In [11]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
#importing all liberaries.
```

In [12]:
```
df=pd.read_csv("Diwali.csv", encoding="unicode_escape")
df
#reading CSV(Comma Seprarated Values).
```

Out[12]:

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

11251 rows × 15 columns

```
- Our Variable Features:

-User_ID: Unique number of each user.
-Cust_name: cust name of users.
-Product_ID: Unique number of each product.
-Gender: F= Female, M= Male.
-Age Group: Age lies between this numbers.
-Age: Age of users.
-Marital_Status: 0= unmarried, 1= married.
-State: Name of states.
-Zone: Name of zones.
-Occupation: Work place of user.
-Product_Category: Type of product.
-Orders: total number of orders.
-Amount: Total Amount/Bill of product.
```

```
-It is an diwali data of customers. cutomers from diffrent states and zone. category of product,product type.
-Every costumer has diffrent userID and productID. After visualizing data it shows that female puchasing power
is more that male, and people with IT occupation spent more money then other occupation.
```

Total number of rows and columns

In [13]:
```
print("Number Of Row",df.shape[0])
print("Number Of Col",df.shape[1])
#total number of rows and columns
```

```
Number Of Row 11251
Number Of Col 15
```

EDA

In [14]:
```
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 [15]:
```
df.isna().sum()
#check for null values.
```

Out[15]:
```
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 [16]:
```
df.drop(["Status","unnamed1"],axis=1,inplace=True)
#drop unrelated/Blank columns
```

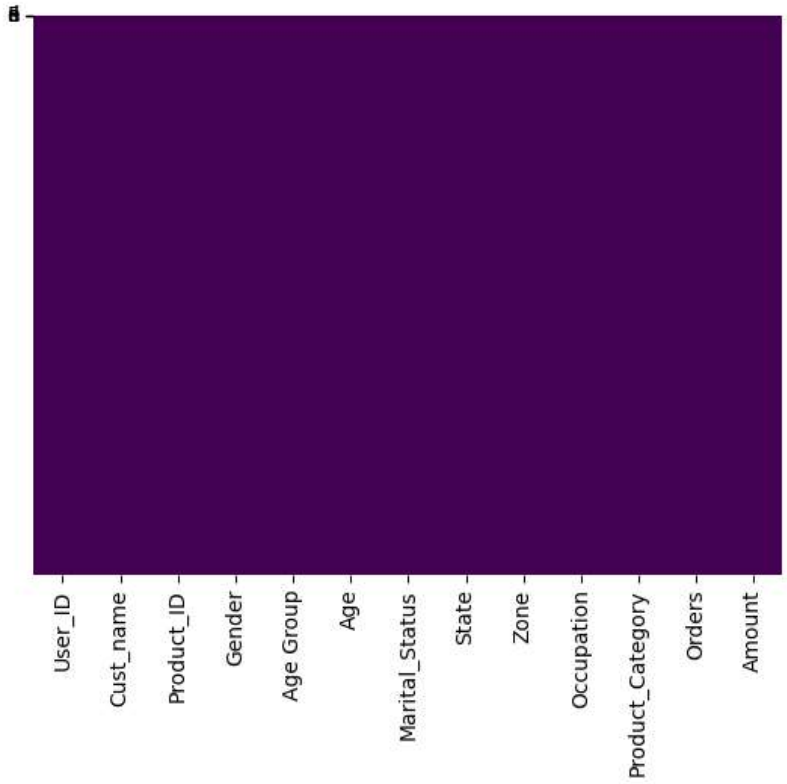
In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 13 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
dtypes: float64(1), int64(4), object(8)
memory usage: 1.1+ MB
```

In [18]:
```
df.isnull().sum()
#showing null values.
```

Out[18]:
```
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 [19]:
```
sns.heatmap(df.isnull(),yticklabels="False",cbar=False,cmap="viridis")
plt.show()
#graphical representation of null values.
```



In [20]:
```
df.dropna(inplace=True)
#dropping null values.
