In [1]:
```
import numpy as np
import pandas as pd
import os
for dirname, _, filenames in os.walk('/kaggle/input/diwali-sales-dataset'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/diwali-sales-dataset/Diwali Sales Data.csv
```

In [2]:
```
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as mso
import matplotlib.ticker as ticker

# Stats
from statsmodels.graphics.gofplots import qqplot
from scipy.stats import shapiro, norm
```

In [3]:
```
df = pd.read_csv("/kaggle/input/diwali-sales-dataset/Diwali Sales Data.csv", encoding='unicode_escape')
```

Data Cleaning

In [4]:
```
df.head()
```

Out[4]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupati
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthca
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	G
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automob
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Constructi
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Fo Processi

In [5]:
```
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 [6]:
```
df.isna().sum()
```

Out[6]:
```
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
Status             11251
unnamed1           11251
dtype: int64
```

In [7]:
```
df.drop(columns=['User_ID', 'Product_ID', 'Status', 'unnamed1'], inplace=True)
```

In [8]:
```
df.isna().sum()
```

Out[8]:
```
Cust_name           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
```

Statistics

In [9]:
```
df.describe()
```

Out[9]:

	Age	Marital_Status	Orders	Amount
count	11251.000000	11251.000000	11251.000000	11239.000000
mean	35.421207	0.420318	2.489290	9453.610858
std	12.754122	0.493632	1.115047	5222.355869
min	12.000000	0.000000	1.000000	188.000000
25%	27.000000	0.000000	1.500000	5443.000000
50%	33.000000	0.000000	2.000000	8109.000000
75%	43.000000	1.000000	3.000000	12675.000000
max	92.000000	1.000000	4.000000	23952.000000

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

Out[10]:

	Cust_name	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category
0	Sanskriti	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto
1	Kartik	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto
2	Bindu	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto
3	Sudevi	M	0-17	16	0	Karnataka	Southern	Construction	Auto
4	Joni	M	26-35	28	1	Gujarat	Western	Food Processing	Auto

In [11]:
```
# We selected numerical and categorical variables to perform the analysis.
cols_num = df.select_dtypes(include = ['float','int']).columns.to_list()
cols_cat = df.select_dtypes(include = ['object', 'category']).columns.to_list()
```

In [12]:
```
# Now let's look at the kurtosis and skewness of each variable.
for col in cols_num:
    print('==' * 30)
    print(f'Variable: {col}\n')
    print(f'Skew = {df[col].skew()}')
    print(f'Kurtosis = {df[col].kurt()}')
    print('==' * 30)
    print('\n')
```

```
============================================================
Variable: Age

Skew = 1.183203633076307
Kurtosis = 2.4642790073598126
============================================================


============================================================
Variable: Marital_Status

Skew = 0.3228963546847678
Kurtosis = -1.896075025783324
============================================================


============================================================
Variable: Orders

Skew = 0.0195789723936794
Kurtosis = -1.3527130581443787
============================================================


============================================================
Variable: Amount

Skew = 0.5580257366658404
Kurtosis = -0.5402092965421805
============================================================
```

1.Age: Skewness (a measure of asymmetry) is 1.183, indicating that the age distribution is right-skewed (positively skewed), with a tail extending to the right. This suggests that there might be relatively fewer older customers. Kurtosis (a measure of the peakedness of the distribution) is 2.464, indicating a leptokurtic distribution. The distribution has heavier tails and a sharper peak compared to a normal distribution.

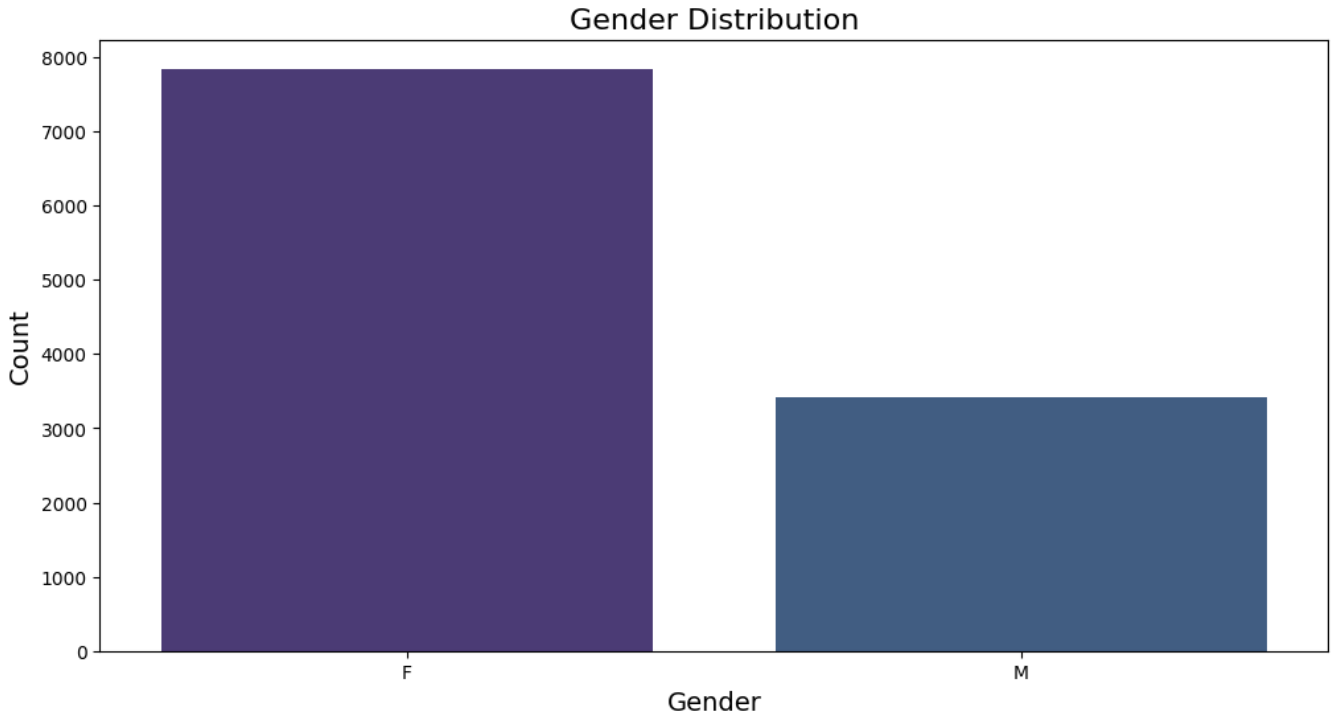
2.Marital_Status: Skewness is 0.323, indicating a slight right skew, but the skew is less pronounced than in the Age variable. Kurtosis is -1.896, indicating a platykurtic distribution. The distribution is flatter and has lighter tails compared to a normal distribution.

3.Orders: Skewness is 0.020, indicating a nearly symmetrical distribution with a very slight right skew. The distribution is close to being symmetric. Kurtosis is -1.353, indicating a platykurtic distribution, similar to Marital_Status.

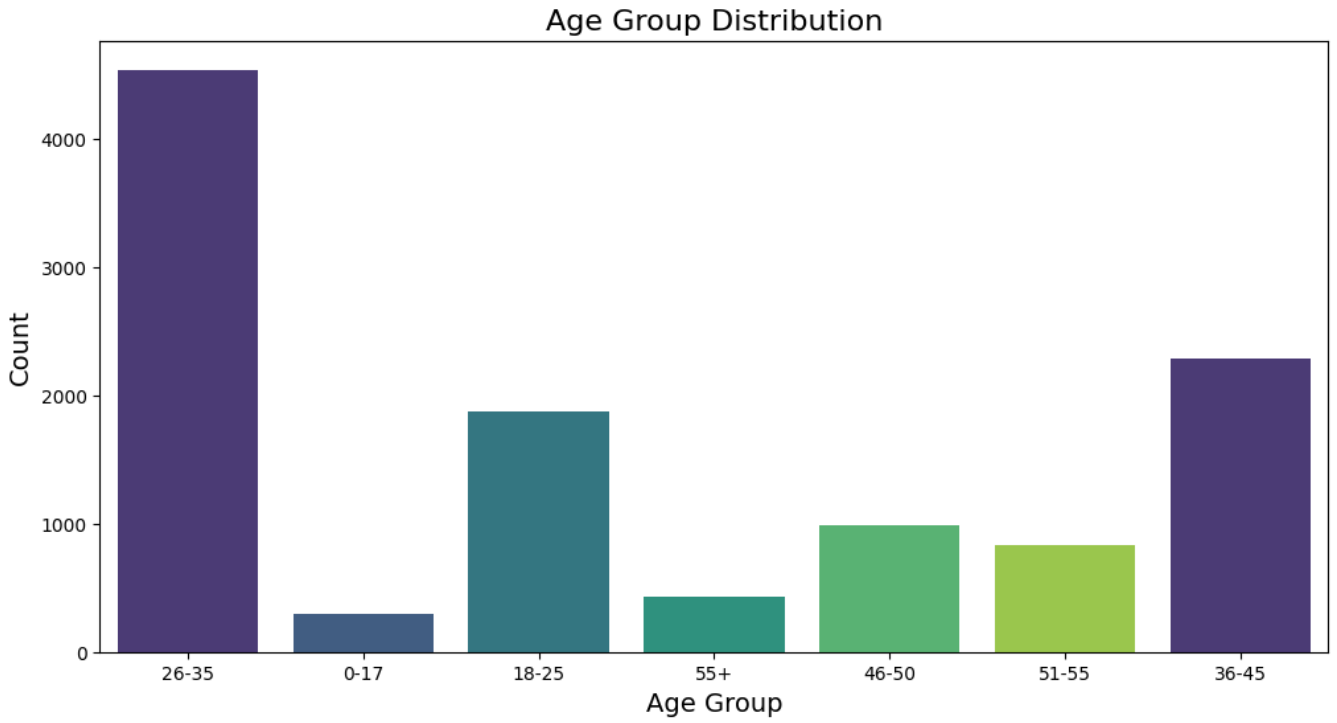
4.Amount: Skewness is 0.558, indicating a right skew, but the skewness is relatively moderate. Kurtosis is -0.540, indicating a platykurtic distribution similar to Marital_Status.

EDA

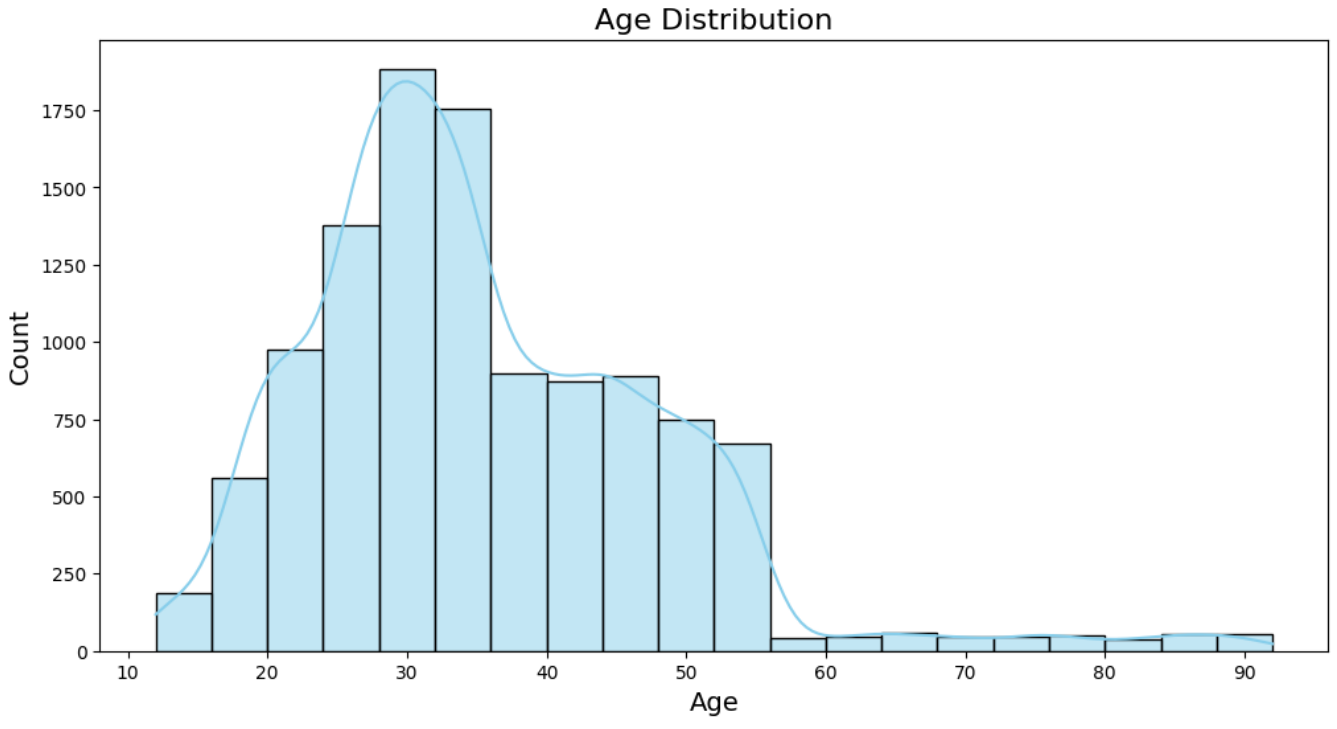
In [13]:
```
# Set a custom color palette
custom_palette = sns.color_palette("viridis")

# Set a larger chart size
plt.figure(figsize=(12, 6))

# Gender (Categorical data) - Countplot
sns.countplot(data=df, x="Gender", palette=custom_palette)
plt.title("Gender Distribution", fontsize=16)
plt.xlabel("Gender", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```



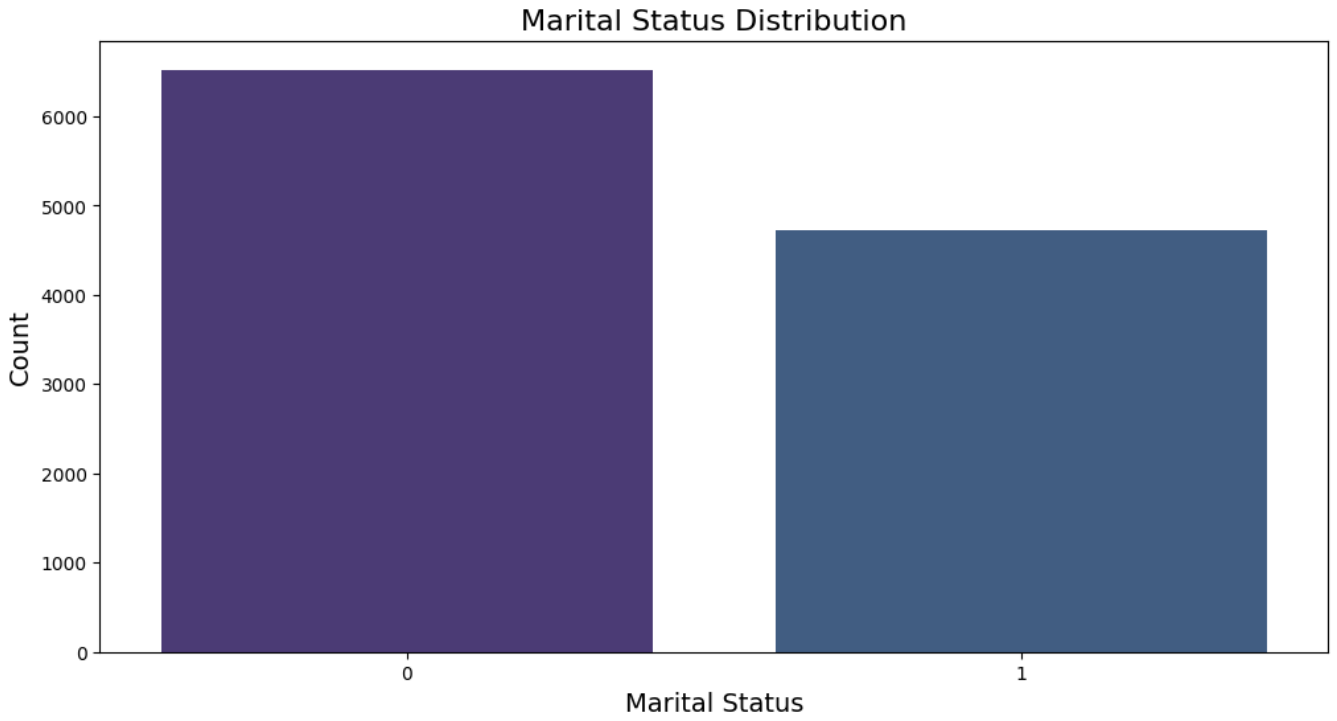
In [14]:
```
# Age Group (Categorical data) - Countplot
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Age Group", palette=custom_palette)
plt.title("Age Group Distribution", fontsize=16)
plt.xlabel("Age Group", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```



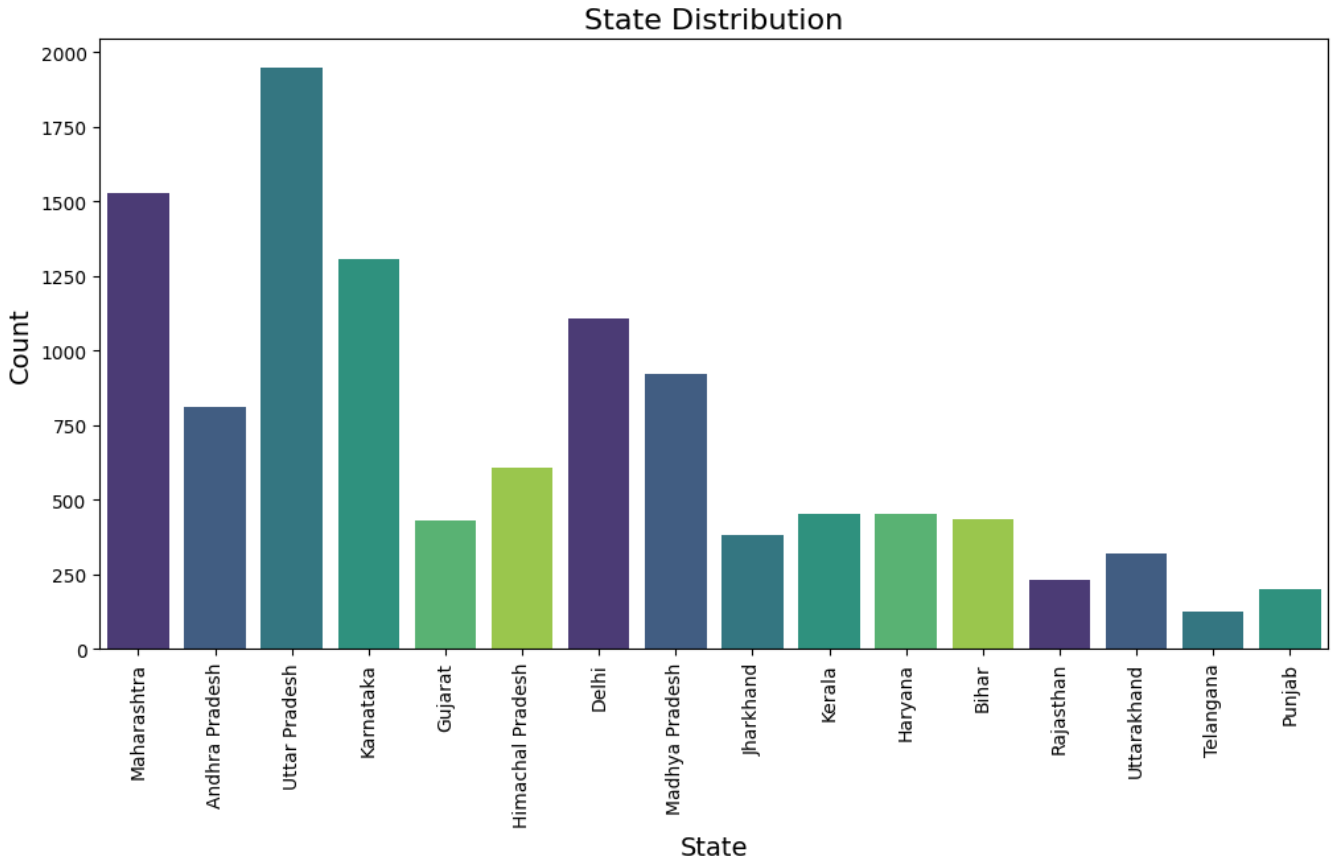
In [15]:
```
# Age (Numerical data) - Histogram
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x="Age", bins=20, kde=True, color='skyblue')
plt.title("Age Distribution", fontsize=16)
plt.xlabel("Age", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```



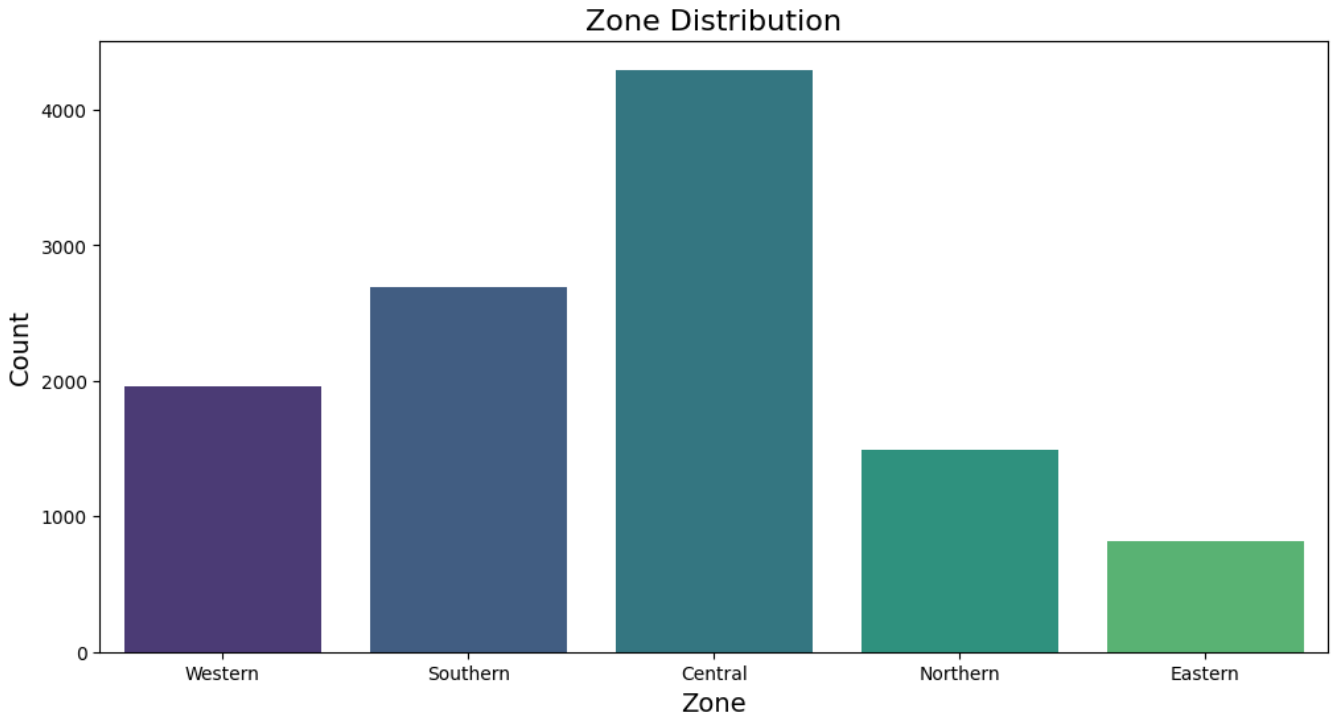
In [16]:
```
# Marital_Status (Categorical data) - Countplot
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Marital_Status", palette=custom_palette)
plt.title("Marital Status Distribution", fontsize=16)
plt.xlabel("Marital Status", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```



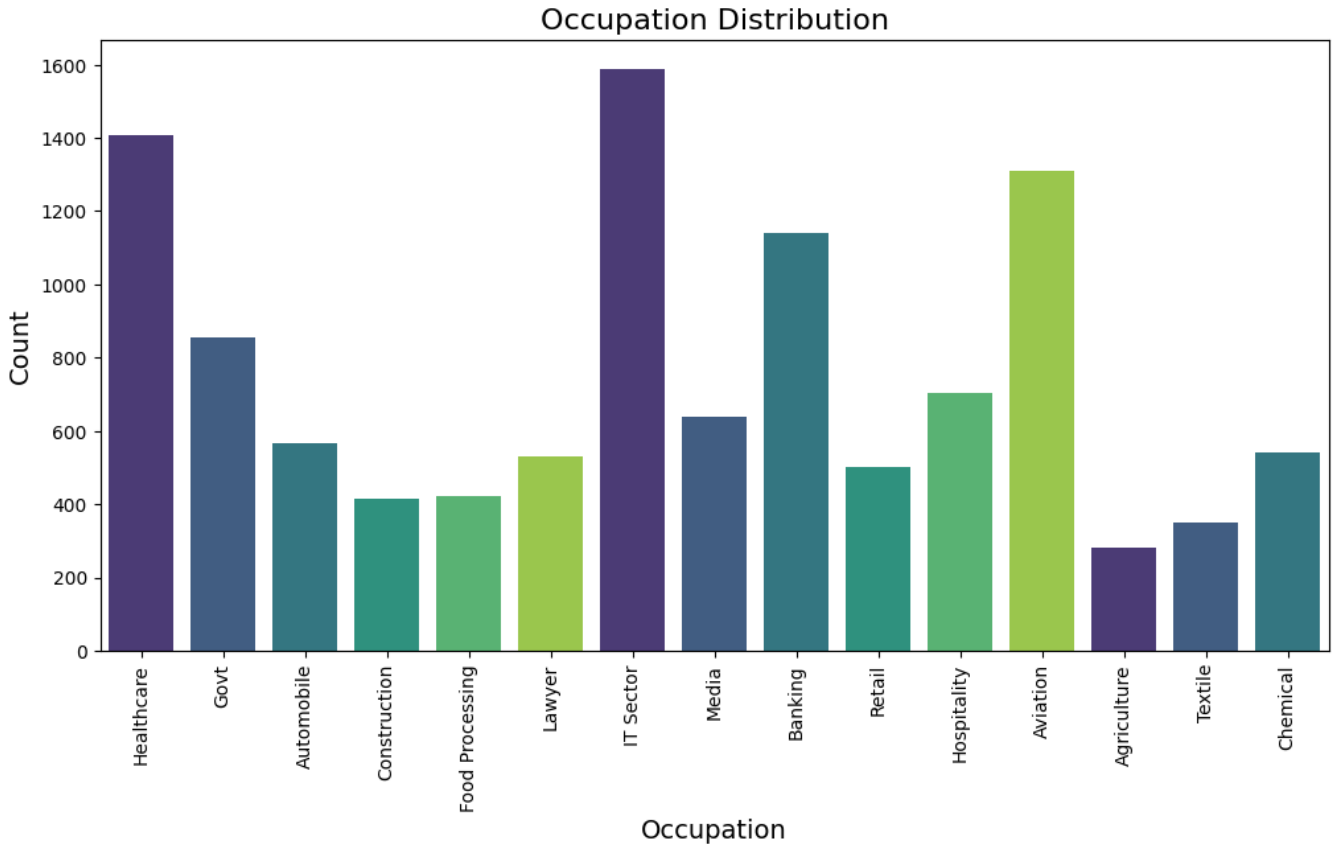
In [17]:
```
# State (Categorical data) - Countplot
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="State", palette=custom_palette)
plt.title("State Distribution", fontsize=16)
plt.xlabel("State", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.xticks(rotation=90)
plt.show()
```



In [18]:
```
# Zone (Categorical data) - Countplot
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Zone", palette=custom_palette)
plt.title("Zone Distribution", fontsize=16)
plt.xlabel("Zone", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```

