Name = Manoj Kumar

Batch = 1st September Batch

course = Data science Placement guarantee Course

Email = manojkumarrajput9990@gmail.com

In [195…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
warnings.simplefilter(action='ignore', category=UserWarning)
```

Imported all the libraries which we need to use for this task

Part 1: Basic Data Exploration and Manipulation

In [2]:

```
data = pd.read_csv("C:\\Users\\msgme\\Downloads\\Assignmnet_dataset.csv")
```

In [5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4000 entries, 0 to 3999
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   CustomerID       4000 non-null   int64  
 1   Name             3340 non-null   object 
 2   Age              3944 non-null   float64
 3   Gender           2939 non-null   object 
 4   City             3328 non-null   object 
 5   PurchaseAmount   3900 non-null   float64
 6   PurchaseDate     4000 non-null   object 
 7   ProductCategory  3366 non-null   object 
dtypes: float64(2), int64(1), object(5)
memory usage: 250.1+ KB
```

In [7]:

```
data.head()
```

Out[7]:

	CustomerID	Name	Age	Gender	City	PurchaseAmount	PurchaseDate	ProductCat
0	8270	Jane Smith	46.0	Female	Los Angeles	648.27	2021-09-12 00:00:00	
1	1860	NaN	30.0	NaN	Houston	185.30	2020-06-30 00:00:00	Gr
2	6390	Jane Smith	38.0	Male	New York	564.92	2021-11-16 00:00:00	Home
3	6191	John Doe	75.0	Female	Houston	981.52	2021-11-11 00:00:00	
4	6734	NaN	38.0	NaN	Chicago	523.13	2020-09-04 00:00:00	Home

In [9]:

```
data.tail()
```

Out[9]:

	CustomerID	Name	Age	Gender	City	PurchaseAmount	PurchaseDate	Produ
3995	9347	Chris Johnson	64.0	NaN	Houston	NaN	2023-04-02 00:00:00	
3996	6592	John Doe	23.0	Female	New York	527.92	2023-09-20 00:00:00	
3997	2724	John Doe	61.0	Female	NaN	223.12	2020-11-10 00:00:00	
3998	4343	Emily White	25.0	NaN	Phoenix	721.52	2022-08-02 00:00:00	
3999	1886	John Doe	24.0	Female	Houston	93.93	2020-08-25 00:00:00	

In [11]:

```
data.describe()
```

Out[11]:

	CustomerID	Age	PurchaseAmount
count	4000.000000	3944.000000	3900.000000
mean	5482.300250	47.774341	505.700428
std	2573.310642	18.834296	287.535960
min	1001.000000	-1.000000	10.250000
25%	3255.000000	32.000000	255.895000
50%	5492.500000	48.000000	506.750000
75%	7704.000000	64.000000	759.177500
max	9996.000000	79.000000	999.930000

1. Data Cleaning:

In [13]:

```
missing_values= data.isnull().sum()
print(missing_values)
```

```
CustomerID            0
Name                660
Age                  56
Gender             1061
City                672
PurchaseAmount      100
PurchaseDate          0
ProductCategory     634
dtype: int64
```

In [15]:

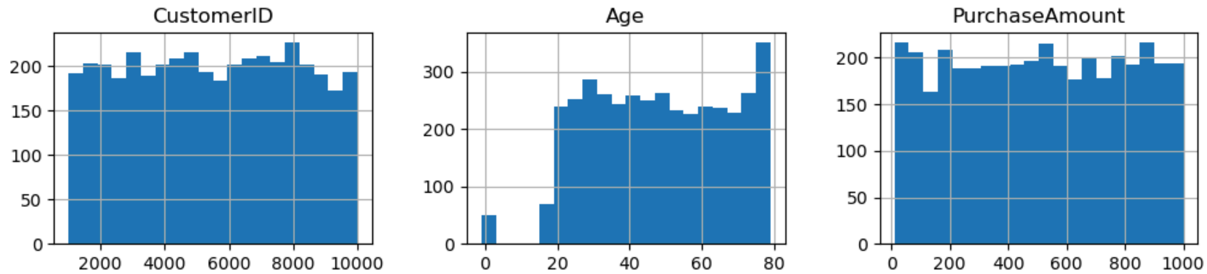
```
print(missing_values[missing_values > 0])
```

```
Name                660
Age                  56
Gender             1061
City                672
PurchaseAmount      100
ProductCategory     634
dtype: int64
```

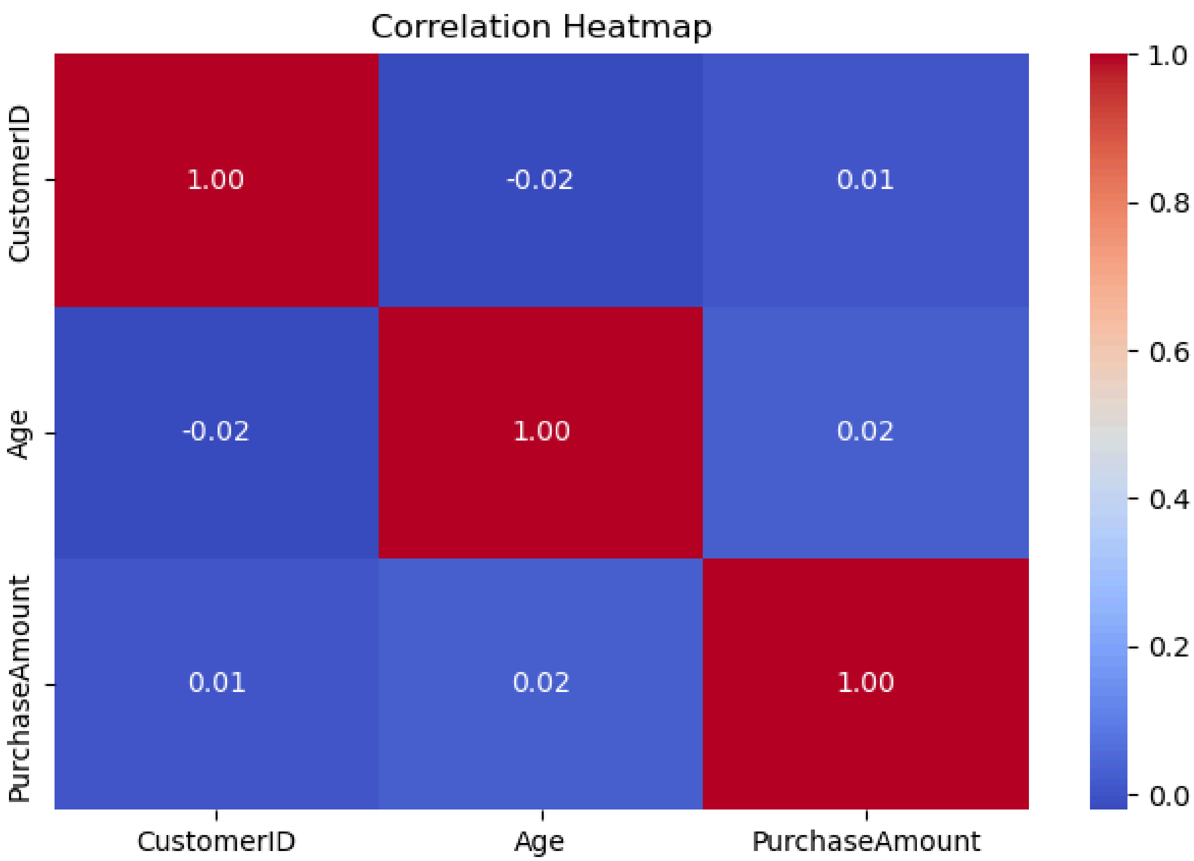
In [17]:

```
num_columns = ['CustomerID','Age','PurchaseAmount']
plt.figure(figsize=(10,8))
data[num_columns].hist(bins=20, figsize=(12, 8), layout=(3, 3))
plt.show()
```

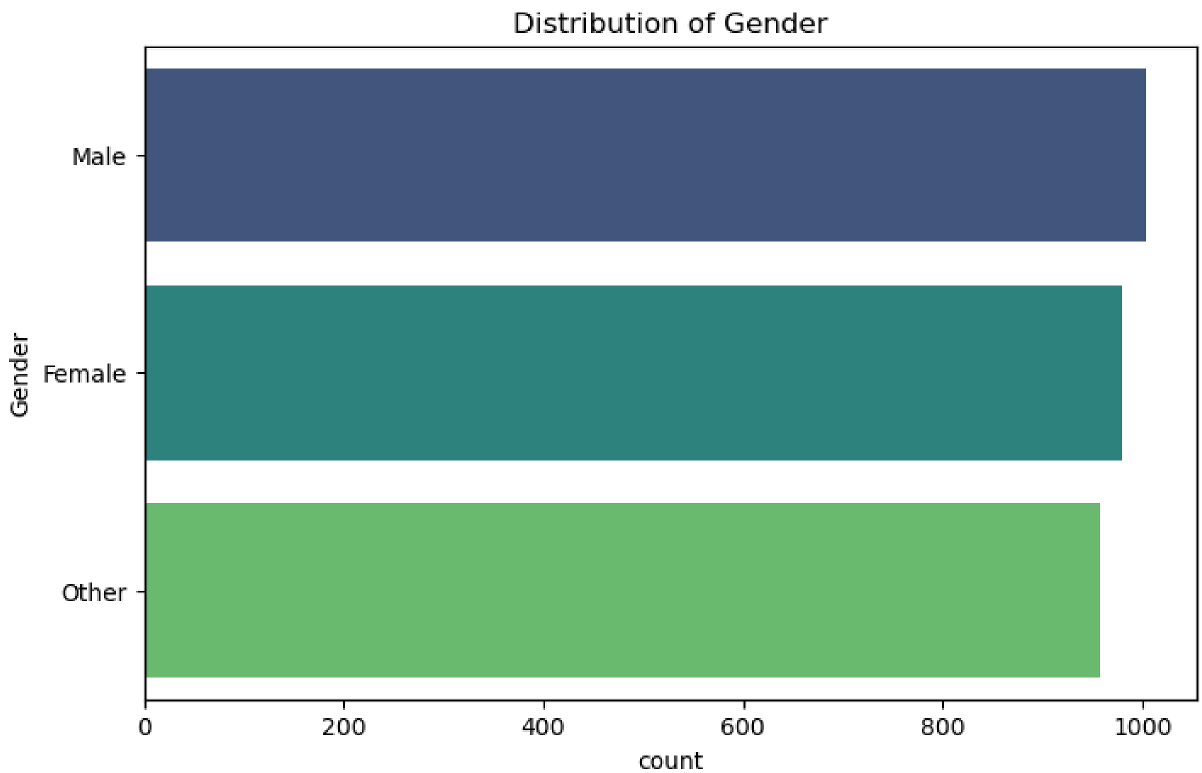
```
<Figure size 1000x800 with 0 Axes>
```



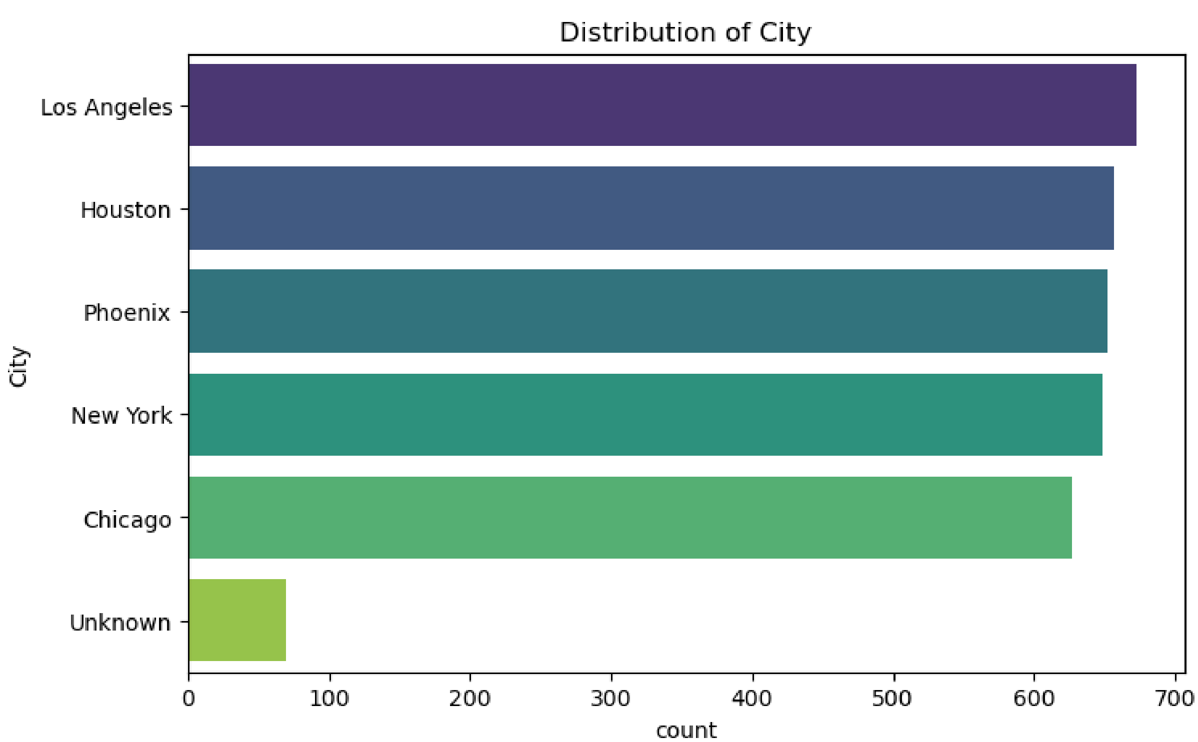
In [19]:
```
plt.figure(figsize=(8, 5))
sns.heatmap(data[num_columns].corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title("Correlation Heatmap")
plt.show()
```



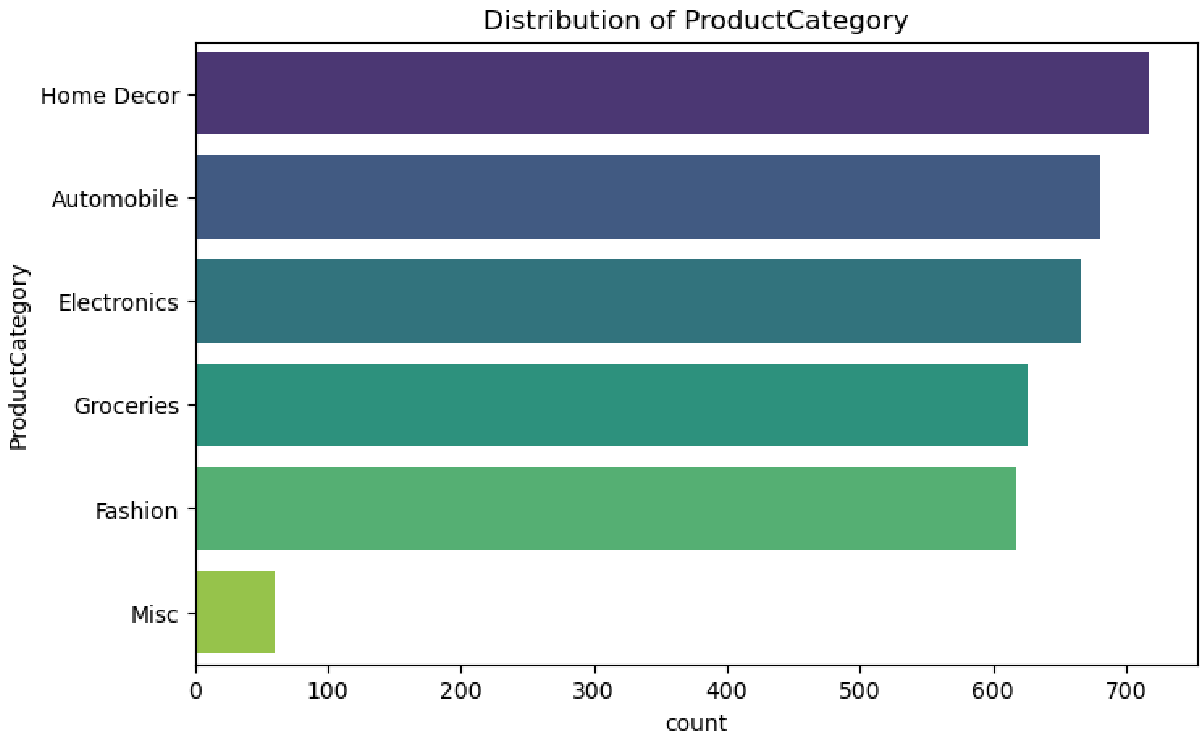
In [21]:
```
property_ty = ['Gender']
for colmn in property_ty:
    plt.figure(figsize=(8, 5))
    sns.countplot(y=data[colmn], order=data[colmn].value_counts().index, palette='v
    plt.title(f"Distribution of {colmn}")
    plt.show()
```



In [23]:
```
property_ty = ['City']
for colmn in property_ty:
    plt.figure(figsize=(8, 5))
    sns.countplot(y=data[colmn], order=data[colmn].value_counts().index, palette='v
    plt.title(f"Distribution of {colmn}")
    plt.show()
```



In [25]:
```
property_ty = ['ProductCategory']
for colmn in property_ty:
    plt.figure(figsize=(8, 5))
    sns.countplot(y=data[colmn], order=data[colmn].value_counts().index, palette='v
    plt.title(f"Distribution of {colmn}")
    plt.show()
```



Handling missing values

Filling missing Name, Gender, City, and ProductCategory with 'Unknown'

In [29]:
```
data['Name'].fillna('Unknown', inplace=True)
data['Gender'].fillna('Unknown', inplace=True)
data['City'].fillna('Unknown', inplace=True)
data['ProductCategory'].fillna('Unknown', inplace=True)
```

Filling missing Age with the median

In [33]:
```
data['Age'].fillna(data['Age'].median(), inplace=True)
```

Filling missing PurchaseAmount with the mean

In [37]:
```
data['PurchaseAmount'].fillna(data['PurchaseAmount'].mean(), inplace=True)
```

In [39]:
```
missing_values= data.isnull().sum()
print(missing_values)
```

```
CustomerID         0
Name               0
Age                0
Gender             0
City               0
PurchaseAmount     0
PurchaseDate       0
ProductCategory    0
dtype: int64
```

Converting PurchaseDate to datetime format

In [43]:
```
data['PurchaseDate'] = pd.to_datetime(data['PurchaseDate'], errors='coerce')
```

Keeping only the date and removing the time component

In [47]:
```
data['PurchaseDate'] = data['PurchaseDate'].dt.date
```

In [49]:
```
print(data['PurchaseDate'].head())
```

```
0    2021-09-12
1    2020-06-30
2    2021-11-16
3    2021-11-11
4    2020-09-04
Name: PurchaseDate, dtype: object
```

Removinhg duplicates from CustomerID and PurchaseDate columns

In [51]:
```
data.drop_duplicates(subset=['CustomerID'], inplace=True)
```

In [53]:
```
data.drop_duplicates(subset=['PurchaseDate'], inplace=True)
```

In [55]:
```
data['CustomerID'].shape[0]
```

Out[55]: 1295

In [57]:
```
data['PurchaseDate'].shape[0]
```

Out[57]: 1295

2: Customer Segmentation by City

In [61]:
```
new_york_customers = data[data['City'] == 'New York']
```

In [63]:
```
print("\nCustomers in New York:\n", new_york_customers[['CustomerID', 'Name', 'City
```

```
Customers in New York:
       CustomerID         Name      City
2            6390   Jane Smith  New York
6            1466   Jane Smith  New York
8            6578      Unknown  New York
21           9666   Jane Smith  New York
22           3558  Emily White  New York
...           ...          ...       ...
3656         2812  Emily White  New York
3664         4340     John Doe  New York
3742         1991  Emily White  New York
3764         8876   Alex Brown  New York
3911         3153  Emily White  New York

[203 rows x 3 columns]
```

3: Total Purchase Amount

In [69]:
```
total_purchase_amount = data['PurchaseAmount'].sum()
```

In [71]:
```
print("\nTotal Purchase Amount:", total_purchase_amount)
```

```
Total Purchase Amount: 651161.903274359
```

4: Total Customer Count

In [75]:
```
data['CustomerID'].sum()
```

Out[75]:
```
7148171
```

In [77]:
```
total_unique_customers = data['CustomerID'].nunique()
```

In [79]:
```
print("\nTotal Unique Customers:", total_unique_customers)
```

```
Total Unique Customers: 1295
```

5: Average Purchase Amount

In [83]:
```
average_purchase_amount = data['PurchaseAmount'].mean()
```

In [85]:
```
print("\nAverage Purchase Amount:", average_purchase_amount)
```

```
Average Purchase Amount: 502.8277245361845
```

Part 2: Intermediate Analysis and Aggregation

1. Purchase Analysis by Product Category

In [89]:
```
total_purchase_by_category = data.groupby('ProductCategory')['PurchaseAmount'].sum(
avg_age_by_category = data.groupby('ProductCategory')['Age'].mean()
```

In [91]:
```
print("\nTotal Purchase by Product Category:\n", total_purchase_by_category)
print("\nAverage Age by Product Category:\n", avg_age_by_category)
```

```
Total Purchase by Product Category:
 ProductCategory
Automobile     117086.672569
Electronics     98657.482141
Fashion        103634.552141
Groceries      103681.531713
Home Decor     107606.572997
Misc            10470.680000
Unknown        110024.411713
Name: PurchaseAmount, dtype: float64

Average Age by Product Category:
 ProductCategory
Automobile     47.839286
Electronics    47.690355
Fashion        47.251185
Groceries      48.046729
Home Decor     47.652174
Misc           52.681818
Unknown        47.736364
Name: Age, dtype: float64
```

2. Recent Purchase Analysis

In [95]:

```
latest_date = data['PurchaseDate'].max()
recent_purchases = data[data['PurchaseDate'] >= (latest_date - pd.Timedelta(days=30
```

In [97]:

```
print("\nRecent Purchases (Last 30 Days):\n", recent_purchases)
```

```
Recent Purchases (Last 30 Days):
      CustomerID          Name   Age   Gender         City  PurchaseAmount  \
23          8849  Chris Johnson  28.0  Unknown      Unknown          915.55
77          3062    Emily White  43.0  Unknown      Houston           75.75
177         1202    Emily White  73.0  Unknown  Los Angeles          786.08
303         8806     Alex Brown  49.0  Unknown      Houston          436.41
386         8253       John Doe  24.0   Female  Los Angeles          935.73
543         6029       John Doe  68.0     Male     New York           19.46
569         5468     Jane Smith  60.0  Unknown     New York          494.38
588         4324        Unknown  58.0  Unknown      Houston          664.42
608         5784    Emily White  75.0  Unknown     New York          564.36
693         2887    Emily White  18.0     Male      Phoenix          172.01
791         9837    Emily White  24.0    Other      Phoenix          963.35
865         2970     Alex Brown  58.0  Unknown  Los Angeles          850.25
896         4756        Unknown  21.0  Unknown      Chicago          286.05
900         7898    Emily White  29.0   Female      Houston          918.46
912         1827  Chris Johnson  56.0  Unknown      Phoenix          407.90
1001        5873        Unknown  67.0     Male      Unknown           87.09
1104        9340    Emily White  55.0   Female      Houston          165.07
1461        8014        Unknown  18.0     Male  Los Angeles          886.43
1517        1158    Emily White  76.0     Male      Houston          473.61
2066        9706     Jane Smith  38.0     Male      Chicago          746.79
2392        1948     Jane Smith  36.0  Unknown     New York           44.65
2830        2588     Alex Brown  78.0    Other      Unknown          908.55
2912        9418       John Doe  23.0     Male      Unknown          489.58
2932        4220  Chris Johnson  45.0     Male      Phoenix          836.62
3030        8763       John Doe  72.0   Female  Los Angeles          184.31
3152        7696     Jane Smith  60.0    Other      Unknown           65.65
3217        2104    Emily White  36.0     Male  Los Angeles          924.24
3664        4340       John Doe  68.0  Unknown     New York          486.43

    PurchaseDate ProductCategory
23    2023-12-22      Automobile
77    2023-12-09      Home Decor
177   2023-12-08         Unknown
303   2023-12-16         Fashion
386   2023-12-13     Electronics
543   2023-12-31      Automobile
569   2023-12-24      Automobile
588   2023-12-17            Misc
608   2023-12-26       Groceries
693   2023-12-28     Electronics
791   2023-12-04       Groceries
865   2023-12-05       Groceries
896   2023-12-23         Fashion
900   2023-12-30      Automobile
912   2023-12-10      Automobile
1001  2024-01-01         Unknown
1104  2023-12-27         Fashion
1461  2023-12-07         Unknown
1517  2023-12-19     Electronics
2066  2023-12-06         Fashion
2392  2023-12-29         Unknown
2830  2023-12-20      Automobile
2912  2023-12-14      Automobile
2932  2023-12-21       Groceries
```

```
3030   2023-12-15         Unknown
3152   2023-12-11         Unknown
3217   2023-12-12       Groceries
3664   2023-12-18      Home Decor
```

3. Gender-Based Purchase Analysis

In [101…
```
total_purchase_by_gender = data.groupby('Gender')['PurchaseAmount'].sum()
```

In [103…
```
print("\nTotal Purchase by Gender:\n", total_purchase_by_gender)
```

```
Total Purchase by Gender:
 Gender
Female     153989.623426
Male       176721.202569
Other      156279.662997
Unknown    164171.414282
Name: PurchaseAmount, dtype: float64
```

4. Age-Based Purchase Segmentation

In [107…
```
def age_group(age):
    if age < 30:
        return 'Below 30'
    elif 30 <= age < 40:
        return '30-40'
    elif 40 <= age < 50:
        return '40-50'
    else:
        return '50+'
```

In [109…
```
data['AgeGroup'] = data['Age'].apply(age_group)
total_purchase_by_age_group = data.groupby('AgeGroup')['PurchaseAmount'].sum()
```

In [111…
```
print("\nTotal Purchase by Age Group:\n", total_purchase_by_age_group)
```

```
Total Purchase by Age Group:
 AgeGroup
30-40       103970.981713
40-50       108606.292997
50+         305804.595995
Below 30    132780.032569
Name: PurchaseAmount, dtype: float64
```

5. Top Transactions

In [113…
```
top_transactions = data.nlargest(5, 'PurchaseAmount')[['CustomerID', 'Name', 'Produ
```

In [115…
```
print("\nTop 5 Transactions:\n", top_transactions)
```

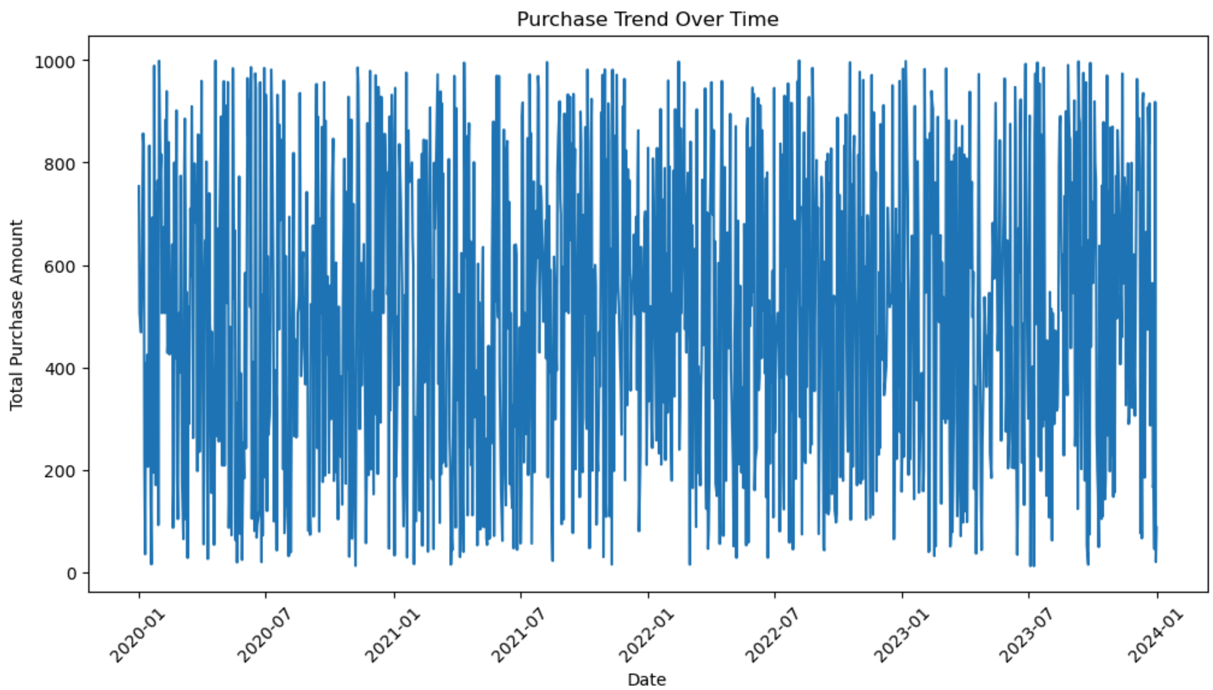
```
Top 5 Transactions:
       CustomerID          Name ProductCategory  PurchaseAmount
2100         4489       Unknown     Electronics          999.46
3627         3890      John Doe            Misc          999.16
7            5426 Chris Johnson      Home Decor          998.86
1900         9648    Jane Smith       Groceries          998.29
905          1814   Emily White     Electronics          997.72
```

Part 3: Advanced Analysis and Insights

1. Purchase Trend Analysis

In [117…

```
plt.figure(figsize=(12,6))
data.groupby('PurchaseDate')['PurchaseAmount'].sum().plot()
plt.xlabel('Date')
plt.ylabel('Total Purchase Amount')
plt.title('Purchase Trend Over Time')
plt.xticks(rotation=45)
plt.show()
```



2. City-Based Purchase Comparison

In [119…

```
avg_purchase_by_city = data.groupby('City')['PurchaseAmount'].mean()
highest_avg_city = avg_purchase_by_city.idxmax()
```

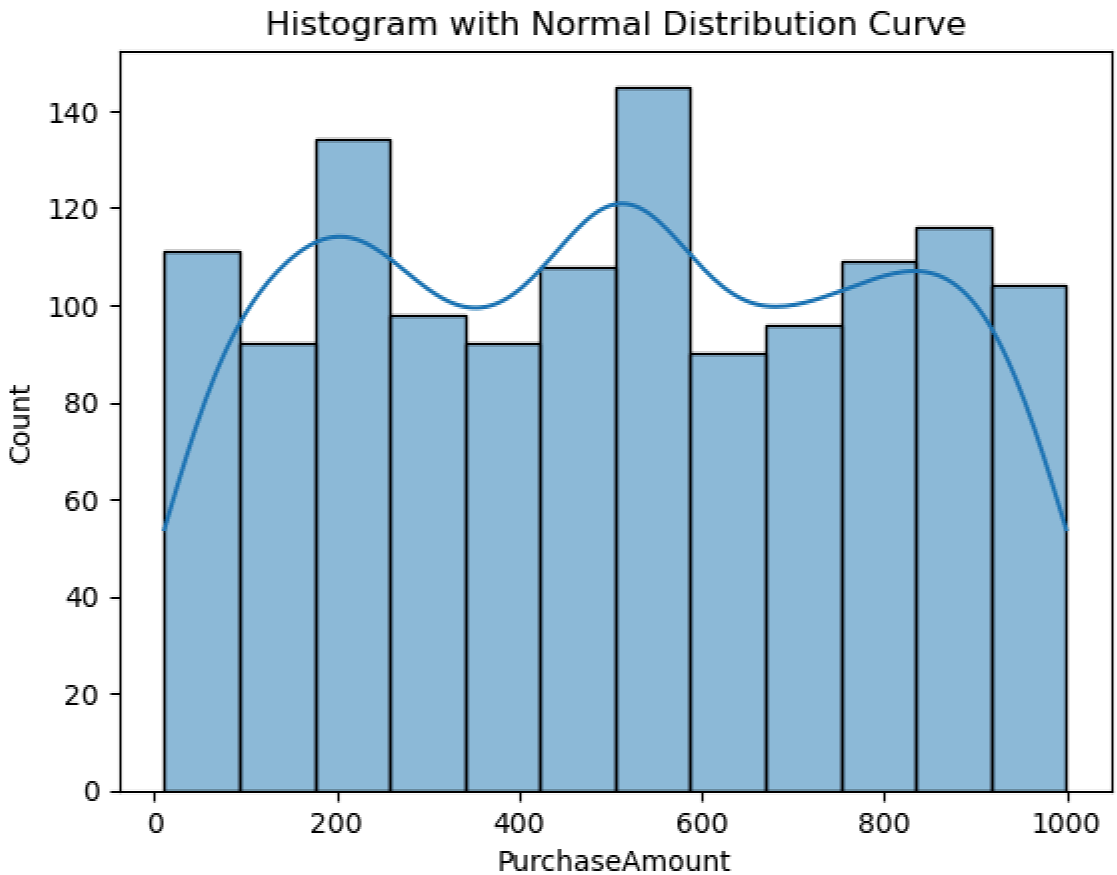
In [121…

```
print("\nCity with Highest Average Purchase Amount:", highest_avg_city)
```

```
City with Highest Average Purchase Amount: Chicago
```

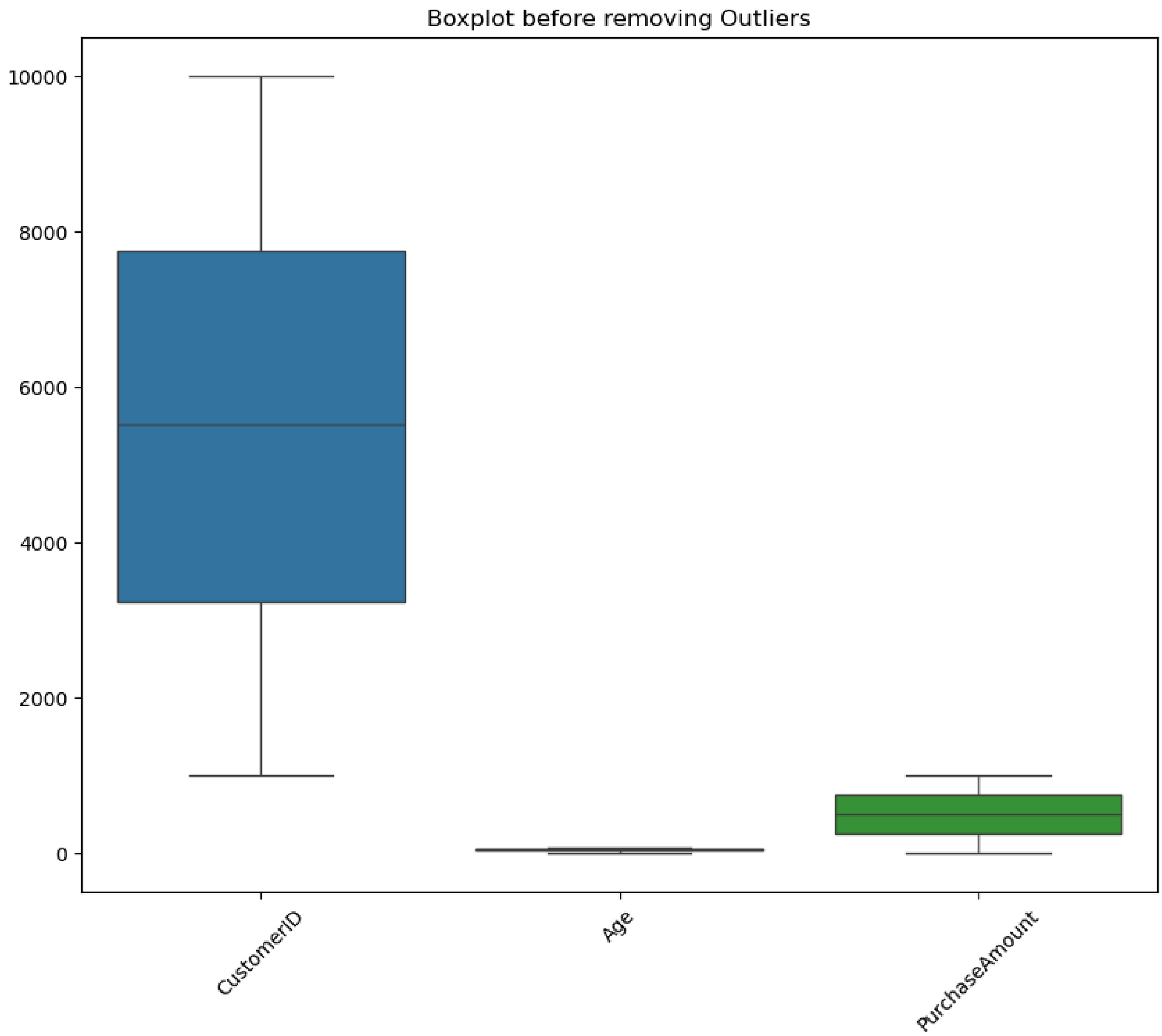
In [123…

```
sns.histplot(data['PurchaseAmount'], kde=True)
plt.title('Histogram with Normal Distribution Curve')
plt.show()
```



In [125…

```
plt.figure(figsize=(10, 8))
sns.boxplot(data=data[num_columns])
plt.title('Boxplot before removing Outliers')
plt.xticks(rotation=45)
plt.show()
```



as you can see there is no outliers in the data.

In [127…

```
data['ProductCategory']
```

Out[127…

```
0          Unknown
1        Groceries
2       Home Decor
3          Unknown
4       Home Decor
           ...
3925       Fashion
3936     Groceries
3938    Automobile
3941       Fashion
3948    Home Decor
Name: ProductCategory, Length: 1295, dtype: object
```

4. Product Category Comparison (Statistical Analysis)

In [135…

```
import seaborn as sns
from scipy import stats
```

Got the unique product categories

In [129…
```
categories = data['ProductCategory'].unique()
```

In [131…
```
print(categories)
```

```
['Unknown' 'Groceries' 'Home Decor' 'Fashion' 'Electronics' 'Automobile'
 'Misc']
```

Created a list of data groups, excluding empty categories

And Checked if there are at least two groups to perform ANOVA

In [133…
```
num_groups = len(categories)

print("Unique Product Categories:", categories)
print("Number of Groups:", num_groups)
```

```
Unique Product Categories: ['Unknown' 'Groceries' 'Home Decor' 'Fashion' 'Electronic
s' 'Automobile'
 'Misc']
Number of Groups: 7
```

In [147…
```
data_groups = [data[data['ProductCategory'] == cat]['PurchaseAmount'] for cat in ca
```

In [151…
```
if len(data_groups) > 1:
    anova_result = stats.f_oneway(*data_groups)
    print("ANOVA Test Result:", anova_result)
else:
    print("Not enough data groups for ANOVA analysis")
```

```
ANOVA Test Result: F_onewayResult(statistic=0.5468369579646548, pvalue=0.77271933786
86309)
```

Got the result as "statistic=0.5468369579646548, pvalue=0.7727193378686309"

F- Statistic is 0.547 this value is the ratio in the groups and ratio between the groups.

This indicates that there can be significant difference between the groups .

P-Value is 0.773 the P-Value is a crucial measure used to access whether the results of this

test are statistically significant. It represents the likelihood that the observed outcomes occurred purely by chance.

If the P-Value is smaller than the chosen threshold (usually 0.05), we can reject the null hypothesis

and can say there is a statistically significant difference between the groups.

If P-Value is 0.05 or higher, we cannot reject null hypothesis and can say there is

no statistically significant difference between the groups.

P-Value is 0.773, which much higher than the usual significance level of 0.05, so we

fail to reject the null hypothesis, it means there is no strong evidence to suggest a

statistically significant difference in " PurchaseAmount" between the "ProductCategory" groups in the data

There is not a strong evidence that means PurchaseAmount are different across the product categories.

I am really sorry as did not have that time to visulize this in power BI as i got this project yesterday

evening so i did not have time to visulize it in power BI , if i will get time than forsure i can create

vizulization in Power BI as well.

In [ ]:

In [ ]: