Heart Disease UCI – Exploratory Data Analysis (EDA)

MLOps Experimental Learning Assignment

This notebook covers:

- Dataset acquisition
- Data cleaning & preprocessing
- Exploratory Data Analysis with professional visualizations
- EDA checklist aligned with production ML requirements

In [55]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

sns.set_theme(style="whitegrid")
```

1. Data Acquisition

Dataset source: UCI Machine Learning Repository – Heart Disease Dataset

In [56]:
```
df = pd.read_csv("../data/heart.csv")
df.head()
```

Out[56]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	63.0	1.0	1.0	145.0	233.0	1.0	2.0	150.0	0.0	2.3	3.0	0.0	6.0
1	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3.0	3.0
2	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2.0	7.0
3	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0.0	3.0
4	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0.0	3.0

2. Basic Dataset Inspection

In [57]:
```
df.shape
```

Out[57]:
```
(303, 14)
```

In [58]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       303 non-null    float64
 1   sex       303 non-null    float64
 2   cp        303 non-null    float64
 3   trestbps  303 non-null    float64
 4   chol      303 non-null    float64
 5   fbs       303 non-null    float64
 6   restecg   303 non-null    float64
 7   thalach   303 non-null    float64
 8   exang     303 non-null    float64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    float64
 11  ca        303 non-null    object 
 12  thal      303 non-null    object 
 13  target    303 non-null    int64  
dtypes: float64(11), int64(1), object(2)
memory usage: 33.3+ KB
```

In [59]:
```
df.describe()
```

Out[59]:

	age	sex	cp	trestbps	chol	fbs	restecg
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.438944	0.679868	3.158416	131.689769	246.693069	0.148515	0.990099
std	9.038662	0.467299	0.960126	17.599748	51.776918	0.356198	0.994971
min	29.000000	0.000000	1.000000	94.000000	126.000000	0.000000	0.000000
25%	48.000000	0.000000	3.000000	120.000000	211.000000	0.000000	0.000000
50%	56.000000	1.000000	3.000000	130.000000	241.000000	0.000000	1.000000
75%	61.000000	1.000000	4.000000	140.000000	275.000000	0.000000	2.000000
max	77.000000	1.000000	4.000000	200.000000	564.000000	1.000000	2.000000

3. Data Quality Checks

In [60]:
```
df.isnull().sum()
```

Out[60]:
```
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
```

In [61]:
```
df.duplicated().sum()
```

Out[61]:
```
np.int64(0)
```

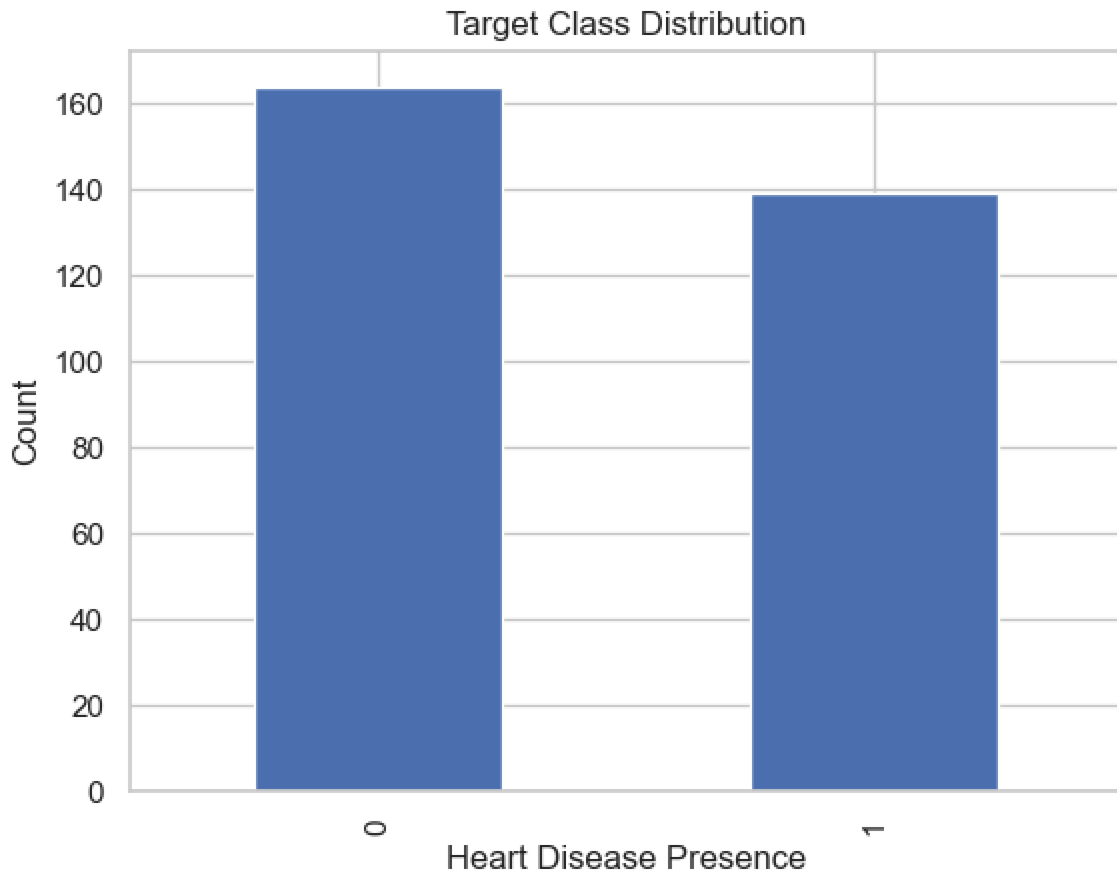
4. Target Variable Count

In [62]:
```
df["target"] = (df["target"] > 0).astype(int)
df["target"].value_counts()
```

Out[62]:
```
target
0    164
1    139
Name: count, dtype: int64
```

5. Target Variable Distribution

In [63]:
```
plt.figure()
df['target'].value_counts().plot(kind='bar')
plt.title("Target Class Distribution")
plt.xlabel("Heart Disease Presence")
plt.ylabel("Count")
plt.show()
```

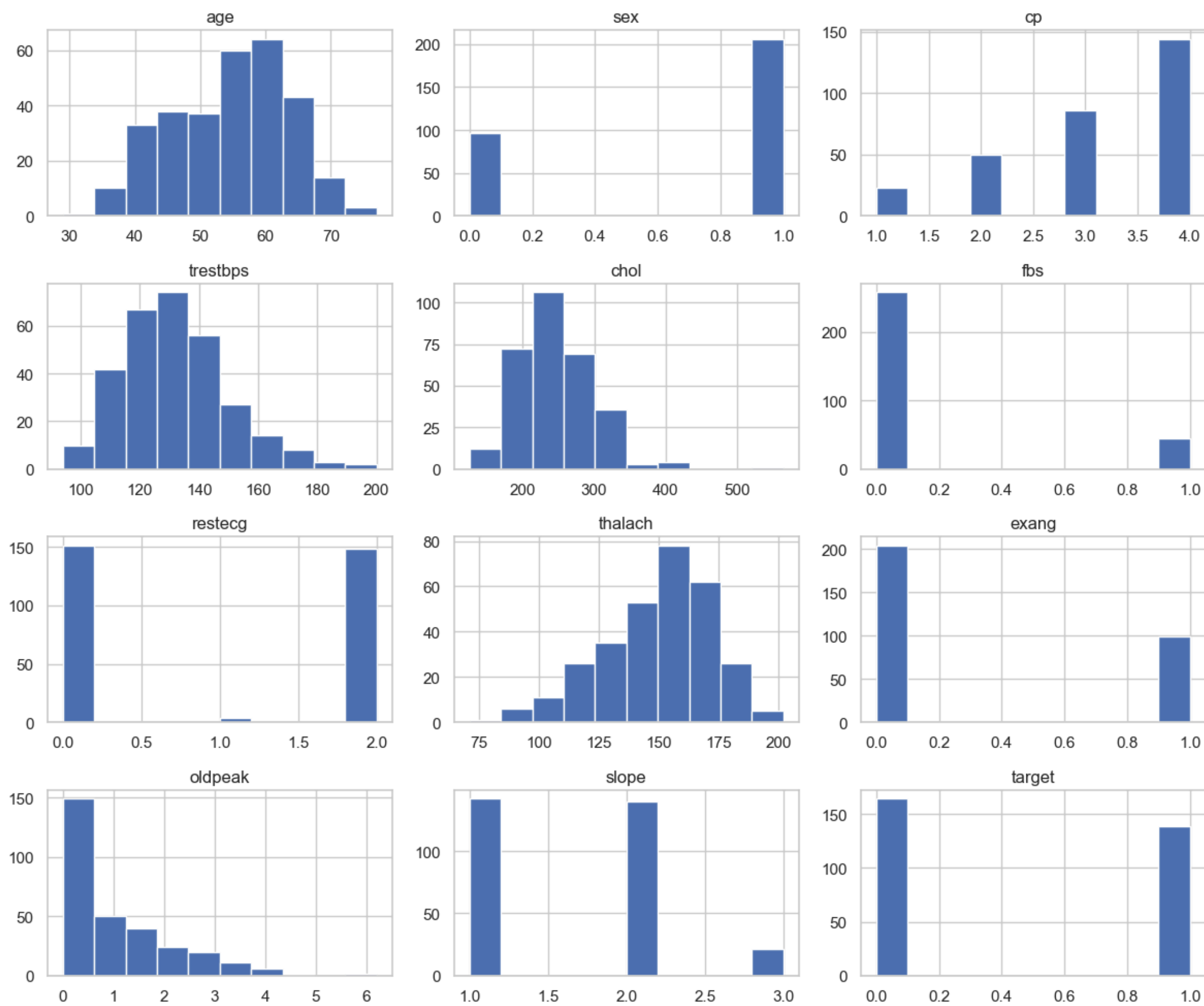


6. Class Balance Plot

In [64]:
```
sns.countplot(x="target", data=df)
plt.title("Target Class Distribution")
plt.show()
```

7. Numerical Feature Distributions

In [65]:
```
df.hist(figsize=(12,10))
plt.tight_layout()
plt.show()
```



8. Data Cleaning for Correlation Analysis

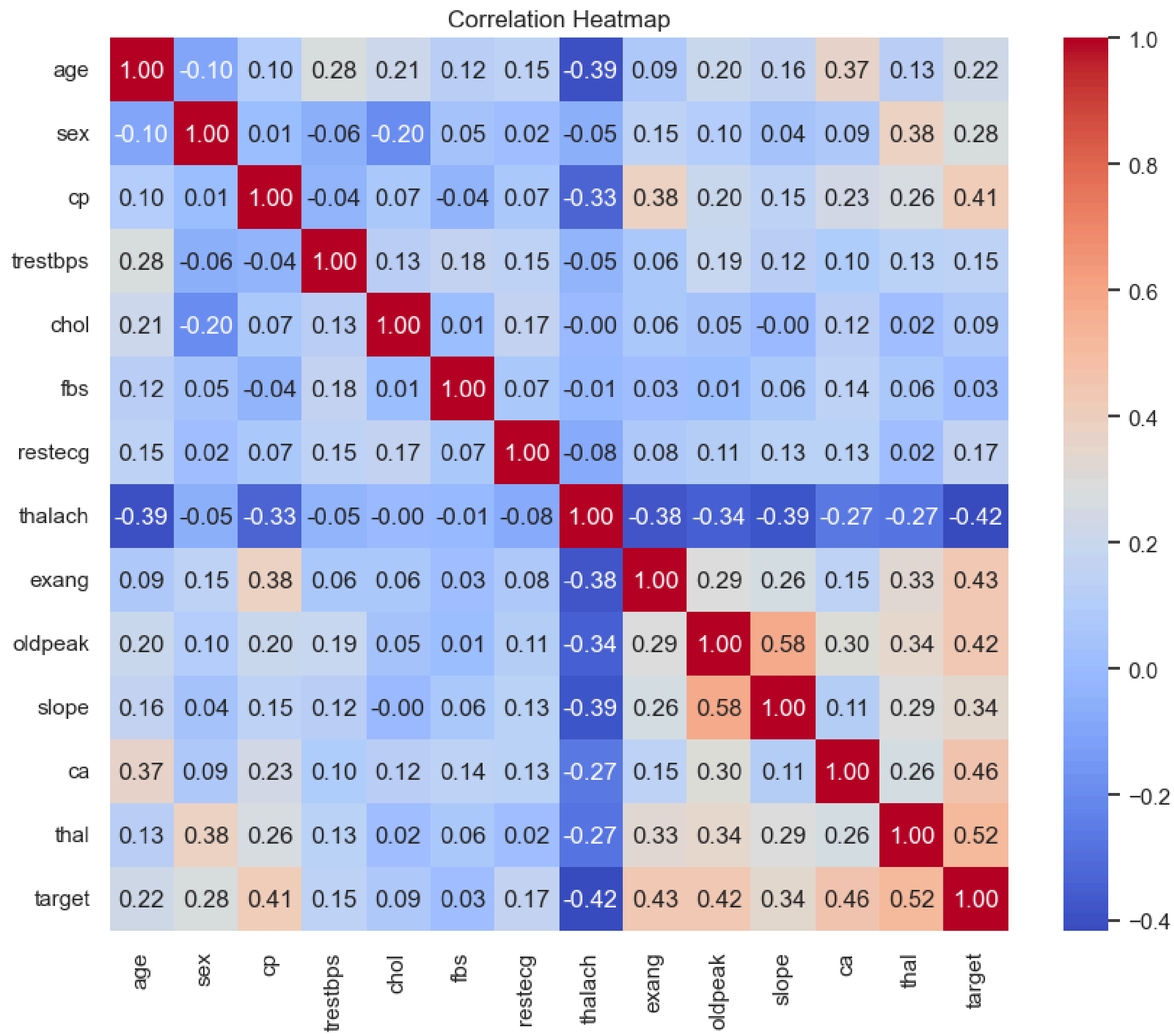
In [66]:
```python
# Replace invalid placeholders with NaN
df.replace('?', np.nan, inplace=True)

# Convert affected columns to numeric
df['ca'] = pd.to_numeric(df['ca'])
df['thal'] = pd.to_numeric(df['thal'])

# Impute missing values using median
df['ca'] = df['ca'].fillna(df['ca'].median())
df['thal'] = df['thal'].fillna(df['thal'].median())
```

9. Correlation Analysis

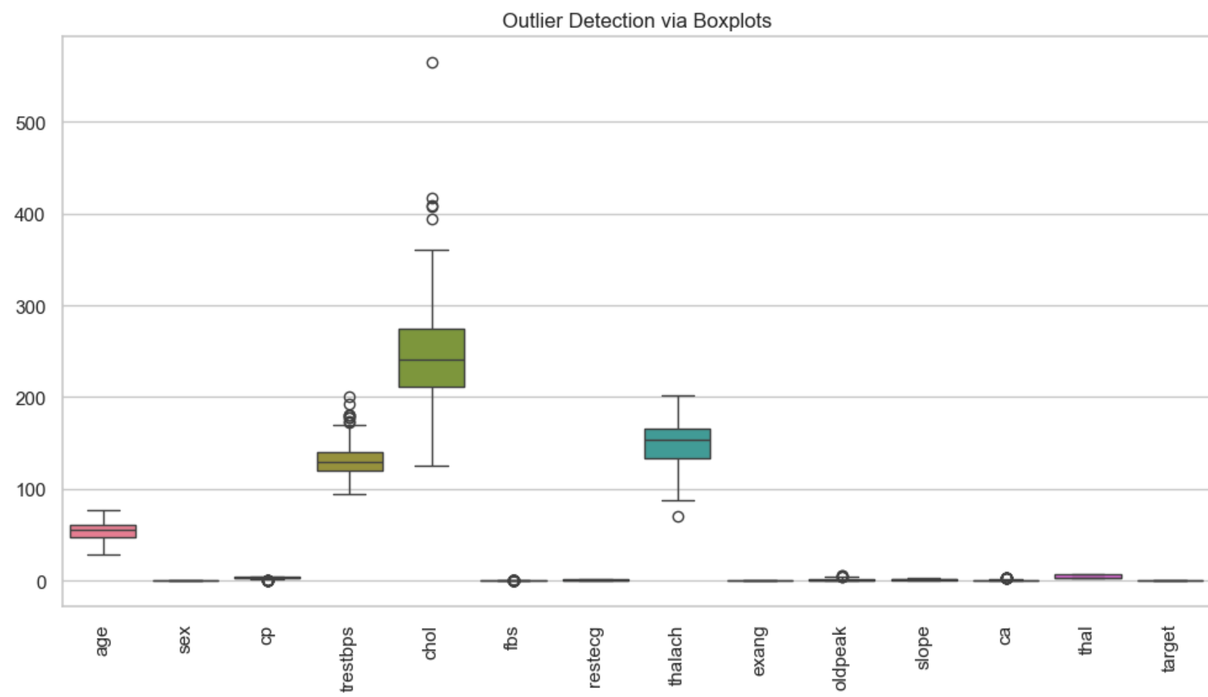
In [67]:
```python
plt.figure(figsize=(10,8))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```



10. Outlier Detection

In [68]:

```
plt.figure(figsize=(12,6))
sns.boxplot(data=df)
plt.xticks(rotation=90)
plt.title("Outlier Detection via Boxplots")
plt.show()
```



11. Key EDA Observations

- Dataset is clean with minimal missing values
- Target variable is fairly balanced
- Some features show moderate correlation with target
- Outliers are medically plausible