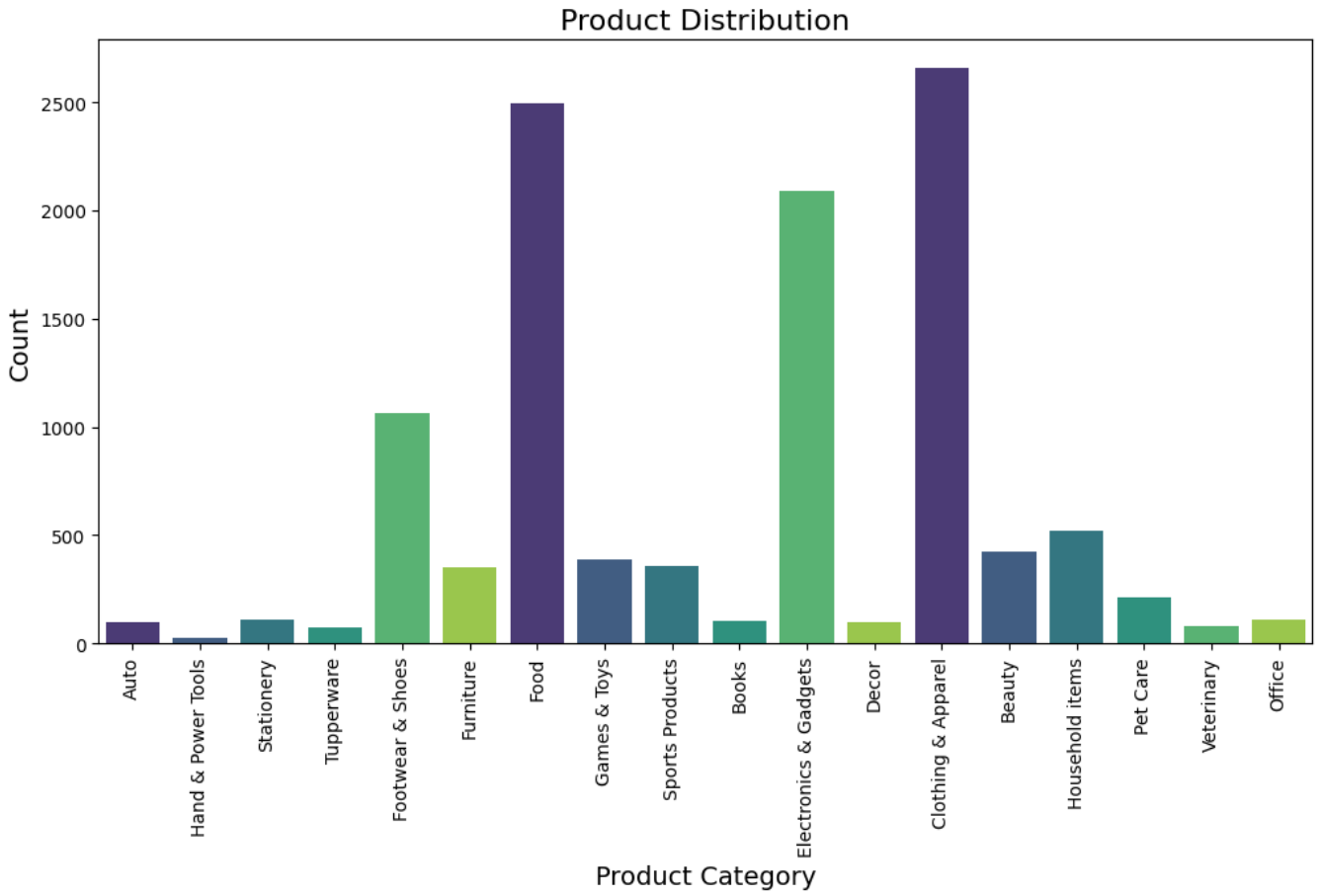


In [19]:
```
# Occupation (Categorical data) - Countplot
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Occupation", palette=custom_palette)
plt.title("Occupation Distribution", fontsize=16)
plt.xlabel("Occupation", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.xticks(rotation=90)
plt.show()
```

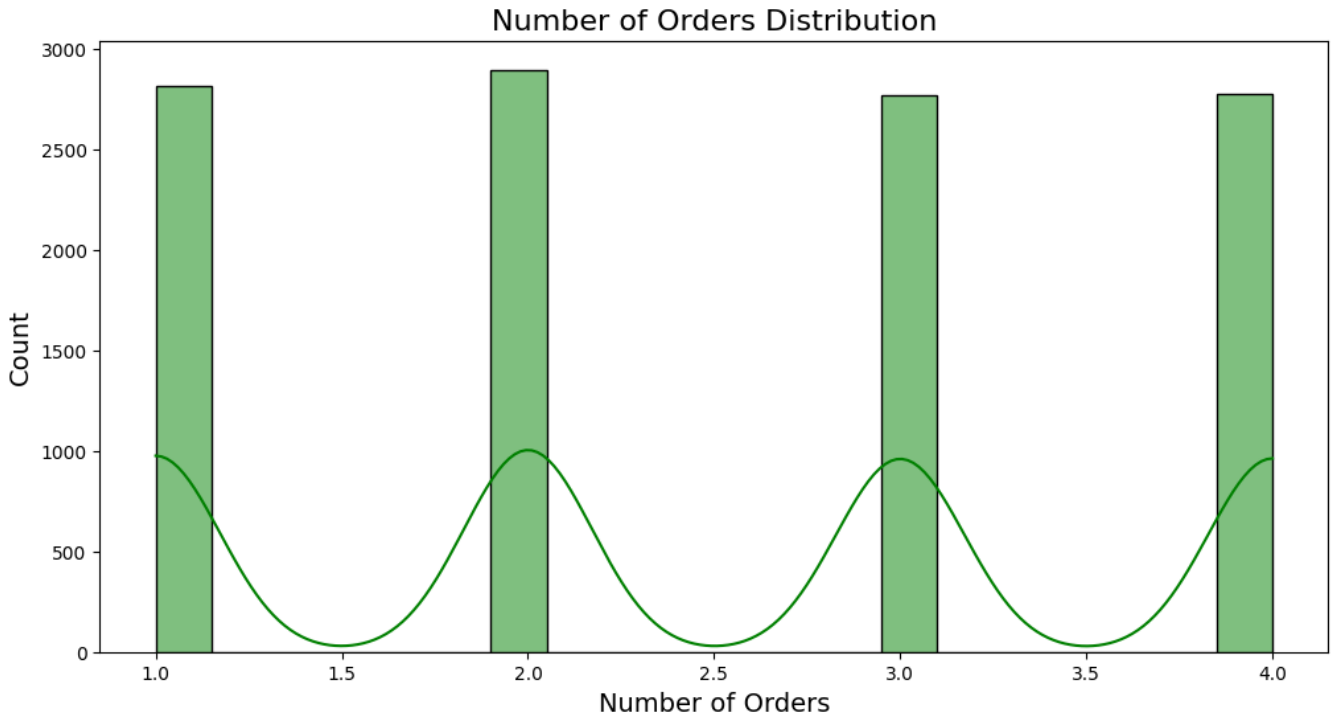


In [20]:

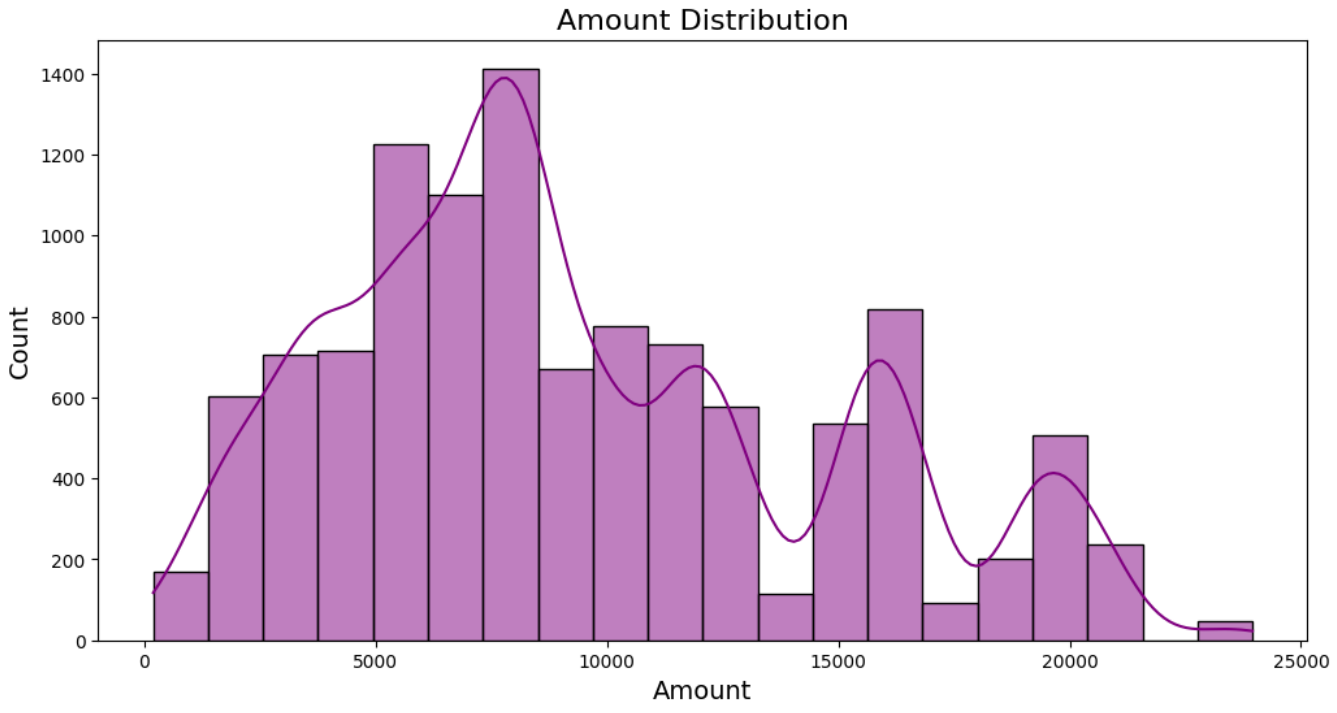
```
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Product_Category", palette=custom_palette)
plt.title("Product Distribution", fontsize=16)
plt.xlabel("Product Category", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.xticks(rotation=90)
plt.show()
```



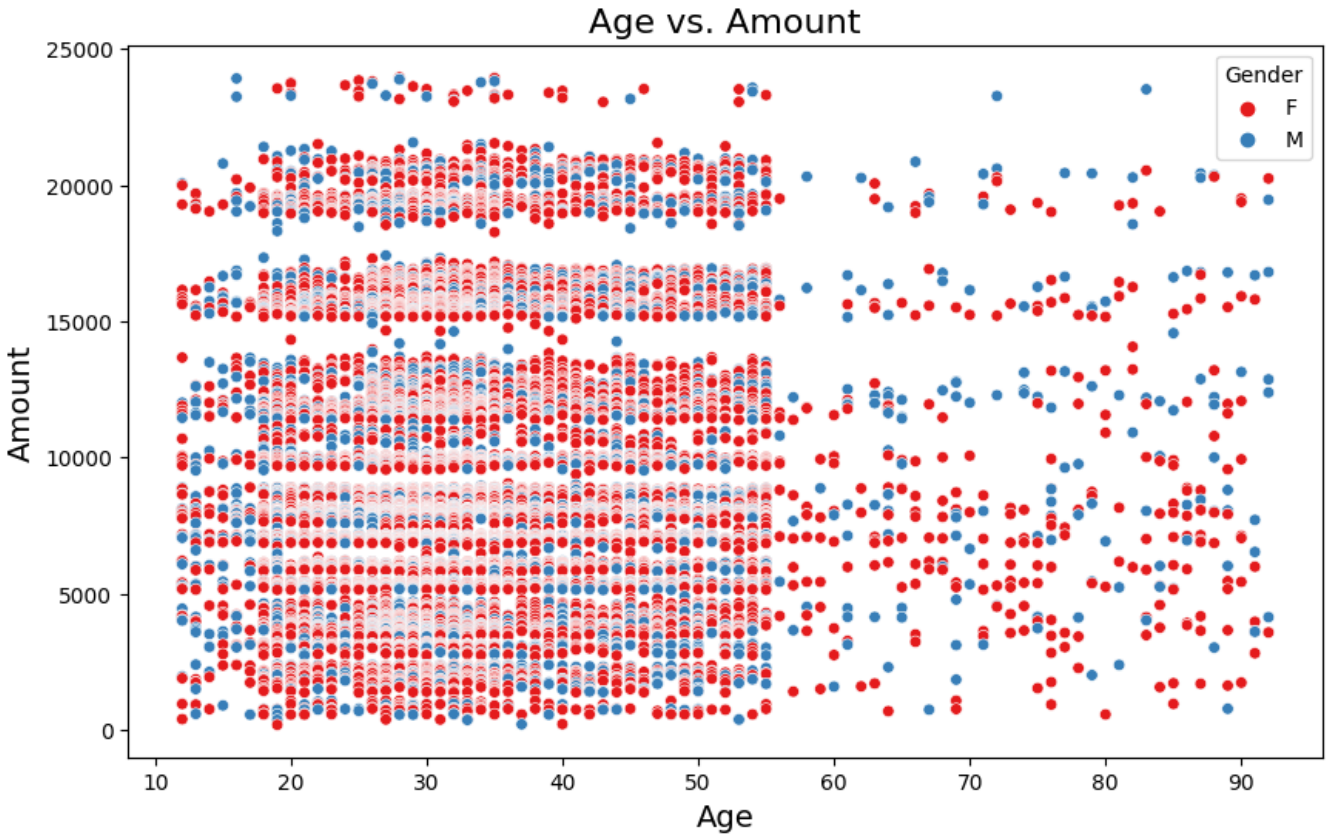
In [21]:
```
# Orders (Numerical data) - Histogram
plt.figure(figsize=(12, 6))
sns.histplot(data=df, x="Orders", bins=20, kde=True, color='green')
plt.title("Number of Orders Distribution", fontsize=16)
plt.xlabel("Number of Orders", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```



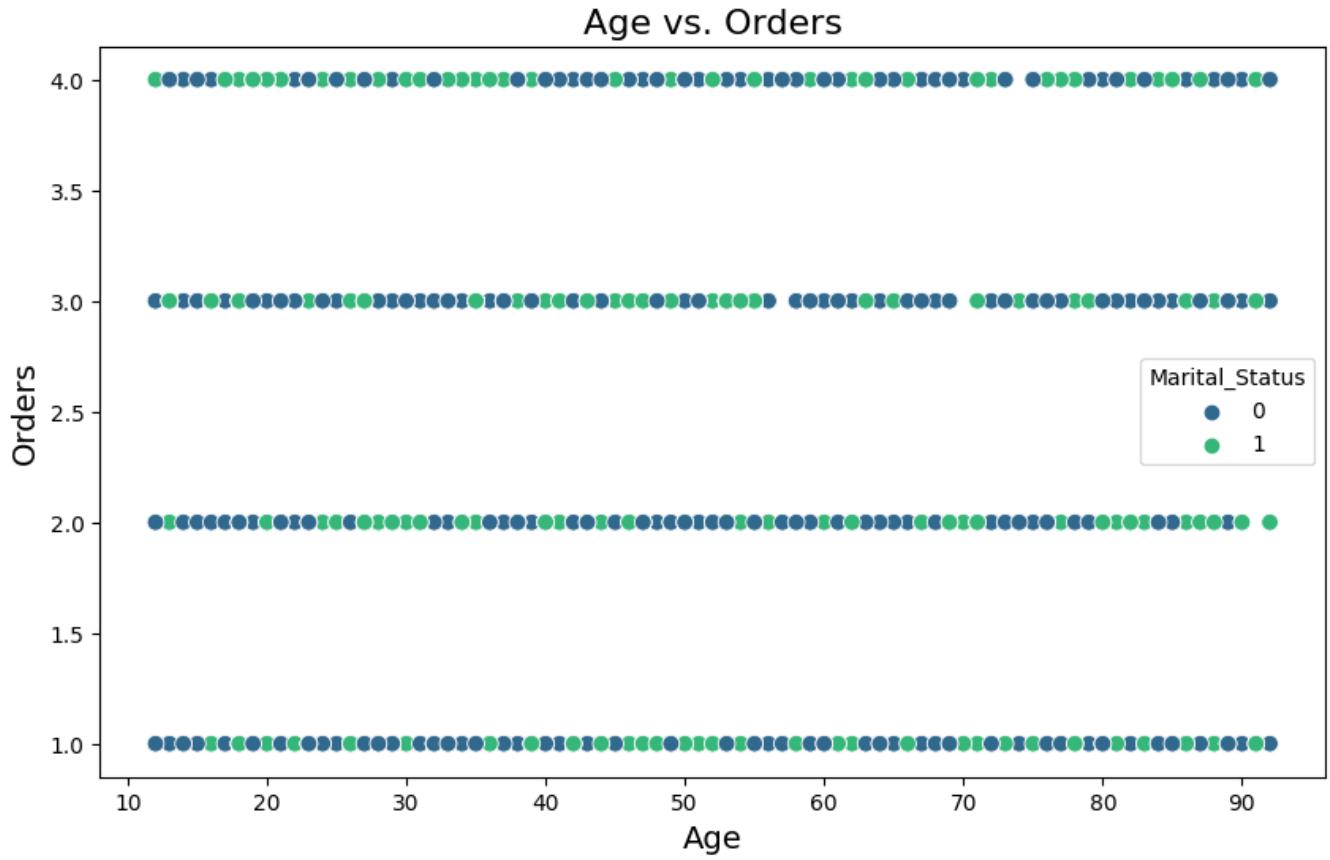
In [22]:
```
# Amount (Numerical data) - Histogram
plt.figure(figsize=(12, 6))
sns.histplot(data=df.dropna(), x="Amount", bins=20, kde=True, color='purple')
plt.title("Amount Distribution", fontsize=16)
plt.xlabel("Amount", fontsize=14)
plt.ylabel("Count", fontsize=14)
plt.show()
```



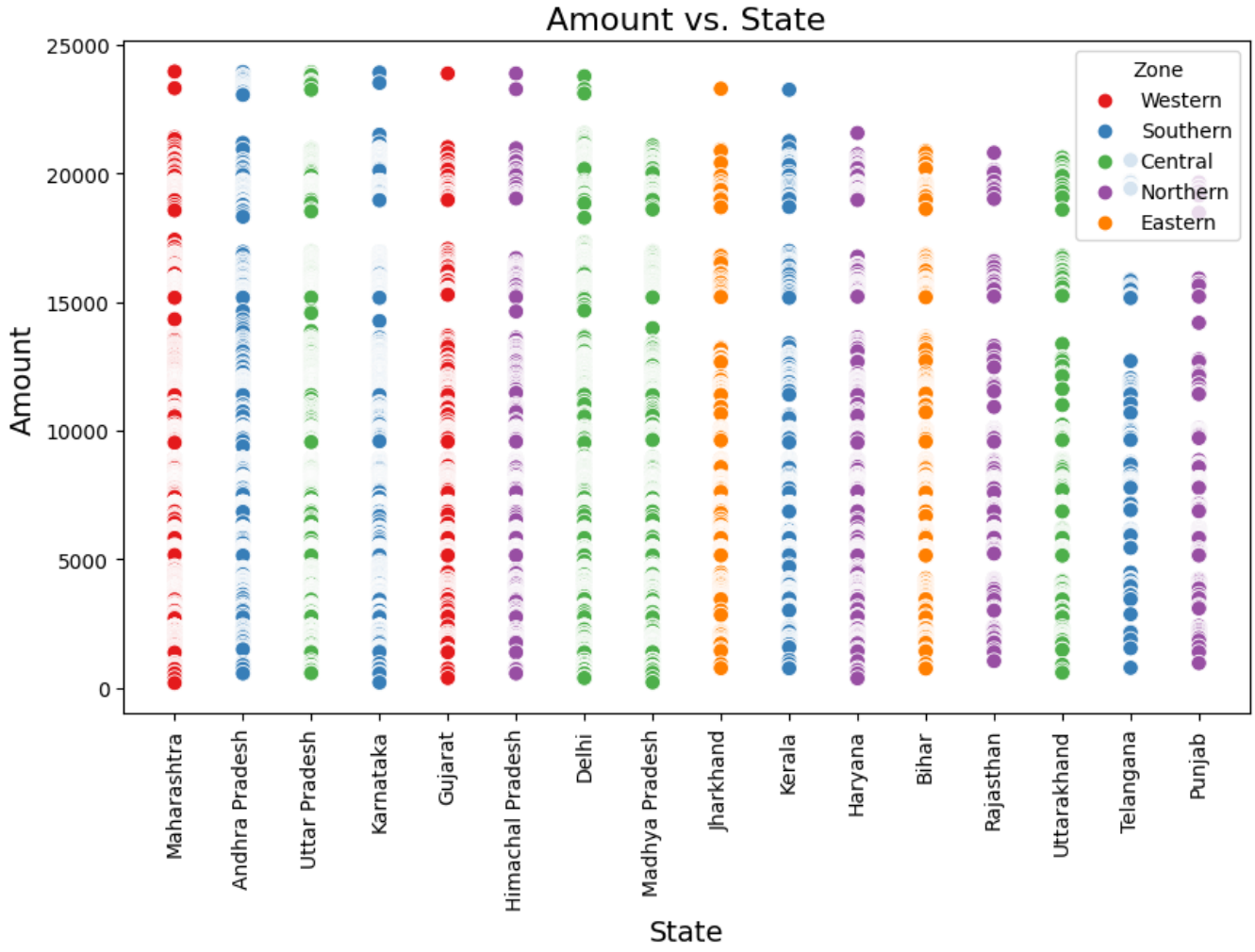
In [23]:
```
markers = {"Male": "o", "Female": "s"}

plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x="Age", y="Amount", hue="Gender", palette="Set1", s=30, markers=markers)
plt.title("Age vs. Amount", fontsize=16)
plt.xlabel("Age", fontsize=14)
plt.ylabel("Amount", fontsize=14)
plt.legend(title="Gender", loc="best")
plt.show()
```



In [24]:
```
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x="Age", y="Orders", hue="Marital_Status", palette="viridis", s=60)
plt.title("Age vs. Orders", fontsize=16)
plt.xlabel("Age", fontsize=14)
plt.ylabel("Orders", fontsize=14)
plt.show()
```



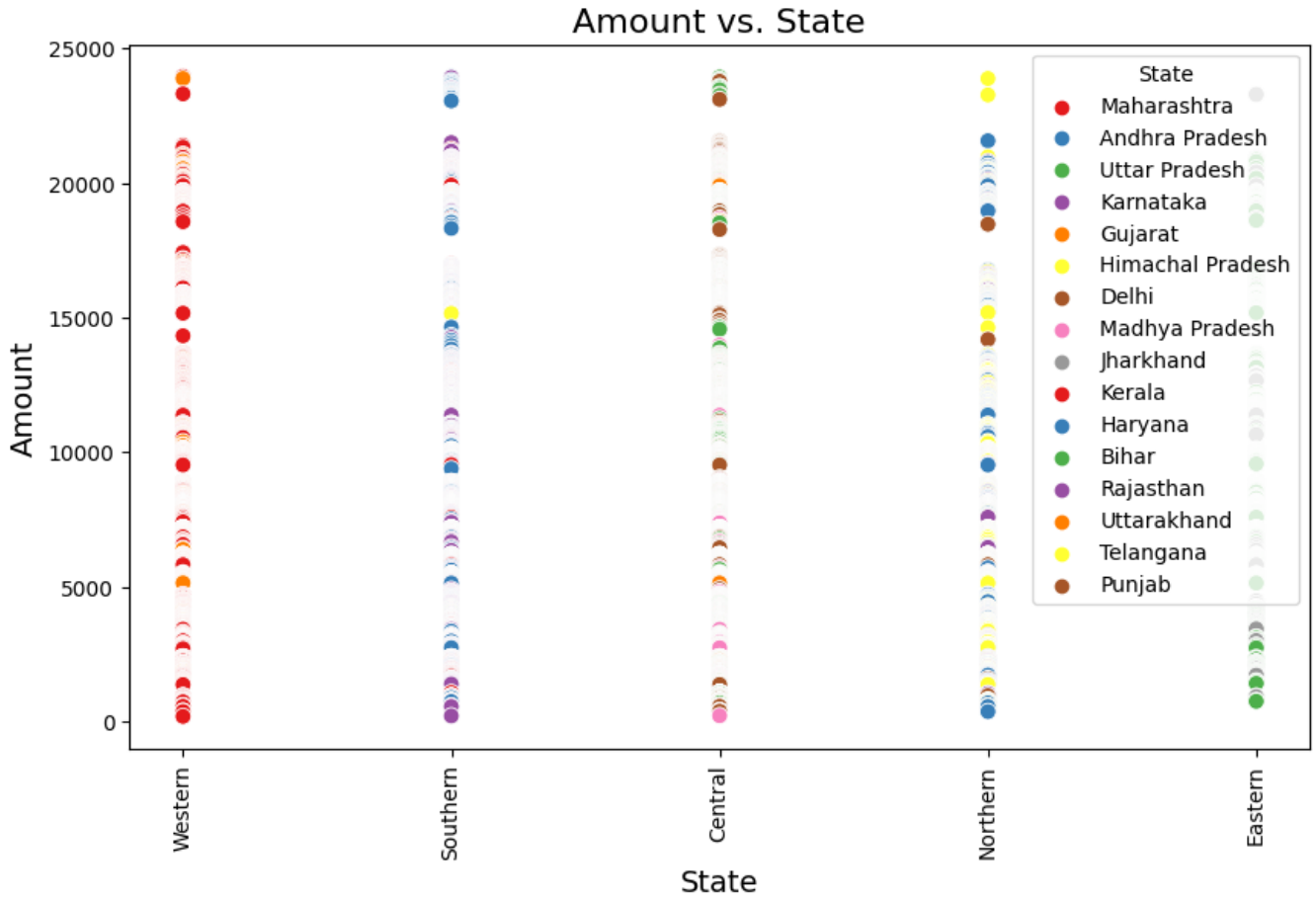
In [25]:
```
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x="State", y="Amount", hue="Zone", palette="Set1", s=60)
plt.title("Amount vs. State", fontsize=16)
plt.xlabel("State", fontsize=14)
plt.ylabel("Amount", fontsize=14)
plt.xticks(rotation=90)
plt.show()
```



In [26]:
```
df.columns
```

Out[26]:
```
Index(['Cust_name', 'Gender', 'Age Group', 'Age', 'Marital_Status', 'State',
       'Zone', 'Occupation', 'Product_Category', 'Orders', 'Amount'],
      dtype='object')
```

In [27]:
```
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x="Zone", y="Amount", hue="State", palette="Set1", s=60)
plt.title("Amount vs. State", fontsize=16)
plt.xlabel("State", fontsize=14)
plt.ylabel("Amount", fontsize=14)
plt.xticks(rotation=90)
plt.show()
```



In [28]:
```
corr_matrix = df[cols_num].corr(method = 'spearman')
mask = np.triu(np.ones_like(corr_matrix, dtype = bool))

fig,ax = plt.subplots(figsize = (10,5))
sns.heatmap(corr_matrix, cmap = 'coolwarm', annot = True, annot_kws = {'fontsiz
e':8, 'fontweight':'bold'},
            linewidths = 1.0, square = True, mask = mask, ax = ax)
ax.set_title('Correlation Matrix', fontsize = 12, fontweight = 'bold', color =
'darkblue')
fig.show()
```