Heart Disease Diagnostic Analysis

import Libraries

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Extract the data

In [3]:
```
data=pd.read_csv('Heart Disease data.csv')
```

Display top 5 rows of data set

In [4]:
```
data.head()
```

Out[4]:

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

```
Attribute Information:

age
sex
```

```
chest pain type (4 values)
resting blood pressure
serum cholestoral in mg/dl
fasting blood sugar > 120 mg/dl
resting electrocardiographic results (values 0,1,2)
maximum heart rate achieved
exercise induced angina
oldpeak = ST depression induced by exercise relative to rest
the slope of the peak exercise ST segment
number of major vessels (0-3) colored by flourosopy
thal: 0 = normal; 1 = fixed defect; 2 = reversable defect
```

Last 5 rows of dataset

In [5]:
```
data.tail()
```

Out[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

Find the total no. of columns and rows

In [6]:
```
print("Number of rows",data.shape[0])
print("Number of columns",data.shape[1])
```

```
Number of rows 1025
Number of columns 14
```

All Information about data

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

Check null values in the dataset

In [11]:
```
data.isnull().sum()
```

Out[11]:
```
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
```

Check for Duplicate values

In [12]:
```
data_dup=data.duplicated().any()
print(data_dup)
```

```
True
```

In [14]:
```
data=data.drop_duplicates()
data.shape
```

Out[14]:
```
(302, 14)
```

get overall statistics about the data set

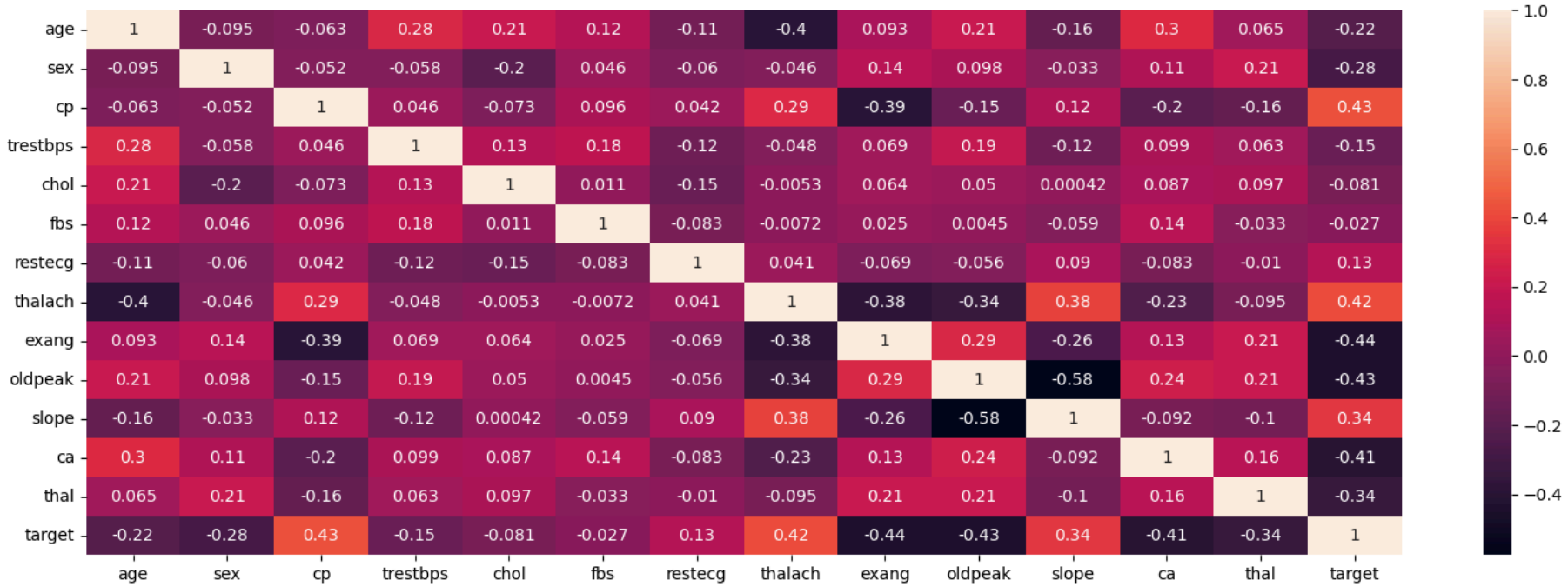
In [15]:
```
data.describe()
```

Out[15]:

ge	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
00	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000
53	0.682119	0.963576	131.602649	246.500000	0.149007	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543	2.314570
97	0.466426	1.032044	17.563394	51.753489	0.356686	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748	0.613026
00	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000
00	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.250000	0.000000	0.000000	1.000000	0.000000	2.000000
00	1.000000	1.000000	130.000000	240.500000	0.000000	1.000000	152.500000	0.000000	0.800000	1.000000	0.000000	2.000000
00	1.000000	2.000000	140.000000	274.750000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000
00	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000

