Data Description:
The dataset you provided contains the following columns:
User_ID: This column represents a unique identifier for each user. It allows us to track individual users across different transactions or interactions. Analyzing user behavior based on this ID can provide insights into user preferences, frequency of purchases, and customer lifetime value.

Product_ID: Similar to User_ID, Product_ID serves as a unique identifier but for products instead of users. It helps in tracking specific products, understanding their popularity, and analyzing purchasing patterns across different product categories.

Gender: Gender provides demographic information about the users. It allows segmentation of the customer base based on gender, enabling gender-specific marketing strategies, product recommendations, and personalized experiences.

Age: Age group categorizes users into different age brackets such as 0-17, 18-25, 26-35, 36-50, and 51+. Understanding the age distribution of customers helps in tailoring products and marketing messages to specific age demographics, as different age groups may have distinct preferences and buying behaviors.

Occupation: Occupation code represents the occupation of the user. Analyzing purchasing behavior based on occupation can provide insights into the spending patterns of different professional groups. This information can be valuable for targeted advertising and product development.

City_Category: City_Category categorizes cities into different categories (A, B, or C). It helps in understanding the geographical distribution of customers and tailoring marketing strategies to suit the preferences and characteristics of customers in different city categories.

Stay_In_Current_City_Years: This column indicates the number of years the user has stayed in the current city. It provides insights into user loyalty and stability, as users who have stayed longer in a city may exhibit different purchasing behaviors compared to those who are relatively new to the city.

Marital_Status: Marital_Status indicates whether the user is married or unmarried. Understanding the marital status of customers helps in creating targeted marketing campaigns, as married and unmarried individuals may have different needs, preferences, and buying behaviors.

Product_Category: Product_Category represents the category code of the product. It allows us to categorize products into different groups, facilitating product analysis, recommendation systems, and personalized marketing strategies based on product preferences.

Purchase: Purchase column contains the amount spent by the user on a particular product. It is the target variable for analysis and prediction tasks, such as predicting customer spending behavior, identifying high-value customers, and optimizing marketing efforts to increase sales and revenue.

In [36]:
```python
import pandas as pd
import numpy as np
df = pd.read_csv("walmart_data.csv")
df
```

0	1000001	P00069042	F	0-17	10	A	2	0
1	1000001	P00248942	F	0-17	10	A	2	0
2	1000001	P00087842	F	0-17	10	A	2	0
3	1000001	P00085442	F	0-17	10	A	2	0
4	1000002	P00285442	M	55+	16	C	4+	0
...	...	...	...	...	...	...	...	...
550063	1006033	P00372445	M	51-55	13	B	1	1
550064	1006035	P00375436	F	26-35	1	C	3	0

```
In [37]: df.head()
```

Out[37]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Categ
0	1000001	P00069042	F	0-17	10	A	2	0	
1	1000001	P00248942	F	0-17	10	A	2	0	
2	1000001	P00087842	F	0-17	10	A	2	0	
3	1000001	P00085442	F	0-17	10	A	2	0	
4	1000002	P00285442	M	55+	16	C	4+	0	

```
In [5]: df.shape
```

```
Out[5]: (550068, 10)
```

```
In [6]: df.dtypes
```

```
Out[6]: User_ID                        int64
        Product_ID                    object
        Gender                        object
        Age                           object
        Occupation                     int64
        City_Category                 object
        Stay_In_Current_City_Years    object
        Marital_Status                 int64
        Product_Category               int64
        Purchase                       int64
        dtype: object
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   User_ID                     550068 non-null  int64
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  int64
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  int64
 8   Product_Category            550068 non-null  int64
 9   Purchase                    550068 non-null  int64
dtypes: int64(5), object(5)
memory usage: 42.0+ MB
```

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

```
Out[9]: Index(['User_ID', 'Product_ID', 'Gender', 'Age', 'Occupation', 'City_Category',
               'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category',
               'Purchase'],
              dtype='object')
```

In [10]:
```
df.describe()
```

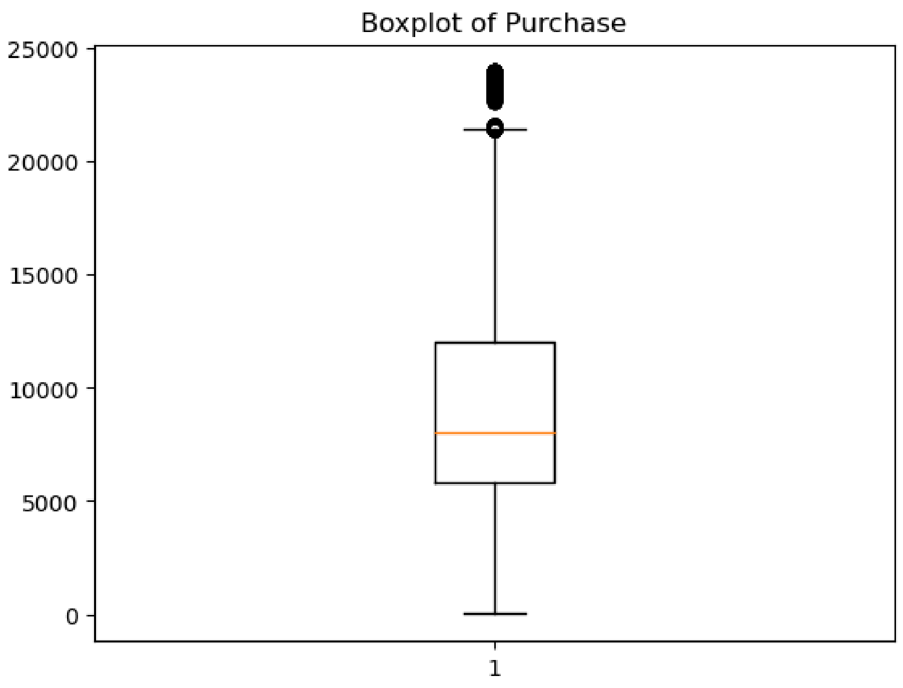
Out[10]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	5.500680e+05	550068.000000	550068.000000	550068.000000	550068.000000
mean	1.003029e+06	8.076707	0.409653	5.404270	9263.968713
std	1.727592e+03	6.522660	0.491770	3.936211	5023.065394
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001516e+06	2.000000	0.000000	1.000000	5823.000000
50%	1.003077e+06	7.000000	0.000000	5.000000	8047.000000
75%	1.004478e+06	14.000000	1.000000	8.000000	12054.000000
max	1.006040e+06	20.000000	1.000000	20.000000	23961.000000

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

```
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

In [12]:
```
# Detecting outliers using boxplot
import matplotlib.pyplot as plt

plt.boxplot(df['Purchase'])
plt.title('Boxplot of Purchase')
plt.show()
```



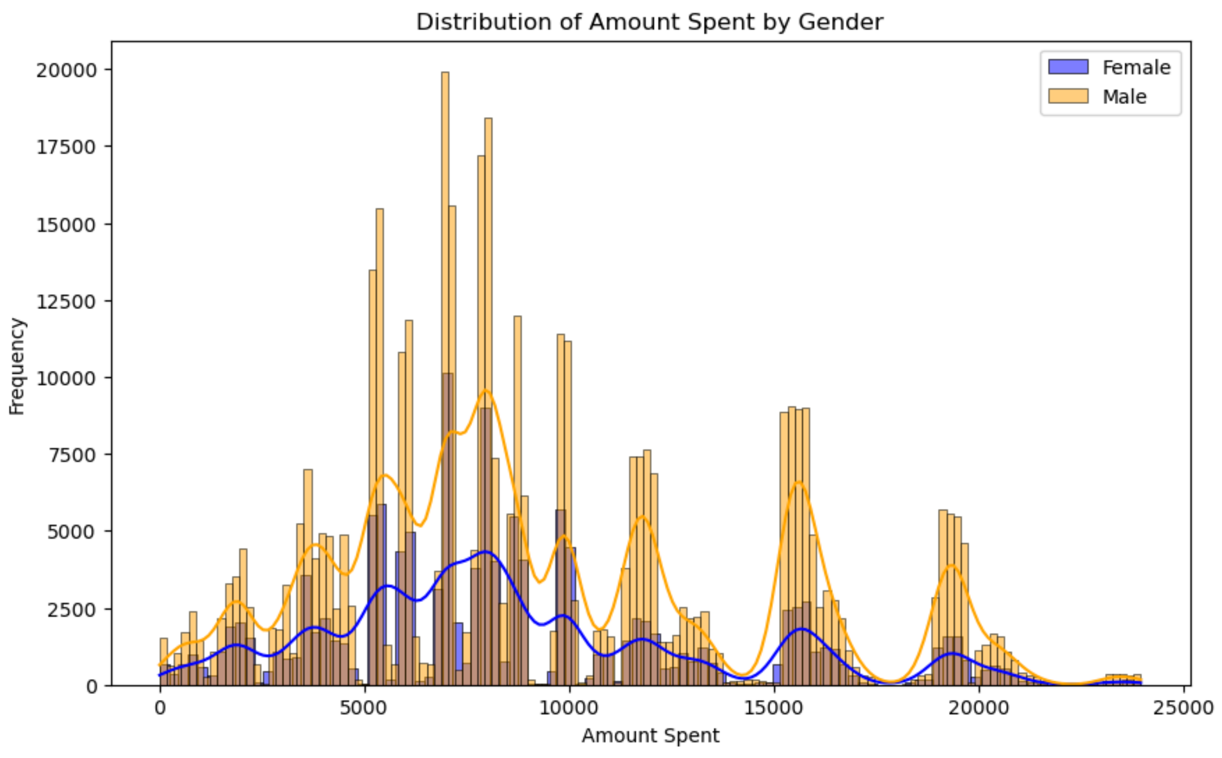
In [13]:
```
# Checking the difference between mean and median
print("Mean:", df['Purchase'].mean())
print("Median:", df['Purchase'].median())
```

```
Mean: 9263.968712959126
Median: 8047.0
```

In [14]:
```
# Tracking the amount spent per transaction of all the 50 million female customers and all the 50 mi
female_avg = df[df['Gender'] == 'F']['Purchase'].mean()
male_avg = df[df['Gender'] == 'M']['Purchase'].mean()

print("Average amount spent by female customers:", female_avg)
print("Average amount spent by male customers:", male_avg)
```

```
Average amount spent by female customers: 8734.565765155476
Average amount spent by male customers: 9437.526040472265
```

In [27]:
```
#visualize
# Filter data for female and male customers
female_data = df[df['Gender'] == 'F']['Purchase']
male_data = df[df['Gender'] == 'M']['Purchase']

import seaborn as sns
plt.figure(figsize=(10, 6))
sns.histplot(female_data, kde=True, color='blue', label='Female')
sns.histplot(male_data, kde=True, color='orange', label='Male')
plt.title('Distribution of Amount Spent by Gender')
plt.xlabel('Amount Spent')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



In [15]:
```
# Inference after computing the average female and male expenses
if female_avg > male_avg:
    print("Female customers spend more on average.")
elif female_avg < male_avg:
    print("Male customers spend more on average.")
else:
    print("Average spending is the same for both genders.")
```

```
Male customers spend more on average.
```

In [17]:
```
sample_size = 1000
female_sample = df[df['Gender'] == 'F']['Purchase'].sample(sample_size)
male_sample = df[df['Gender'] == 'M']['Purchase'].sample(sample_size)
# Confidence interval calculation
female_mean = female_sample.mean()
male_mean = male_sample.mean()
female_std = female_sample.std()
male_std = male_sample.std()
female_std_err = female_std / np.sqrt(sample_size)
male_std_err = male_std / np.sqrt(sample_size)
```

In [18]:
```
z_score = 1.96

# Confidence intervals
female_ci_low = female_mean - z_score * female_std_err
female_ci_high = female_mean + z_score * female_std_err
male_ci_low = male_mean - z_score * male_std_err
male_ci_high = male_mean + z_score * male_std_err

print("95% Confidence Interval for average spending of female customers:", (female_ci_low, female_ci
print("95% Confidence Interval for average spending of male customers:", (male_ci_low, male_ci_high)
```

```
95% Confidence Interval for average spending of female customers: (8258.032922121323, 8836.92107787
8678)
95% Confidence Interval for average spending of male customers: (9377.431562738095, 10002.276437261
904)
```

In [22]:
```
# Convert 'Age' column to numeric values
df['Age'] = df['Age'].apply(lambda x: int(x.split('+')[0]) if '+' in x else int(x.split('-')[0]))

# Checking the structure
print(df.head())
```

```
   User_ID Product_ID Gender  Age  Occupation City_Category  \
0  1000001  P00069042      F    0          10             A
1  1000001  P00248942      F    0          10             A
2  1000001  P00087842      F    0          10             A
3  1000001  P00085442      F    0          10             A
4  1000002  P00285442      M   55          16             C

  Stay_In_Current_City_Years  Marital_Status  Product_Category  Purchase
0                          2               0                 3      8370
1                          2               0                 1     15200
2                          2               0                12      1422
3                          2               0                12      1057
4                         4+               0                 8      7969
```

In [19]:
```
# Conclude the results and check if the confidence intervals of average male and female spends are o
if female_ci_low > male_ci_high or male_ci_low > female_ci_high:
    print("The confidence intervals do not overlap.")
else:
    print("The confidence intervals overlap.")
```

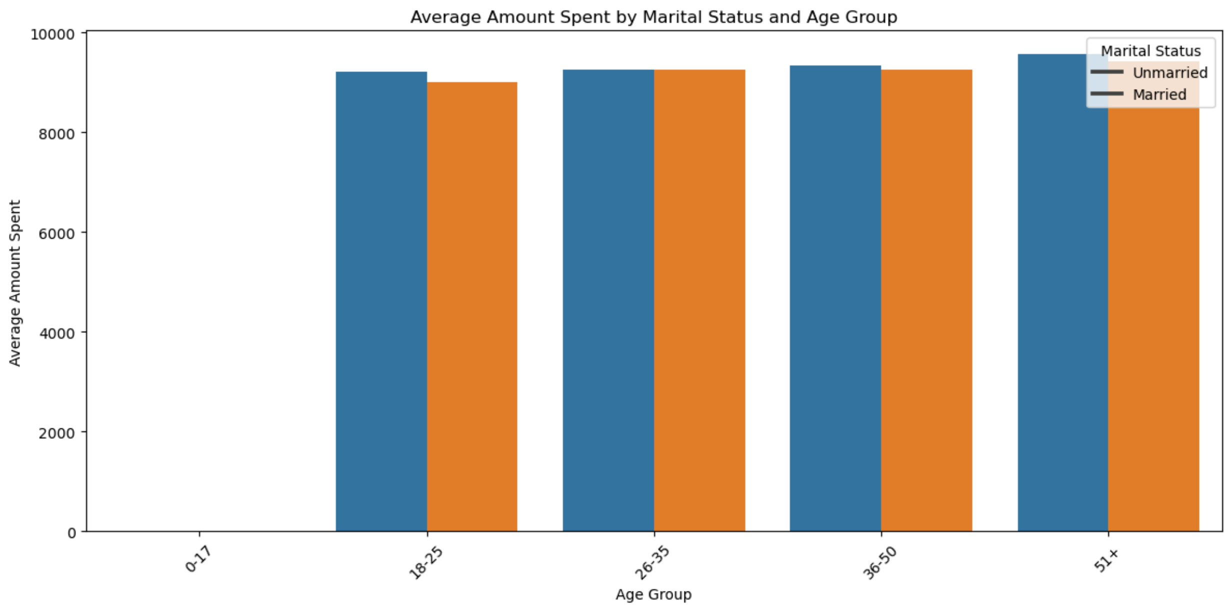
```
The confidence intervals do not overlap.
```

In [23]:
```
age_bins = [0, 17, 25, 35, 50, np.inf]
age_labels = ['0-17', '18-25', '26-35', '36-50', '51+']
df['AgeGroup'] = pd.cut(df['Age'], bins=age_bins, labels=age_labels)

# Grouping by marital status and age group
married_avg = df[df['Marital_Status'] == 1]['Purchase'].mean()
unmarried_avg = df[df['Marital_Status'] == 0]['Purchase'].mean()

print("Average amount spent by married customers:", married_avg)
print("Average amount spent by unmarried customers:", unmarried_avg)
# Conclusion
if married_avg > unmarried_avg:
    print("Married customers spend more on average.")
elif married_avg < unmarried_avg:
    print("Unmarried customers spend more on average.")
else:
    print("Average spending is the same for married and unmarried customers.")
```

```
Average amount spent by married customers: 9261.174574082374
Average amount spent by unmarried customers: 9265.907618921507
Unmarried customers spend more on average.
```

In [29]:
```
grouped_data = df.groupby(['Marital_Status', 'AgeGroup'])['Purchase'].mean().reset_index()
```

In [35]:
```
# Plotting
plt.figure(figsize=(12, 6))
sns.barplot(data=grouped_data, x='AgeGroup', y='Purchase', hue='Marital_Status')
plt.title('Average Amount Spent by Marital Status and Age Group')
plt.xlabel('Age Group')
plt.ylabel('Average Amount Spent')
plt.xticks(rotation=45)
plt.legend(title='Marital Status', labels=['Unmarried', 'Married'])
plt.tight_layout()
plt.show()
```



```
**Observations based on the analysis of average spending by marital status and age group:

**Average Spending by Marital Status:

The average amount spent by married customers is approximately $9261.17.
The average amount spent by unmarried customers is slightly higher, around $9265.91.
Based on this analysis, unmarried customers seem to spend slightly more on average compared to
married customers.

**Average Spending by Age Group and Marital Status:

Married customers tend to spend less on average across all age groups compared to unmarried
customers.
The spending pattern varies across different age groups, with the highest average spending observed
in the age group of 36-50 for both married and unmarried customers.
```

```
Unmarried customers generally exhibit higher average spending compared to married customers across
all age categories.

**Conclusion:

Unmarried customers show a slightly higher tendency to spend more on average compared to married
customers.
Age also influences spending behavior, with customers in the 36-50 age group demonstrating the
highest average spending.
Businesses can leverage these insights to tailor marketing strategies and product offerings to
different customer segments based on their marital status and age, potentially leading to increased
sales and customer satisfaction.
Based on the insights gained from the analysis of average spending by marital status and age group,
here are some recommendations for businesses:

**Targeted Marketing Campaigns:

Develop targeted marketing campaigns that cater to the preferences and behaviors of unmarried
customers, who tend to spend slightly more on average.
Tailor marketing messages and promotions to resonate with the interests and lifestyles of unmarried
individuals, highlighting products or services that appeal to this demographic.

**Product Assortment and Pricing:

Adjust product assortments and pricing strategies to align with the spending patterns of different
age groups and marital statuses.
Offer a diverse range of products and price points to cater to the varying preferences and budgets
of customers across different segments.

**Customer Experience Enhancement:

Enhance the overall customer experience, both online and offline, to attract and retain customers
from different demographic segments.
Personalize the shopping experience by offering relevant product recommendations, promotions, and
discounts based on customer demographics and past purchase behavior.

**Loyalty Programs and Rewards:

Implement loyalty programs and rewards programs tailored to the needs and preferences of different
customer segments.
Offer incentives and rewards that are attractive to unmarried customers, such as exclusive
discounts, early access to new products, or personalized offers based on their interests.
her average spending.

**Customer Engagement and Feedback:

Engage with customers through surveys, feedback mechanisms, and social media channels to understand
their evolving needs and preferences.
Actively listen to customer feedback and incorporate suggestions into product development,
marketing strategies, and overall business operations.
By implementing these recommendations, businesses can better cater to the diverse needs and
preferences of their customer base, drive higher engagement and loyalty, and ultimately, increase
sales and revenue.
```