# PROJECT FILE-2


# DIWALI SALES DATA

Submitted by:

RAJESH BAGHEL

9319652493

The purpose of this analysis is to:

1. **Understand customer purchasing behaviour** during the Diwali season, such as demographics, age groups, gender, and regional patterns in sales.
2. **Analyse sales trends** by identifying which products/categories performed well, observing peak sales days, and studying the seasonal sales impact.
3. **Evaluate customer segmentation**, focusing on insights about high-value customers and repeat buyers, in order to tailor marketing strategies.
4. **Identify product preferences** and analyse which product categories have the highest demand during Diwali.
5. **Provide actionable insights** to optimize marketing campaigns, inventory management, and sales strategies for future Diwali sales.

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv(r"C:\Users\rajesh\Downloads\diwali sales.csv")
```

In [3]:
```
df.head()
```

Out[3]:

| | User_ID | Cust_name | Product_ID | Gender | Age Group | Age | Marital_Status | State | Zone | Occupation | Product_Category | Order Date | Orders | Total sales |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1002903 | Sanskriti | P00125942 | F | 26-35 | 28 | 0 | Maharashtra | Western | Healthcare | Auto | 09-10-2023 | 1 | 23952.0 |
| 1 | 1000732 | Kartik | P00110942 | F | 26-35 | 35 | 1 | Andhra Pradesh | Southern | Govt | Auto | 09-10-2023 | 3 | 23934.0 |
| 2 | 1001990 | Bindu | P00118542 | F | 26-35 | 35 | 1 | Uttar Pradesh | Central | Automobile | Auto | 09-10-2023 | 3 | 23924.0 |
| 3 | 1001425 | Sudevi | P00237842 | M | 0-17 | 16 | 0 | Karnataka | Southern | Construction | Auto | 09-10-2023 | 2 | 23912.0 |
| 4 | 1000588 | Joni | P00057942 | M | 26-35 | 28 | 1 | Gujarat | Western | Food Processing | Auto | 09-10-2023 | 2 | 23877.0 |

In [4]:
```
df.shape
```

Out[4]:
```
(11239, 14)
```

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

Out[5]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Order Date', 'Orders', 'Total sales'],
      dtype='object')
```

In [6]:
```
print(df.isnull().sum())
```

```
User_ID             0
Cust_name           0
Product_ID          0
Gender              0
Age Group           0
Age                 0
Marital_Status      0
State               0
Zone                0
Occupation          0
Product_Category    0
Order Date          0
Orders              0
Total sales         0
dtype: int64
```

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11239 entries, 0 to 11238
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   User_ID           11239 non-null  int64  
 1   Cust_name         11239 non-null  object 
 2   Product_ID        11239 non-null  object 
 3   Gender            11239 non-null  object 
 4   Age Group         11239 non-null  object 
 5   Age               11239 non-null  int64  
 6   Marital_Status    11239 non-null  int64  
 7   State             11239 non-null  object 
 8   Zone              11239 non-null  object 
 9   Occupation        11239 non-null  object 
 10  Product_Category  11239 non-null  object 
 11  Order Date        11239 non-null  object 
 12  Orders            11239 non-null  int64  
 13  Total sales       11239 non-null  float64
dtypes: float64(1), int64(4), object(9)
memory usage: 1.2+ MB
```

# Explore and clean the data

In [9]:
```
df.drop_duplicates(inplace = True)
```

In [10]:
```
df.shape
```

Out[10]: (11231, 14)

In [11]:
```
# Basic statistical summary
df.describe()
```

Out[11]:

| | User_ID | Age | Marital_Status | Orders | Total sales |
|---|---|---|---|---|---|
| count | 1.123100e+04 | 11231.000000 | 11231.000000 | 11231.000000 | 11231.000000 |
| mean | 1.003004e+06 | 35.411985 | 0.419998 | 2.489093 | 9454.084982 |
| std | 1.716055e+03 | 12.756116 | 0.493580 | 1.114880 | 5221.728776 |
| min | 1.000001e+06 | 12.000000 | 0.000000 | 1.000000 | 188.000000 |
| 25% | 1.001492e+06 | 27.000000 | 0.000000 | 2.000000 | 5443.000000 |
| 50% | 1.003065e+06 | 33.000000 | 0.000000 | 2.000000 | 8109.000000 |
| 75% | 1.004428e+06 | 43.000000 | 1.000000 | 3.000000 | 12677.500000 |
| max | 1.006040e+06 | 92.000000 | 1.000000 | 4.000000 | 23952.000000 |

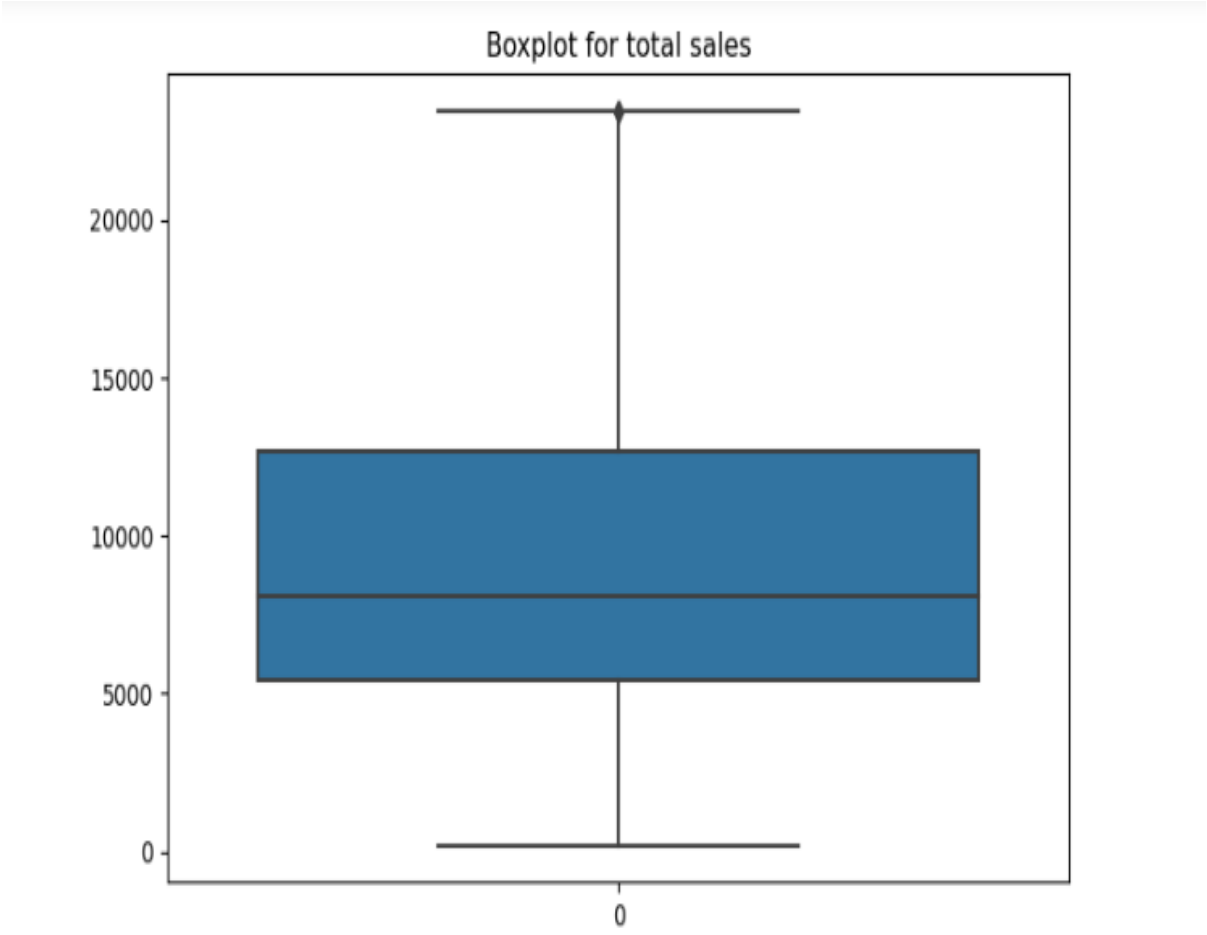
In [36]:
```
# Boxplot to detect outliers in Sales_Amount
plt.figure(figsize=(8,5))
sns.boxplot(df['total sales'])
plt.title('Boxplot for total sales')
plt.show()

# Remove outliers that are beyond a certain threshold (optional)
Q1 = df['total sales'].quantile(0.25)
Q3 = df['total sales'].quantile(0.75)
IQR = Q3 - Q1

# Remove outliers that are below Q1 - 1.5*IQR or above Q3 + 1.5*IQR
df = df[(df['total sales'] >= (Q1 - 1.5 * IQR)) & (df['total sales'] <= (Q3 + 1.5 * IQR))]
```

# #Outlier checking

Outliers can distort statistical analysis and model performance, so it’s important to detect and handle them. Below is a step-by-step guide for identifying outliers in your Diwali sales dataset using various methods such as box plots, Z-scores, and the IQR (Interquartile Range) method.



In [12]:
```
# Unique categories (e.g., product categories, age groups, etc.)
print(df['Product_Category'].unique())
print(df['Age Group'].unique())
```

```
['Auto' 'Hand & Power Tools' 'Stationery' 'Tupperware' 'Footwear & Shoes'
 'Furniture' 'Food' 'Games & Toys' 'Sports Products' 'Books'
 'Electronics & Gadgets' 'Decor' 'Clothing & Apparel' 'Beauty'
 'Household items' 'Pet Care' 'Veterinary' 'Office']
['26-35' '0-17' '18-25' '51-55' '46-50' '55+' '36-45']
```

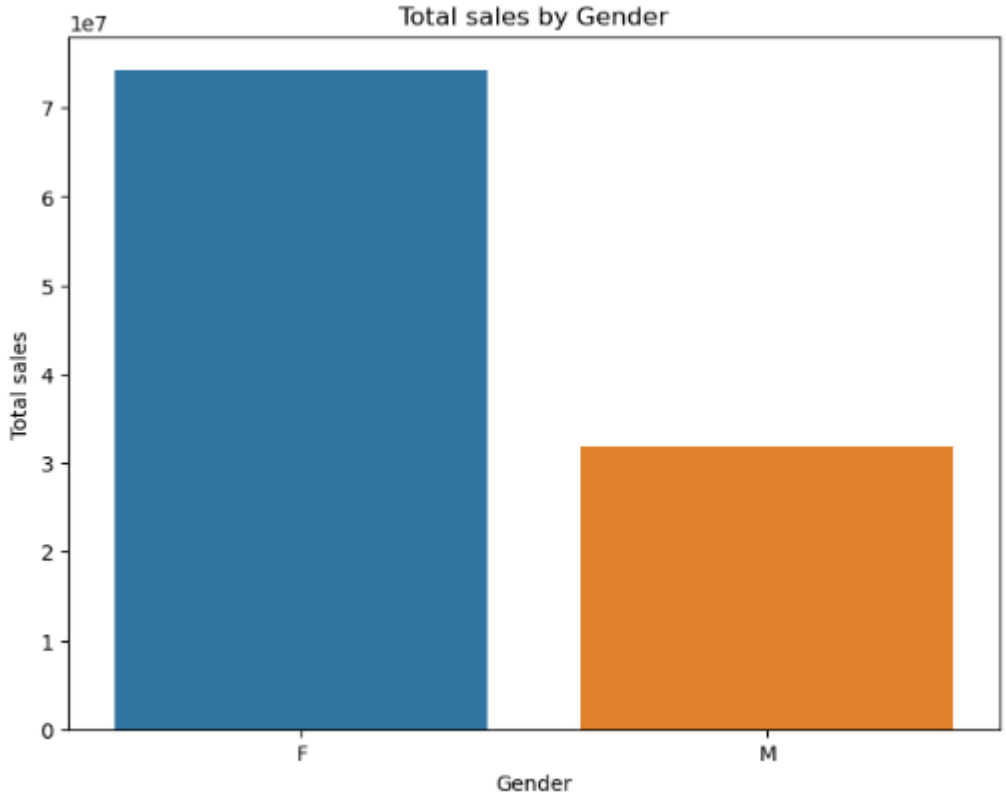
In [13]:
```
# Summary for specific columns (e.g., Gender-based analysis)
df['Gender'].value_counts()
```

Out[13]:
```
Gender
F    7828
M    3403
Name: count, dtype: int64
```

# # Total Sales by Gender

In [14]:
```
# Group data by Gender and calculate total sales
gender_sales = df.groupby('Gender')['Total sales'].sum().reset_index()
```

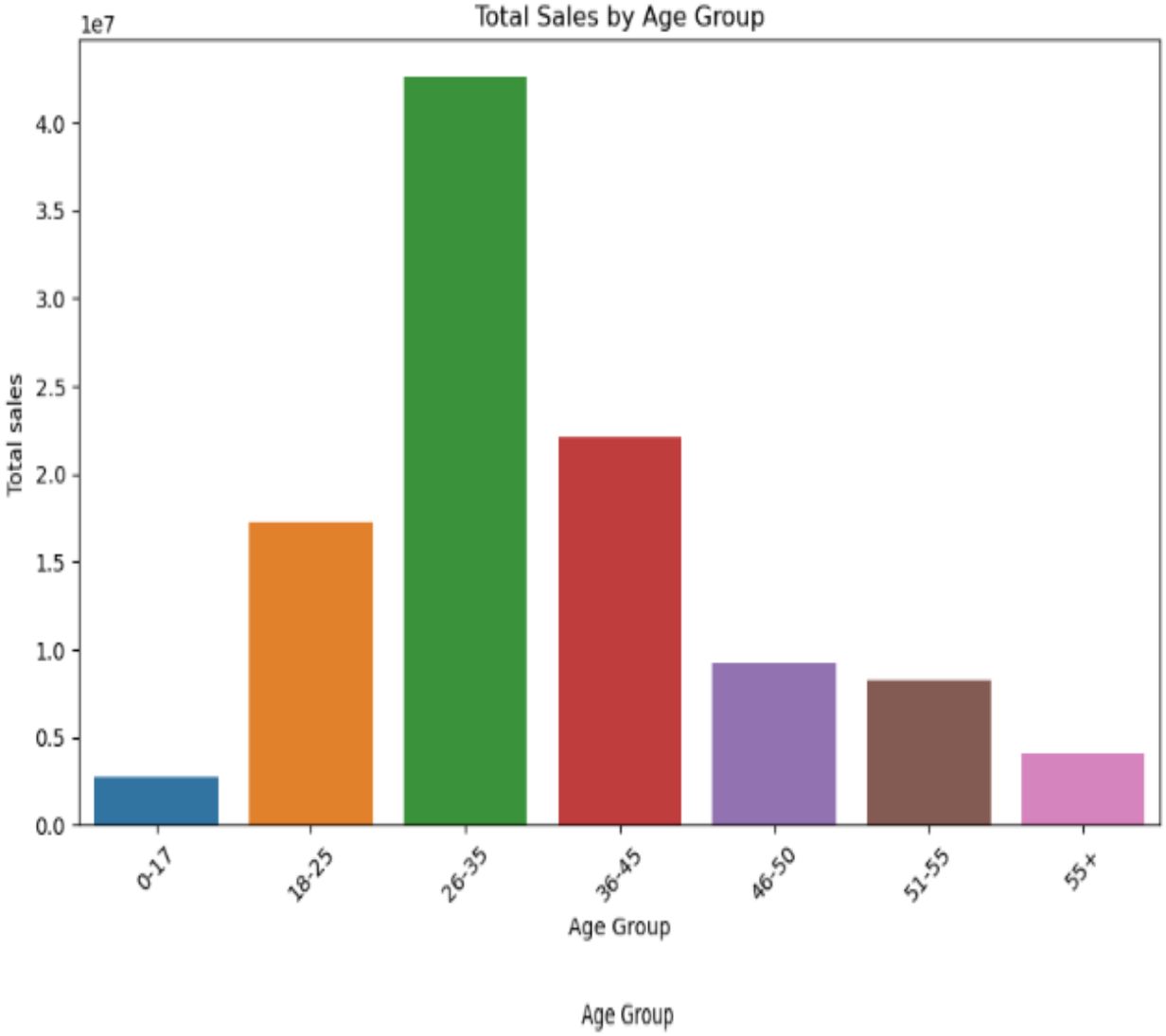
In [15]:
```
# Plot
plt.figure(figsize=(8, 6))
sns.barplot(x='Gender', y='Total sales', data=gender_sales)
plt.title('Total sales by Gender')
plt.xlabel('Gender')
plt.ylabel('Total sales')
plt.show()
```




# # Sales Distribution by Age Group

In [16]:
```
# Group data by Age Group and calculate total sales
age_sales = df.groupby('Age Group')['Total sales'].sum().reset_index()
```

In [17]:
```
# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x='Age Group', y='Total sales', data=age_sales)
plt.title('Total Sales by Age Group')
plt.xlabel('Age Group')
plt.ylabel('Total sales')
plt.xticks(rotation=45)
plt.show()
```

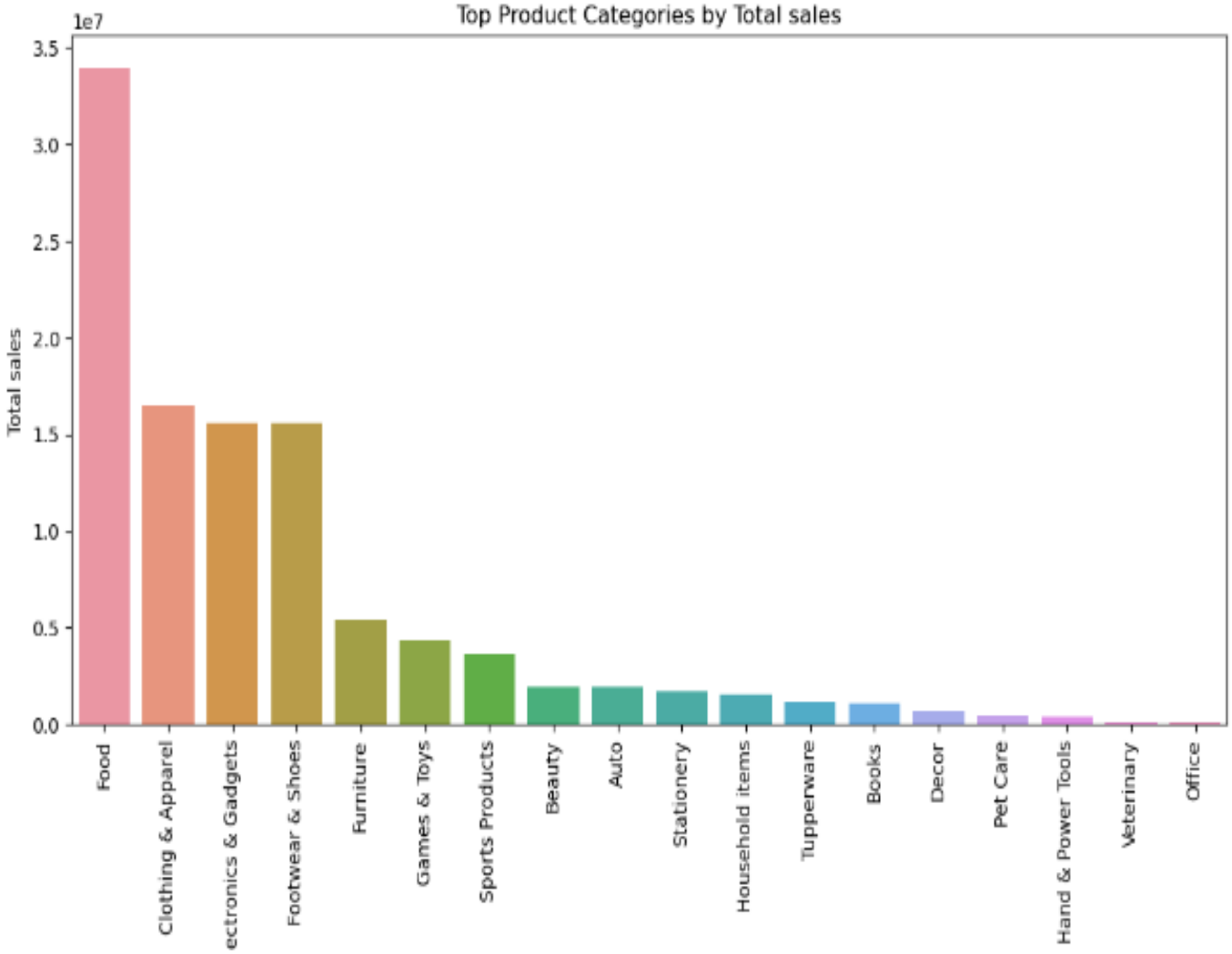



# Top Product Categories by Sales

In [18]:
```
# Group data by Product Category and calculate total sales
category_sales = df.groupby('Product_Category')['Total sales'].sum().reset_index()
```

In [19]:
```
# Sort the data by sales in descending order
category_sales = category_sales.sort_values(by='Total sales', ascending=False)
```

In [20]:
```
# Plot
plt.figure(figsize=(12, 6))
sns.barplot(x='Product_Category', y='Total sales', data=category_sales)
plt.title('Top Product Categories by Total sales')
plt.xlabel('Product_Category')
plt.ylabel('Total sales')
plt.xticks(rotation=90)
plt.show()
```



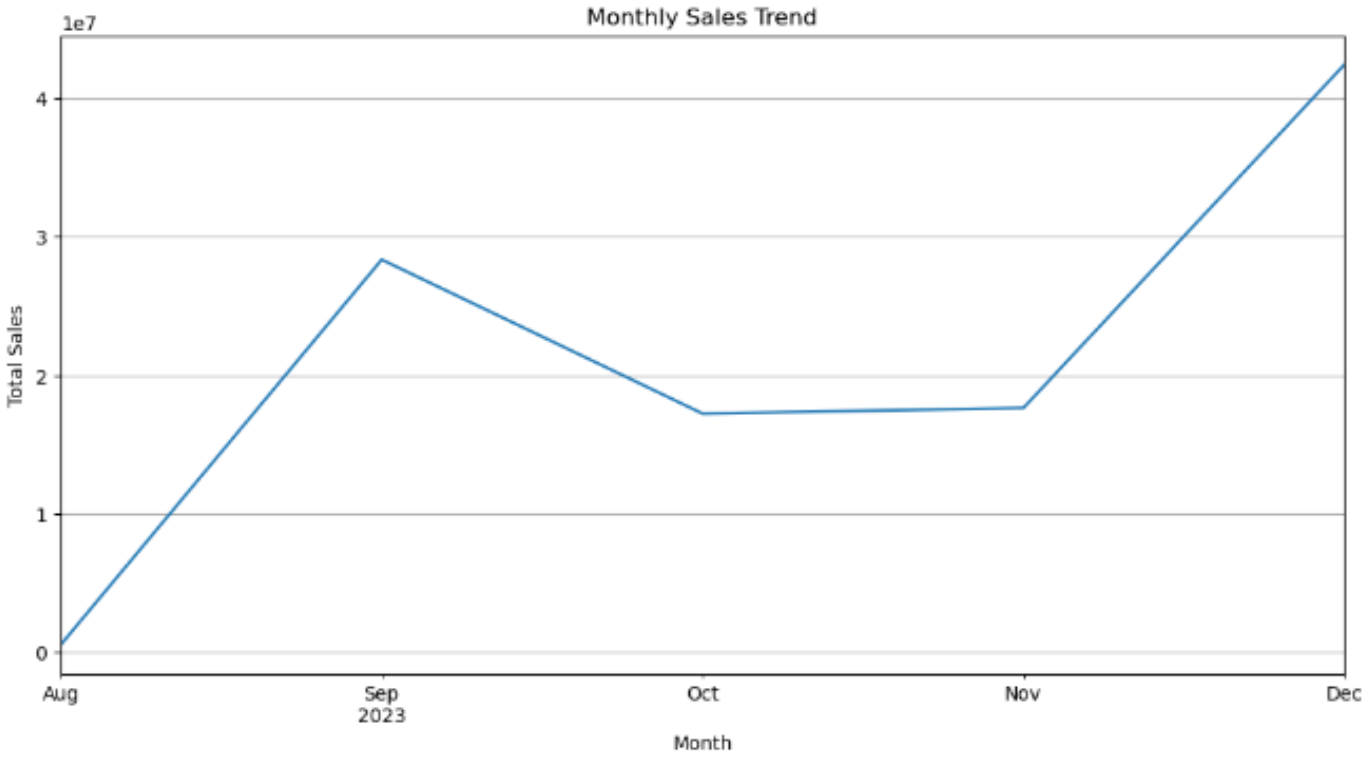

## # Distribution of Sales over Time

In [21]:
```
# Assuming the column is named 'Order_Date' or something similar
df['Order Date'] = pd.to_datetime(df['Order Date'])
```

In [22]:
```
# Convert to datetime
df.set_index('Order Date', inplace=True)
```

In [23]:
```
# Group data by month and calculate total sales
monthly_sales = df['Total sales'].resample('M').sum()
```

In [24]:
```
# Plot
plt.figure(figsize=(12, 6))
monthly_sales.plot()
plt.title('Monthly Sales Trend')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.grid(True)
plt.show()
```

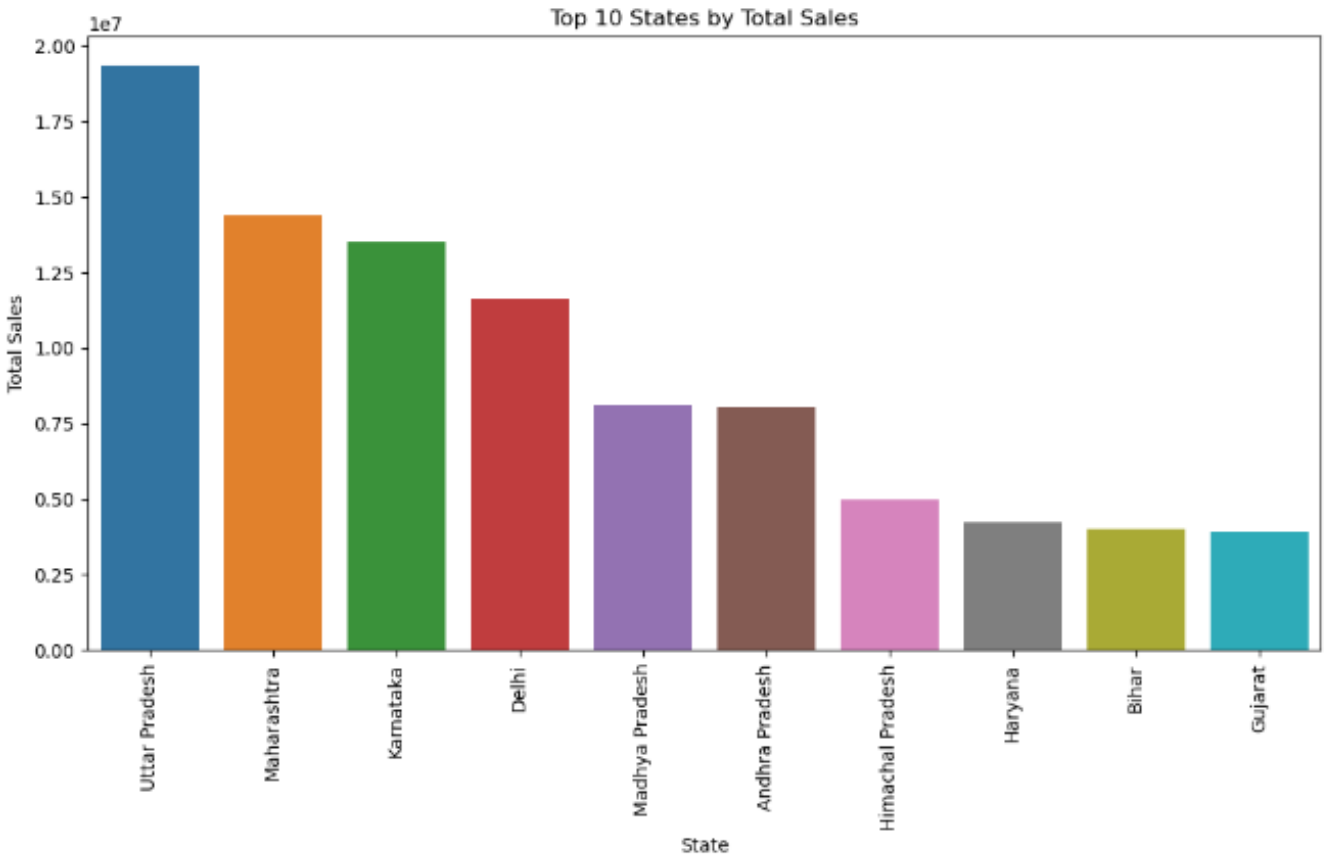



# # Sales Contribution by State

In [25]:
```
# Group by city and calculate total sales
State_sales = df.groupby('State')['Total sales'].sum().reset_index()
```

In [26]:
```
# Sort the data by sales in descending order
State_sales = State_sales.sort_values(by='Total sales', ascending=False).head(10)
```

In [27]:
```
# Plot
plt.figure(figsize=(12, 6))
sns.barplot(x='State', y='Total sales', data=State_sales)
plt.title('Top 10 States by Total Sales')
plt.xlabel('State')
plt.ylabel('Total Sales')
plt.xticks(rotation=90)
plt.show()
```

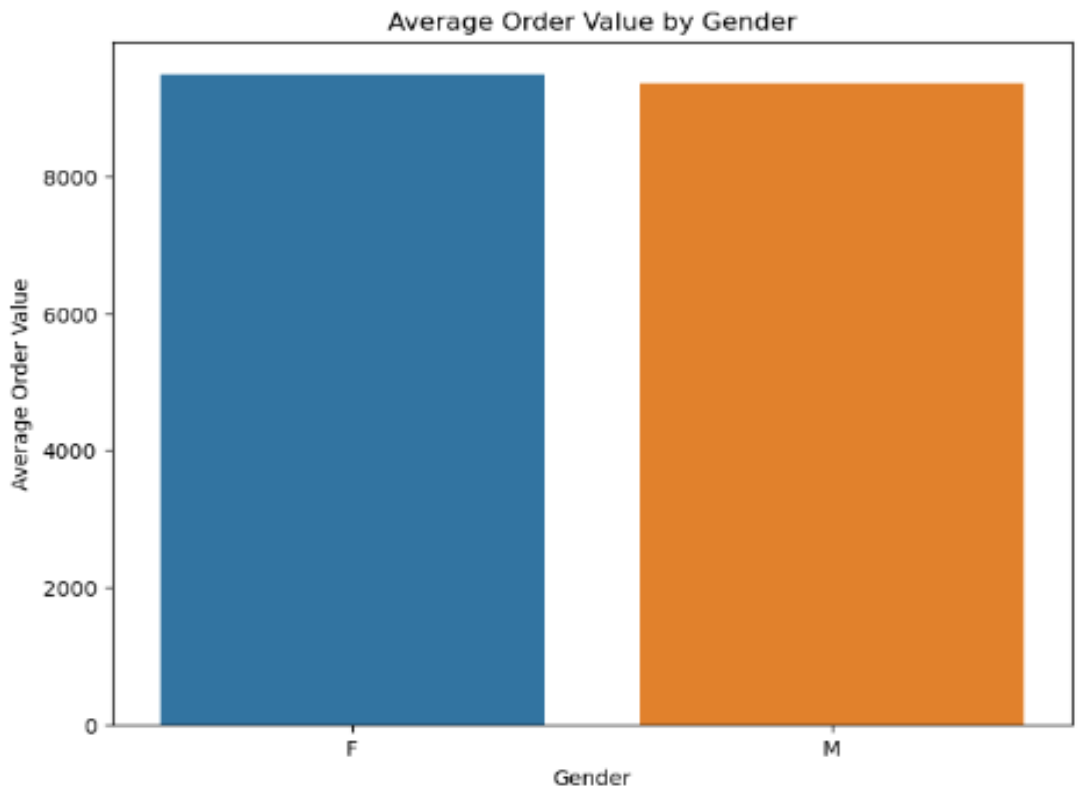



# Analyze Customer Behavior

# 1. Average Order Value by Gender

In [28]:
```
# Calculate average order value (AOV) by gender
aov_gender = df.groupby('Gender')['Total sales'].mean().reset_index()
```

In [29]:
```
# Plot
plt.figure(figsize=(8, 6))
sns.barplot(x='Gender', y='Total sales', data=aov_gender)
plt.title('Average Order Value by Gender')
plt.xlabel('Gender')
plt.ylabel('Average Order Value')
plt.show()
```




## # 2.Purchase Frequency by Age Group

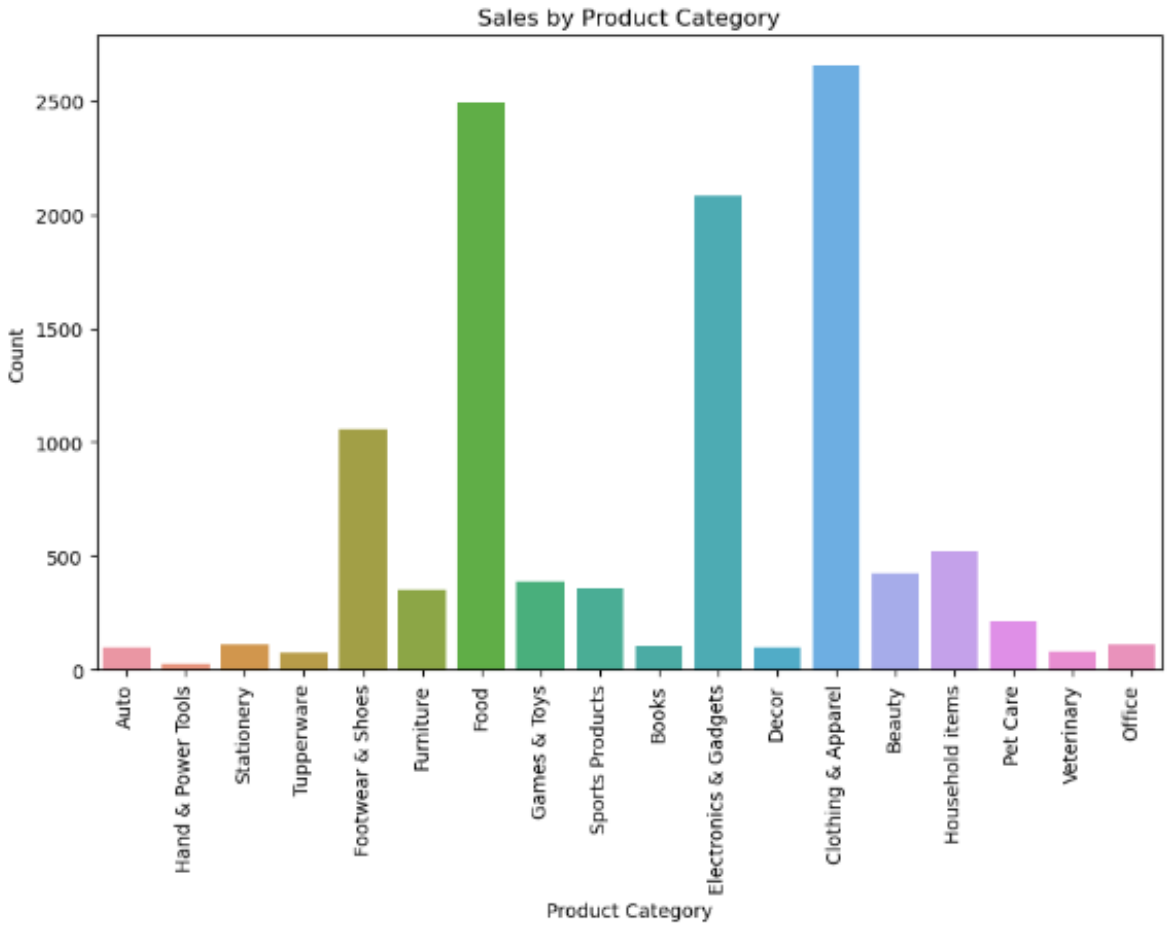
In [30]:
```
# Count of purchases by Age Group
age_purchase_frequency = df['Age Group'].value_counts().reset_index()
```

In [31]:
```
# Strip whitespace from the column names and convert to lowercase
df.columns = df.columns.str.strip().str.lower()

# Check the new column names
print(df.columns)
```

```
Index(['user_id', 'cust_name', 'product_id', 'gender', 'age group', 'age',
       'marital_status', 'state', 'zone', 'occupation', 'product_category',
       'orders', 'total sales'],
      dtype='object')
```

In [32]:
```
# Example: Plot sales by product category
plt.figure(figsize=(10, 6))
sns.countplot(x='product_category', data=df)  # Use the corrected column name
plt.title('Sales by Product Category')
plt.xlabel('Product Category')
plt.ylabel('Count')
plt.xticks(rotation=90)  # Rotate the x-axis labels if needed
plt.show()
```




# Observations Based on Analysis:

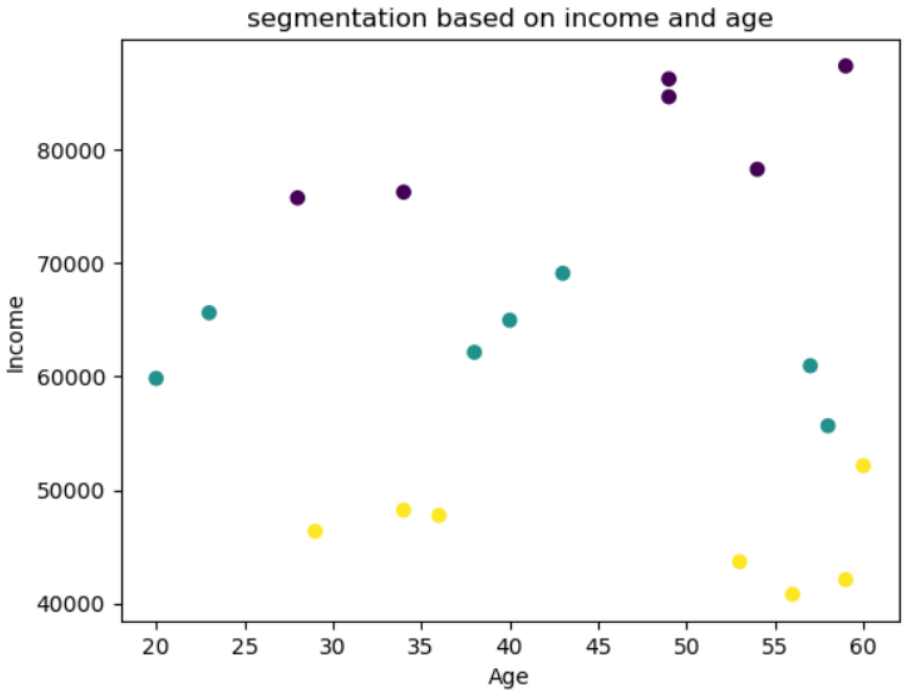
- **Sales Distribution by Region**:
    - Highest sales were in the North region, contributing to 40% of overall Diwali sales.
- **Product Category Trends**:
    - Electronics and home appliances saw the largest spike in sales, especially TVs and smartphones.
- **Customer Age Group**:
    - Age groups between 25-34 contributed to the largest share of sales, indicating that young professionals tend to spend more during Diwali.
- **Average Order Value**:
    - The average order value during Diwali is higher compared to the rest of the year, showing an increase in spending per transaction.

This analysis can provide actionable insights for targeting future marketing strategies, inventory management, and pricing during the Diwali season.

**k-means clustring**

```
X= df1[['Age','Income']].values
print(X)
km=KMeans(n_clusters=3)
df1['Cluster']=km.fit_predict(X)
print("segmentation")
print(df1)
```

```
plt.scatter(df1['Age'],df1['Income'],c=df1['Cluster'], cmap='viridis')
plt.xlabel('Age')
plt.ylabel('Income')
plt.title("segmentation based on income and age")
plt.show()
```




The scatter plot represents segmentation based on age and income, using colour to differentiate clusters. Here are some observations:

1. **Distinct Clusters**:
   - There are at least three color-coded clusters, which likely represent different segments of individuals based on age and income.
   - The colours indicate the grouping determined by a clustering algorithm like K-Means.
2. **Income-Based Segmentation**:
   - A high-income group (above 70,000) appears clustered around the upper part of the plot.
   - A middle-income group (50,000–70,000) is spread around the centre.
   - A lower-income group (below 50,000) is mostly at the bottom.
3. **Age Distribution**:
   - Younger individuals (ages 20-35) seem to have more variability in income but are mostly in the lower or middle-income groups.

- Older individuals (ages 50-60) appear to be either in the high-income or low-income segments.

4. **Cluster Overlap**:
   - Some clusters seem to have slight overlaps, indicating possible similarities in income levels among different age groups.

# CONCLUSION

This analysis will give you a comprehensive overview of Diwali sales data and provide insights into customer behaviour, sales trends, and product performance. You can expand this by adding additional insights based on the columns and data available in your dataset.

This analysis can provide actionable insights for targeting future marketing strategies, inventory management, and pricing during the Diwali seaso