In [24]:
```
plt.figure(figsize=(18,6))
sns.heatmap(data.corr(),annot=True)
```

Out[24]: <Axes: >



How many people have heart disease and how many don't

In [25]:
```
data.columns
```

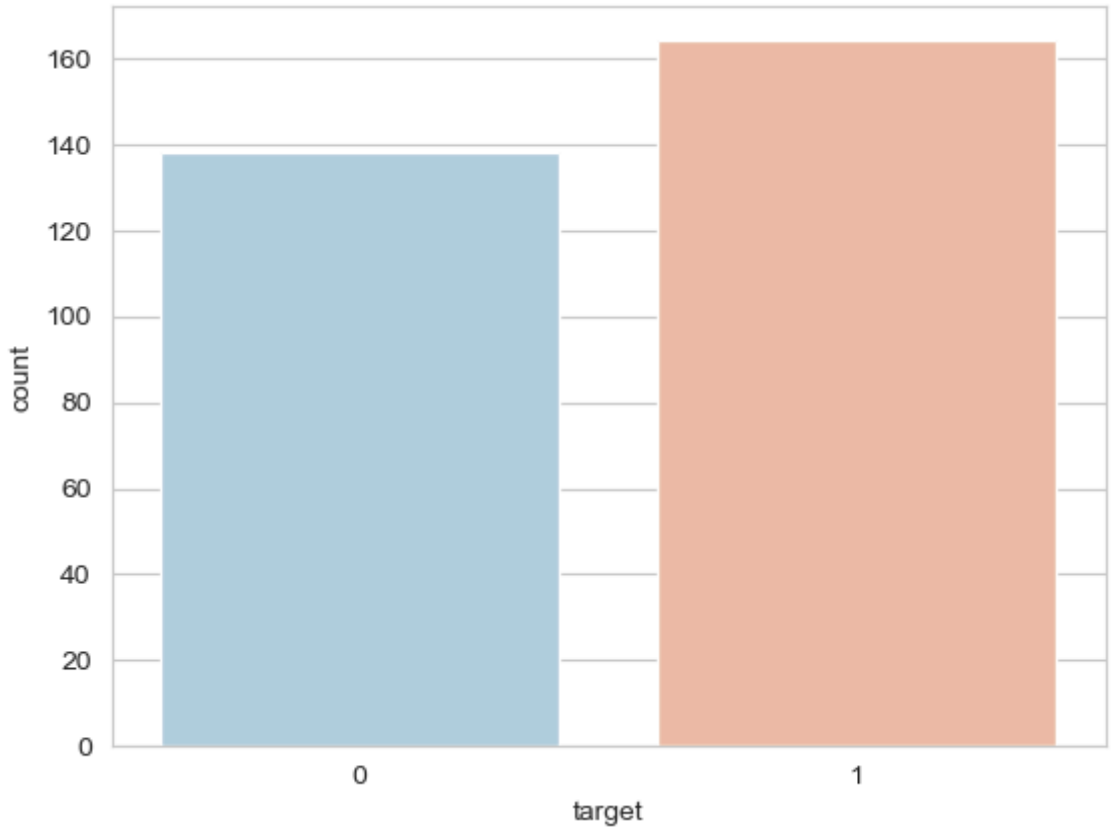
Out[25]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [26]:
```
data['target'].value_counts()
```

Out[26]:
```
target
1    164
0    138
Name: count, dtype: int64
```

In [30]:
```
sns.set_style('whitegrid')
sns.countplot(x='target',data=data,palette='RdBu_r')
```

Out[30]:
```
<Axes: xlabel='target', ylabel='count'>
```



In [31]:
```
data.columns
```

Out[31]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

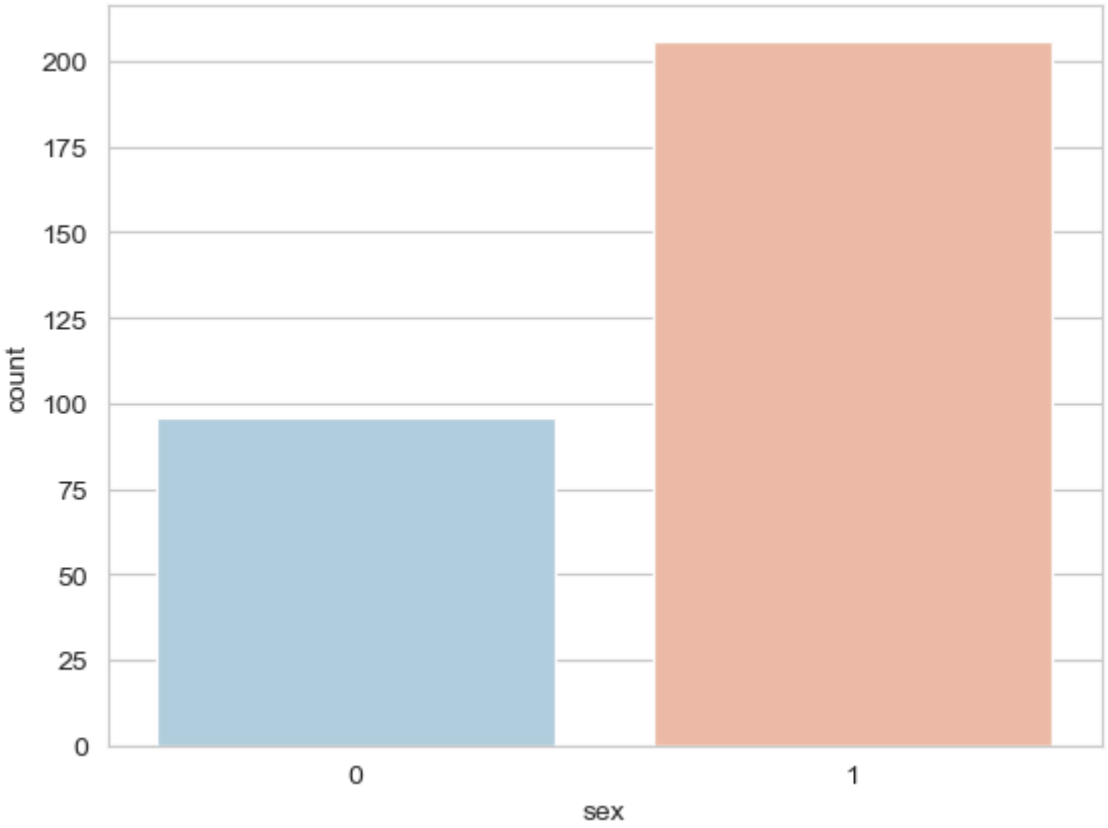
Find count of male and female in the dataset

In [32]:
```
data['sex'].value_counts()
```

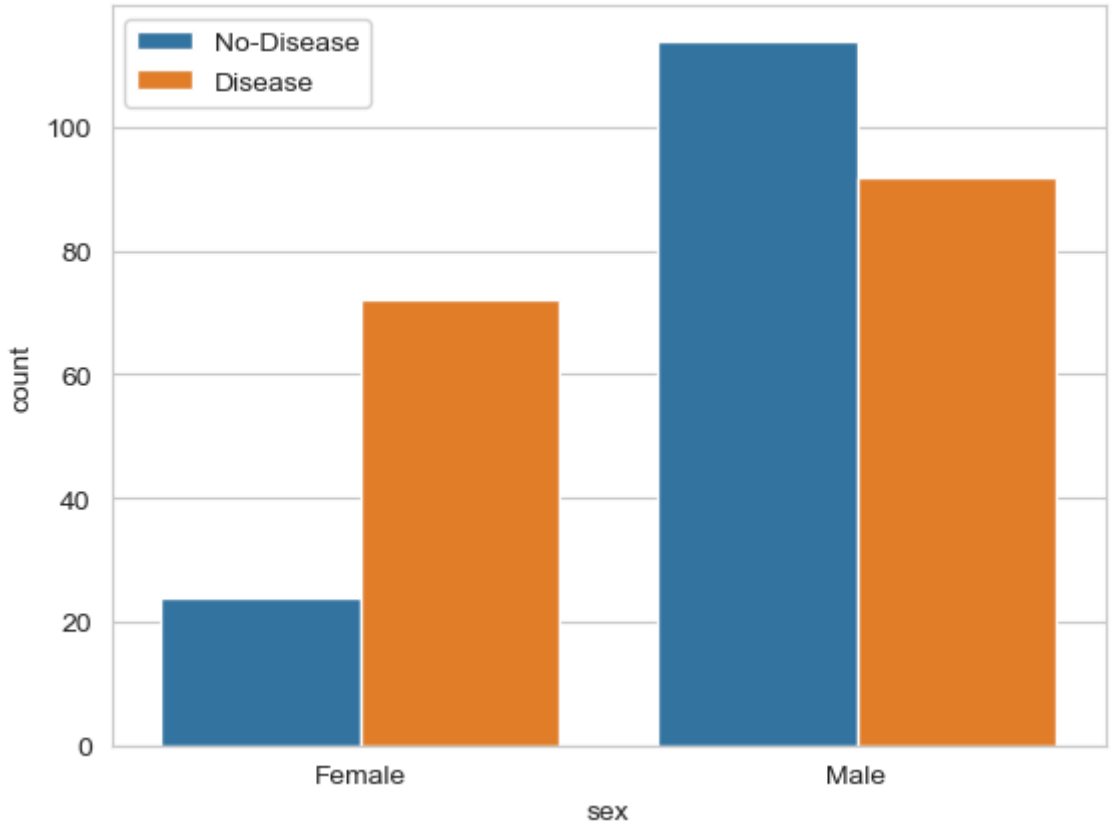
Out[32]:
```
sex
1    206
0     96
Name: count, dtype: int64
```

In [40]:
```
sns.set_style('whitegrid')
sns.countplot(x='sex',data=data,palette='RdBu_r')
```

Out[40]:
```
<Axes: xlabel='sex', ylabel='count'>
```

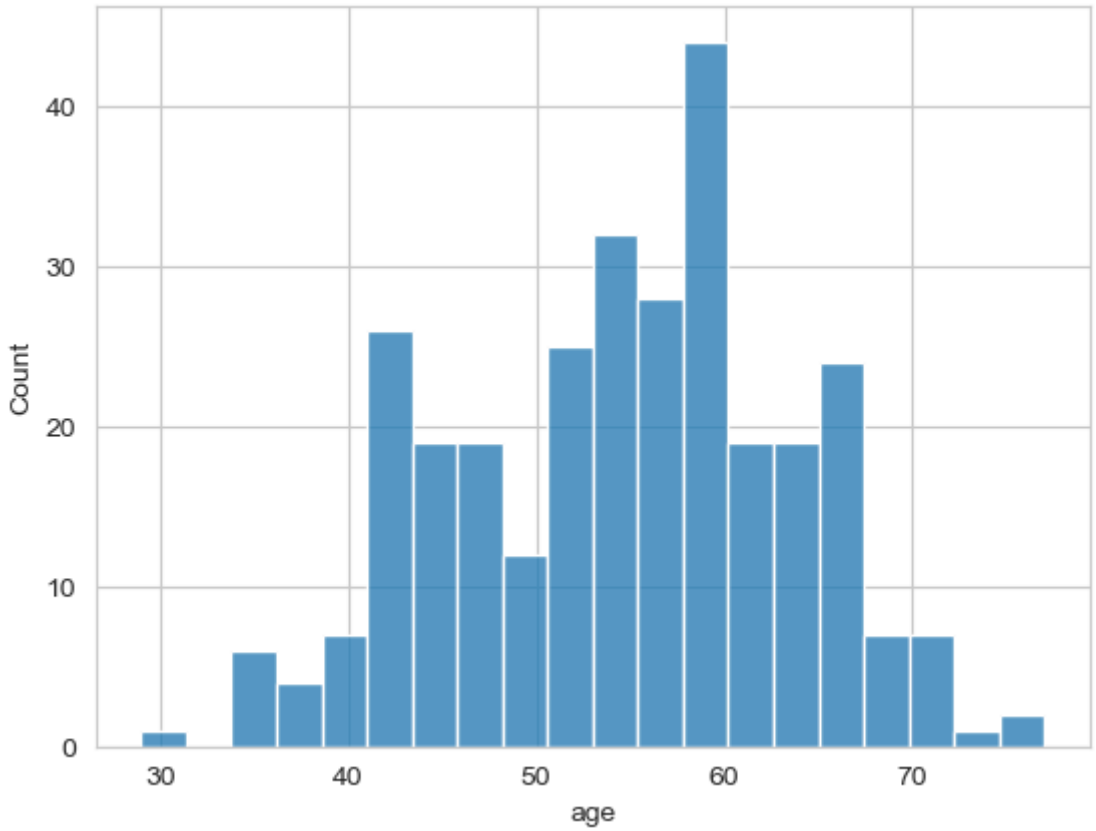


In [63]:
```
sns.countplot(data=data, x='sex', hue="target")

plt.xticks([1,0],['Male','Female'])
plt.legend(labels=["No-Disease", "Disease"])
plt.show()
```



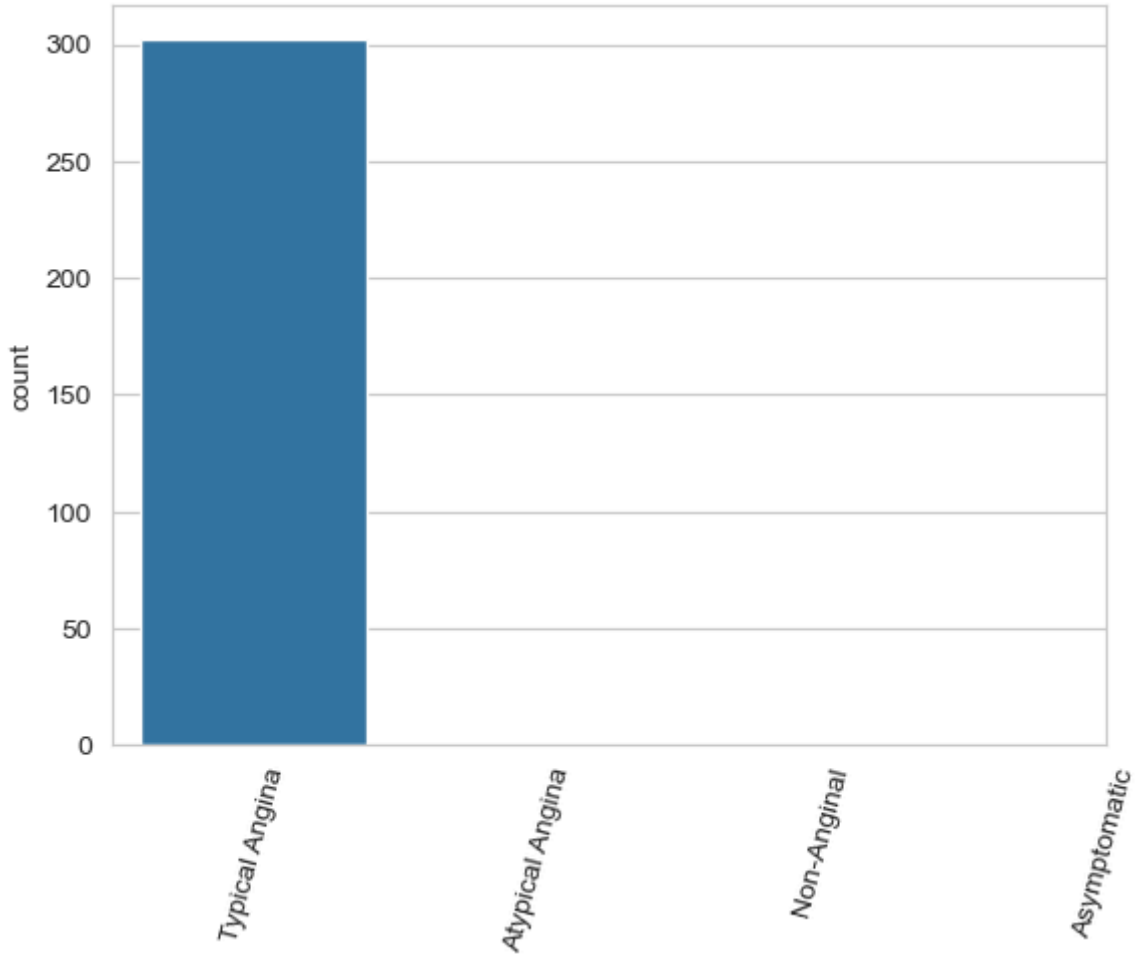
Check age distribution in the dataset

In [45]:
```
sns.histplot(data['age'],bins=20)
plt.show()
```



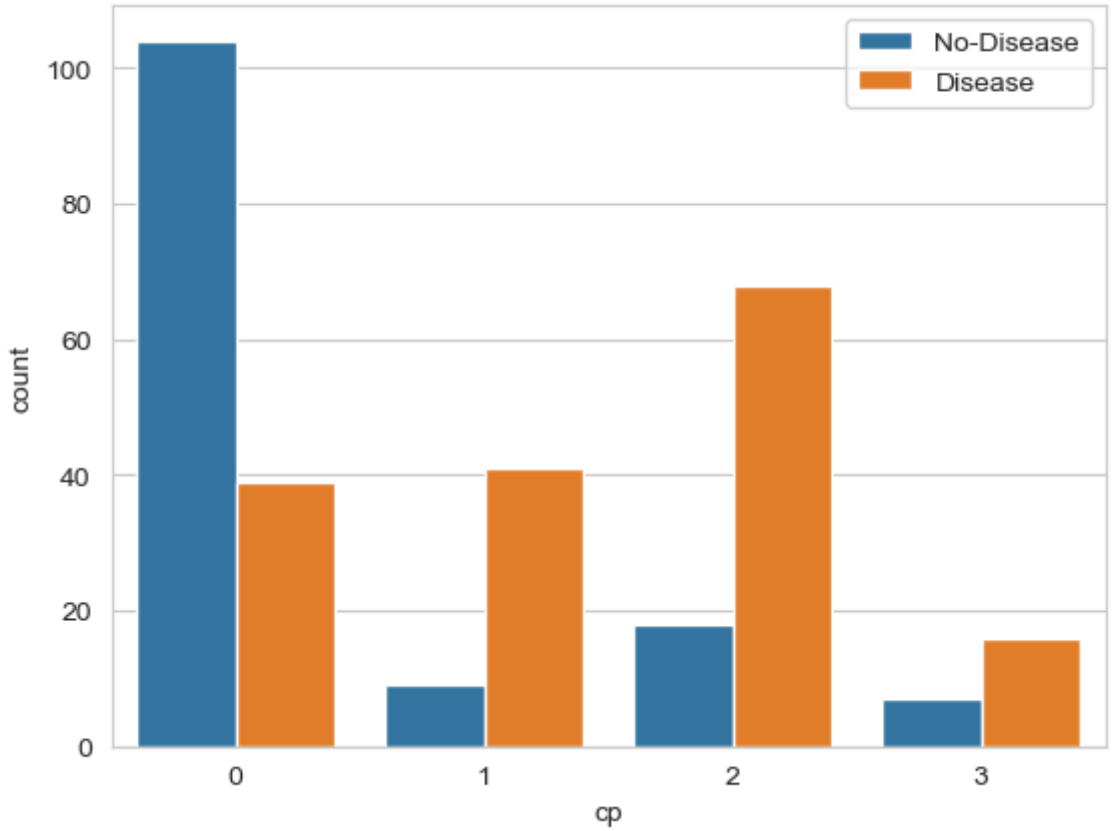
Chest Pain

In [64]:
```
# Create a count plot for 'cp'
sns.countplot(data['cp'])

# Customize x-axis labels
plt.xticks([0, 1, 2, 3], ["Typical Angina", "Atypical Angina", "Non-Anginal", "Asymptomatic"], rotation=75)

# Display the plot
plt.show()
```



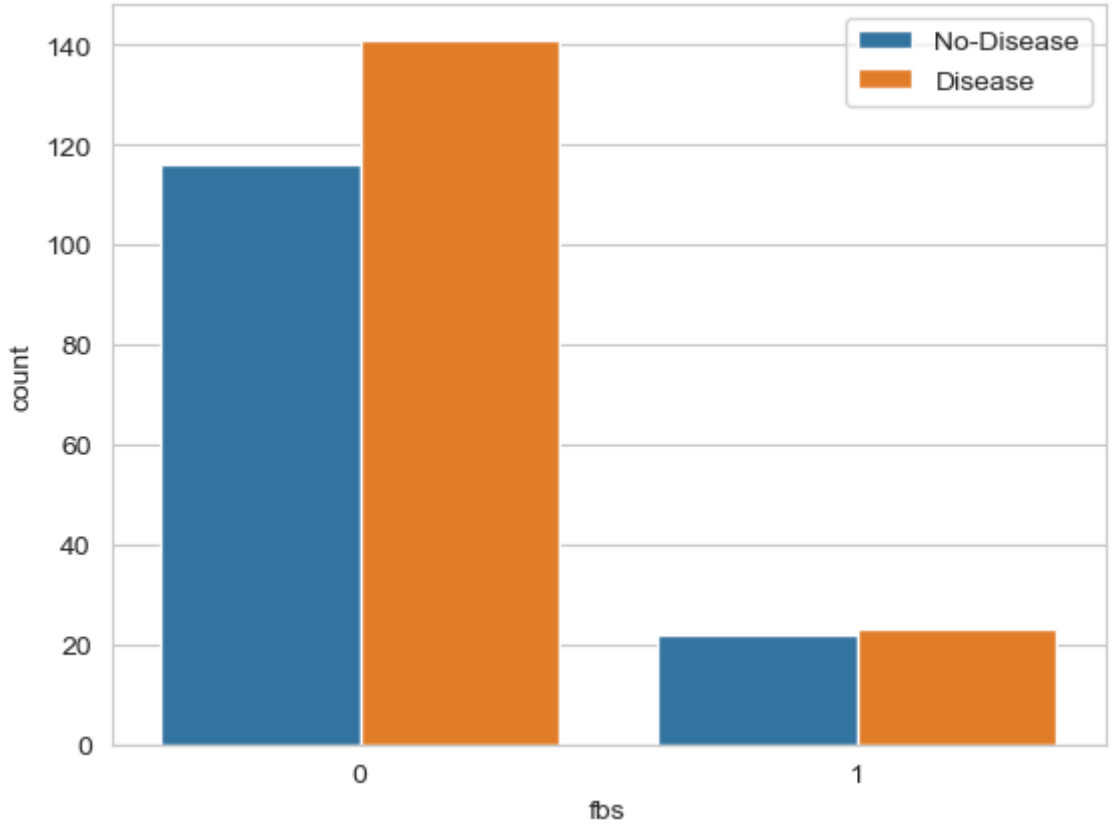
Show the chest pain distribution as per target variable

In [61]:
```
sns.countplot(x="cp",hue="target",data=data)
plt.legend(labels=["No-Disease", "Disease"])
plt.show()
```



Fasting Blood sugar according to the target variable

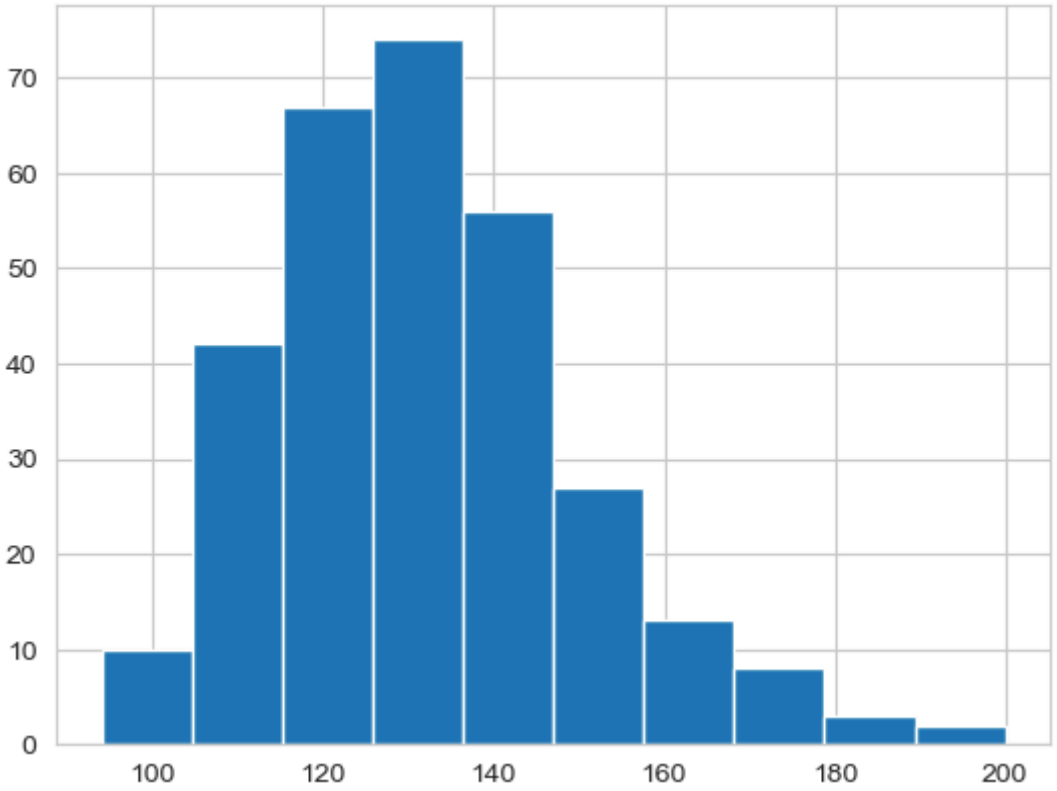
In [62]:
```
sns.countplot(x="fbs",hue="target",data=data)
plt.legend(labels=["No-Disease", "Disease"])

plt.show()
```



Check Resting Blood Pressure

In [65]:
```
data['trestbps'].hist()
```

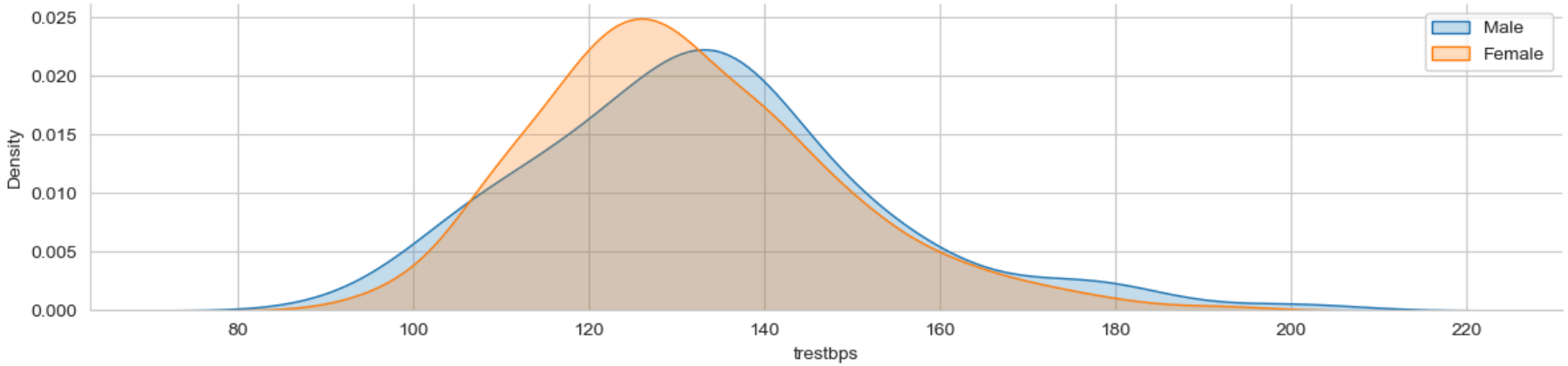
Out[65]: <Axes: >



Compare Resting Blood Pressure As Per Sex Column

In [71]:
```
g = sns.FacetGrid(data, hue="sex", aspect=4)
g.map(sns.kdeplot, 'trestbps', fill=True)
plt.legend(labels=['Male', 'Female'])  # Corrected argument name
plt.tight_layout()  # Adjust figure layout
plt.show()
```

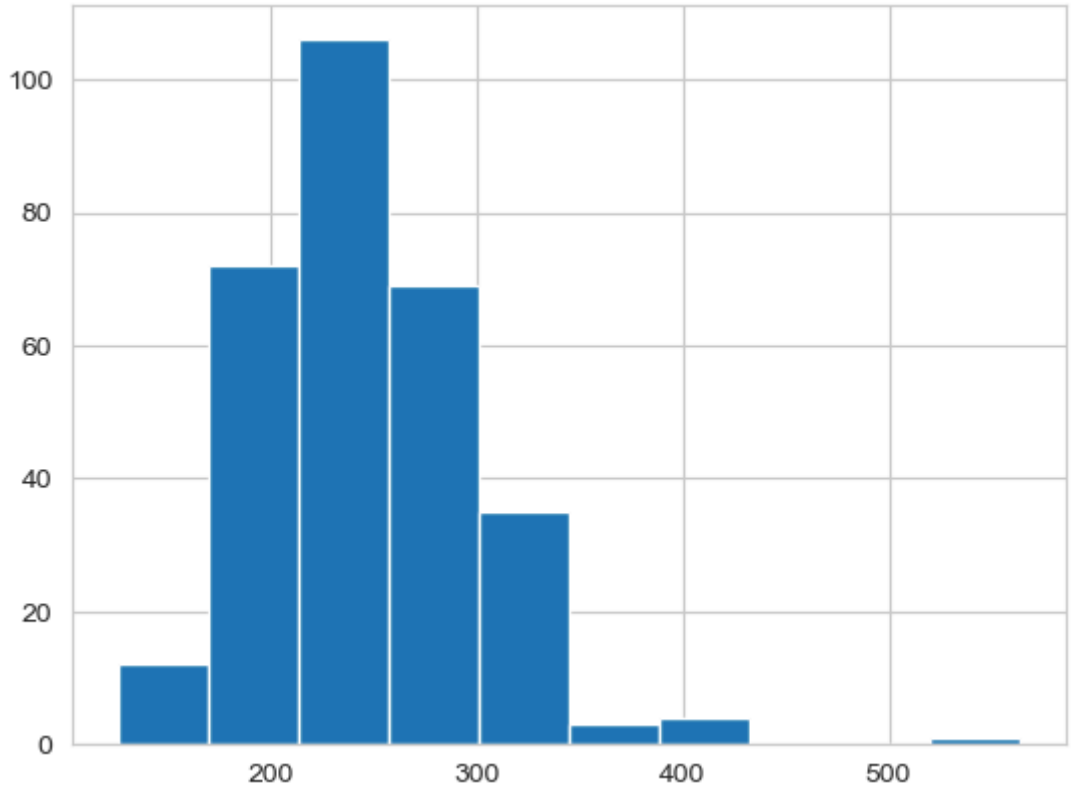
```
C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tig
ht
  self._figure.tight_layout(*args, **kwargs)
C:\Users\ARVIND KAPASIYA\AppData\Local\Temp\ipykernel_21716\613643646.py:4: UserWarning: The figure layout has change
d to tight
  plt.tight_layout()  # Adjust figure layout
```



Show distribution of serum cholestrol

In [72]:
```
data['chol'].hist()
```

Out[72]: <Axes: >



Plot continuous Variable

In [79]:
```
cate_val=[]
cont_val=[]

for column in data.columns:
    if data [column].nunique() <=10:
        cate_val.append(column)
    else:
        cont_val.append(column)
```

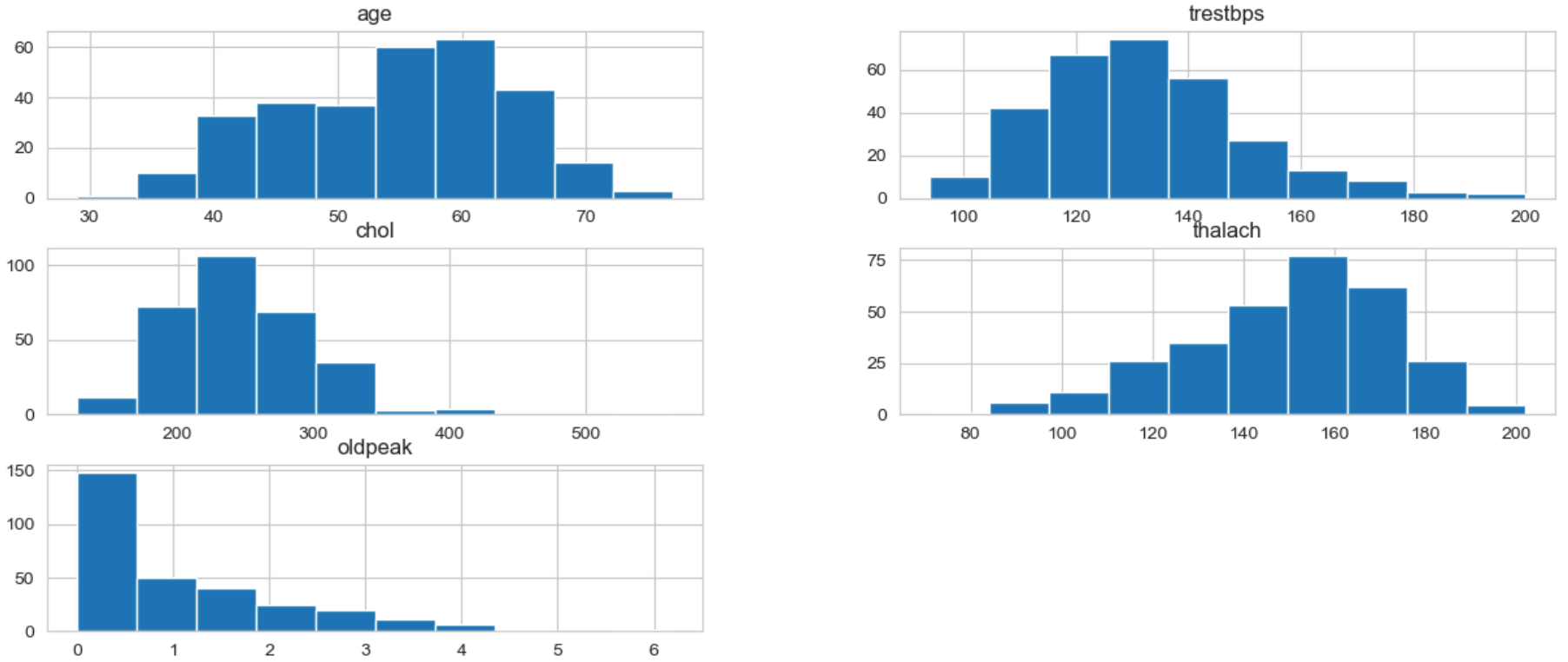
In [80]:
```
cate_val
```

Out[80]:
```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

In [81]:
```
cont_val
```

Out[81]:
```
['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
```

In [83]:
```
data.hist(cont_val,figsize=(15,6))
plt.show()
```



In [ ]: