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

```
df=pd.read_csv("heart_disease_dataset.csv")
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 16 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Age                      1000 non-null   int64
 1   Gender                   1000 non-null   object
 2   Cholesterol              1000 non-null   int64
 3   Blood Pressure           1000 non-null   int64
 4   Heart Rate               1000 non-null   int64
 5   Smoking                  1000 non-null   object
 6   Alcohol Intake           660 non-null    object
 7   Exercise Hours           1000 non-null   int64
 8   Family History           1000 non-null   object
 9   Diabetes                 1000 non-null   object
 10  Obesity                  1000 non-null   object
 11  Stress Level             1000 non-null   int64
 12  Blood Sugar              1000 non-null   int64
 13  Exercise Induced Angina  1000 non-null   object
 14  Chest Pain Type          1000 non-null   object
 15  Heart Disease            1000 non-null   int64
dtypes: int64(8), object(8)
memory usage: 125.1+ KB
```

```
df.describe()
```

	Age	Cholesterol	Blood Pressure	Heart Rate	Exercise Hours	Stress Level	Blood Sugar	Heart Disease
count	1000.000000	1000.000000	1000.0000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000
mean	52.293000	249.939000	135.2810	79.204000	4.529000	5.646000	134.941000	0.392000
std	15.727126	57.914673	26.3883	11.486092	2.934241	2.831024	36.699624	0.488441
min	25.000000	150.000000	90.0000	60.000000	0.000000	1.000000	70.000000	0.000000
25%	39.000000	200.000000	112.7500	70.000000	2.000000	3.000000	104.000000	0.000000
50%	52.000000	248.000000	136.0000	79.000000	4.500000	6.000000	135.000000	0.000000
75%	66.000000	299.000000	159.0000	89.000000	7.000000	8.000000	167.000000	1.000000
max	79.000000	349.000000	179.0000	99.000000	9.000000	10.000000	199.000000	1.000000

```
df.head()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina	
0	75	Female	228	119	66	Current	Heavy	1	No	No	Yes	8	119	Yes	
1	48	Male	204	165	62	Current	NaN	5	No	No	No	9	70	Yes	
2	53	Male	234	91	67	Never	Heavy	3	Yes	No	Yes	5	196	Yes	
3	69	Female	192	90	72	Current	NaN	4	No	Yes	No	7	107	Yes	
4	62	Female	172	163	93	Never	NaN	6	No	Yes	No	2	183	Yes	A

Next steps: Generate code with df New interactive sheet

```
df.head(1)
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
0	75	Female	228	119	66	Current	Heavy	1	No	No	Yes	8	119	Yes

Next steps: Generate code with df New interactive sheet

```
df.tail()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
995	56	Female	269	111	86	Never	Heavy	5	No	Yes	Yes	10	120	No
996	78	Female	334	145	76	Never	NaN	6	No	No	No	10	196	Yes
997	79	Male	151	179	81	Never	Moderate	4	Yes	No	Yes	8	189	Yes
998	60	Female	326	151	68	Former	NaN	8	Yes	Yes	No	5	174	Yes
999	53	Male	226	116	82	Current	NaN	6	No	No	Yes	5	161	Yes

```
df.sample()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
426	79	Male	221	99	91	Current	Moderate	0	Yes	No	No	7	186	No

```
df.shape
```

```
(1000, 16)
```

```
df.dtypes
```

	0
Age	int64
Gender	object
Cholesterol	int64
Blood Pressure	int64
Heart Rate	int64
Smoking	object
Alcohol Intake	object
Exercise Hours	int64
Family History	object
Diabetes	object
Obesity	object
Stress Level	int64
Blood Sugar	int64
Exercise Induced Angina	object
Chest Pain Type	object
Heart Disease	int64

dtype: object

```
df.columns
```

```
Index(['Age', 'Gender', 'Cholesterol', 'Blood Pressure', 'Heart Rate',
       'Smoking', 'Alcohol Intake', 'Exercise Hours', 'Family History',
       'Diabetes', 'Obesity', 'Stress Level', 'Blood Sugar',
       'Exercise Induced Angina', 'Chest Pain Type', 'Heart Disease'],
      dtype='object')
```

```
df.index
```

```
RangeIndex(start=0, stop=1000, step=1)
```

```
df.isnull().sum()
```

	0
Age	0
Gender	0
Cholesterol	0
Blood Pressure	0
Heart Rate	0
Smoking	0
Alcohol Intake	340
Exercise Hours	0
Family History	0
Diabetes	0
Obesity	0
Stress Level	0
Blood Sugar	0
Exercise Induced Angina	0
Chest Pain Type	0
Heart Disease	0

dtype: int64

```
df.notnull()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
0	True	True	True	True	True	True	True	True	True	True	True	True	True	True
1	True	True	True	True	True	True	False	True	True	True	True	True	True	True
2	True	True	True	True	True	True	True	True	True	True	True	True	True	True
3	True	True	True	True	True	True	False	True	True	True	True	True	True	True
4	True	True	True	True	True	True	False	True	True	True	True	True	True	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	True	True	True	True	True	True	True	True	True	True	True	True	True	True
996	True	True	True	True	True	True	False	True	True	True	True	True	True	True
997	True	True	True	True	True	True	True	True	True	True	True	True	True	True
998	True	True	True	True	True	True	False	True	True	True	True	True	True	True
999	True	True	True	True	True	True	False	True	True	True	True	True	True	True

1000 rows × 16 columns

```
df.dropna()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
0	75	Female	228	119	66	Current	Heavy	1	No	No	Yes	8	119	Yes
2	53	Male	234	91	67	Never	Heavy	3	Yes	No	Yes	5	196	Yes
6	64	Female	211	105	86	Former	Heavy	8	Yes	Yes	Yes	2	120	No
7	60	Female	208	148	83	Never	Moderate	4	No	Yes	Yes	2	113	Yes
8	37	Female	317	137	66	Current	Heavy	3	No	Yes	Yes	5	114	No
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
991	26	Female	215	100	74	Never	Heavy	7	No	Yes	No	10	135	No
992	28	Female	220	102	73	Current	Moderate	7	Yes	Yes	Yes	10	102	No
994	52	Male	248	159	76	Former	Moderate	9	No	Yes	Yes	2	152	Yes
995	56	Female	269	111	86	Never	Heavy	5	No	Yes	Yes	10	120	No
997	79	Male	151	179	81	Never	Moderate	4	Yes	No	Yes	8	189	Yes

60 rows × 16 columns

```
df['Alcohol Intake']=df['Alcohol Intake'].fillna(df['Alcohol Intake'].mode(),inplace=True)
```

```
/tmp/ipython-input-2743795674.py:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chair
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are set

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = c

  df['Alcohol Intake']=df['Alcohol Intake'].fillna(df['Alcohol Intake'].mode(),inplace=True)
```

```
df
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
0	75	Female	228	119	66	Current	Heavy	1	No	No	Yes	8	119	Yes
1	48	Male	204	165	62	Current	NaN	5	No	No	No	9	70	Yes
2	53	Male	234	91	67	Never	Heavy	3	Yes	No	Yes	5	196	Yes
3	69	Female	192	90	72	Current	NaN	4	No	Yes	No	7	107	Yes
4	62	Female	172	163	93	Never	NaN	6	No	Yes	No	2	183	Yes
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	56	Female	269	111	86	Never	Heavy	5	No	Yes	Yes	10	120	No
996	78	Female	334	145	76	Never	NaN	6	No	No	No	10	196	Yes
997	79	Male	151	179	81	Never	Moderate	4	Yes	No	Yes	8	189	Yes
998	60	Female	326	151	68	Former	NaN	8	Yes	Yes	No	5	174	Yes
999	53	Male	226	116	82	Current	NaN	6	No	No	Yes	5	161	Yes

1000 rows × 16 columns

Next steps: Generate code with df New interactive sheet

```
df.isnull()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina
)	False	False	False	False	False	False	False	False	False	False	False	False	False	False
	False	False	False	False	False	False	True	False	False	False	False	False	False	False
!	False	False	False	False	False	False	False	False	False	False	False	False	False	False
!	False	False	False	False	False	False	True	False	False	False	False	False	False	False
!	False	False	False	False	False	False	True	False	False	False	False	False	False	False
.	...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	False	False	False	False	False	False	False	False	False	False	False	False	False	False
96	False	False	False	False	False	False	True	False	False	False	False	False	False	False
97	False	False	False	False	False	False	False	False	False	False	False	False	False	False
98	False	False	False	False	False	False	True	False	False	False	False	False	False	False
99	False	False	False	False	False	False	True	False	False	False	False	False	False	False

00 rows × 16 columns

```
df.isnull().mode()*100
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Alcohol Intake	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina	C
0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	

```
df=df.drop('Alcohol Intake',axis=1)
df
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina	Chest P T
0	75	Female	228	119	66	Current	1	No	No	Yes	8	119	Yes	Aty An
1	48	Male	204	165	62	Current	5	No	No	No	9	70	Yes	Ty An
2	53	Male	234	91	67	Never	3	Yes	No	Yes	5	196	Yes	Aty An
3	69	Female	192	90	72	Current	4	No	Yes	No	7	107	Yes	Non-ang
4	62	Female	172	163	93	Never	6	No	Yes	No	2	183	Yes	Asympton
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
995	56	Female	269	111	86	Never	5	No	Yes	Yes	10	120	No	Non-ang
996	78	Female	334	145	76	Never	6	No	No	No	10	196	Yes	Ty An
997	79	Male	151	179	81	Never	4	Yes	No	Yes	8	189	Yes	Asympton
998	60	Female	326	151	68	Former	8	Yes	Yes	No	5	174	Yes	Aty An
999	53	Male	226	116	82	Current	6	No	No	Yes	5	161	Yes	Asympton

1000 rows × 15 columns

Next steps: Generate code with df New interactive sheet

```
df.nunique()
```

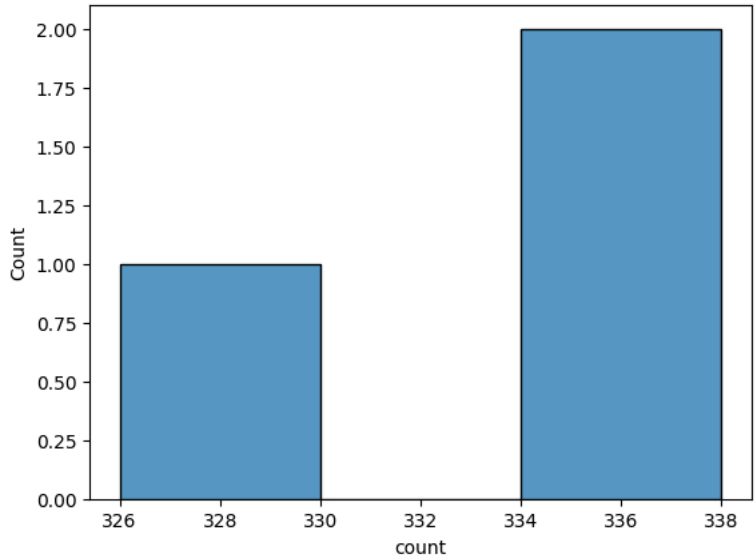
	0
Age	55
Gender	2
Cholesterol	200
Blood Pressure	90
Heart Rate	40
Smoking	3
Exercise Hours	10
Family History	2
Diabetes	2
Obesity	2
Stress Level	10
Blood Sugar	130
Exercise Induced Angina	2
Chest Pain Type	4
Heart Disease	2

dtype: int64

```
a=df['Smoking'].value_counts()
```

```
sns.histplot(a)
```

```
<Axes: xlabel='count', ylabel='Count'>
```

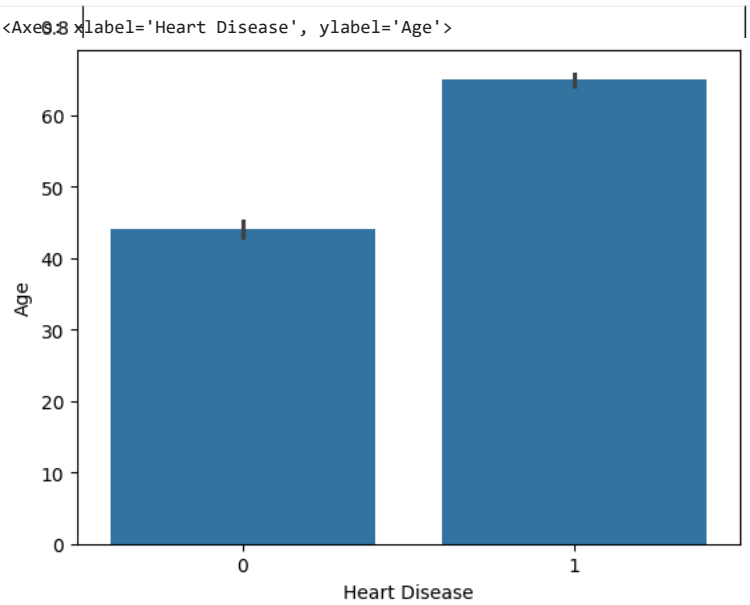



```
sns.scatterplot(df['Heart Disease'])
```

```
<Axes: ylabel='Heart Disease'>
```


```
sns.barplot(x='Heart Disease',y='Age',data=df)
```

```
<Axes: xlabel='Heart Disease', ylabel='Age'>
```




```
df.columns
```

```
Index(['Age', 'Gender', 'Cholesterol', 'Blood Pressure', 'Heart Rate',
       'Smoking', 'Exercise Hours', 'Family History', 'Diabetes', 'Obesity',
       'Stress Level', 'Blood Sugar', 'Exercise Induced Angina',
       'Chest Pain Type', 'Heart Disease'],
      dtype='object')
```

```
df.head()
```

	Age	Gender	Cholesterol	Blood Pressure	Heart Rate	Smoking	Exercise Hours	Family History	Diabetes	Obesity	Stress Level	Blood Sugar	Exercise Induced Angina	Chest Pai Typ
0	75	Female	228	119	66	Current	1	No	No	Yes	8	119	Yes	Atypic Angir
1	48	Male	204	165	62	Current	5	No	No	No	9	70	Yes	Typic Angir
2	53	Male	234	91	67	Never	3	Yes	No	Yes	5	196	Yes	Atypic Angir
3	69	Female	192	90	72	Current	4	No	Yes	No	7	107	Yes	Non-angin Pa
4	62	Female	172	163	93	Never	6	No	Yes	No	2	183	Yes	Asymptomat

Next steps: Generate code with df New interactive sheet

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
```

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()

categorical_cols = ['Gender', 'Smoking', 'Family History', 'Diabetes', 'Obesity', 'Exercise Induced Angina', 'Chest Pain Type
for col in categorical_cols:
    df[col] = le.fit_transform(df[col])
```

```
x=df.drop('Heart Disease',axis=1)
y=df['Heart Disease']

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```