```

In [21]:
```
df.info()
#checking datatypes of columns.
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11239 entries, 0 to 11250
Data columns (total 13 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  Orders            11239 non-null  int64  
 12  Amount            11239 non-null  float64
dtypes: float64(1), int64(4), object(8)
memory usage: 1.2+ MB
```

In [22]:
```
df["Amount"]=df["Amount"].astype("int")
#changing datatype of column.
```

In [23]:
```
df.info()
#checking if datatype is change or not.
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11239 entries, 0 to 11250
Data columns (total 13 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  Orders            11239 non-null  int64  
 12  Amount            11239 non-null  int32  
dtypes: int32(1), int64(4), object(8)
memory usage: 1.2+ MB
```

In [24]:
```
df.columns
#name of all colunmns.
```

Out[24]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [25]:
```
df.rename(columns={"Marital_Status":"Single/Married"})
#changing name of column/temporary.
```

Out[25]:

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

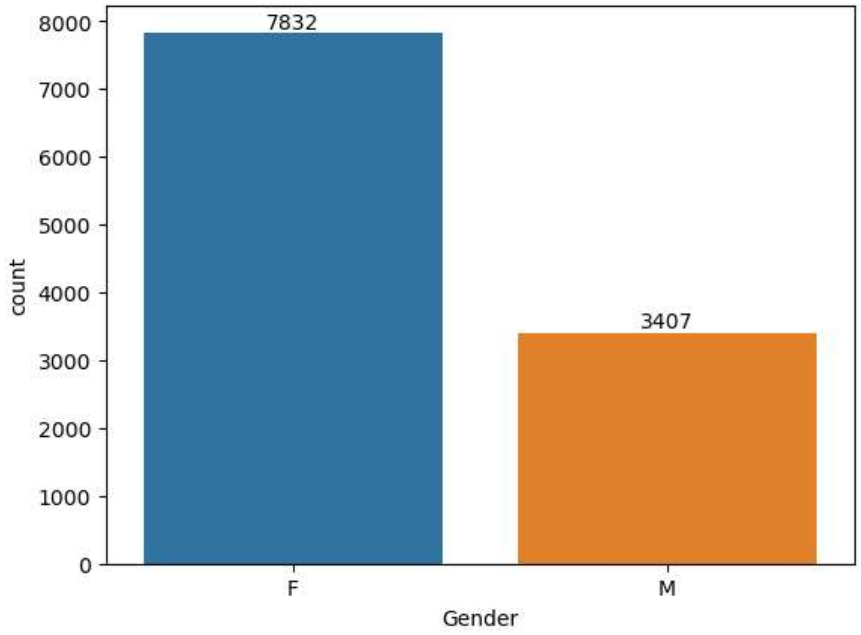
