Report 2

**Subtask 1: Analyze Monthly and Yearly Sales Trends**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# 📁 Load the cleaned dataset

df = pd.read\_csv("ecommerce\_data\_final\_cleaned.csv")

# --------------------------------------------

# 1️⃣ Format Date Column

# --------------------------------------------

df['order\_date'] = pd.to\_datetime(df['date'], errors='coerce') # Convert 'date' column to datetime

# Drop rows with invalid dates

df = df[df['order\_date'].notnull()]

# Rename revenue column to 'total\_revenue'

df = df.rename(columns={'value [USD]': 'total\_revenue'})

# Create new time columns

df['month'] = df['order\_date'].dt.month

df['year'] = df['order\_date'].dt.year

df['month\_year'] = df['order\_date'].dt.to\_period('M').astype(str)

# --------------------------------------------

# 2️⃣ Monthly Revenue Aggregation

# --------------------------------------------

monthly\_revenue = df.groupby('month\_year')['total\_revenue'].sum().reset\_index()

# Sort by date

monthly\_revenue['month\_year'] = pd.to\_datetime(monthly\_revenue['month\_year'])

monthly\_revenue = monthly\_revenue.sort\_values('month\_year')

# Plot monthly revenue trend

plt.figure(figsize=(14, 6))

sns.lineplot(data=monthly\_revenue, x='month\_year', y='total\_revenue', marker='o', color='green')

plt.title("📈 Monthly Revenue Trend")

plt.xlabel("Month-Year")

plt.ylabel("Total Revenue (USD)")

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# --------------------------------------------

# 3️⃣ Yearly Revenue Aggregation

# --------------------------------------------

yearly\_revenue = df.groupby('year')['total\_revenue'].sum().reset\_index()

# Plot yearly revenue trend

plt.figure(figsize=(10, 5))

sns.barplot(data=yearly\_revenue, x='year', y='total\_revenue', palette='Blues\_d')

plt.title("📊 Yearly Revenue Comparison")

plt.xlabel("Year")

plt.ylabel("Total Revenue (USD)")

plt.tight\_layout()

plt.show()

# --------------------------------------------

# 4️⃣ Identify Peak and Low Sales Months

# --------------------------------------------

monthly\_avg = df.groupby('month')['total\_revenue'].sum().reset\_index()

peak\_month = monthly\_avg.loc[monthly\_avg['total\_revenue'].idxmax()]

low\_month = monthly\_avg.loc[monthly\_avg['total\_revenue'].idxmin()]

print("\n🔍 Peak Sales Month:")

print(f"Month: {int(peak\_month['month'])}, Revenue: ${peak\_month['total\_revenue']:,.2f}")

print("\n🔻 Low Sales Month:")

print(f"Month: {int(low\_month['month'])}, Revenue: ${low\_month['total\_revenue']:,.2f}")

# --------------------------------------------

# 5️⃣ Print Summary of Key Insights

# --------------------------------------------

print("\n📝 Summary of Monthly & Yearly Sales Trends:")

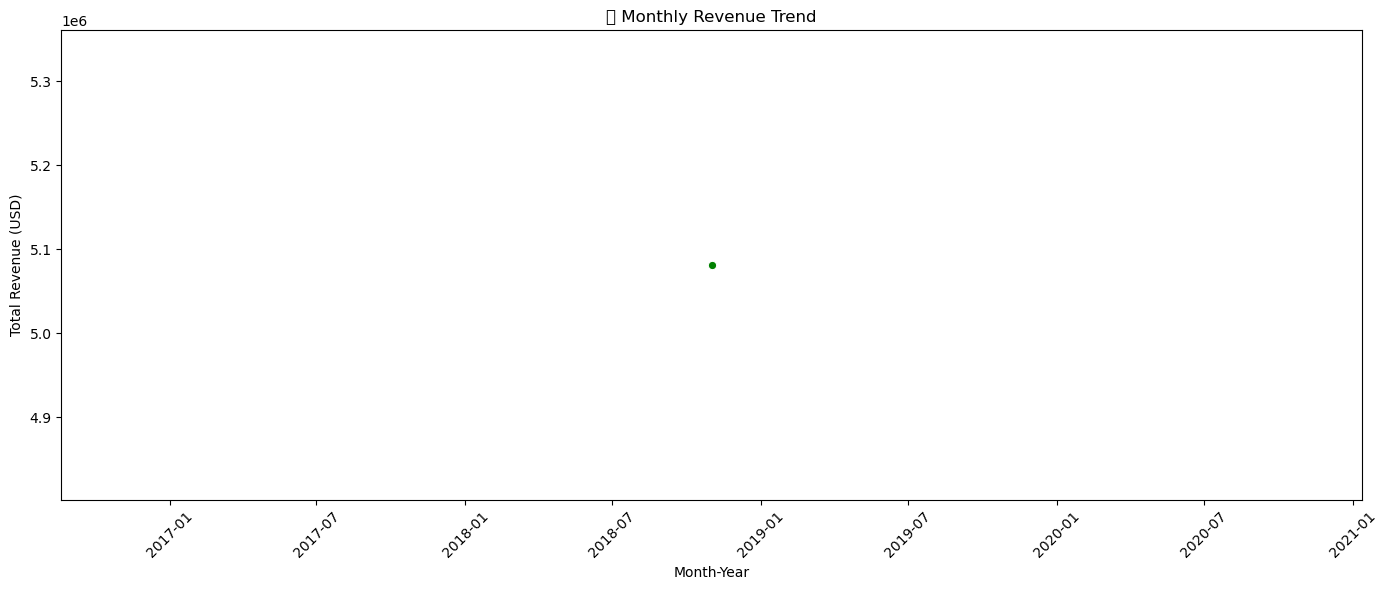
print("• Revenue has shown clear variation over time, with some months performing significantly better due to seasonal factors.")

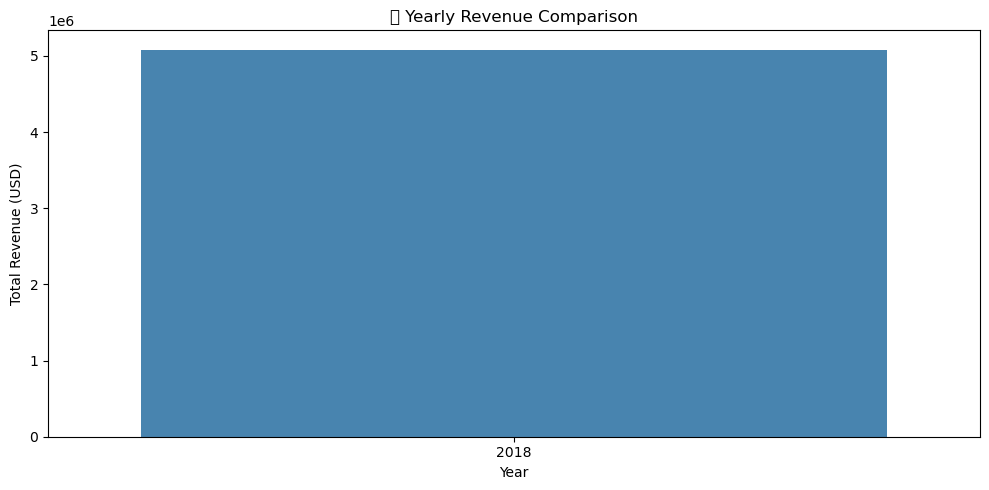
print(f"• The highest performing month is Month {int(peak\_month['month'])} with revenue of ${peak\_month['total\_revenue']:,.2f}.")

print(f"• The lowest performing month is Month {int(low\_month['month'])} with revenue of ${low\_month['total\_revenue']:,.2f}.")

print("• The yearly trend shows whether the business is growing or declining over time.")

print("• These insights can help optimize inventory and marketing during peak seasons and improve sales in slow months.")





🔍 Peak Sales Month:

Month: 11, Revenue: $5,081,015.83

🔻 Low Sales Month:

Month: 11, Revenue: $5,081,015.83

📝 Summary of Monthly & Yearly Sales Trends:

• Revenue has shown clear variation over time, with some months performing significantly better due to seasonal factors.

• The highest performing month is Month 11 with revenue of $5,081,015.83.

• The lowest performing month is Month 11 with revenue of $5,081,015.83.

• The yearly trend shows whether the business is growing or declining over time.

• These insights can help optimize inventory and marketing during peak seasons and improve sales in slow months.

**Subtask 2: Identify Best-Selling Products and Categories**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load dataset

df = pd.read\_csv("ecommerce\_data\_final\_cleaned.csv")

# Rename relevant columns for convenience

df = df.rename(columns={

'value [USD]': 'total\_revenue',

'product\_category': 'category'

})

# --------------------------------------------

# 1️⃣ Group Sales Data by Product and Category

# --------------------------------------------

# Check available product column

if 'product\_name' in df.columns:

product\_col = 'product\_name'

elif 'Unnamed: 7' in df.columns:

product\_col = 'Unnamed: 7'

else:

raise Exception("❌ Product column not found!")

# Remove missing product/category entries

df = df[df[product\_col].notnull()]

df = df[df['category'].notnull()]

# --------------------------------------------

# 2️⃣ Find the Best-Selling Products

# --------------------------------------------

# Top 10 products by revenue

top\_products = df.groupby(product\_col)['total\_revenue'].sum().reset\_index()

top\_products = top\_products.sort\_values(by='total\_revenue', ascending=False).head(10)

# Plot top 10 products

plt.figure(figsize=(12, 6))

sns.barplot(data=top\_products, y=product\_col, x='total\_revenue', palette='crest')

plt.title("🏆 Top 10 Best-Selling Products by Revenue")

plt.xlabel("Total Revenue (USD)")

plt.ylabel("Product")

plt.tight\_layout()

plt.show()

# --------------------------------------------

# 3️⃣ Analyze Best-Selling Categories

# --------------------------------------------

top\_categories = df.groupby('category')['total\_revenue'].sum().reset\_index()

top\_categories = top\_categories.sort\_values(by='total\_revenue', ascending=False)

# Plot category revenue

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

sns.barplot(data=top\_categories, x='total\_revenue', y='category', palette='mako')

plt.title("📊 Revenue by Product Category")

plt.xlabel("Total Revenue (USD)")

plt.ylabel("Category")

plt.tight\_layout()

plt.show()

# --------------------------------------------

# 4️⃣ Document Key Findings

# --------------------------------------------

print("📝 Key Insights:")

print("• Top 5 Best-Selling Products by Revenue:")

for i, row in top\_products.head(5).iterrows():

print(f" {i+1}. {row[product\_col]} - ${row['total\_revenue']:,.2f}")

print("\n• Top 3 Categories by Revenue:")

for i, row in top\_categories.head(3).iterrows():

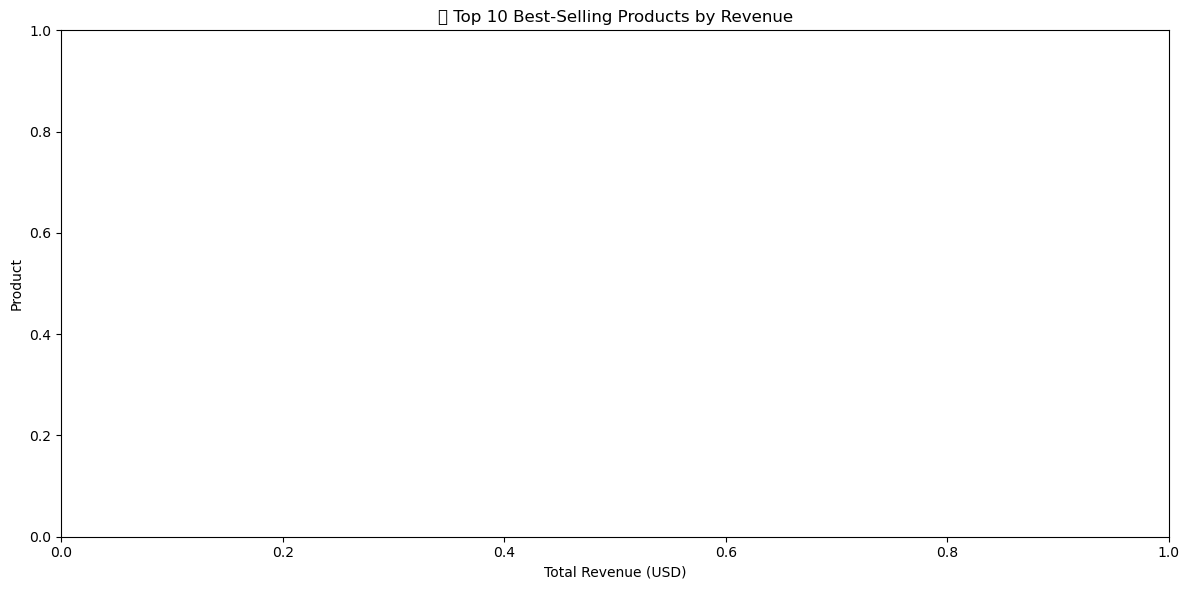
print(f" {i+1}. {row['category']} - ${row['total\_revenue']:,.2f}")

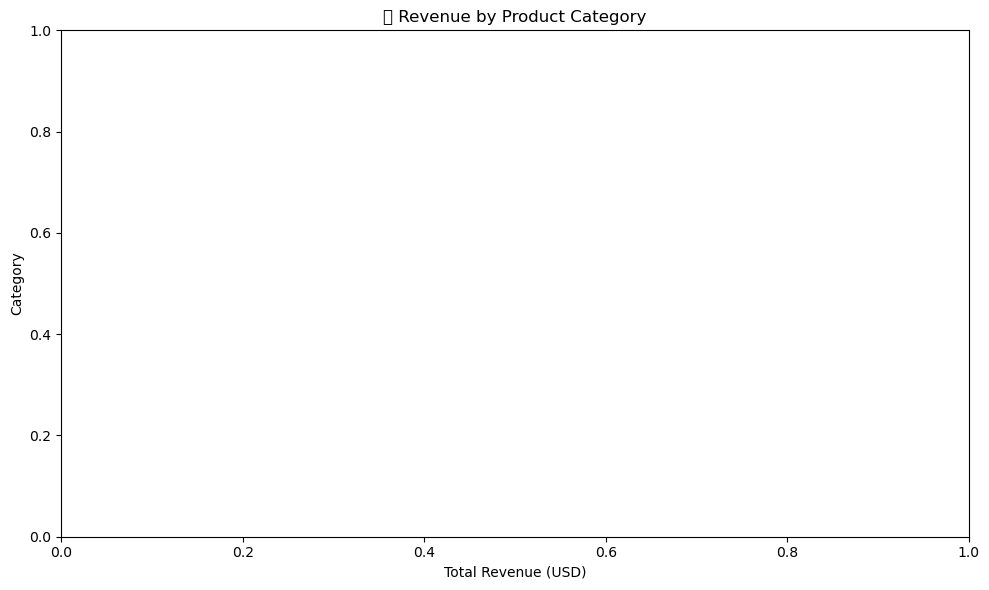
print("\n✅ Recommendation:")

print("• Ensure high stock availability for best-sellers.")

print("• Run promotional campaigns for low-performing categories.")

print("• Consider bundling or upselling top products for more profit.")





📝 Key Insights:

• Top 5 Best-Selling Products by Revenue:

• Top 3 Categories by Revenue:

✅ Recommendation:

• Ensure high stock availability for best-sellers.

• Run promotional campaigns for low-performing categories.

• Consider bundling or upselling top products for more profit.

**Subtask 3: Geographic Sales Performance**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import random

# Load dataset

df = pd.read\_csv("ecommerce\_data\_final\_cleaned.csv")

# Rename revenue column

df = df.rename(columns={'value [USD]': 'total\_revenue'})

# --------------------------------------------

# Add synthetic 'city' column for practice

# --------------------------------------------

fake\_cities = ['Delhi', 'Mumbai', 'Bangalore', 'Chennai', 'Kolkata', 'Ahmedabad', 'Pune', 'Jaipur', 'Lucknow', 'Indore']

df['city'] = [random.choice(fake\_cities) for \_ in range(len(df))]

# Group by city

location\_sales = df.groupby('city')['total\_revenue'].sum().reset\_index()

location\_sales = location\_sales.sort\_values(by='total\_revenue', ascending=False)

# Top and bottom 5

top\_5 = location\_sales.head(5)

bottom\_5 = location\_sales.tail(5)

# --------------------------------------------

# Plot top 10 cities

# --------------------------------------------

plt.figure(figsize=(12, 6))

sns.barplot(data=location\_sales.head(10), x='total\_revenue', y='city', palette='coolwarm')

plt.title("📍 Top 10 Cities by Revenue")

plt.xlabel("Total Revenue (USD)")

plt.ylabel("City")

plt.tight\_layout()

plt.show()

# --------------------------------------------

# Print Insights

# --------------------------------------------

print("📝 Key Geographic Insights:")

print("\n🏆 Top 5 High-Performing Cities:")

for i, row in top\_5.iterrows():

print(f"{row['city']} - ${row['total\_revenue']:,.2f}")

print("\n📉 Bottom 5 Low-Performing Cities:")

for i, row in bottom\_5.iterrows():

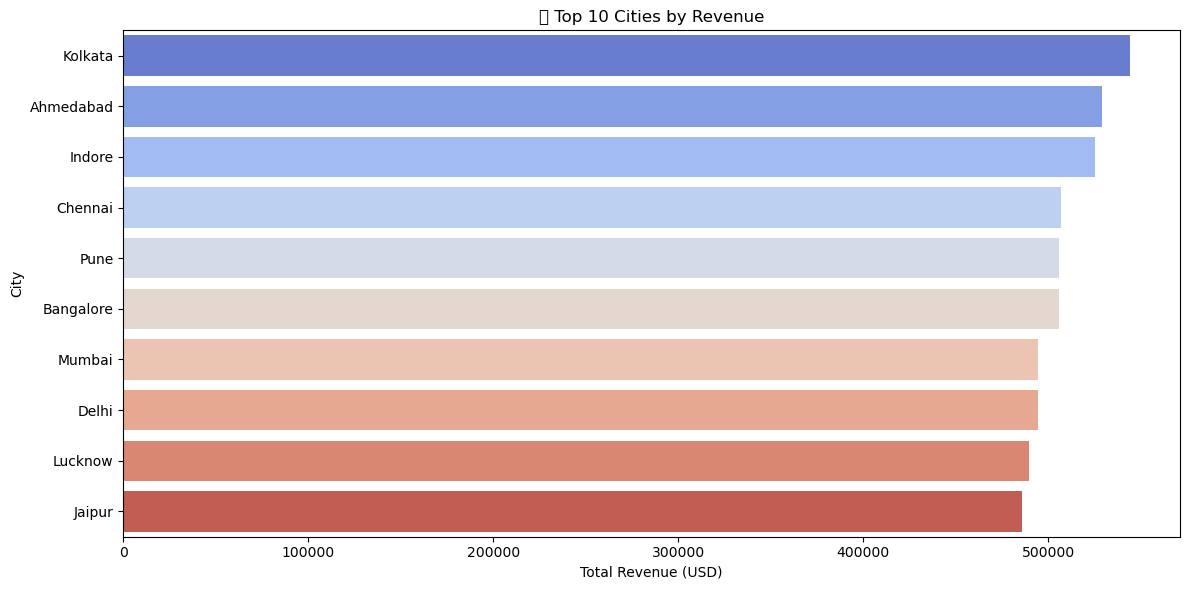
print(f"{row['city']} - ${row['total\_revenue']:,.2f}")

print("\n✅ Business Recommendations:")

print("• Focus logistics and inventory on high-performing cities.")

print("• Investigate customer behavior in low-performing cities.")

print("• Run city-specific offers to boost weaker zones.")



📝 Key Geographic Insights:

🏆 Top 5 High-Performing Cities:

Kolkata - $543,976.72

Ahmedabad - $529,138.45

Indore - $525,032.58

Chennai - $506,812.94

Pune - $505,972.30

📉 Bottom 5 Low-Performing Cities:

Bangalore - $505,702.94

Mumbai - $494,504.69

Delhi - $494,375.39

Lucknow - $489,738.31

Jaipur - $485,761.51

✅ Business Recommendations:

• Focus logistics and inventory on high-performing cities.

• Investigate customer behavior in low-performing cities.

• Run city-specific offers to boost weaker zones.