Diwali Sales Analysis

Problem Statement

Analyze the given Diwali sales data to understand customer purchasing behavior based on demographics (age group, marital status, gender), geographic location, and product categories. Based on this analysis, provide actionable recommendations to optimize sales strategies and enhance performance during the Diwali festival.

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

In [2]:
```
df = pd.read_csv(r"D:\Data\Hira Siddiqui\Downloads\Diwali Sales Data.csv", encoding = 'unicode_escape')
df
```

Out[2]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	F
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	
...	...	...	...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	

11251 rows × 15 columns

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

Out[5]: (11251, 15)

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   User_ID           11251 non-null  int64  
 1   Cust_name         11251 non-null  object 
 2   Product_ID        11251 non-null  object 
 3   Gender            11251 non-null  object 
 4   Age Group         11251 non-null  object 
 5   Age               11251 non-null  int64  
 6   Marital_Status    11251 non-null  int64  
 7   State             11251 non-null  object 
 8   Zone              11251 non-null  object 
 9   Occupation        11251 non-null  object 
 10  Product_Category  11251 non-null  object 
 11  Orders            11251 non-null  int64  
 12  Amount            11239 non-null  float64
 13  Status            0 non-null      float64
 14  unnamed1          0 non-null      float64
dtypes: float64(3), int64(4), object(8)
memory usage: 1.3+ MB
```

In [9]:
```
df.drop(['Status','unnamed1'], inplace = True, axis = 1 )
```

In [13]:
```
df.isnull().sum()
```

Out[13]:
```
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
Orders               0
Amount              12
dtype: int64
```

In [15]:
```
df.dropna(inplace = True)
```

In [17]:
```
df.isnull().sum()
```

Out[17]:
```
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
Orders              0
Amount              0
dtype: int64
```

In [19]:
```
df['Amount'] = df['Amount'].astype(int)
```

In [21]:
```
df['Amount'].dtypes
```

Out[21]:
```
dtype('int32')
```

In [23]:
```
df[['Amount','Age','Orders']].describe()
```

Out[23]:

	Amount	Age	Orders
count	11239.000000	11239.000000	11239.000000
mean	9453.610553	35.410357	2.489634
std	5222.355168	12.753866	1.114967
min	188.000000	12.000000	1.000000
25%	5443.000000	27.000000	2.000000
50%	8109.000000	33.000000	2.000000
75%	12675.000000	43.000000	3.000000
max	23952.000000	92.000000	4.000000

In [25]:
```
df
```

Out[25]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	P
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	
...	...	...	...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	

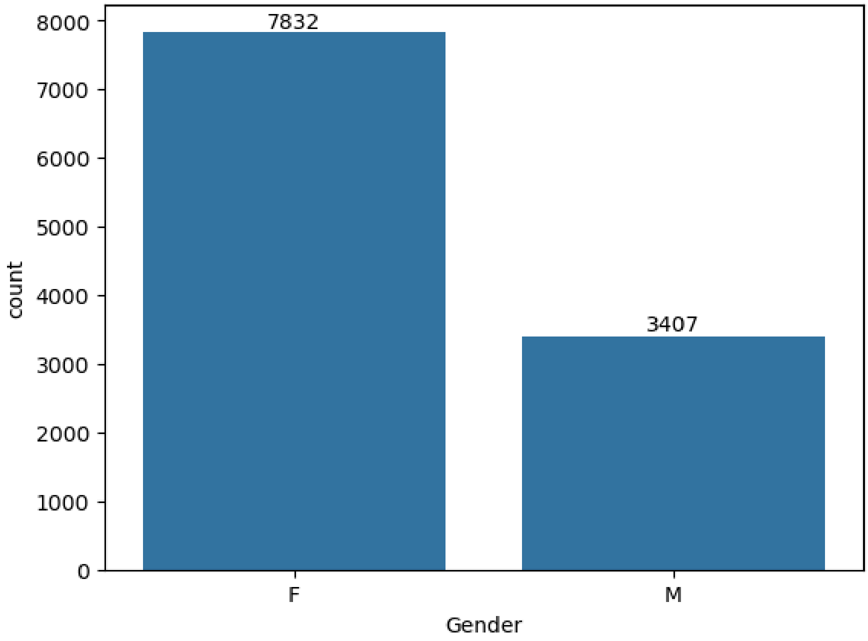
11239 rows × 13 columns

Exploaratory Data Analysis

In [28]:
```
df.columns
```

Out[28]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [30]:
```
gender_count = sns.countplot(x= 'Gender', data = df)