11239 rows × 13 columns

In [26]:
```
df.describe()
#describe() method gives description of the dataframe.
```

Out[26]:

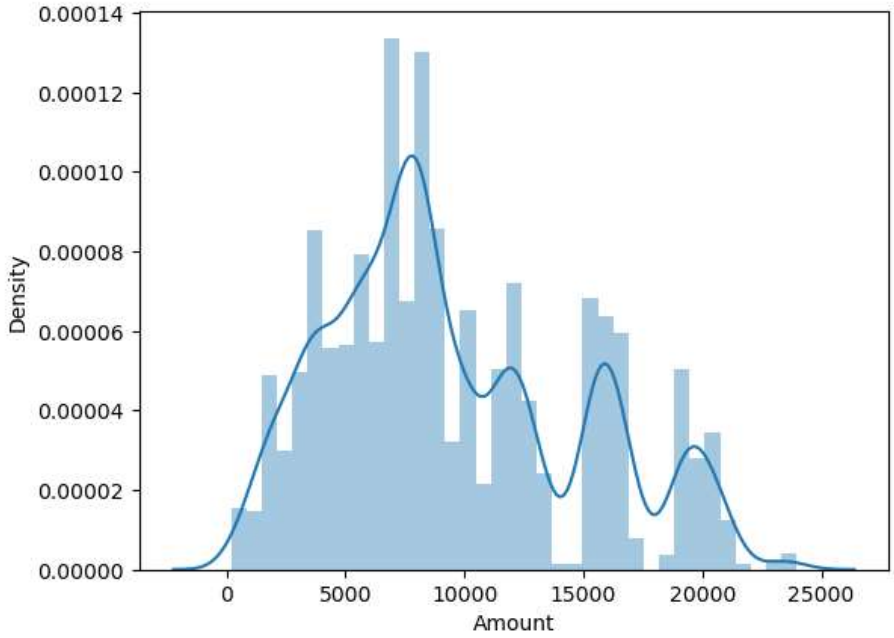
	User_ID	Age	Marital_Status	Orders	Amount
count	1.123900e+04	11239.000000	11239.000000	11239.000000	11239.000000
mean	1.003004e+06	35.410357	0.420055	2.489634	9453.610553
std	1.716039e+03	12.753866	0.493589	1.114967	5222.355168
min	1.000001e+06	12.000000	0.000000	1.000000	188.000000
25%	1.001492e+06	27.000000	0.000000	2.000000	5443.000000
50%	1.003064e+06	33.000000	0.000000	2.000000	8109.000000
75%	1.004426e+06	43.000000	1.000000	3.000000	12675.000000
max	1.006040e+06	92.000000	1.000000	4.000000	23952.000000

In [27]:
```
m1=sns.countplot(x="Gender",data=df)
for bars in m1.containers:
    m1.bar_label(bars)
#females purchasing power is more as compare to males.
```



In [85]:
```
sns.distplot(df["Amount"])
#it is right skewness.
#mean is greater then median.
```

Out[85]: <Axes: xlabel='Amount', ylabel='Density'>



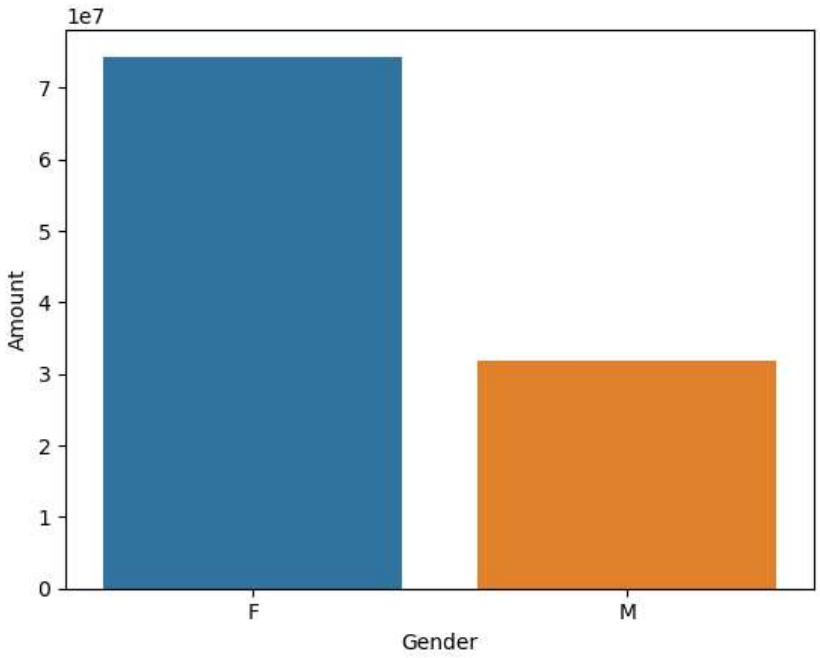
In [29]:
```
df.groupby(["Gender"], as_index=False)["Amount"].sum().sort_values(by="Amount",ascending=False)
#total amount spent by male and female.
```

Out[29]:

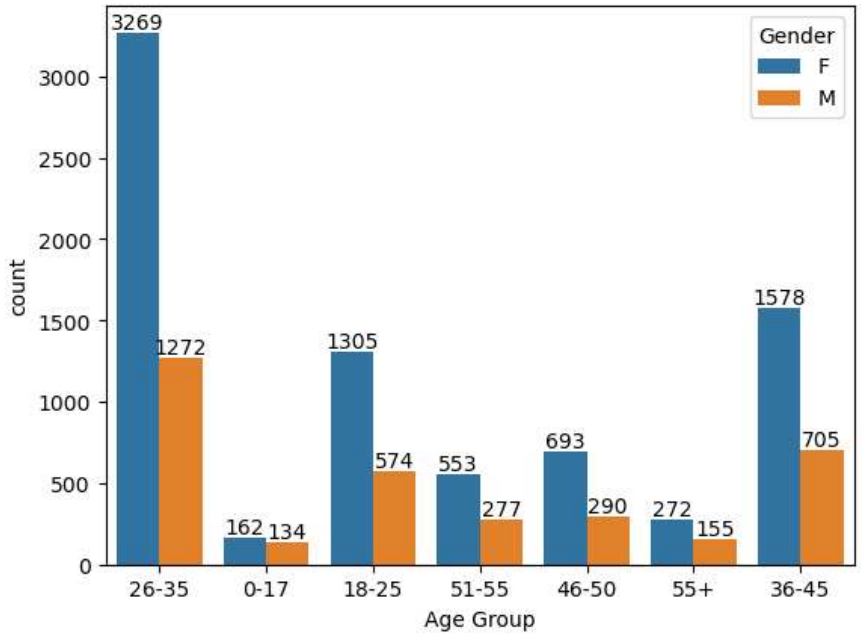
	Gender	Amount
0	F	74335853
1	M	31913276

In [30]:
```
sales=df.groupby(["Gender"], as_index=False)["Amount"].sum().sort_values(by="Amount",ascending=False)
sns.barplot(x="Gender",y="Amount",data=sales)
#from the below graph we say that most of the buyers are female.
```

Out[30]: <Axes: xlabel='Gender', ylabel='Amount'>

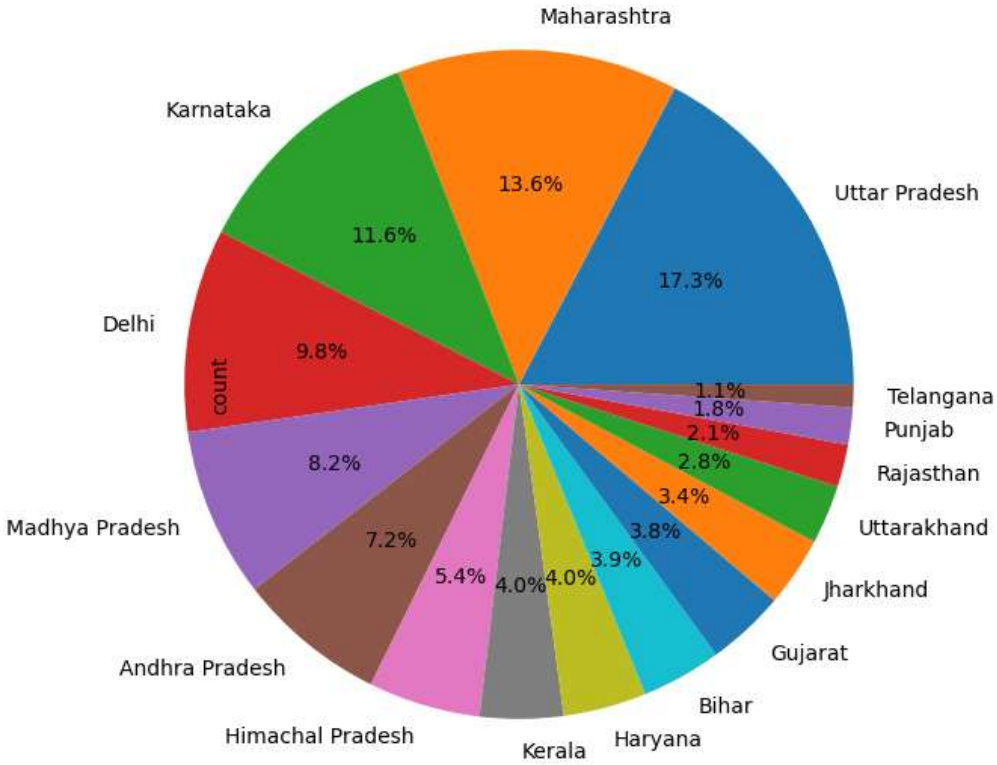


In [31]:
```
m2=sns.countplot(data=df,x="Age Group",hue="Gender")
for bars in m2.containers:
    m2.bar_label(bars)
#from the below graph we can say that most of the costumer are from age group of 26-35 years female.
```



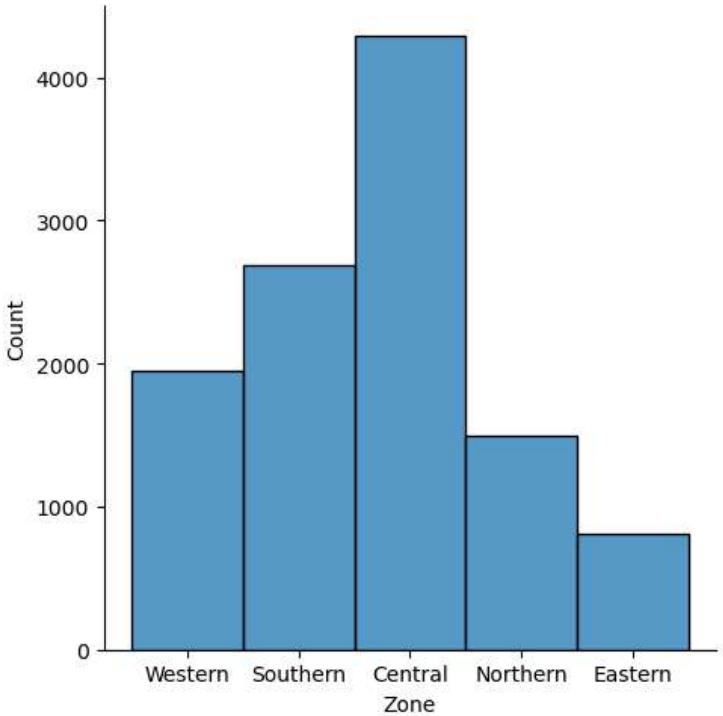
In [77]:
```
df["State"].value_counts().plot.pie(radius=1.5,autopct="%1.1f%%")
#after observing we say that most of the customers are from Utter Pradesh and least customers are from Telangana
```

Out[77]: <Axes: ylabel='count'>



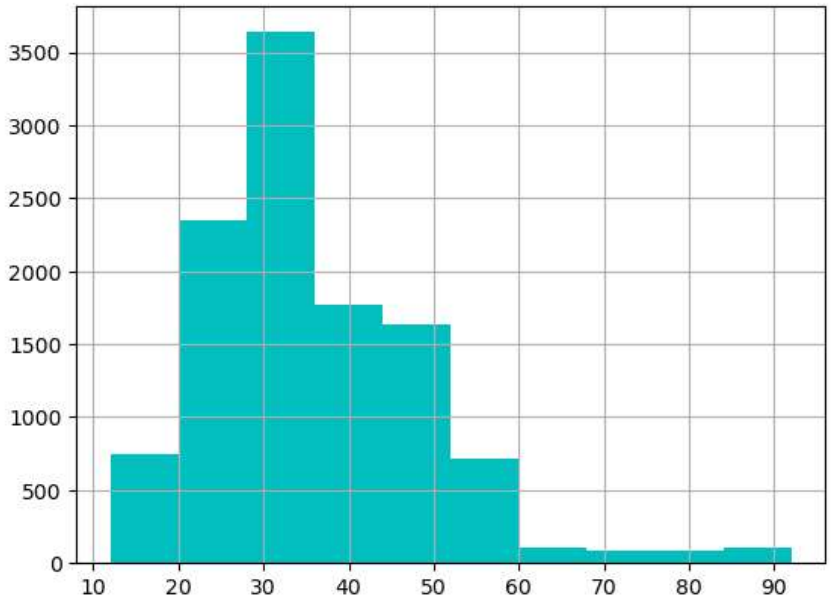
In [66]:
```
sns.displot(df["Zone"])
#central zone has many customers compare to other states.
```

Out[66]: <seaborn.axisgrid.FacetGrid at 0x248f53ab750>

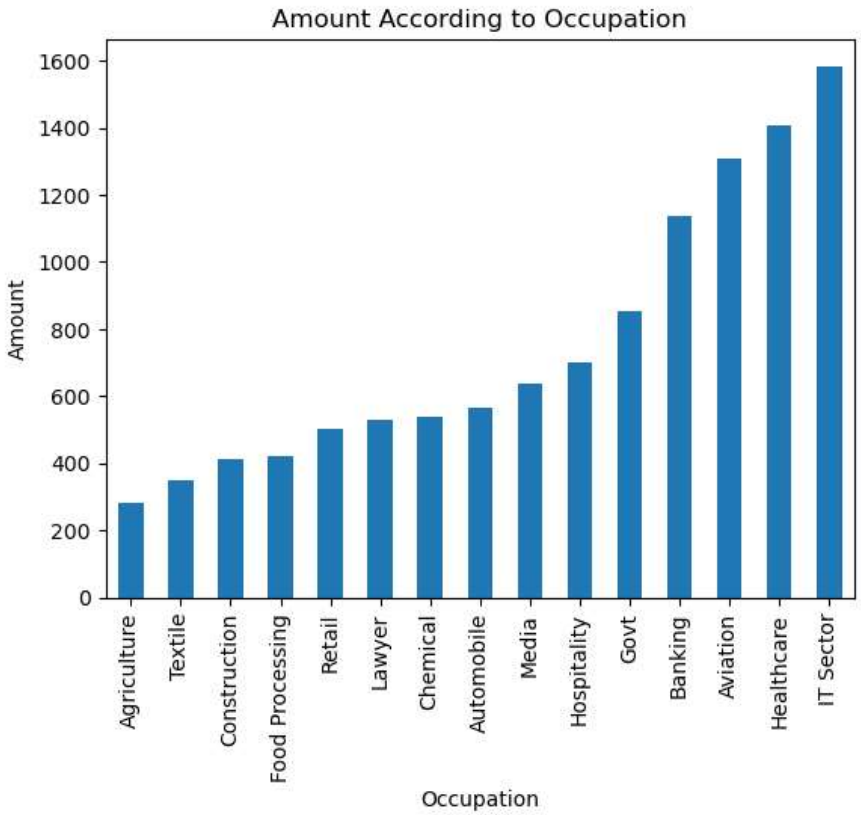


In [86]:
```
df["Age"].hist(color="c")
```

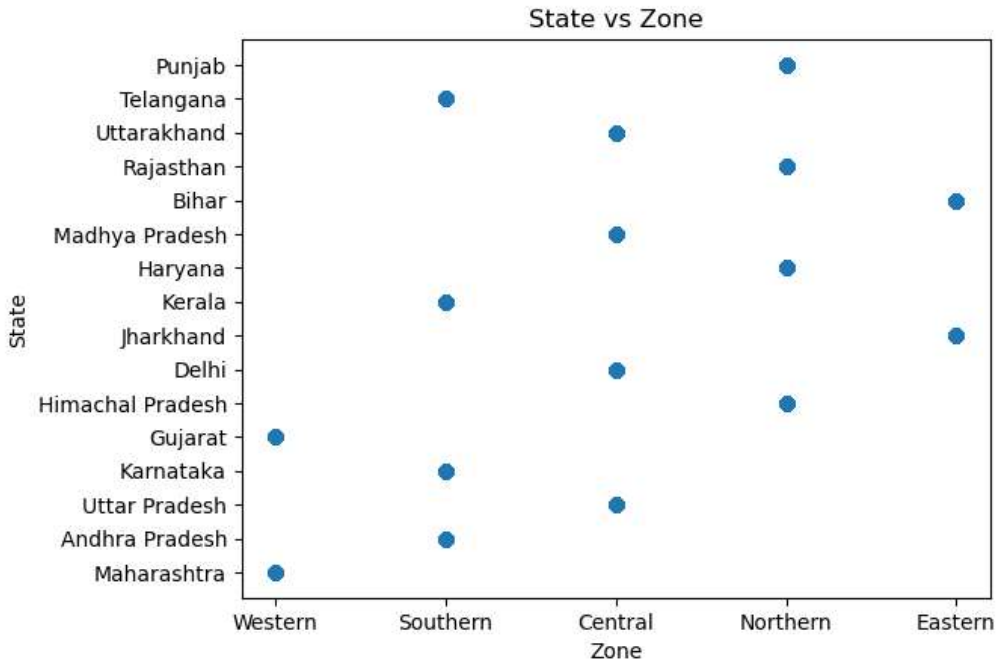
Out[86]: <Axes: >



In [78]:
```
df["Occupation"].value_counts().sort_values(ascending=True).plot(kind="bar")
plt.title("Amount According to Occupation")
plt.xlabel("Occupation")
plt.ylabel("Amount")
plt.show()
#show that customers from IT occupation spend more money than other occupation.
```

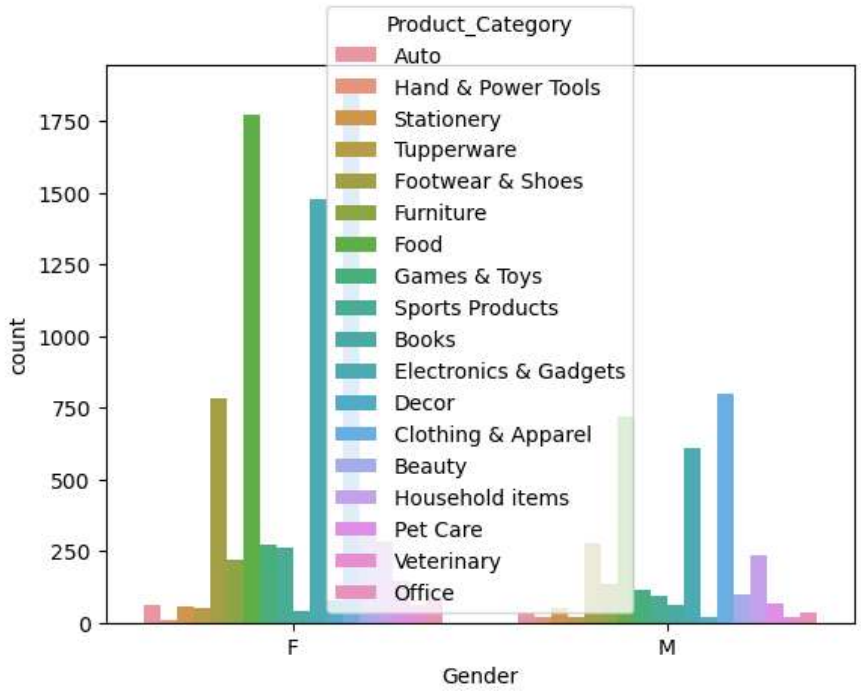


In [83]:
```
plt.scatter(x=df["Zone"],y=df["State"])
plt.title("State vs Zone")
plt.xlabel("Zone")
plt.ylabel("State")
plt.show()
#shows states comes under which zone.
```



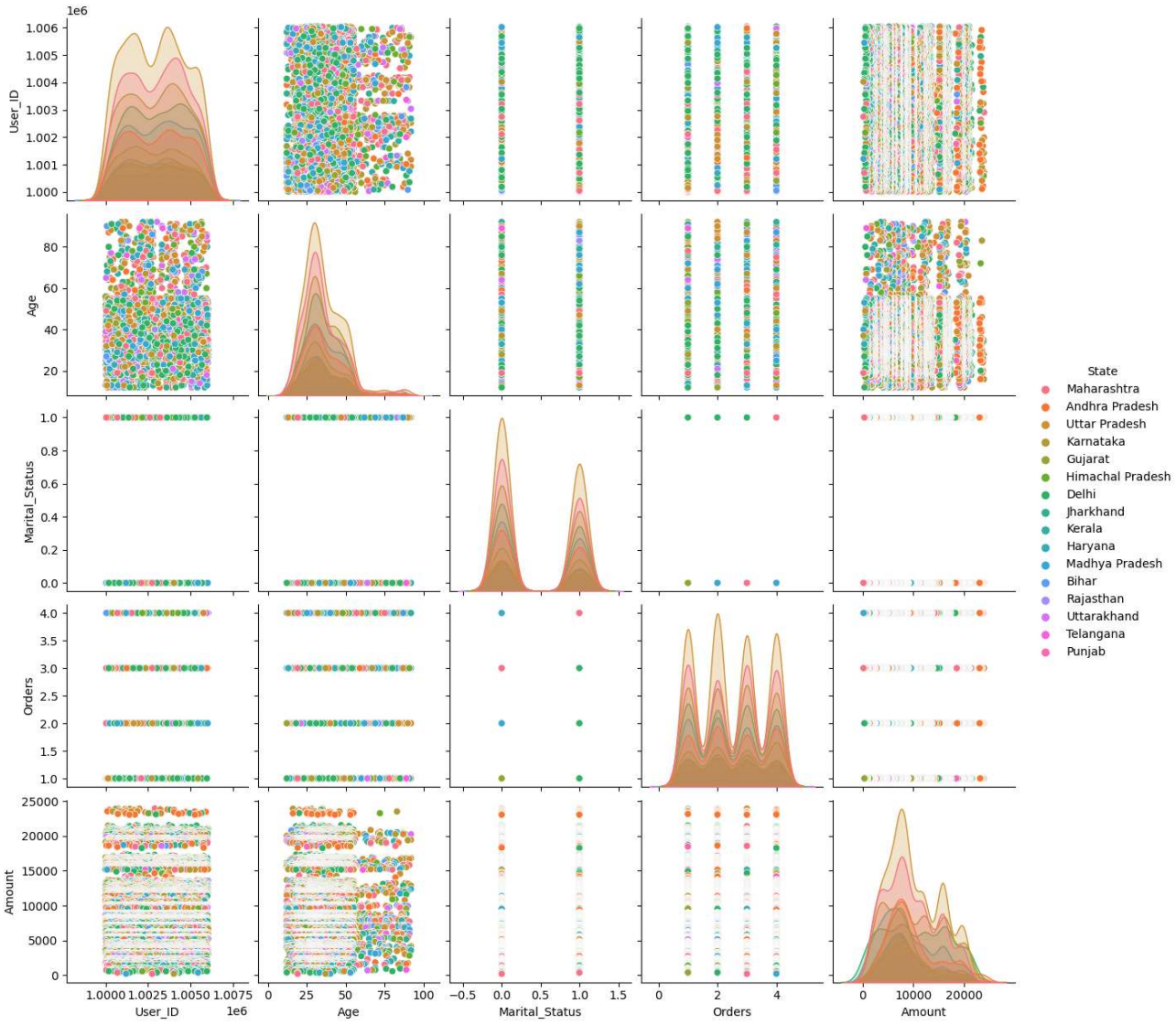
In [58]:
```
sns.countplot(data=df,x="Gender",hue="Product_Category")
#count of category.
#Every colour is representing product category.
#food is purchase more by female and cloting&Apparel is purchase more by male.
```

Out[58]: <Axes: xlabel='Gender', ylabel='count'>



In [152]:
```
sns.pairplot(df,hue="State")
#ploting pair of state, every state has its colour in right side.
#their are right skewness and 0 skewness.
#if it is right skewness mean is greater then median.
```

Out[152]: <seaborn.axisgrid.PairGrid at 0x29d6438df50>



In [ ]:

In [ ]: