Setup

In [10]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load datasets
file_path = "C:/Users/ASUS/Downloads/"
transaction_data = pd.read_excel(file_path + "QVI_transaction_data.xlsx")  # Use
customer_data = pd.read_csv(file_path + "QVI_purchase_behaviour.csv")

# Check the structure of the transaction data
print(transaction_data.info())
print(transaction_data.head(10))
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 264836 entries, 0 to 264835
Data columns (total 8 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   DATE            264836 non-null  int64  
 1   STORE_NBR       264836 non-null  int64  
 2   LYLTY_CARD_NBR  264836 non-null  int64  
 3   TXN_ID          264836 non-null  int64  
 4   PROD_NBR        264836 non-null  int64  
 5   PROD_NAME       264836 non-null  object 
 6   PROD_QTY        264836 non-null  int64  
 7   TOT_SALES       264836 non-null  float64
dtypes: float64(1), int64(6), object(1)
memory usage: 16.2+ MB
None
    DATE  STORE_NBR  LYLTY_CARD_NBR  TXN_ID  PROD_NBR  \
0  43390          1            1000       1         5   
1  43599          1            1307     348        66   
2  43605          1            1343     383        61   
3  43329          2            2373     974        69   
4  43330          2            2426    1038       108   
5  43604          4            4074    2982        57   
6  43601          4            4149    3333        16   
7  43601          4            4196    3539        24   
8  43332          5            5026    4525        42   
9  43330          7            7150    6900        52   

                                  PROD_NAME  PROD_QTY  TOT_SALES  
0    Natural Chip        Compny SeaSalt175g         2        6.0  
1                  CCs Nacho Cheese    175g         3        6.3  
2    Smiths Crinkle Cut  Chips Chicken 170g         2        2.9  
3    Smiths Chip Thinly  S/Cream&Onion 175g         5       15.0  
4  Kettle Tortilla ChpsHny&Jlpno Chili 150g         3       13.8  
5  Old El Paso Salsa   Dip Tomato Mild 300g         1        5.1  
6  Smiths Crinkle Chips Salt & Vinegar 330g         1        5.7  
7     Grain Waves         Sweet Chilli 210g         1        3.6  
8   Doritos Corn Chip Mexican Jalapeno 150g         1        3.9  
9     Grain Waves Sour    Cream&Chives 210G         2        7.2  
```

Convert DATE column to datetime format

In [12]:
```
transaction_data['DATE'] = pd.to_datetime(transaction_data['DATE'], origin='1899
```

In [14]:
```
# Check unique product names
print(transaction_data['PROD_NAME'].value_counts())
```

```
PROD_NAME
Kettle Mozzarella   Basil & Pesto 175g      3304
Kettle Tortilla ChpsHny&Jlpno Chili 150g    3296
Cobs Popd Swt/Chlli &Sr/Cream Chips 110g    3269
Tyrrells Crisps     Ched & Chives 165g      3268
Cobs Popd Sea Salt  Chips 110g              3265
                                            ... 
RRD Pc Sea Salt     165g                    1431
Woolworths Medium   Salsa 300g              1430
NCC Sour Cream &    Garden Chives 175g      1419
French Fries Potato Chips 175g              1418
WW Crinkle Cut      Original 175g           1410
Name: count, Length: 114, dtype: int64
```

Remove Salsa Product

In [17]:
```
transaction_data = transaction_data[~transaction_data['PROD_NAME'].str.contains(
```

Summary statistics

In [20]:
```
print(transaction_data.describe())
```

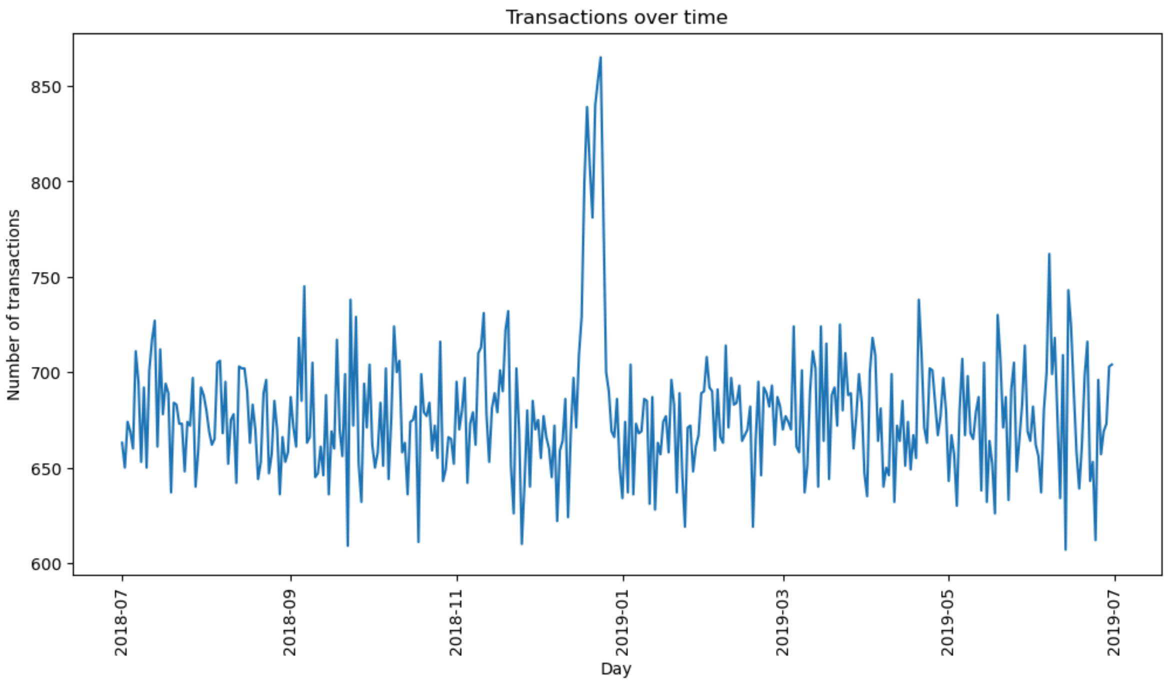
```
                                DATE      STORE_NBR  LYLTY_CARD_NBR  \
count                         246742  246742.000000    2.467420e+05   
mean   2018-12-30 01:19:01.211467520     135.051098    1.355310e+05   
min              2018-07-01 00:00:00       1.000000    1.000000e+03   
25%              2018-09-30 00:00:00      70.000000    7.001500e+04   
50%              2018-12-30 00:00:00     130.000000    1.303670e+05   
75%              2019-03-31 00:00:00     203.000000    2.030840e+05   
max              2019-06-30 00:00:00     272.000000    2.373711e+06   
std                              NaN      76.787096    8.071528e+04   

             TXN_ID       PROD_NBR       PROD_QTY      TOT_SALES  
count  2.467420e+05  246742.000000  246742.000000  246742.000000  
mean   1.351311e+05      56.351789       1.908062       7.321322  
min    1.000000e+00       1.000000       1.000000       1.700000  
25%    6.756925e+04      26.000000       2.000000       5.800000  
50%    1.351830e+05      53.000000       2.000000       7.400000  
75%    2.026538e+05      87.000000       2.000000       8.800000  
max    2.415841e+06     114.000000     200.000000     650.000000  
std    7.814772e+04      33.695428       0.659831       3.077828  
```

Calculations

In [22]:
```
# Filter out the outlier
transaction_data = transaction_data[transaction_data['LYLTY_CARD_NBR'] != 226000
```

In [24]:
```
# Count the number of transactions by date
transactions_by_day = transaction_data['DATE'].value_counts().reset_index()
```

```python
transactions_by_day.columns = ['DATE', 'N']

# Plot transactions over time
plt.figure(figsize=(12, 6))
sns.lineplot(data=transactions_by_day, x='DATE', y='N')
plt.title('Transactions over time')
plt.xlabel('Day')
plt.ylabel('Number of transactions')
plt.xticks(rotation=90)
plt.show()
```

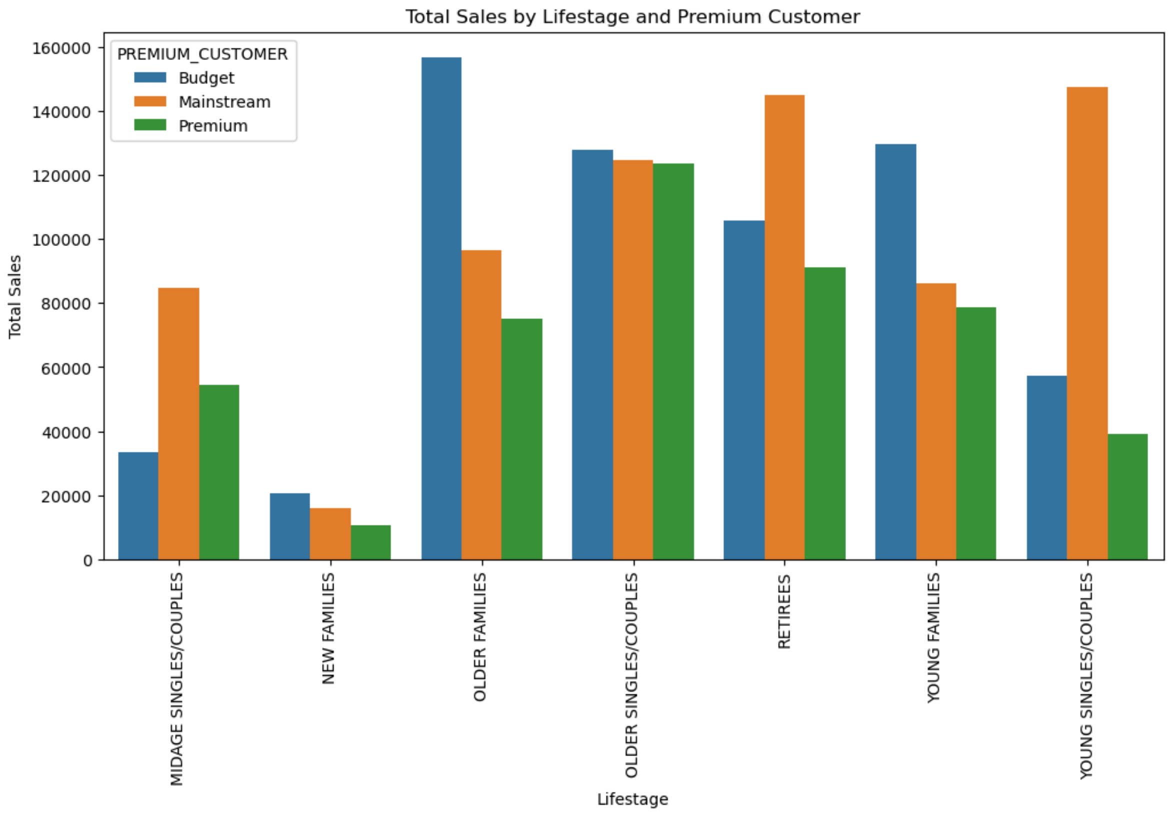


Transactions and Customer Behavior

In [30]:
```python
# Extract pack size from PROD_NAME using a raw string
transaction_data['PACK_SIZE'] = transaction_data['PROD_NAME'].str.extract(r'(\d+

# Create brand column
transaction_data['BRAND'] = transaction_data['PROD_NAME'].str.split().str[0].str
```

In [34]:
```python
# Merge transaction data with customer data
data = pd.merge(transaction_data, customer_data, on='LYLTY_CARD_NBR', how='left'
```

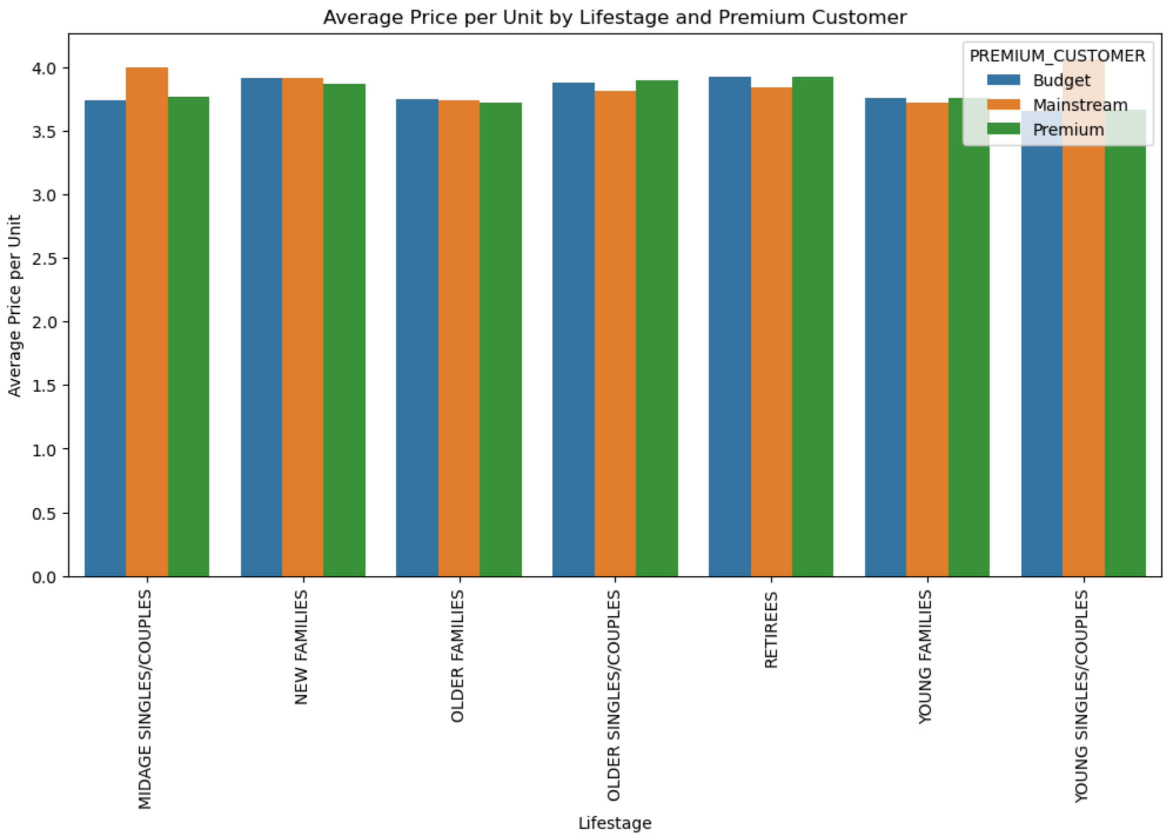
In [36]:
```python
# Total sales by LIFESTAGE and PREMIUM_CUSTOMER
sales = data.groupby(['LIFESTAGE', 'PREMIUM_CUSTOMER'])['TOT_SALES'].sum().reset

# Create plot
plt.figure(figsize=(12, 6))
sns.barplot(data=sales, x='LIFESTAGE', y='TOT_SALES', hue='PREMIUM_CUSTOMER')
plt.title('Total Sales by Lifestage and Premium Customer')
plt.xlabel('Lifestage')
plt.ylabel('Total Sales')
plt.xticks(rotation=90)
plt.show()
```



Average price per unit

In [43]:
```
data['price_per_unit'] = data['TOT_SALES'] / data['PROD_QTY']
avg_price = data.groupby(['LIFESTAGE', 'PREMIUM_CUSTOMER'])['price_per_unit'].me
```

In [46]:
```
plt.figure(figsize=(12, 6))
sns.barplot(data=avg_price, x='LIFESTAGE', y='price_per_unit', hue='PREMIUM_CUST
plt.title('Average Price per Unit by Lifestage and Premium Customer')
plt.xlabel('Lifestage')
plt.ylabel('Average Price per Unit')
plt.xticks(rotation=90)
plt.show()
```


Average Price per Unit by Lifestage and Premium Customer
Average Price per Unit
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
MIDAGE SINGLES/COUPLES
NEW FAMILIES
OLDER FAMILIES
OLDER SINGLES/COUPLES
RETIREES
YOUNG FAMILIES
YOUNG SINGLES/COUPLES
Lifestage
PREMIUM_CUSTOMER
Budget
Mainstream
Premium