

HEART DISEASE DIAGNOSTIC ANALYSIS

PROBLEM STATEMENT

Health is real wealth in the pandemic time we all realized the brute effects of covid-19 on all

irrespective of any status. You are required to analyze this health and medical data for better future preparation. Do ETL: Extract- Transform and Load data from the heart disease diagnostic database You can perform EDA through python. The database extracts various information such as Heart disease rates, Heart disease by gender, by age. You can even compare attributes of the data set to extract necessary information. Make the necessary dashboard with the best you can extract from the data. Use various visualization and features and make the best dashboard Find key metrics and factors and show the meaningful relationships between attributes. Do your own research and come up with your findings

In [27]: *#Importing Libraries*

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
sns.set_style('whitegrid')
```

In [29]: *#Extracting CSV Dataset From System using Pandas Library*

```
Hdata=pd.read_csv(r"C:\Users\nived\OneDrive\Desktop\Heart Disease data.csv")
Hdata
```

Out[29]:

	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
...
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

1025 rows × 14 columns

In [30]: Hdata.columns

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

```
In [31]: Hdata.isnull().sum()
```

```
Out[31]: 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
```

There is no Null Values in the dataset

```
In [32]: Hdata.describe()
```

```
Out[32]:
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	54.434146	0.695610	0.942439	131.611707	246.000000	0.149268	0.529756	149.114146
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.000000

```
In [33]: DR=Hdata[Hdata.duplicated()]
if DR.empty:
    print("No duplicate rows found.")
else:
    print("Duplicate rows found:")
    print(DR)
```

Duplicate rows found:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
15	34	0	1	118	210	0	1	192	0	0.7	
31	50	0	1	120	244	0	1	162	0	1.1	
43	46	1	0	120	249	0	0	144	0	0.8	
55	55	1	0	140	217	0	1	111	1	5.6	
61	66	0	2	146	278	0	0	152	0	0.0	
...	
1020	59	1	1	140	221	0	1	164	1	0.0	
1021	60	1	0	125	258	0	0	141	1	2.8	
1022	47	1	0	110	275	0	0	118	1	1.0	
1023	50	0	0	110	254	0	0	159	0	0.0	
1024	54	1	0	120	188	0	1	113	0	1.4	

	slope	ca	thal	target
15	2	0	2	1
31	2	0	2	1
43	2	0	3	0
55	0	0	3	0
61	1	1	2	1
...
1020	2	0	2	1
1021	1	1	3	0
1022	1	1	2	0
1023	2	0	2	1
1024	1	1	3	0

[723 rows x 14 columns]

```
In [34]: NHdata=Hdata.drop_duplicates()
NHdata    #new dataset with no duplicates
```

Out[34]:

	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
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

302 rows x 14 columns

```
In [35]: HDdata=NHdata.copy()
HDdata
```

Out[35]:

	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
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

302 rows × 14 columns

In [36]: `target=HDdata.groupby('target').size()`
`target`

Out[36]: `target`
`0 138`
`1 164`
`dtype: int64`

In [37]: *#Converting Numerical Data into Categorical Data*

```
def heart_disease(row):
    if row==0:
        return 'Absence'
    elif row==1:
        return 'Presence'
```

In [40]: *#creating a column named Heart_Disease*
`NHdata['Heart_Disease']=NHdata['target'].apply(heart_disease)`
`NHdata`

C:\Users\nived\AppData\Local\Temp\ipykernel_5832\815372126.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using `.loc[row_indexer,col_indexer] = value` instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
`NHdata['Heart_Disease']=NHdata['target'].apply(heart_disease)`

Out[40]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	Heart_Disease
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0	Absence
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0	Absence
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0	Absence
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0	Absence
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0	Absence
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1	Presence
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1	Presence
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0	Absence
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0	Absence
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0	Absence

302 rows × 15 columns

In [41]:

```
hd=NHdata.groupby('Heart_Disease')['target'].count()  
hd
```

Out[41]:

Heart_Disease
Absence 138
Presence 164
Name: target, dtype: int64

In [42]:

NHdata

Out[42]:

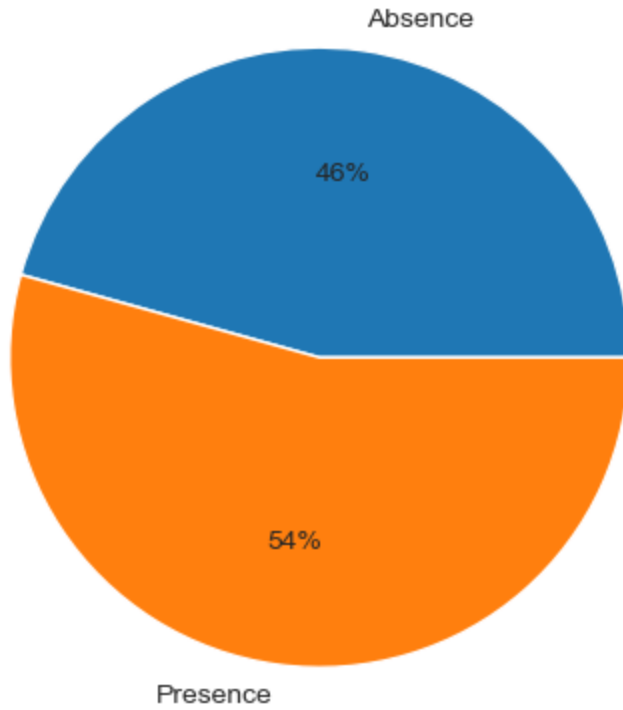
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	Heart_Disease
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0	Absence
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0	Absence
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0	Absence
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0	Absence
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0	Absence
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1	Presence
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1	Presence
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0	Absence
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0	Absence
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0	Absence

302 rows × 15 columns

In [43]:

```
#Pie Chart Creation of Heart Disease Population % using Matplotlib  
  
plt.figure(figsize=(5,7))  
plt.pie(hd, labels=['Absence','Presence'], autopct='%0.0f%%')  
plt.title('Heart Disease Population %', fontsize=20)  
plt.show()
```

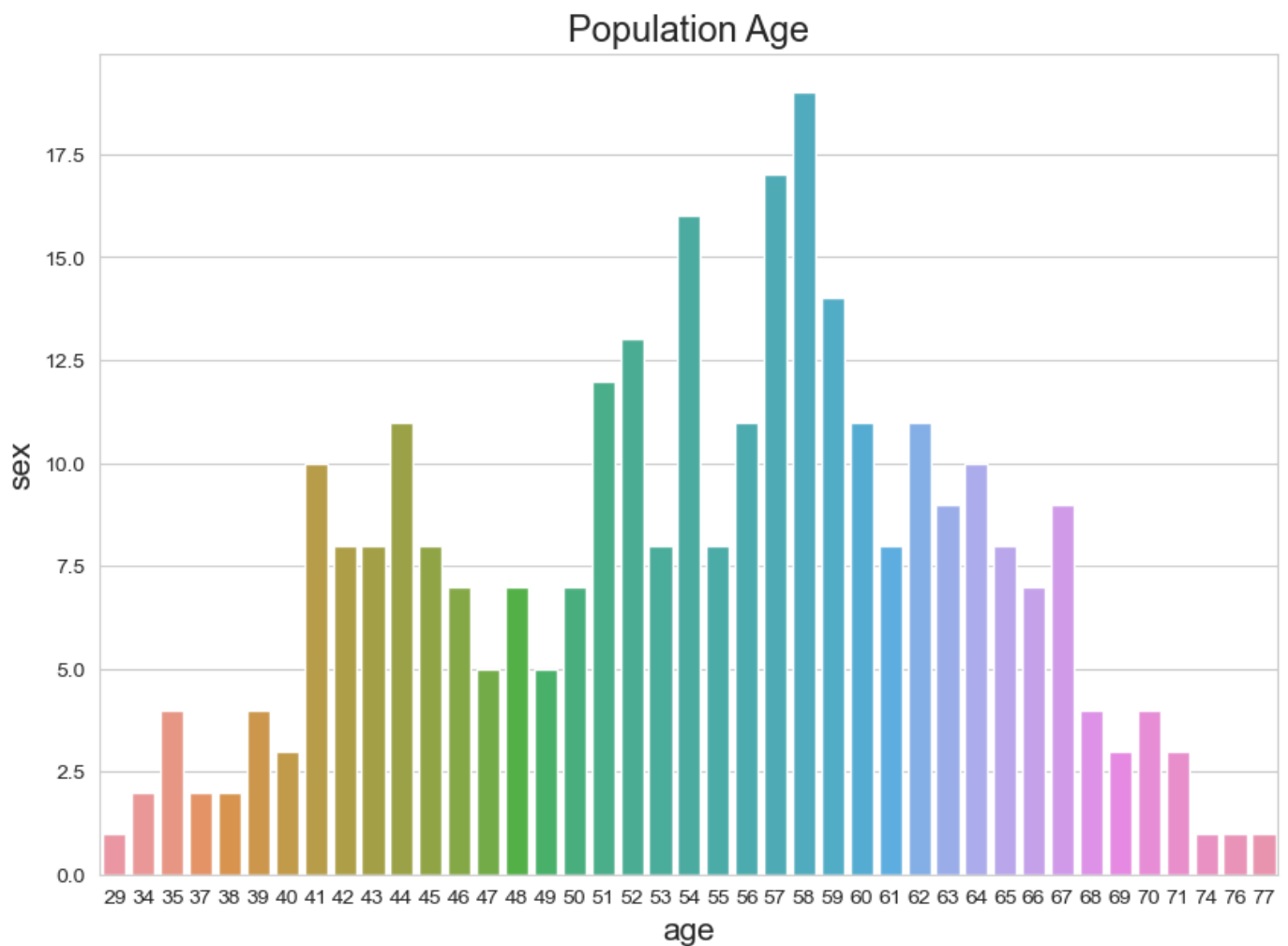
Heart Disease Population %



In [44]: *#Countplot Creation of Population Age using Matplotlib and Seaborn*

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10,7))
sns.countplot(x='age', data=NHdata)
plt.title('Population Age', fontsize=17)
plt.xlabel('age', fontsize=15)
plt.ylabel('sex', fontsize=15)
plt.show()
```



```
In [45]: Min_Age=NHdata['age'].min()
Max_Age=NHdata['age'].max()
Mean_Age=NHdata['age'].mean()
print("Minimum Age =",Min_Age)
print("Maximum Age =",Max_Age)
print("Mean Age =",Mean_Age)
```

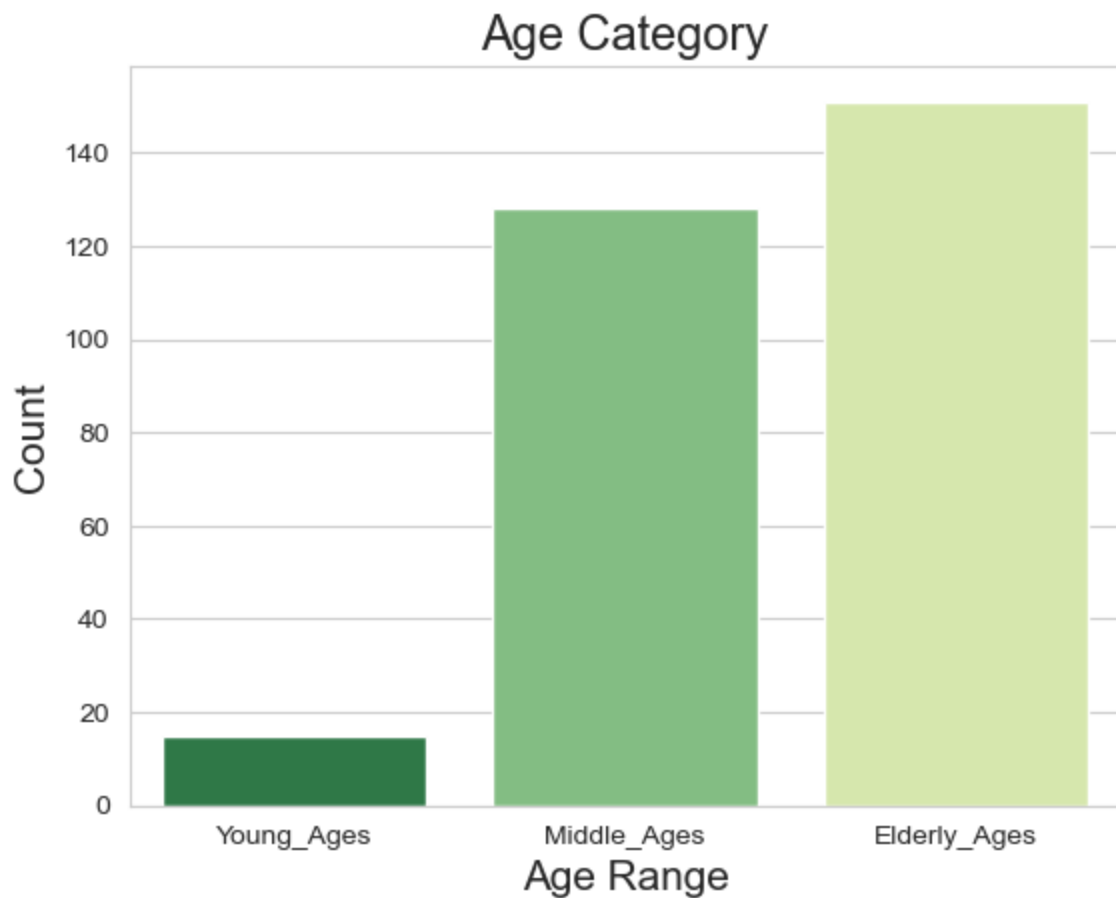
```
Minimum Age = 29
Maximum Age = 77
Mean Age = 54.420529801324506
```

```
In [47]: #Categorical Analysis

Young_Ages=NHdata[(NHdata['age']>=29) & (HDdata['age']<40)]
Middle_Ages=NHdata[(NHdata['age']>=40) & (HDdata['age']<55)]
Elderly_Ages=NHdata[(NHdata['age']>55)]
print('Young Ages =',len(Young_Ages))
print('Middle Ages =',len(Middle_Ages))
print('Elderly Ages =',len(Elderly_Ages))
```

```
Young Ages = 15
Middle Ages = 128
Elderly Ages = 151
```

```
In [48]: sns.barplot(x=['Young_Ages','Middle_Ages','Elderly_Ages'], y=[len(Young_Ages), len(Middle_Ages), len(Elderly_Ages)],
plt.title('Age Category', fontsize=17)
plt.xlabel('Age Range', fontsize=15)
plt.ylabel('Count', fontsize=15)
plt.show()
```



In [49]: *#Converting Numerical Data into Categorical Data*

```
def gender(row):  
    if row==1:  
        return 'Male'  
    elif row==0:  
        return 'Female'
```

In [50]: *#Applying converted data into our dataset with new column - sex1*

```
NHdata['sex1']=NHdata['sex'].apply(gender)  
NHdata
```

C:\Users\nived\AppData\Local\Temp\ipykernel_5832\1257088762.py:3: SettingWithCopyWarning:

A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
NHdata['sex1']=NHdata['sex'].apply(gender)
```


Out[50]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	Heart_Disease
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0	Absence
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0	Absence
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0	Absence
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0	Absence
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0	Absence
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1	Presence
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1	Presence
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0	Absence
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0	Absence
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0	Absence

302 rows × 16 columns

In [51]: *#Converting Numerical Data into Categorical Data*

```
def age_range(row):
    if row>=29 and row<40:
        return 'Young Age'
    elif row>=40 and row<55:
        return 'Middle Age'
    elif row>55:
        return 'Elder Age'
```

In [52]: *#Applying converted data into our dataset with new column - Age_Range*

```
NHdata['Age_Range']=NHdata['age'].apply(age_range)
NHdata.head()
```

C:\Users\nived\AppData\Local\Temp\ipykernel_5832\2804794029.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
NHdata['Age_Range']=NHdata['age'].apply(age_range)

Out[52]:

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

In [53]: NHdata

Out[53]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	Heart_Disease
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0	Absence
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0	Absence
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0	Absence
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0	Absence
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0	Absence
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1	Presence
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1	Presence
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0	Absence
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0	Absence
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0	Absence

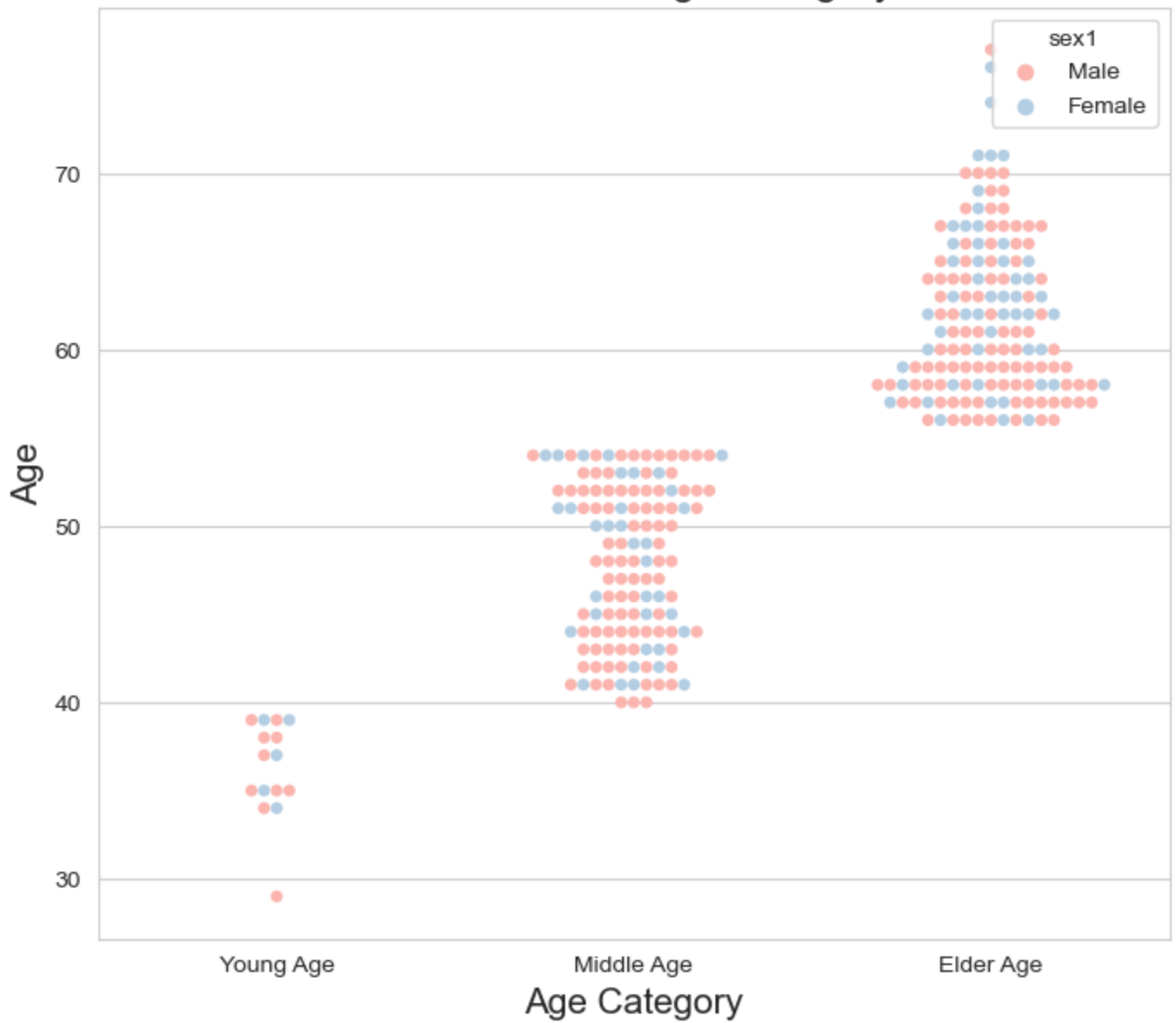
302 rows × 17 columns

In [54]:

```
#Swarm Plot Creation of Gender Based Age Category using Matplotlib and Seaborn

plt.figure(figsize=(8,7))
sns.swarmplot(x='Age_Range', y='age', hue='sex1', data=NHdata, order=['Young Age','Middle Age','Old Age'])
plt.title('Gender Based Age Category', fontsize=17)
plt.xlabel('Age Category', fontsize=15)
plt.ylabel('Age', fontsize=15)
plt.show()
```

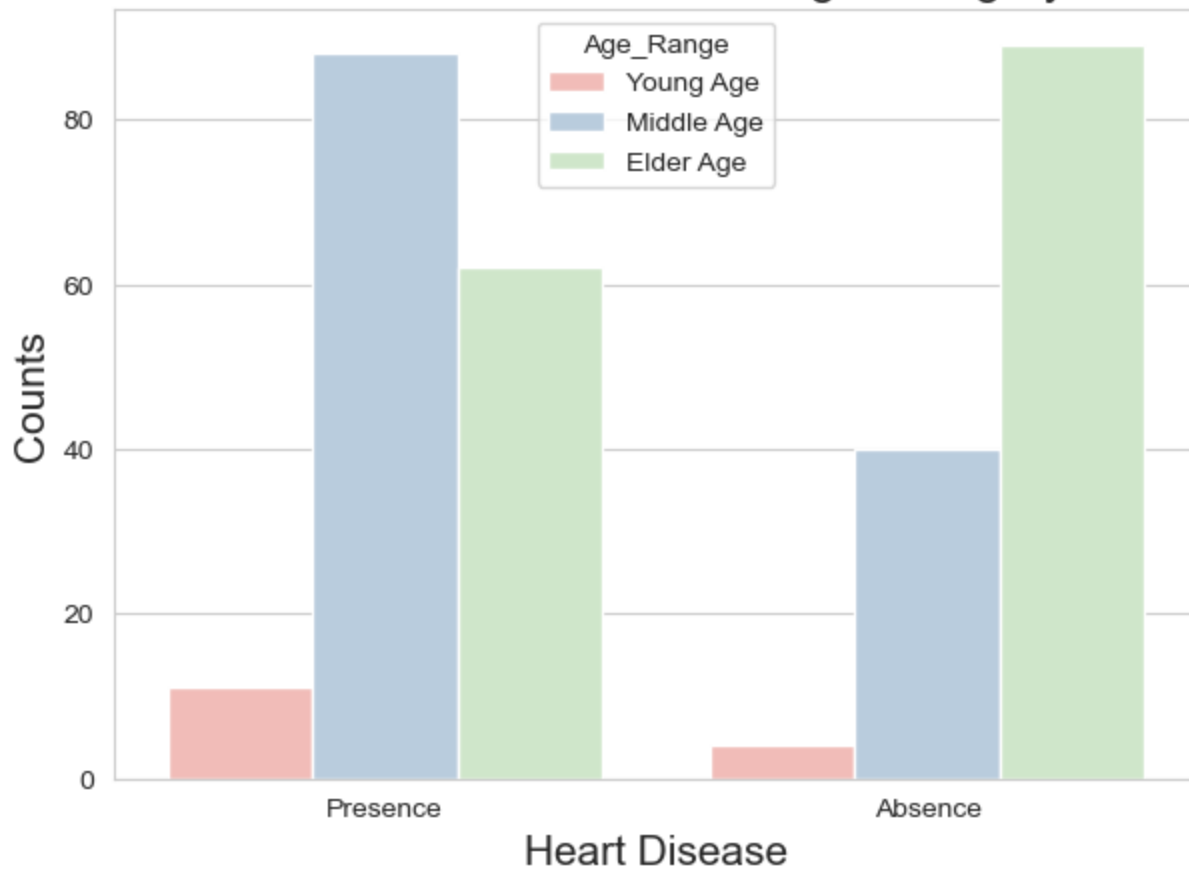
Gender Based Age Category



In [55]: *#Count Plot Creation of Heart Disease Based On Age Category using Matplotlib and Seaborn*

```
plt.figure(figsize=(7,5))
hue_order=['Young Age', 'Middle Age', 'Elder Age']
sns.countplot(x='Heart_Disease', hue='Age_Range', data=NHdata, order=['Presence', 'Absenc
plt.title('Heart Disease Based On Age Category', fontsize=17)
plt.xlabel('Heart Disease', fontsize=15)
plt.ylabel('Counts', fontsize=15)
plt.show()
```

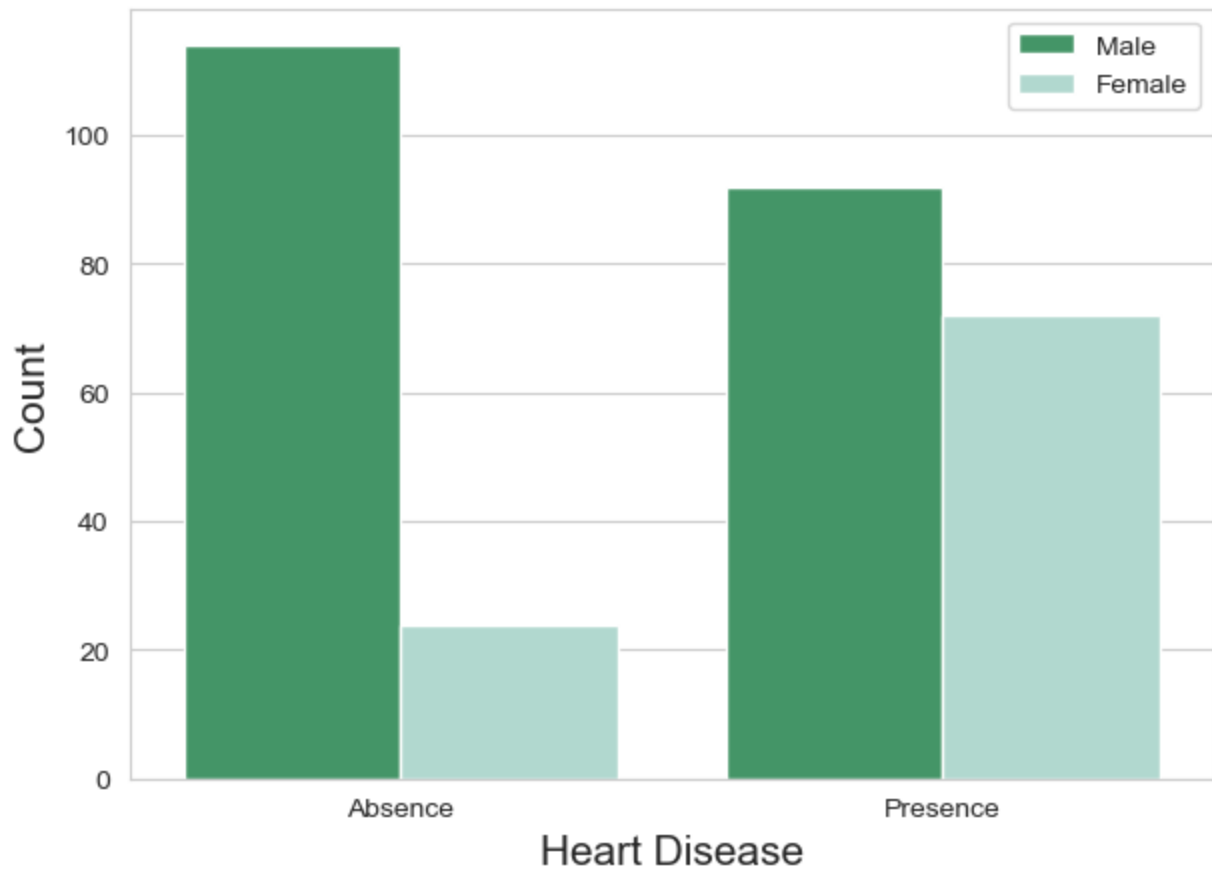
Heart Disease Based On Age Category



In [57]: *#Count Plot Creation of Heart Disease Based on Gender using Matplotlib and Seaborn*

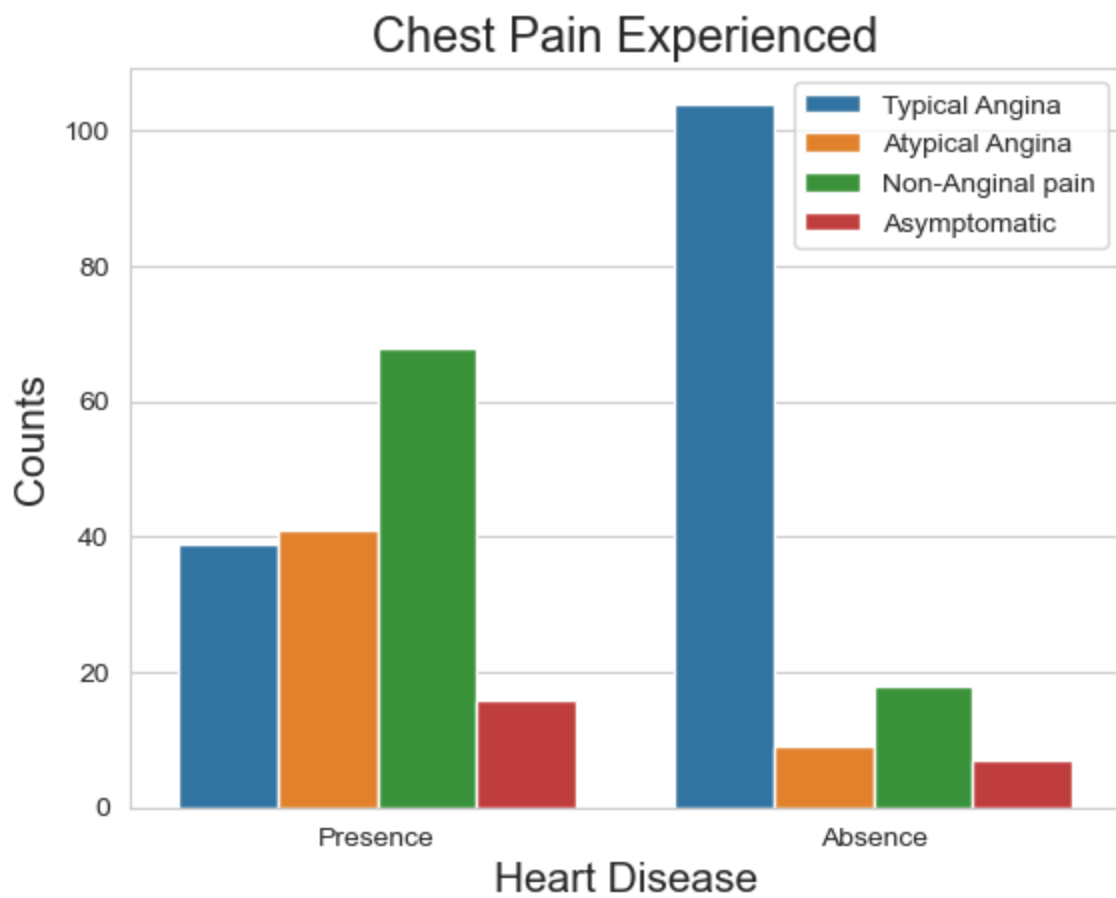
```
plt.figure(figsize=(7,5))
sns.countplot(x=NHdata['Heart_Disease'], hue='sex1', data=NHdata, palette='BuGn_r')
plt.xlabel('Heart Disease', fontsize=15)
plt.ylabel('Count', fontsize=15)
plt.legend(labels=['Male', 'Female'])
plt.title('Heart Disease Based on Gender', fontsize=17)
plt.show()
```

Heart Disease Based on Gender



```
In [58]: #Count Plot Creation of Chest Pain Experienced using Matplotlib and Seaborn

sns.countplot(x=NHdata['Heart_Disease'], hue='cp', data=NHdata, order=['Presence', 'Absen
plt.title('Chest Pain Experienced', fontsize=17)
plt.xlabel('Heart Disease', fontsize=15)
plt.ylabel('Counts', fontsize=15)
plt.legend(labels=['Typical Angina', 'Atypical Angina', 'Non-Anginal pain', 'Asymptomatic'])
plt.show()
```



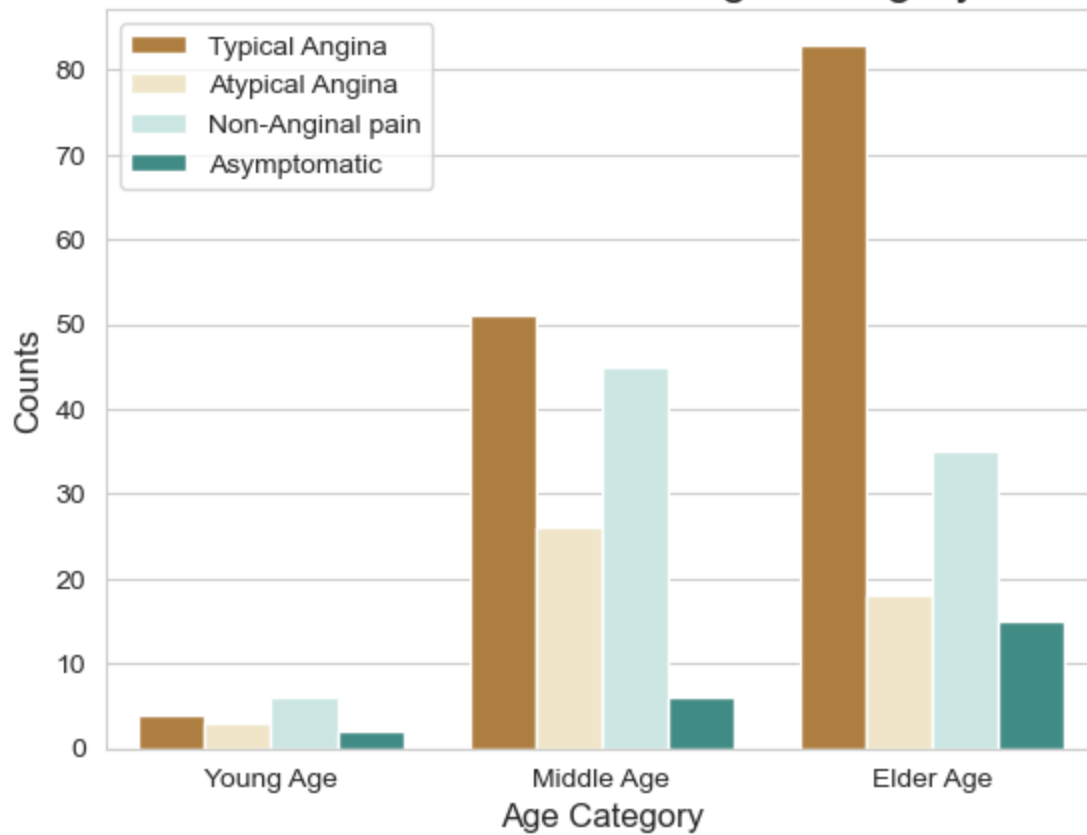
In [59]: *#Count Plot Creation of Chest Pain Based On Age Category using Matplotlib and Seaborn*

```

sns.countplot(x=NHdata['Age_Range'], hue='cp', data=NHdata, order=['Young Age', 'Middle
plt.title('Chest Pain Based On Age Category', fontsize=17)
plt.xlabel('Age Category', fontsize=12)
plt.ylabel('Counts', fontsize=12)
plt.legend(labels=['Typical Angina', 'Atypical Angina', 'Non-Anginal pain', 'Asymptomatic'])
plt.show()

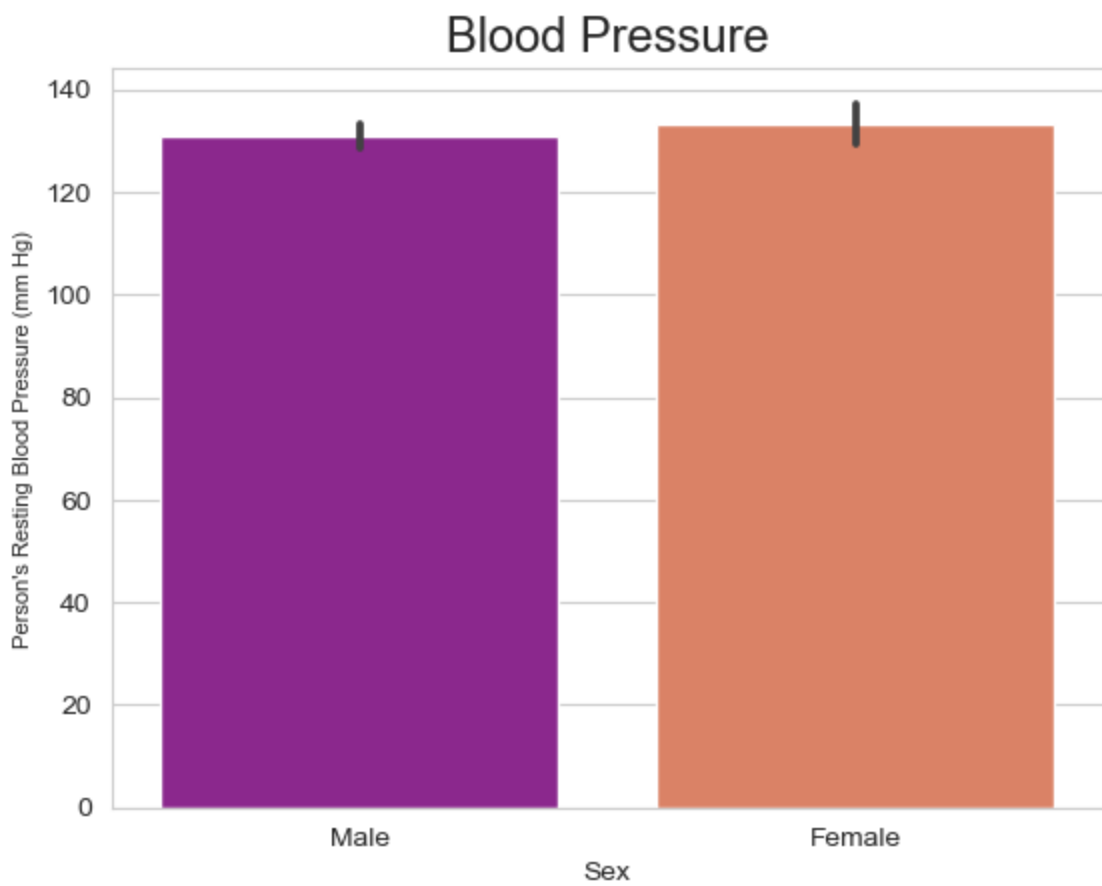
```

Chest Pain Based On Age Category



In [60]: *#Bar Plot Creation of Person's Resting Blood Pressure (mm Hg) using Matplotlib and Seabo*

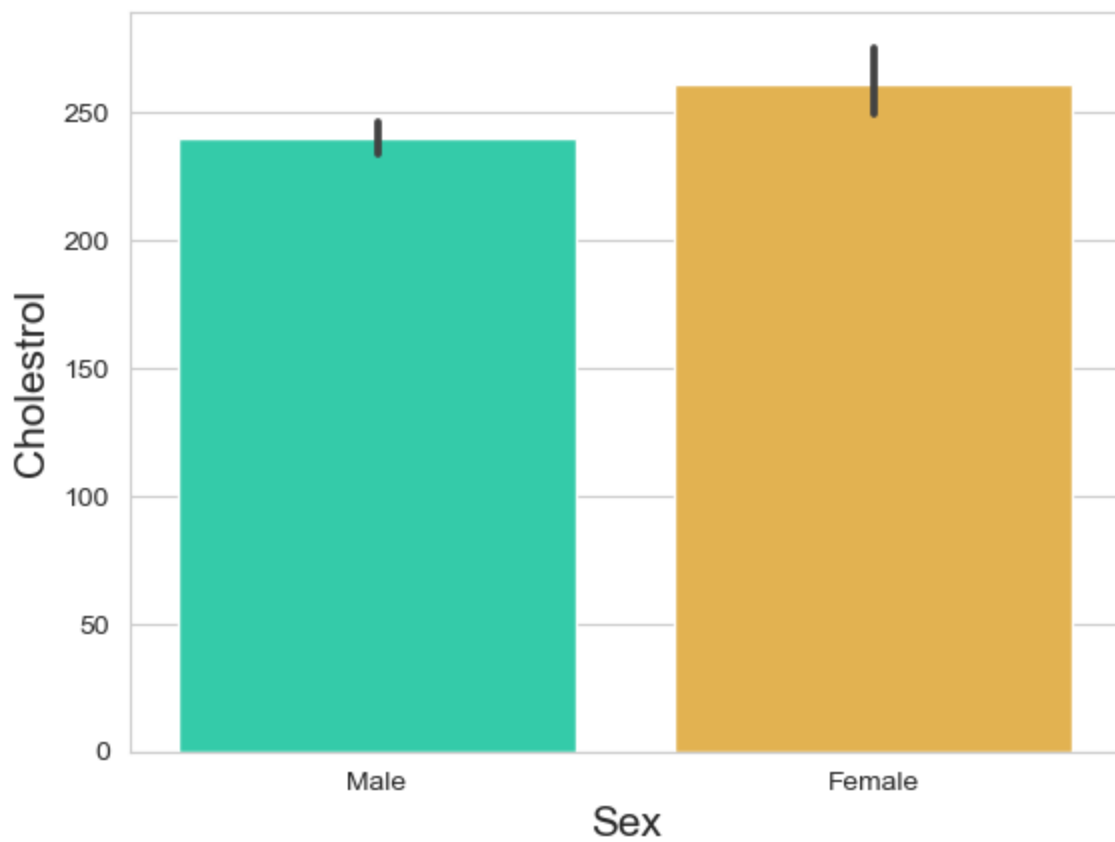
```
sns.barplot(x='sex1', y='trestbps', data=NHdata, palette='plasma')
plt.title("Blood Pressure", fontsize=17)
plt.xlabel('Sex', fontsize=10)
plt.ylabel("Person's Resting Blood Pressure (mm Hg)", fontsize=8)
plt.show()
```



In [62]: *#Bar Plot Creation of Cholestrol Level Based On Gender using Matplotlib and Seaborn*

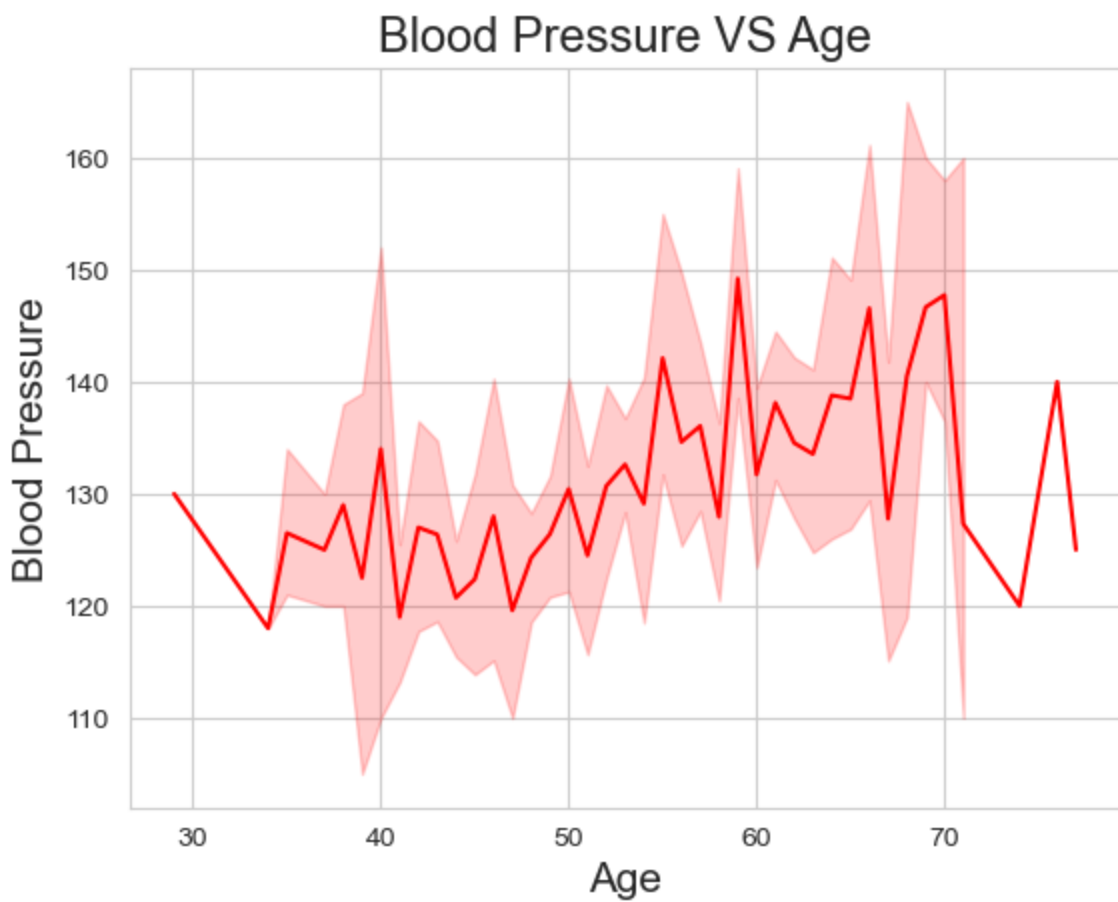
```
sns.barplot(x='sex1', y='chol', data=NHdata, palette='turbo')
plt.title("Cholestrol Level Based On Gender", fontsize=17)
plt.xlabel('Sex', fontsize=15)
plt.ylabel("Cholestrol", fontsize=15)
plt.show()
```


Cholesterol Level Based On Gender



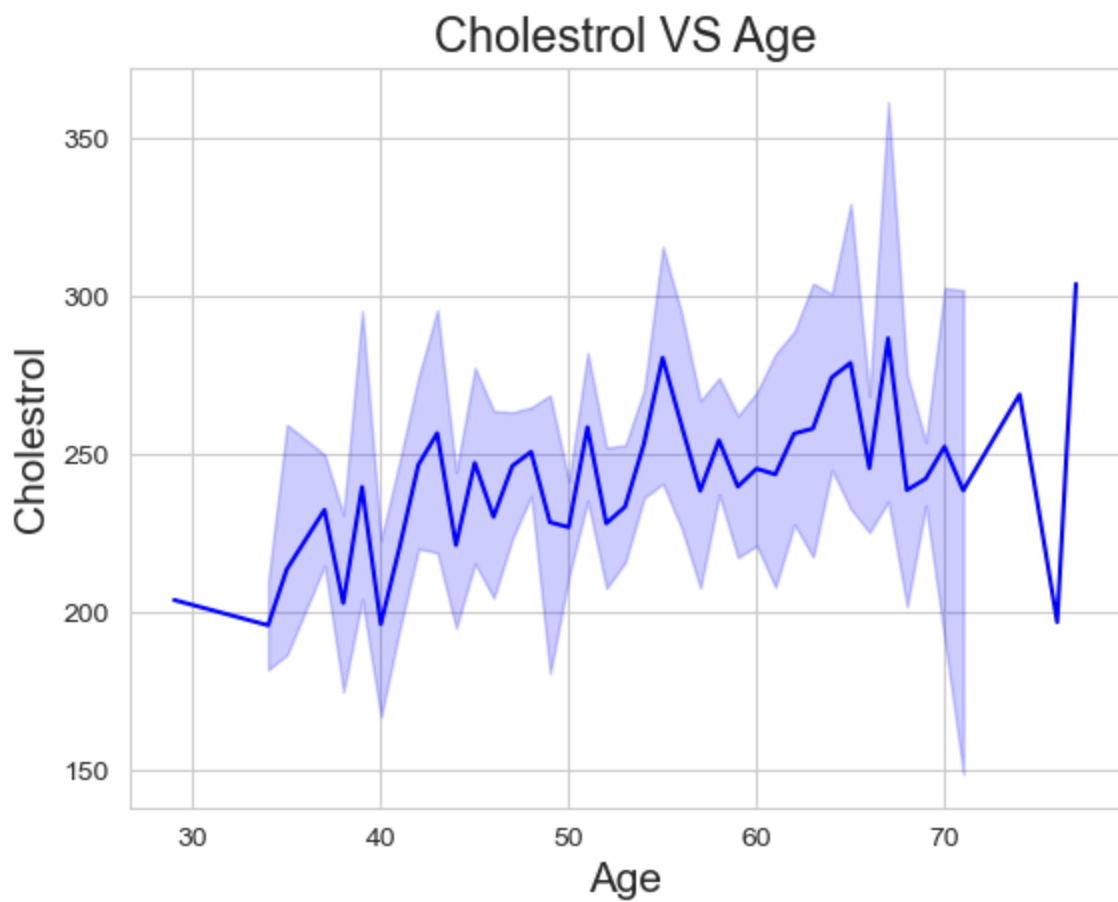
In [63]: *#Line Plot Creation of Blood Pressure VS Age using Matplotlib and Seaborn*

```
sns.lineplot(x='age', y='trestbps', data=NHdata, color='r')
plt.title('Blood Pressure VS Age', fontsize=17)
plt.xlabel('Age', fontsize=15)
plt.ylabel('Blood Pressure', fontsize=15)
plt.show()
```



In [64]: *#Line Plot Creation of Cholestrol VS Age using Matplotlib and Seaborn*

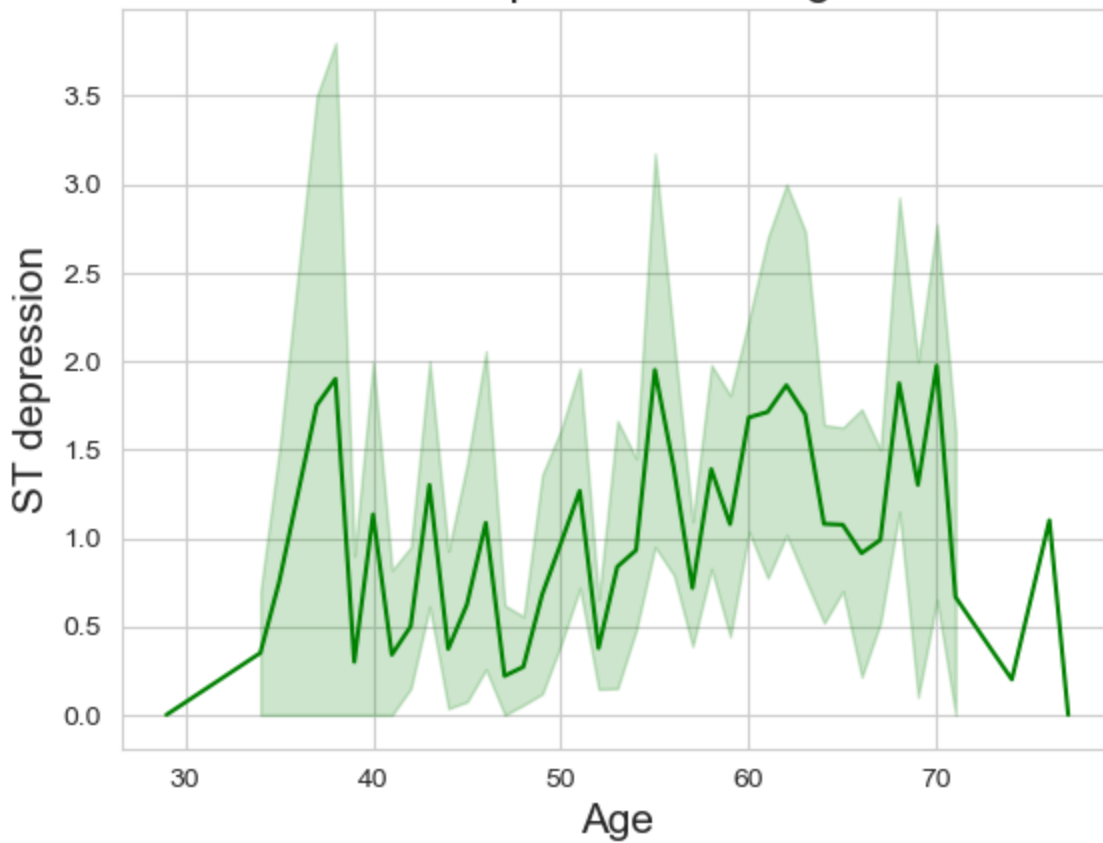
```
sns.lineplot(x='age', y='chol', data=NHdata, color='b')
plt.title('Cholestrol VS Age', fontsize=17)
plt.xlabel('Age', fontsize=15)
plt.ylabel('Cholestrol', fontsize=15)
plt.show()
```



In [65]: *#Line Plot Creation of ST Depression VS Age using Matplotlib and Seaborn*

```
sns.lineplot(x='age', y='oldpeak', data=NHdata, color='g')
plt.title('ST Depression VS Age', fontsize=17)
plt.xlabel('Age', fontsize=15)
plt.ylabel('ST depression', fontsize=15)
plt.show()
```

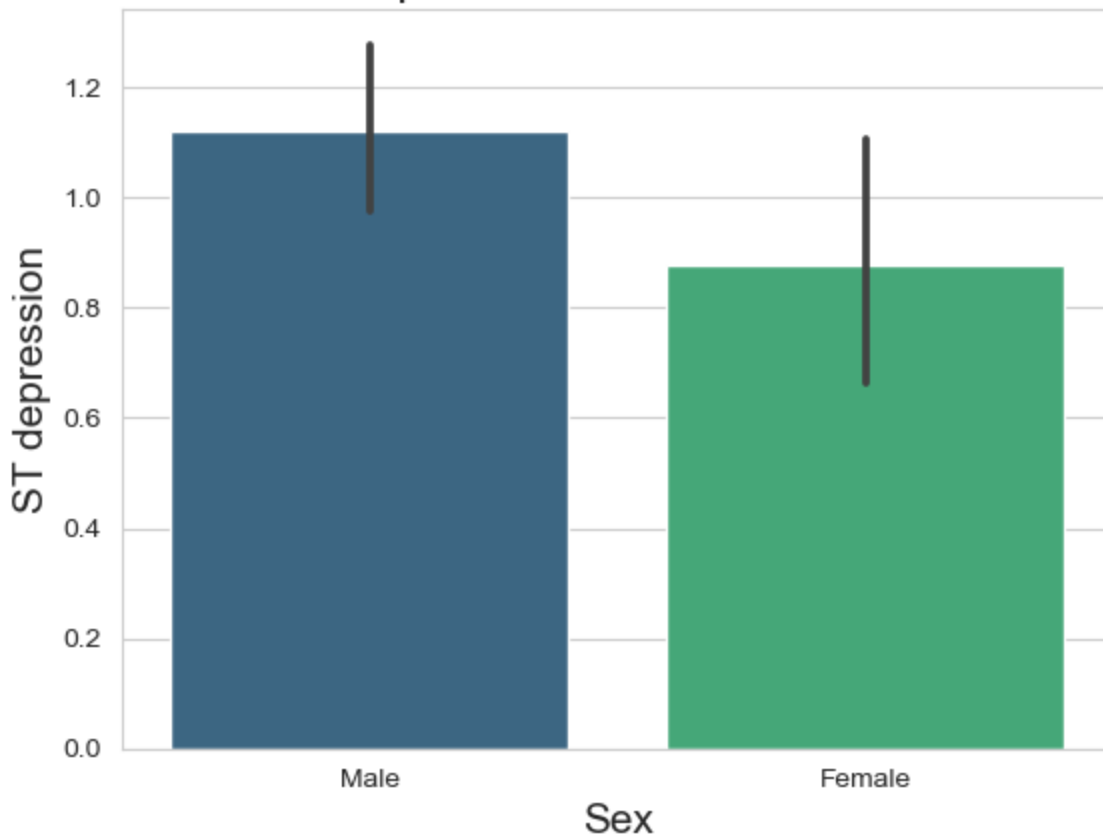
ST Depression VS Age



In [66]: *#Bar Plot Creation of ST depression VS Heart Disease using Matplotlib and Seaborn*

```
sns.barplot(x='sex1', y='oldpeak', data=NHdata, palette='viridis')
plt.title('ST depression VS Heart Disease', fontsize=17)
plt.xlabel('Sex', fontsize=15)
plt.ylabel('ST depression', fontsize=15)
plt.show()
```

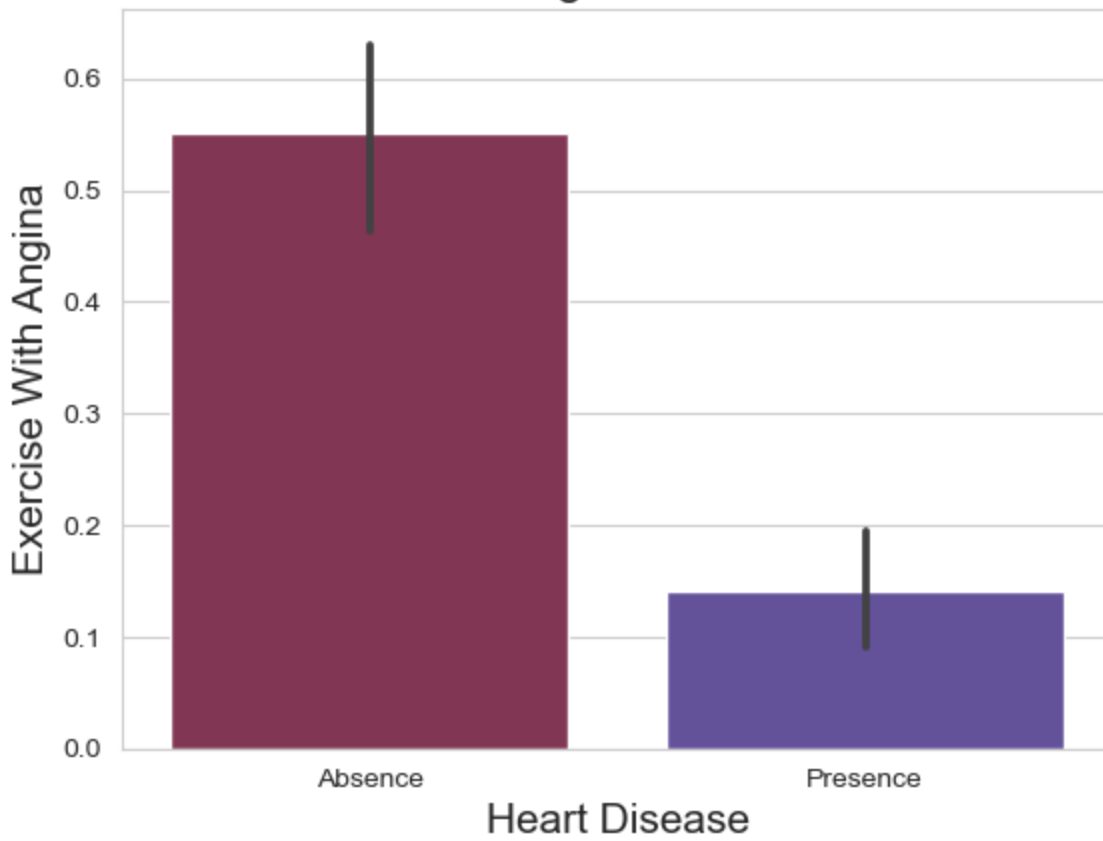
ST depression VS Heart Disease



In [67]: *#Bar Plot Creation of Exercise With Angina VS Heart Disease using Matplotlib and Seaborn*

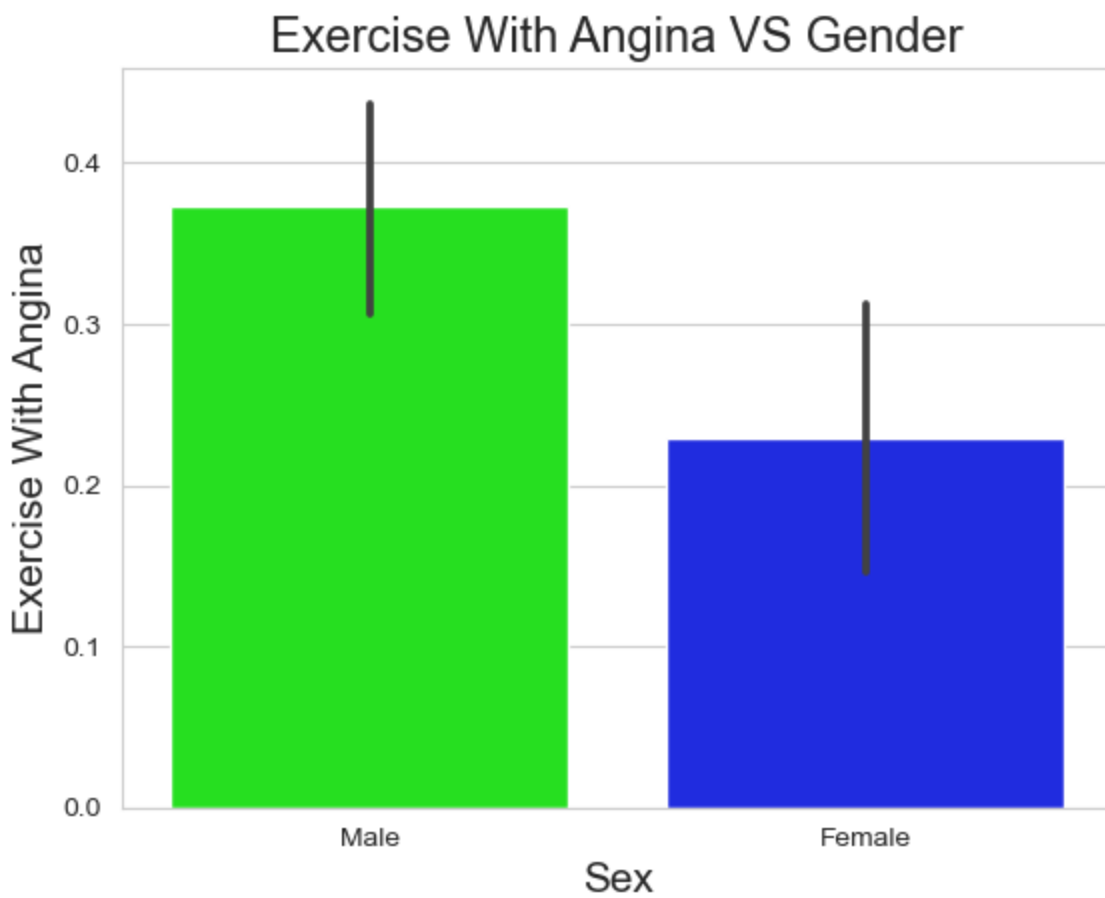
```
sns.barplot(x='Heart_Disease', y='exang', data=NHdata, palette='twilight_r')
plt.title('Exercise With Angina VS Heart Disease', fontsize=17)
plt.xlabel('Heart Disease', fontsize=15)
plt.ylabel('Exercise With Angina', fontsize=15)
plt.show()
```

Exercise With Angina VS Heart Disease



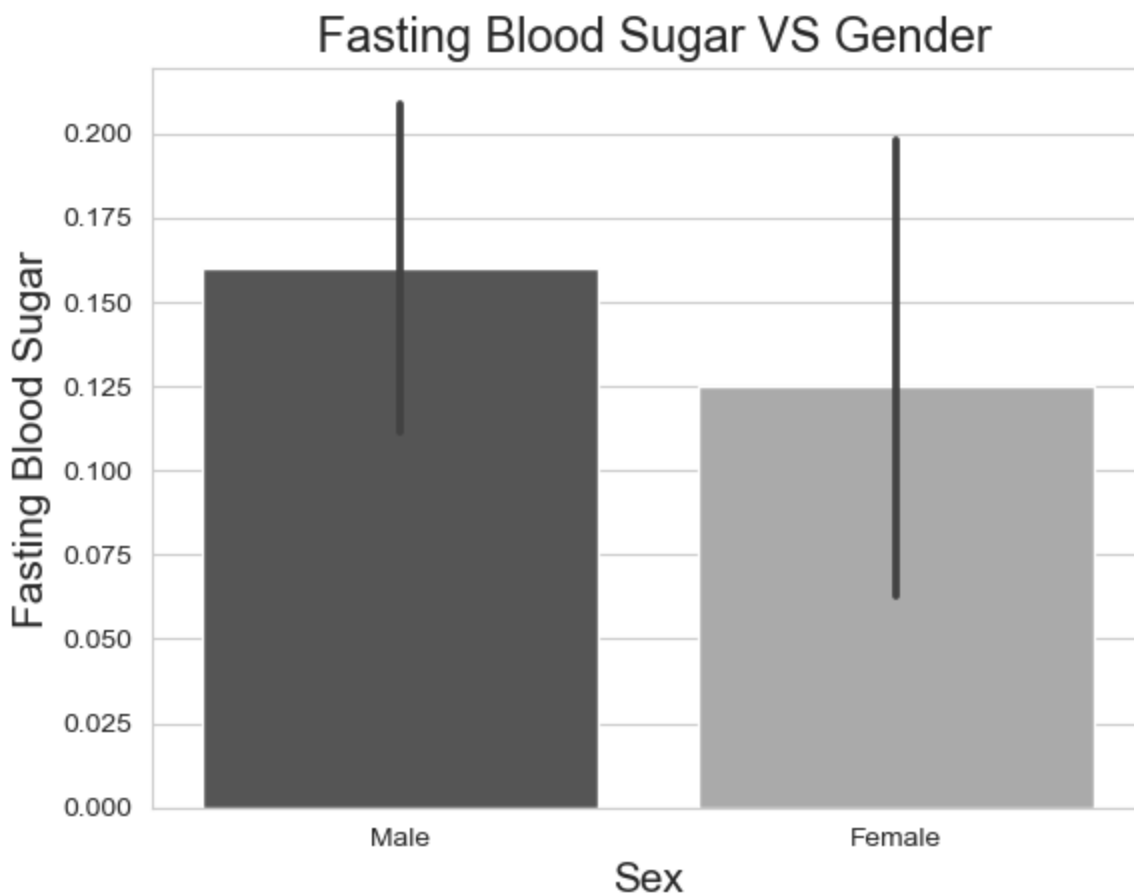
In [68]: *#Bar Plot Creation of Exercise With Angina VS Gender using Matplotlib and Seaborn*

```
sns.barplot(x='sex1', y='exang', data=NHdata, palette='hsv')
plt.title('Exercise With Angina VS Gender', fontsize=17)
plt.xlabel('Sex', fontsize=15)
plt.ylabel('Exercise With Angina', fontsize=15)
plt.show()
```



In [69]: *#Bar Plot Creation of Fasting Blood Sugar VS Gender using Matplotlib and Seaborn*

```
sns.barplot(y='fbs', x='sex1', data=NHdata, palette='binary_r')
plt.title(' Fasting Blood Sugar VS Gender', fontsize=17)
plt.xlabel('Sex', fontsize=15)
plt.ylabel('Fasting Blood Sugar', fontsize=15)
plt.show()
```



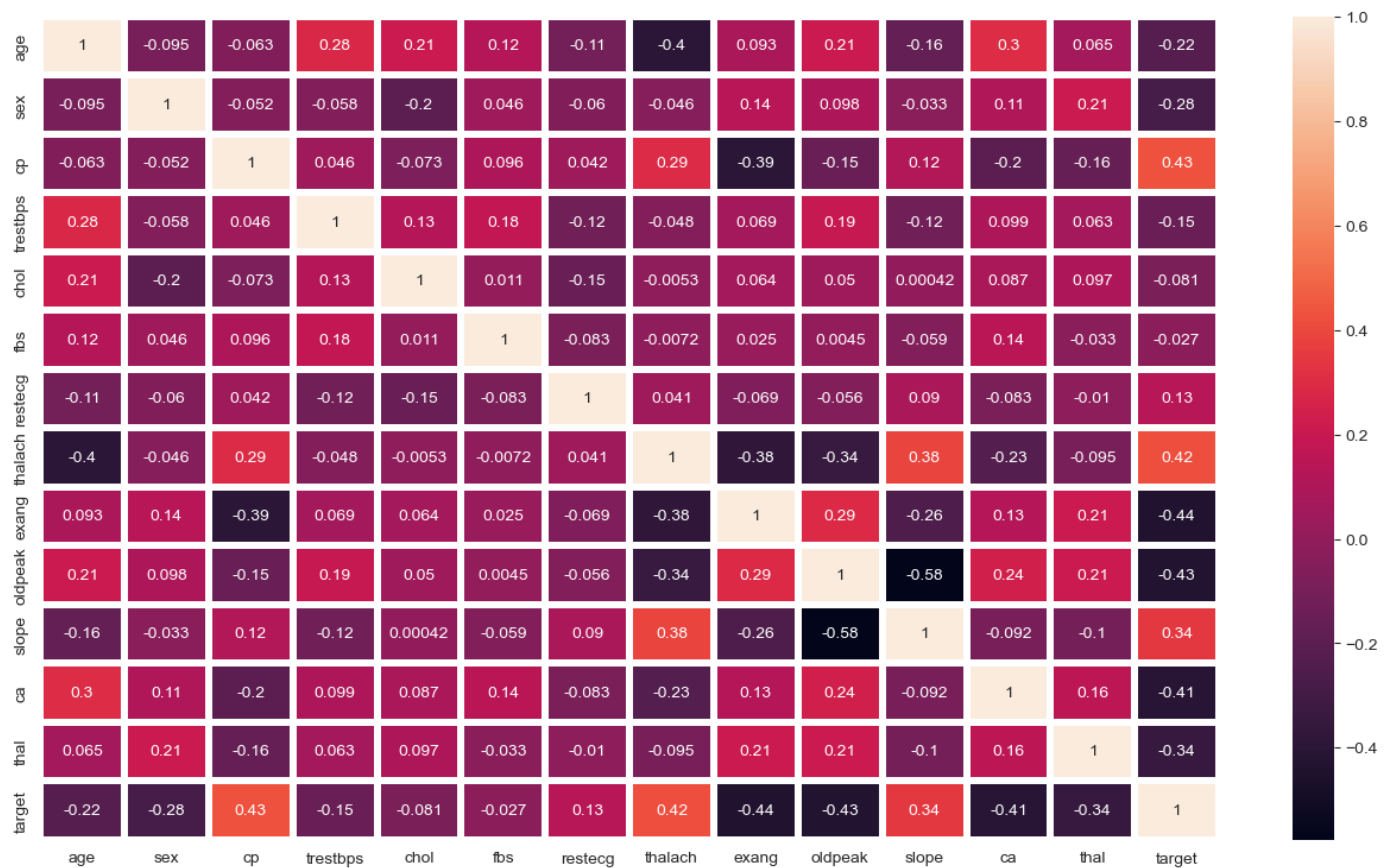
In [70]: *#Heatmap Creation using Seaborn*

```
plt.figure(figsize=(16,9))  
sns.heatmap(NHdata.corr(), annot=True, linewidth=3)
```

C:\Users\nived\AppData\Local\Temp\ipykernel_5832\3001713304.py:4: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.

```
sns.heatmap(NHdata.corr(), annot=True, linewidth=3)
```

Out[70]: <Axes: >



In []:

In [71]: `NHdata.to_excel(r"C:\Users\nived\OneDrive\Desktop\Heart Disease data2.xlsx")`

In [72]: `NHdata`

Out[72]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target	Heart_Disease
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0	Absence
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0	Absence
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0	Absence
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0	Absence
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0	Absence
...
723	68	0	2	120	211	0	0	115	0	1.5	1	0	2	1	Presence
733	44	0	2	108	141	0	1	175	0	0.6	1	0	2	1	Presence
739	52	1	0	128	255	0	1	161	1	0.0	2	1	3	0	Absence
843	59	1	3	160	273	0	0	125	0	0.0	2	0	2	0	Absence
878	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0	Absence

302 rows × 17 columns

In []: