In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as st
%matplotlib inline

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

In [3]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [4]:
```
df=pd.read_csv(r'C:\Users\hp\OneDrive\Documents\Desktop\heart.csv')
df
```

Out[4]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2

303 rows × 14 columns

In [5]:
```
df.shape
```

Out[5]:
```
(303, 14)
```

In [6]:
```
df.head()
```

Out[6]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	ta
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	

In [7]:
```
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    int64  
 1   sex       303 non-null    int64  
 2   cp        303 non-null    int64  
 3   trestbps  303 non-null    int64  
 4   chol      303 non-null    int64  
 5   fbs       303 non-null    int64  
 6   restecg   303 non-null    int64  
 7   thalach   303 non-null    int64  
 8   exang     303 non-null    int64  
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64  
 11  ca        303 non-null    int64  
 12  thal      303 non-null    int64  
 13  target    303 non-null    int64  
dtypes: float64(1), int64(13)
```

In [8]:
```
df.dtypes
```

Out[8]:
```
age           int64
sex           int64
cp            int64
trestbps      int64
chol          int64
fbs           int64
restecg       int64
thalach       int64
exang         int64
oldpeak     float64
slope         int64
ca            int64
thal          int64
target        int64
dtype: object
```

In [9]:
```
df.describe()
```

Out[9]:

	age	sex	cp	trestbps	chol	fbs	restecg
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000

In [10]:
```
df.columns
```

Out[10]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalac
h',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [11]:
```
df['target'].nunique()
```

Out[11]:
```
2
```

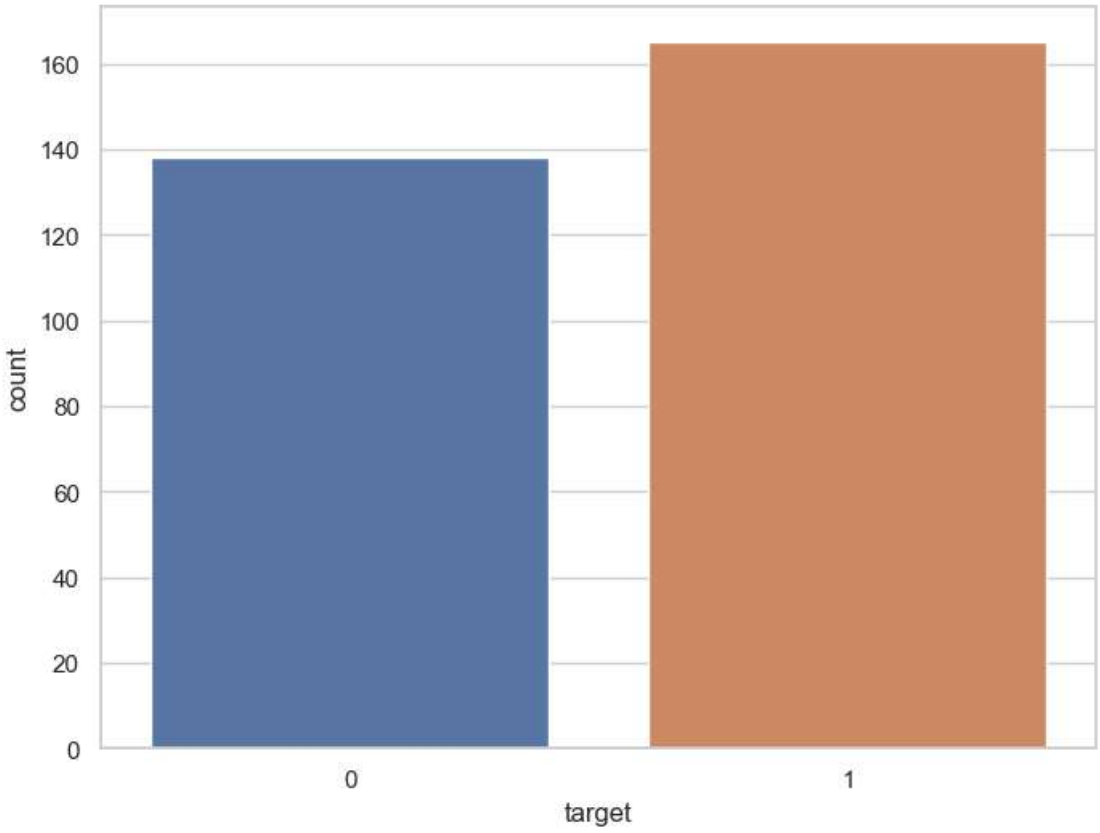
In [12]:
```
df["target"].unique()
```

Out[12]:
```
array([1, 0], dtype=int64)
```

In [13]:
```
df["target"].value_counts()
```

Out[13]:
```
1    165
0    138
Name: target, dtype: int64
```

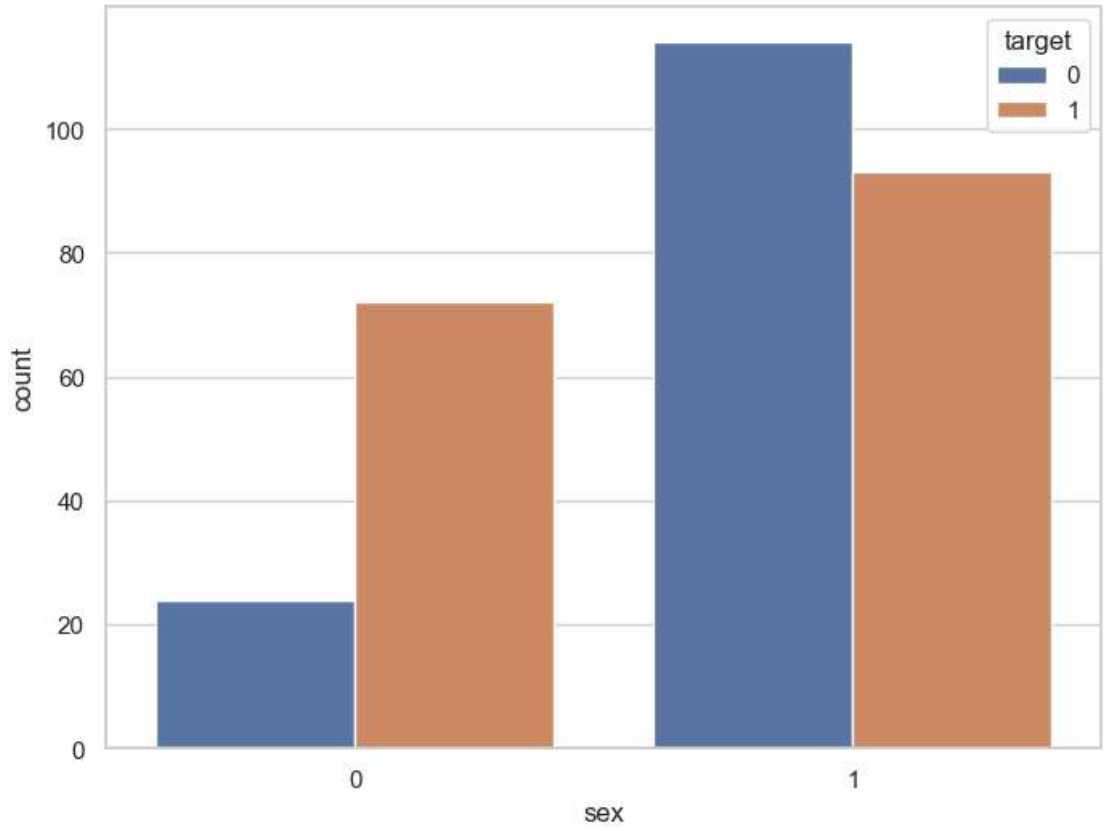
In [14]:
```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(x="target", data=df)
plt.show()
```



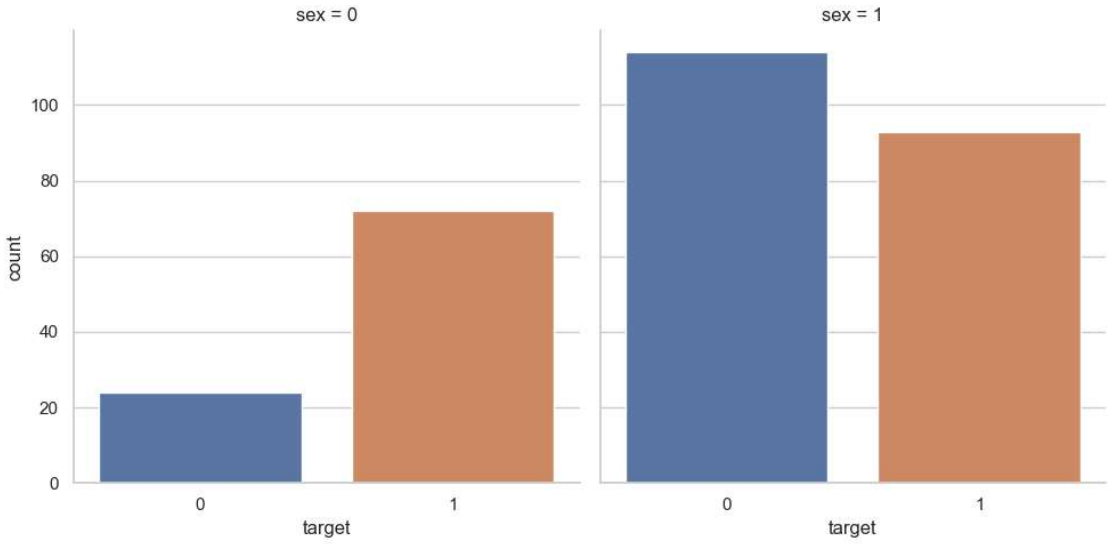
In [15]:
```
df.groupby("sex")["target"].value_counts()
```

Out[15]:
```
sex  target
0    1          72
     0          24
1    0         114
     1          93
Name: target, dtype: int64
```

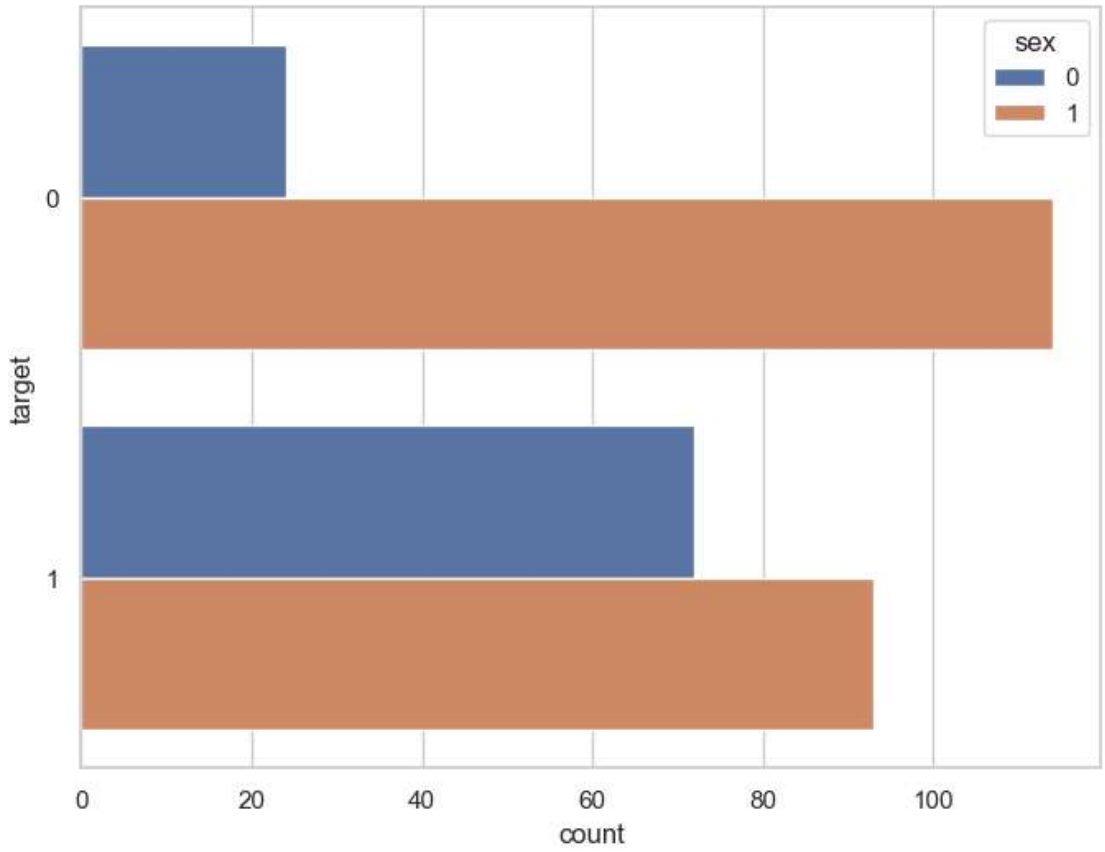
In [16]:
```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(x="sex", hue="target", data=df)
plt.show()
```



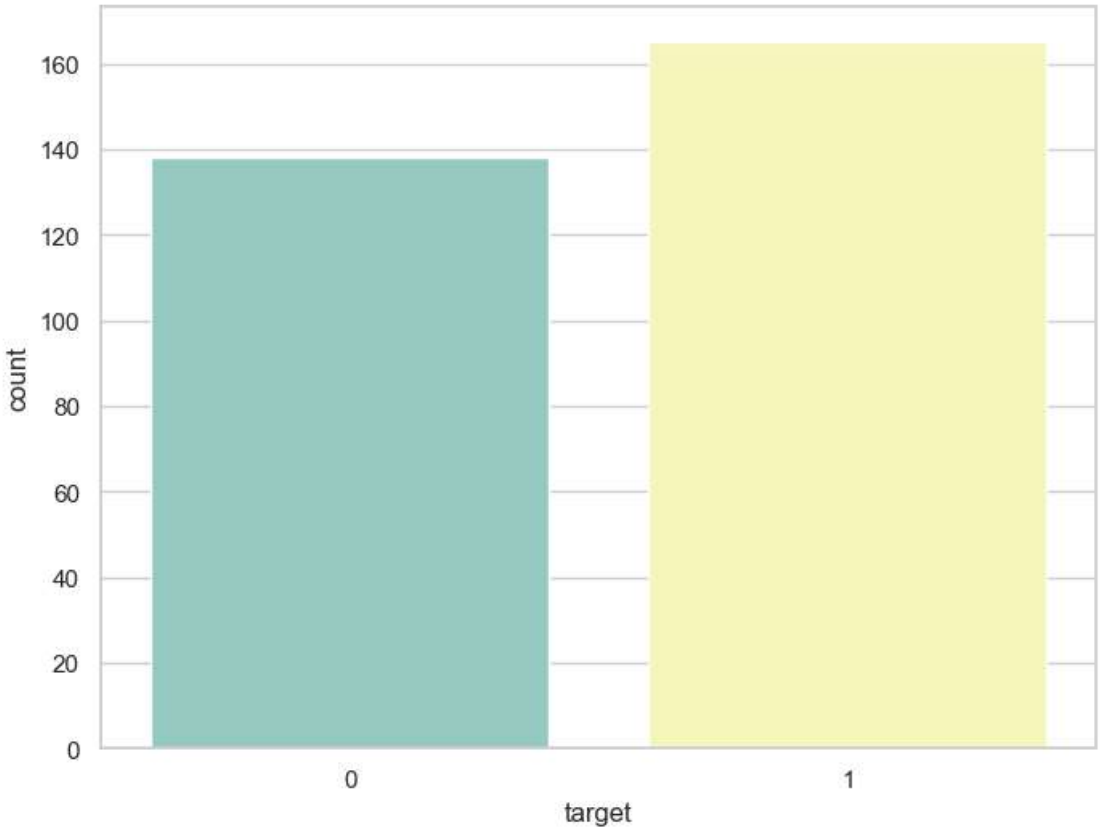
In [17]:
```
ax = sns.catplot(x="target", col="sex", data=df, kind="count", height=5, a
```



In [18]:
```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.countplot(y="target", hue="sex", data=df)
plt.show()
```

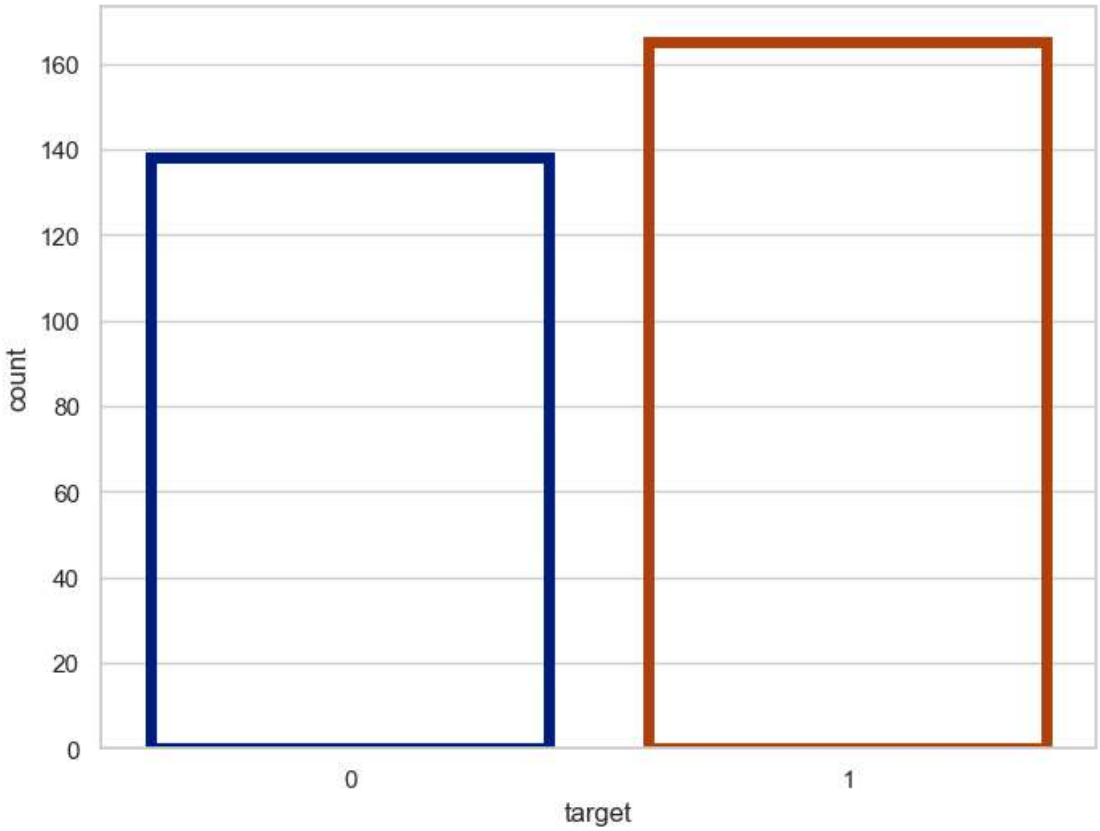


In [19]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="target", data=df, palette="Set3")
plt.show()
```

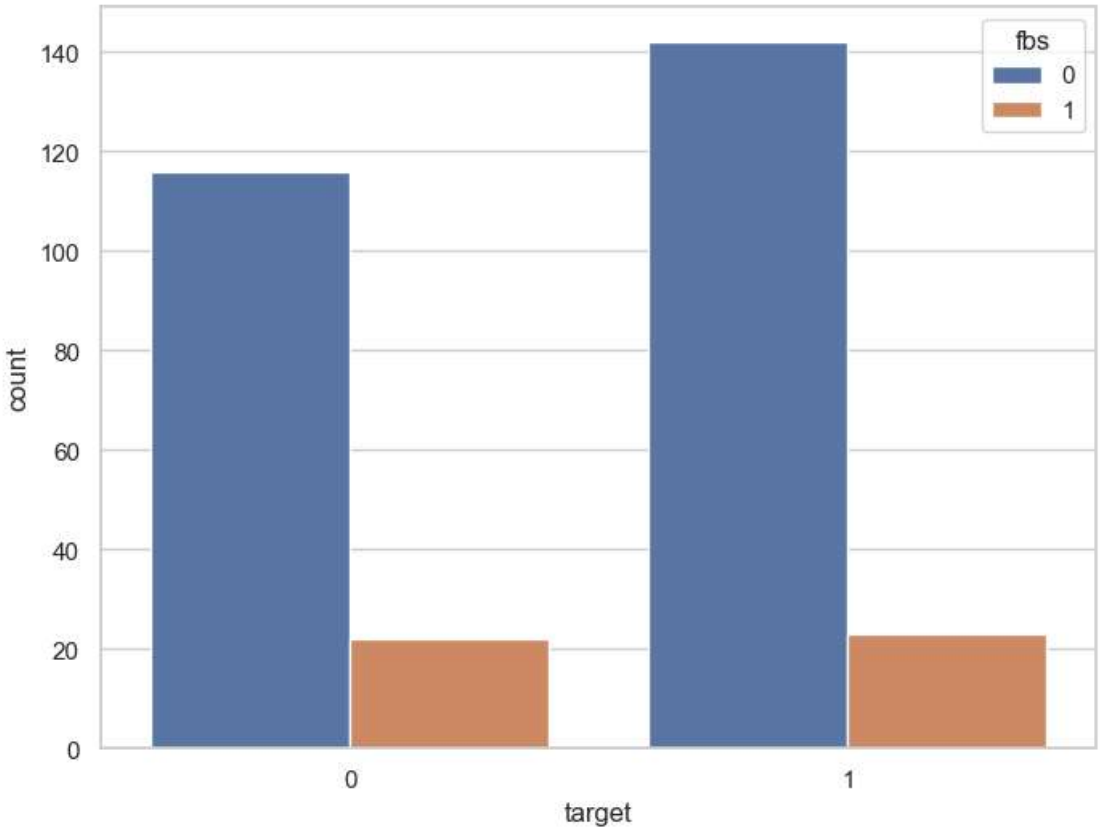


we can use plt.bar keywords arguments for a different looks:

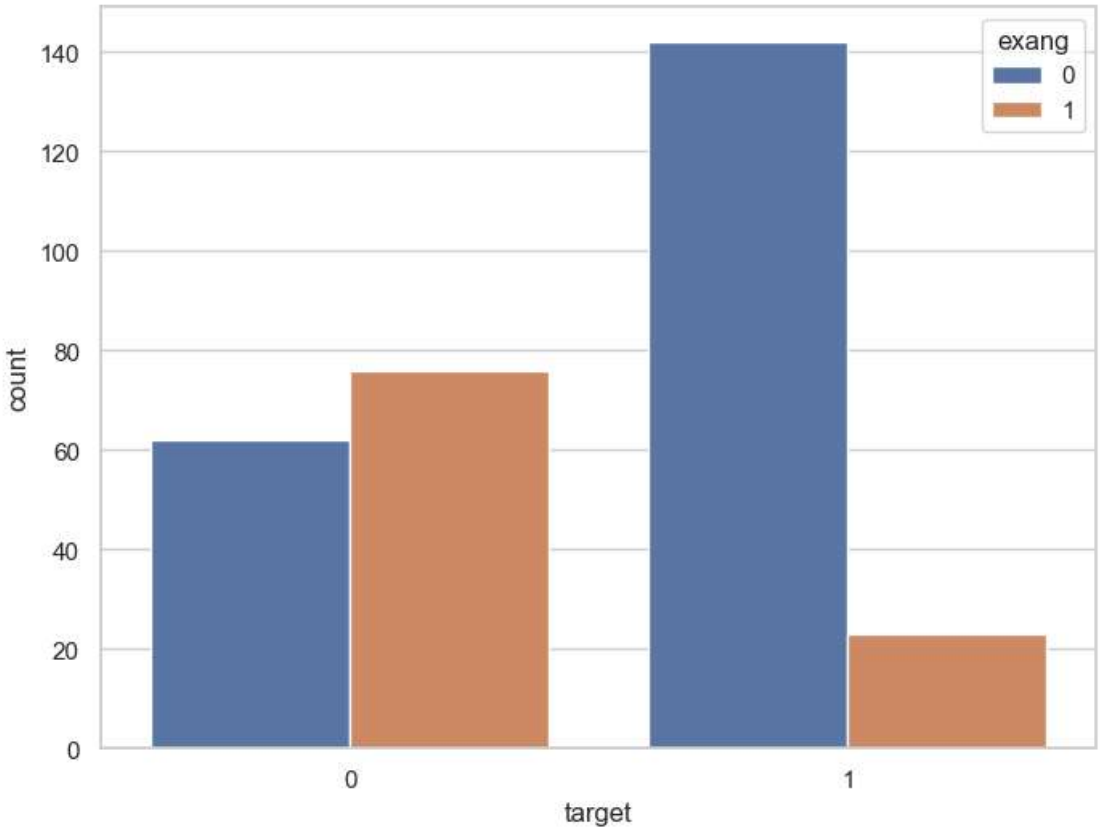
In [20]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="target", data=df, facecolor=(0,0,0,0),linewidth=5, e
plt.show()
```



In [21]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="target", hue="fbs", data=df)
plt.show()
```



In [22]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="target",hue="exang", data=df)
plt.show()
```



Biveriate Analysis

In [23]:
```
correlation = df.corr()
```

In [24]:
```
correlation["target"].sort_values(ascending=False)
```

Out[24]:
```
target      1.000000
cp          0.433798
thalach     0.421741
slope       0.345877
restecg     0.137230
fbs        -0.028046
chol       -0.085239
trestbps   -0.144931
age        -0.225439
sex        -0.280937
thal       -0.344029
ca         -0.391724
oldpeak    -0.430696
exang      -0.436757
Name: target, dtype: float64
```

In [25]:
```
# explore cp variable

# cp stand for chest pain type
# first iwill check the number if unique values in cp  variabl
```

In [26]:
```
df["cp"].nunique()
```

Out[26]:
```
4
```

In [27]:
```
df["cp"].unique()
```

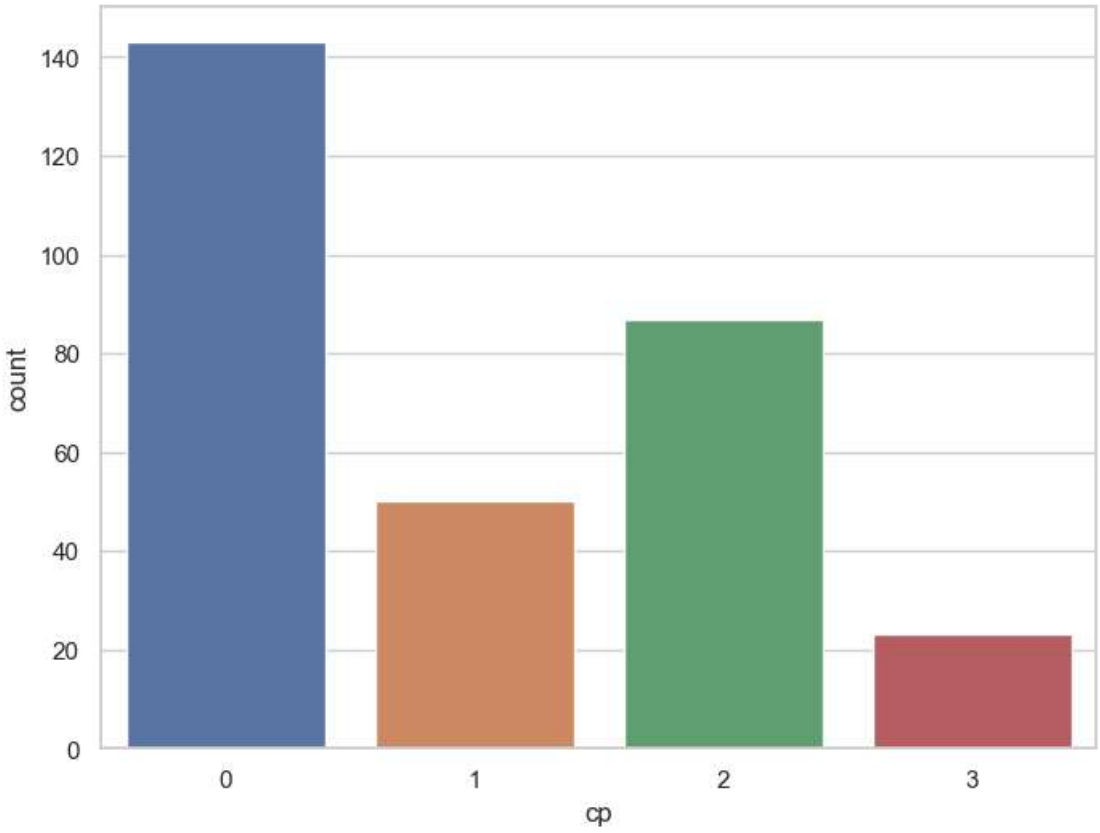
Out[27]:
```
array([3, 2, 1, 0], dtype=int64)
```

In [28]:
```
df["cp"].value_counts()
```

Out[28]:
```
0    143
2     87
1     50
3     23
Name: cp, dtype: int64
```

In [29]:
```
# visualize the frequency distribution of cp variable
```

In [30]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="cp", data=df)
plt.show()
```

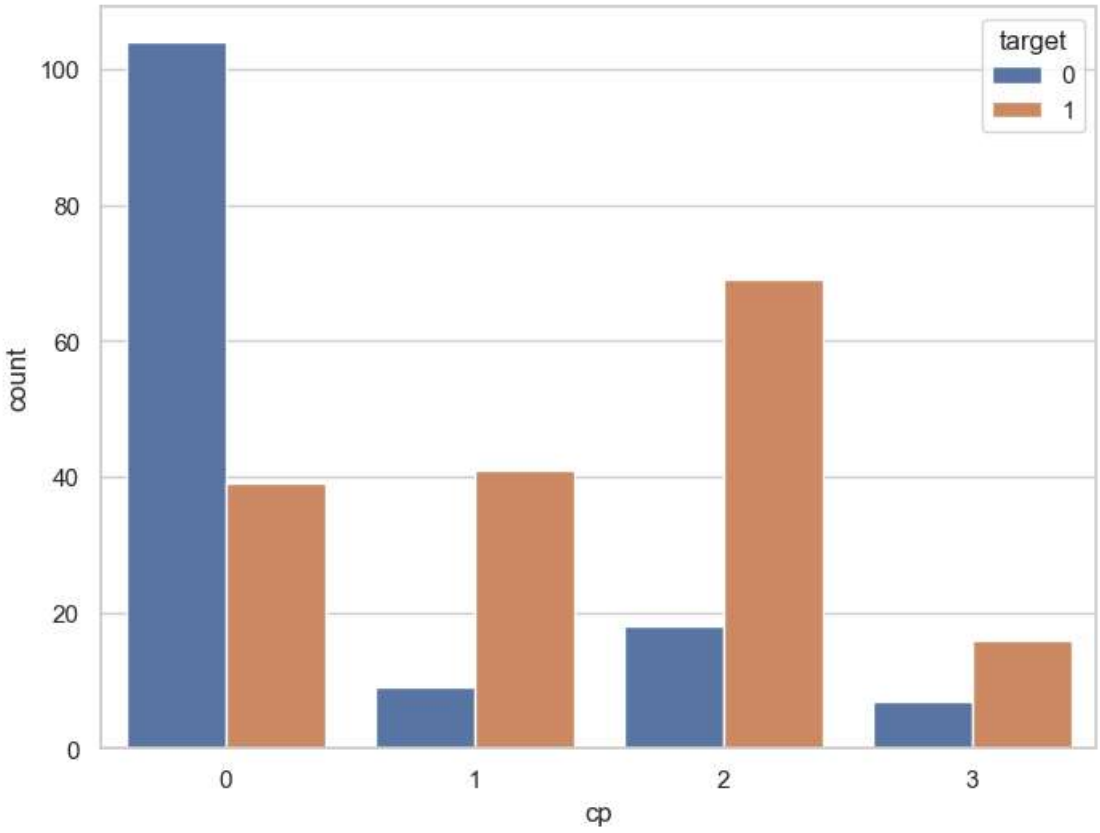


Frequency distribution of target variable wrt cp

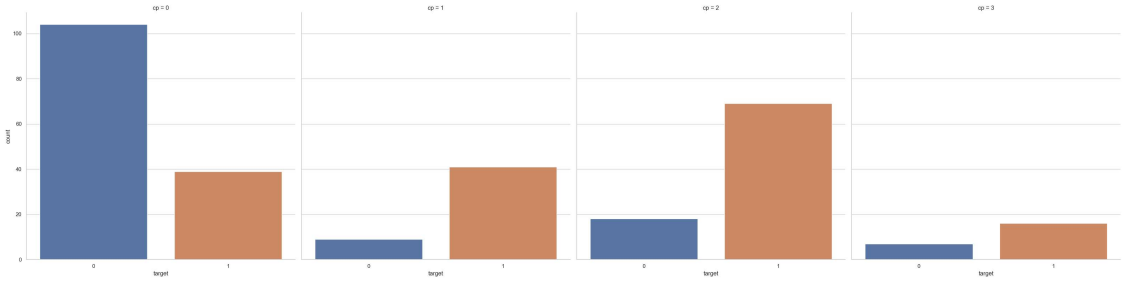
In [31]:
```
df.groupby("cp")["target"].value_counts()
```

Out[31]:
```
cp  target
0   0         104
    1          39
1   1          41
    0           9
2   1          69
    0          18
3   1          16
    0           7
Name: target, dtype: int64
```

In [32]:
```
f, ax = plt.subplots(figsize=(8, 6))
ax = sns.countplot(x="cp",hue="target", data=df)
plt.show()
```



In [33]:
```
ax = sns.catplot(x="target", col="cp", data=df, kind="count", height=8, as
```



In [34]:
```
# Analysis of target and thalach variable
```

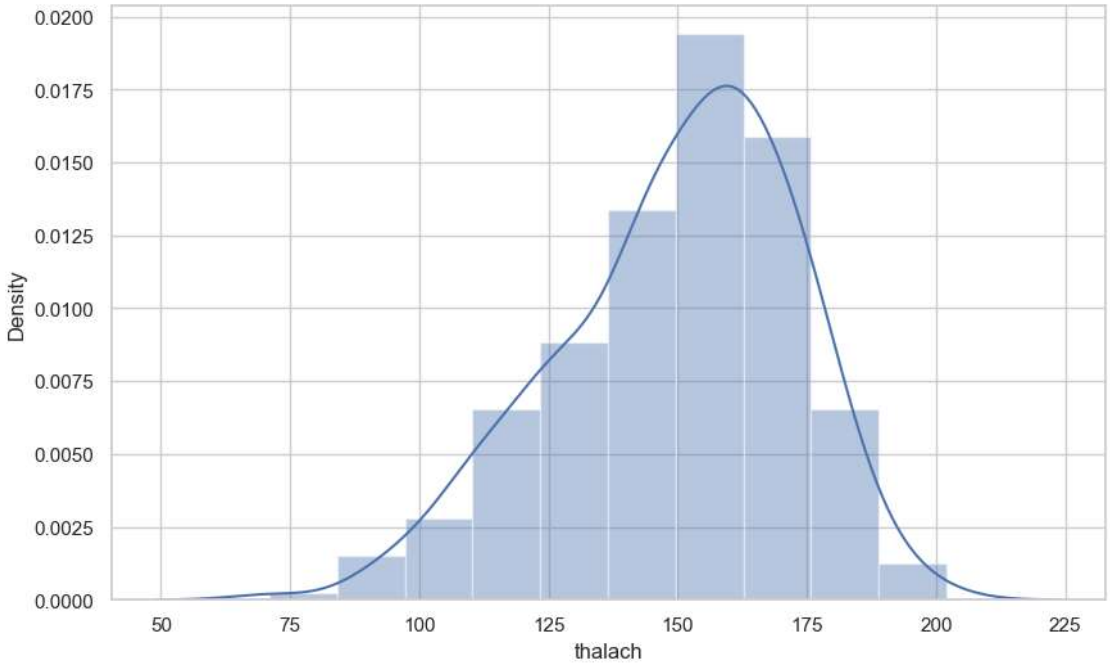
In [35]:
```
df["thalach"].nunique()
```

Out[35]: 91

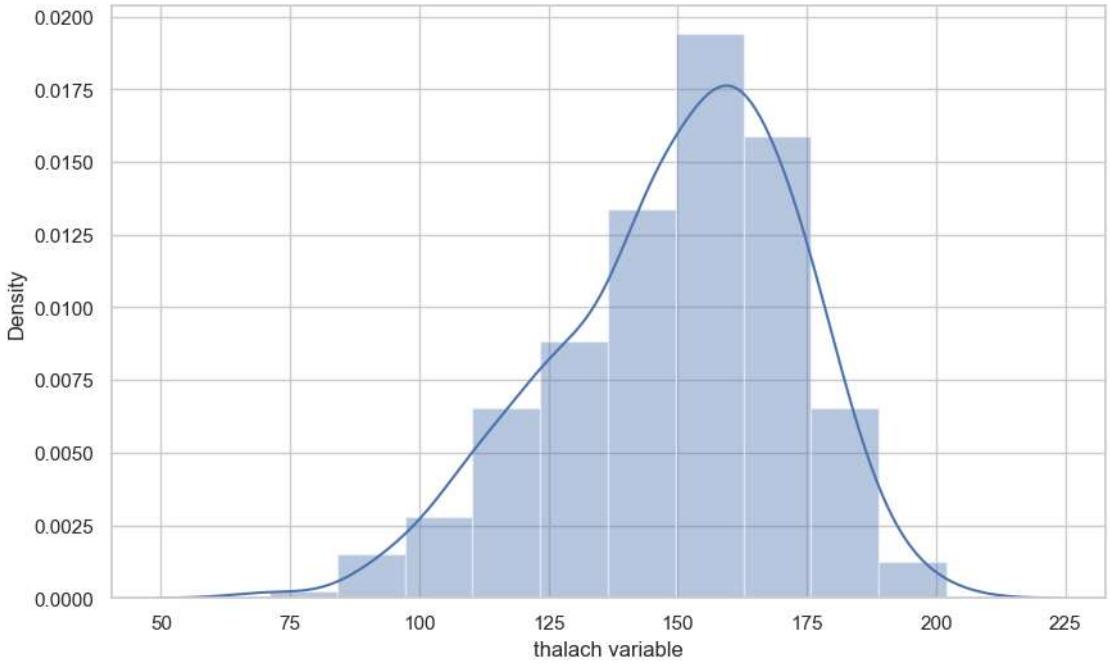
In [36]:
```
df["thalach"].unique()
```

Out[36]:
```
array([150, 187, 172, 178, 163, 148, 153, 173, 162, 174, 160, 139, 171,
       144, 158, 114, 151, 161, 179, 137, 157, 123, 152, 168, 140, 188,
       125, 170, 165, 142, 180, 143, 182, 156, 115, 149, 146, 175, 186,
       185, 159, 130, 190, 132, 147, 154, 202, 166, 164, 184, 122, 169,
       138, 111, 145, 194, 131, 133, 155, 167, 192, 121,  96, 126, 105,
       181, 116, 108, 129, 120, 112, 128, 109, 113,  99, 177, 141, 136,
        97, 127, 103, 124,  88, 195, 106,  95, 117,  71, 118, 134,  90],
      dtype=int64)
```

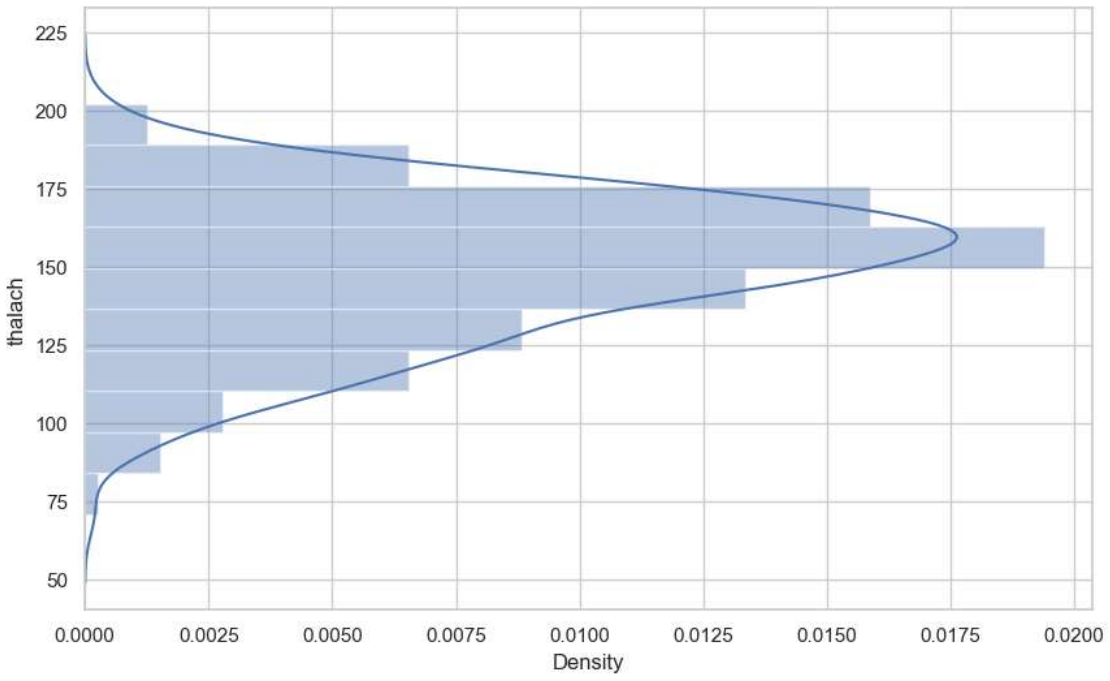
In [38]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
ax = sns.distplot(x, bins=10)
plt.show()
```



In [40]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
x = pd.Series(x, name="thalach variable")
ax = sns.distplot(x, bins=10)
plt.show()
```

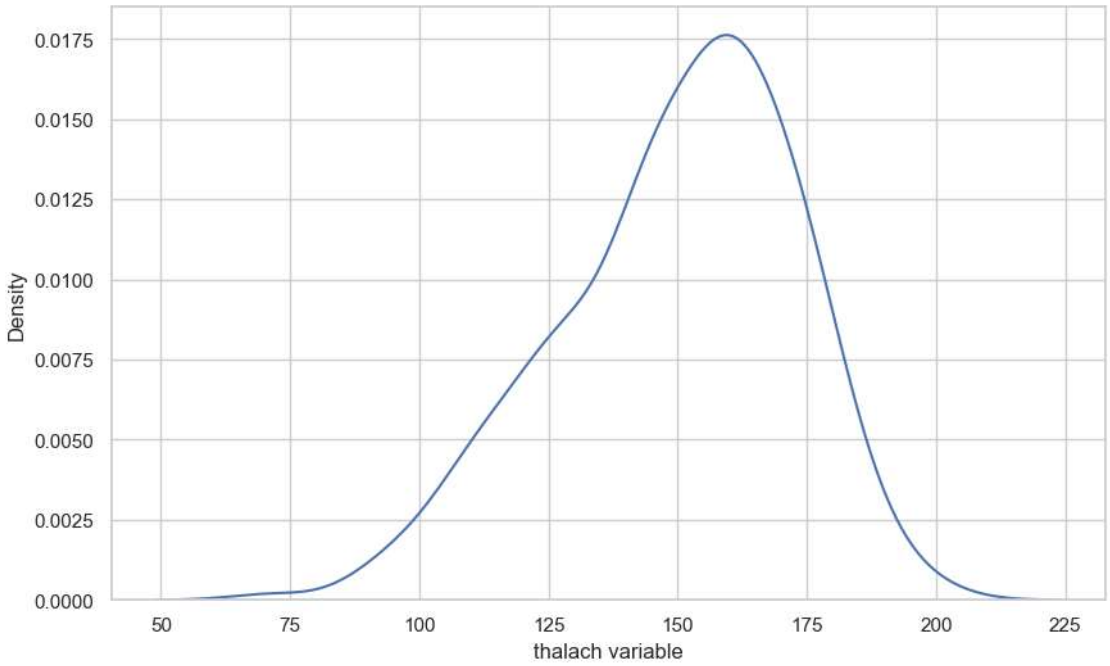


In [44]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
ax = sns.distplot(x, bins=10, vertical=True)
plt.show()
```

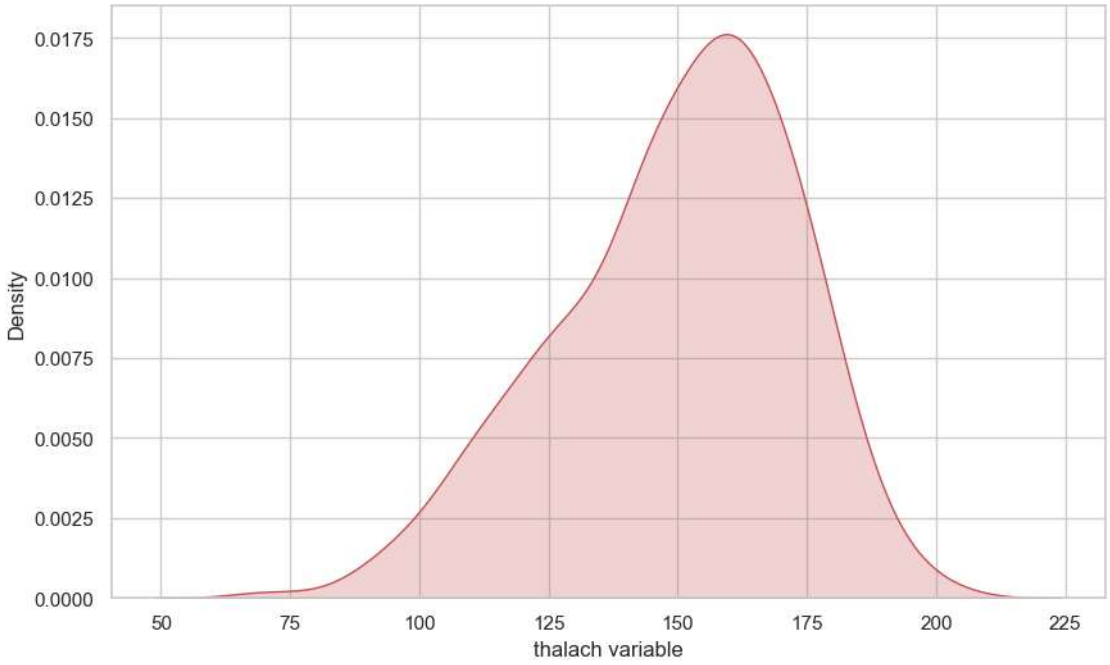


In [45]:
```
# seaborn kernel Density Estimation (KDE)Plot
```

In [47]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
x = pd.Series(x, name="thalach variable")
ax = sns.kdeplot(x)
plt.show()
```

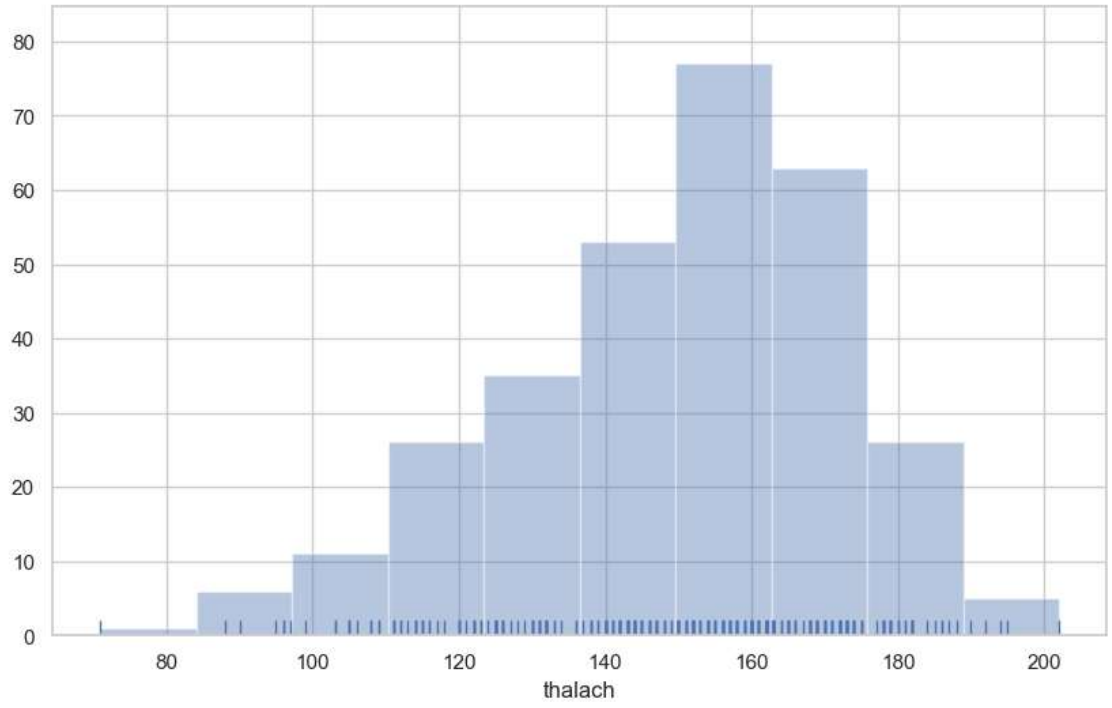


In [48]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
x = pd.Series(x, name="thalach variable")
ax = sns.kdeplot(x, shade=True, color="r")
plt.show()
```

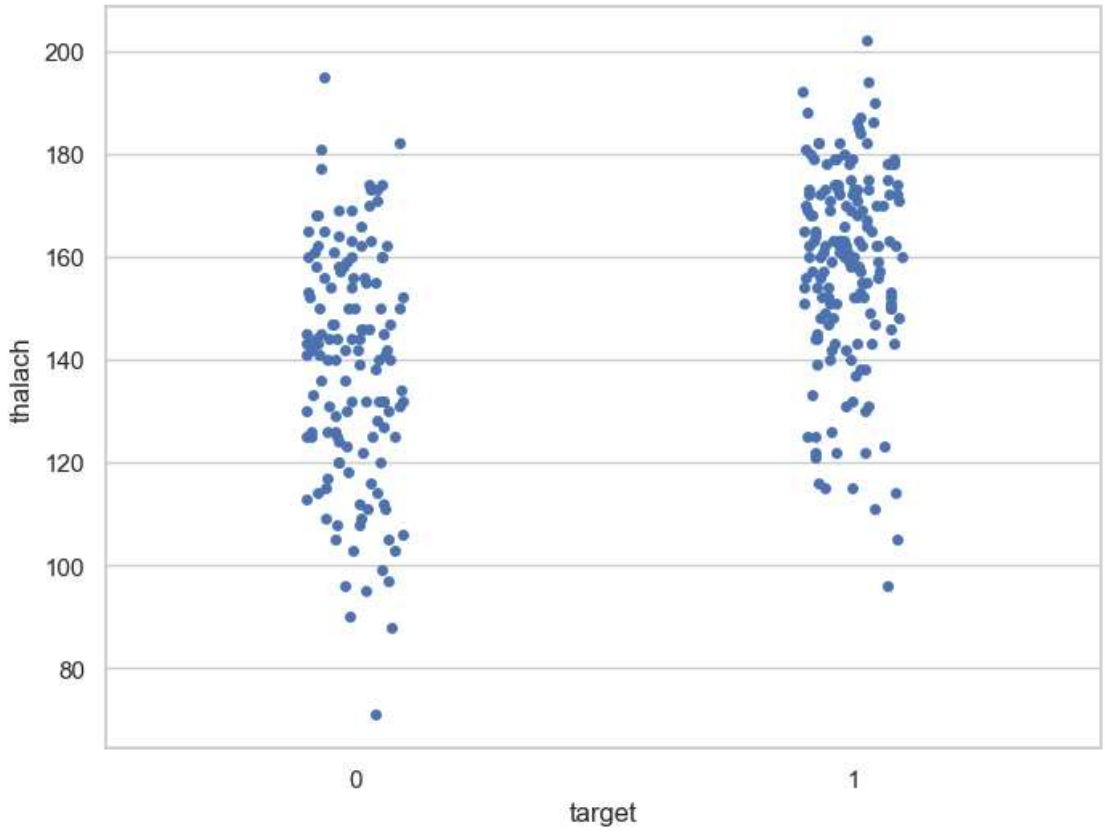


Histogram

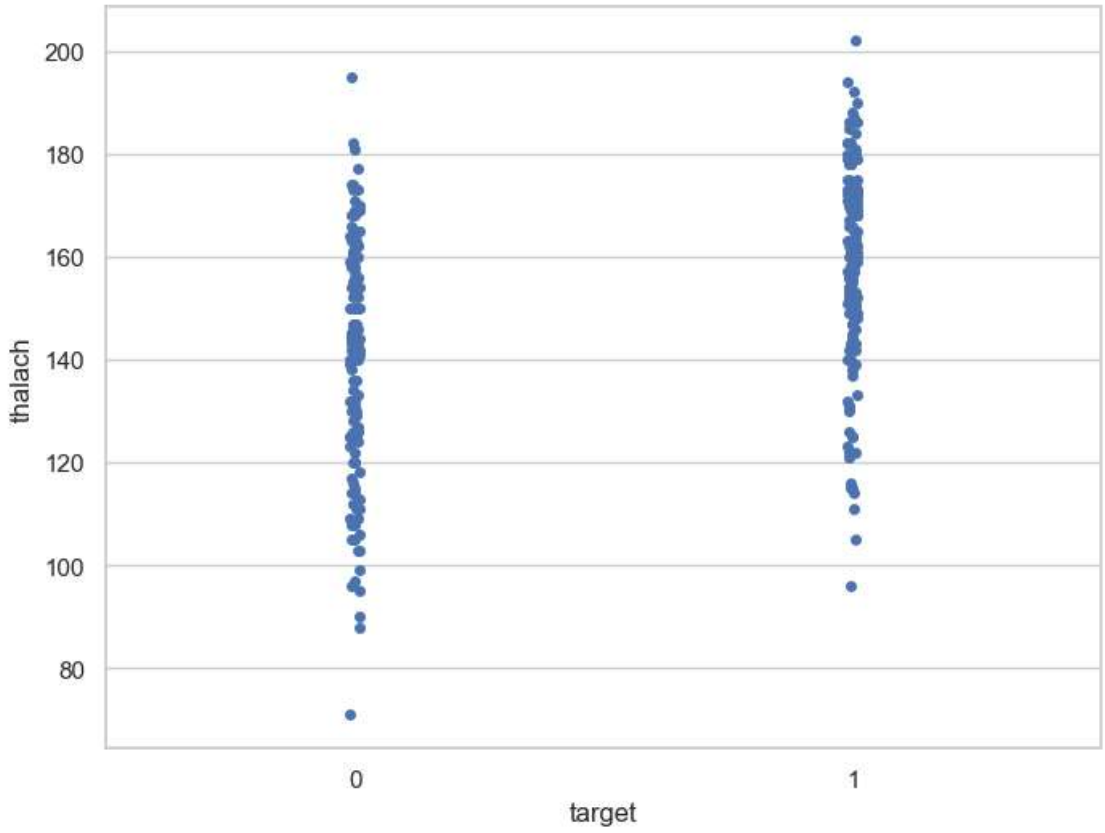
In [50]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df['thalach']
ax = sns.distplot(x, kde=False, rug=True, bins=10)
plt.show()
```



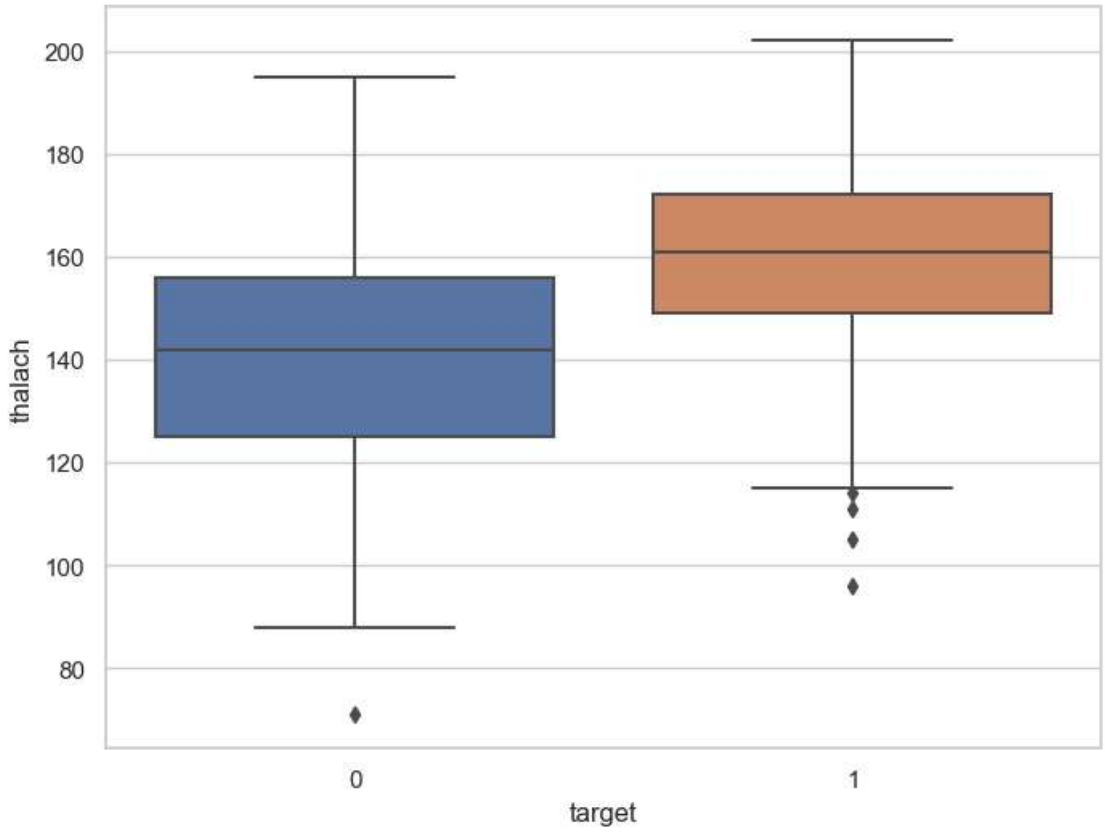
In [52]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.stripplot(x="target", y="thalach", data=df)
plt.show()
```



In [53]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.stripplot(x="target", y="thalach", data=df, jitter = 0.01)
plt.show()
```



In [54]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x="target", y="thalach", data=df)
plt.show()
```



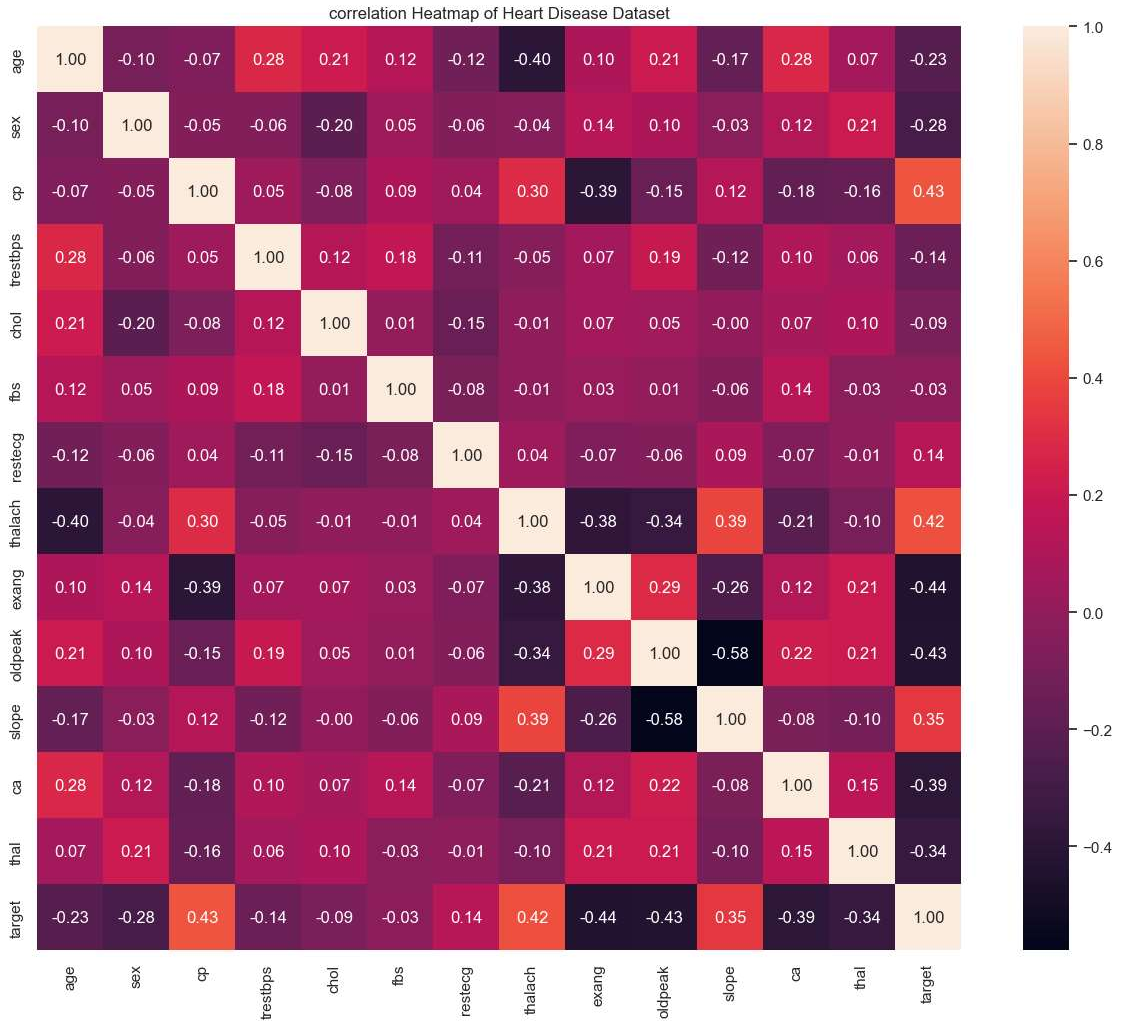
Findings of Bivariate Analysis

Findings of Bivariate Analysis are as follows –

- There is no variable which has strong positive correlation with `target` variable.
- There is no variable which has strong negative correlation with `target` variable.
- There is no correlation between `target` and `fbs` .
- The `cp` and `thalach` variables are mildly positively correlated with `target` variable.
- We can see that the `thalach` variable is slightly negatively skewed.
- The people suffering from heart disease (target = 1) have relatively higher heart rate (thalach) as compared to people who are not suffering from heart disease (target = 0).
- The people suffering from heart disease (target = 1) have relatively higher heart rate (thalach) as compared to people who are not suffering from heart disease (target = 0).

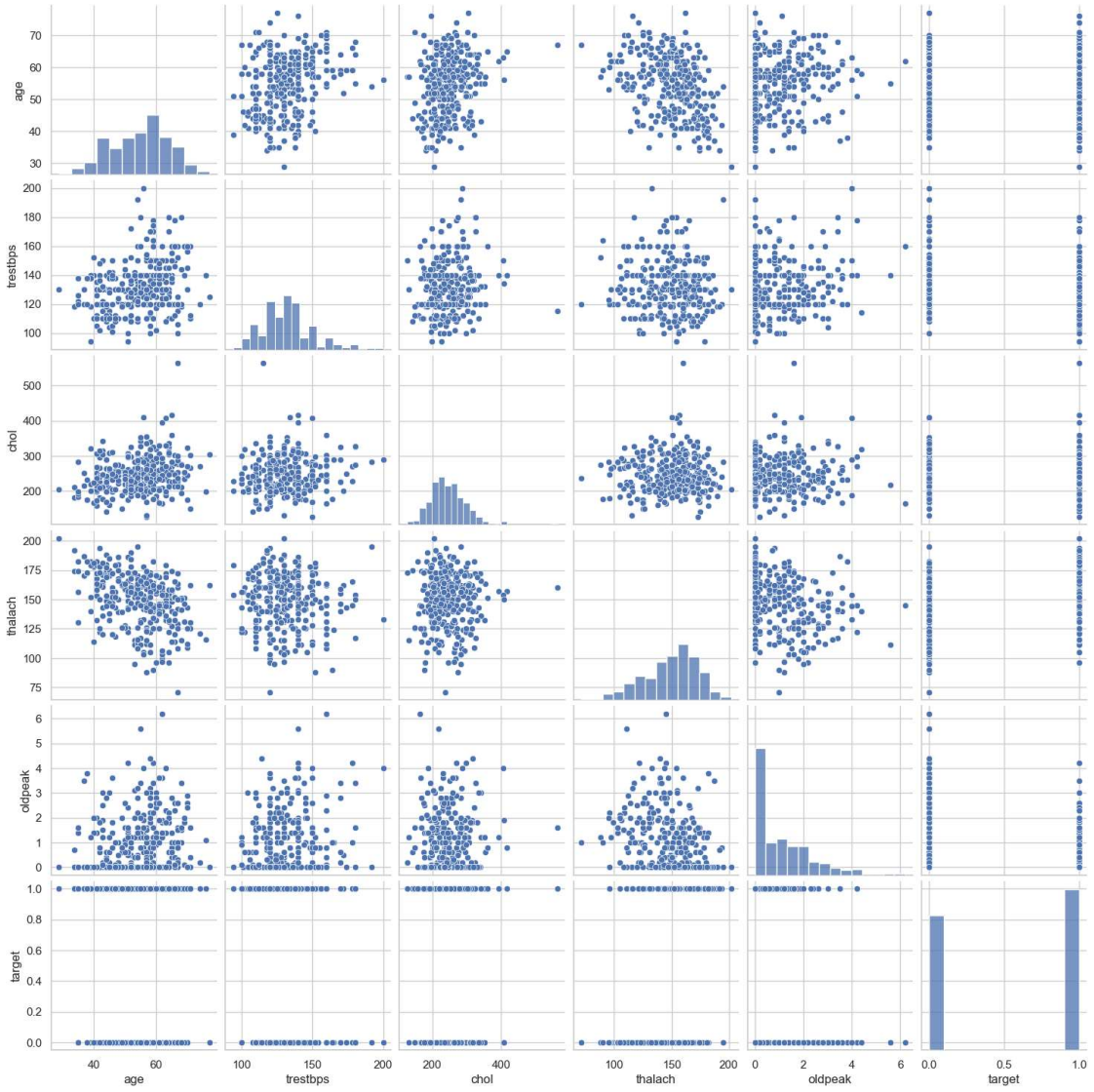
multivariante analysis

In [56]:
```
plt.figure(figsize=(16,12))
plt.title("correlation Heatmap of Heart Disease Dataset")
a = sns.heatmap(correlation, square=True, annot=True, fmt='.2f', linecolor
a.set_xticklabels(a.get_xticklabels(), rotation=90)
a.set_yticklabels(a.get_yticklabels(), rotation=90)
plt.show()
```



pair plot

In [57]:
```
num_var = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'target' ]
sns.pairplot(df[num_var], kind="scatter", diag_kind="hist")
plt.show()
```



In [58]:
```
# Analysis of age and other variables
```

In [59]:
```
df["age"].nunique()
```

Out[59]: 41

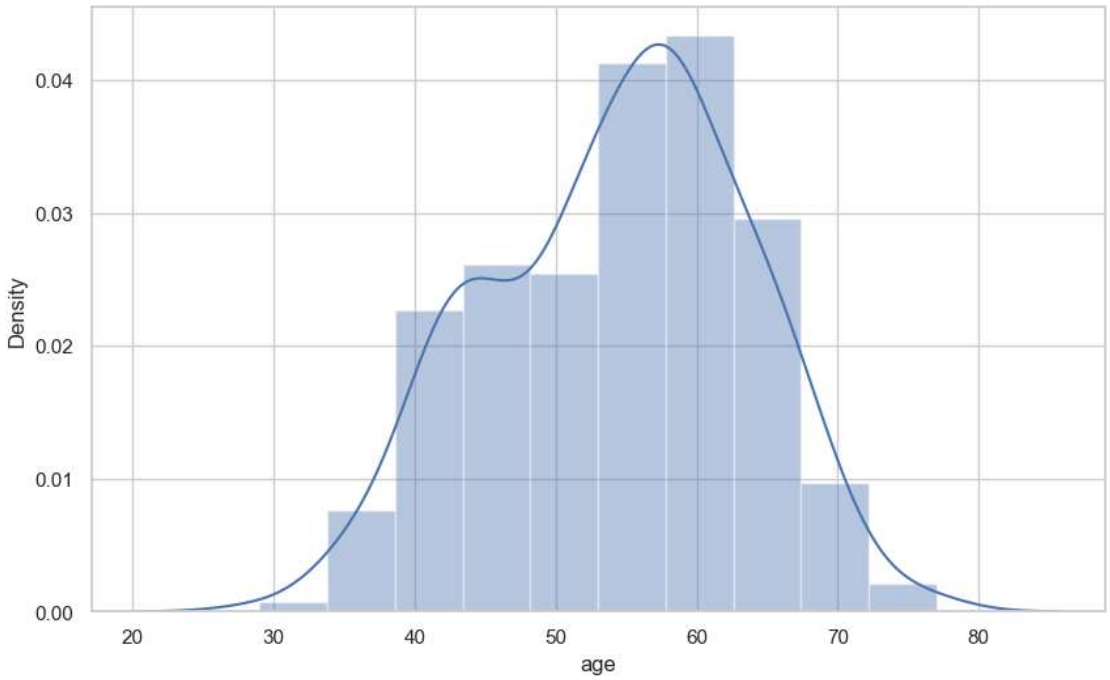
view statical summary of age variable

In [61]:
```
df["age"].describe()
```

Out[61]:
```
count    303.000000
mean      54.366337
std        9.082101
min       29.000000
25%       47.500000
50%       55.000000
75%       61.000000
max       77.000000
Name: age, dtype: float64
```

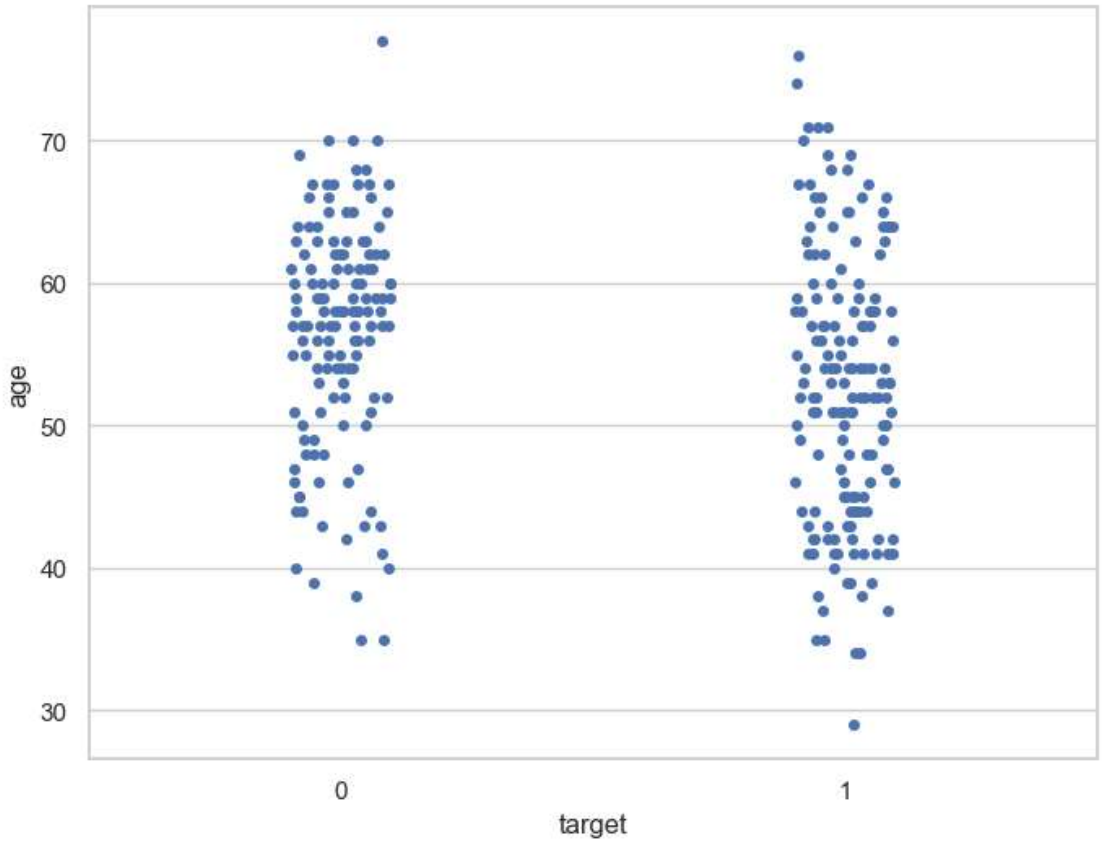
plot the distribution of age variable

In [65]:
```
f, ax = plt.subplots(figsize=(10,6))
x = df["age"]
ax = sns.distplot(x, bins=10)
plt.show()
```

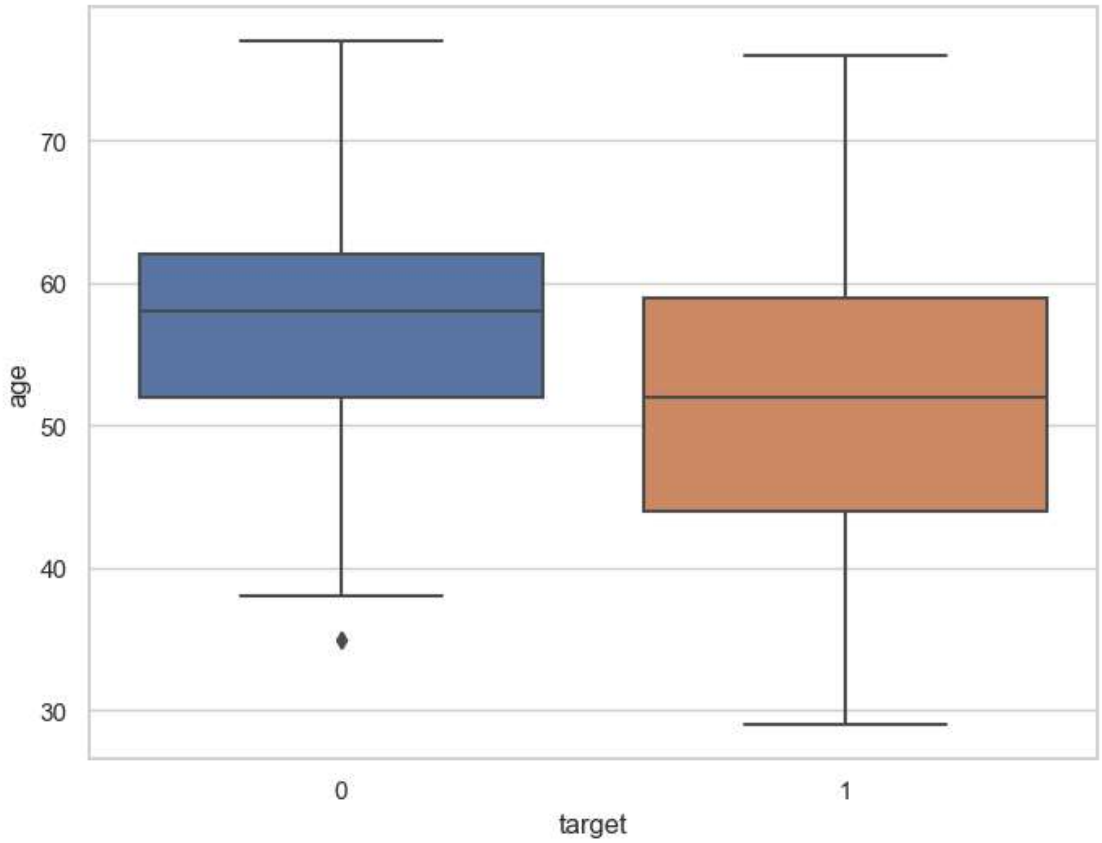


analyse the age and target varible

In [70]:
```
f, ax = plt.subplots(figsize=(8,6))
sns.stripplot(x="target", y="age", data=df)
plt.show()
```

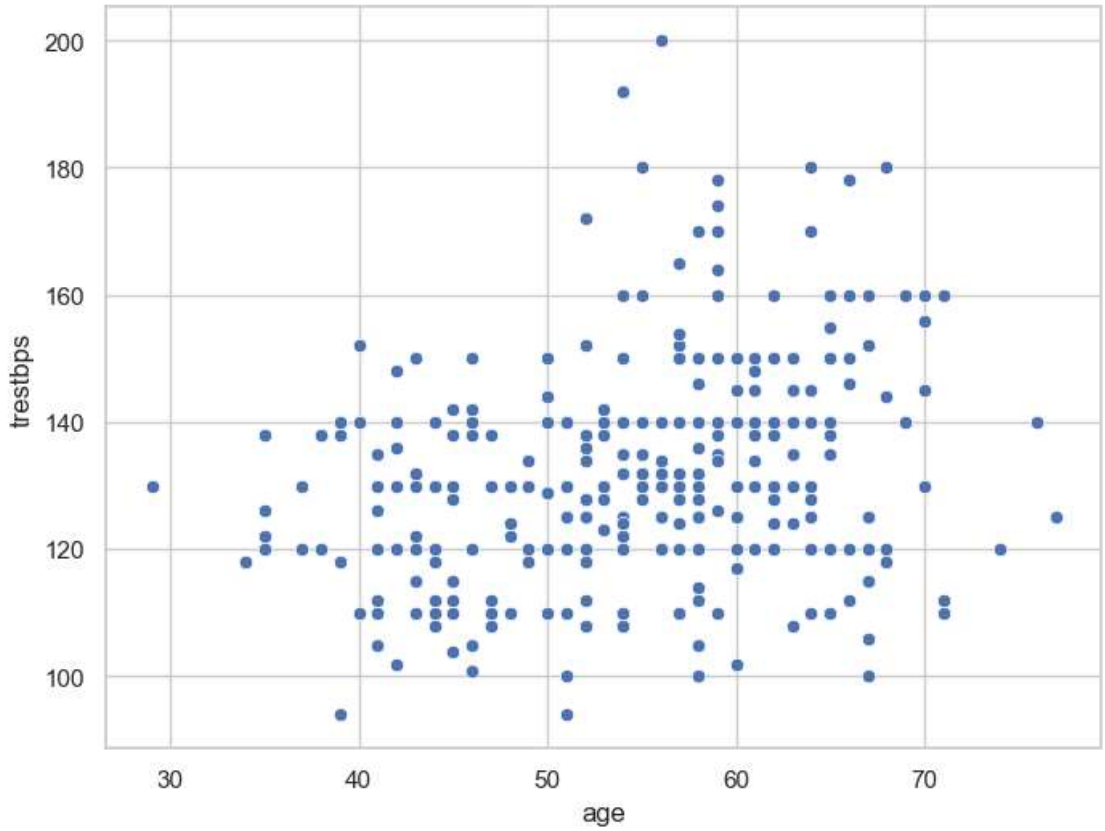


In [71]:
```
f, ax = plt.subplots(figsize=(8,6))
sns.boxplot(x="target", y="age", data=df)
plt.show()
```



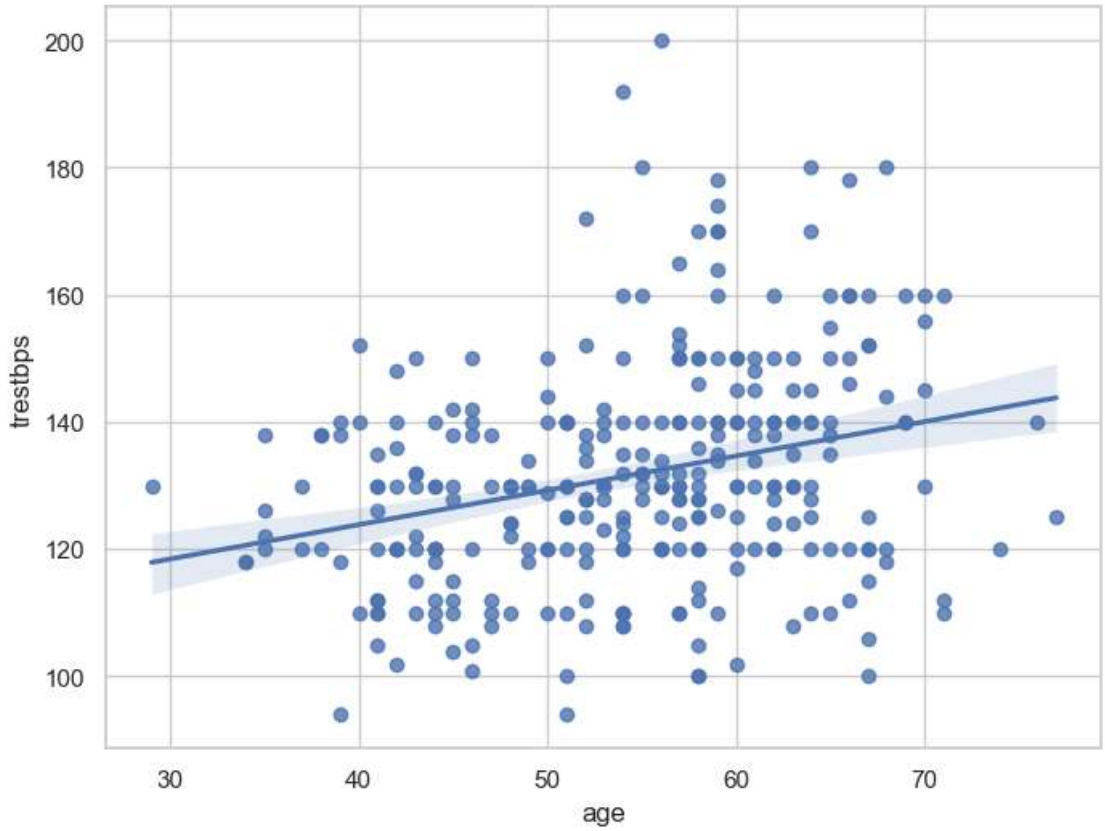
analyze age and trestbps variable

In [73]:
```
s, ax = plt.subplots(figsize=(8,6))
ax = sns.scatterplot(x="age", y="trestbps", data=df)
plt.show()
```

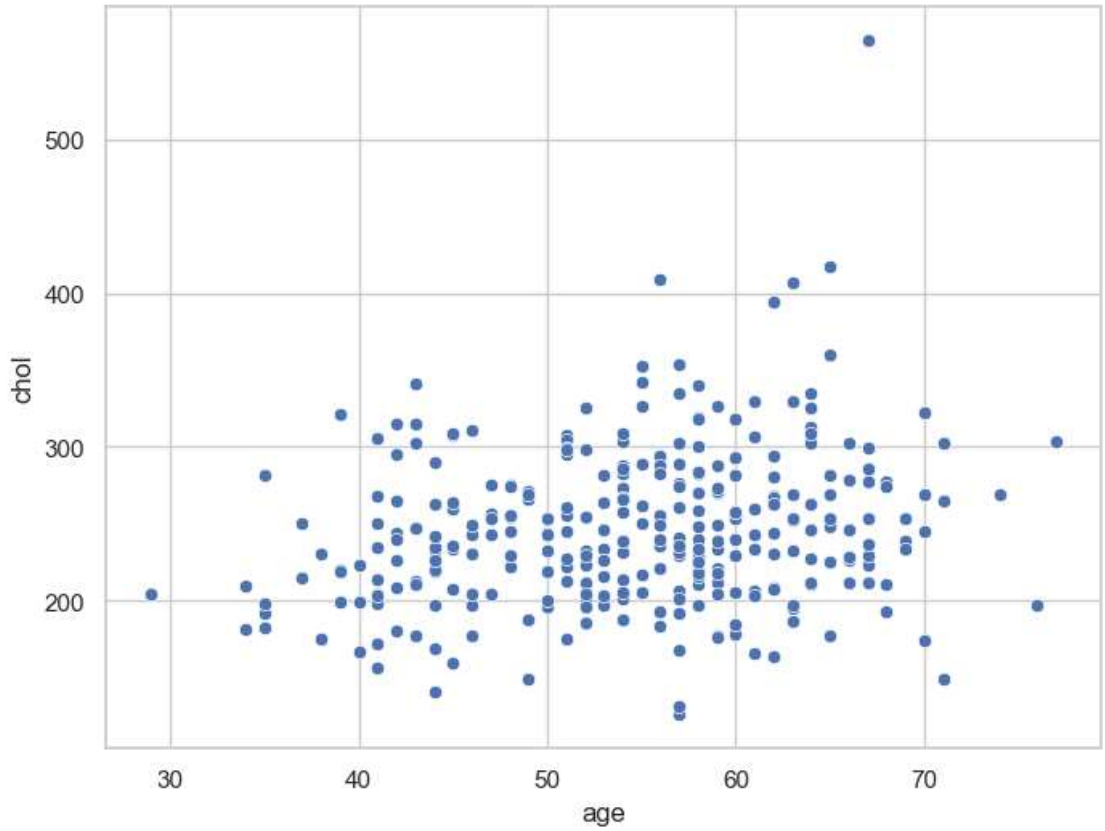


the above acatter plot shows that there is no correlation between age and trestbps variable

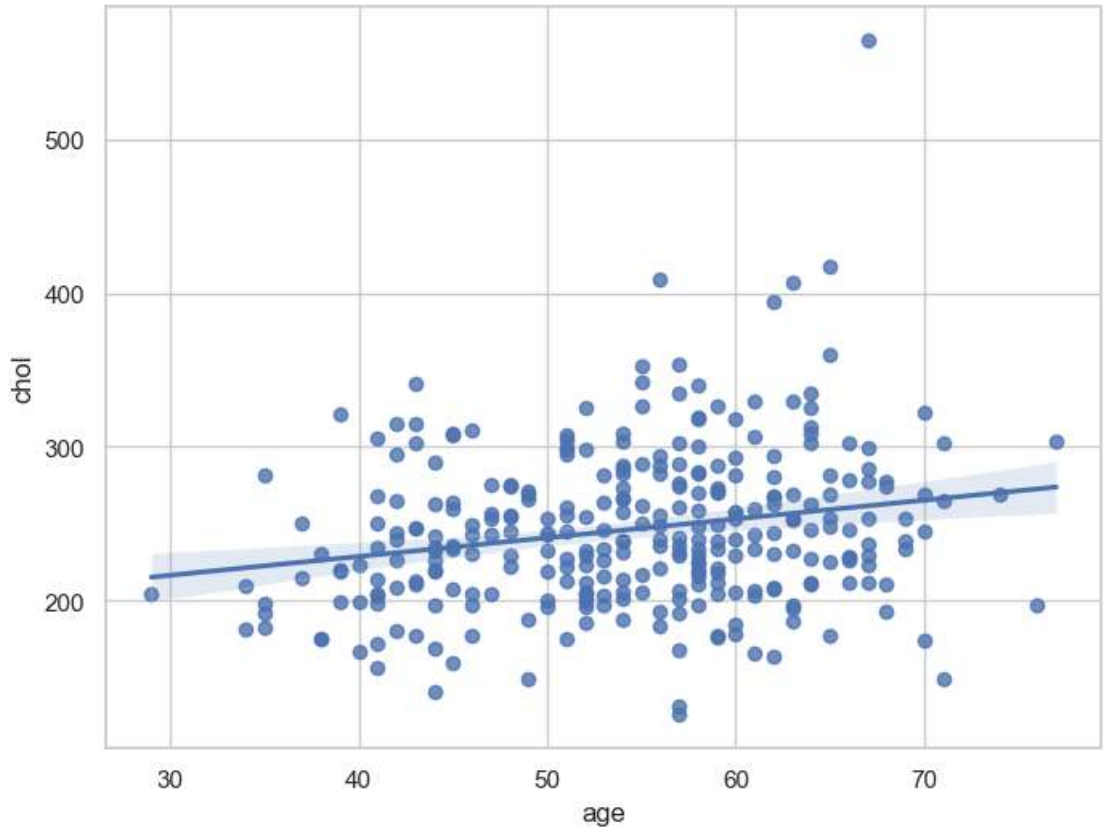
In [74]:
```
f, ax = plt.subplots(figsize=(8,6))
sns.regplot(x="age", y="trestbps", data=df)
plt.show()
```



In [75]:
```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.scatterplot(x="age", y="chol", data=df)
plt.show()
```

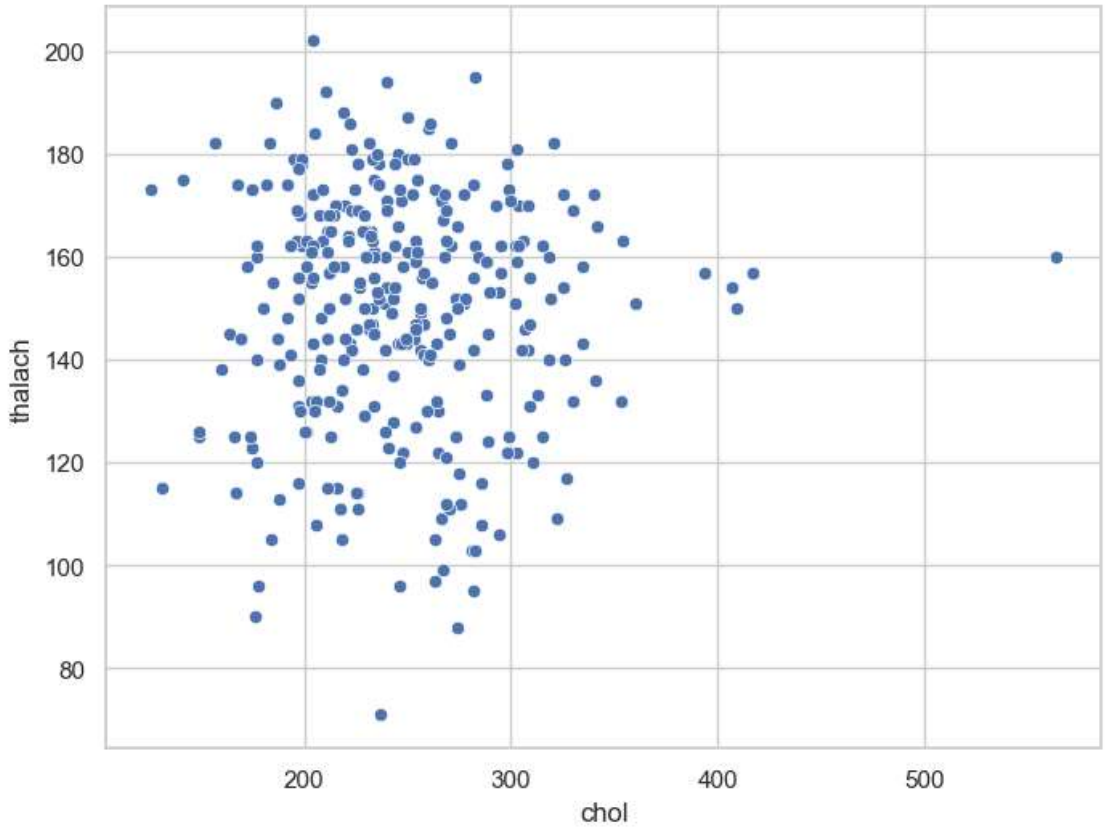


In [76]:
```
f, ax = plt.subplots(figsize=(8,6))
sns.regplot(x="age", y="chol", data=df)
plt.show()
```

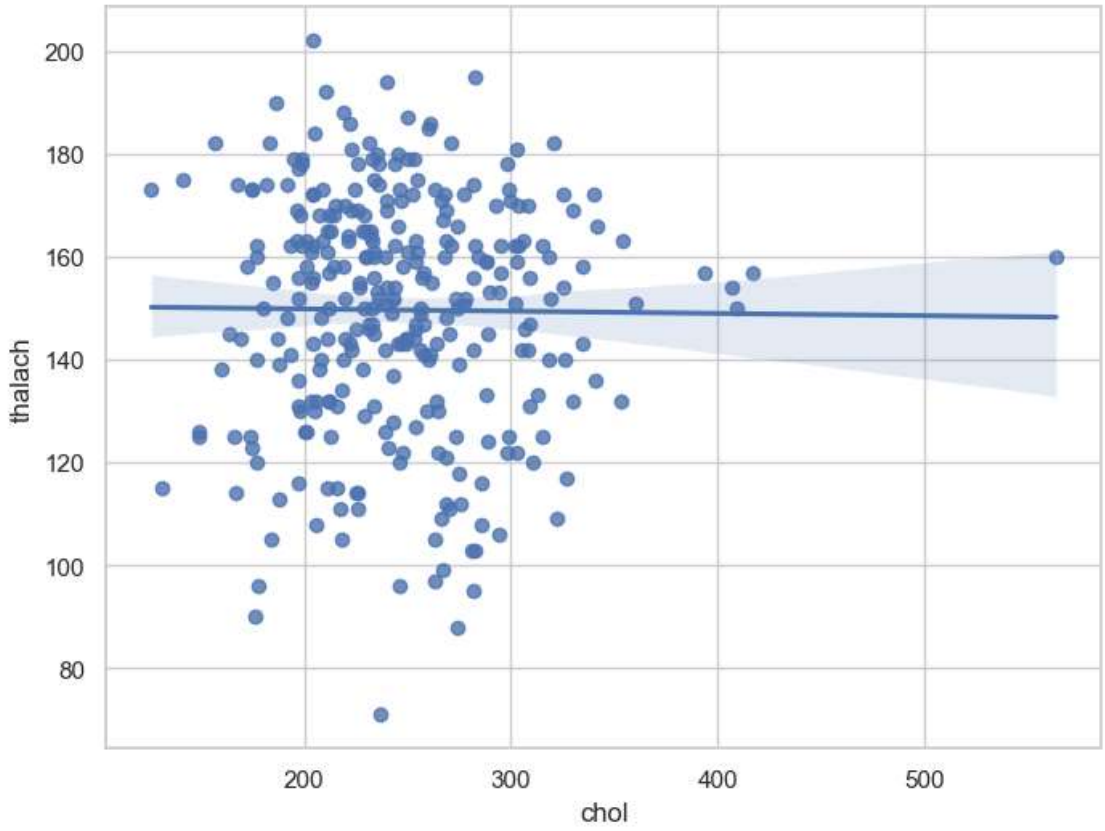


analyze chol and thalach variable

In [78]:
```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.scatterplot(x="chol", y="thalach", data=df)
plt.show()
```



In [79]:
```
f, ax = plt.subplots(figsize=(8,6))
ax = sns.regplot(x="chol", y="thalach", data=df)
plt.show()
```



In [80]:
```
#Dealing with missing value
```

In [81]:
```
df.isnull().sum()
```

Out[81]:
```
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 [82]:
```
# Check with Assert statement
```

In [85]:
```
assert pd.notnull(df).all().all()
```

In [86]:
```
assert (df >= 0).all().all()
```

outlier detection

i will make a boxplot to visualise outlier in the continous numerical variables

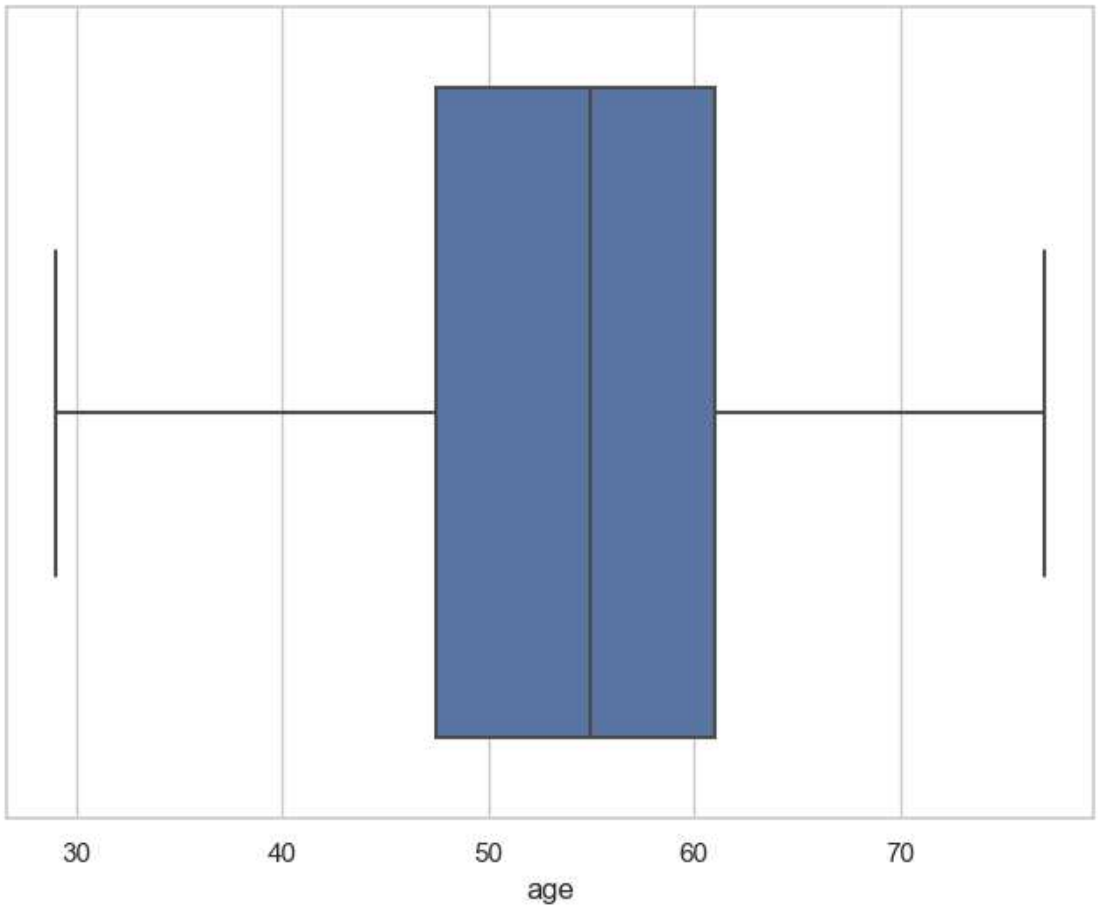
In [90]:
```
df["age"].describe()
```

Out[90]:
```
count    303.000000
mean      54.366337
std        9.082101
min       29.000000
25%       47.500000
50%       55.000000
75%       61.000000
max       77.000000
Name: age, dtype: float64
```

box plot of age variable

In [91]:
```
f, ax= plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["age"])
plt.show
```

Out[91]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

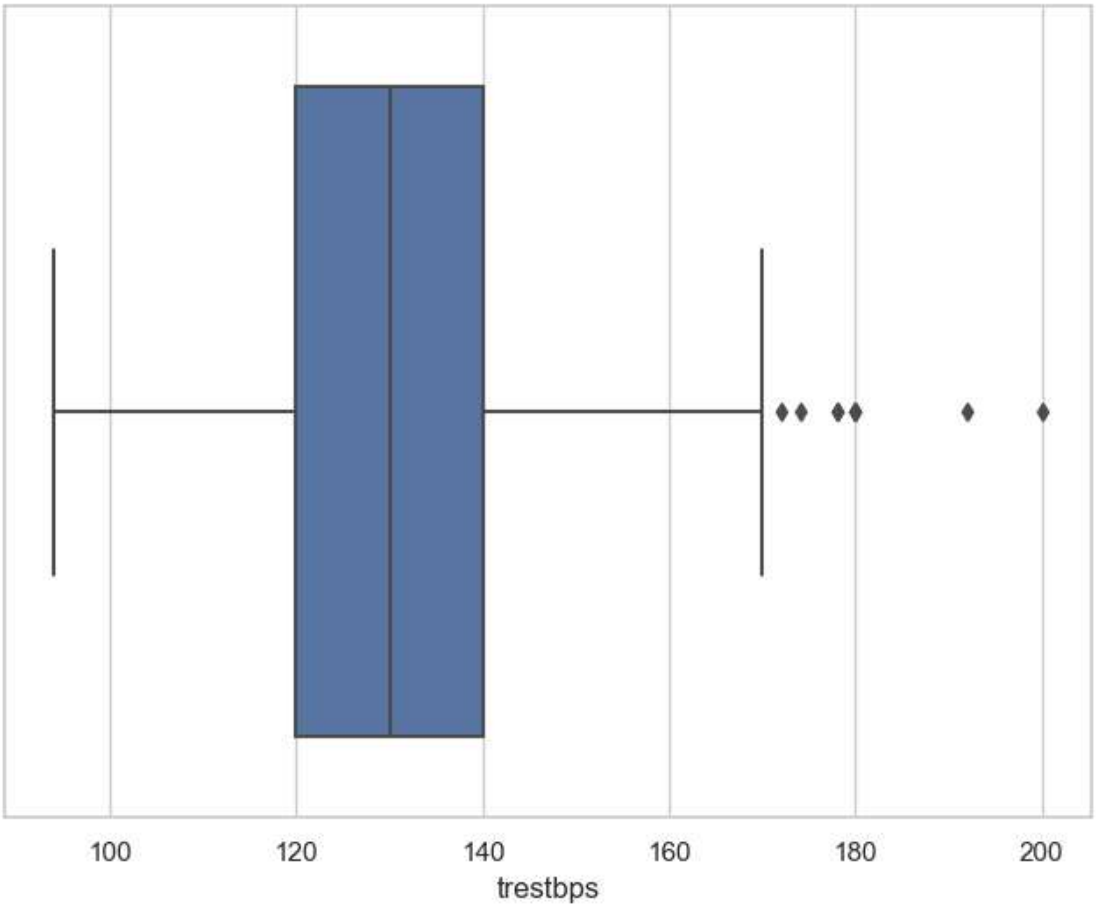


In [92]:
```
df["trestbps"].describe()
```

Out[92]:
```
count    303.000000
mean     131.623762
std       17.538143
min       94.000000
25%      120.000000
50%      130.000000
75%      140.000000
max      200.000000
Name: trestbps, dtype: float64
```

box-plot of trestbps variable

In [94]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["trestbps"])
plt.show()
```



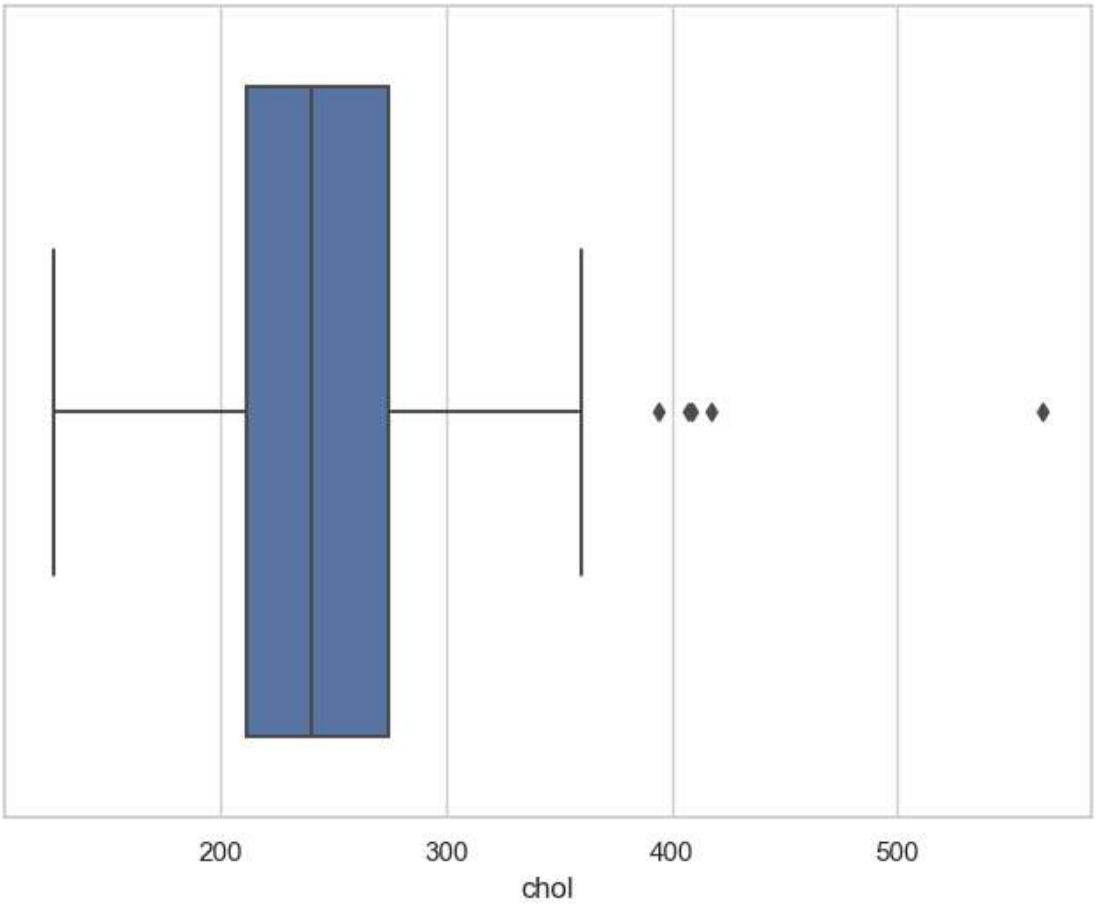
chol variable

In [95]:
```
df["chol"].describe()
```

Out[95]:
```
count    303.000000
mean     246.264026
std       51.830751
min      126.000000
25%      211.000000
50%      240.000000
75%      274.500000
max      564.000000
Name: chol, dtype: float64
```

Boxplot of chol variables

In [97]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["chol"])
plt.show()
```



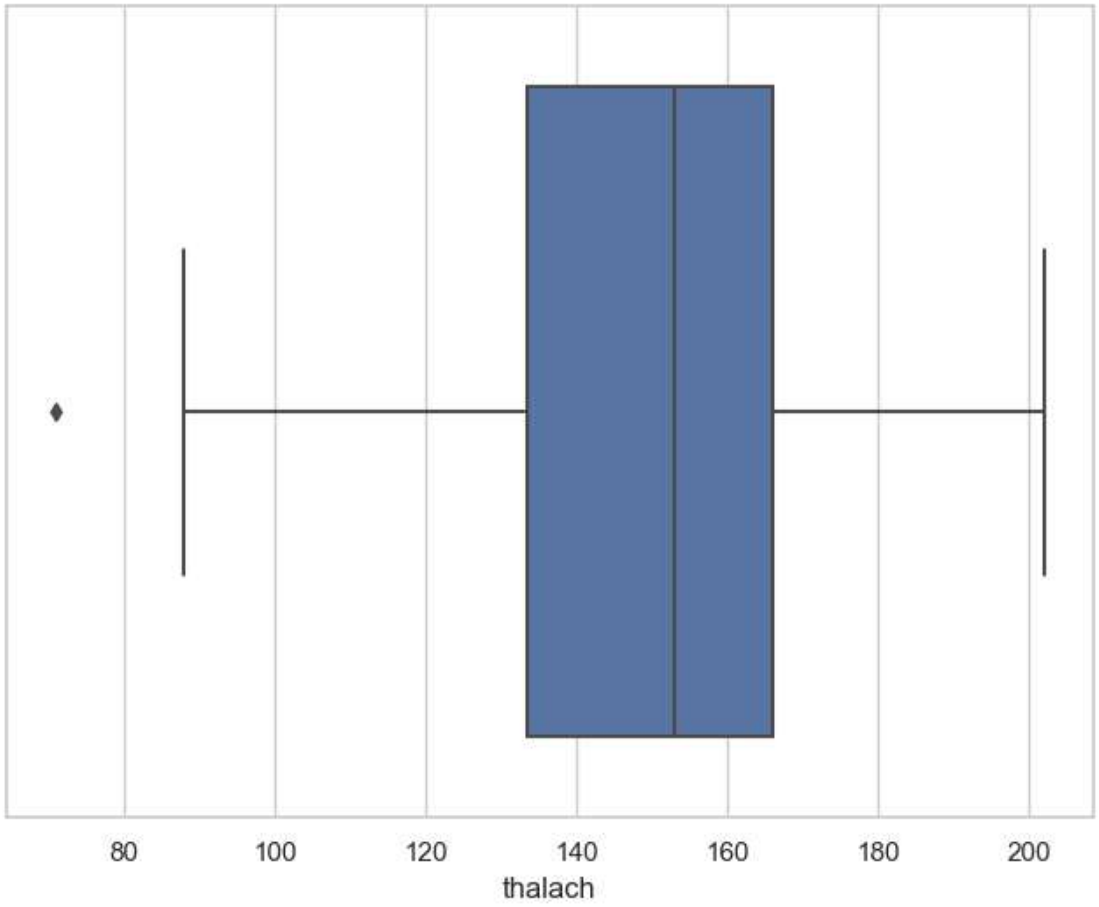
thalach variable

In [98]:
```
df["thalach"].describe()
```

Out[98]:
```
count    303.000000
mean     149.646865
std       22.905161
min       71.000000
25%      133.500000
50%      153.000000
75%      166.000000
max      202.000000
Name: thalach, dtype: float64
```

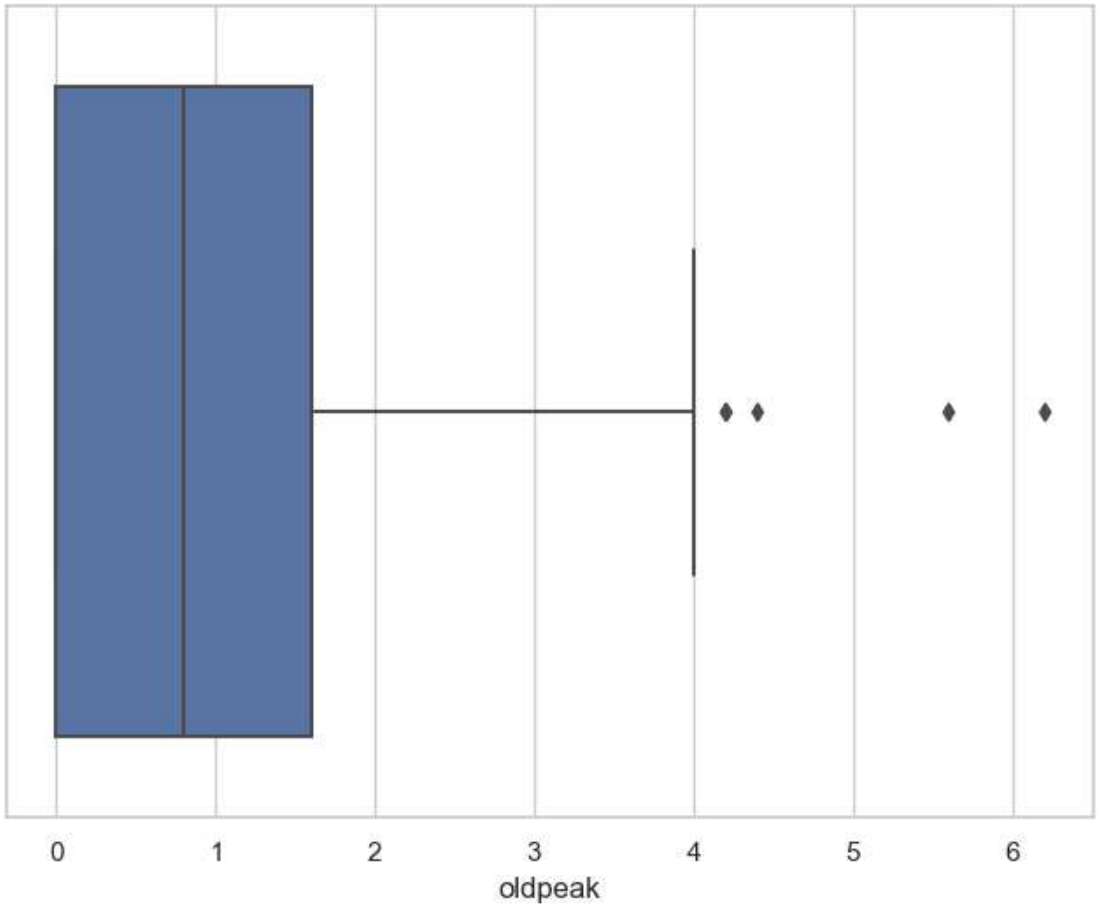
boxplot of thalach variable

In [101]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["thalach"])
plt.show()
```



oldpeak variable

In [104]:
```
f, ax = plt.subplots(figsize=(8, 6))
sns.boxplot(x=df["oldpeak"])
plt.show()
```



In [ ]:
```

```