

```
In [1]: import pandas as pd  
print("Setup successful ✓", pd.__version__)
```

Setup successful ✓ 2.3.2

```
In [ ]: cursor = conn.cursor()  
cursor.execute("SELECT DATABASE();")  
result = cursor.fetchone()  
print("You are connected to database:", result[0])  
  
cursor.close()  
conn.close()
```

You are connected to database: uber_analytics

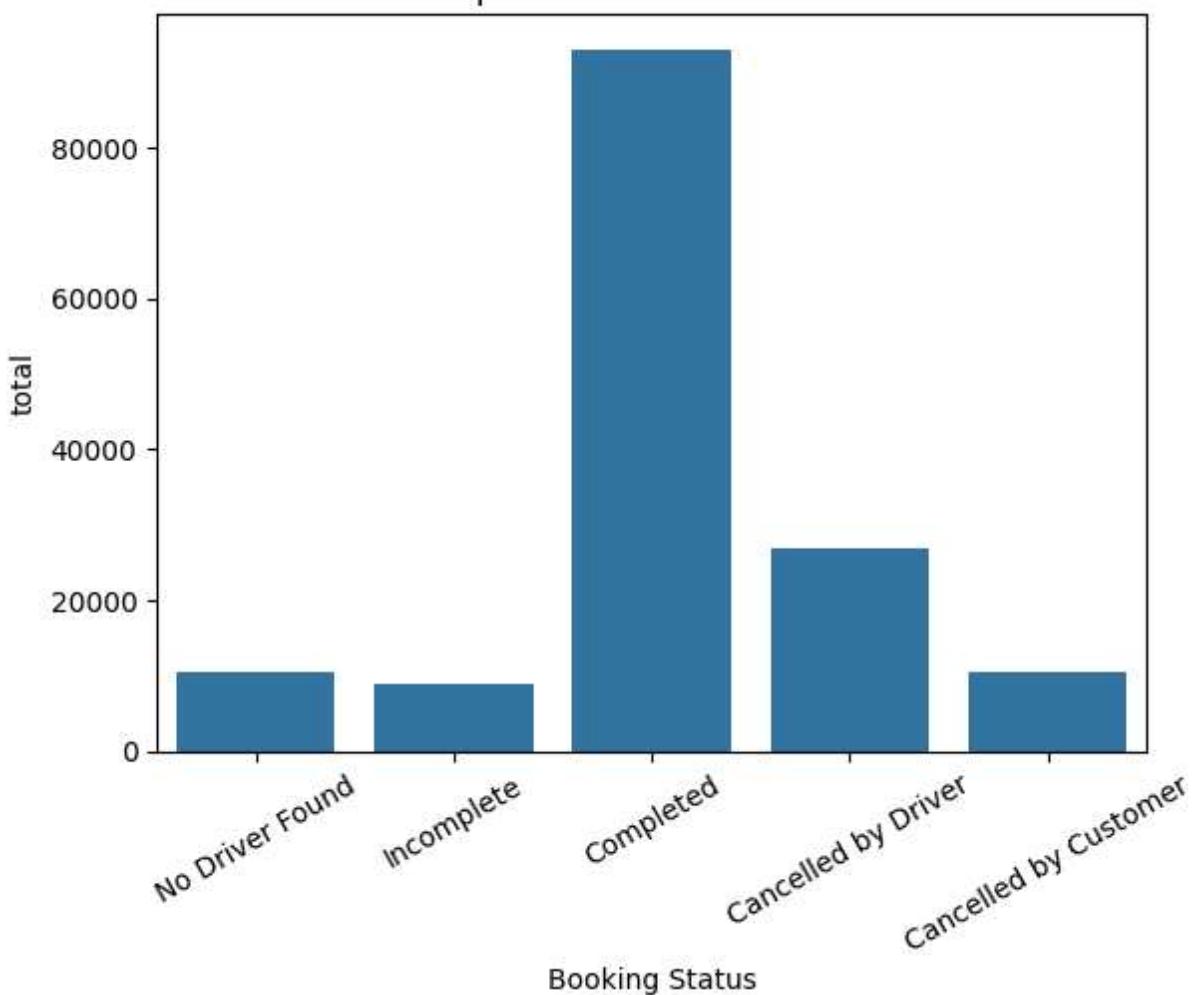
```
In [1]: import pandas as pd  
from sqlalchemy import create_engine  
import matplotlib.pyplot as plt  
import seaborn as sns  
engine = create_engine("mysql+pymysql://root:Poojadeep@231@localhost/uber_analytics")
```

```
In [10]: import pandas as pd  
from sqlalchemy import create_engine  
from urllib.parse import quote_plus  
  
# credentials  
user = "root"  
password = "Poojadeep@231"  
host = "localhost"  
database = "uber_analytics"  
  
# encode password safely  
password_enc = quote_plus(password)  
  
# connect to MySQL  
engine = create_engine(f"mysql+pymysql://{{user}}:{{password_enc}}@{{host}}/{{database}}")  
  
# test query  
df = pd.read_sql("SELECT COUNT(*) AS total_rides FROM ncr_ride_bookings;", engine)  
print(df)
```

total_rides
0 150000

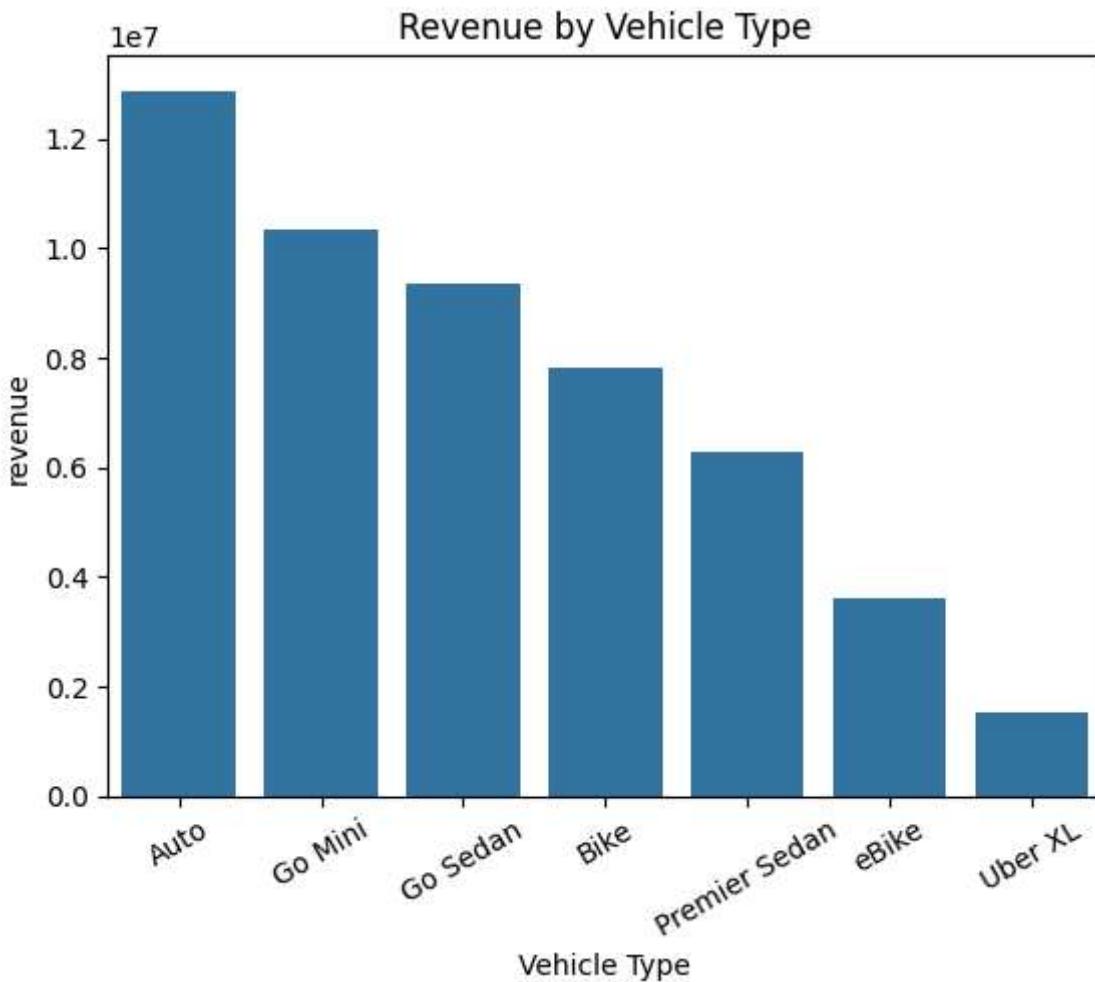
```
In [14]: sns.barplot(data=df_status, x="Booking Status", y="total")  
plt.title("Completed vs Cancelled Rides")  
plt.xticks(rotation=30)  
plt.show()
```

Completed vs Cancelled Rides



```
In [ ]: query = """
SELECT `Vehicle Type`, SUM(`Booking Value`) AS revenue
FROM ncr_ride_bookings
GROUP BY `Vehicle Type`
ORDER BY revenue DESC;
"""

df_vehicle = pd.read_sql(query, engine)
sns.barplot(data=df_vehicle, x="Vehicle Type", y="revenue")
plt.title("Revenue by Vehicle Type")
plt.xticks(rotation=30)
plt.show()
```



In []:

```
import seaborn as sns
import matplotlib.pyplot as plt

query = """
SELECT `Payment Method`, SUM(`Booking Value`) AS revenue
FROM ncr_ride_bookings
GROUP BY `Payment Method`
ORDER BY revenue DESC;
"""

df_payment = pd.read_sql(query, engine)

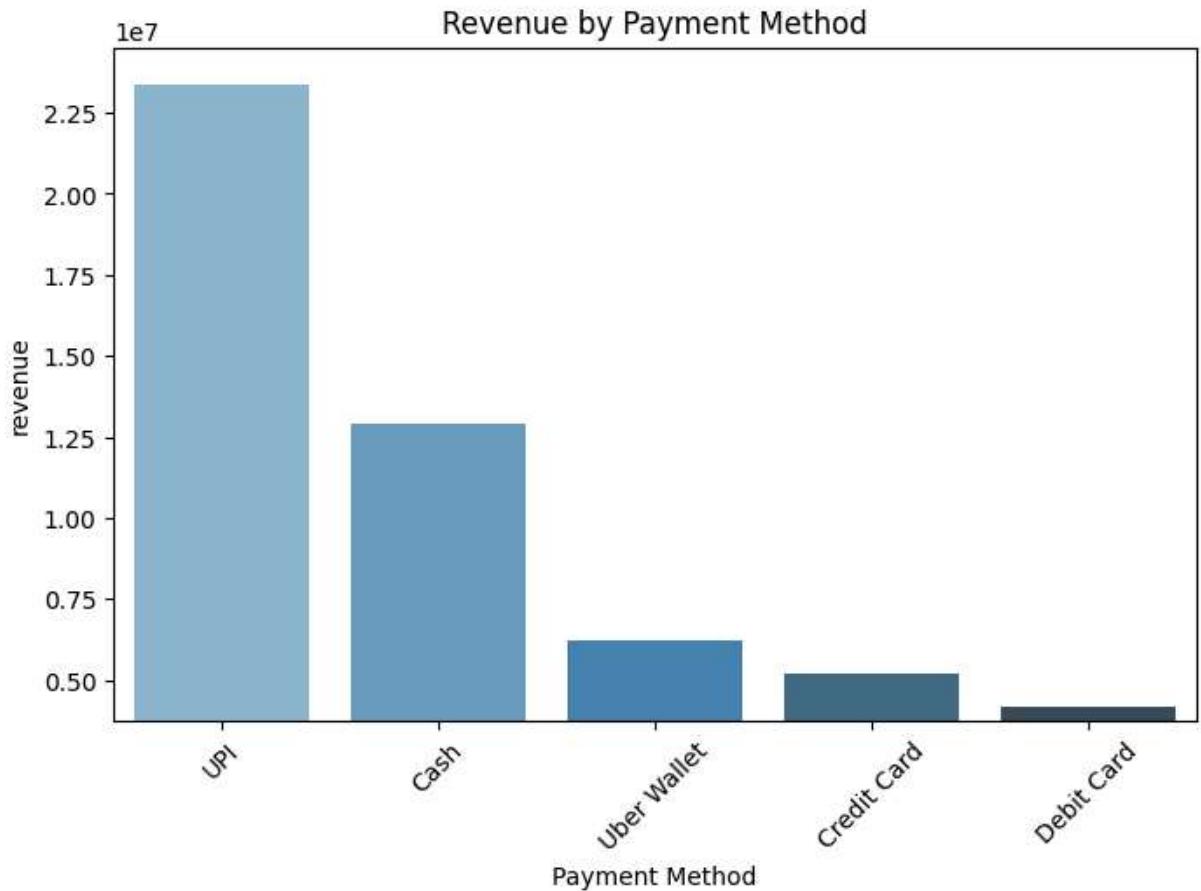
plt.figure(figsize=(8,5))
ax = sns.barplot(data=df_payment, x="Payment Method", y="revenue", palette="Blues_d

ax.set_yticks(df_payment["revenue"].min() * 0.9, df_payment["revenue"].max() * 1.05)

plt.title("Revenue by Payment Method")
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\deepb\AppData\Local\Temp\ipykernel_15688\2818143199.py:14: FutureWarning:
Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

ax = sns.barplot(data=df_payment, x="Payment Method", y="revenue", palette="Blues_
d")
```



```
In [ ]: query = """
SELECT `Vehicle Type`, AVG(`Customer Rating`) AS avg_customer_rating
FROM ncr_ride_bookings
GROUP BY `Vehicle Type`
ORDER BY avg_customer_rating DESC;
"""

df_ratings = pd.read_sql(query, engine)
print(df_ratings)
plt.figure(figsize=(8,5))
ax = sns.barplot(data=df_ratings, x="Vehicle Type", y="avg_customer_rating", palette="Blues_d")
ax.set_ybound(df_ratings["avg_customer_rating"].min() * 0.98,
              df_ratings["avg_customer_rating"].max() * 1.02)

plt.title("Average Customer Rating by Vehicle Type")
plt.xticks(rotation=45)
plt.show()
```

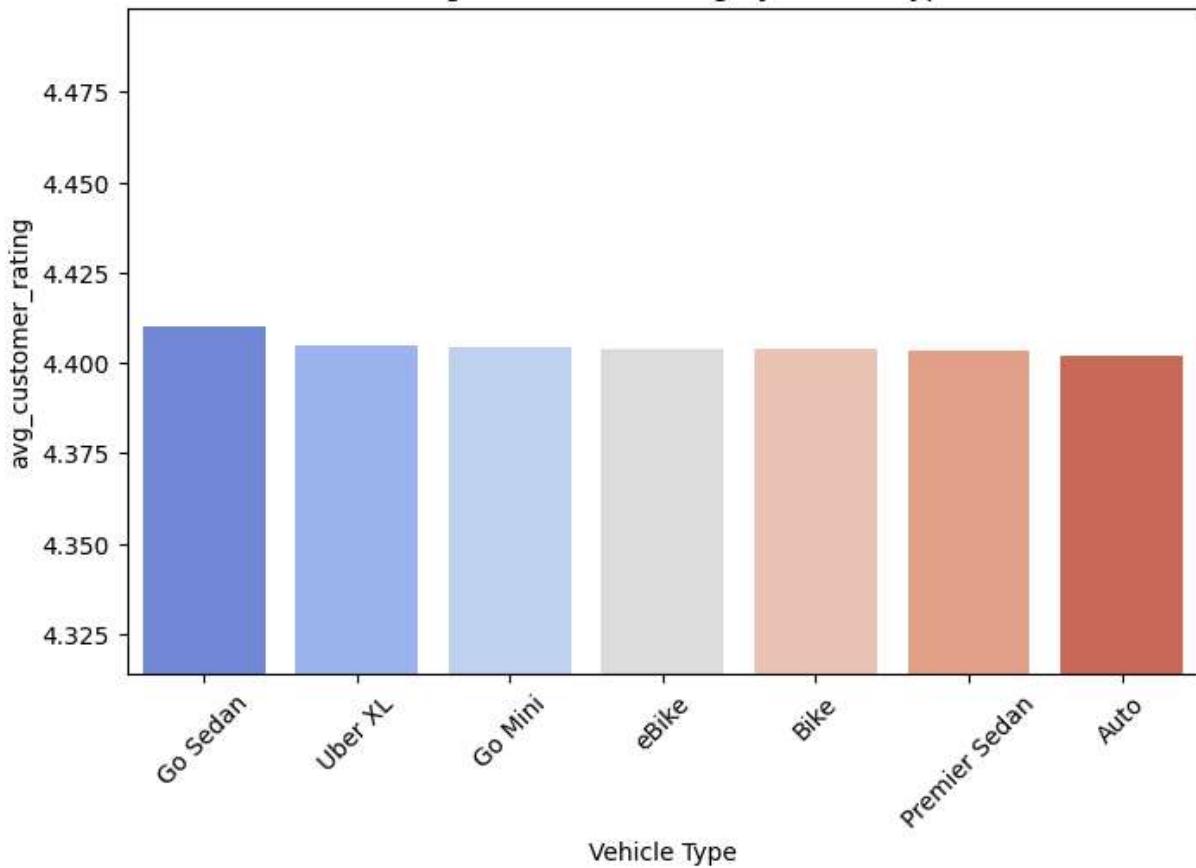
```
Vehicle Type    avg_customer_rating
0      Go Sedan        4.409996
1      Uber XL         4.404851
2      Go Mini          4.404297
3      eBike             4.403954
4      Bike              4.403940
5  Premier Sedan       4.403457
6      Auto              4.402000
```

C:\Users\deepb\AppData\Local\Temp\ipykernel_15688\84968116.py:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1 4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
ax = sns.barplot(data=df_ratings, x="Vehicle Type", y="avg_customer_rating", palette="coolwarm")
```

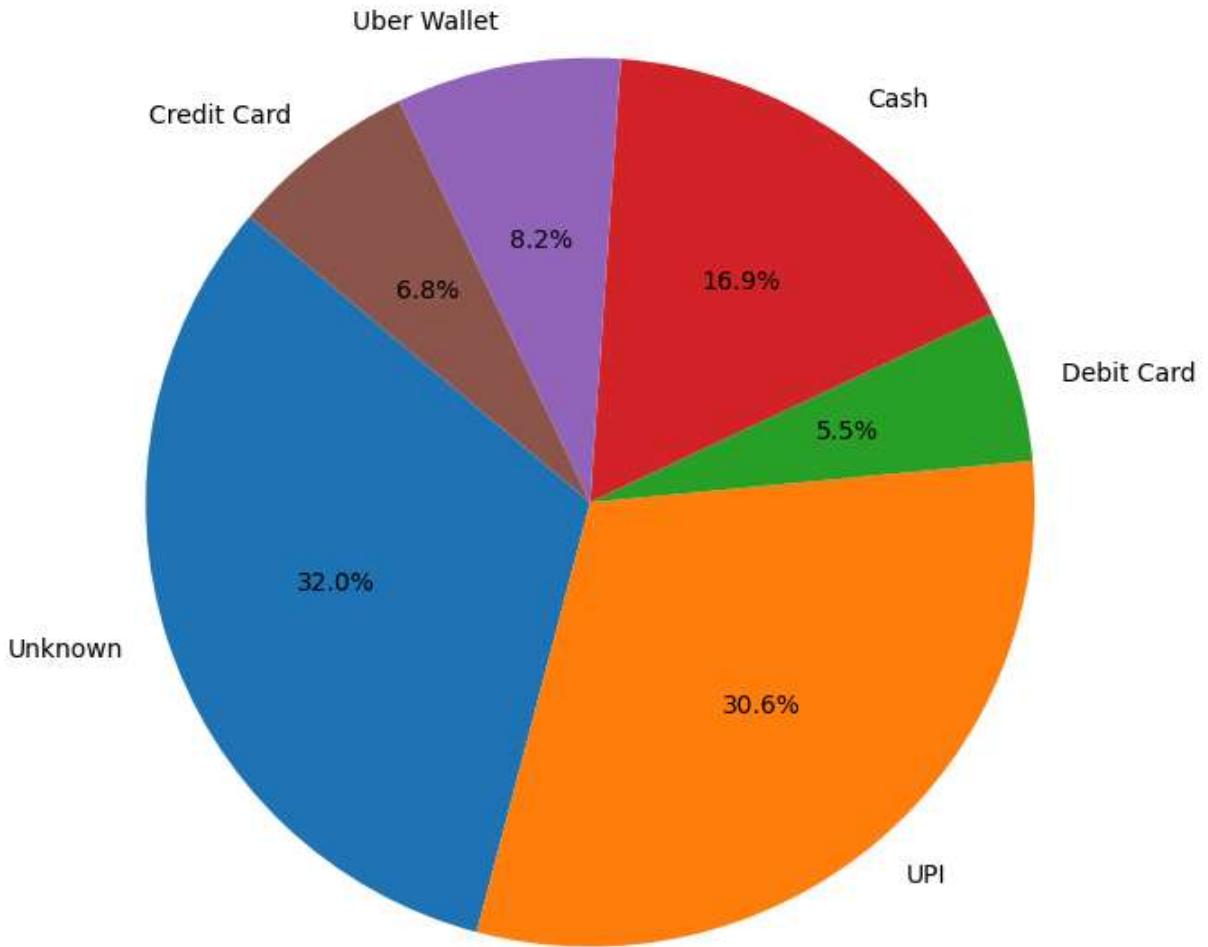
Average Customer Rating by Vehicle Type



```
In [ ]: query = """
SELECT `Payment Method`, COUNT(*) AS total_rides
FROM ncr_ride_bookings
GROUP BY `Payment Method`;
"""

df_pie = pd.read_sql(query, engine)
df_pie['Payment Method'] = df_pie['Payment Method'].fillna('Unknown')
plt.figure(figsize=(8,8))
plt.pie(df_pie['total_rides'], labels=df_pie['Payment Method'], autopct='%1.1f%%',
plt.title('Share of Rides by Payment Method')
plt.show()
```

Share of Rides by Payment Method



```
In [49]: from urllib.parse import quote_plus
from sqlalchemy import create_engine

# Encode special characters in password
password = quote_plus("Poojadeep@231") # @ becomes %40

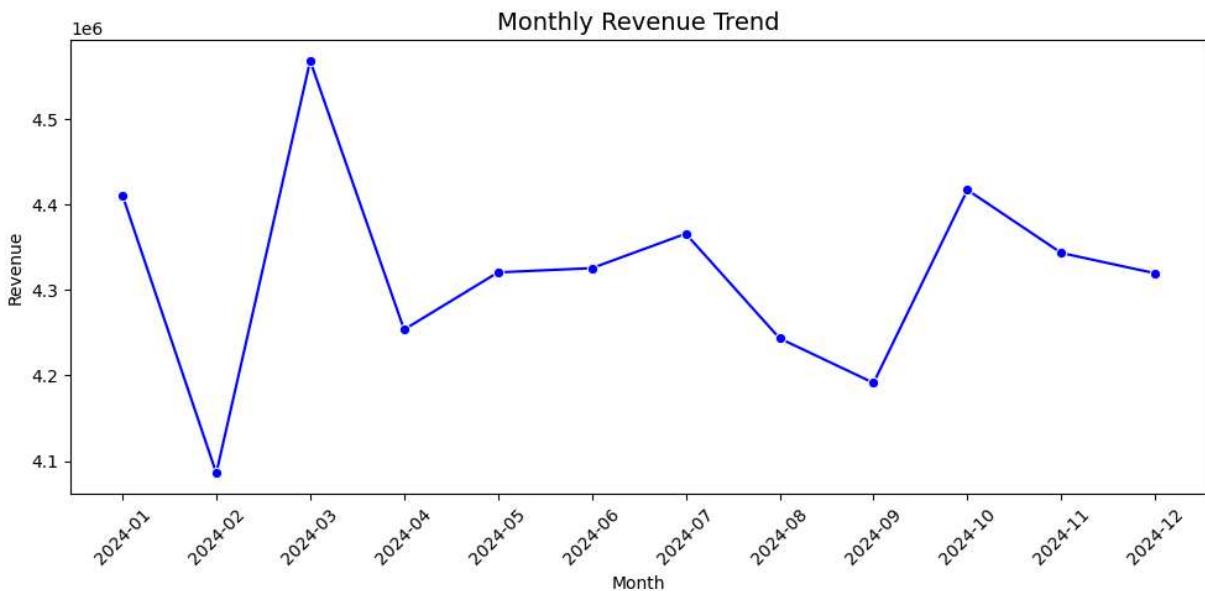
# Create engine (replace host, port, database)
engine = create_engine(f"mysql+pymysql://root:{password}@localhost:3306/uber_analyt
```

```
In [ ]: query = """
SELECT DATE_FORMAT(Date, '%Y-%m') AS month,
       SUM(`Booking Value`) AS total_revenue
FROM ncr_ride_bookings
GROUP BY month
ORDER BY month;
"""

df_monthly = pd.read_sql(query, engine)

plt.figure(figsize=(10,5))
sns.lineplot(data=df_monthly, x="month", y="total_revenue", marker="o", color="blue")
```

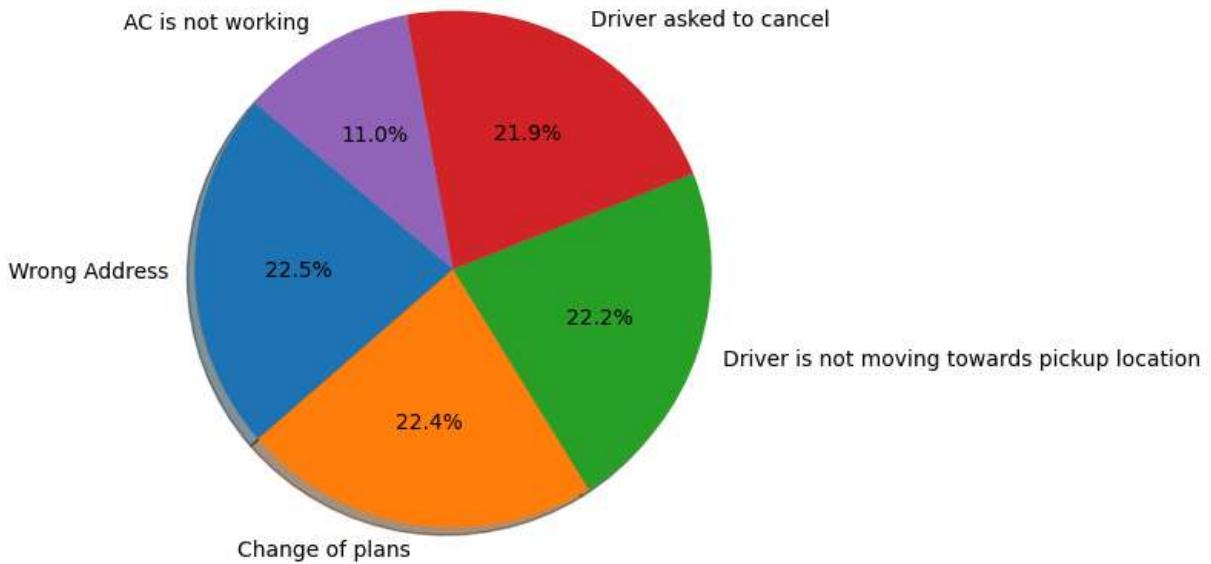
```
plt.title("Monthly Revenue Trend", fontsize=14)
plt.xlabel("Month")
plt.ylabel("Revenue")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



```
In [56]: query = """
SELECT `Reason for cancelling by Customer`
FROM ncr_ride_bookings
WHERE `Reason for cancelling by Customer` IS NOT NULL;
"""

df_cancel = pd.read_sql(query, engine)
cancellation_counts = df_cancel['Reason for cancelling by Customer'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(
    cancellation_counts,
    labels=cancellation_counts.index,
    autopct='%1.1f%%',
    startangle=140,
    shadow=True
)
plt.title("Reasons for Cancelling by Customer")
plt.axis("equal") # makes it a perfect circle
plt.tight_layout()
plt.show()
```

Reasons for Cancelling by Customer

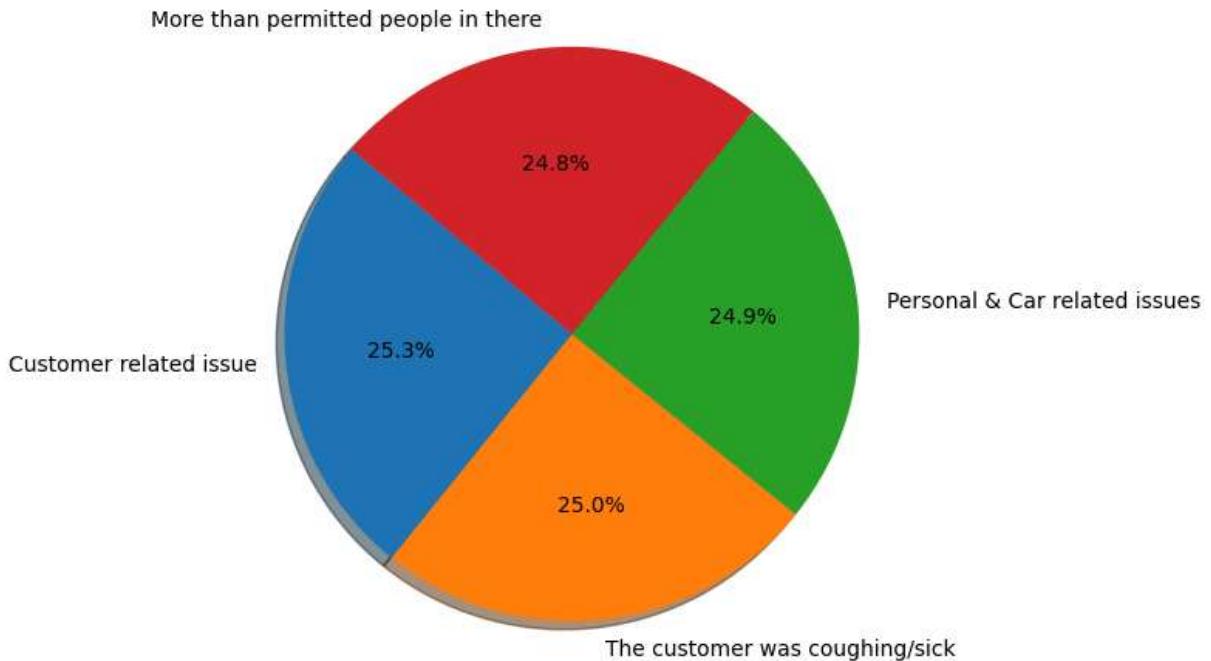


In [57]:

```
query = """
SELECT `Driver Cancellation Reason`
FROM ncr_ride_bookings
WHERE `Driver Cancellation Reason` IS NOT NULL;
"""

df_driver_cancel = pd.read_sql(query, engine)
cancellation_counts = df_driver_cancel['Driver Cancellation Reason'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(
    cancellation_counts,
    labels=cancellation_counts.index,
    autopct='%1.1f%%',
    startangle=140,
    shadow=True
)
plt.title("Driver Cancellation Reasons - Pie Chart")
plt.axis("equal") # keeps the pie chart round
plt.tight_layout()
plt.show()
```

Driver Cancellation Reasons - Pie Chart



```
In [59]: query = """
SELECT `Avg VTAT`, `Avg CTAT`
FROM ncr_ride_bookings
WHERE `Avg VTAT` IS NOT NULL AND `Avg CTAT` IS NOT NULL;
"""
df_time = pd.read_sql(query, engine)

df_time['Avg VTAT'] = pd.to_numeric(df_time['Avg VTAT'], errors='coerce')
df_time['Avg CTAT'] = pd.to_numeric(df_time['Avg CTAT'], errors='coerce')

avg_vtat = df_time['Avg VTAT'].mean()
avg_ctat = df_time['Avg CTAT'].mean()

print(f"Avg VTAT: {avg_vtat:.2f} minutes")
print(f"Avg CTAT: {avg_ctat:.2f} minutes")

avg_df = pd.DataFrame({
    'Type': ['Driver to Pickup (VTAT)', 'Trip Duration (CTAT)'],
    'Avg Time (min)': [avg_vtat, avg_ctat]
})
```

```

})

plt.figure(figsize=(6, 5))
sns.barplot(x='Type', y='Avg Time (min)', data=avg_df, palette='Blues')
plt.title("Average Time Comparison: VTAT vs CTAT")
plt.ylabel("Time (minutes)")
plt.xlabel("")
plt.xticks(rotation=10)
plt.tight_layout()
plt.show()

```

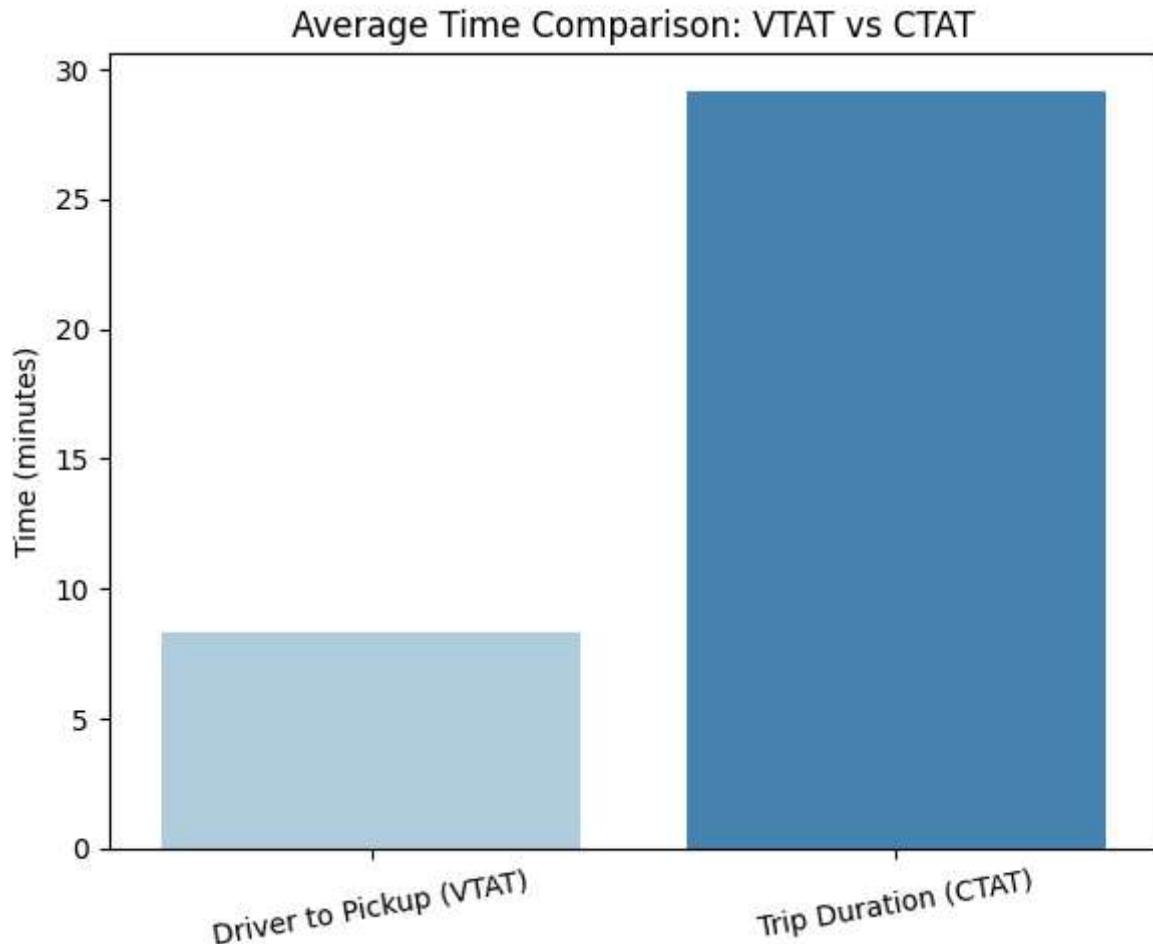
Avg VTAT: 8.29 minutes

Avg CTAT: 29.15 minutes

C:\Users\deepb\AppData\Local\Temp\ipykernel_15688\1287129420.py:23: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

```
sns.barplot(x='Type', y='Avg Time (min)', data=avg_df, palette='Blues')
```



```
In [ ]: query = "SELECT `Vehicle Type` FROM ncr_ride_bookings;"
df_vehicle = pd.read_sql(query, engine)
plt.figure(figsize=(8,5))
sns.countplot(
    x='Vehicle Type',
    data=df_vehicle,
    order=df_vehicle['Vehicle Type'].value_counts().index,
```

```

        palette='pastel'
    )
plt.title("Most Preferred Uber Vehicle Types")
plt.ylabel("Number of Bookings")
plt.xlabel("Vehicle Type")
plt.xticks(rotation=20)
for p in plt.gca().patches:
    plt.gca().annotate(
        format(p.get_height(), ','),
        (p.get_x() + p.get_width() / 2., p.get_height()),
        ha='center', va='center',
        xytext=(0, 8),
        textcoords='offset points'
    )
plt.tight_layout()
plt.show()

```

C:\Users\deepb\AppData\Local\Temp\ipykernel_15688\3823284624.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1 4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

sns.countplot(