for x in gender_count.containers:
    gender_count.bar_label(x)
```



In [31]:
```
grouped = df.groupby('Gender', as_index=False)['Amount'].sum()
grouped
```

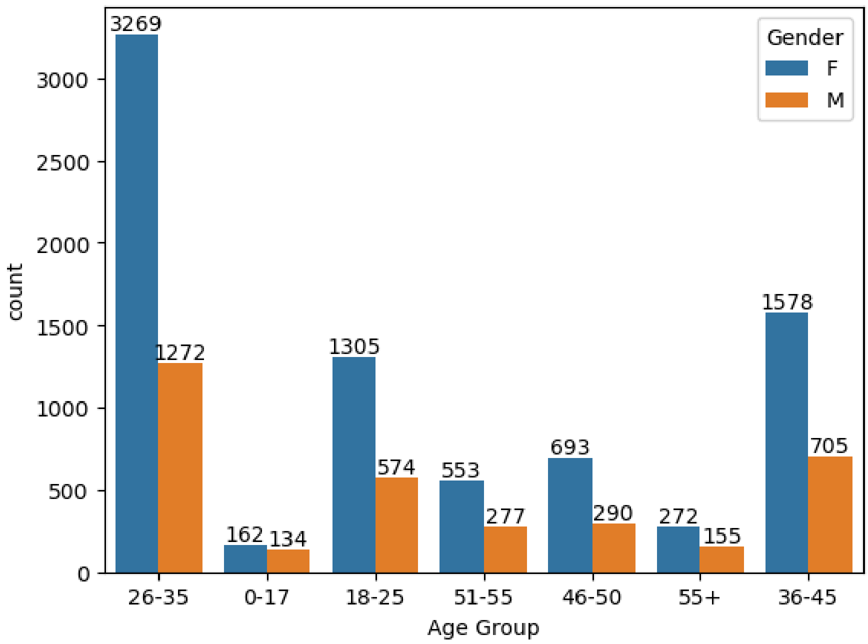
Out[31]:

	Gender	Amount
0	F	74335853
1	M	31913276

From the above graph, we can see which gender has mostly acquired in the sales

In [109…
```
# sales_amount = df.groupby(['Gender'],as_index = False)['Amount'].sum().sort_values(by = 'Amount', ascending=

# sns.barplot(x='Gender', y='Amount', data=sales_amount, palette='viridis',hue='Gender')
# plt.tight_layout()
# plt.title('Total Sales Amount by Gender and Category')
# # plt.xlabel('Gender')
# #plt.ylabel('Total Sales Amount')
# plt.xticks(rotation=45)
```

In [36]:
```
gender_count = sns.countplot(x= 'Age Group', data = df, hue = 'Gender')

for x in gender_count.containers:
    gender_count.bar_label(x)
```



The above graph shows the count of the gender according to Age Group.

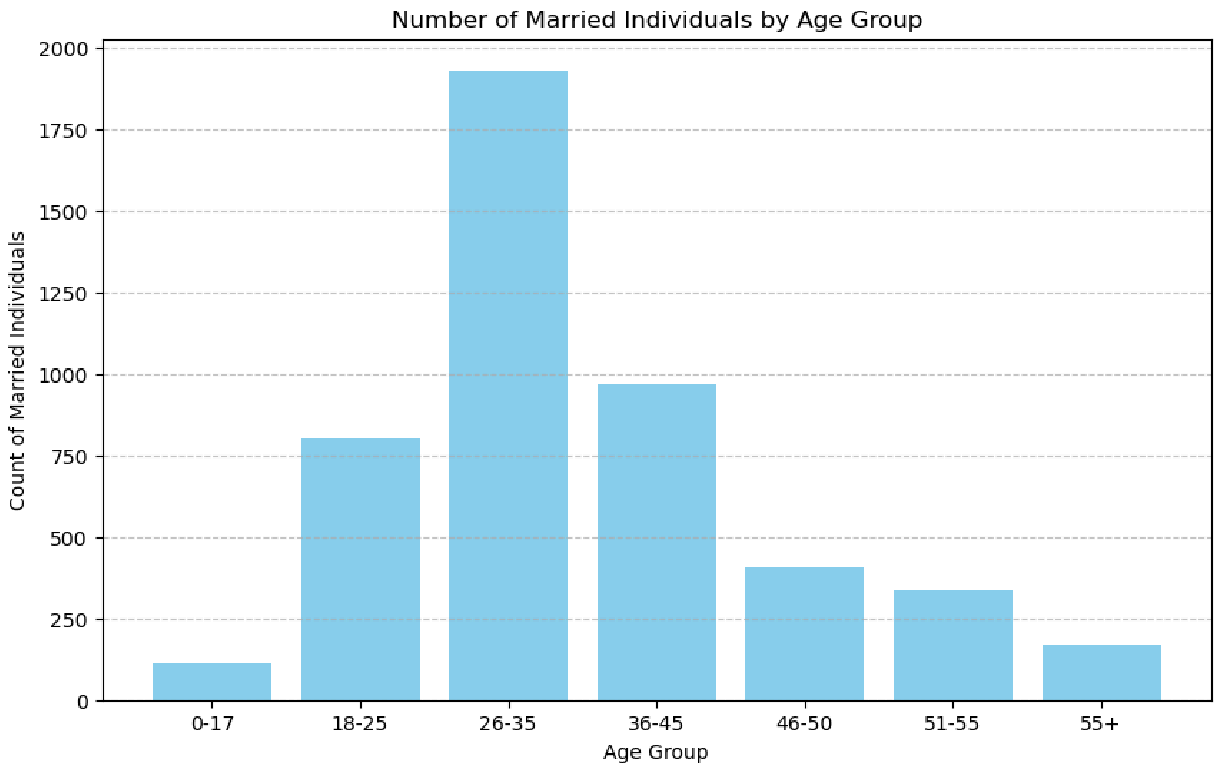
In [86]:
```
data_by_group = pd.DataFrame(df)
data_by_group
```

Out[86]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	F
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	
...	...	...	...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	

11239 rows × 13 columns

In [96]:
```
age_marital_status = data_by_group[['Age Group', 'Marital_Status']]
aggregated_data = age_marital_status.groupby('Age Group')['Marital_Status'].sum().reset_index()

plt.figure(figsize=(10, 6))
plt.bar(aggregated_data['Age Group'], aggregated_data['Marital_Status'], color='skyblue')
plt.xlabel('Age Group')
plt.ylabel('Count of Married Individuals')
plt.title('Number of Married Individuals by Age Group')
plt.grid(axis='y', linestyle='--', alpha=0.7)
```



In [107…
```
# age_group_sales = df.groupby(['Age Group'],as_index = False)['Amount'].sum().sort_values(by = 'Amount', asce
```

```
# sns.barplot(x ='Age Group' , y='Amount' , data = age_group_sales)
```

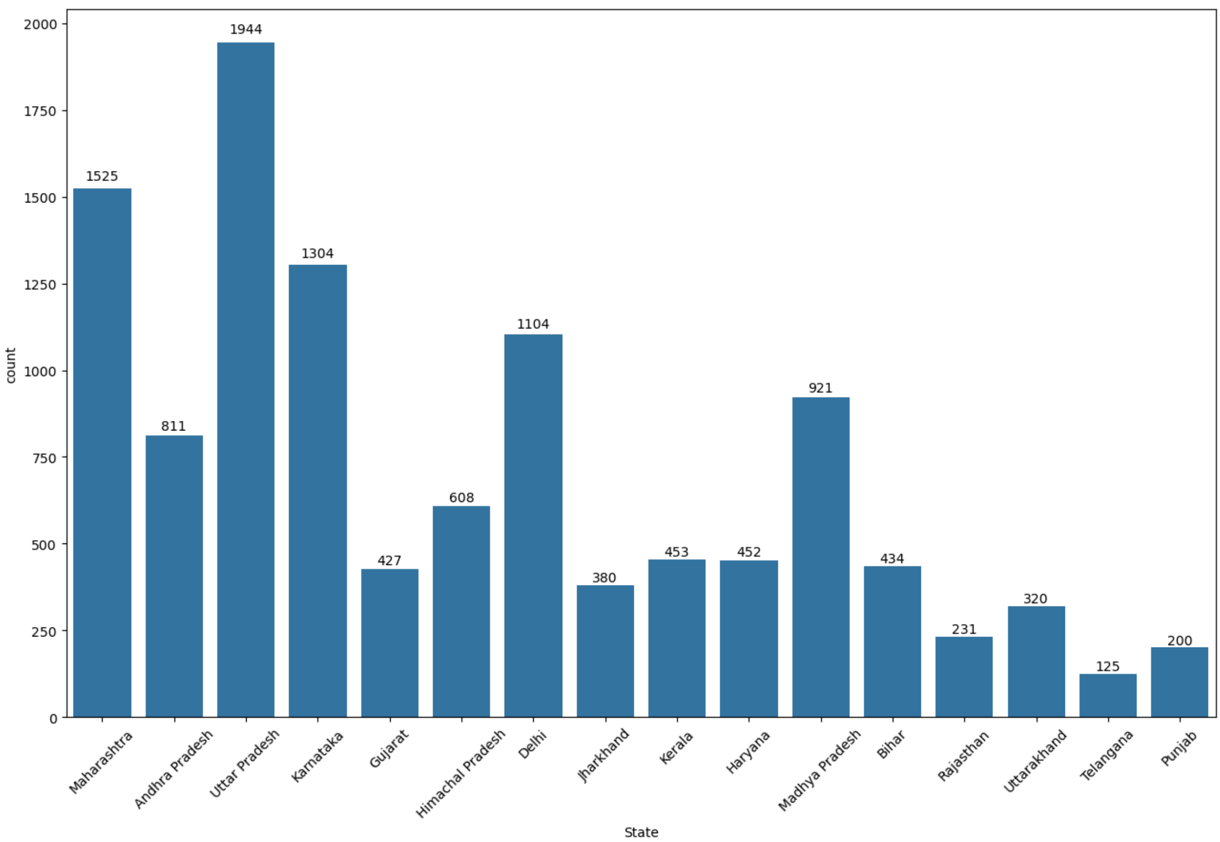
In [41]:
```
product_by_state = df[['State','Product_Category']]
product_by_state
```

Out[41]:

	State	Product_Category
0	Maharashtra	Auto
1	Andhra Pradesh	Auto
2	Uttar Pradesh	Auto
3	Karnataka	Auto
4	Gujarat	Auto
...	...	...
11246	Maharashtra	Office
11247	Haryana	Veterinary
11248	Madhya Pradesh	Office
11249	Karnataka	Office
11250	Maharashtra	Office

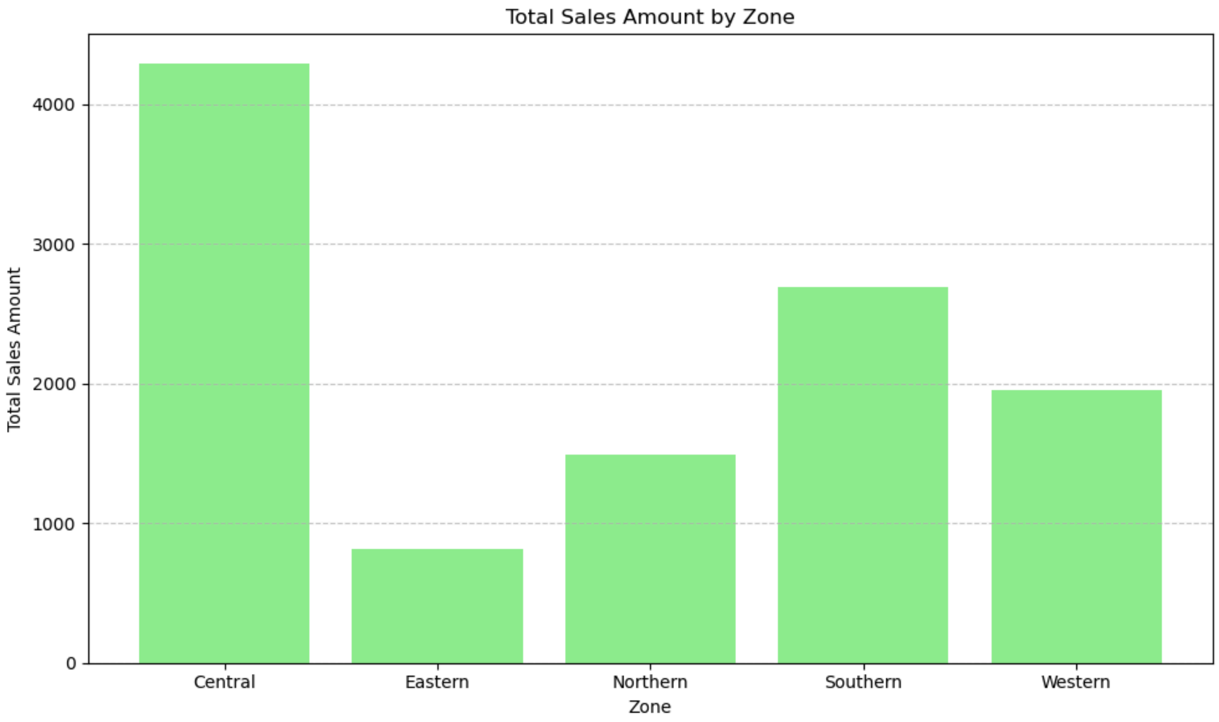
11239 rows × 2 columns

In [43]:
```
data = product_by_state

plt.figure(figsize=(13, 9))
ax = sns.countplot(x='State', data=product_by_state)

for bar in ax.patches:
    height = bar.get_height()
    ax.text(
        bar.get_x() + bar.get_width() / 2,
        height + 0.01 * height,
        int(height),  ha='center',  va='bottom')
plt.xticks(rotation=45)
plt.tight_layout()
```

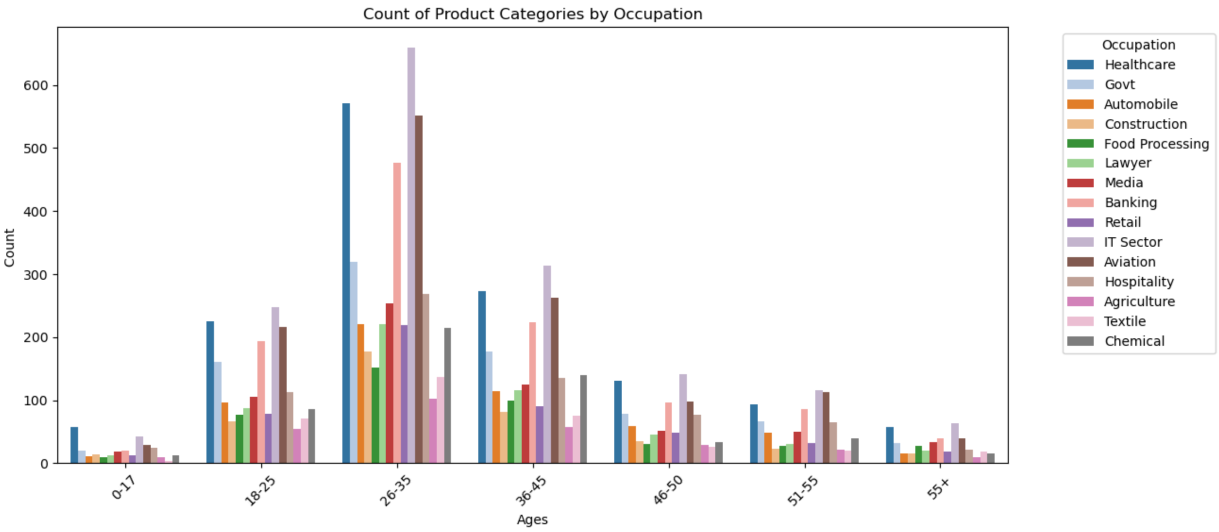


The above insight shows how many product categories were purchases by States. The top 3 most states are Uttar Pradesh, Maharashtra and Karnataka

In [115…
```
sales_by_zone = pd.DataFrame(df)
zone_sales = sales_by_zone.groupby('Zone')['Amount'].count().reset_index()

plt.figure(figsize=(10, 6))
plt.bar(zone_sales['Zone'], zone_sales['Amount'], color='lightgreen')
plt.xlabel('Zone')
plt.ylabel('Total Sales Amount')
plt.title('Total Sales Amount by Zone')
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
```

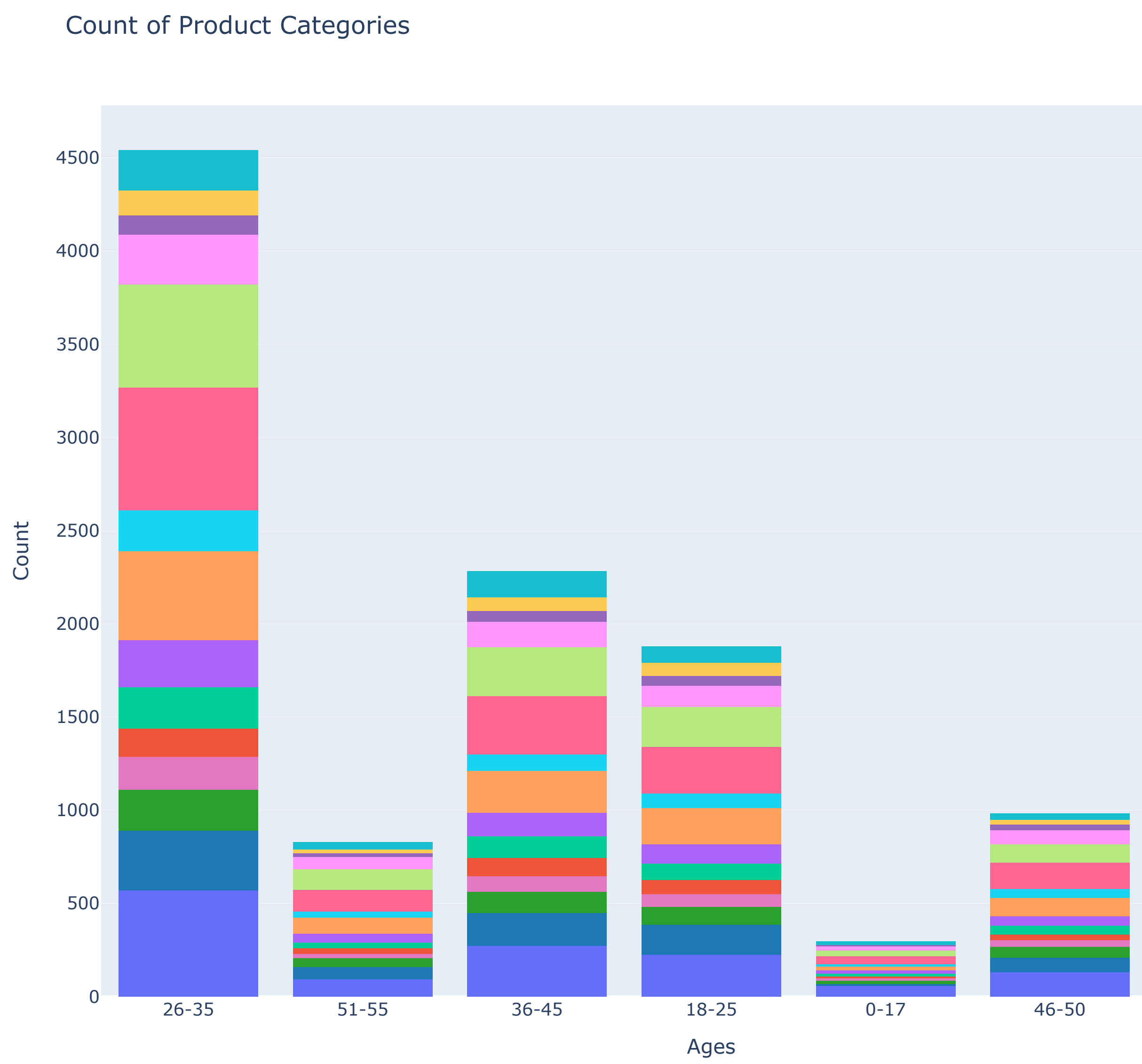


In [49]:
```
dat = df[['Age Group','Occupation']]
dayta = pd.DataFrame(dat)

age_order = sorted(dayta['Age Group'].unique())
dayta['Age Group'] = pd.Categorical(dayta['Age Group'], categories=age_order, ordered=True)

palette = sns.color_palette("Set2")

plt.figure(figsize=(13, 6))
sns.countplot(data=dayta, x='Age Group',  hue='Occupation', palette=sns.color_palette("tab20", n_colors=15))
plt.title('Count of Product Categories by Occupation')
plt.xlabel('Ages')
plt.ylabel('Count ')
plt.legend(title='Occupation',  bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.show()
```



The above insight shows products purchased by Age Groups. Here we can see the 26-35 age group has highest purchases. In 26-35 age group, the people who are in Govt occupation have the highest purchases

In [51]:
```
dat = pd.DataFrame(dat)

pastel_colors = ['#636EFA','#1f77b4','#2ca02c','#e377c2','#EF553B', '#00CC96', '#AB63FA', '#FFA15A', '#19D3F3'
                 '#FECB52', '#17becf', '#8c564b']

fig = px.histogram(dayta, x='Age Group', color='Occupation', title='Count of Product Categories',
                   color_discrete_sequence=pastel_colors, labels={'x': 'Ages', 'count': 'Count'})

fig.update_layout(
    xaxis_title='Ages',  yaxis_title='Count',    #xaxis_tickangle=-45,
     width=1100,    height=800)
```



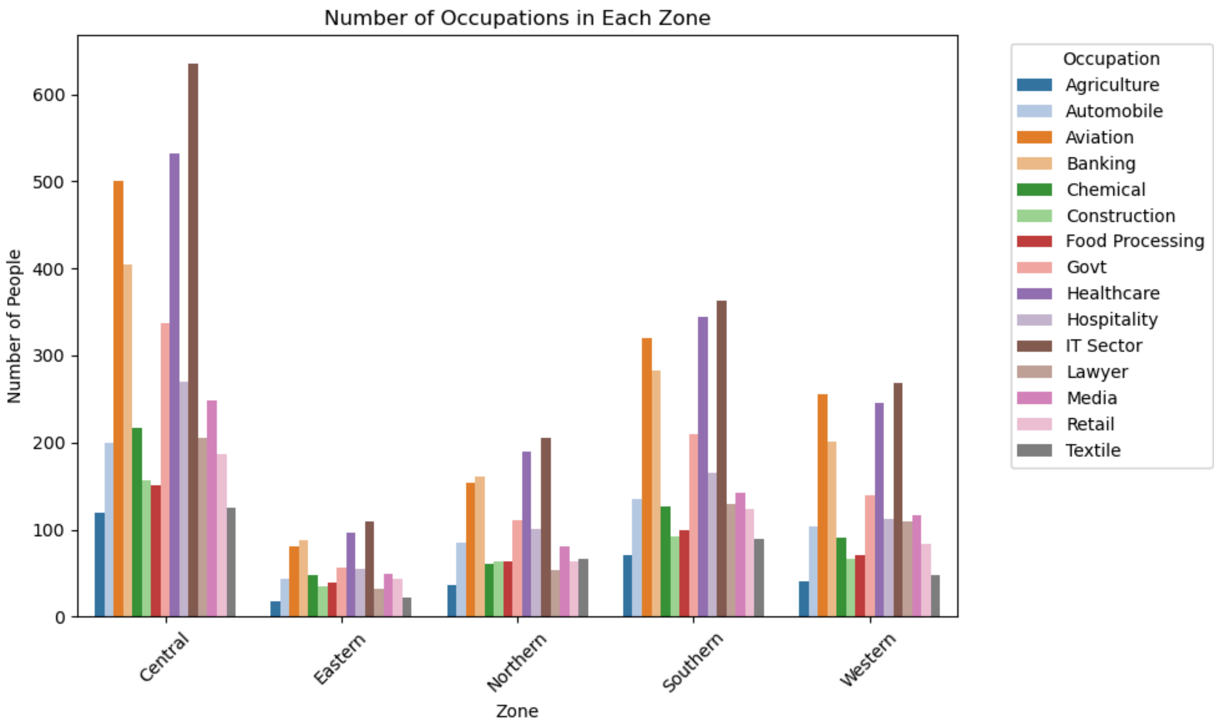
In [52]:
```
occupation_by_zone = df[['Occupation','Zone']]
occupation_by_zone
```

Out[52]:

	Occupation	Zone
0	Healthcare	Western
1	Govt	Southern
2	Automobile	Central
3	Construction	Southern
4	Food Processing	Western
...	...	...
11246	Chemical	Western
11247	Healthcare	Northern
11248	Textile	Central
11249	Agriculture	Southern
11250	Healthcare	Western

11239 rows × 2 columns

In [56]:
```
occupation_count = occupation_by_zone.groupby(['Zone', 'Occupation']).size().reset_index(name='Count')

plt.figure(figsize=(10, 6))

#palette = sns.color_palette("Set2")
sns.barplot(data=occupation_count, x='Zone', y='Count', hue='Occupation', palette=sns.color_palette("tab20", n
plt.title('Number of Occupations in Each Zone')
plt.xlabel('Zone')
plt.ylabel('Number of People')
plt.legend(title='Occupation',  bbox_to_anchor=(1.05, 1), loc='upper left')
plt.xticks(rotation=45)
plt.tight_layout()
```

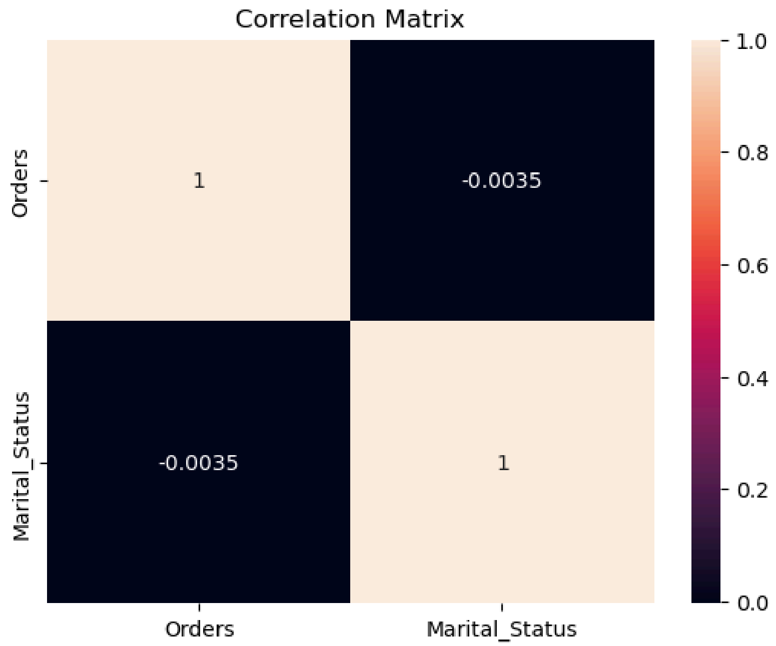


The above insights show that which occupation from a specific zone has the highest influence.

1. Zones with a high number of IT professionals or business executives have greater spending potential during Diwali.
2. Invest more in advertising and special promotions in zones with a significant presence of high-income occupations to maximize return on investment.

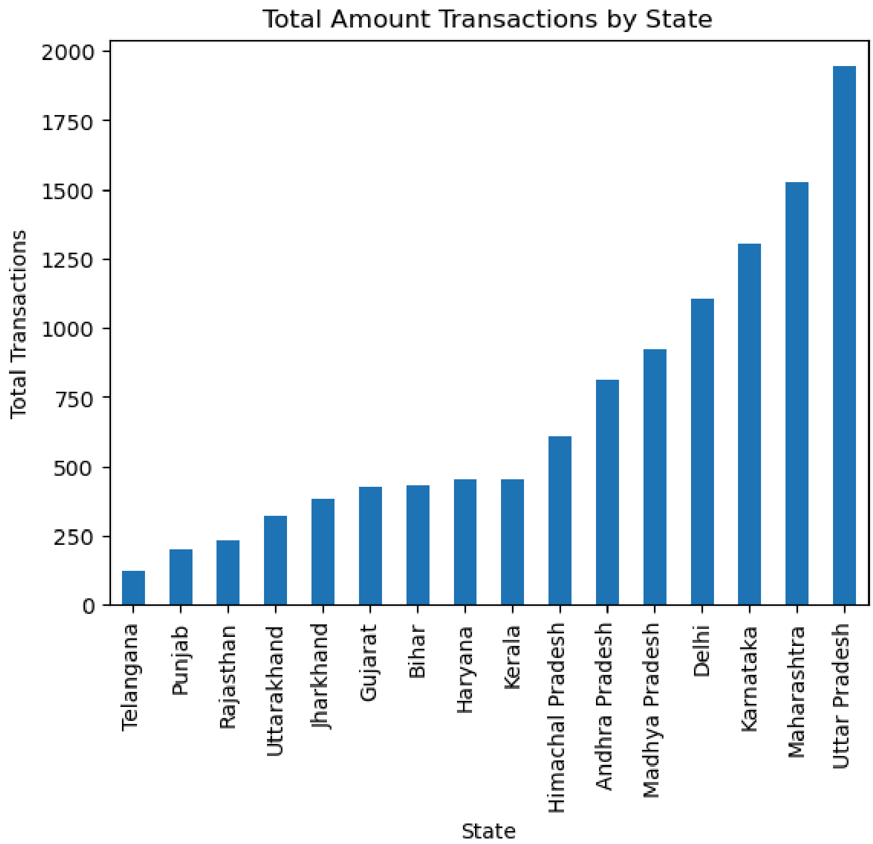
In [182…
```
# sns.scatterplot(data=df, x='Orders', y='Amount')
# plt.title('Orders vs Amount Spent')
# plt.xlabel('Number of Orders')
# plt.ylabel('Amount Spent')
# plt.show()
```

In [94]:
```
correlation_matrix = df[[ 'Orders', 'Marital_Status']].corr()
sns.heatmap(correlation_matrix, annot=True)
plt.title('Correlation Matrix')
plt.show()
```



In [60]:
```
state_amounts = df.groupby('State')['Amount'].count().sort_values()
state_amounts.plot(kind='bar')
plt.title('Total Amount Transactions by State')
plt.xlabel('State')
plt.ylabel('Total Transactions')
```

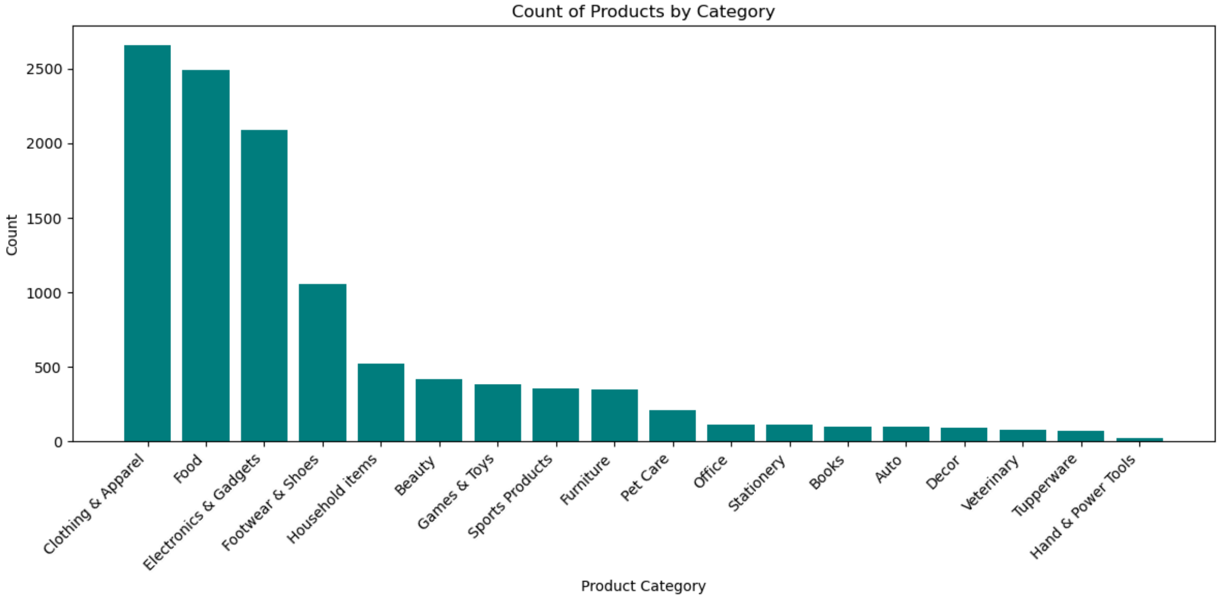
Out[60]: `Text(0, 0.5, 'Total Transactions')`



The above insight shows States with high transaction volumes indicate strong customer activity and engagement, marking them as key markets. Conversely, states with fewer transactions may have lower engagement or smaller market presence, warranting further analysis to understand potential causes.

In [177…
```
product_by_age = pd.DataFrame(df)

category_counts = product_by_age['Product_Category'].value_counts().reset_index()
category_counts.columns = ['Product_Category', 'Age']
```

```
plt.figure(figsize=(12, 6))
plt.bar(category_counts['Product_Category'], category_counts['Age'], color='teal')

plt.title('Count of Products by Category')
plt.xlabel('Product Category')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
```



From the above analysis, we can see that 'Clothing & Apparel', 'Food', and 'Electronics & Gadgets' have the highest sales counts.

RECOMMENDATIONS

1. Conduct further research to evaluate the market potential. Assess if the low transaction volume is due to limited customer demand or other factors such as competition or market conditions.
2. Implement loyalty programs and personalized offers to boost engagement and repeat transactions.
3. To capitalize on high sales for 'Clothing & Apparel', 'Food', and 'Electronics & Gadgets', implement targeted promotions and discounts, such as seasonal sales and bundle offers. Enhance product visibility through prominent placement and focused advertising on relevant platforms like social media and tech blogs. Additionally, engage customers with loyalty programs and optimized online experiences to drive repeat business and attract new customers.
4. To boost sales in lower-performing categories like 'Stationery', 'Books', and 'Decor', implement targeted promotions and special offers to increase visibility. Focus on niche marketing and personalized recommendations to attract and engage customers effectively.

In [ ]: