In [4]:

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

In [6]:

```
# Loading the dataset
df = pd.read_excel("FEV-data-Excel.xlsx")
df.head()
```

Out[6]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]	Range (WLTP) [km]	...	Permissable gross weight [kg]	Maximum load capacity [kg]	Number of seats
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0	438	...	3130.0	640.0	5
1	Audi e-tron 50 quattro	Audi	e-tron 50 quattro	308400	313	540	disc (front + rear)	4WD	71.0	340	...	3040.0	670.0	5
2	Audi e-tron S quattro	Audi	e-tron S quattro	414900	503	973	disc (front + rear)	4WD	95.0	364	...	3130.0	565.0	5
3	Audi e-tron Sportback 50 quattro	Audi	e-tron Sportback 50 quattro	319700	313	540	disc (front + rear)	4WD	71.0	346	...	3040.0	640.0	5
4	Audi e-tron Sportback 55 quattro	Audi	e-tron Sportback 55 quattro	357000	360	664	disc (front + rear)	4WD	95.0	447	...	3130.0	670.0	5

5 rows × 25 columns

TASK 1: budget of 350,000 PLN and wants an EV with a minimum range of 400 km.

- filter out EVs that meet these criteria.

In [7]:

```
filtered_df = df[(df["Minimal price (gross) [PLN]"] <= 350000) & (df["Range (WLTP) [km]"] >= 400)]
filtered_df.head()
```

Out[7]:

	Car full name	Make	Model	Minimal price (gross) [PLN]	Engine power [KM]	Maximum torque [Nm]	Type of brakes	Drive type	Battery capacity [kWh]	Range (WLTP) [km]	...	Permissable gross weight [kg]	Maximum load capacity [kg]	Number of seats
0	Audi e-tron 55 quattro	Audi	e-tron 55 quattro	345700	360	664	disc (front + rear)	4WD	95.0	438	...	3130.0	640.0	5
8	BMW iX3	BMW	iX3	282900	286	400	disc (front + rear)	2WD (rear)	80.0	460	...	2725.0	540.0	5
15	Hyundai Kona electric 64kWh	Hyundai	Kona electric 64kWh	178400	204	395	disc (front + rear)	2WD (front)	64.0	449	...	2170.0	485.0	5
18	Kia e-Niro 64kWh	Kia	e-Niro 64kWh	167990	204	395	disc (front + rear)	2WD (front)	64.0	455	...	2230.0	493.0	5
20	Kia e-Soul 64kWh	Kia	e-Soul 64kWh	160990	204	395	disc (front + rear)	2WD (front)	64.0	452	...	1682.0	498.0	5

5 rows × 25 columns

- Group them by the manufacturer (Make)

In [10]:

```
grouped_by_make = filtered_df.groupby("Make")
grouped_by_make.size()
```

Out[10]:

```
Make
Audi             1
BMW              1
Hyundai          1
Kia              2
Mercedes-Benz    1
Tesla            3
Volkswagen       3
dtype: int64
```

- The average battery capacity for each manufacturer.

In [11]:

```
avg_battery_by_make = grouped_by_make["Battery capacity [kWh]"].mean()
avg_battery_by_make.sort_values(ascending=False)
```

Out[11]:

```
Make
Audi             95.000000
BMW              80.000000
Mercedes-Benz    80.000000
Volkswagen       70.666667
Tesla            68.000000
Hyundai          64.000000
Kia              64.000000
Name: Battery capacity [kWh], dtype: float64
```

Analysis

- Only a small number of EV cars met price and range criteria.
- Kia, Tesla and Volkswagen offered more option compare to others.
- By battery capacity Audi delivers the highest mean of 95KWH.
- In this findings Audi gives a better range but the cost is much higher compare to other cars.

Task 2:some EVs have unusually high or low energy consumption.

- outliers in the mean- Energy consumption [kWh/100 km] column.

In [13]:

```
col = 'mean - Energy consumption [kWh/100 km]'

# Calculate Q1, Q3, IQR
Q1 = df[col].quantile(0.25)
Q3 = df[col].quantile(0.75)
IQR = Q3 - Q1

# Outlier limits
lower_limit = Q1 - 1.5 * IQR
upper_limit = Q3 + 1.5 * IQR

# Filter outliers
outliers = df[(df[col] < lower_limit) | (df[col] > upper_limit)]

# result
print("Outlier EVs based on energy consumption:\n")
print(outliers[['Car full name', col]])

# result in table
outliers[['Car full name', col]]



# here i used Interquartile Range(IQR) to find out outliners, according to dataset there is no vehicle that con
```

```
Outlier EVs based on energy consumption:

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

Out[13]:

Car full name	mean - Energy consumption [kWh/100 km]

Analysis

- Here i used Interquartile Range(IQR) to find out outliners, according to dataset there is no vehicle that consume nor high nor too low.
- I am getting Empty dataframe that means no values in the dataset that align with this dataset.
- I cross cheked by viewing the dataset but i dont find any values in this conditions.

TASK 3: strong relationship between battery capacity and range.

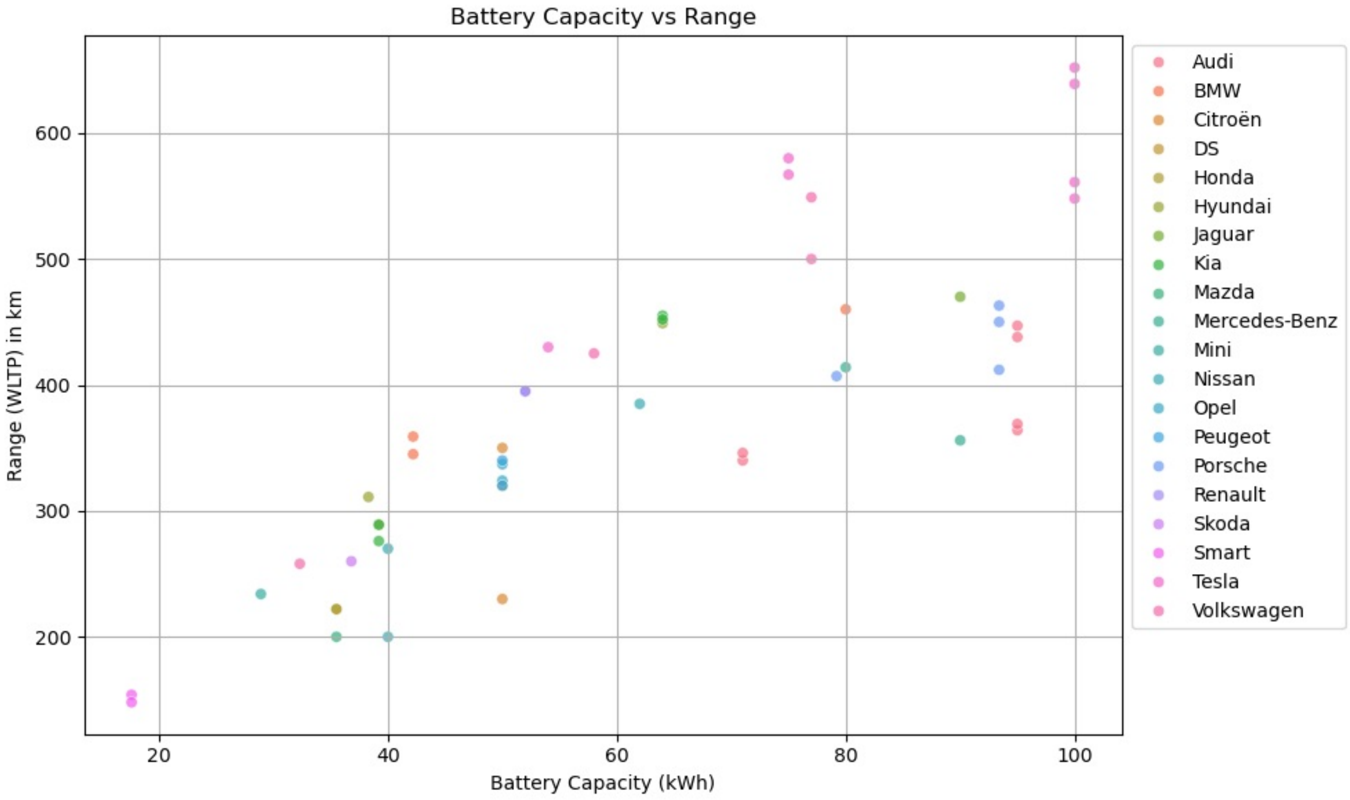
- suitable plot to visualize.

In [14]:

```
# plot size
plt.figure(figsize=(10, 6))

# scatter plot
sns.scatterplot(
    data=df,
    x='Battery capacity [kWh]',
    y='Range (WLTP) [km]',
    hue='Make',
    alpha=0.7
)

