1 Load and Explore the Dataset

Load the dataset using pandas

In [9]:
```
import pandas as pd
df = pd.read_csv("Heart Disease UCI.csv")
```

Display first 10 rows

In [10]:
```
df.head(11)
```

Out[10]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	th
0	69	1	0	160	234	1	2	131	0	0.1	1	1	
1	69	0	0	140	239	0	0	151	0	1.8	0	2	
2	66	0	0	150	226	0	0	114	0	2.6	2	0	
3	65	1	0	138	282	1	2	174	0	1.4	1	1	
4	64	1	0	110	211	0	2	144	1	1.8	1	0	
5	64	1	0	170	227	0	2	155	0	0.6	1	0	
6	63	1	0	145	233	1	2	150	0	2.3	2	0	
7	61	1	0	134	234	0	0	145	0	2.6	1	2	
8	60	0	0	150	240	0	0	171	0	0.9	0	0	
9	59	1	0	178	270	0	2	145	0	4.2	2	0	
10	59	1	0	170	288	0	2	159	0	0.2	1	0	

Check data types

In [8]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 297 entries, 0 to 296
Data columns (total 14 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   age        297 non-null    int64  
 1   sex        297 non-null    int64  
 2   cp         297 non-null    int64  
 3   trestbps   297 non-null    int64  
 4   chol       297 non-null    int64  
 5   fbs        297 non-null    int64  
 6   restecg    297 non-null    int64  
 7   thalach    297 non-null    int64  
 8   exang      297 non-null    int64  
 9   oldpeak    297 non-null    float64
 10  slope      297 non-null    int64  
 11  ca         297 non-null    int64  
 12  thal       297 non-null    int64  
 13  condition  297 non-null    int64  
dtypes: float64(1), int64(13)
memory usage: 32.6 KB
```

Find missing values

In [11]:
```
df.isna().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
condition    0
dtype: int64
```

Display summary statistics

In [12]:
```
df.describe()
```

Out[12]:

	age	sex	cp	trestbps	chol	fbs	rest
count	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000	297.000(
mean	54.542088	0.676768	2.158249	131.693603	247.350168	0.144781	0.996(
std	9.049736	0.468500	0.964859	17.762806	51.997583	0.352474	0.994(
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000(
25%	48.000000	0.000000	2.000000	120.000000	211.000000	0.000000	0.000(
50%	56.000000	1.000000	2.000000	130.000000	243.000000	0.000000	1.000(
75%	61.000000	1.000000	3.000000	140.000000	276.000000	0.000000	2.000(
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000(

In [ ]:

2 Gender Distribution Analysis

Count number of males and females

In [15]:
```
print(f"number of Male = {len(df[df["sex"] == 1])}")
print(f"number of Female = {len(df[df["sex"] == 0])}")
```

```
number of Male = 201
number of Female = 96
```

Calculate percentage distribution using NumPy

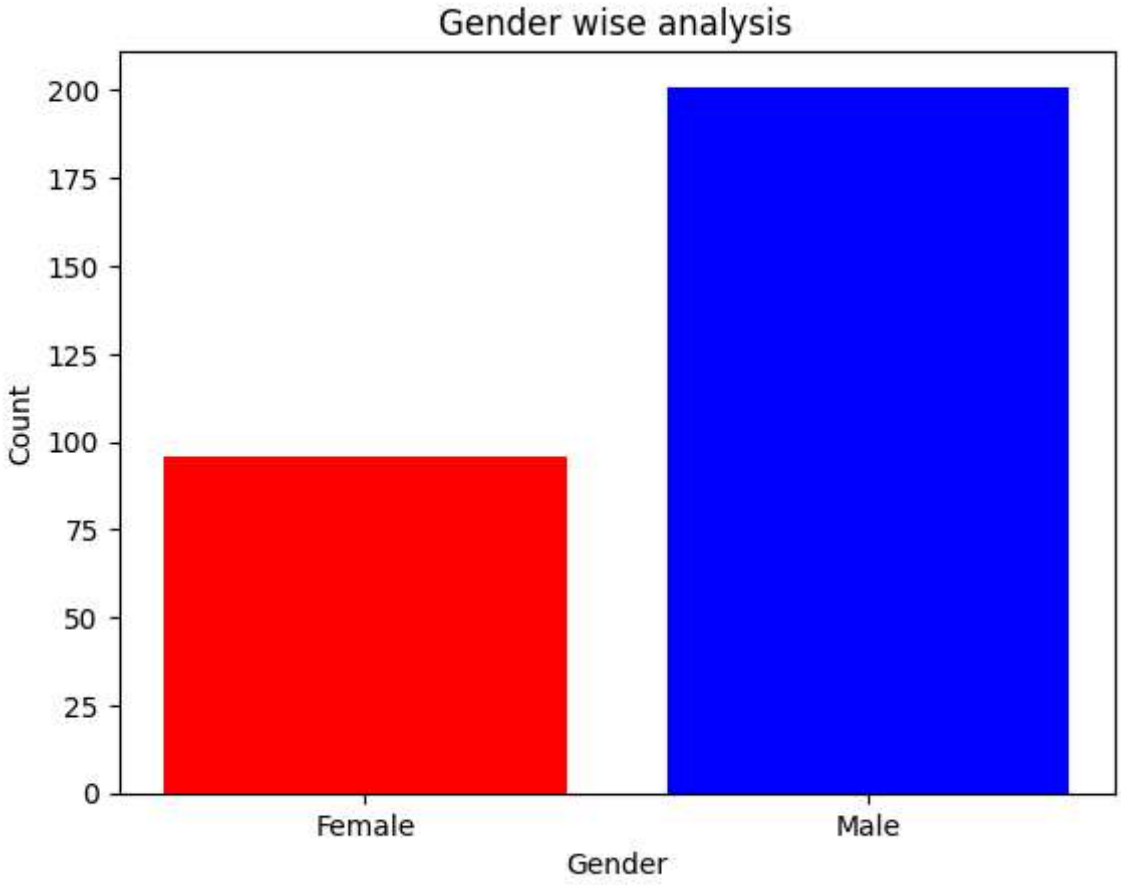
In [32]:
```
import numpy as np
total = len(df)
male_percentage = (len(df[df['sex'] == 1]) / total) * 100.0
female_percentage = (len(df[df['sex'] == 0]) / total) * 100.0
print(f"male percentage = {male_percentage}%")
print(f"female percentage = {female_percentage}%")
```

```
male percentage = 67.67676767676768%
female percentage = 32.323232323232325%
```

Plot a bar chart using Matplotlib

In [24]:
```
import matplotlib.pyplot as plt
gender = df.groupby("sex").count()
print(gender)

plt.bar(['Female','Male'],gender.age,color = ['r','b'])
plt.xlabel("Gender")
plt.ylabel("Count")
plt.title("Gender wise analysis")
plt.show()
```

```
        age   cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slope  \
sex
0        96   96        96    96   96       96       96     96       96     96
1       201  201       201   201  201      201      201    201      201    201

        ca  thal  condition
sex
0       96    96         96
1      201   201        201
```



In [ ]:

3 Age Analysis

Find:

Minimum age

Maximum age

Mean age

Median age

In [33]:
```
df['age'].min()
```

Out[33]: 29

In [34]:
```
df['age'].max()
```

Out[34]: 77

In [35]:
```
df['age'].mean()
```

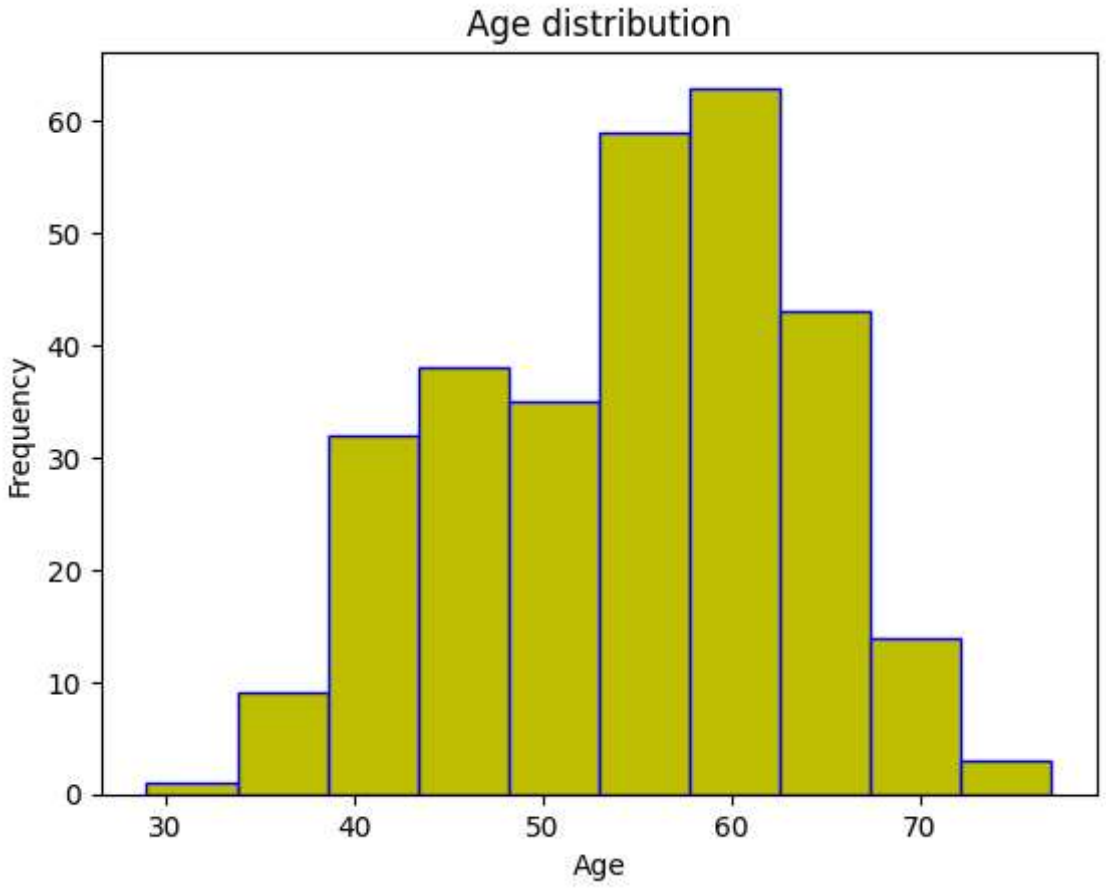
Out[35]: 54.54208754208754

In [37]:
```
df['age'].median()
```

Out[37]: 56.0

Plot histogram of age distribution

In [44]:
```
plt.hist(df['age'],edgecolor = 'b',color = 'y')
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.title("Age distribution")
plt.show()
```



In [ ]:

4 Target Variable Analysis

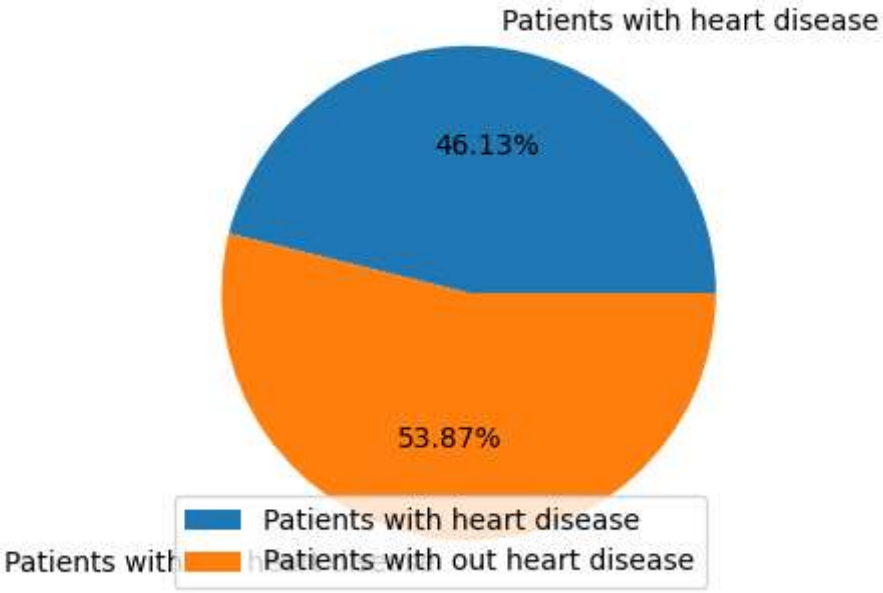
Count number of patients with and without heart disease

In [47]:
```
print(f"Number of patiets with out herat disease  = {len(df[df['condition'] == 0])}")
print(f"Number of patiets with herat disease  = {len(df[df['condition'] == 1])}")
```

```
Number of patiets with out herat disease  = 160
Number of patiets with herat disease  = 137
```

Plot pie chart

In [54]:
```
values = [len(df[df['condition'] == 1]),len(df[df['condition'] == 0])]
labels = ['Patients with heart disease','Patients with out heart disease']
plt.figure(figsize = (4,4))
plt.pie(values,labels = labels,autopct = "%2.2f%%")
plt.legend(loc = 3)
plt.show()
```



Calculate disease percentage

In [55]:
```
total = len(df)
disease = len(df[df['condition'] == 1])
print(f"disease percentage = {(disease / total)*100.0}%")
```

```
disease percentage = 46.12794612794613%
```

5 . Correlation Between Age and Cholesterol

In [ ]:
```

```

Calculate correlation using df.corr()

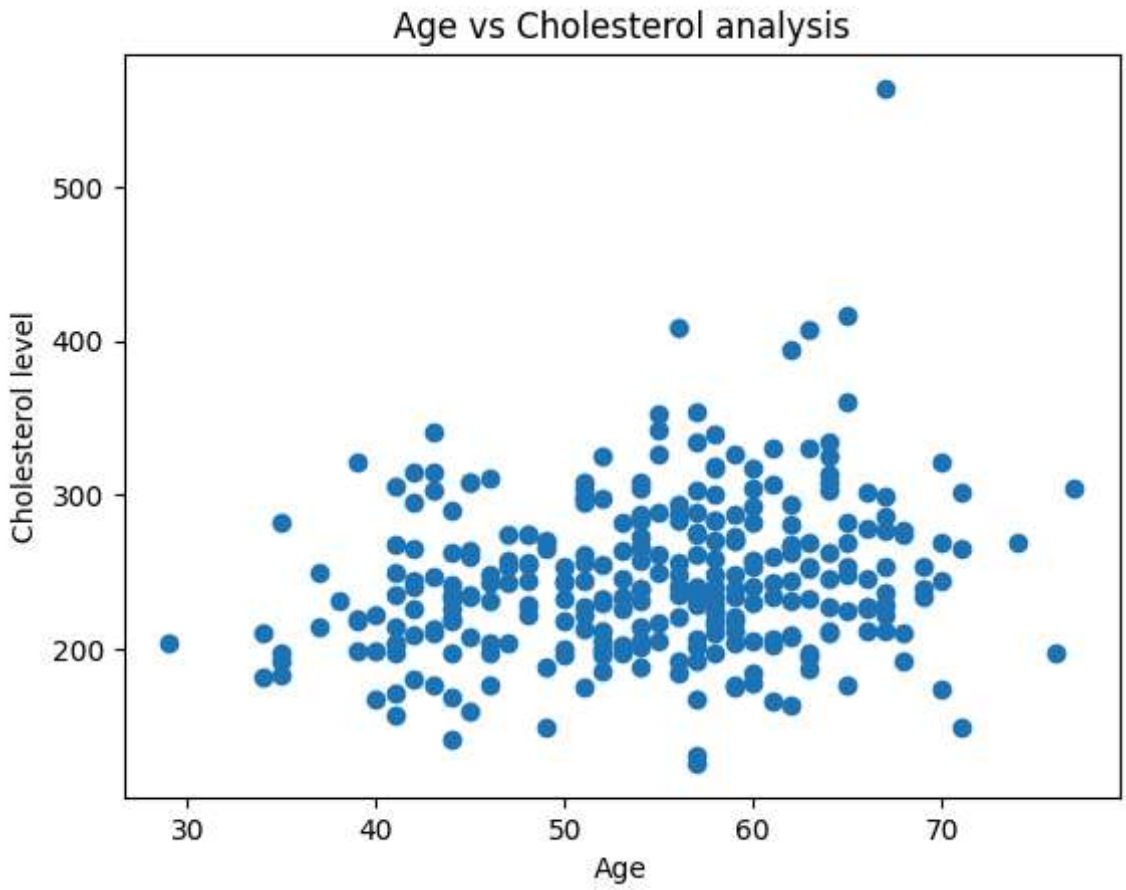
In [72]:
```
df.corr()
```

Out[72]:

	age	sex	cp	trestbps	chol	fbs	restecg	
age	1.000000	-0.092399	0.110471	0.290476	0.202644	0.132062	0.149917	-(
sex	-0.092399	1.000000	0.008908	-0.066340	-0.198089	0.038850	0.033897	-(
cp	0.110471	0.008908	1.000000	-0.036980	0.072088	-0.057663	0.063905	-(
trestbps	0.290476	-0.066340	-0.036980	1.000000	0.131536	0.180860	0.149242	-(
chol	0.202644	-0.198089	0.072088	0.131536	1.000000	0.012708	0.165046	-(
fbs	0.132062	0.038850	-0.057663	0.180860	0.012708	1.000000	0.068831	-(
restecg	0.149917	0.033897	0.063905	0.149242	0.165046	0.068831	1.000000	-(
thalach	-0.394563	-0.060496	-0.339308	-0.049108	-0.000075	-0.007842	-0.072290	
exang	0.096489	0.143581	0.377525	0.066691	0.059339	-0.000893	0.081874	-(
oldpeak	0.197123	0.106567	0.203244	0.191243	0.038596	0.008311	0.113726	-(
slope	0.159405	0.033345	0.151079	0.121172	-0.009215	0.047819	0.135141	-(
ca	0.362210	0.091925	0.235644	0.097954	0.115945	0.152086	0.129021	-(
thal	0.120795	0.370556	0.266275	0.130612	0.023441	0.051038	0.013612	-(
condition	0.227075	0.278467	0.408945	0.153490	0.080285	0.003167	0.166343	-(

Plot scatter plot (Age vs Cholesterol)

In [74]:
```
plt.scatter(df['age'],df['chol'])
plt.xlabel("Age")
plt.ylabel("Cholesterol level")
plt.title("Age vs Cholesterol analysis")
plt.show()
```



Interpret relationship

Cholesterol levels tend to increase with age, indicating a positive correlation between age and cholesterol

6. Chest Pain Type vs Disease

Group by cp and calculate disease rate

In [5]:
```
cpy = df.groupby('cp',as_index = False)['condition'].mean()*100
cpy.rename(columns = {'condition':'disease_rate'},inplace = True)
cpy
```

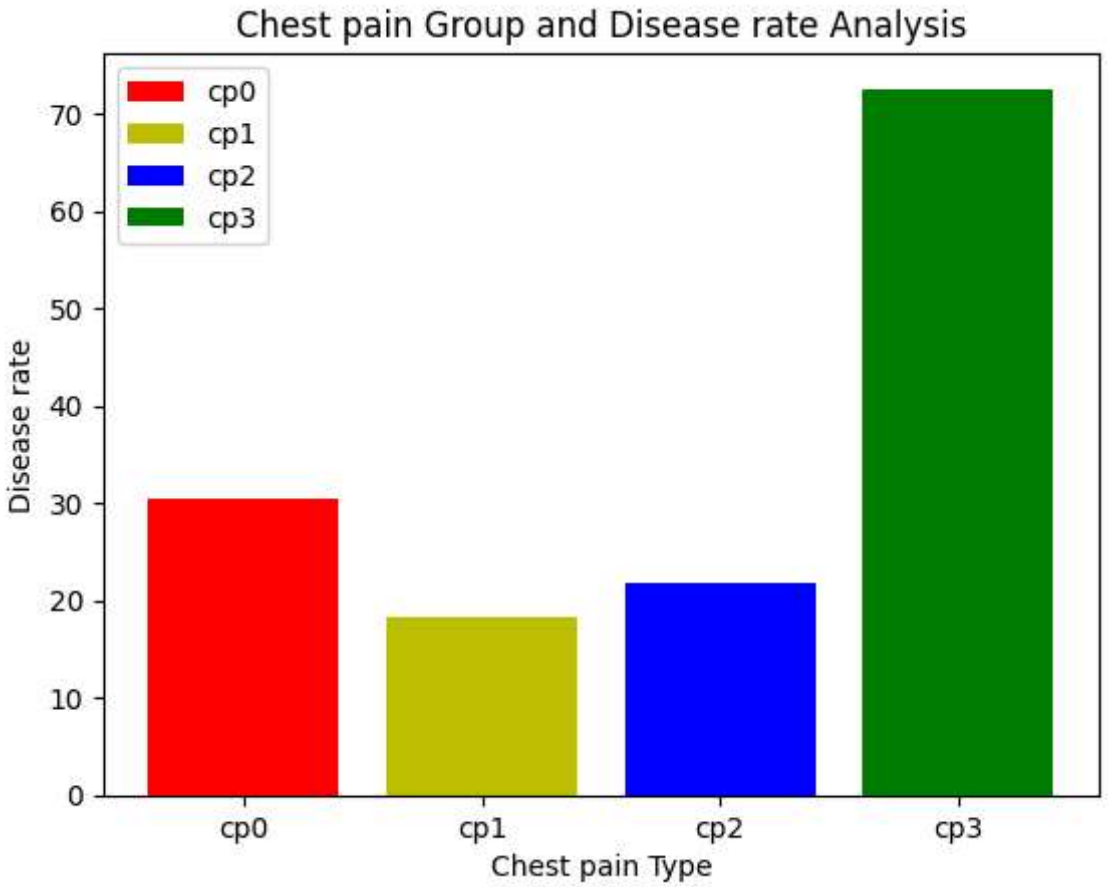
Out[5]:

	cp	disease_rate
0	0	30.434783
1	100	18.367347
2	200	21.686747
3	300	72.535211

Plot grouped bar chart

In [19]:
```
import matplotlib.pyplot as plt
plt.bar(['cp0','cp1','cp2','cp3'],cpy['disease_rate'],label = ['cp0','cp1','cp2',
plt.xlabel("Chest pain Type")
plt.ylabel("Disease rate")
```

```
plt.legend()
plt.title("Chest pain Group and Disease rate Analysis")
plt.show()
```



Identify which chest pain type is most risky

In [ ]:

from the graph chest pain type 3 is the most risky one with disease rate 72.535211%

7 .Average Cholesterol by Gender

Group by sex

Calculate mean cholesterol

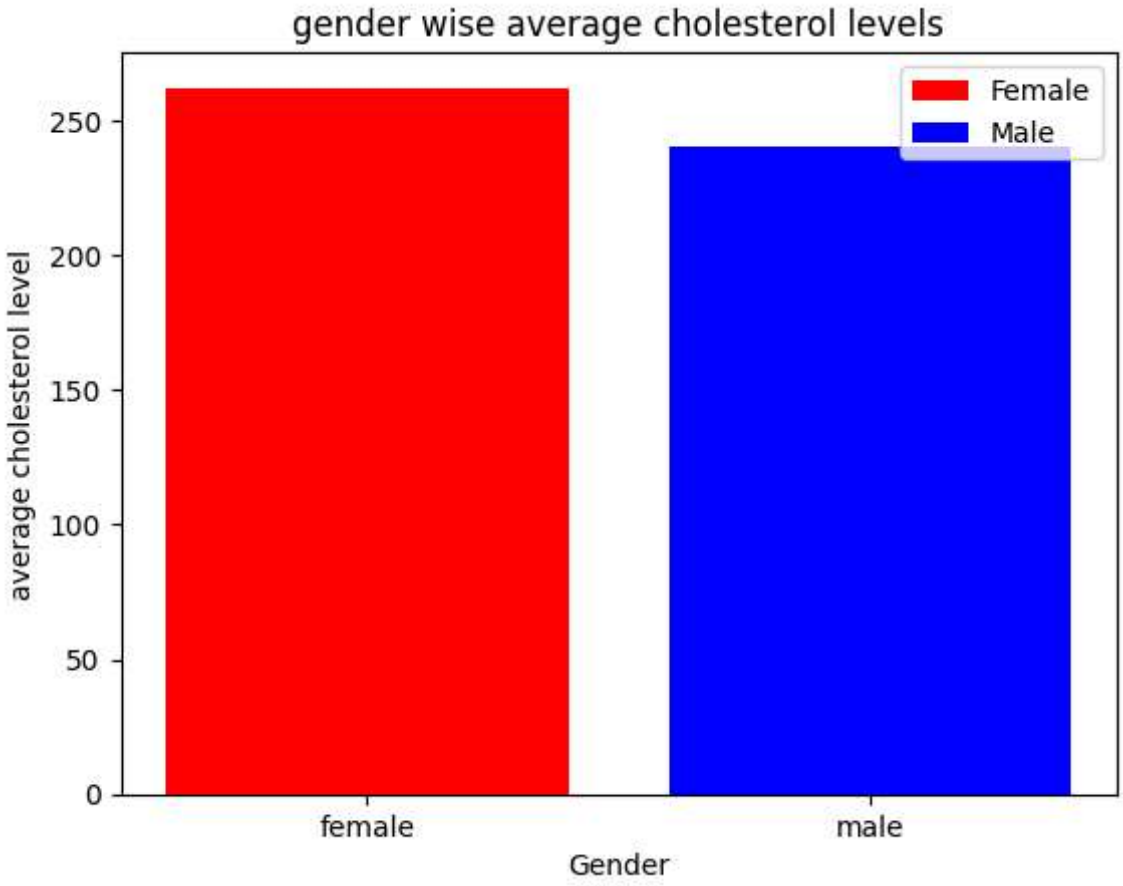
Visualize using bar plot

In [85]:
```
s = df.groupby('sex',as_index = False)['chol'].mean()
s
```

Out[85]:

	sex	chol
0	0	262.229167
1	1	240.243781

In [90]:
```
plt.bar(["female",'male'],s['chol'],label = ["Female","Male"],color =['r','b'] )
plt.xlabel("Gender")
plt.ylabel("average cholesterol level")
plt.legend()
plt.title("gender wise average cholesterol levels")

plt.show()
```



8.Resting Blood Pressure Analysis

Find:

Average BP

Patients with BP > 140

In [93]:
```
avg_bp = df['trestbps'].mean()
avg_bp
```

Out[93]: 131.69360269360268

In [94]:
```
df[df['trestbps'] > 140]
```

Out[94]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	tl
0	69	1	0	160	234	1	2	131	0	0.1	1	1	
2	66	0	0	150	226	0	0	114	0	2.6	2	0	
5	64	1	0	170	227	0	2	155	0	0.6	1	0	
6	63	1	0	145	233	1	2	150	0	2.3	2	0	
8	60	0	0	150	240	0	0	171	0	0.9	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
263	50	1	3	150	243	0	2	128	0	2.6	1	0	
264	50	1	3	144	200	0	2	126	1	0.9	1	0	
277	45	1	3	142	309	0	2	147	1	0.0	1	3	
285	43	1	3	150	247	0	0	171	0	1.5	0	0	
292	40	1	3	152	223	0	0	181	0	0.0	0	0	

66 rows × 14 columns

Compare disease presence in high BP group

In [145]:
```
h_bp = df[df['trestbps']>140][df['condition'] == 1]
#print("Number of patients with high bp and herart disease  = ",len(h_bp))
h_bp
```

```
C:\Users\Administrator\AppData\Local\Temp\ipykernel_7568\3504453456.py:1: UserWarni
ng: Boolean Series key will be reindexed to match DataFrame index.
  h_bp = df[df['trestbps']>140][df['condition'] == 1]
```

Out[145]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	tl
10	59	1	0	170	288	0	2	159	0	0.2	1	0	
11	59	1	0	160	273	0	2	125	0	0.0	0	0	
26	66	1	1	160	246	0	0	120	1	0.0	1	3	
35	57	1	1	154	232	0	2	164	0	0.0	0	1	
45	54	1	1	192	283	0	2	195	0	0.0	0	1	
74	70	1	2	160	269	0	0	112	1	2.9	1	1	
76	68	1	2	180	274	1	2	150	1	1.6	1	0	
80	67	1	2	152	212	0	2	150	0	0.8	1	0	
136	46	1	2	150	231	0	0	147	0	3.6	1	0	
157	70	1	3	145	174	0	0	125	1	2.6	2	0	
159	68	1	3	144	193	1	0	141	0	3.4	1	2	
160	67	1	3	160	286	0	2	108	1	1.5	1	3	
167	66	0	3	178	228	1	0	165	1	1.0	1	2	
170	65	0	3	150	225	0	2	114	0	1.0	1	3	
175	64	1	3	145	212	0	2	132	0	2.0	1	2	
180	63	0	3	150	407	0	2	154	0	4.0	1	3	
186	62	0	3	160	164	0	2	145	0	6.2	2	3	
190	62	0	3	150	244	0	0	154	1	1.4	1	0	
193	61	0	3	145	307	0	2	146	1	1.0	1	0	
197	61	1	3	148	203	0	0	161	0	0.0	0	1	
199	60	1	3	145	282	0	2	142	1	2.8	1	2	
201	60	0	3	150	258	0	2	157	0	2.6	1	2	
205	60	0	3	158	305	0	2	161	0	0.0	0	0	
206	59	1	3	170	326	0	2	140	1	3.4	2	0	
209	59	0	3	174	249	0	0	143	1	0.0	1	0	
210	59	1	3	164	176	1	2	90	0	1.0	1	2	
217	58	1	3	146	218	0	0	105	0	2.0	1	1	
219	58	0	3	170	225	1	2	146	1	2.8	1	2	
220	58	1	3	150	270	0	2	111	1	0.8	0	0	
224	57	1	3	150	276	0	2	112	1	0.6	1	1	
225	57	1	3	165	289	1	2	124	0	1.0	1	3	
226	57	1	3	152	274	0	0	88	1	1.2	1	1	
234	56	0	3	200	288	1	2	133	1	4.0	2	2	

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	tl
241	55	1	3	160	289	0	2	145	1	0.8	1	1	
242	55	0	3	180	327	0	1	117	1	3.4	1	0	
263	50	1	3	150	243	0	2	128	0	2.6	1	0	
264	50	1	3	144	200	0	2	126	1	0.9	1	0	
277	45	1	3	142	309	0	2	147	1	0.0	1	3	
292	40	1	3	152	223	0	0	181	0	0.0	0	0	

9 . Maximum Heart Rate vs Disease

Compare average thalach for:

Disease patients

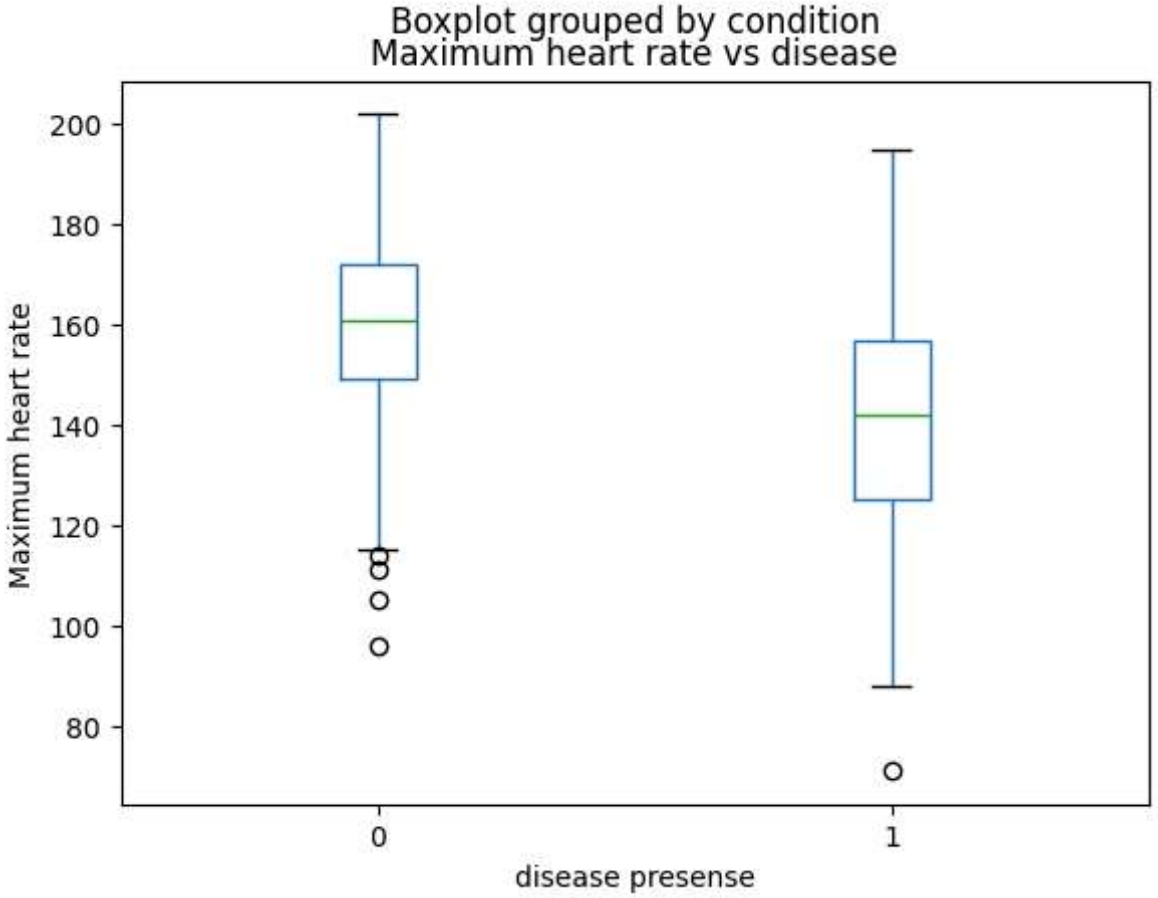
Non-disease patients

In [103]:
```
print("avg thalach for disease = ",df[df['condition'] == 1]['thalach'].mean())
print("avg thalach for non disease  = ",df[df['condition'] == 0]['thalach'].mean()
```

```
avg thalach for disease =  139.1094890510949
avg thalach for non disease  =  158.58125
```

Plot boxplot

In [142]:
```
df.boxplot(column  = 'thalach',by = 'condition',grid = False)
plt.title("Maximum heart rate vs disease")
plt.xlabel("disease presense")
plt.ylabel("Maximum heart rate")
plt.show()
```



10.Exercise Induced Angina Impact

Calculate disease percentage in:

exang = 1

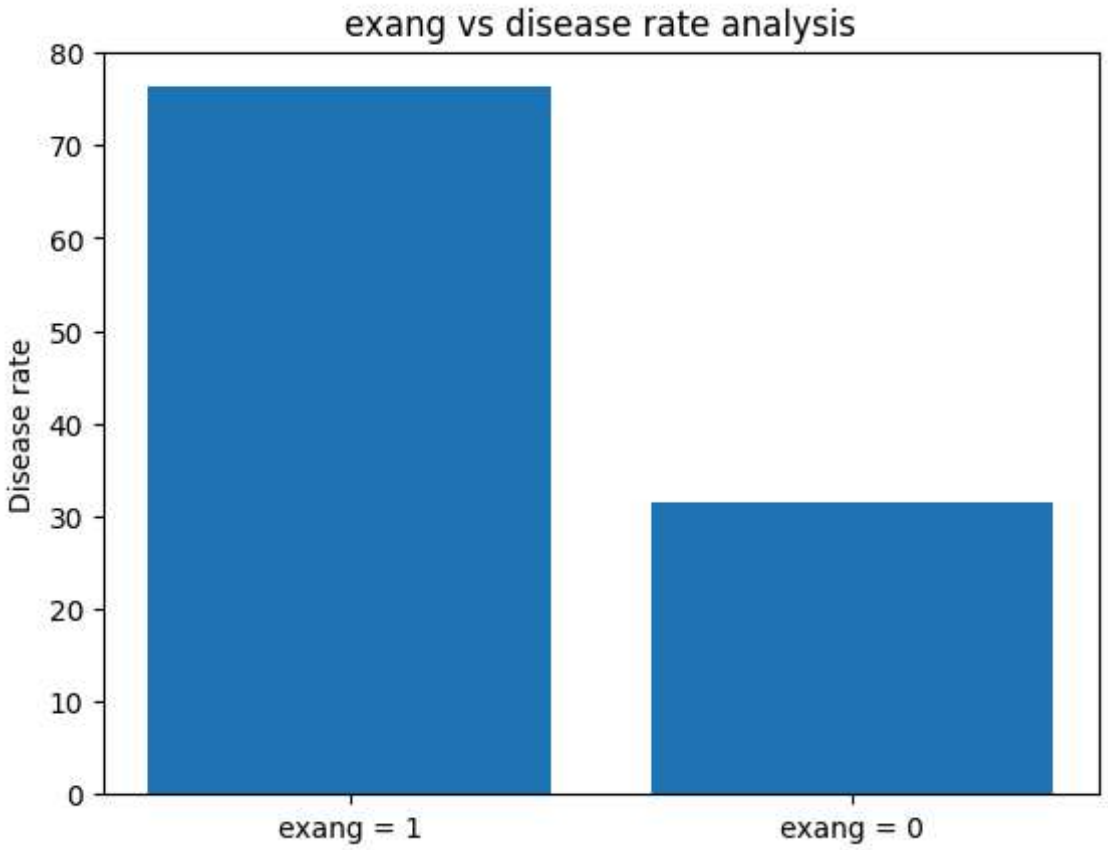
exang = 0

Visualize using bar chart

In [20]:
```
print("disease percentage in exang = 1 is",df[df['exang'] == 1]['condition'].mean(
print("disease percentage in exang = 0 is",df[df['exang'] == 0]['condition'].mean(
```

```
disease percentage in exang = 1 is 76.28865979381443
disease percentage in exang = 0 is 31.5
```

In [21]:
```
a = [df[df['exang'] == 1]['condition'].mean()*100,df[df['exang'] == 0]['condition'
plt.bar(["exang = 1","exang = 0"],a)

plt.ylabel("Disease rate")
plt.title("exang vs disease rate analysis")
plt.show()
```



11. ST Depression (oldpeak) Analysis

Calculate mean oldpeak by target

Plot histogram for both classes

Identify trend

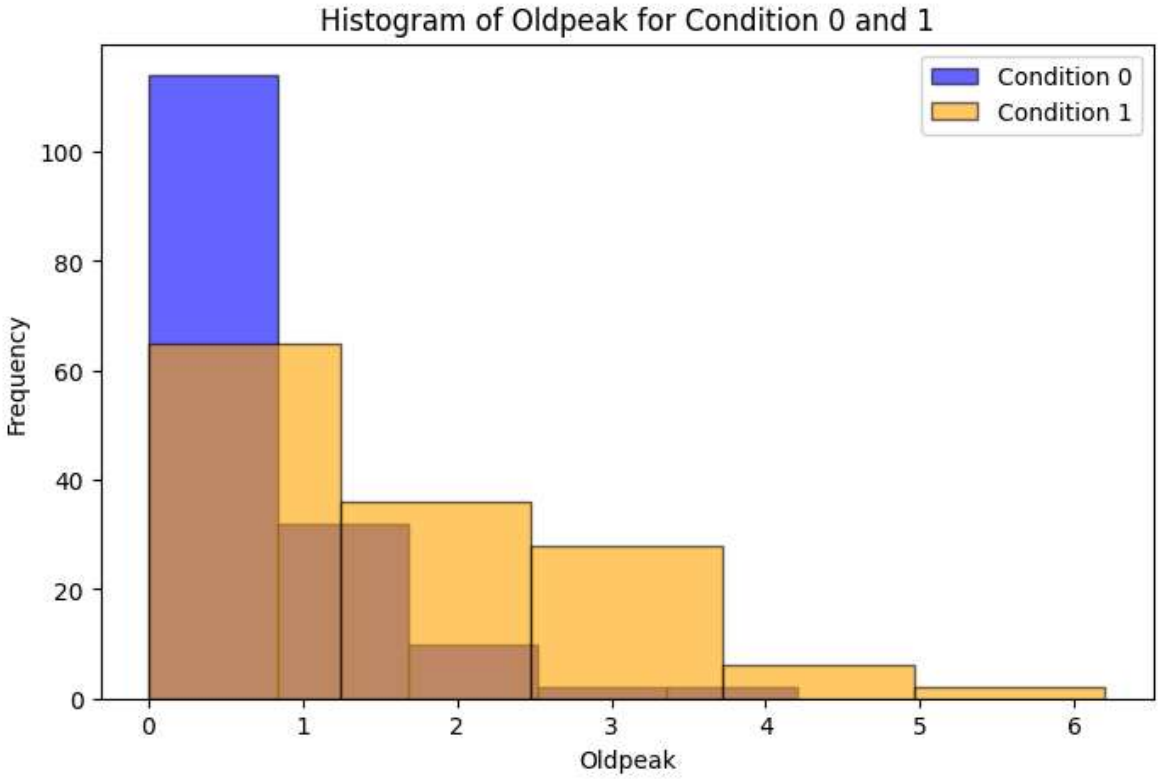
In [115]:
```
mean_oldpeak_by_target = df.groupby('condition',as_index = False)['oldpeak'].mean(
mean_oldpeak_by_target
```

Out[115]:

	condition	oldpeak
0	0	0.598750
1	1	1.589051

In [117]:
```
oldpeak_cond0 = df[df['condition'] == 0]['oldpeak']
oldpeak_cond1 = df[df['condition'] == 1]['oldpeak']




plt.figure(figsize=(8, 5))
plt.hist(oldpeak_cond0, bins=5, alpha=0.6, label='Condition 0', color='blue', edge
plt.hist(oldpeak_cond1, bins=5, alpha=0.6, label='Condition 1', color='orange', ec


plt.xlabel('Oldpeak')
```

```
plt.ylabel('Frequency')
plt.title('Histogram of Oldpeak for Condition 0 and 1')
plt.legend()



plt.show()
```



----------Trend---------- Condition 0 → Higher average oldpeak and wider spread. Condition 1 → Lower average oldpeak and more concentrated distribution.

12. Number of Major Vessels (ca) Impact

Group by ca

Calculate disease probability

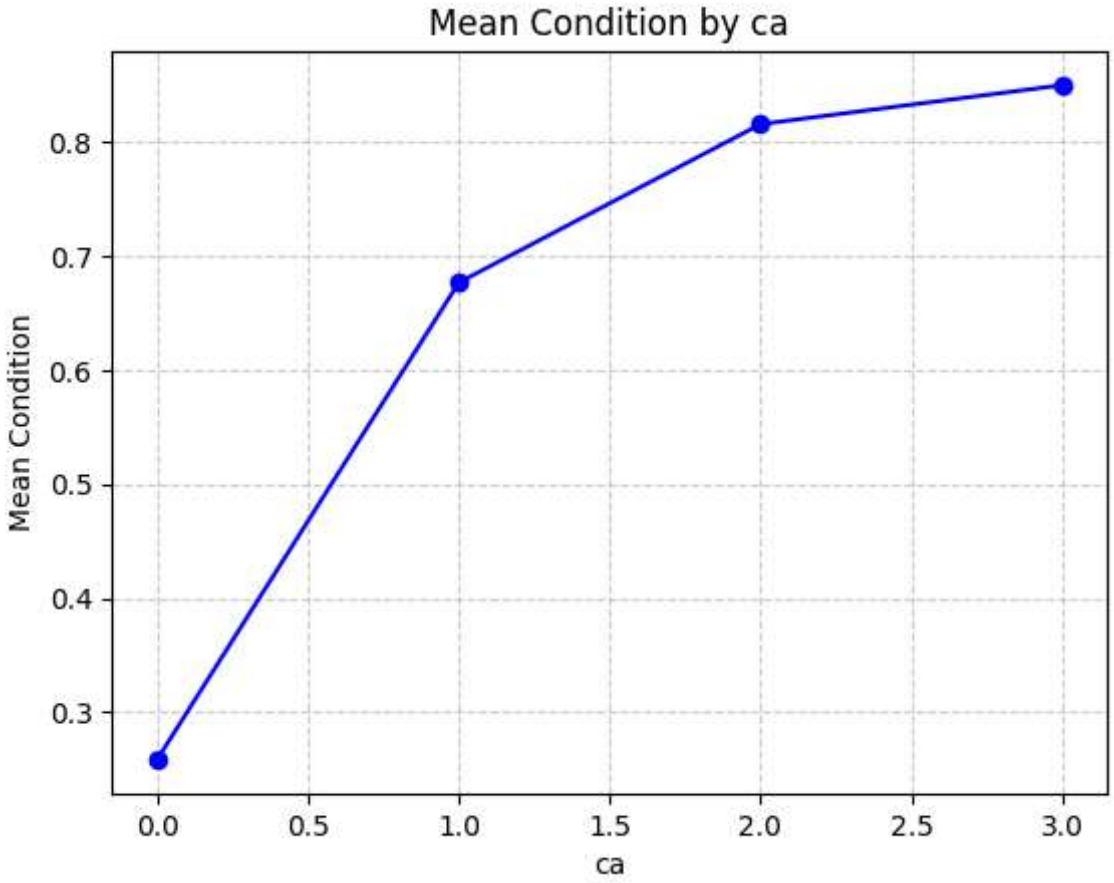
Plot line chart

In [22]:
```
ca = df.groupby('ca')['condition'].mean()*100
ca
```

Out[22]:
```
ca
0    25.862069
1    67.692308
2    81.578947
3    85.000000
Name: condition, dtype: float64
```

In [121]:
```
plt.plot(ca.index, ca.values, marker='o', linestyle='-', color='blue')

# Labels and title
plt.xlabel('ca')
```

```
plt.ylabel('Mean Condition')
plt.title('Mean Condition by ca')
plt.grid(True, linestyle='--', alpha=0.6)
```



13. Thalassemia vs Disease

Cross-tabulate thal and target

Convert to percentage

Plot stacked bar chart

In [127]:
```
cross_tab = pd.crosstab(df['thal'],df['condition'])
cross_tab
```

Out[127]:

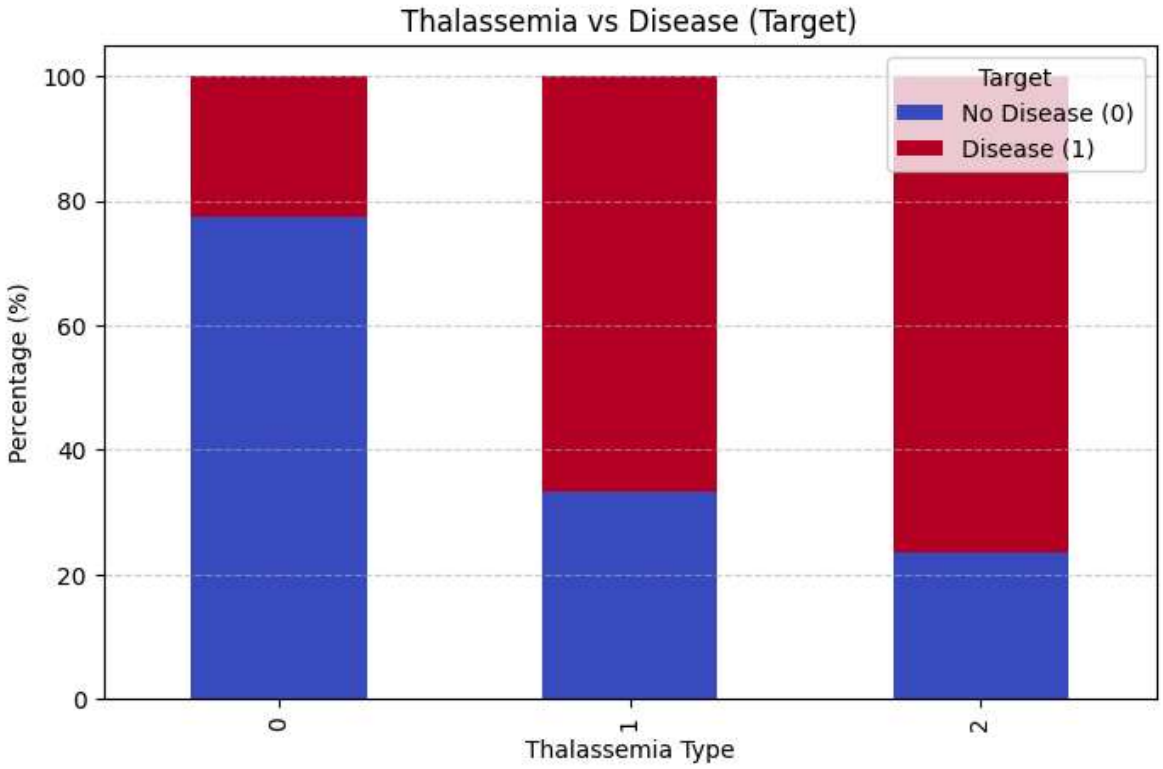
condition	0	1
thal		
0	127	37
1	6	12
2	27	88

In [126]:
```
cross_tab_percent = cross_tab.div(cross_tab.sum(axis=1), axis=0) * 100
cross_tab_percent
```

Out[126]:

condition	0	1
thal		
0	77.439024	22.560976
1	33.333333	66.666667
2	23.478261	76.521739

In [128]:
```
cross_tab_percent.plot(kind='bar', stacked=True, figsize=(8, 5), colormap='coolwar


plt.xlabel('Thalassemia Type')
plt.ylabel('Percentage (%)')
plt.title('Thalassemia vs Disease (Target)')
plt.legend(title='Target', labels=['No Disease (0)', 'Disease (1)'])
plt.grid(axis='y', linestyle='--', alpha=0.6)

# Show plot
plt.show()
```



14. Multi-Factor Risk Analysis

Find patients with:

Age > 50

Cholesterol > 240

BP > 140

Calculate percentage having disease

Use NumPy filtering

In [131]:
```
p = df[(df["age"] > 50) & (df['chol'] > 240) & (df['trestbps']>140)]
p
```

Out[131]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	tl
9	59	1	0	178	270	0	2	145	0	4.2	2	0	
10	59	1	0	170	288	0	2	159	0	0.2	1	0	
11	59	1	0	160	273	0	2	125	0	0.0	0	0	
13	58	0	0	150	283	1	2	162	0	1.0	0	0	
16	52	1	0	152	298	1	0	178	0	1.2	1	0	
24	71	0	1	160	302	0	0	162	0	0.4	0	2	
25	70	1	1	156	245	0	2	143	0	0.0	0	0	
26	66	1	1	160	246	0	0	120	1	0.0	1	3	
45	54	1	1	192	283	0	2	195	0	0.0	0	1	
74	70	1	2	160	269	0	0	112	1	2.9	1	1	
76	68	1	2	180	274	1	2	150	1	1.6	1	0	
81	67	0	2	152	277	0	0	172	0	0.0	0	1	
82	66	0	2	146	278	0	2	152	0	0.0	1	1	
84	65	0	2	155	269	0	0	148	0	0.8	0	0	
85	65	0	2	160	360	0	2	151	0	0.8	0	0	
92	61	1	2	150	243	1	0	137	1	1.0	1	0	
160	67	1	3	160	286	0	2	108	1	1.5	1	3	
178	64	0	3	180	325	0	0	154	1	0.0	0	0	
180	63	0	3	150	407	0	2	154	0	4.0	1	3	
190	62	0	3	150	244	0	0	154	1	1.4	1	0	
193	61	0	3	145	307	0	2	146	1	1.0	1	0	
199	60	1	3	145	282	0	2	142	1	2.8	1	2	
201	60	0	3	150	258	0	2	157	0	2.6	1	2	
205	60	0	3	158	305	0	2	161	0	0.0	0	0	
206	59	1	3	170	326	0	2	140	1	3.4	2	0	
209	59	0	3	174	249	0	0	143	1	0.0	1	0	
220	58	1	3	150	270	0	2	111	1	0.8	0	0	
224	57	1	3	150	276	0	2	112	1	0.6	1	1	
225	57	1	3	165	289	1	2	124	0	1.0	1	3	
226	57	1	3	152	274	0	0	88	1	1.2	1	1	
234	56	0	3	200	288	1	2	133	1	4.0	2	2	
241	55	1	3	160	289	0	2	145	1	0.8	1	1	
242	55	0	3	180	327	0	1	117	1	3.4	1	0	

In [133]:
```
print(f"percentage with disease = {len(p[p['condition'] == 1])/len(p) * 100.0}%")
```

```
percentage with disease = 66.66666666666666%
```

15. Create Risk Score (Custom Analysis)

Create new column:

risk_score = (chol/200) + (trestbps/120) + (oldpeak)

Classify patients as:

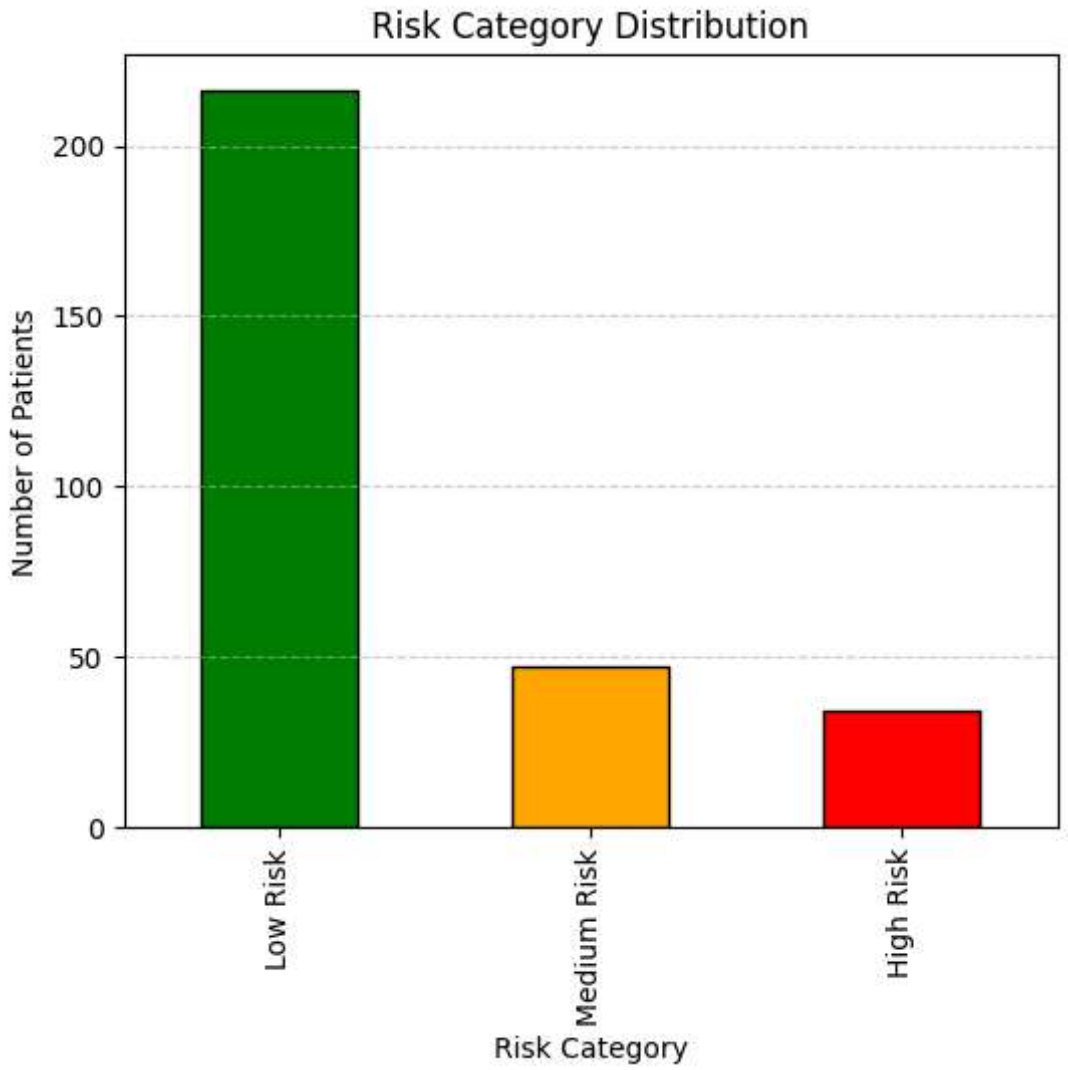
Low Risk

Medium Risk

High Risk

Visualize distribution

In [135]:
```
df['risk_score'] = (df['chol'] / 200) + (df['trestbps'] / 120) + df['oldpeak']


def classify_risk(score):
    if score < 4:
        return 'Low Risk'
    elif score < 5:
        return 'Medium Risk'
    else:
        return 'High Risk'

df['risk_category'] = df['risk_score'].apply(classify_risk)




plt.figure(figsize=(6, 5))
df['risk_category'].value_counts().plot(kind='bar', color=['green', 'orange', 'red

plt.title('Risk Category Distribution')
plt.xlabel('Risk Category')
plt.ylabel('Number of Patients')
plt.grid(axis='y', linestyle='--', alpha=0.6)
plt.show()
```



Expected Insights

Does cholesterol strongly impact heart disease?

In [31]:
```
df['chol'].corr(df['condition'])
```

Out[31]: 0.08028475098000244

Cholesterol shows only a weak linear correlation with heart disease

Is male population more vulnerable?

In [37]:
```
df['sex'].corr(df['condition'])
```

Out[37]: 0.278466696653796

male show heigher vulnurability but the relation ship is moderate

Does exercise-induced angina significantly increase risk?

In [39]:
```
print("Exercise Induced angina correlation :",df['exang'].corr(df['condition']))
```

```
Exercise Induced angina correlation : 0.42135549045645343
```

Exersie induced angina is a significant risk indicator in the dataset

Which feature has strongest correlation with disease?

In [41]:
```
correlation = df.corr(numeric_only = True)['condition'].sort_values(ascending = Fa
correlation
```

Out[41]:
```
condition    1.000000
thal         0.520516
ca           0.463189
oldpeak      0.424052
exang        0.421355
cp           0.408945
slope        0.333049
sex          0.278467
age          0.227075
restecg      0.166343
trestbps     0.153490
chol         0.080285
fbs          0.003167
thalach     -0.423817
Name: condition, dtype: float64
```

from the data the herat rate have heigher correlation with disease