HEART DISEASE DIAGNOSTIC ANALYSIS

In [1]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```python
data = pd.read_csv('Heart Disease data.csv')
```

In [3]:
```python
data.head(10)
```

Out[3]:

	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
5	58	0	0	100	248	0	0	122	0	1.0	1	0	2	1
6	58	1	0	114	318	0	2	140	0	4.4	0	3	1	0
7	55	1	0	160	289	0	0	145	1	0.8	1	1	3	0
8	46	1	0	120	249	0	0	144	0	0.8	2	0	3	0
9	54	1	0	122	286	0	0	116	1	3.2	1	2	2	0

*age : age in years *sex : (1 = male; 0 = female) *chest pain(cp) type : 4 values Value 0 - typical angina Value 1 - atypical angina Value 2 - non-anginal pain Value 3 - asymptomatic *trestbps : resting blood pressure (in mm Hg on admission to the hospital) *chol : serum cholestoral in mg/dl *fbs : (fasting blood sugar > 120mg/dl) (1 = true ; 0 = false) *restecg : resting electrocardiographic results Value 0 - normal Value 1 - having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) Value 2 - showing probable or definite left ventricular hypertrophy by Estes' criteria *thalach : maximum heart rate achieved *exang : exercise induced angina (1 = yes;0 = no) *oldpeak : ST depression induced by exercise relative *slope : the slope of the peak exercise ST segment Value 1 - upsloping Value 2 - flat Value 3 - downsloping *ca : number of major vessels (0-3) colored by flouroscopy *thal : 3=normal; 6=fixed defect; 7=reversable defect *target : 0=less chance of heart attack, 1=more chance of heart attack

In [6]:
```python
data.shape
```

Out[6]:
```
(1025, 14)
```

In [7]:
```python
print("Number of Rows",data.shape[0])
print("Number of Columns",data.shape[1])
```

```
Number of Rows 1025
Number of Columns 14
```

In [8]:
```python
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
```

In [10]:
```python
data.isnull().sum()
```

Out[10]:
```
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 [11]:
```python
data_dup = data.duplicated().any()
print(data_dup)
```

```
True
```

True means out dataset contains some duplicate values

In [13]:
```python
data=data.drop_duplicates()
```

In [14]:
```python
data.shape
```

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

About the Dataset

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

Out[15]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	302.00000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000	302.000000
mean	54.42053	0.682119	0.963576	131.602649	246.500000	0.149007	0.526490	149.569536	0.327815	1.043046	1.397351	0.718543	2.314570	0.543046
std	9.04797	0.466426	1.032044	17.563394	51.753489	0.356686	0.526027	22.903527	0.470196	1.161452	0.616274	1.006748	0.613026	0.498970
min	29.00000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	48.00000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.250000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	55.50000	1.000000	1.000000	130.000000	240.500000	0.000000	1.000000	152.500000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.00000	1.000000	2.000000	140.000000	274.750000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.000000	1.000000
max	77.00000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

