**Task 1: Exploratory Data Analysis (EDA)**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

customers = pd.read\_csv('Customers.csv')

products = pd.read\_csv('Products.csv')

transactions = pd.read\_csv('Transactions.csv')

print(customers.info())

print(products.describe())

print(transactions.isnull().sum())

sns.histplot(transactions['TotalValue'])

plt.title('Distribution of Total Transaction Values')

plt.show()

**Task 2: Lookalike Model**

from sklearn.metrics.pairwise import cosine\_similarity

data = pd.merge(transactions, customers, on='CustomerID')

data = pd.merge(data, products, on='ProductID')

customer\_product\_matrix = data.pivot\_table(index='CustomerID', columns='ProductID', values='Quantity', fill\_value=0)

similarity\_matrix = cosine\_similarity(customer\_product\_matrix)

similarity\_df = pd.DataFrame(similarity\_matrix, index=customer\_product\_matrix.index, columns=customer\_product\_matrix.index)

lookalikes = {}

for customer in similarity\_df.index[:20]:

top\_similar = similarity\_df[customer].sort\_values(ascending=False)[1:4]

lookalikes[customer] = list(top\_similar.index)

lookalike\_df = pd.DataFrame.from\_dict(lookalikes, orient='index', columns=['Lookalike1', 'Lookalike2', 'Lookalike3'])

lookalike\_df.to\_csv('Lookalike.csv', index\_label='CustomerID')

**Task 3: Customer Segmentation**

from sklearn.cluster import KMeans

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import davies\_bouldin\_score

import matplotlib.pyplot as plt

features = data[['Price', 'Quantity', 'TotalValue']] # Example features

scaler = StandardScaler()

scaled\_features = scaler.fit\_transform(features)

kmeans = KMeans(n\_clusters=3, random\_state=42)

clusters = kmeans.fit\_predict(scaled\_features)

data['Cluster'] = clusters

db\_index = davies\_bouldin\_score(scaled\_features, clusters)

print(f"Davies-Bouldin Index: {db\_index}")

sns.scatterplot(x=scaled\_features[:, 0], y=scaled\_features[:, 1], hue=clusters, palette='viridis')

plt.title('Customer Clusters')

plt.show()