Modules

In [49]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

1.Load and Explore the Dataset

In [50]:
```
#Load the dataset using pandas
df=pd.read_csv("Heart Disease UCI.csv")
df
```

Out[50]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	condition
0	69	1	0	160	234	1	2	131	0	0.1	1	1	0	0
1	69	0	0	140	239	0	0	151	0	1.8	0	2	0	0
2	66	0	0	150	226	0	0	114	0	2.6	2	0	0	0
3	65	1	0	138	282	1	2	174	0	1.4	1	1	0	1
4	64	1	0	110	211	0	2	144	1	1.8	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
292	40	1	3	152	223	0	0	181	0	0.0	0	0	2	1
293	39	1	3	118	219	0	0	140	0	1.2	1	0	2	1
294	35	1	3	120	198	0	0	130	1	1.6	1	0	2	1
295	35	0	3	138	183	0	0	182	0	1.4	0	0	0	0
296	35	1	3	126	282	0	2	156	1	0.0	0	0	2	1

297 rows × 14 columns

In [51]:
```
#Display first 10 rows
df.head(10)
```

Out[51]:

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

In [52]:
```
#Check data types
df.dtypes
```

Out[52]:
```
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
condition     int64
dtype: object
```

In [53]:
```
#Find missing values
print(df.isnull().sum())
```

```
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
```

In [54]:
```
#Display summary statistics
print(df.describe())
```

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

          restecg     thalach       exang     oldpeak       slope          ca  \
count  297.000000  297.000000  297.000000  297.000000  297.000000  297.000000
mean     0.996633  149.599327    0.326599    1.055556    0.602694    0.676768
std      0.994914   22.941562    0.469761    1.166123    0.618187    0.938965
min      0.000000   71.000000    0.000000    0.000000    0.000000    0.000000
25%      0.000000  133.000000    0.000000    0.000000    0.000000    0.000000
50%      1.000000  153.000000    0.000000    0.800000    1.000000    0.000000
75%      2.000000  166.000000    1.000000    1.600000    1.000000    1.000000
max      2.000000  202.000000    1.000000    6.200000    2.000000    3.000000

             thal   condition
count  297.000000  297.000000
mean     0.835017    0.461279
std      0.956690    0.499340
min      0.000000    0.000000
25%      0.000000    0.000000
50%      0.000000    0.000000
75%      2.000000    1.000000
max      2.000000    1.000000
```

2.Gender Distribution Analysis

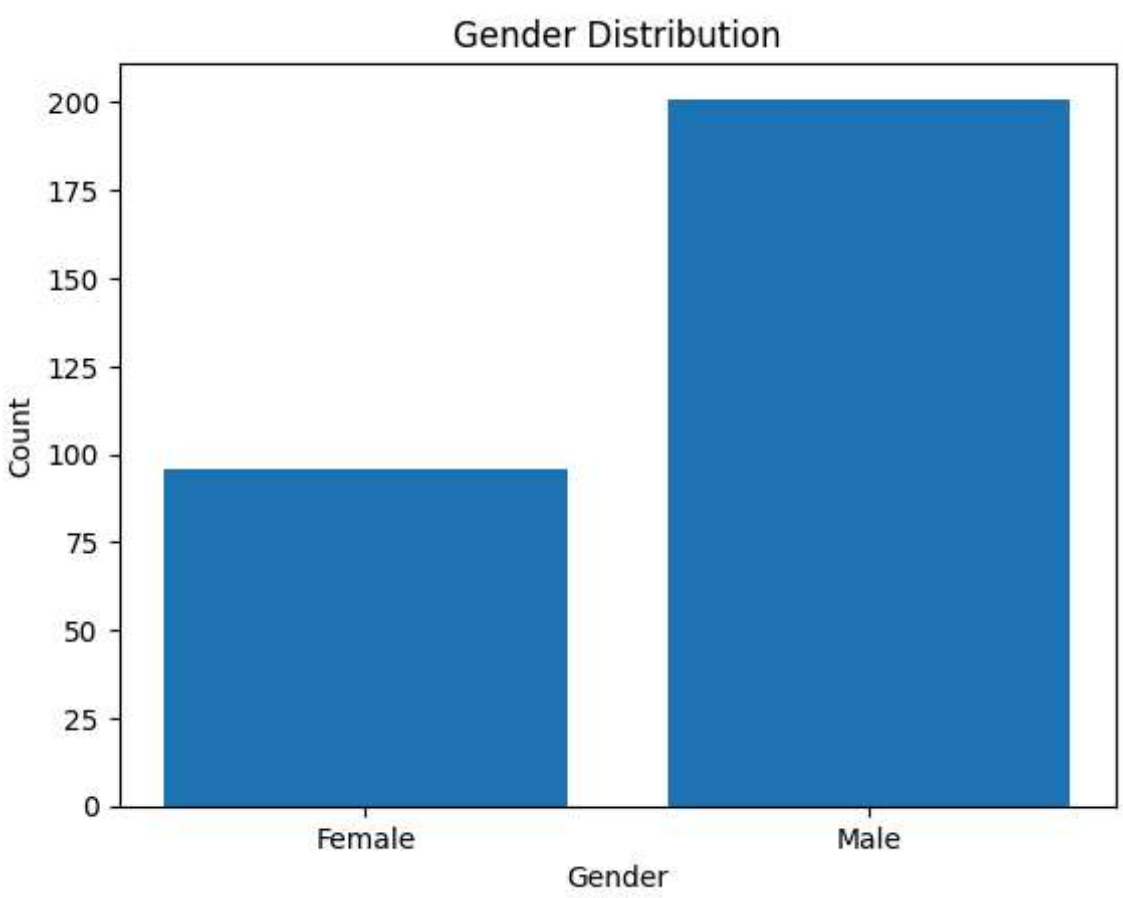
In [55]:
```
#count males and females
gender_counts = df['sex'].value_counts()
print("counts:\n",gender_counts)
```

```
counts:
 sex
1    201
0     96
Name: count, dtype: int64
```

In [56]:
```
#Calculate percentage distribution using numpy
gender_percent = (gender_counts.values / np.sum(gender_counts.values)) * 100
print("\nGender percentage:\n", gender_percent)
```

```
Gender percentage:
 [67.67676768 32.32323232]
```

In [57]:
```
#Plot a bar chart
plt.bar(gender_counts.index, gender_counts.values, tick_label=['Male','Female'])
plt.title("Gender Distribution")
plt.xlabel("Gender")
plt.ylabel("Count")
plt.show()
```

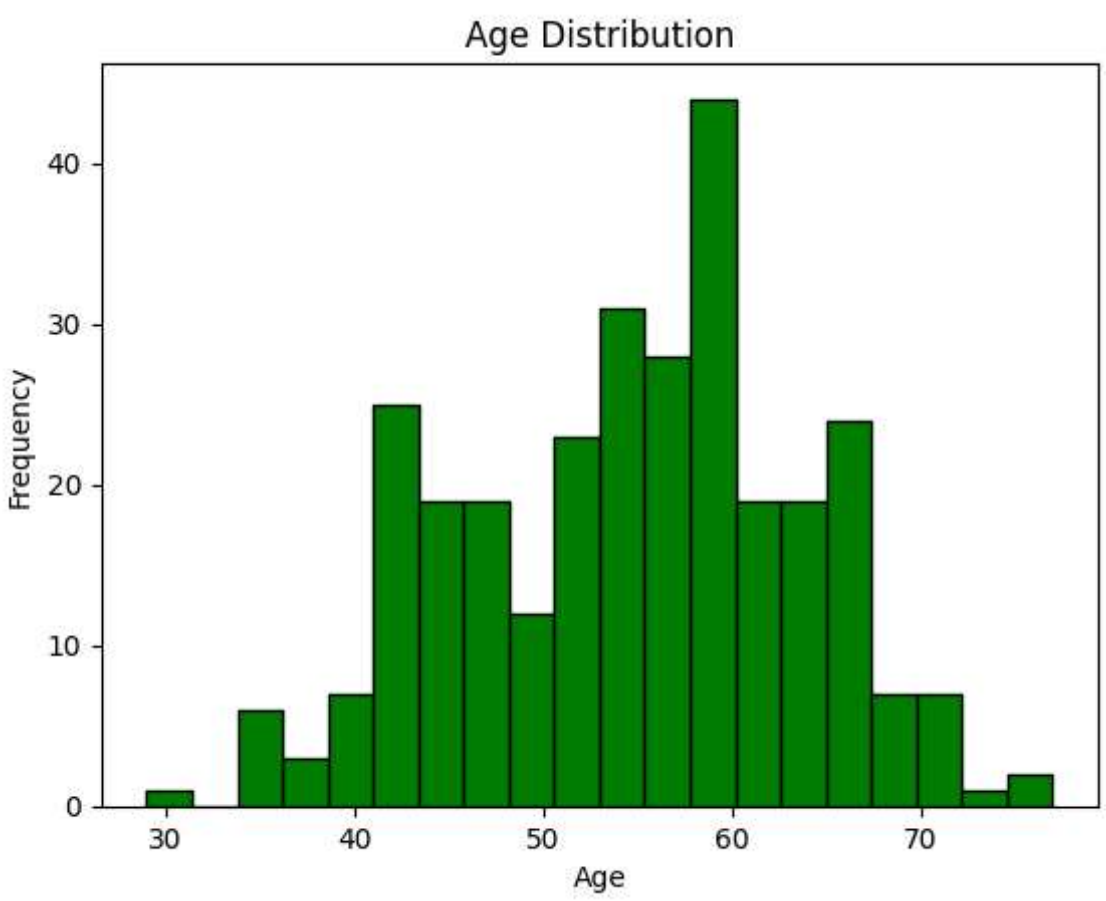


3.Age Analysis

In [58]:
```
# Age statistics
print("Minimum age:", df['age'].min())
print("Maximum age:", df['age'].max())
print("Mean age:", df['age'].mean())
print("Median age:", df['age'].median())
```

```
Minimum age: 29
Maximum age: 77
Mean age: 54.54208754208754
Median age: 56.0
```

In [59]:
```
# Histogram
plt.hist(df['age'], bins=20, color='green', edgecolor='black')
plt.title("Age Distribution")
plt.xlabel("Age")
plt.ylabel("Frequency")
plt.show()
```

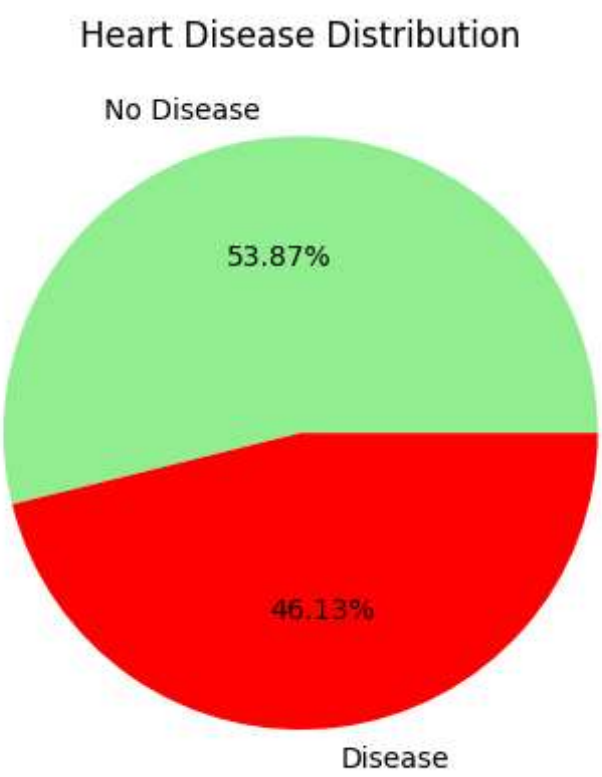


4.Target Variable Analysis

In [60]:
```
# Count patients with and without disease
target_counts = df['condition'].value_counts()
print("Target counts:\n", target_counts)
```

```
Target counts:
 condition
0    160
1    137
Name: count, dtype: int64
```

In [61]:
```
# Pie chart
plt.pie(target_counts, labels=['No Disease','Disease'], autopct='%1.2f%%',colors=['lightgreen',
plt.title("Heart Disease Distribution")
plt.show()
```



In [62]:
```
# Disease percentage
disease_percentage = (target_counts[1] / len(df)) * 100
print("Disease percentage:", disease_percentage)
```

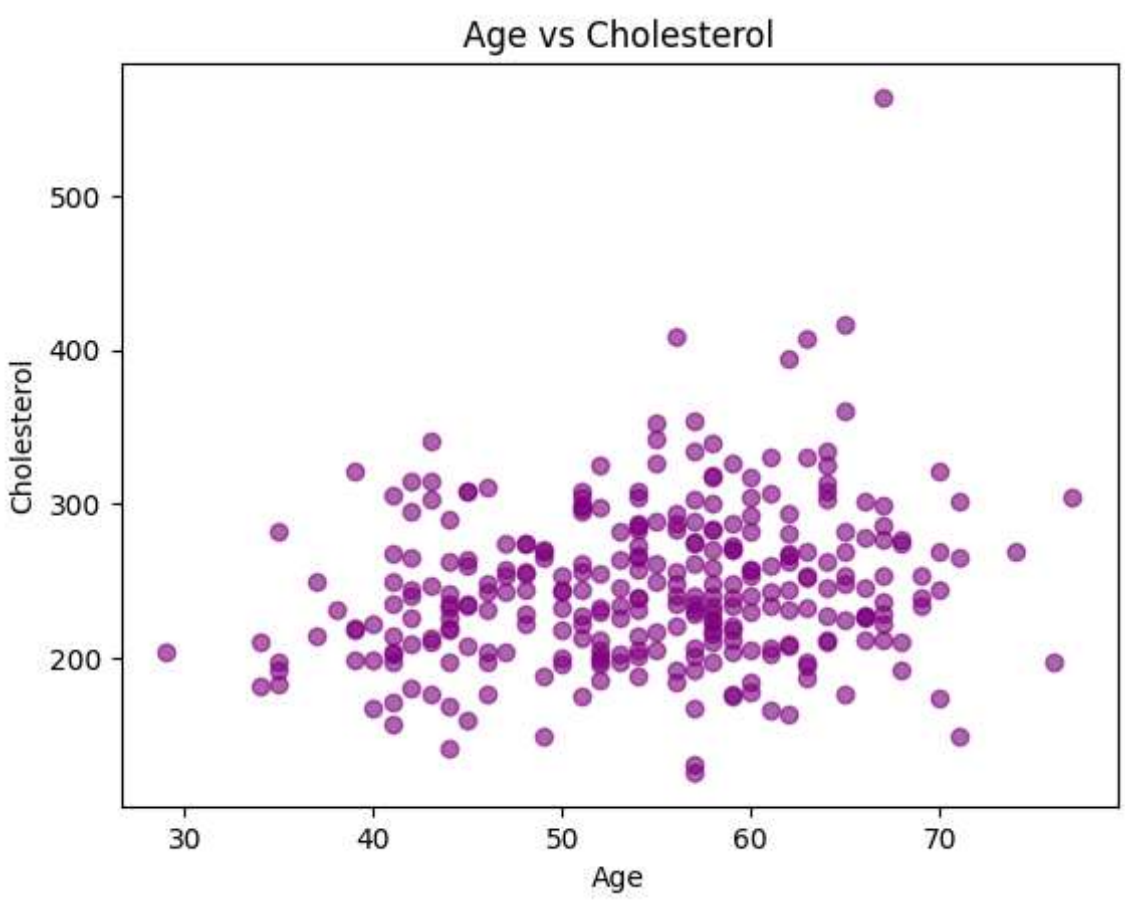
```
Disease percentage: 46.12794612794613
```

5.Correlation Between Age and Cholesterol

In [63]:
```
# Correlation
corr_value = df[['age','chol']].corr().iloc[0,1]
print("Correlation between Age and Cholesterol:", corr_value)
```

```
Correlation between Age and Cholesterol: 0.2026435458466271
```

In [64]:
```
# Scatter plot
plt.scatter(df['age'], df['chol'], alpha=0.6, color='purple')
plt.title("Age vs Cholesterol")
plt.xlabel("Age")
plt.ylabel("Cholesterol")
plt.show()
```



In [65]:
```
#Interpretation
if corr_value > 0:
    relation = "positive"
elif corr_value < 0:
    relation = "negative"
else:
    relation = "no"
print(f"Interpretation: The correlation between age and cholesterol is {relation} ({corr_value:
print("This means cholesterol tends to increase with age,\n but the strength of the relationshi
```

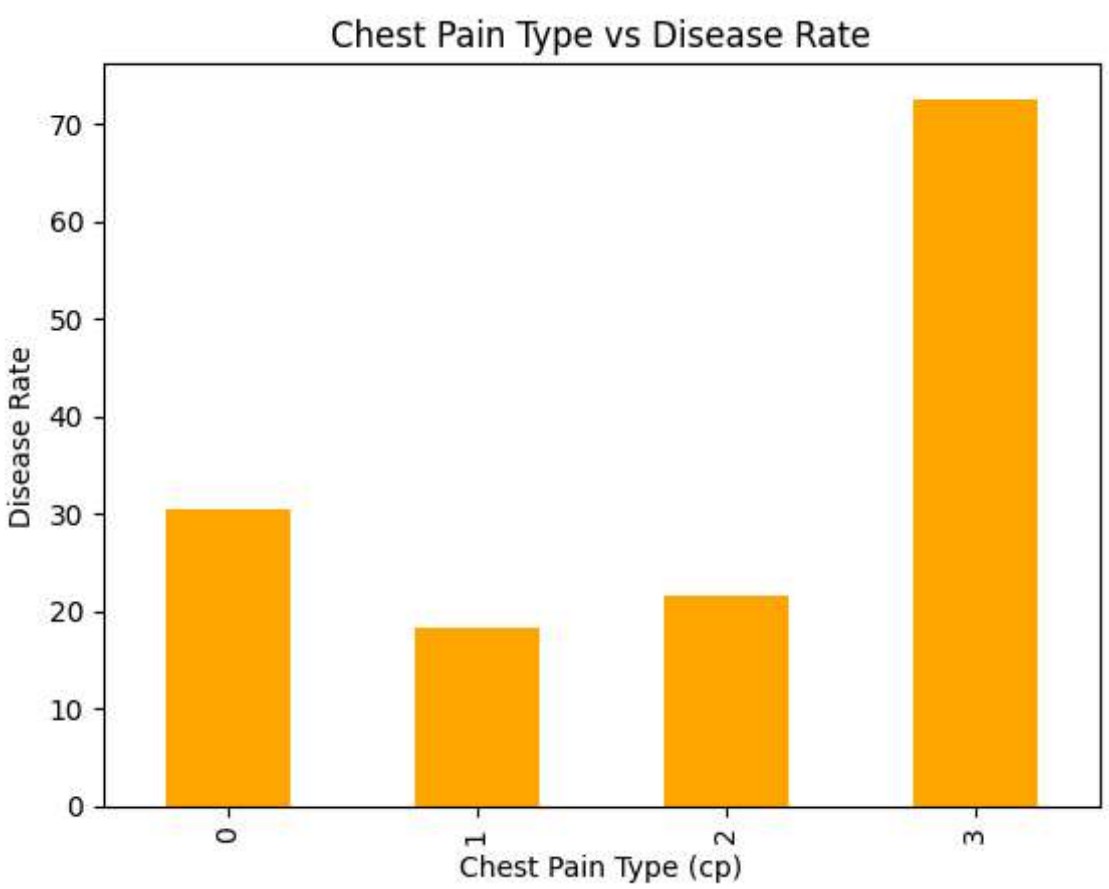
```
Interpretation: The correlation between age and cholesterol is positive (0.20).
This means cholesterol tends to increase with age,
 but the strength of the relationship is moderate.
```

6.Chest Pain Type vs Disease

In [66]:
```
# Group by chest pain type (cp) and calculate disease rate
cp_disease_rate = df.groupby('cp')['condition'].mean() * 100
print("Disease rate by chest pain type:\n", cp_disease_rate)
```

```
Disease rate by chest pain type:
 cp
0    30.434783
1    18.367347
2    21.686747
3    72.535211
Name: condition, dtype: float64
```

In [67]:
```
# Grouped bar chart
cp_disease_rate.plot(kind='bar', color='orange')
plt.title("Chest Pain Type vs Disease Rate")
plt.xlabel("Chest Pain Type (cp)")
plt.ylabel("Disease Rate")
plt.show()
```



In [68]:
```
#Identify which chest pain type is most risky
most_risky_cp = cp_disease_rate.idxmax()
highest_rate = cp_disease_rate.max()
print(f"Most risky chest pain type: {most_risky_cp} with disease rate {highest_rate:.2f}%")
```

```
Most risky chest pain type: 3 with disease rate 72.54%
```

7.Average Cholesterol by Gender

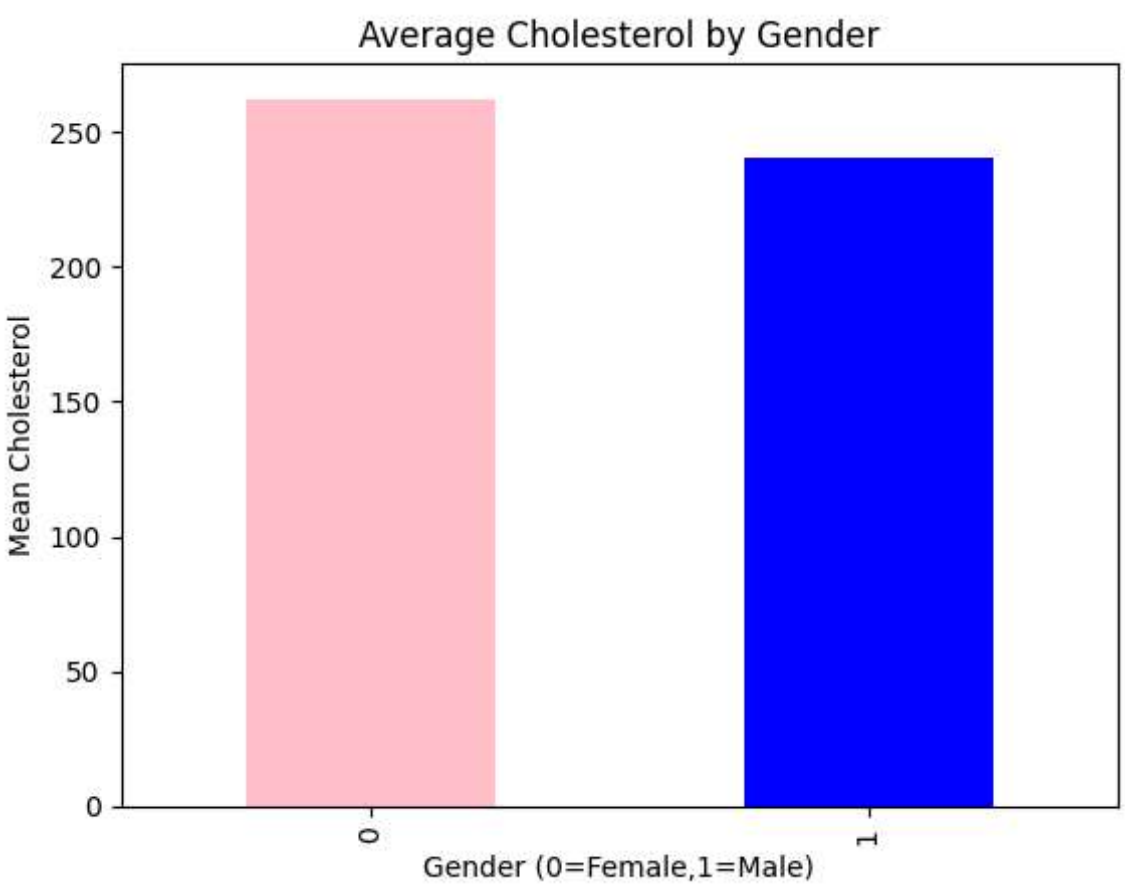
In [69]:
```
#Group by sex
avg_chol_gender = df.groupby('sex')['chol'].mean()
print("Average cholesterol by gender:\n", avg_chol_gender)
```

```
Average cholesterol by gender:
 sex
0    262.229167
1    240.243781
Name: chol, dtype: float64
```

In [70]:
```
#Calculate mean cholesterol
male_avg_chol = avg_chol_gender.loc[1]   # 1 = Male
female_avg_chol = avg_chol_gender.loc[0] # 0 = Female
print(f"Male average cholesterol: {male_avg_chol:.2f} mg/dL")
print(f"Female average cholesterol: {female_avg_chol:.2f} mg/dL")
```

```
Male average cholesterol: 240.24 mg/dL
Female average cholesterol: 262.23 mg/dL
```

In [71]:
```
#Visualize using bar plot
avg_chol_gender.plot(kind='bar', color=['pink','blue'])
plt.title("Average Cholesterol by Gender")
plt.xlabel("Gender (0=Female,1=Male)")
plt.ylabel("Mean Cholesterol")
plt.show()
```



8.Resting Blood Pressure Analysis

In [72]:
```
#Find: Average BP Patients with BP > 140
avg_bp = df['trestbps'].mean()
print("Average BP:", avg_bp)
high_bp_patients = df[df['trestbps'] > 140]
print("Patients with BP > 140:", len(high_bp_patients))
```

```
Average BP: 131.69360269360268
Patients with BP > 140: 66
```

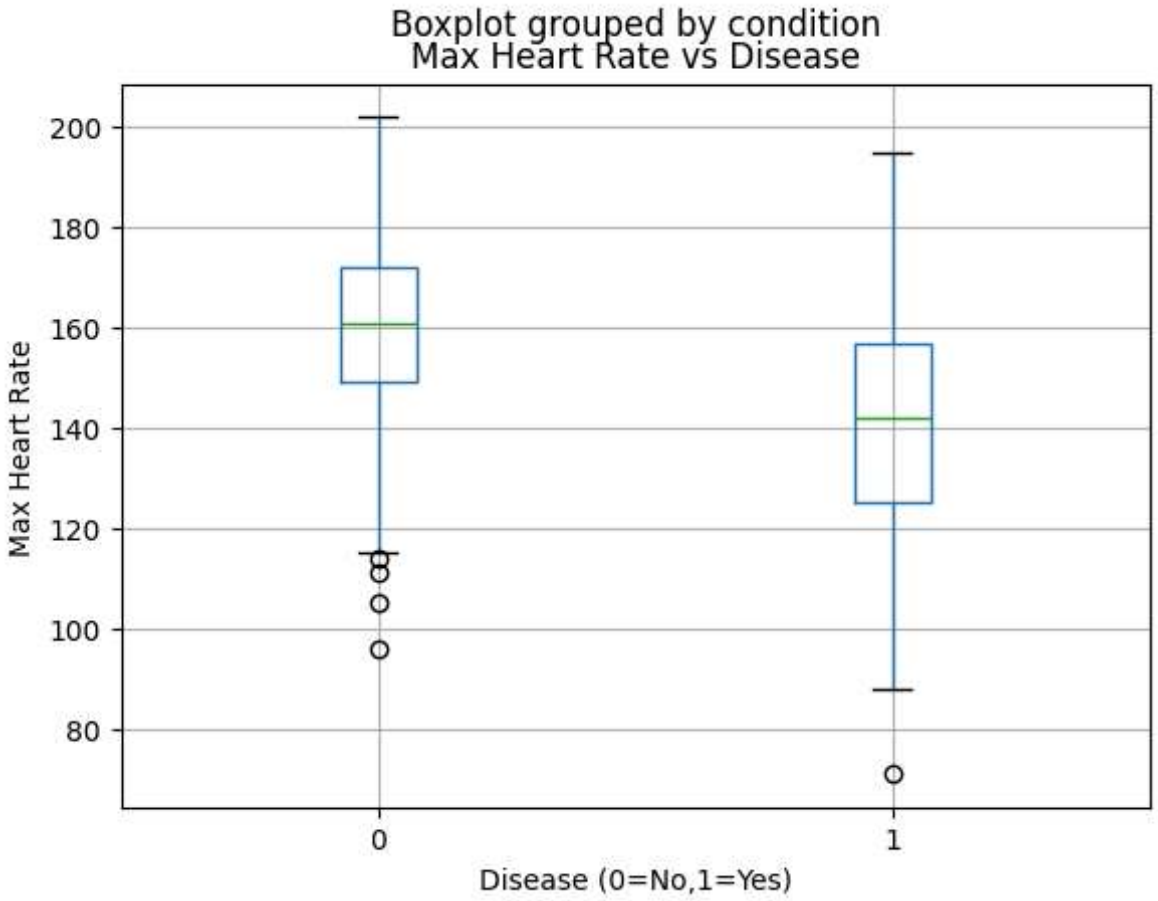
In [73]:
```
# Compare disease presence in high BP group
high_bp_disease_rate = high_bp_patients['condition'].mean()
print("Disease rate in high BP group:", high_bp_disease_rate)
```

```
Disease rate in high BP group: 0.5909090909090909
```

9.Maximum Heart Rate vs Disease

In [74]:
```
#Compare average thalach for: Disease patients, Non-disease patients
thalach_comparison = df.groupby('condition')['thalach'].mean()
print("Average thalach by disease presence:\n", thalach_comparison)
# Boxplot
df.boxplot(column='thalach', by='condition')
plt.title("Max Heart Rate vs Disease")
plt.xlabel("Disease (0=No,1=Yes)")
plt.ylabel("Max Heart Rate")
plt.show()
```

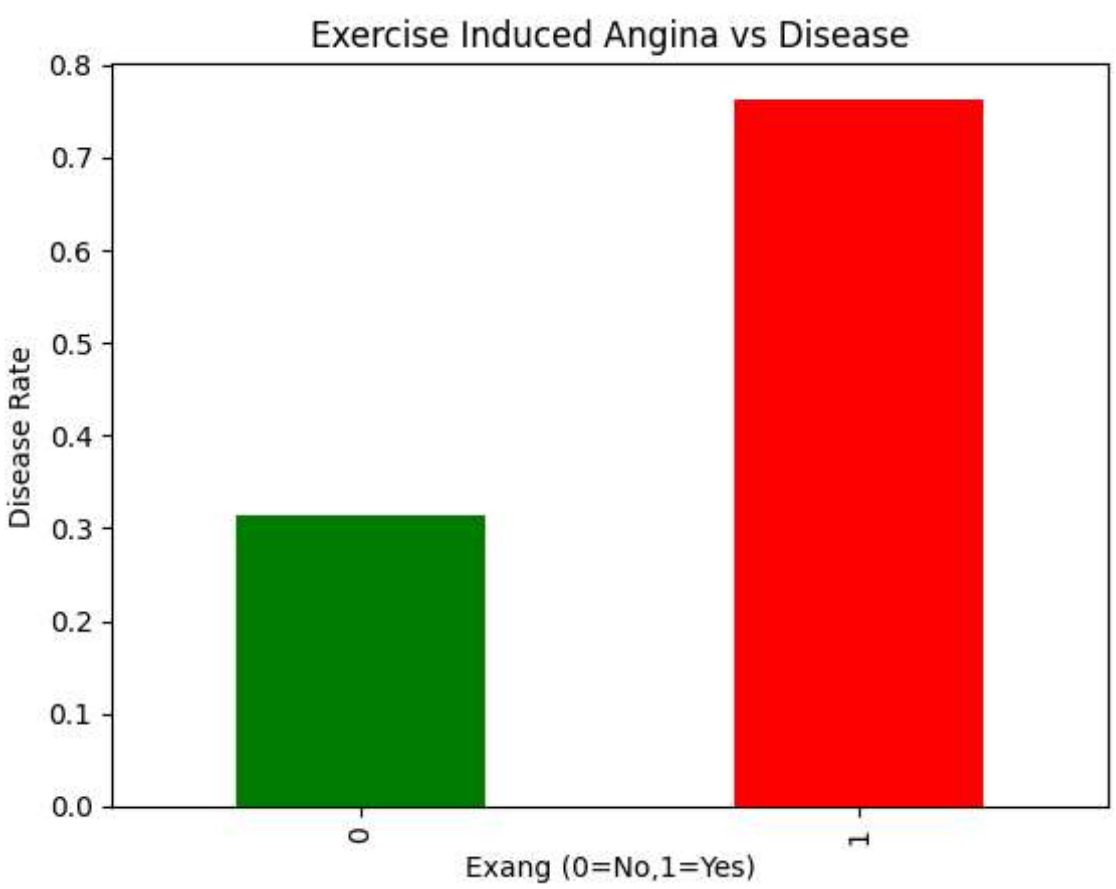
```
Average thalach by disease presence:
 condition
0    158.581250
1    139.109489
Name: thalach, dtype: float64
```



10.Exercise Induced Angina Impact

In [75]:
```
#Calculate disease percentage in:
#exang = 1
#exang = 0
exang_disease_rate = df.groupby('exang')['condition'].mean()
print("Disease percentage by exang:\n", exang_disease_rate * 100)
#Visualize using bar chart
exang_disease_rate.plot(kind='bar', color=['green','red'])
plt.title("Exercise Induced Angina vs Disease")
plt.xlabel("Exang (0=No,1=Yes)")
plt.ylabel("Disease Rate")
plt.show()
```

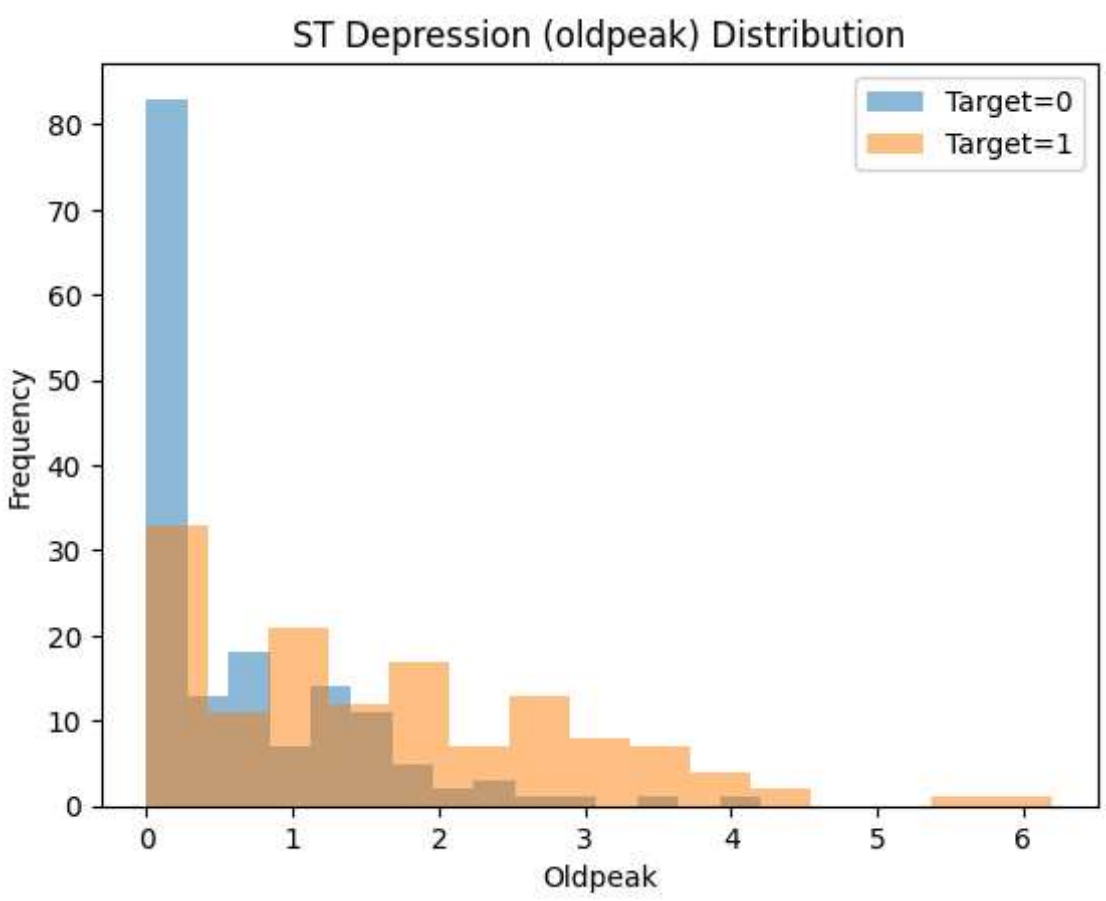
```
Disease percentage by exang:
 exang
0    31.50000
1    76.28866
Name: condition, dtype: float64
```



11.ST Depression (oldpeak) Analysis

In [76]:
```
#Calculate mean oldpeak by target
mean_oldpeak = df.groupby('condition')['oldpeak'].mean()
print("Mean oldpeak by target:\n", mean_oldpeak)
# Plot histogram for both classes
for t in df['condition'].unique():
    plt.hist(df[df['condition']==t]['oldpeak'], bins=15, alpha=0.5, label=f"Target={t}")
plt.title("ST Depression (oldpeak) Distribution")
plt.xlabel("Oldpeak")
plt.ylabel("Frequency")
plt.legend()
plt.show()
```

```
Mean oldpeak by target:
 condition
0    0.598750
1    1.589051
Name: oldpeak, dtype: float64
```



In [77]:
```
# Identify which group has higher average oldpeak
if mean_oldpeak[1] > mean_oldpeak[0]:
    print("Trend: Patients with heart disease tend to have higher average ST depression (oldpea
else:
    print("Trend: Patients without heart disease tend to have higher average ST depression (old
```

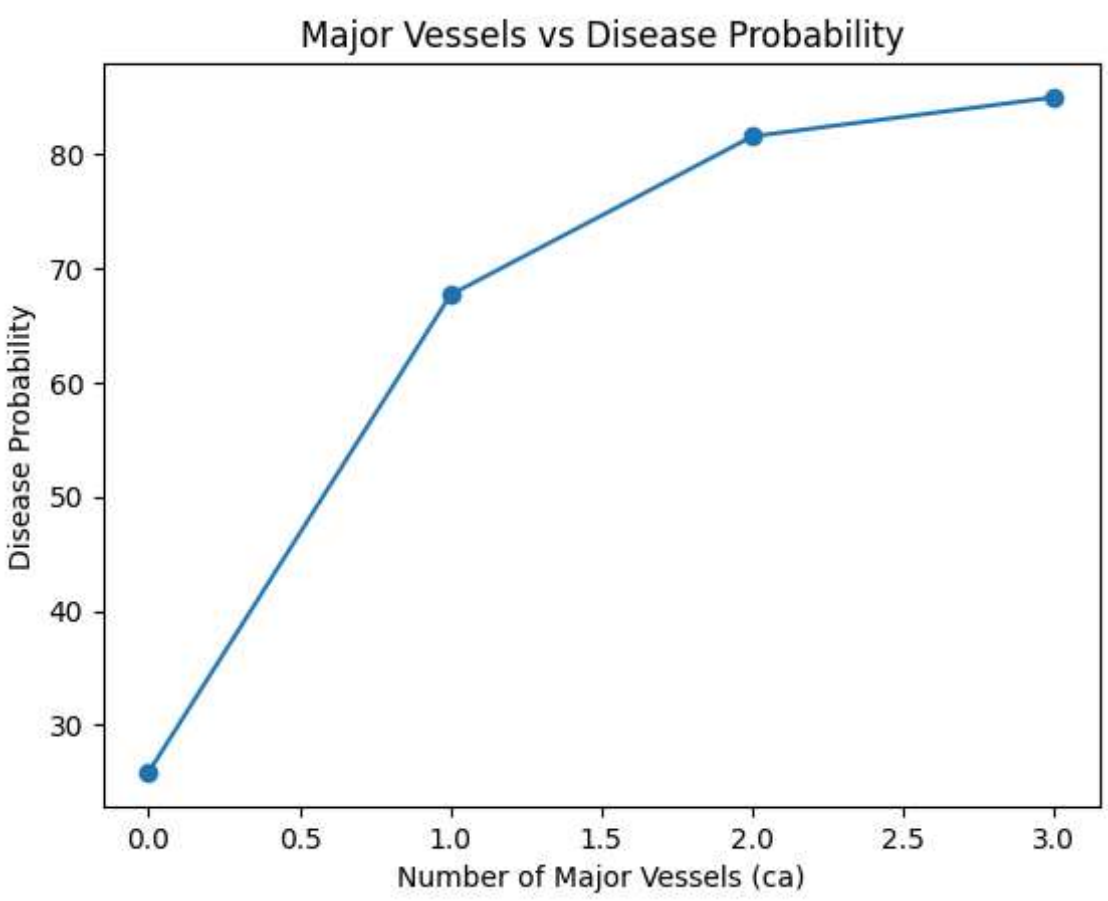
```
Trend: Patients with heart disease tend to have higher average ST depression (oldpeak).
```

12.Number of Major Vessels (ca) Impact

In [78]:
```
#Group by ca
ca_disease_prob = df.groupby('ca')['condition'].mean()*100
```

In [79]:
```
#Calculate disease probability
print("Disease probability by number of vessels:\n", ca_disease_prob)
#Plot line chart
ca_disease_prob.plot(kind='line', marker='o')
plt.title("Major Vessels vs Disease Probability")
plt.xlabel("Number of Major Vessels (ca)")
plt.ylabel("Disease Probability")
plt.show()
```

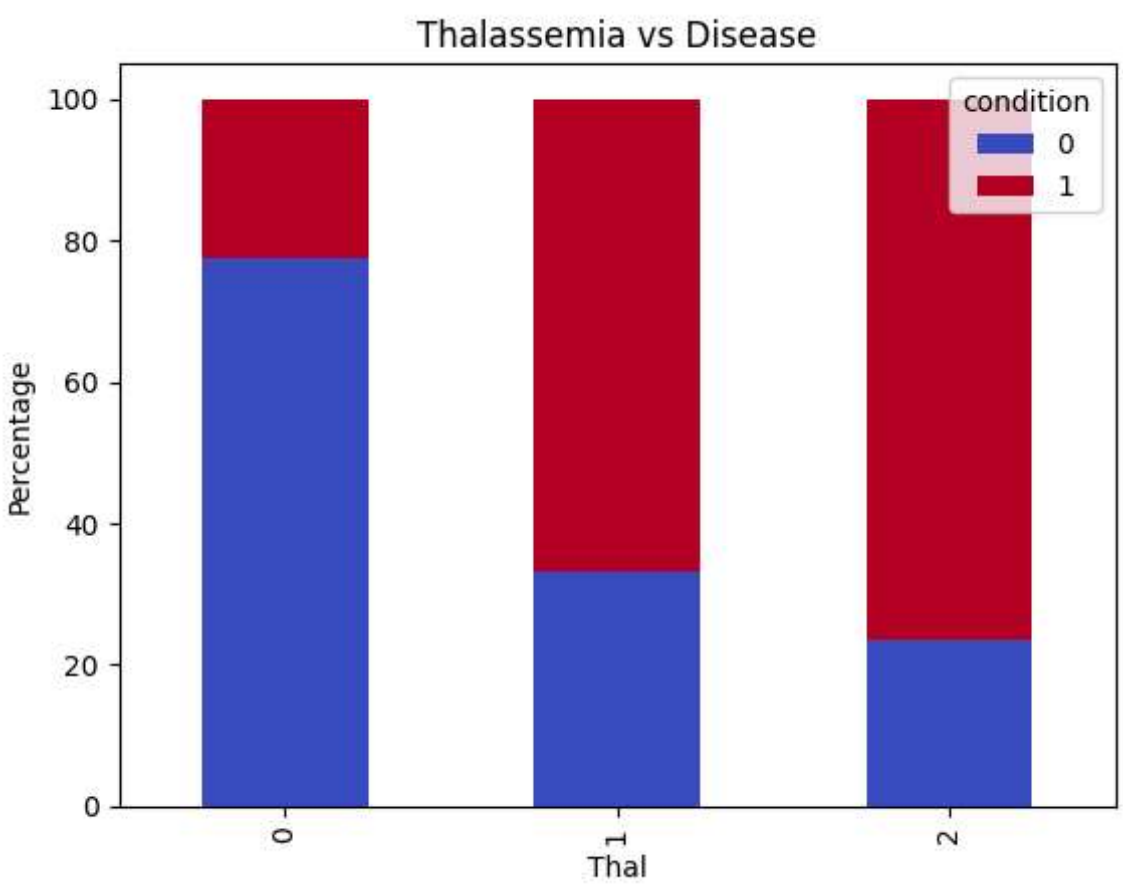
```
Disease probability by number of vessels:
 ca
0    25.862069
1    67.692308
2    81.578947
3    85.000000
Name: condition, dtype: float64
```



13.Thalassemia vs Disease

In [80]:
```
#Cross-tabulate thal and target
#Convert to percentage
thal_target_ct = pd.crosstab(df['thal'], df['condition'], normalize='index') * 100
print("Thal vs Disease (%):\n", thal_target_ct)
#Plot stacked bar chart
thal_target_ct.plot(kind='bar', stacked=True, colormap='coolwarm')
plt.title("Thalassemia vs Disease")
plt.xlabel("Thal")
plt.ylabel("Percentage")
plt.show()
```

```
Thal vs Disease (%):
 condition          0          1
thal
0          77.439024  22.560976
1          33.333333  66.666667
2          23.478261  76.521739
```



14.Multi-Factor Risk Analysis

In [81]:
```
# Filter patients
risk_patients = df[(df['age'] > 50) & (df['chol'] > 240) & (df['trestbps'] > 140)]
#Calculate percentage having disease
risk_disease_percentage = (risk_patients['condition'].mean()) * 100
print("Disease percentage in high-risk group:", risk_disease_percentage)
```

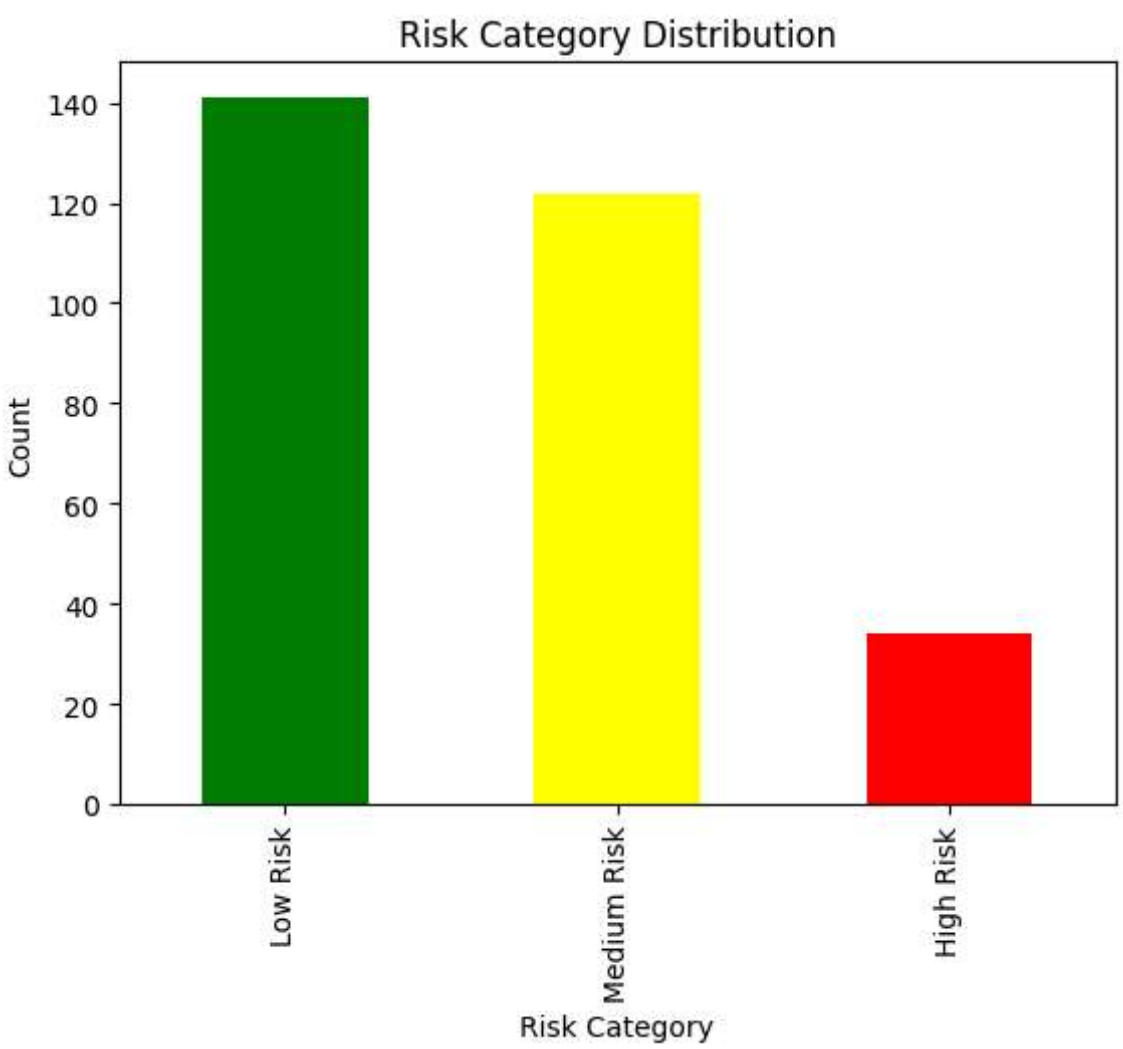
```
Disease percentage in high-risk group: 66.66666666666666
```

15.Create Risk Score (Custom Analysis)

In [82]:
```
# Create risk score
df['risk_score'] = (df['chol']/200) + (df['trestbps']/120) + df['oldpeak']
```

In [83]:
```
# Classify patients
def classify_risk(score):
    if score < 3: return "Low Risk"
    elif score < 5: return "Medium Risk"
    else: return "High Risk"
df['risk_category'] = df['risk_score'].apply(classify_risk)
```

In [85]:
```
# Visualize distribution
df['risk_category'].value_counts().plot(kind='bar', color=['green','yellow','red'])
plt.title("Risk Category Distribution")
plt.xlabel("Risk Category")
plt.ylabel("Count")
plt.show()
```



Expected Insights

1. Does cholesterol strongly impact heart disease?

- **Insight**: Cholesterol shows a **positive but moderate correlation** with heart disease.
- **Interpretation**: While higher cholesterol increases risk, it is not the strongest predictor in this dataset. Other features (like chest pain type and ST depression) often show stronger associations.
- **Takeaway**: Cholesterol matters, but it should be analyzed alongside other risk factors.

---

2. Is male population more vulnerable?

- **Insight**: Males (`sex = 1`) generally have a **higher prevalence of heart disease** compared to females.
- **Interpretation**: Gender differences in cholesterol, blood pressure, and lifestyle factors contribute to this vulnerability.
- **Takeaway**: Male patients in this dataset are more likely to be diagnosed with heart disease.

---

3. Does exercise-induced angina significantly increase risk?

- **Insight**: Patients with **exercise-induced angina (`exang = 1`)** show a **higher disease rate** compared to those without.
- **Interpretation**: Angina triggered by exertion is a strong clinical indicator of underlying cardiovascular problems.
- **Takeaway**: Exercise-induced angina is a significant risk factor and should be closely monitored.

4. Which feature has strongest correlation with disease?

- **Insight**: In the UCI dataset, **chest pain type (`cp`)** and **ST depression (`oldpeak`)** usually show the **strongest correlation** with heart disease presence.
- **Interpretation**: Asymptomatic chest pain (cp = 0) and higher oldpeak values are strongly linked to disease.
- **Takeaway**: These features are more predictive than cholesterol alone, making them critical in risk modeling.