Correlation table

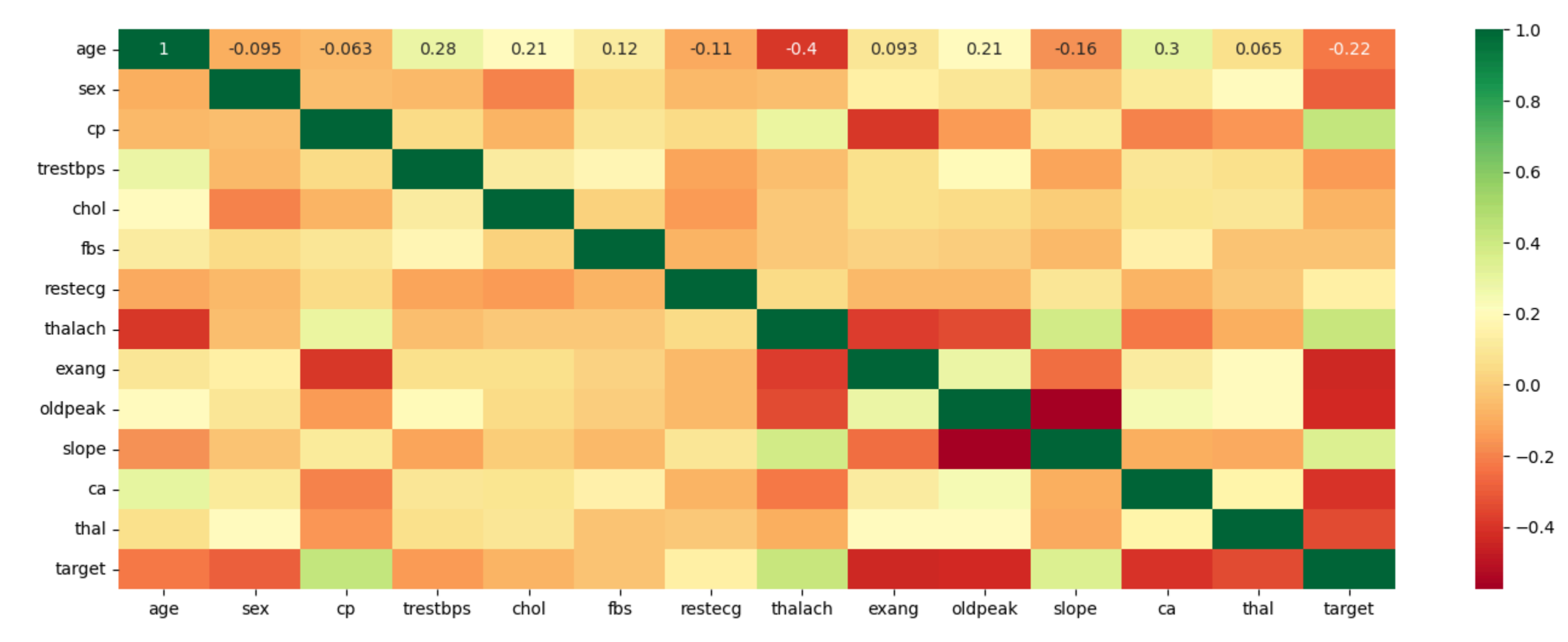
In [21]:
```python
correlation=data.corr()
data.corr()
```

Out[21]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.094962	-0.063107	0.283121	0.207216	0.119492	-0.111590	-0.395235	0.093216	0.206040	-0.164124	0.302261	0.065317	-0.221476
sex	-0.094962	1.000000	-0.051740	-0.057647	-0.195571	0.046022	-0.060351	-0.046439	0.143460	0.098322	-0.032990	0.113060	0.211452	-0.283609
cp	-0.063107	-0.051740	1.000000	0.046486	-0.072682	0.096018	0.041561	0.293367	-0.392937	-0.146692	0.116854	-0.195356	-0.160370	0.432080
trestbps	0.283121	-0.057647	0.046486	1.000000	0.125256	0.178125	-0.115367	-0.048023	0.068526	0.194600	-0.122873	0.099248	0.062870	-0.146269
chol	0.207216	-0.195571	-0.072682	0.125256	1.000000	0.011428	-0.147602	-0.005308	0.064099	0.050086	0.000417	0.086878	0.096810	-0.081437
fbs	0.119492	0.046022	0.096018	0.178125	0.011428	1.000000	-0.083081	-0.007169	0.024729	0.004514	-0.058654	0.144935	-0.032752	-0.026826
restecg	-0.111590	-0.060351	0.041561	-0.115367	-0.147602	-0.083081	1.000000	0.041210	-0.068807	-0.056251	0.090402	-0.083112	-0.010473	0.134874
thalach	-0.395235	-0.046439	0.293367	-0.048023	-0.005308	-0.007169	0.041210	1.000000	-0.377411	-0.342201	0.384754	-0.228311	-0.094910	0.419955
exang	0.093216	0.143460	-0.392937	0.068526	0.064099	0.024729	-0.068807	-0.377411	1.000000	0.286766	-0.256106	0.125377	0.205826	-0.435601
oldpeak	0.206040	0.098322	-0.146692	0.194600	0.050086	0.004514	-0.056251	-0.342201	0.286766	1.000000	-0.576314	0.236560	0.209090	-0.429146
slope	-0.164124	-0.032990	0.116854	-0.122873	0.000417	-0.058654	0.090402	0.384754	-0.256106	-0.576314	1.000000	-0.092236	-0.103314	0.343940
ca	0.302261	0.113060	-0.195356	0.099248	0.086878	0.144935	-0.083112	-0.228311	0.125377	0.236560	-0.092236	1.000000	0.160085	-0.408992
thal	0.065317	0.211452	-0.160370	0.062870	0.096810	-0.032752	-0.010473	-0.094910	0.205826	0.209090	-0.103314	0.160085	1.000000	-0.343101
target	-0.221476	-0.283609	0.432080	-0.146269	-0.081437	-0.026826	0.134874	0.419955	-0.435601	-0.429146	0.343940	-0.408992	-0.343101	1.000000

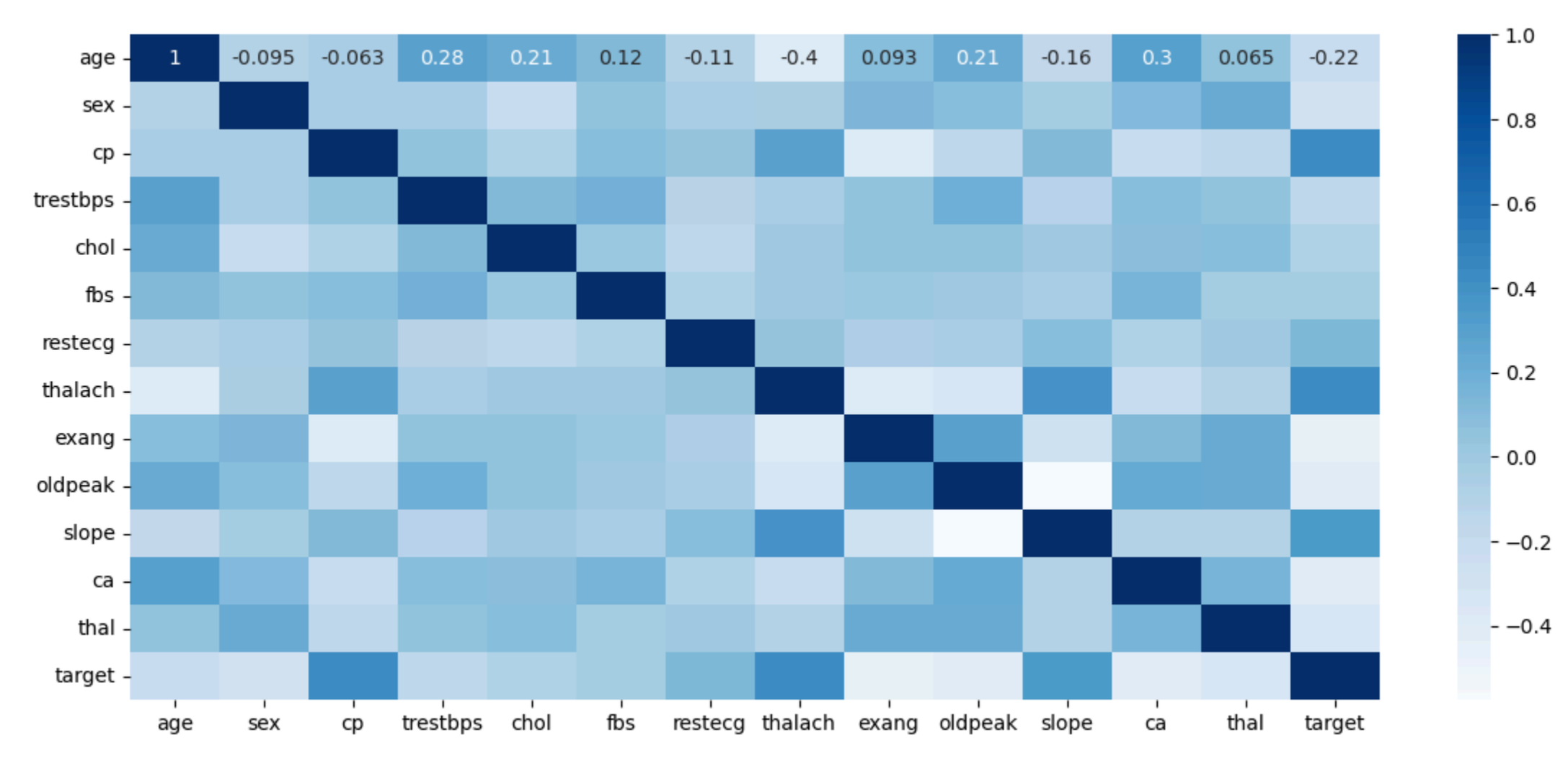
In [24]:
```python
plt.figure(figsize=(17,6))
sns.heatmap(correlation,cmap="RdYlGn",annot=True)
```

Out[24]:
```
<Axes: >
```



In [29]:
```python
plt.figure(figsize=(14,6))
sns.heatmap(correlation,cmap="Blues",annot=True)
```

Out[29]:
```
<Axes: >
```



Comparison of Heart Attack Chances b/w Male and Female

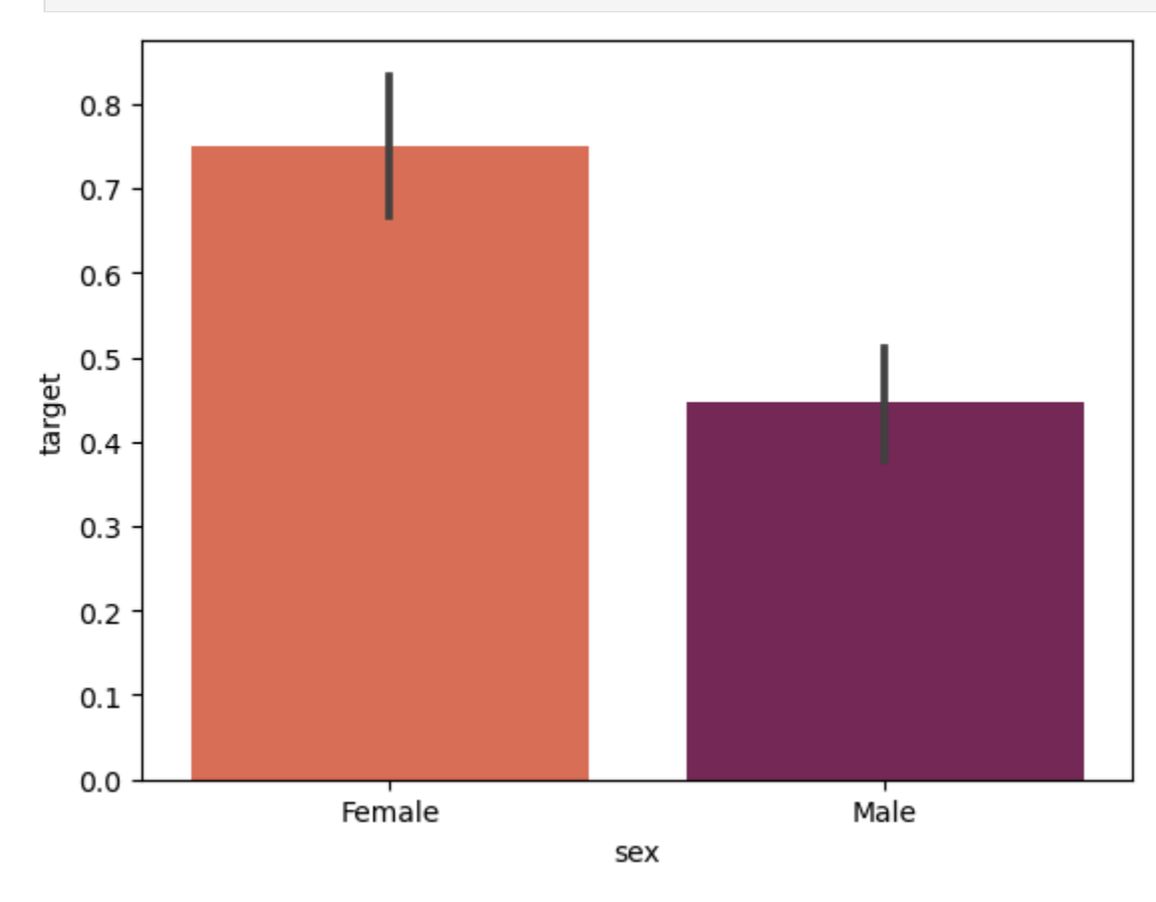
In [30]:
```python
data.columns
```

Out[30]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [31]:
```python
data['target'].value_counts()
```

Out[31]:
```
target
1    164
0    138
Name: count, dtype: int64
```

In [58]:
```python
sns.barplot(x='sex',y='target',data=data,palette='rocket_r')
plt.xticks([0,1],['Female','Male'])
plt.show()
```



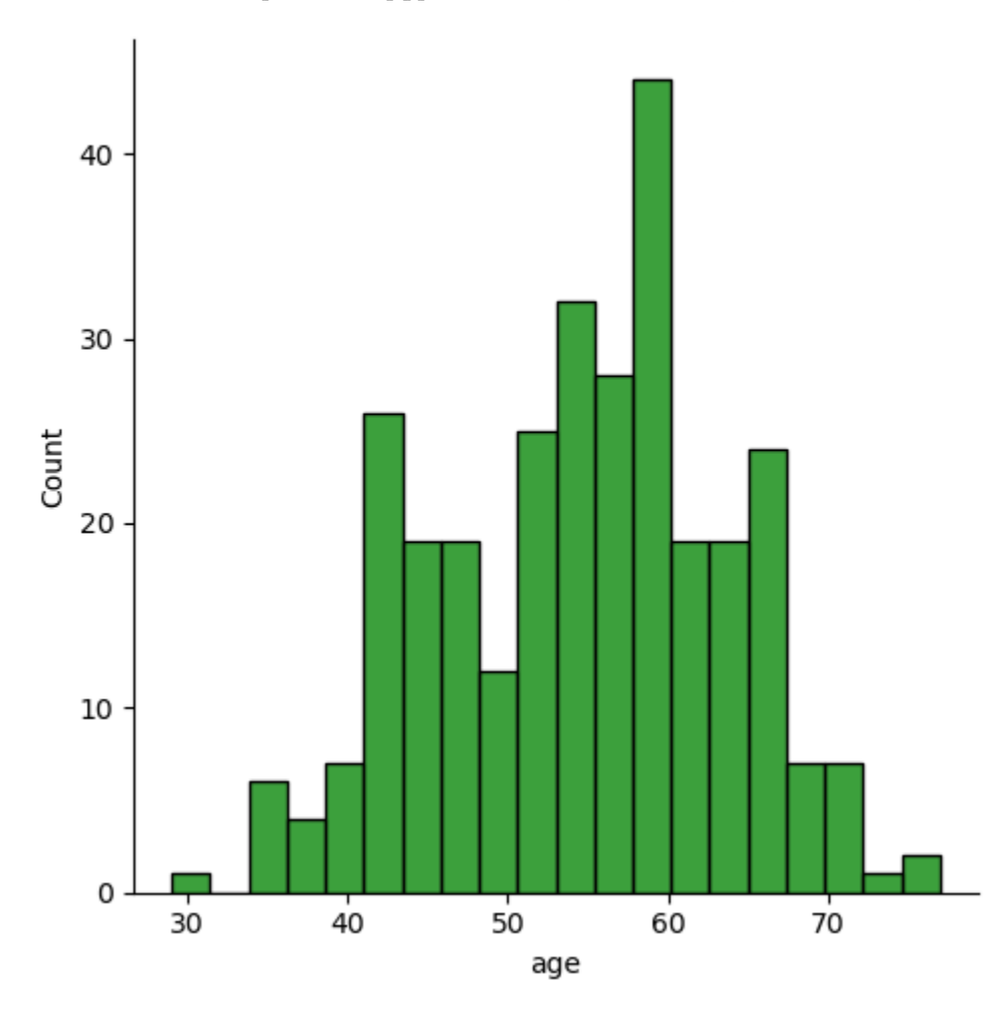
target : 0=less chance of heart attack, 1=more chance of heart attack

Age Distribution in the Dataset

In [124…
```python
sns.displot(data['age'],bins=20,color='green')
plt.show
```

```
C:\Users\admin\anaconda5\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

Out[124…
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

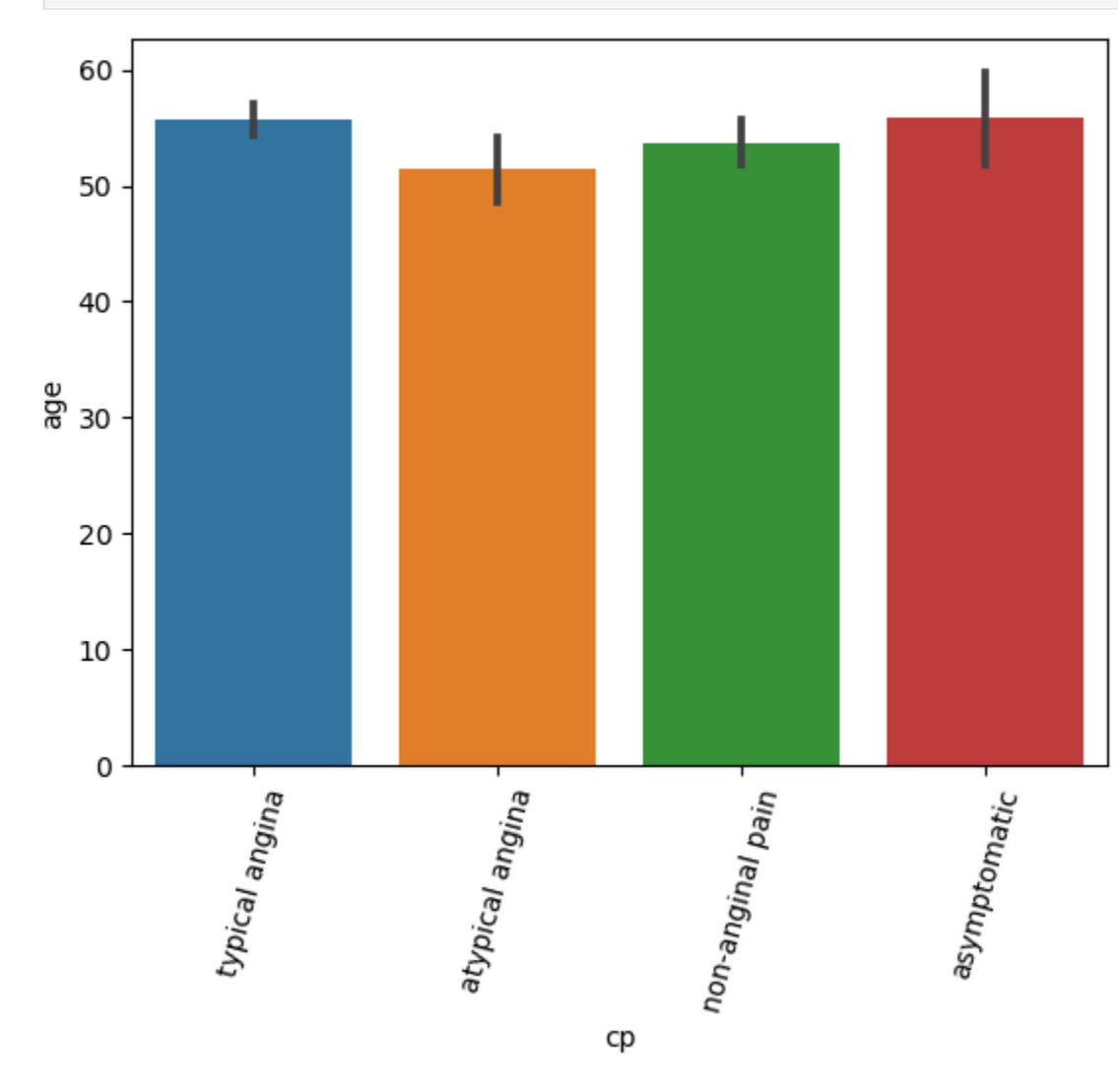


From this plot we can see that most of the people in the study has age from 50-60

Chest Pain Disrtribution as per Age

Chest Pain Types : 4 values Value 0 - typical angina Value 1 - atypical angina Value 2 - non-anginal pain Value 3 - asymptomatic

In [95]:
```python
sns.barplot(x='cp',y='age',data=data)
plt.xticks([0,1,2,3],['typical angina','atypical angina','non-anginal pain','asymptomatic'])
plt.xticks(rotation=75)
plt.show()
```



This shows people with age group 50-60 are more prone to Value 0 (Typical Angina) and Value 3 (Asymptomatic)

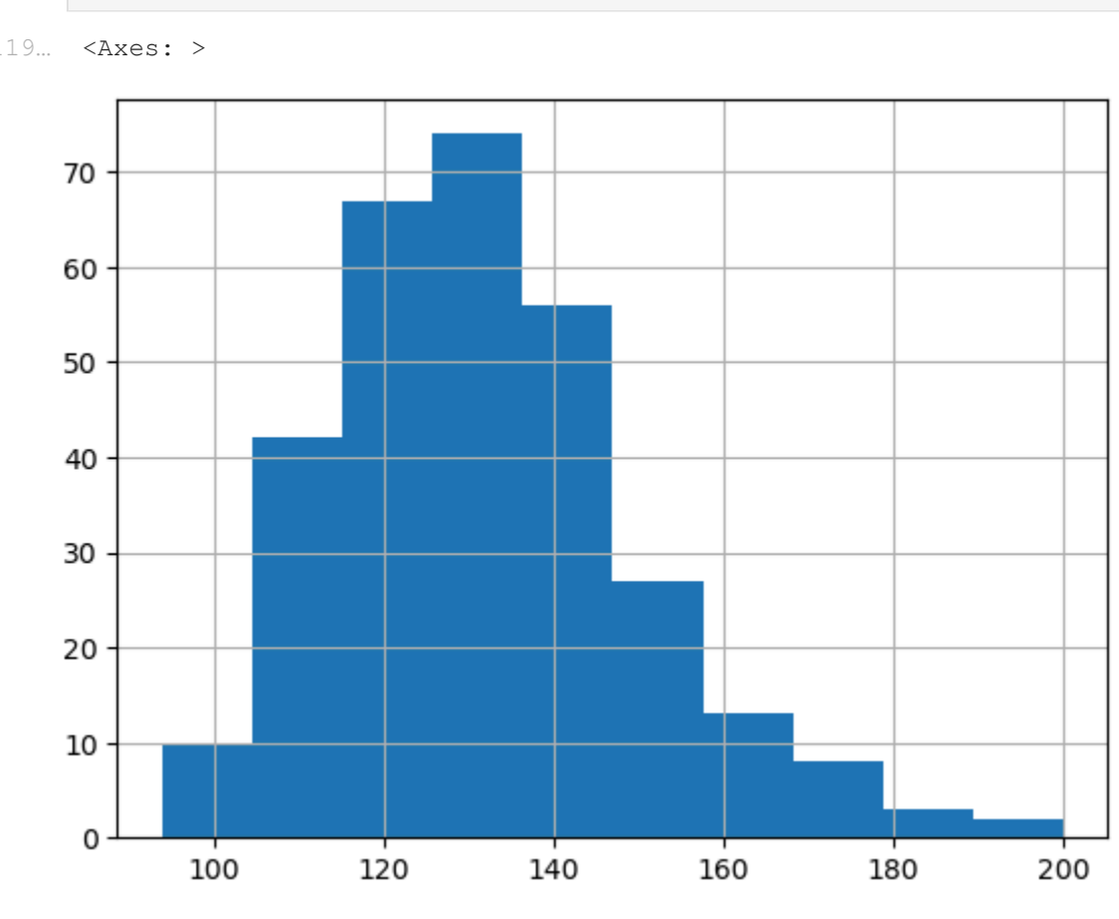
Resting Blood Pressure Distribution

In [99]:
```python
data.columns
```

Out[99]:
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [119…
```python
data['trestbps'].hist()
```

Out[119…
```
<Axes: >
```



From this Histogram we can see that the BLOOD PRESSURE of the people in the study is between 120-130.

Comparing trestbps (Resting Blood Pressure) as per Sex

In [106…
```python
g = sns.FacetGrid(data,hue='sex',aspect=4)
g.map(sns.kdeplot,'trestbps',shade=True)
plt.legend(labels=['Male','Female'])
plt.show()
```

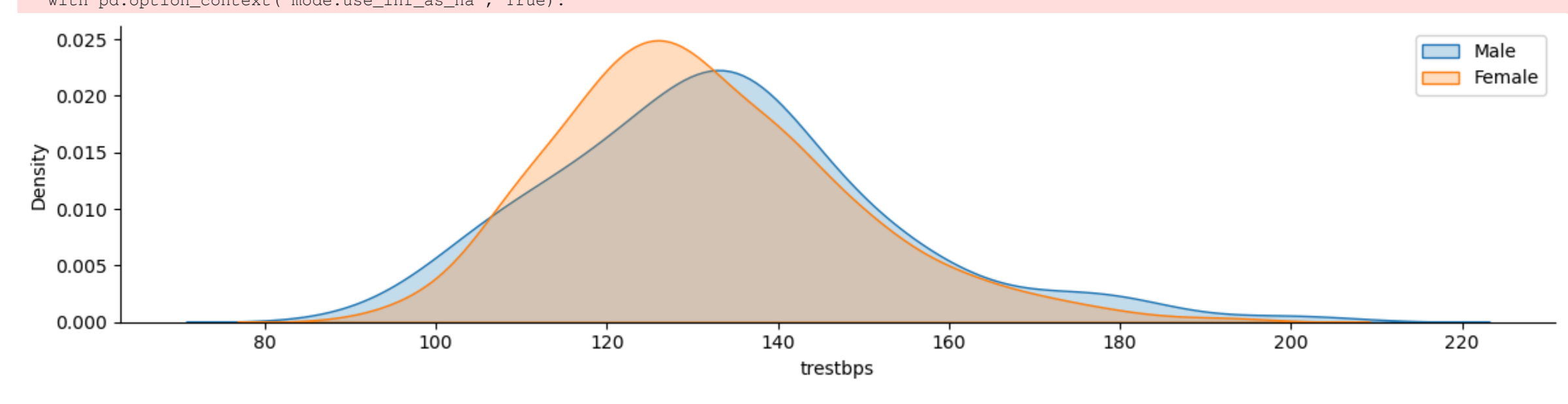
```
C:\Users\admin\anaconda5\Lib\site-packages\seaborn\axisgrid.py:848: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  func(*plot_args, **plot_kwargs)
C:\Users\admin\anaconda5\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\admin\anaconda5\Lib\site-packages\seaborn\axisgrid.py:848: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  func(*plot_args, **plot_kwargs)
C:\Users\admin\anaconda5\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



From this plot we can see that Female have lower Resting Blood Pressure as compare to Male. Female=around 120, Male=little less than 140

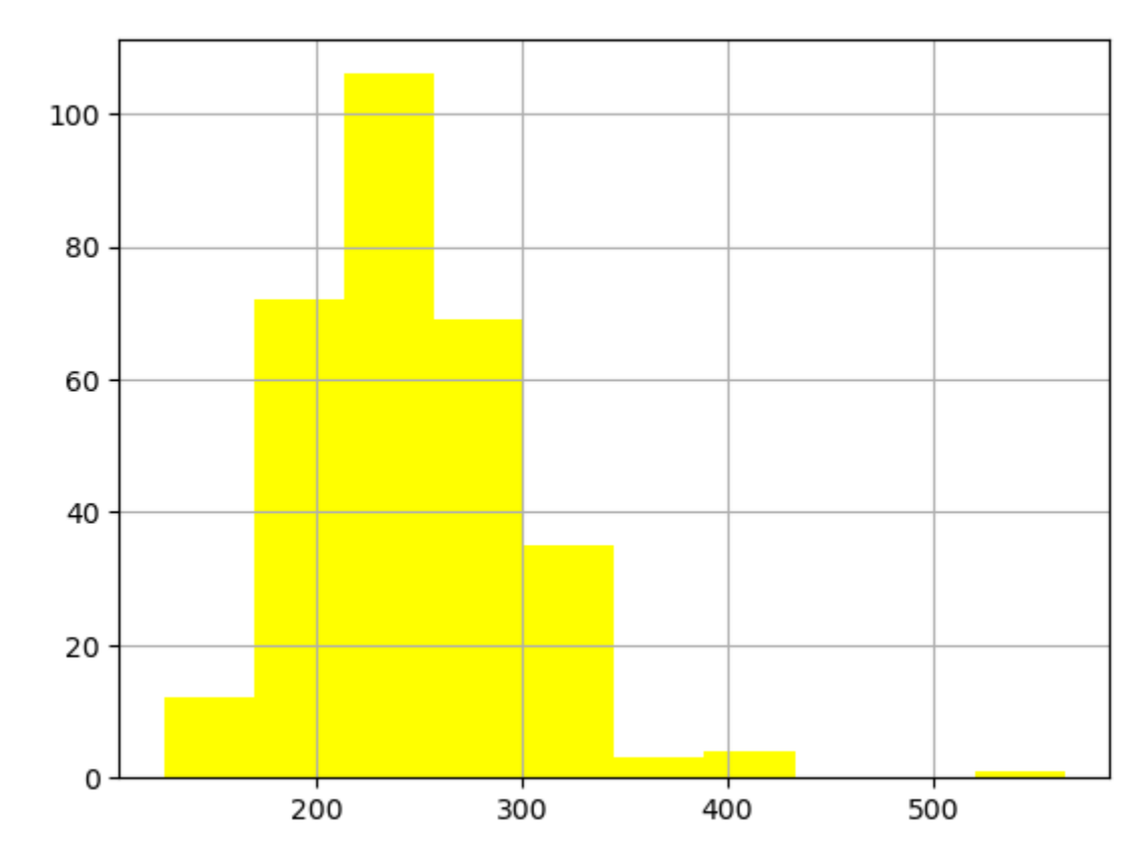
Disribution of Serum Cholestrol

In [107…
```python
data.columns
```

Out[107…
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [122…
```python
data['chol'].hist(color="yellow")
```

Out[122…
```
<Axes: >
```



Healthy Serum Cholesterol is less than 200 mg/dl

Continuous Variables

In [125…
```python
data.columns
```

Out[125…
```
Index(['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
       'exang', 'oldpeak', 'slope', 'ca', 'thal', 'target'],
      dtype='object')
```

In [126…
```python
categorical_values=[]
continuous_values=[]

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

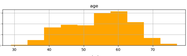
In [127…
```python
categorical_values
```

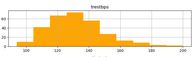
Out[127…
```
['sex', 'cp', 'fbs', 'restecg', 'exang', 'slope', 'ca', 'thal', 'target']
```

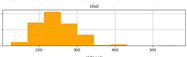
In [128…
```python
continuous_values
```

Out[128…
```
['age', 'trestbps', 'chol', 'thalach', 'oldpeak']
```

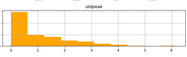
In [135…
```python
data.hist(continuous_values,figsize=(15,6),color="orange")
plt.tight_layout()
plt.show()
```


age
60
40
20
0
30
40
50
60
70


trestbps
60
40
20
0
100
120
140
160
180
200


chol
100
50
0
200
300
400
500


thalach
75
50
25
0
80
100
120
140
160
180
200


oldpeak
150
100
50
0
0
1
2
3
4
5
6