Laxmi Charitable Trust's
Sheth L.U.J College of Arts & Sir
M.V. College
Of Science & Commerce

RAM KUMAR SINGH

PRACTICAL NO.10

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Load and Inspect the Dataset
df = pd.read_csv("heart.csv")  # Upload or use correct path

# Inspect the first few rows
print(df.head())
```

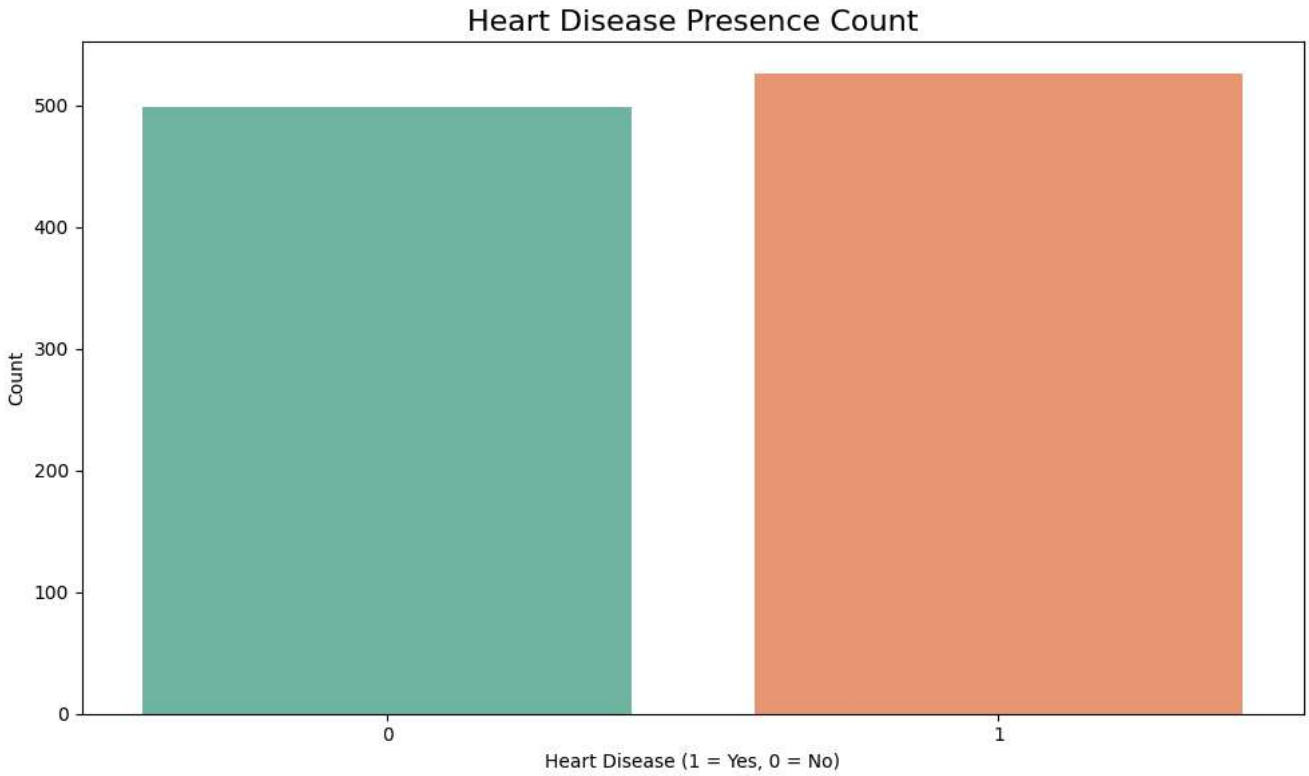
```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   52    1   0       125   212    0        1      168      0      1.0      2
1   53    1   0       140   203    1        0      155      1      3.1      0
2   70    1   0       145   174    0        1      125      1      2.6      0
3   61    1   0       148   203    0        1      161      0      0.0      2
4   62    0   0       138   294    1        1      106      0      1.9      1

   ca  thal  target
0   2     3       0
1   0     3       0
2   0     3       0
3   1     3       0
4   3     2       0
```

```
# Step 2: Data Preparation
# Check for missing values
print("\nMissing Values:\n", df.isnull().sum())

# Fill missing values with median (if any)
df.fillna(df.median(), inplace=True)
```

```
Missing Values:
 age         0
sex          0
cp           0
trestbps     0
chol         0
fbs          0
restecg      0
thalach      0
exang        0
oldpeak      0
slope        0
ca           0
thal         0
target       0
dtype: int64
```

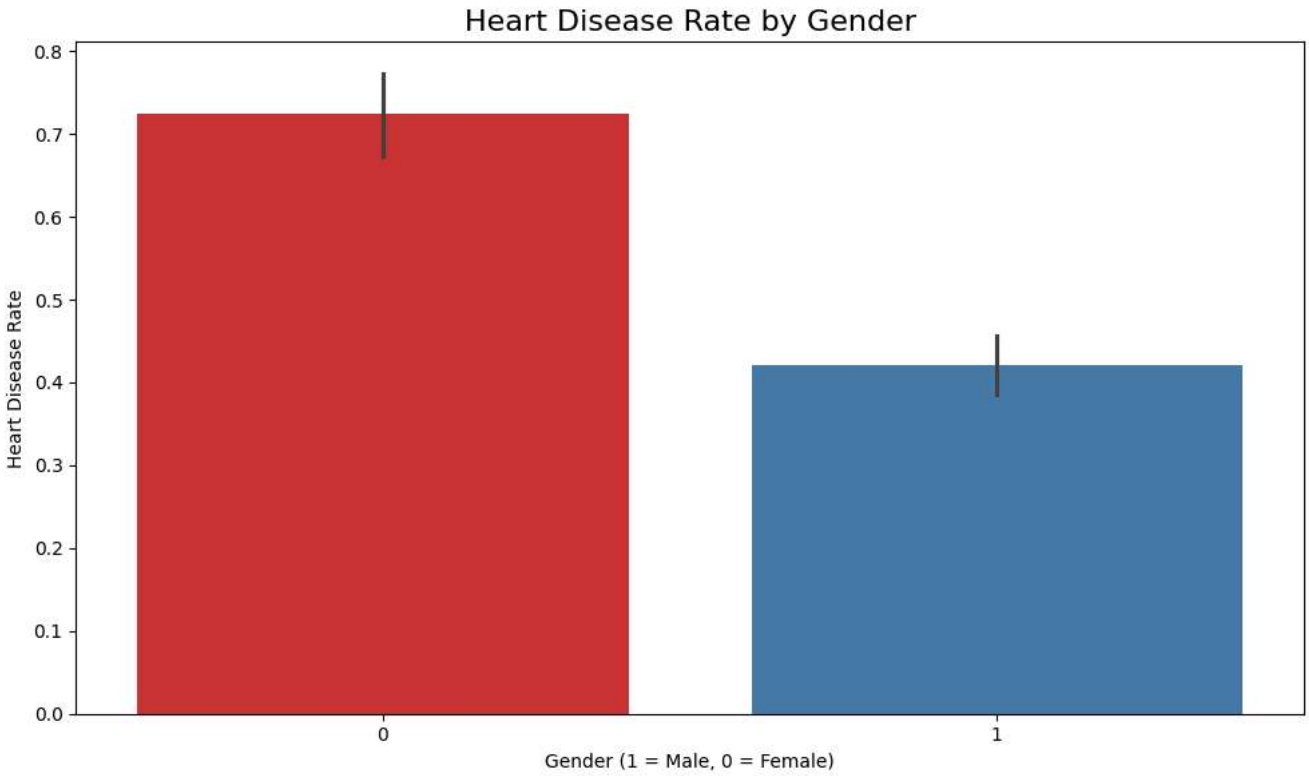
```
# Step 3: Create Visualizations

# 3a. Heart Disease Count (Bar plot)
plt.figure(figsize=(10, 6))
sns.barplot(x=df["target"].value_counts().index,
            y=df["target"].value_counts().values,
            palette="Set2")
plt.title("Heart Disease Presence Count", fontsize=16)
plt.xlabel("Heart Disease (1 = Yes, 0 = No)")
plt.ylabel("Count")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-4030631264.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.barplot(x=df["target"].value_counts().index,
```



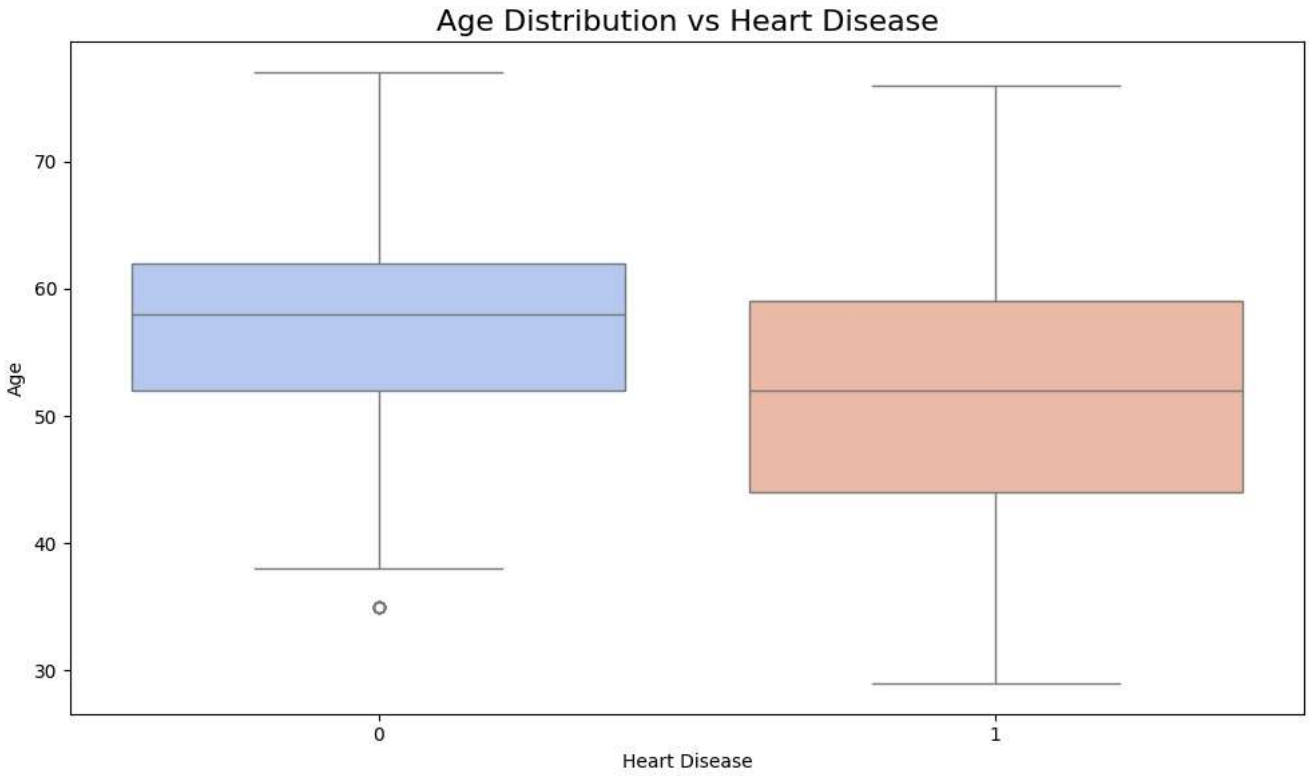

```
# 3b. Gender vs Heart Disease (Bar plot)
plt.figure(figsize=(10, 6))
sns.barplot(x="sex", y="target", data=df, palette="Set1")
plt.title("Heart Disease Rate by Gender", fontsize=16)
plt.xlabel("Gender (1 = Male, 0 = Female)")
plt.ylabel("Heart Disease Rate")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2465445334.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.barplot(x="sex", y="target", data=df, palette="Set1")
```

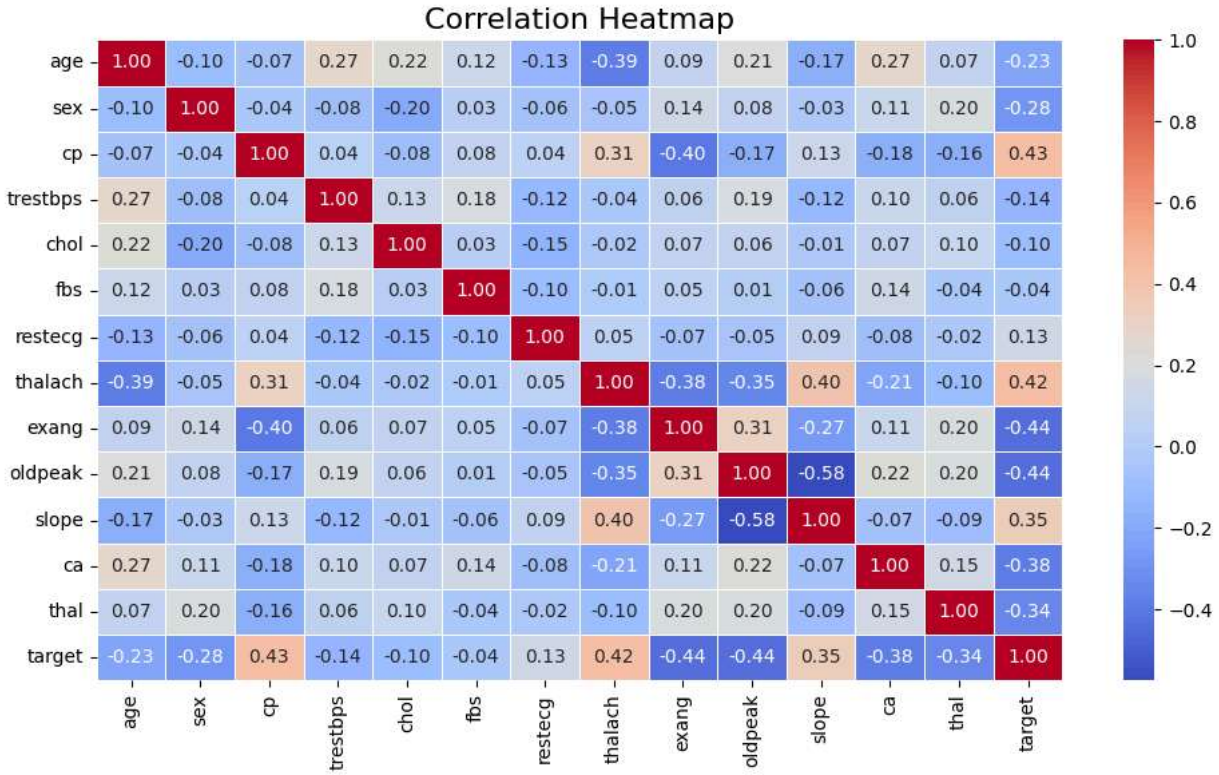



```
# 3c. Age Distribution by Heart Disease (Box plot)
plt.figure(figsize=(10, 6))
sns.boxplot(x="target", y="age", data=df, palette="coolwarm")
plt.title("Age Distribution vs Heart Disease", fontsize=16)
plt.xlabel("Heart Disease")
plt.ylabel("Age")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2011739192.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.boxplot(x="target", y="age", data=df, palette="coolwarm")
```

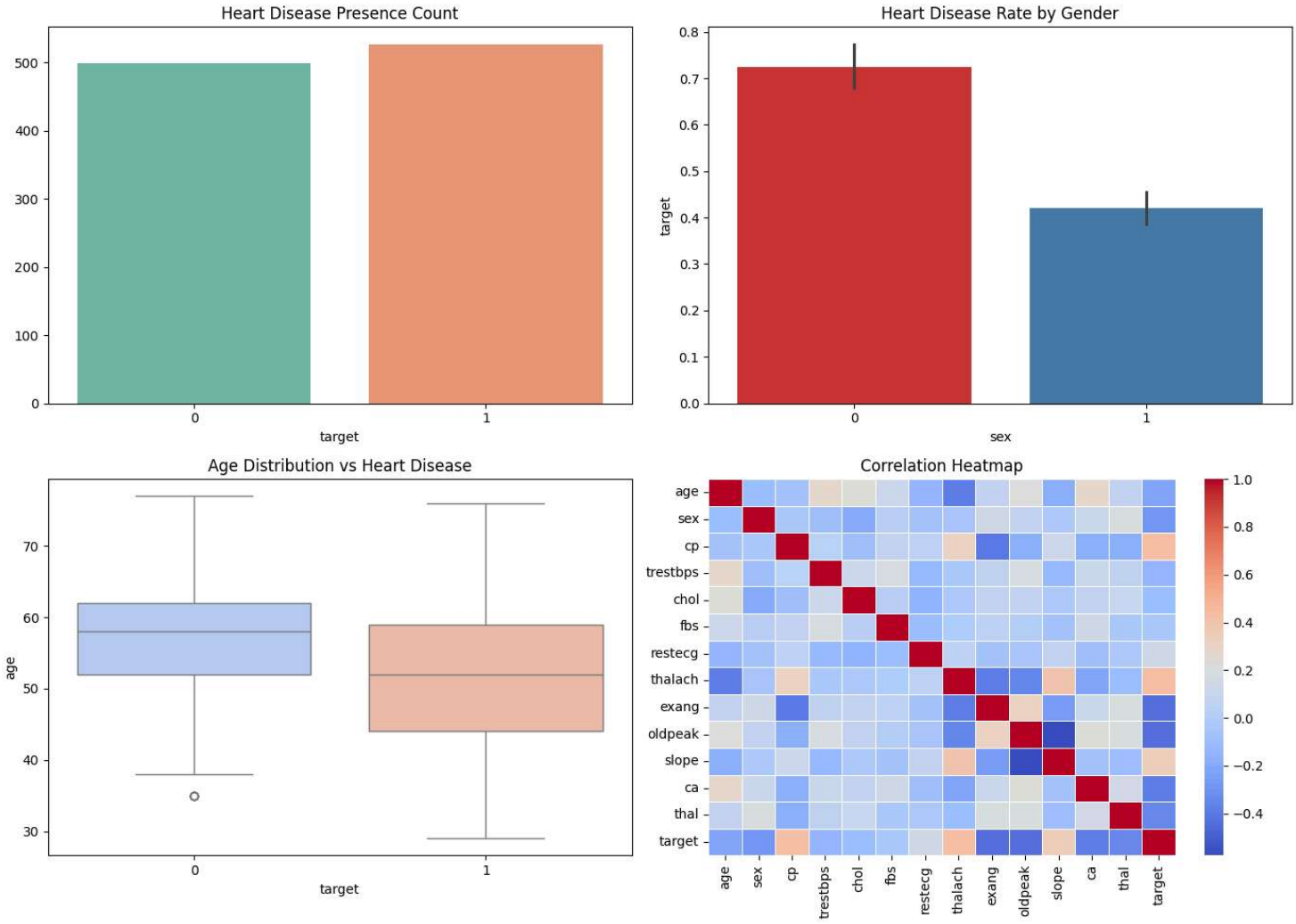



```
# 3d. Heatmap of Correlation between Numerical Features
plt.figure(figsize=(10, 6))
corr = df.corr()
sns.heatmap(corr, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Correlation Heatmap", fontsize=16)
plt.tight_layout()
plt.show()
```



```
# Step 4: Combine Visualizations Like Titanic Practical

fig, axes = plt.subplots(2, 2, figsize=(14, 10))

# Heart Disease Count
sns.barplot(x=df["target"].value_counts().index,
            y=df["target"].value_counts().values,
            palette="Set2", ax=axes[0, 0])
axes[0, 0].set_title("Heart Disease Presence Count", fontsize=12)

# Heart Disease by Gender
sns.barplot(x="sex", y="target", data=df, palette="Set1", ax=axes[0, 1])
axes[0, 1].set_title("Heart Disease Rate by Gender", fontsize=12)

# Age Distribution vs Disease
sns.boxplot(x="target", y="age", data=df, palette="coolwarm", ax=axes[1, 0])
axes[1, 0].set_title("Age Distribution vs Heart Disease", fontsize=12)

# Correlation Heatmap
sns.heatmap(corr, annot=False, cmap="coolwarm", linewidths=0.5, ax=axes[1, 1])
axes[1, 1].set_title("Correlation Heatmap", fontsize=12)

plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-3105601955.py:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.barplot(x=df["target"].value_counts().index,
/tmp/ipython-input-3105601955.py:12: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.barplot(x="sex", y="target", data=df, palette="Set1", ax=axes[0, 1])
/tmp/ipython-input-3105601955.py:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.boxplot(x="target", y="age", data=df, palette="coolwarm", ax=axes[1, 0])
```




```
# Step 5: Key Insights
insights = {
    "Heart Disease Count": "More patients are diagnosed with heart disease (Target=1) than not.",
    "Gender Impact": "Heart disease is more common in males based on dataset distribution.",
    "Age Factor": "Middle-aged to older individuals show higher heart disease impact.",
    "Correlation": "Chest pain type (cp), maximum heart rate (thalach), and oldpeak show strong association with disease."
}

print("\n📌 KEY INSIGHTS:")
for key, value in insights.items():
    print(f"{key}: {value}\n")
```

📌 KEY INSIGHTS:
Heart Disease Count: More patients are diagnosed with heart disease (Target=1) than not.

Gender Impact: Heart disease is more common in males based on dataset distribution.

Age Factor: Middle-aged to older individuals show higher heart disease impact.

Correlation: Chest pain type (cp), maximum heart rate (thalach), and oldpeak show strong association with disease.

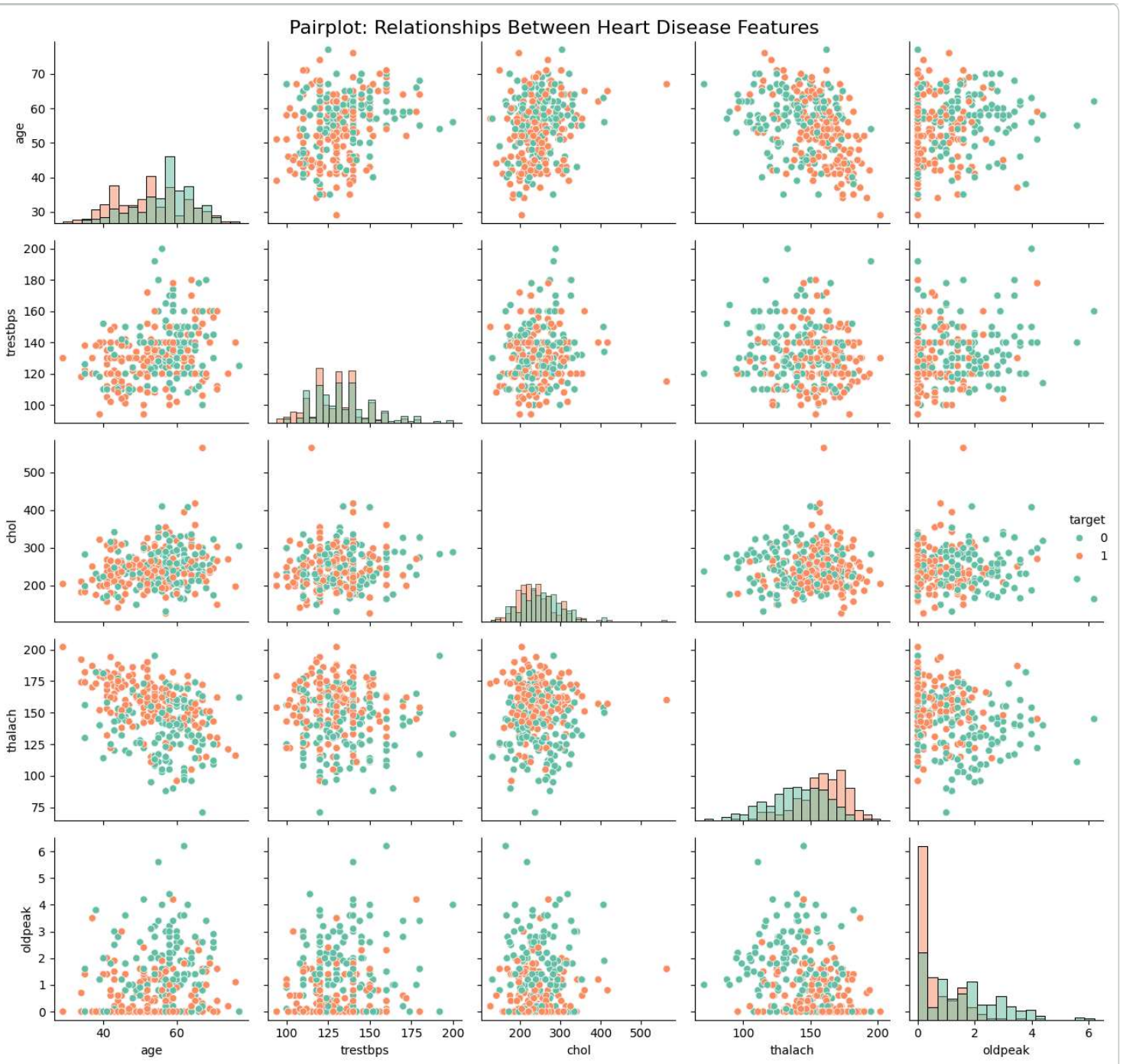
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Step 1: Load and Inspect the Dataset
# Heart Disease dataset (UCI/Kaggle) - make sure heart.csv is in your working directory
df = pd.read_csv("heart.csv")

# Inspect the first few rows
print(df.head())
```

```
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
0   52    1   0       125   212    0        1      168      0      1.0      2
1   53    1   0       140   203    1        0      155      1      3.1      0
2   70    1   0       145   174    0        1      125      1      2.6      0
3   61    1   0       148   203    0        1      161      0      0.0      2
4   62    0   0       138   294    1        1      106      0      1.9      1

   ca  thal  target
0   2     3       0
1   0     3       0
2   0     3       0
3   1     3       0
4   3     2       0
```

```
# Step 2: Data Preparation
# Check for missing values
print(df.isnull().sum())

# If there are any missing values, fill them with median
df = df.fillna(df.median(numeric_only=True))
```

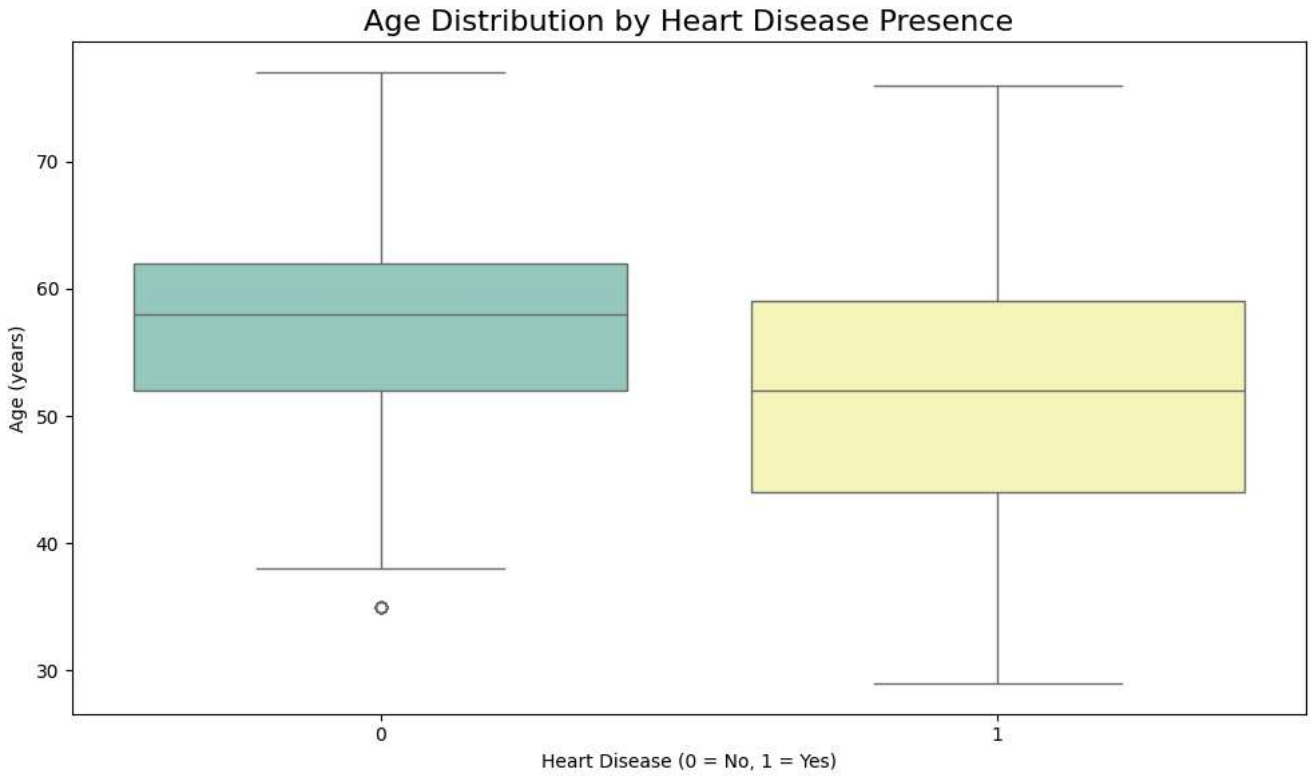
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

```
# Step 3: Create Visualizations

# 3a. Pairplot to show relationships between selected features
# (using a subset so it is readable)
pairplot_features = ["age", "trestbps", "chol", "thalach", "oldpeak", "target"]
sns.pairplot(df[pairplot_features], hue="target", palette="Set2", diag_kind="hist")
plt.suptitle("Pairplot: Relationships Between Heart Disease Features", fontsize=16)
plt.tight_layout()
plt.show()
```



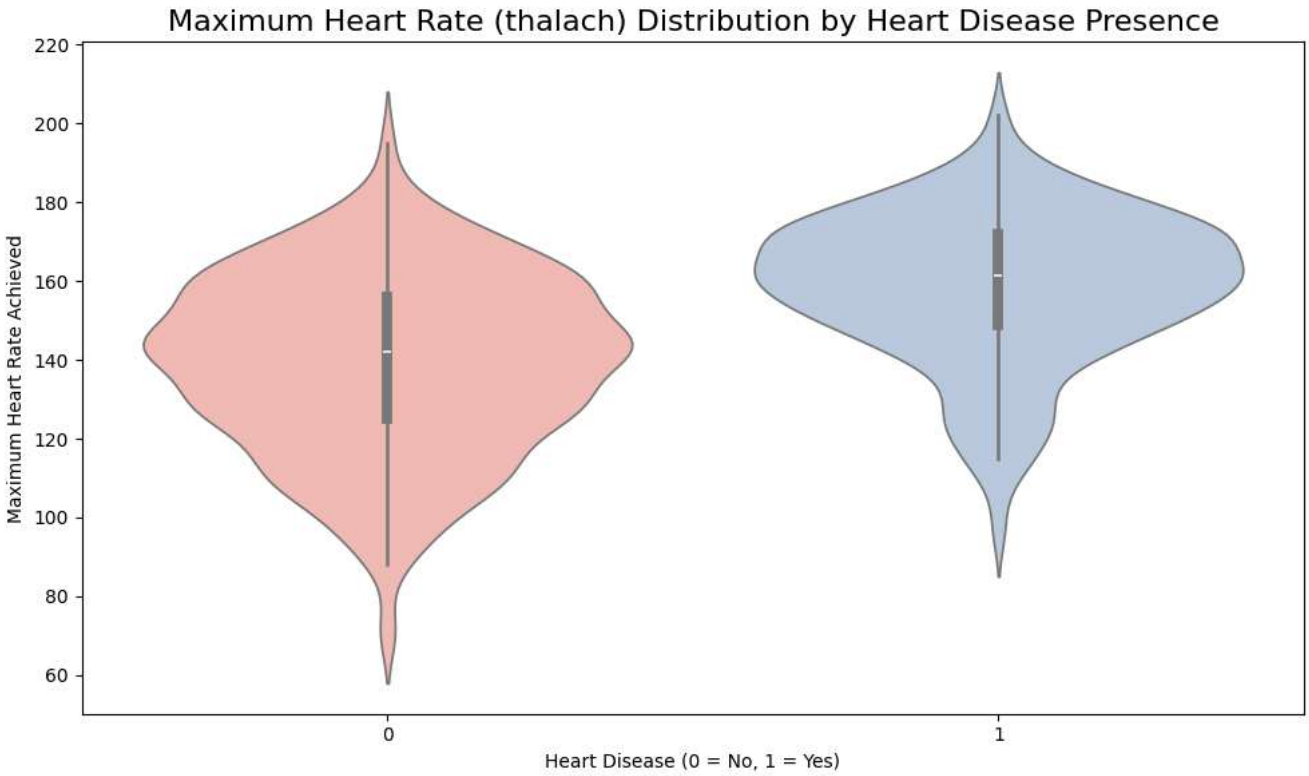

```
# 3b. Boxplot for Age distribution by Heart Disease presence
plt.figure(figsize=(10, 6))
sns.boxplot(x="target", y="age", data=df, palette="Set3")
plt.title("Age Distribution by Heart Disease Presence", fontsize=16)
plt.xlabel("Heart Disease (0 = No, 1 = Yes)")
plt.ylabel("Age (years)")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2033562449.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.boxplot(x="target", y="age", data=df, palette="Set3")
```

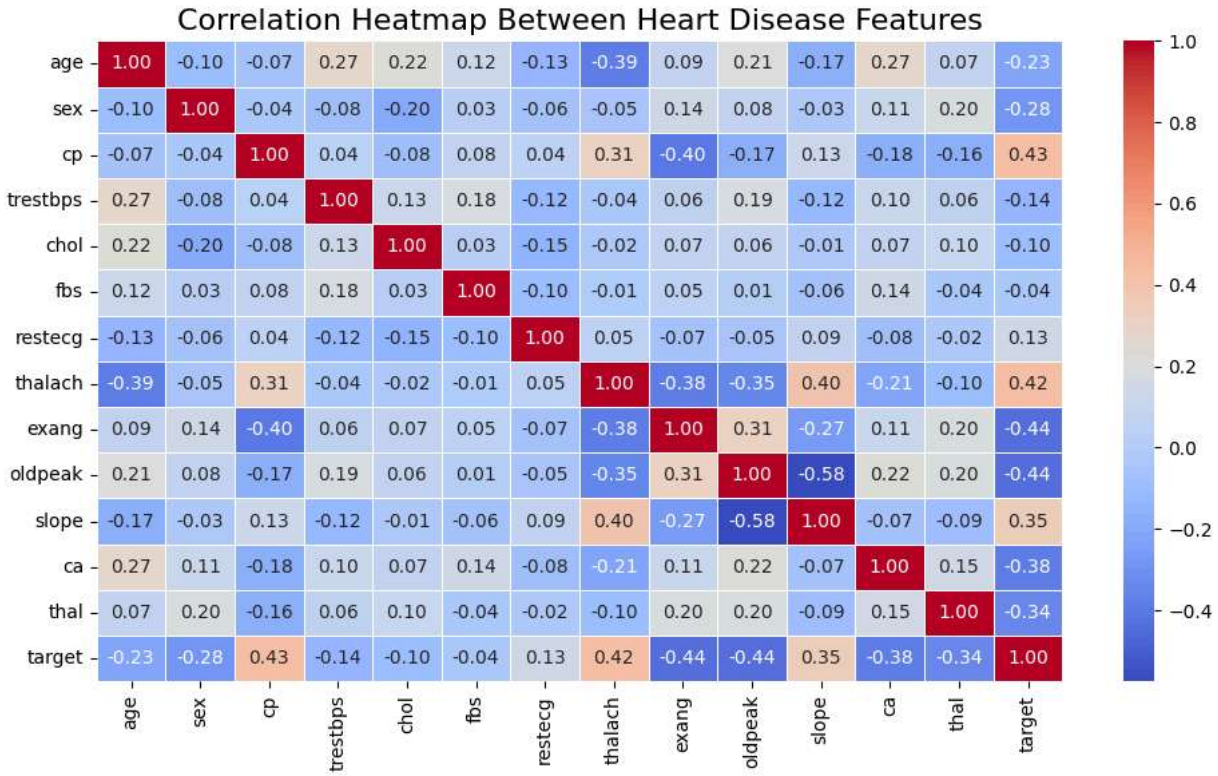



```
# 3c. Violin plot for Maximum Heart Rate (thalach) distribution by Heart Disease presence
plt.figure(figsize=(10, 6))
sns.violinplot(x="target", y="thalach", data=df, palette="Pastel1")
plt.title("Maximum Heart Rate (thalach) Distribution by Heart Disease Presence", fontsize=16)
plt.xlabel("Heart Disease (0 = No, 1 = Yes)")
plt.ylabel("Maximum Heart Rate Achieved")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-1715860433.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.violinplot(x="target", y="thalach", data=df, palette="Pastel1")
```



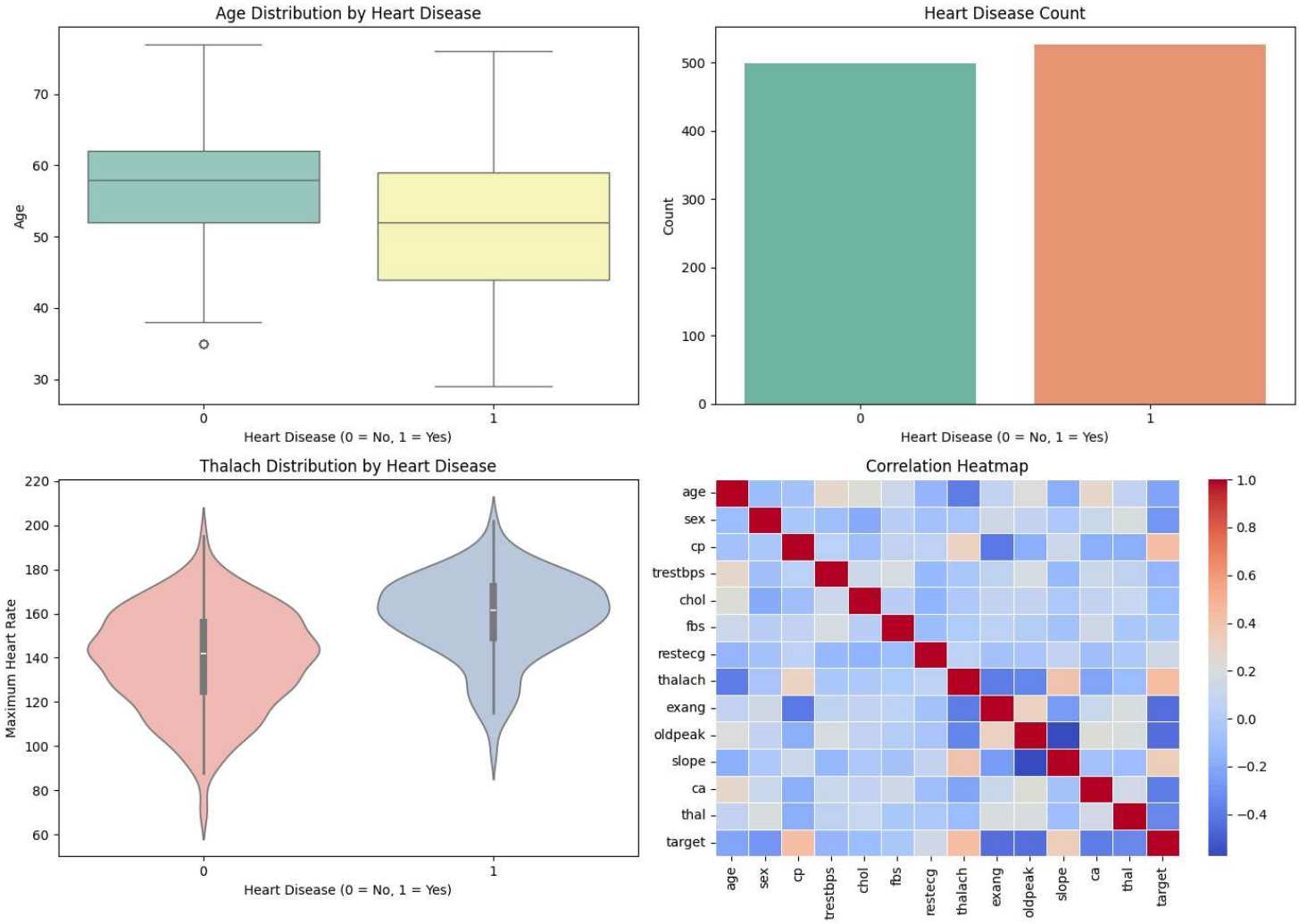

```
# 3d. Heatmap of Correlation between features
plt.figure(figsize=(10, 6))
correlation_matrix = df.corr()
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Correlation Heatmap Between Heart Disease Features", fontsize=16)
plt.tight_layout()
plt.show()
```




```
# Step 4: Combine the Visualizations for Storytelling
# (Pairplot is kept separate, like in your Iris code)
```

```
fig, axes = plt.subplots(2, 2, figsize=(14, 10))

# Boxplot for Age by Heart Disease (top-left)
sns.boxplot(x="target", y="age", data=df, palette="Set3", ax=axes[0, 0])
axes[0, 0].set_title("Age Distribution by Heart Disease", fontsize=12)
axes[0, 0].set_xlabel("Heart Disease (0 = No, 1 = Yes)")
axes[0, 0].set_ylabel("Age")

# Violin plot for Thalach by Heart Disease (bottom-left)
sns.violinplot(x="target", y="thalach", data=df, palette="Pastel1", ax=axes[1, 0])
axes[1, 0].set_title("Thalach Distribution by Heart Disease", fontsize=12)
axes[1, 0].set_xlabel("Heart Disease (0 = No, 1 = Yes)")
axes[1, 0].set_ylabel("Maximum Heart Rate")

# Barplot: Heart Disease count (top-right)
target_counts = df["target"].value_counts().sort_index()
sns.barplot(x=target_counts.index, y=target_counts.values, palette="Set2", ax=axes[0, 1])
axes[0, 1].set_title("Heart Disease Count", fontsize=12)
axes[0, 1].set_xlabel("Heart Disease (0 = No, 1 = Yes)")
axes[0, 1].set_ylabel("Count")

# Heatmap for correlation (bottom-right)
sns.heatmap(correlation_matrix, annot=False, cmap="coolwarm", fmt=".2f",
            linewidths=0.5, ax=axes[1, 1])
axes[1, 1].set_title("Correlation Heatmap", fontsize=12)

plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-2074282334.py:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.boxplot(x="target", y="age", data=df, palette="Set3", ax=axes[0, 0])
/tmp/ipython-input-2074282334.py:13: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.violinplot(x="target", y="thalach", data=df, palette="Pastel1", ax=axes[1, 0])
/tmp/ipython-input-2074282334.py:20: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.barplot(x=target_counts.index, y=target_counts.values, palette="Set2", ax=axes[0, 1])
```




```
# Step 5: Key Insights
insights = {
    "Overall Trend": "The heart disease dataset shows clear patterns between clinical features and the presence of heart disease (target).",
    "Pairplot": "The pairplot reveals that higher values of oldpeak and lower maximum heart rate (thalach) are more common among patients with
    "Age Distribution": "Patients with heart disease tend to be in the middle to older age groups, although some younger patients are also aff
    "Thalach Distribution": "Patients without heart disease generally achieve higher maximum heart rates compared to those with heart disease
    "Correlation Insights": "Features such as oldpeak, chest pain type (cp), thalach, and ST depression show noticeable correlation with the t
}

for key, value in insights.items():
    print(f"{key}: {value}\n")
```

Overall Trend: The heart disease dataset shows clear patterns between clinical features and the presence of heart disease (target).

Pairplot: The pairplot reveals that higher values of oldpeak and lower maximum heart rate (thalach) are more common among patients with heart d

Age Distribution: Patients with heart disease tend to be in the middle to older age groups, although some younger patients are also affected.

Thalach Distribution: Patients without heart disease generally achieve higher maximum heart rates compared to those with heart disease.

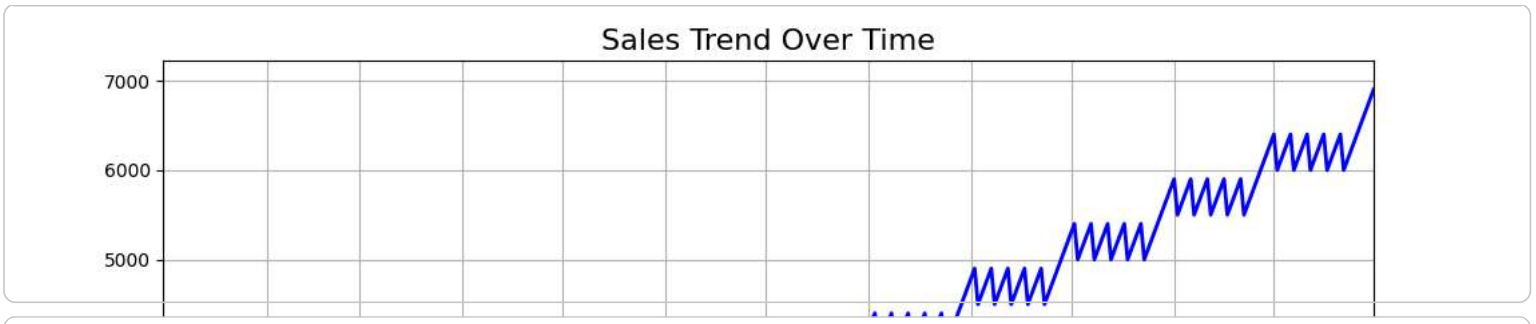
Correlation Insights: Features such as oldpeak, chest pain type (cp), thalach, and ST depression show noticeable correlation with the target va

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Step 1: Load and inspect the dataset
# Fix: Ensure Region list matches correct length
data = {
    'Date': pd.date_range(start='2023-01-01', periods=365, freq='D'),
    'Product Category': ['Electronics', 'Clothing', 'Furniture', 'Toys', 'Books'] * 73,
    'Sales': [500 + (i % 5) * 100 + (i // 30) * 500 for i in range(365)],
    'Region': ['North', 'South', 'East', 'West'] * 91 + ['North']  # TOTAL = 364 + 1 = 365
}

df = pd.DataFrame(data)
```

```
# Step 2: Clean and prepare the data
df['Month'] = df['Date'].dt.to_period('M')
df['Year'] = df['Date'].dt.year

print(df.head())
print("\nMissing Values:\n", df.isnull().sum())
```

```
        Date Product Category  Sales Region    Month  Year
0 2023-01-01      Electronics    500  North  2023-01  2023
1 2023-01-02         Clothing    600  South  2023-01  2023
2 2023-01-03        Furniture    700   East  2023-01  2023
3 2023-01-04             Toys    800   West  2023-01  2023
4 2023-01-05            Books    900  North  2023-01  2023

Missing Values:
 Date                0
Product Category    0
Sales               0
Region              0
Month               0
Year                0
dtype: int64
```

```
# 3a. Overall Sales Trend (Line Plot)
plt.figure(figsize=(10, 6))
df.groupby('Date')['Sales'].sum().plot(kind='line', color='b', linewidth=2)
plt.title("Sales Trend Over Time", fontsize=16)
plt.xlabel("Date")
plt.ylabel("Total Sales")
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```




```
# 3b. Sales by Region (Bar Plot)
plt.figure(figsize=(10, 6))
region_sales = df.groupby('Region')['Sales'].sum().sort_values(ascending=False)
sns.barplot(x=region_sales.index, y=region_sales.values, palette='Set2')
plt.title("Sales by Region", fontsize=16)
plt.xlabel("Region")
plt.ylabel("Total Sales")
plt.tight_layout()
plt.show()
```

```
/tmp/ipython-input-3447084752.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False`

  sns.barplot(x=region_sales.index, y=region_sales.values, palette='Set2')
```