# labels and title
plt.title('Battery Capacity vs Range')
plt.xlabel('Battery Capacity (kWh)')
plt.ylabel('Range (WLTP) in km')
plt.grid(True)
plt.legend(loc='best', bbox_to_anchor=(1, 1))
plt.tight_layout()
plt.show()
```



- Highlighting insights

- Here we can see a battery capacity and range are indicate strong relation.
- Companies like smart, tesla, volkswagen have higher range compare to other companies while battery capacity exceeds 50 kWh.
- when the battery capacity is in 20-40 majority of all companies fall in range of 200-300 km.
- Efficiency varry by brands not all car gives same range.
- Cars with 60-8- battery capacity have the sweet spot customers because og the range which is slightly less than 80-100 battery capacity cars.
- for majority of car battery capacity they still stuck at a range of 400-500.

TASK 4: EV recommendation class

In [17]:

```
class EVRecommender:
    def __init__(self, data):
        self.data = data

    def recommend(self):
        try:
            # Get user inputs inside the notebook
            budget = int(input(" Enter your budget (PLN): "))
            min_range = int(input(" Enter minimum range required (km): "))
            min_battery = int(input(" Enter minimum battery capacity (kWh): "))
        except ValueError:
            print( "Invalid input. Please enter numbers only.")
            return

        # Apply filters
        filtered = self.data[
            (self.data['Minimal price (gross) [PLN]'] <= budget) &
            (self.data['Range (WLTP) [km]'] >= min_range) &
            (self.data['Battery capacity [kWh]'] >= min_battery)
        ]

        # Check and display result
        if filtered.empty:
            print(" No EVs match your criteria.")
        else:
            top_3 = filtered.sort_values(by='Range (WLTP) [km]', ascending=False).head(3)
            print("\n Top EV Recommendations:\n")
            display(top_3[['Car full name', 'Make', 'Minimal price (gross) [PLN]',
                          'Range (WLTP) [km]', 'Battery capacity [kWh]']])

recommender = EVRecommender(df)
recommender.recommend()
```

```
 Top EV Recommendations:
```

	Car full name	Make	Minimal price (gross) [PLN]	Range (WLTP) [km]	Battery capacity [kWh]
35	Renault Zoe R135	Renault	142900	395	52.0
34	Renault Zoe R110	Renault	135900	395	52.0
9	Citroën ë-C4	Citroën	125000	350	50.0

Analysis

- I have developed an interactive EV recommendation system using class EVRecommender.
- Once user provide the input it will generate top 3 cars which are in those conditions.
- When the input of battery capacity reduce the more option we have on the budget cars.
- It simplifies the user selection.

Task 5: Inferential Statistics– Hypothesis Testing:

In [45]:

```
from scipy.stats import ttest_ind

df.columns = df.columns.str.strip().str.lower()

# engine power for Tesla and Audi
tesla_power = df[df['make'] == 'Tesla']['engine power [km]'].dropna()
audi_power = df[df['make'] == 'Audi']['engine power [km]'].dropna()

# means
print(f" Tesla average engine power: {round(tesla_power.mean(), 2)} KM")
print(f" Audi average engine power: {round(audi_power.mean(), 2)} KM")

# Perform t-test
t_stat, p_value = ttest_ind(tesla_power, audi_power, equal_var=False)

# test results
print("\n Two-Sample T-Test Result:")
print(f"T-Statistic: {round(t_stat, 2)}")
print(f"P-Value: {round(p_value, 4)}")

# result
if p_value < 0.05:
    print("\n Conclusion: There is a **significant difference** in average engine power between Tesla and Audi
else:
    print("\n Conclusion: There is **no significant difference** in average engine power between Tesla and Audi
```

```
 Tesla average engine power: 533.0 KM
 Audi average engine power: 392.0 KM

 Two-Sample T-Test Result:
T-Statistic: 1.79
P-Value: 0.1068

✖ Conclusion: There is **no significant difference** in average engine power between Tesla and Audi EVs.
```

- Highlighting insights and recommendation

- we should educate buyers on Real-World Range vs Battery Size.
- Highlighting energy efficient models.
- Offering More EVs Under 350,000 PLN with High Range.
- Optimize Battery-to-Range Efficiency.
- we should use Data-Driven EV Recommendation Systems.
- investigate Underperforming Models and improve its features and efficiency.

TASK 6: video explanation Link

In [ ]:

```
https://drive.google.com/file/d/1WOtS6HyG7uU-4XIDYJpMtHUpf4yKis79/view?usp=sharing
```