In [ ]:

```
   ### Task 2: some EVs have unusually high or low energy consumption. Finding outliers in the mean - Energy consumption [kWh/100 km] colum


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

#Step-1: Loading the data
data=pd.read_excel("FEV-data-Excel.xlsx")

#Step-2: Droped rows with missing values(NaN) in the energy consumption column
data=data.dropna(subset=["mean - Energy consumption [kWh/100 km]"])

#Step-3: View Result
data_cleaned=data[['Car full name', "mean - Energy consumption [kWh/100 km]"]]
data_cleaned
```

Out[ ]:

	Car full name	mean - Energy consumption [kWh/100 km]
0	Audi e-tron 55 quattro	24.45
1	Audi e-tron 50 quattro	23.80
2	Audi e-tron S quattro	27.55
3	Audi e-tron Sportback 50 quattro	23.30
4	Audi e-tron Sportback 55 quattro	23.85
5	Audi e-tron Sportback S quattro	27.20
6	BMW i3	13.10
7	BMW i3s	14.30
8	BMW iX3	18.80
10	DS DS3 Crossback e-tense	15.60
11	Honda e	17.20
12	Honda e Advance	17.50
13	Hyundai Ioniq electric	13.80
14	Hyundai Kona electric 39.2kWh	15.00
15	Hyundai Kona electric 64kWh	15.40
16	Jaguar I-Pace	21.20
17	Kia e-Niro 39.2kWh	15.30
18	Kia e-Niro 64kWh	15.90
19	Kia e-Soul 39.2kWh	15.60
20	Kia e-Soul 64kWh	15.70
21	Mazda MX-30	14.50
22	Mercedes-Benz EQC	21.85
23	Mini Cooper SE	16.75
24	Nissan Leaf	18.50
25	Nissan Leaf e+	17.10

	Car full name	mean - Energy consumption [kWh/100 km]
26	Opel Corsa-e	16.65
27	Opel Mokka-e	17.60
28	Peugeot e-208	16.40
30	Porsche Taycan 4S (Performance)	23.40
31	Porsche Taycan 4S (Performance Plus)	24.10
32	Porsche Taycan Turbo	24.85
33	Porsche Taycan Turbo S	25.10
34	Renault Zoe R110	16.50
35	Renault Zoe R135	16.50
36	Skoda Citigo-e iV	15.45
37	Smart fortwo EQ	16.35
38	Smart forfour EQ	17.00
46	Volkswagen e-up!	14.00
47	Volkswagen ID.3 Pro Performance	15.40
48	Volkswagen ID.3 Pro S	15.90
49	Volkswagen ID.4 1st	18.00
50	Citroën ë-Spacetourer (M)	25.20
51	Mercedes-Benz EQV (long)	28.20
52	Nissan e-NV200 evalia	25.90

In [ ]:

```
#Step-4: Outliers are values that are unusually high or low compared to the rest. To detect them, we use the IQR method (Interquartile Range
#Step-5: Q1 gets the 25th percentile (Q1)- 25% of the cars use less energy than this and assume it as the “lower edge” of the typical range
Q1= data_cleaned["mean - Energy consumption [kWh/100 km]"].quantile(0.25)

#Step-6: Q3 gets the 75th percentile (Q3)- 75% of the cars use less energy than this and assume it as the “upper edge” of the typical range
Q3=data_cleaned["mean - Energy consumption [kWh/100 km]"].quantile(0.75)

#Step-7: IQR calculates the Interquartile Range, or the middle 50% of the data. It's the range between Q1 and Q3 i.e. "normal" energy usage
IQR=Q3-Q1
```

```
#Step-8: These lines defined below what counts as an outlier.
        #Anything below lower_bound is too low (outlier).
        #Anything above upper_bound is too high (outlier).
        #And 1.5 is common default value.
lower_bound= Q1-1.5*IQR
upper_bound= Q3+1.5*IQR


#Step-9: Filter outliers
outliers=data_cleaned[(data_cleaned["mean - Energy consumption [kWh/100 km]"]<lower_bound) |
(data_cleaned["mean - Energy consumption [kWh/100 km]"]>upper_bound)]

#Step-10: Show the outliers
print("Outliers in Energy Consumption:")
print(outliers[['Car full name', 'mean - Energy consumption [kWh/100 km]']])
print(f"\nTotal outliers found: {len(outliers)}")
```

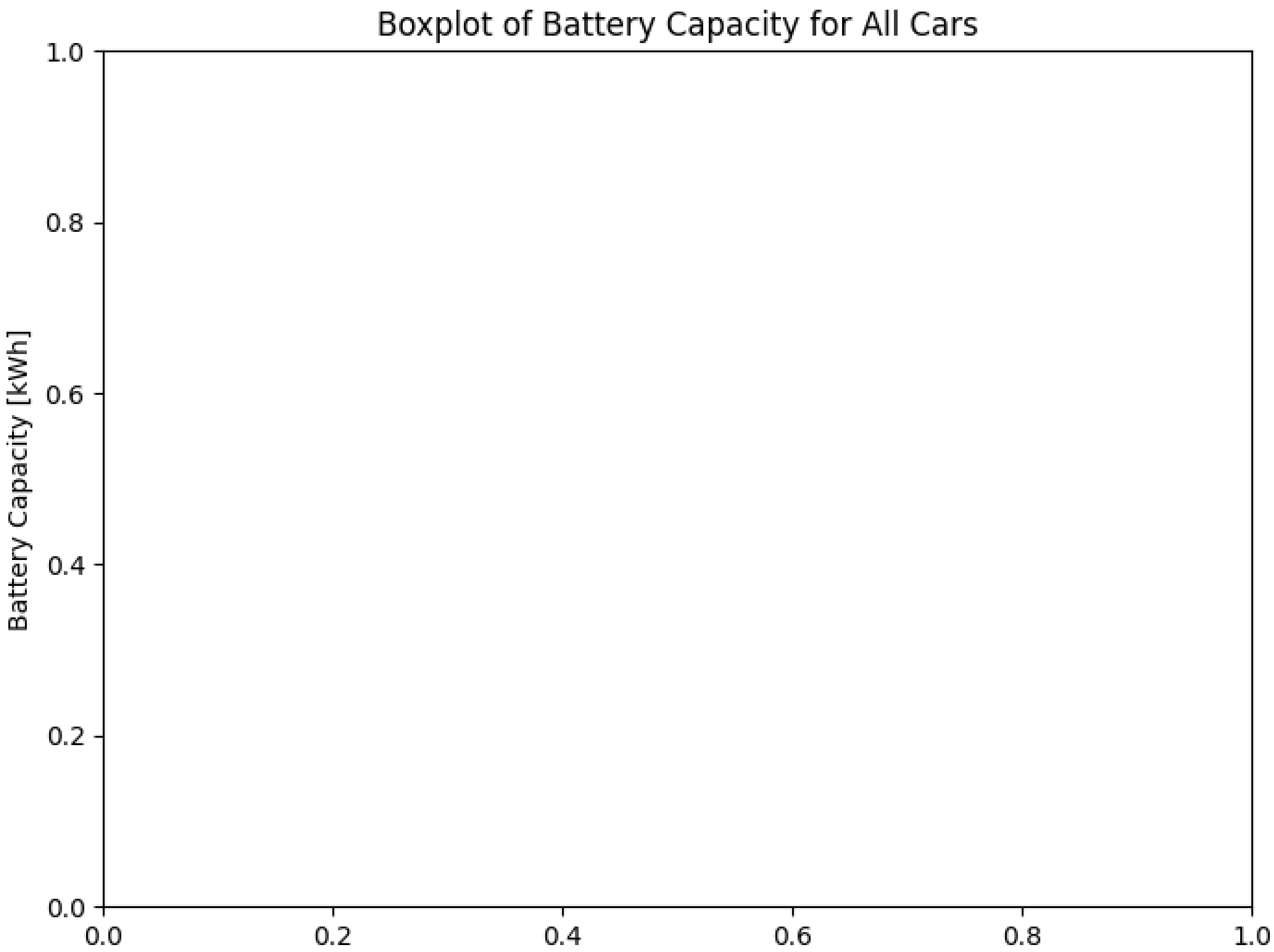
```
Outliers in Energy Consumption:
Empty DataFrame
Columns: [Car full name, mean - Energy consumption [kWh/100 km]]
Index: []

Total outliers found: 0
```

In [ ]:

```
#Step-11: Drawed the box plot to show ouliers but there are no outliers as such

plt.figure(figsize=(8, 6))
sns.boxplot(y=outliers['mean - Energy consumption [kWh/100 km]'])
plt.title('Boxplot of Battery Capacity for All Cars')
plt.ylabel('Battery Capacity [kWh]')
plt.show()
```


Boxplot of Battery Capacity for All Cars
Battery Capacity [kWh]
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: