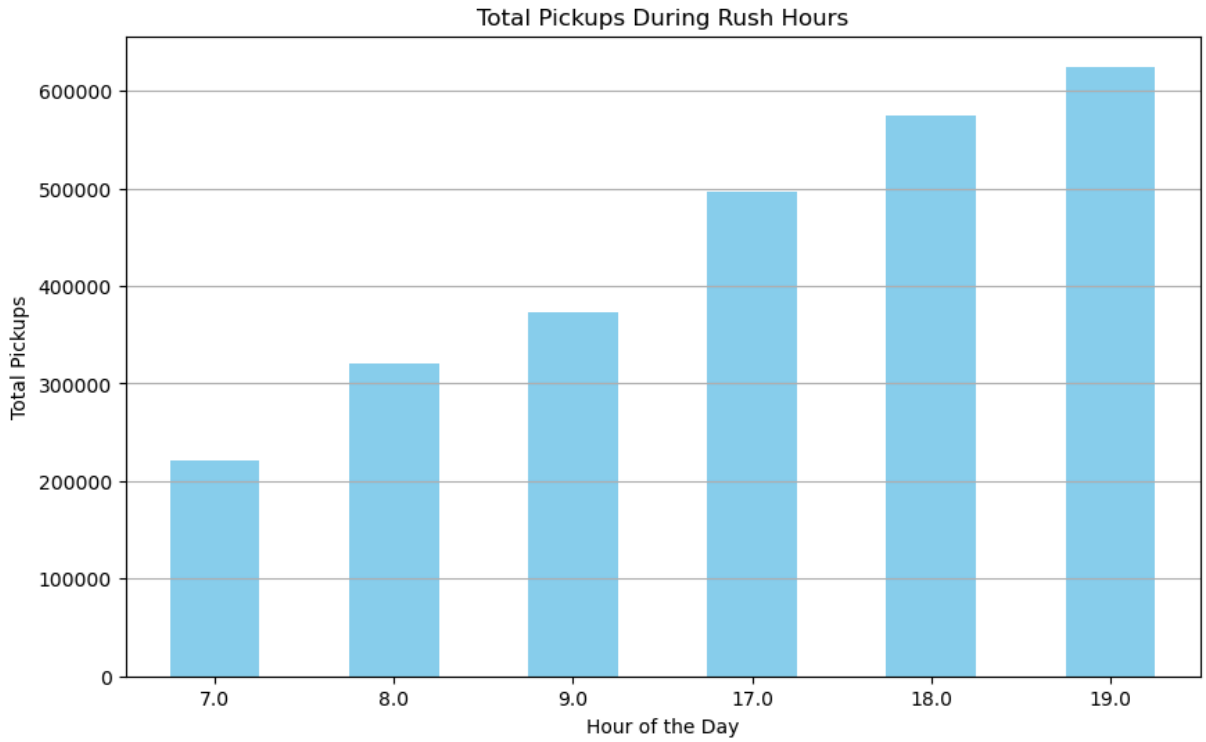
In [61]:
```
rush_hours_pickups = df[df['pickup_dt'].dt.hour.isin([7, 8, 9, 17, 18, 19])].groupby(df['p
print(rush_hours_pickups)
```

```
pickup_dt
7.0     220406.0
8.0     320601.0
9.0     372725.0
17.0    496190.0
18.0    575359.0
19.0    624742.0
Name: pickups, dtype: float64
```

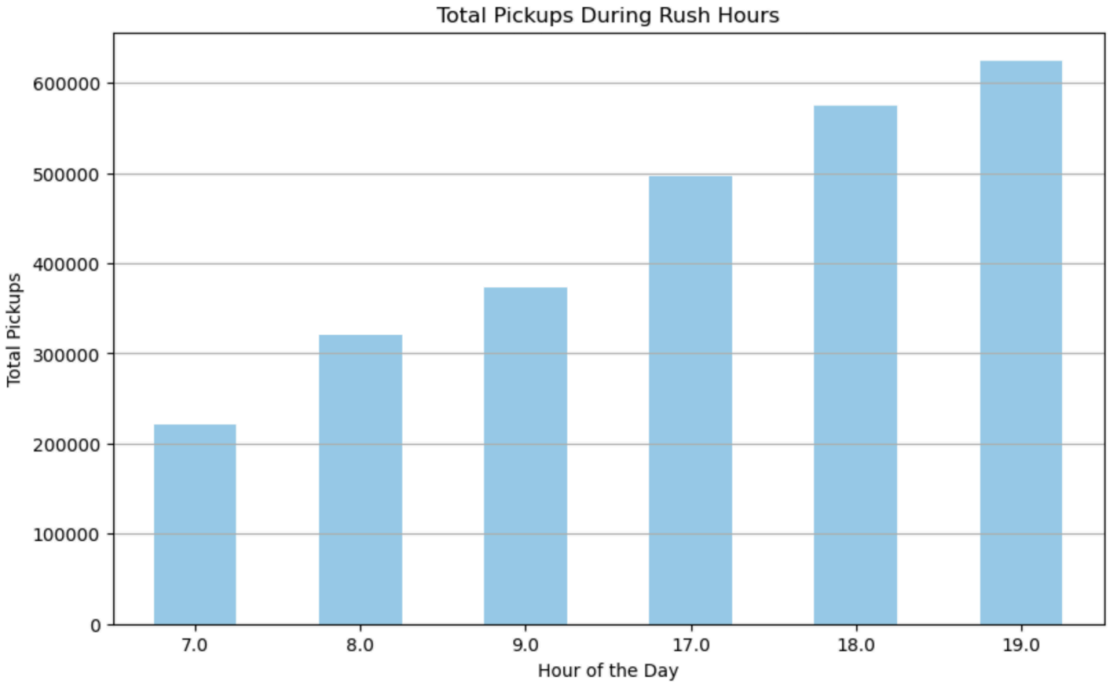
In [124]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.43.44.pn

# Displaying the image
plt.figure(dpi=60)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

```
pickup_dt
7.0      220406.0
8.0      320601.0
9.0      372725.0
17.0     496190.0
18.0     575359.0
19.0     624742.0
```

In [62]:
```
# Filter for rush hours and group by hour, then sum the pickups
rush_hours = [7, 8, 9, 17, 18, 19]
rush_hours_pickups = df[df['pickup_dt'].dt.hour.isin(rush_hours)].groupby(df['pickup_dt'].

# Plot the results
plt.figure(figsize=(10, 6))
rush_hours_pickups.plot(kind='bar', color='skyblue')
plt.title('Total Pickups During Rush Hours')
plt.xlabel('Hour of the Day')
plt.ylabel('Total Pickups')
plt.xticks(rotation=0)
plt.grid(axis='y')

# Show the plot
plt.show()
```



In [126]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.43.57.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



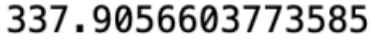
3. What is the average number of pickups during late-night hours (e.g., 12 AM - 4 AM)?

Output

In [63]:
```
late_night_avg_pickups = df[df['pickup_dt'].dt.hour.isin([0, 1, 2, 3, 4])]['pickups'].mean
print(late_night_avg_pickups)
```

```
337.9056603773585
```

In [123]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.41.51.pn

# Displaying the image
plt.figure(dpi=60)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



5. Borough Comparison

1. How do pickup trends differ between boroughs during different weather conditions?

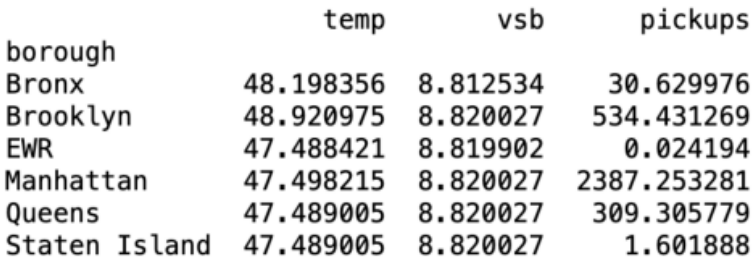
Output

Numerical represenatation

In [64]:
```
pickup_trends_weather = df.groupby(['borough'])[['temp', 'vsb', 'pickups']].mean()
print(pickup_trends_weather)
```

```
                    temp       vsb      pickups
borough
Bronx          48.198356  8.812534    30.629976
Brooklyn       48.920975  8.820027   534.431269
EWR            47.488421  8.819902     0.024194
Manhattan      47.498215  8.820027  2387.253281
Queens         47.489005  8.820027   309.305779
Staten Island  47.489005  8.820027     1.601888
```

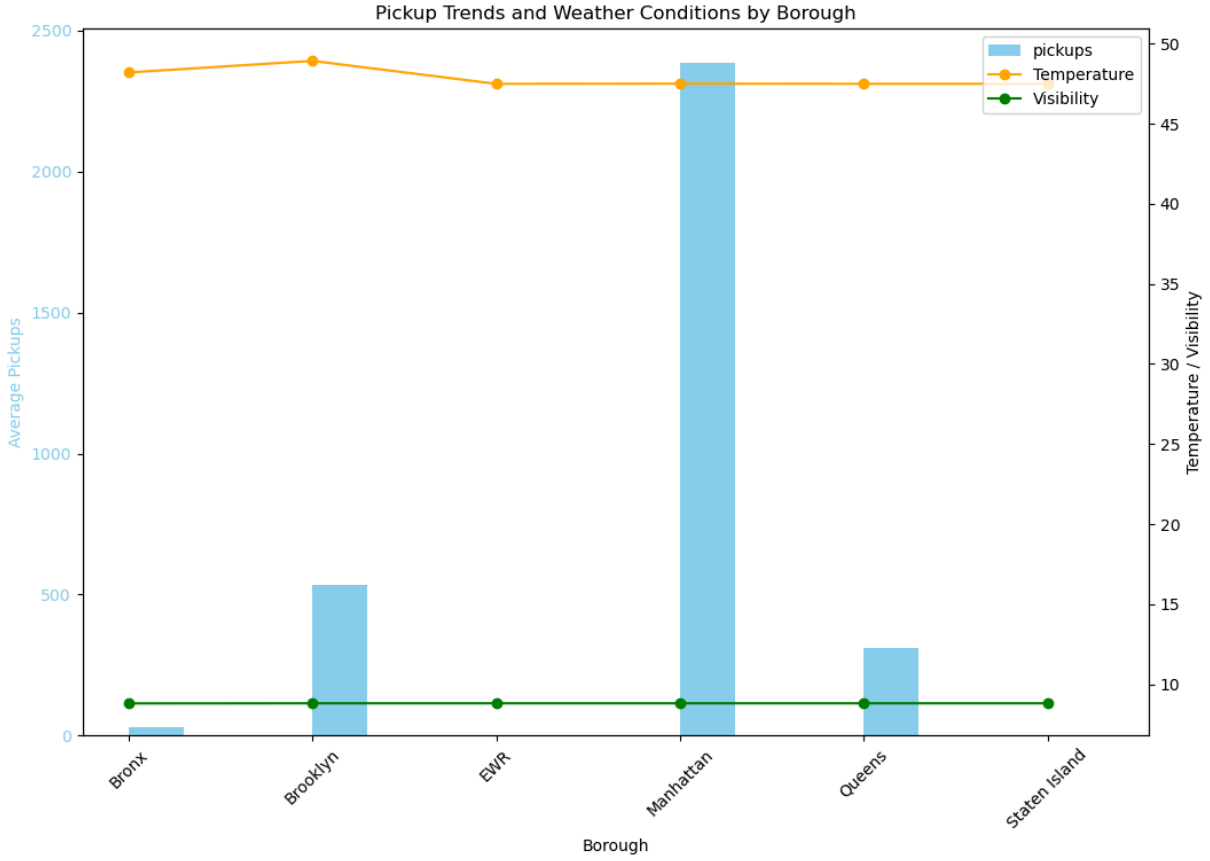
In [161]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.14.15.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



borough	temp	vsb	pickups
Bronx	48.198356	8.812534	30.629976
Brooklyn	48.920975	8.820027	534.431269
EWR	47.488421	8.819902	0.024194
Manhattan	47.498215	8.820027	2387.253281
Queens	47.489005	8.820027	309.305779
Staten Island	47.489005	8.820027	1.601888

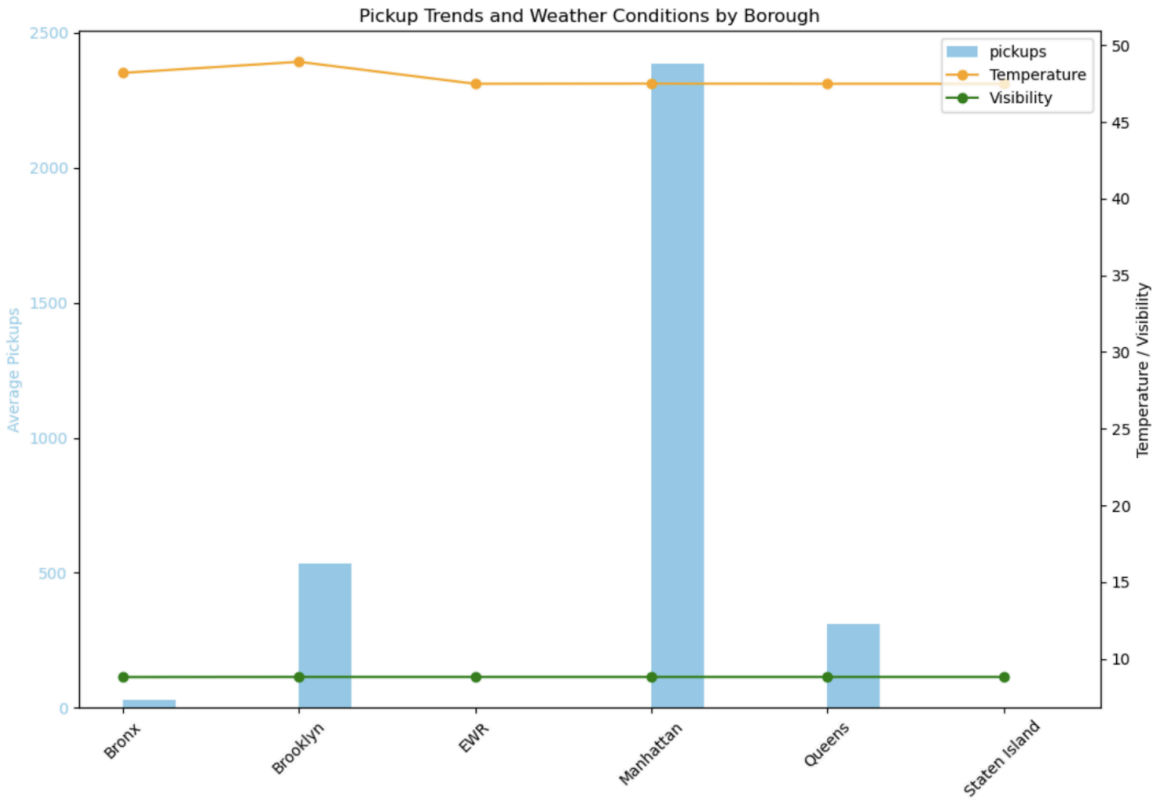
Graphical representation

In [65]:
```
#graphical representation

pickup_trends_weather = df.groupby('borough')[['temp', 'vsb', 'pickups']].mean()
fig, ax1 = plt.subplots(figsize=(12, 8))
pickup_trends_weather['pickups'].plot(kind='bar', color='skyblue', ax=ax1, position=0, widt
ax1.set_title('Pickup Trends and Weather Conditions by Borough')
ax1.set_xlabel('Borough')
ax1.set_ylabel('Average Pickups', color='skyblue')
ax1.tick_params(axis='y', labelcolor='skyblue')
ax1.set_xticklabels(pickup_trends_weather.index, rotation=45)
ax2 = ax1.twinx()
ax2.plot(pickup_trends_weather.index, pickup_trends_weather['temp'], color='orange', marker
ax2.plot(pickup_trends_weather.index, pickup_trends_weather['vsb'], color='green', marker='
ax2.set_ylabel('Temperature / Visibility', color='black')
ax2.tick_params(axis='y', labelcolor='black')
fig.legend(loc='upper right', bbox_to_anchor=(1,1), bbox_transform=ax1.transAxes)
plt.show()
```



Output

In [120]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.37.07.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



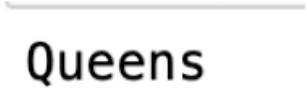
2. Which borough shows the highest increase in pickups during holidays?

In [66]:
```
oliday_increase = df.groupby(['borough', 'hday'])['pickups'].mean().unstack().apply(lambda
rint(holiday_increase)
```

```
Queens
```

Output

In [118]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.34.09.pn

# Displaying the image
plt.figure(dpi=40)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



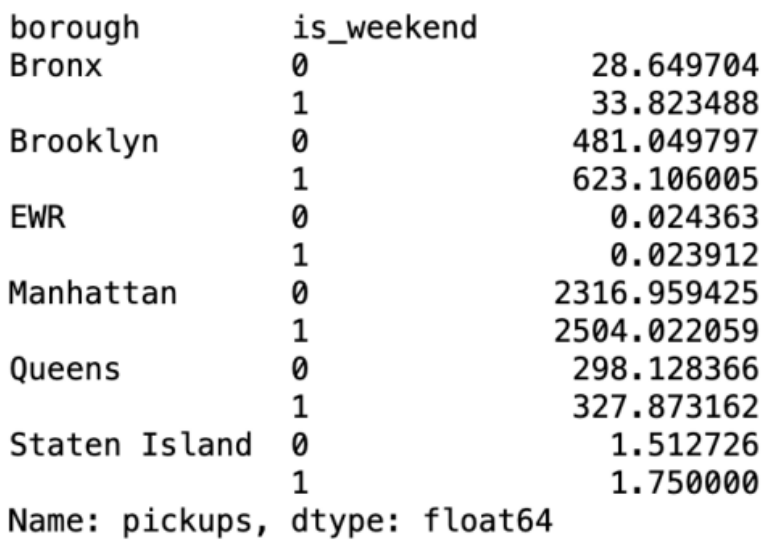
3. How does the number of pickups compare between weekdays and weekends for each borough?

Output

In [67]:
```
#Numerical representation
df['pickup_dt'] = pd.to_datetime(df['pickup_dt'], format='%m/%d/%Y %H:%M', errors='coerce'
df['weekday'] = df['pickup_dt'].dt.dayofweek
df['is_weekend'] = df['weekday'].apply(lambda x: 1 if x >= 5 else 0)
weekday_weekend_comparison = df.groupby(['borough', 'is_weekend'])['pickups'].mean()
print(weekday_weekend_comparison)
```

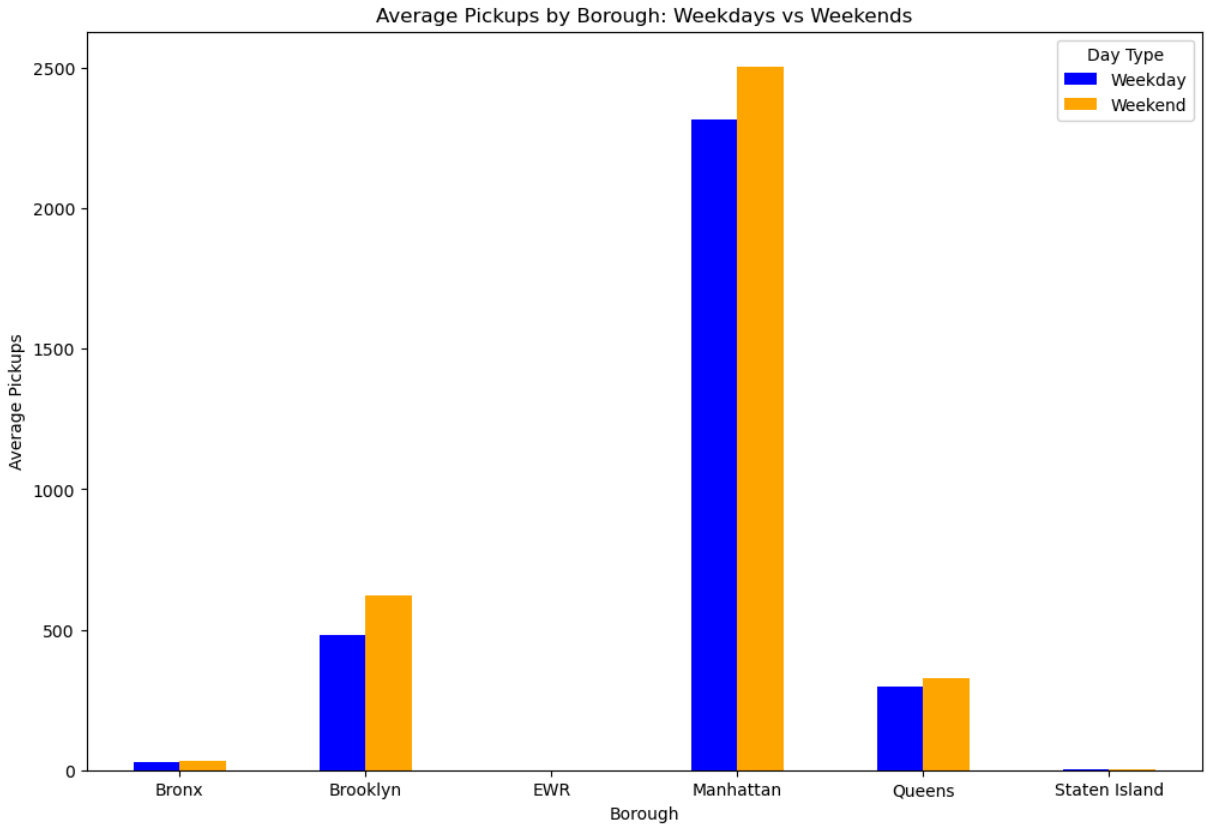
```
borough        is_weekend
Bronx          0              28.649704
               1              33.823488
Brooklyn       0             481.049797
               1             623.106005
EWR            0               0.024363
               1               0.023912
Manhattan      0            2316.959425
               1            2504.022059
Queens         0             298.128366
               1             327.873162
Staten Island  0               1.512726
               1               1.750000
Name: pickups, dtype: float64
```

In [115]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.32.41.pn

# Displaying the image
plt.figure(dpi=100)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

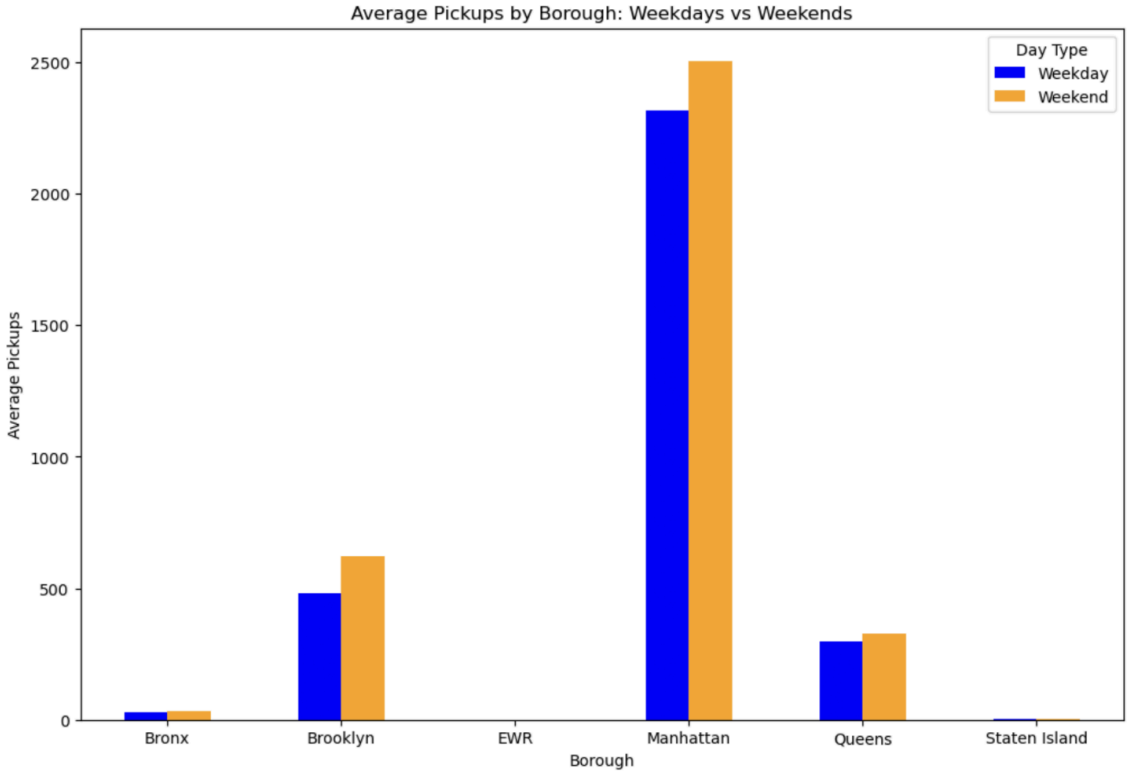


visual representation using bar graph

In [68]:
```
# visual representation using bar graph
df['pickup_dt'] = pd.to_datetime(df['pickup_dt'], format='%m/%d/%Y %H:%M', errors='coerce'
df['weekday'] = df['pickup_dt'].dt.dayofweek
df['is_weekend'] = df['weekday'].apply(lambda x: 1 if x >= 5 else 0)
weekday_weekend_comparison = df.groupby(['borough', 'is_weekend'])['pickups'].mean().unsta
ax = weekday_weekend_comparison.plot(kind='bar', figsize=(12, 8), color=['blue', 'orange']
ax.set_title('Average Pickups by Borough: Weekdays vs Weekends')
ax.set_xlabel('Borough')
ax.set_ylabel('Average Pickups')
ax.set_xticklabels(weekday_weekend_comparison.index, rotation=0)
ax.legend(['Weekday', 'Weekend'], title='Day Type')
plt.show()
```



In [113]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.30.42.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

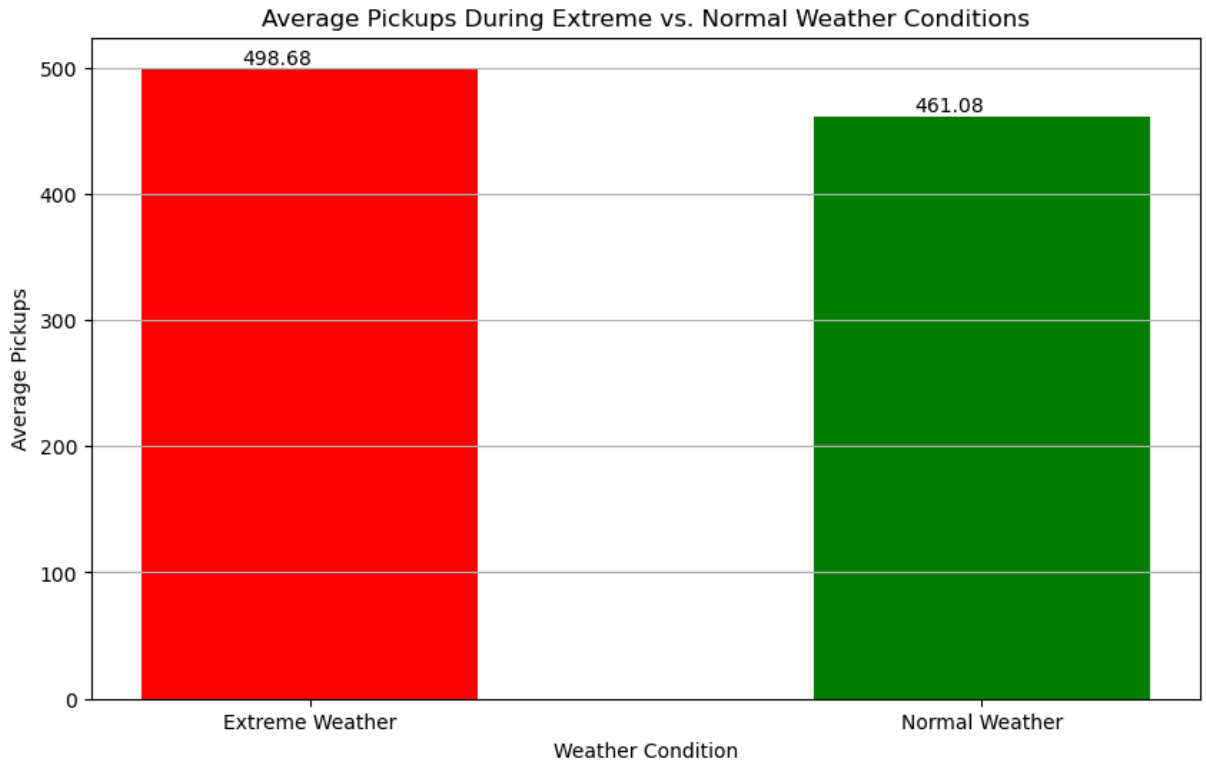


6. Weather Extremes

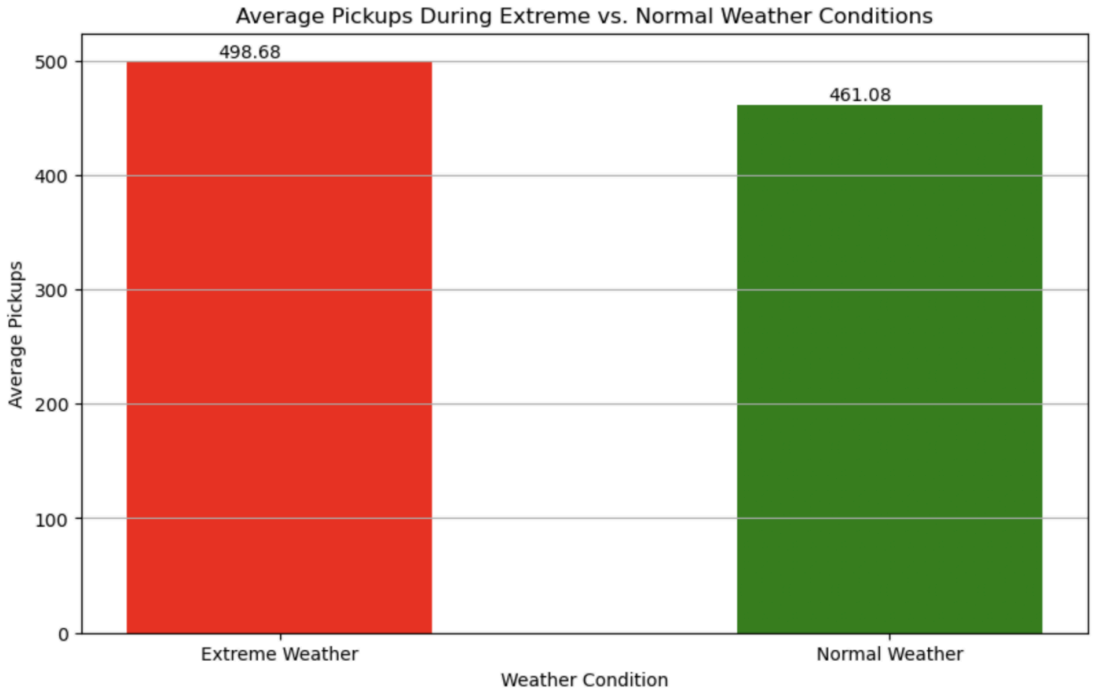
1. How do extreme weather conditions (e.g., very high or very low temperatures, heavy rainfall, snowstorms) affect the number of pickups?

Output

In [69]:
```
df['pickup_dt'] = pd.to_datetime(df['pickup_dt'], errors='coerce')
extreme_conditions = (
    (df['temp'] > 30) |
    (df['pcp01'] > 0.1)
)
extreme_weather_pickups = df[extreme_conditions]['pickups'].mean()
normal_weather_pickups = df[~extreme_conditions]['pickups'].mean()
pickup_trends = pd.DataFrame({
    'Condition': ['Extreme Weather', 'Normal Weather'],
    'Average Pickups': [extreme_weather_pickups, normal_weather_pickups]
})
plt.figure(figsize=(10, 6))
bars = plt.bar(pickup_trends['Condition'], pickup_trends['Average Pickups'], color=['red',
plt.title('Average Pickups During Extreme vs. Normal Weather Conditions')
plt.xlabel('Weather Condition')
plt.ylabel('Average Pickups')
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2 - 0.1, yval + 0.1, round(yval, 2), va='bottom
plt.grid(axis='y')
plt.show()
```



In [112]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.29.49.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



2. What is the impact of visibility less than 1 mile on the number of pickups?

Output

In [70]:
```
average_pickups_low_visibility = df[df['vsb'] < 1]['pickups'].mean()
print(average_pickups_low_visibility )
```

```
567.004555808656
```

In [110]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.28.36.pn

# Displaying the image
plt.figure(dpi=60)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

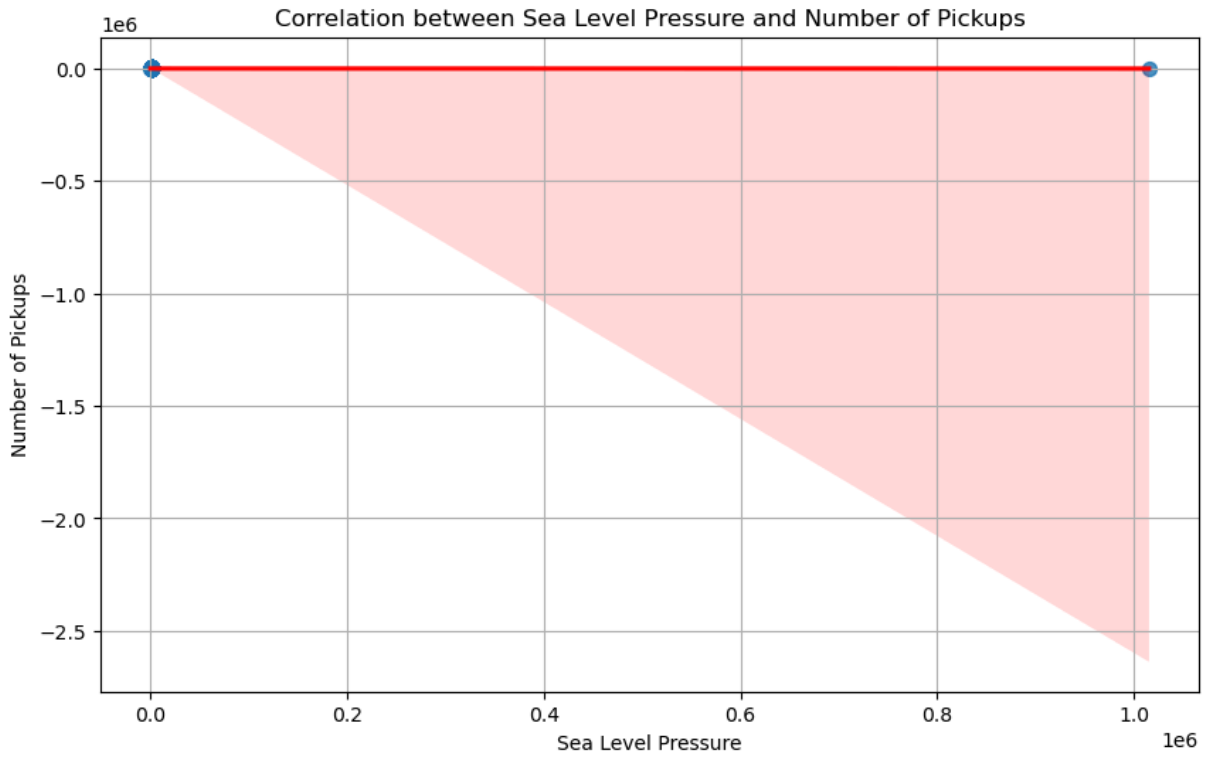
```
567.004555808656
```

7. Data Correlations

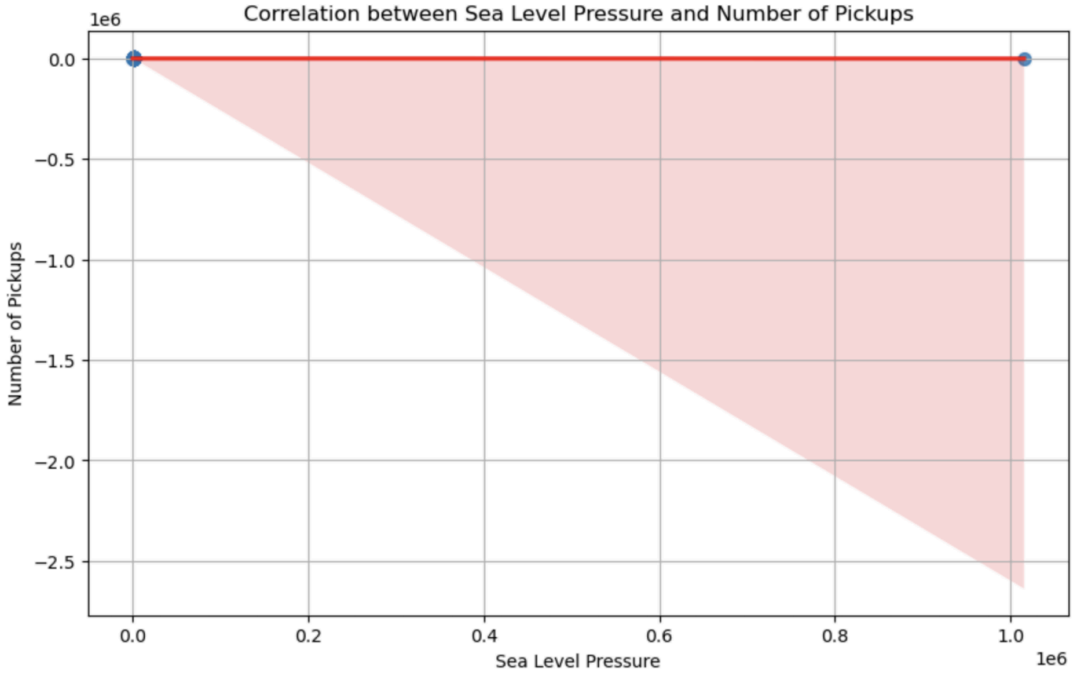
1. Is there a correlation between sea level pressure and the number of pickups?

Output

In [80]:
```
# Create scatter plot with regression line
plt.figure(figsize=(10, 6))
sns.regplot(x='slp', y='pickups', data=df, scatter_kws={'s': 50}, line_kws={'color': 'red'
plt.title('Correlation between Sea Level Pressure and Number of Pickups')
plt.xlabel('Sea Level Pressure')
plt.ylabel('Number of Pickups')
plt.grid(True)
plt.show()
```



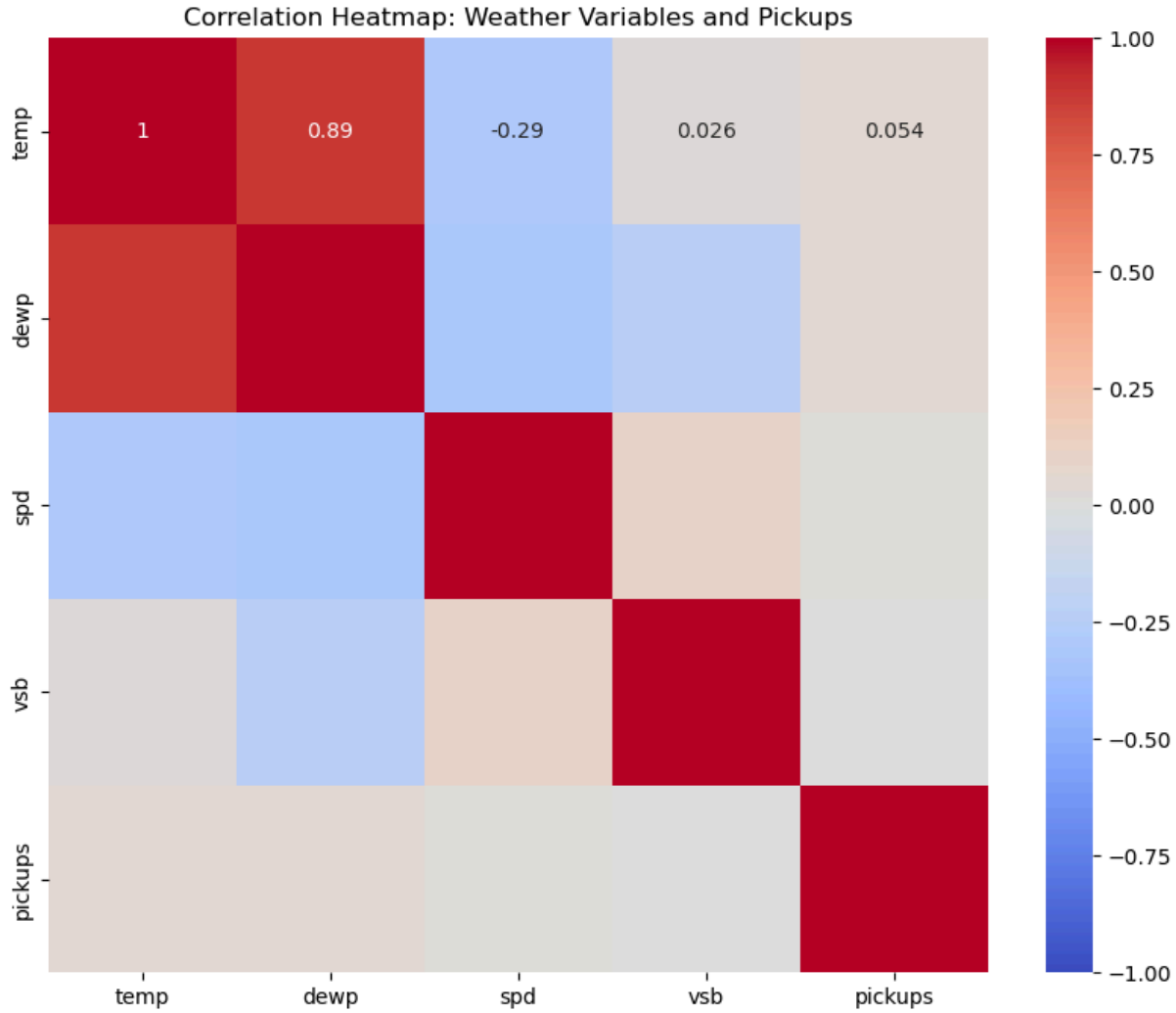
In [106]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.26.35.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



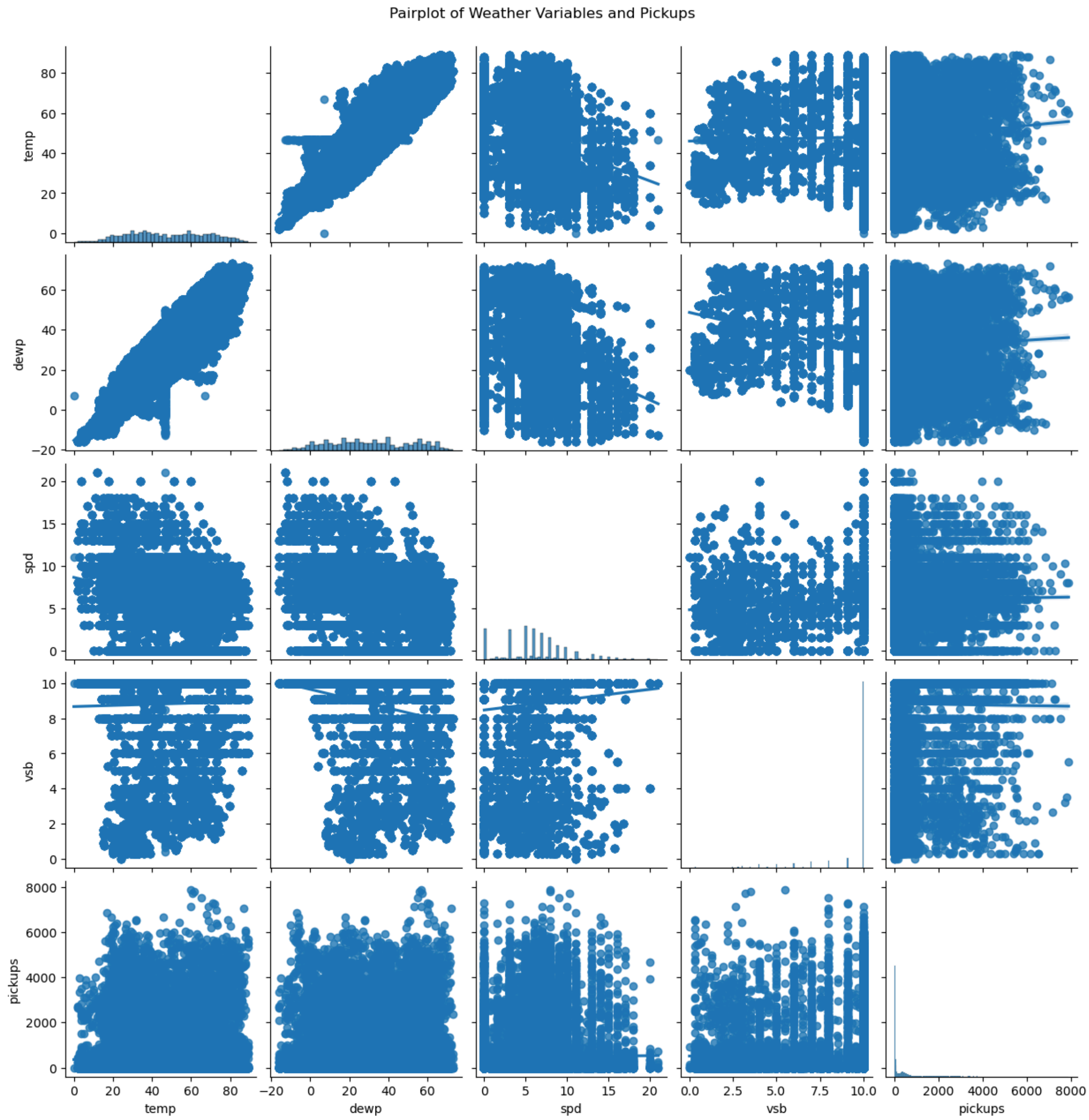
2. How do different weather variables (temperature, dew point, wind speed, visibility) collectively impact the number of pickups?

Output

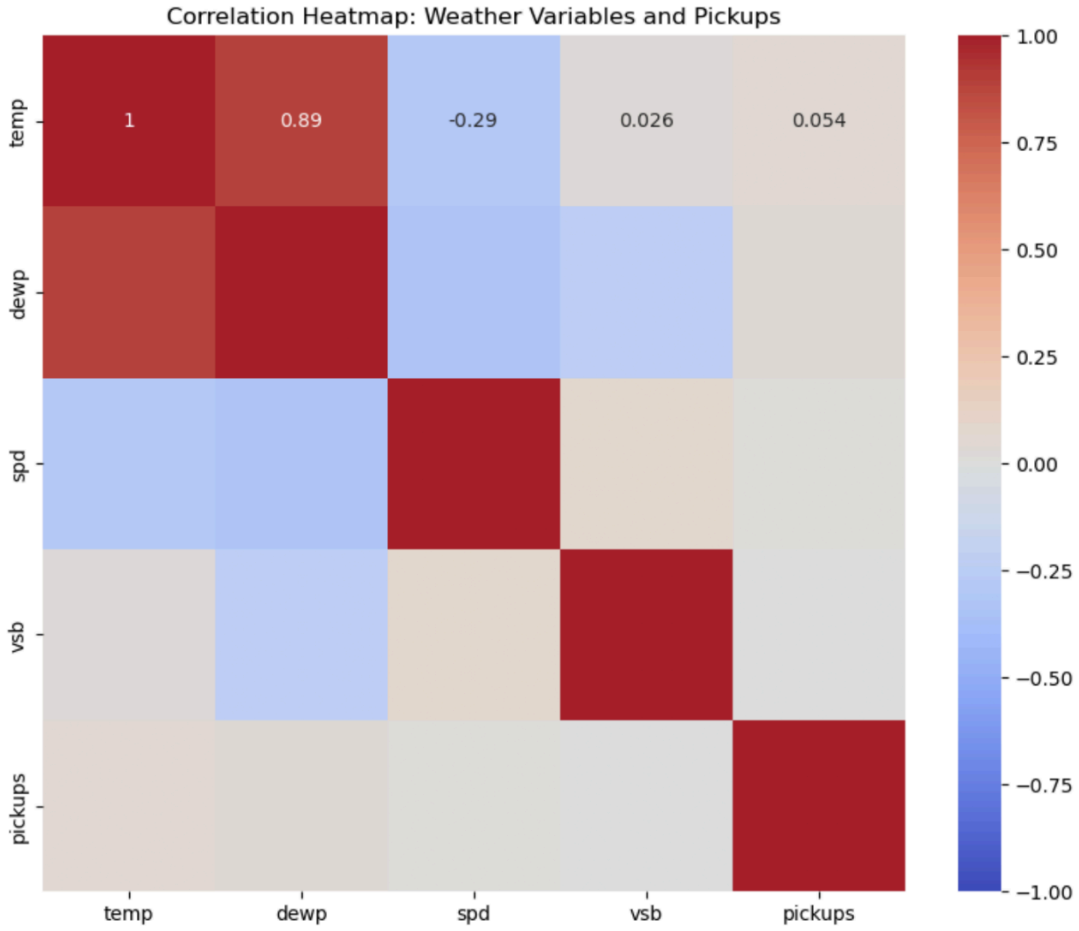
In [104]:
```
# Compute correlation matrix
corr_matrix = df[['temp', 'dewp', 'spd', 'vsb', 'pickups']].corr()

# Plot heatmap of correlations
plt.figure(figsize=(10, 8))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Heatmap: Weather Variables and Pickups')
plt.show()

# Pairplot to visualize relationships
sns.pairplot(df, vars=['temp', 'dewp', 'spd', 'vsb', 'pickups'], kind='reg')
plt.suptitle('Pairplot of Weather Variables and Pickups', y=1.02)
plt.show()
```



```
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_
inf_as_na option is deprecated and will be removed in a future version. Convert inf value
s to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_
inf_as_na option is deprecated and will be removed in a future version. Convert inf value
s to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_
inf_as_na option is deprecated and will be removed in a future version. Convert inf value
s to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_
inf_as_na option is deprecated and will be removed in a future version. Convert inf value
s to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/anaconda3/lib/python3.11/site-packages/seaborn/_oldcore.py:1119: FutureWarning: use_
inf_as_na option is deprecated and will be removed in a future version. Convert inf value
s to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

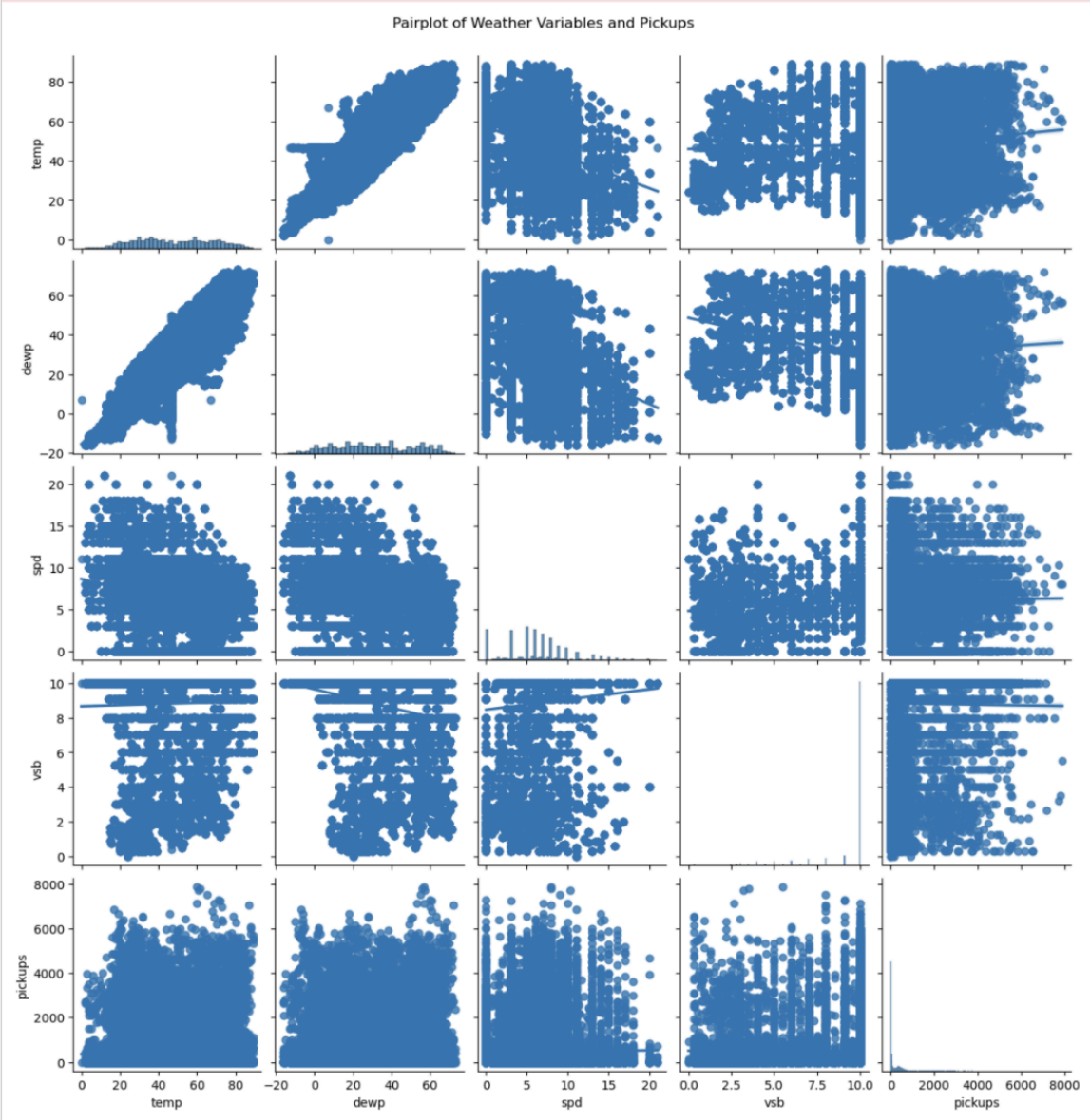


In [163]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.18.01.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```



Pairplot

In [164]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 17.19.34.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

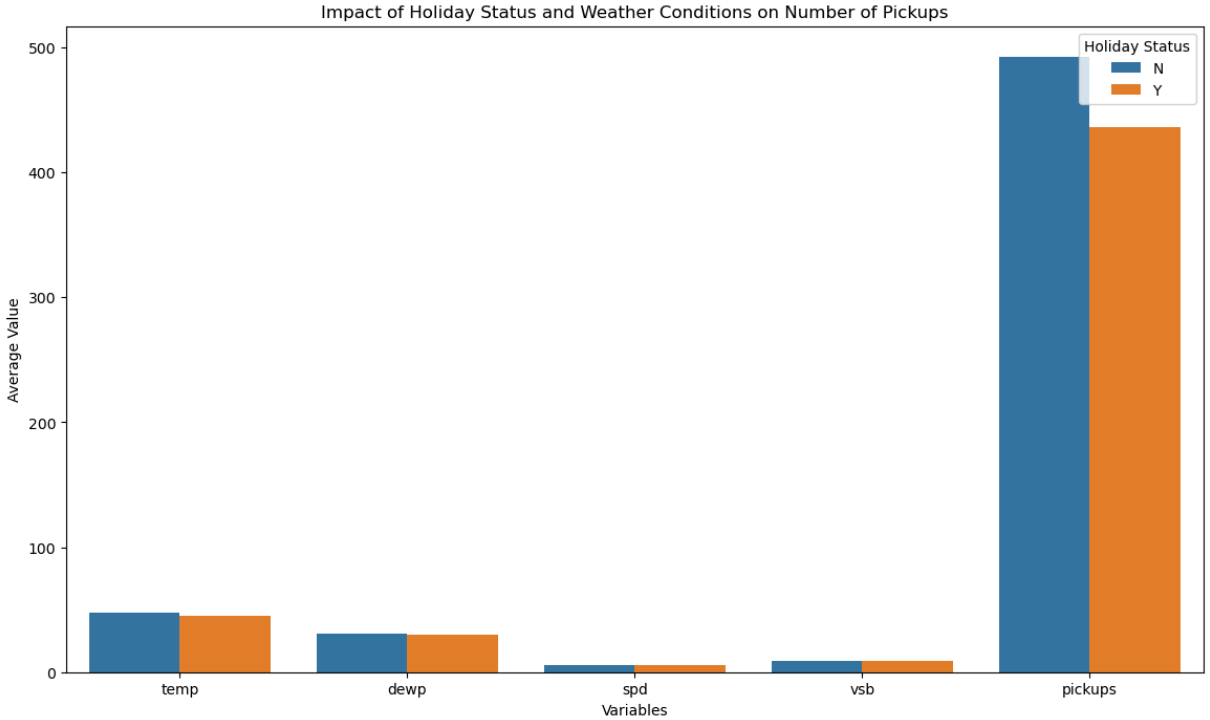


3. What is the relationship between holiday status and weather conditions on the number of pickups?

Output

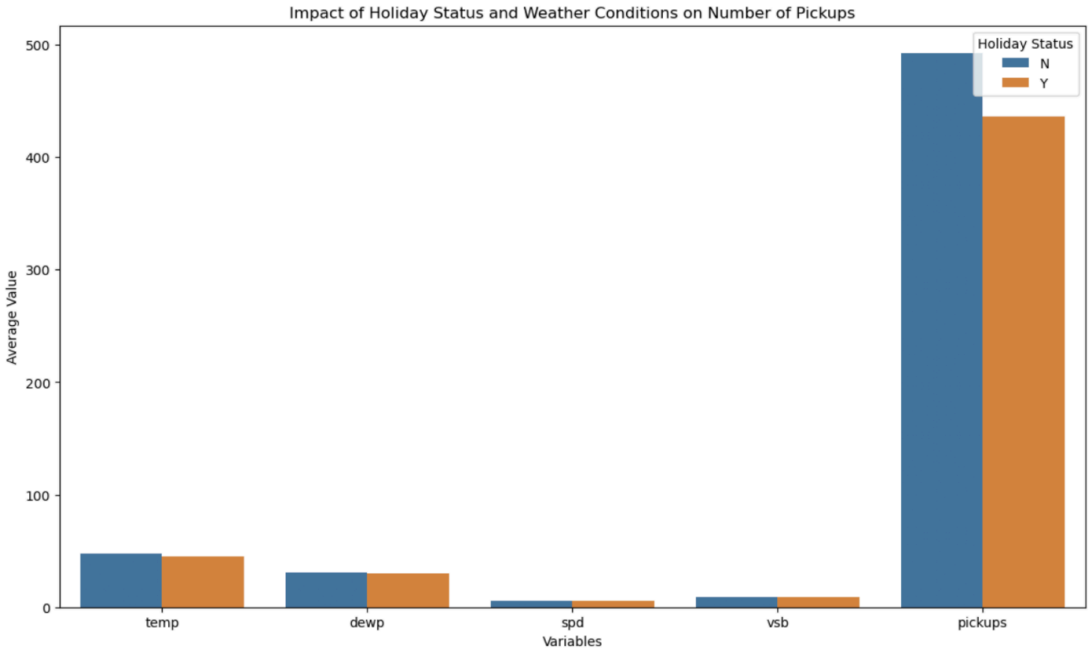
In [98]:
```
df = df.dropna(subset=['hday', 'temp', 'dewp', 'spd', 'vsb', 'pickups'])

# Group by 'hday' and calculate mean weather conditions and pickups
holiday_weather_pickups = df.groupby('hday')[['temp', 'dewp', 'spd', 'vsb', 'pickups']].me

# Print the result
print(holiday_weather_pickups)

# Melt the DataFrame for easier plotting with seaborn
melted_df = holiday_weather_pickups.melt(id_vars='hday', value_vars=['temp', 'dewp', 'spd'

# Create a bar plot
plt.figure(figsize=(14, 8))
sns.barplot(data=melted_df, x='variable', y='value', hue='hday')
plt.title('Impact of Holiday Status and Weather Conditions on Number of Pickups')
plt.ylabel('Average Value')
plt.xlabel('Variables')
plt.legend(title='Holiday Status', loc='upper right')
plt.show()
```

	hday	temp	dewp	spd	vsb	pickups
0	N	47.987781	30.855647	5.990062	8.807282	492.402752
1	Y	45.288670	30.025060	5.859869	9.088994	436.544723



In [103]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.23.17.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

	hday	temp	dewp	spd	vsb	pickups
0	N	47.987781	30.855647	5.990062	8.807282	492.402752
1	Y	45.288670	30.025060	5.859869	9.088994	436.544723

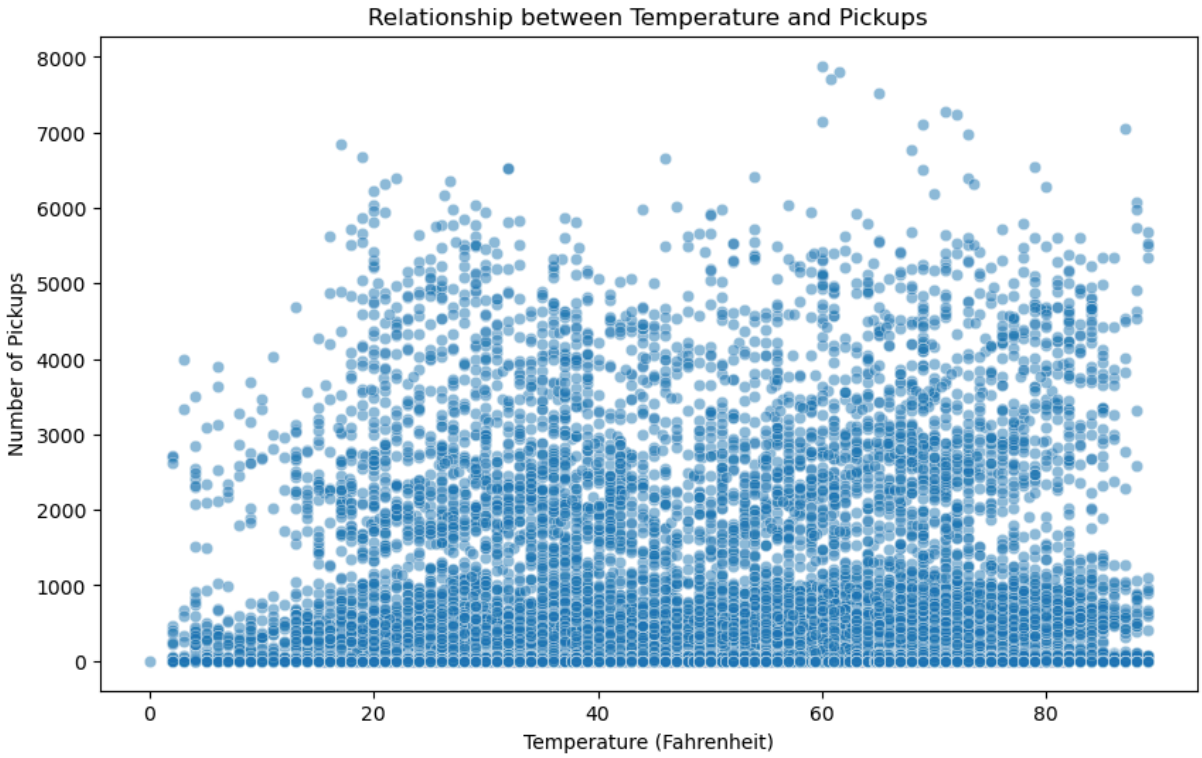


8. Growth Insights

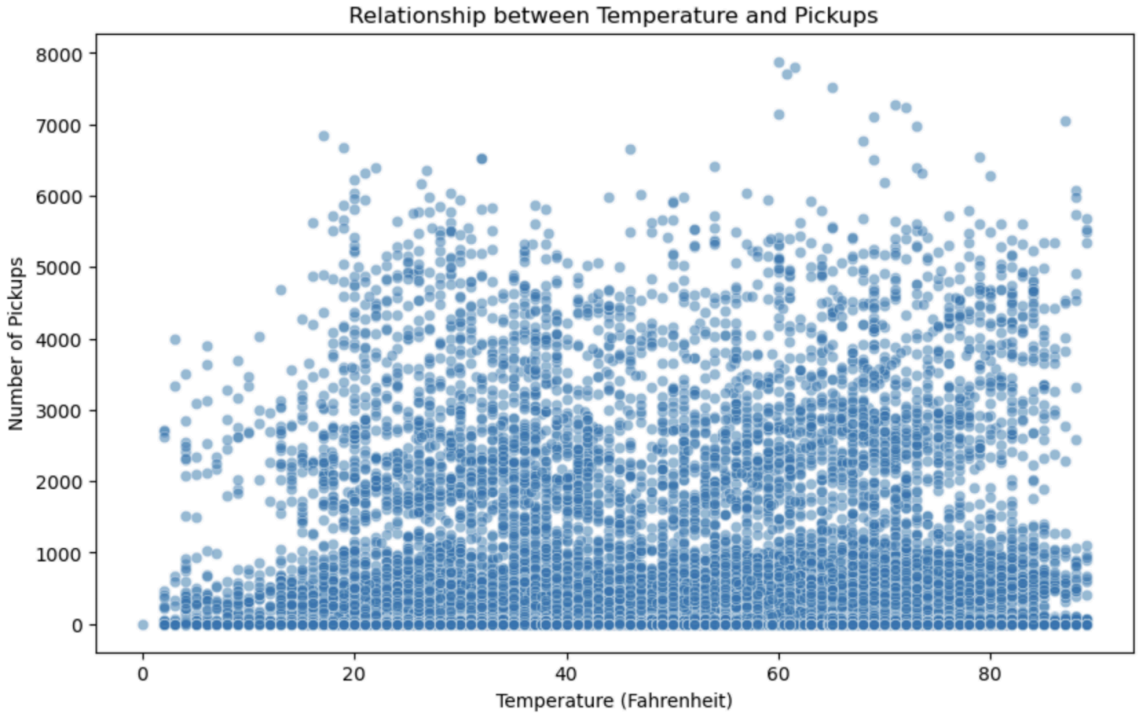
1. Which weather conditions are most favorable for Uber pickups, and how can this information be used to optimize driver availability?

Output

In [99]:
```
# Plotting the relationship between temperature and pickups
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='temp', y='pickups', alpha=0.5)
plt.title('Relationship between Temperature and Pickups')
plt.xlabel('Temperature (Fahrenheit)')
plt.ylabel('Number of Pickups')
plt.show()
```



In [102]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.22.36.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```

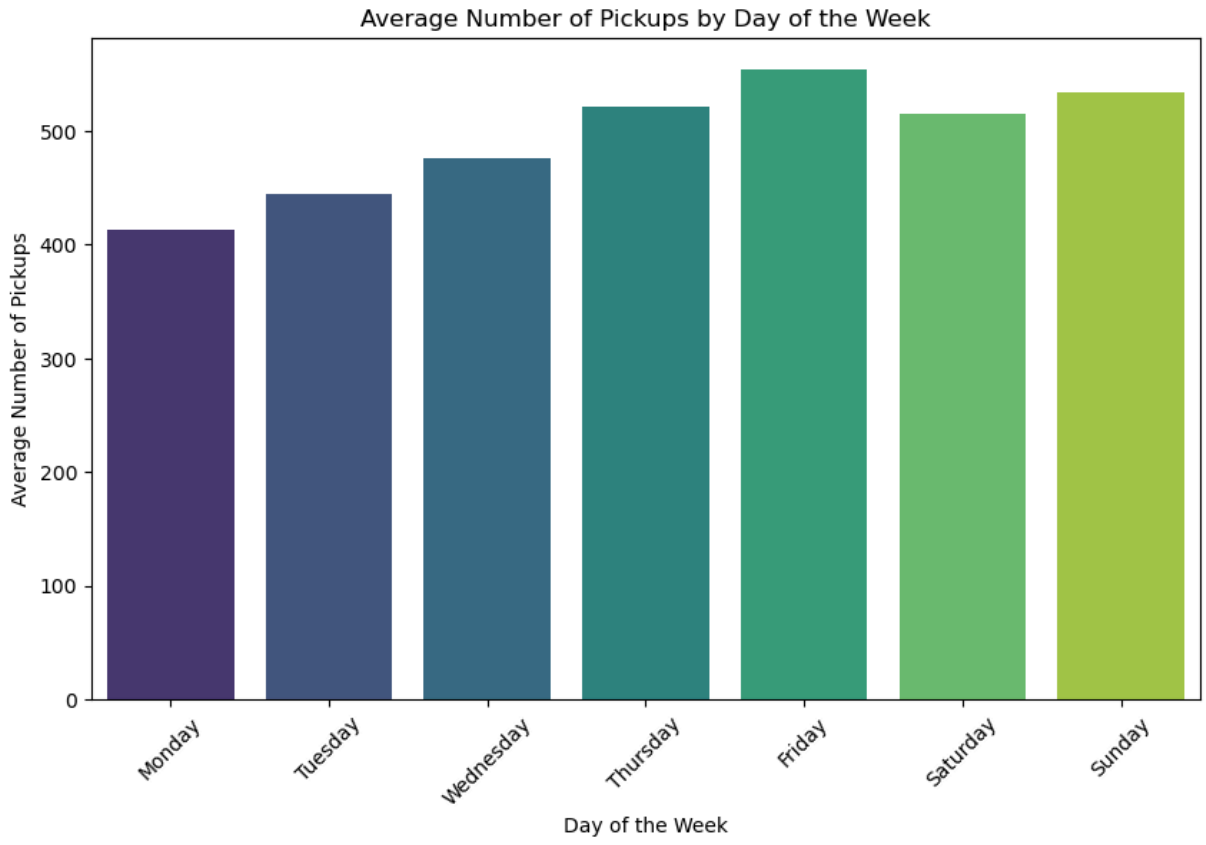


2. Based on the data, what recommendations can be made to Uber to increase pickups during low-demand periods?

Output

In [100]:
```
f['day_of_week'] = df['pickup_dt'].dt.day_name()

# Group by day of the week and calculate average pickups
vg_pickups_per_day = df.groupby('day_of_week')['pickups'].mean().reset_index()

# Sort days of the week in chronological order
ays_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
vg_pickups_per_day['day_of_week'] = pd.Categorical(avg_pickups_per_day['day_of_week'], cate
vg_pickups_per_day = avg_pickups_per_day.sort_values('day_of_week')

# Plotting the average pickups per day
lt.figure(figsize=(10, 6))
ns.barplot(data=avg_pickups_per_day, x='day_of_week', y='pickups', palette='viridis')
lt.title('Average Number of Pickups by Day of the Week')
lt.xlabel('Day of the Week')
lt.ylabel('Average Number of Pickups')
lt.xticks(rotation=45)
lt.show()
```

```
/opt/anaconda3/lib/python3.11/site-packages/seaborn/categorical.py:641: FutureWarning: Th
e default of observed=False is deprecated and will be changed to True in a future version
of pandas. Pass observed=False to retain current behavior or observed=True to adopt the f
uture default and silence this warning.
  grouped_vals = vals.groupby(grouper)
```



In [101]:
```
img = mpimg.imread('/Users/vishal/Desktop/screenshots/Screenshot 2024-06-23 at 16.21.44.pn

# Displaying the image
plt.figure(dpi=300)
plt.imshow(img)
plt.axis('off')  # Hide the axis
plt.show()
```