In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
df = pd.read_csv("heart.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak
0	40	M	ATA	140	289	0	Normal	172	N	0.0
1	49	F	NAP	160	180	0	Normal	156	N	1.0
2	37	M	ATA	130	283	0	ST	98	N	0.0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5
4	54	M	NAP	150	195	0	Normal	122	N	0.0

In [4]:
```
df.info()
```

Out[4]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Age             918 non-null    int64  
 1   Sex             918 non-null    object 
 2   ChestPainType   918 non-null    object 
 3   RestingBP       918 non-null    int64  
 4   Cholesterol     918 non-null    int64  
 5   FastingBS       918 non-null    int64  
 6   RestingECG      918 non-null    object 
 7   MaxHR           918 non-null    int64  
 8   ExerciseAngina  918 non-null    object 
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object 
 11  HeartDisease    918 non-null    int64  
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

In [5]:
```
df.describe()
```

Out[5]:

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

In [6]:
```
df.isnull().sum()
```

Out[6]:
```
Age               0
Sex               0
ChestPainType     0
RestingBP         0
Cholesterol       0
FastingBS         0
RestingECG        0
MaxHR             0
ExerciseAngina    0
Oldpeak           0
ST_Slope          0
HeartDisease      0
dtype: int64
```

In [7]:
```
df.shape
```

Out[7]:
```
(918, 12)
```

In [8]:
```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
from sklearn.preprocessing import MinMaxScaler
scalar = MinMaxScaler()
```

In [9]:
```
df['Sex'] = label_encoder.fit_transform(df['Sex'])
df['ChestPainType'] = label_encoder.fit_transform(df['ChestPainType'])
df['RestingECG'] = label_encoder.fit_tranaform(df['RestingECG'])
df['ExerciseAngina'] = label_encoder.fit_transform(df['ExerciseAngina'])
df['ST_Slope'] = label_encoder.fit_transform(df['ST_Slope'])
```

In [10]:
```
df
```

Out[10]:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpe
0	40	1	1	140	289	0	1	172	0	
1	49	0	2	160	180	0	1	156	0	
2	37	1	1	130	283	0	2	98	0	
3	48	0	0	138	214	0	1	108	1	
4	54	1	2	150	195	0	1	122	0	
...	...	...	...	...	...	...	...	...	...	...
913	45	1	3	110	264	0	1	132	0	
914	68	1	0	144	193	1	1	141	0	
915	57	1	0	130	131	0	1	115	1	
916	57	0	1	130	236	0	0	174	0	
917	38	1	2	138	175	0	1	173	0	

918 rows × 12 columns

In [11]:
```
def remove_outliers_iqr(df):
    # Calculate the first quartile (Q1) and third quartile (Q3)
    q1 = np.percentile(df, 25)
    q3 = np.percentile(df, 75)

    # Calculate the interquartile range (IQR)
    iqr = q3 - q1

    # Define the lower and upper bounds for outliers
    lower_bound = q1 - 1.5 * iqr
    upper_bound = q3 + 1.5 * iqr

    # Remove outliers
    df = [x for x in df if lower_bound <= x <= upper_bound]

    return df
```

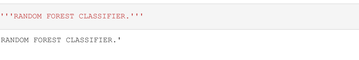
In [ ]:
```

```

In [12]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

In [13]:
```
plt.figure(figsize=(20,10))
sns.heatmap(df.corr(), annot=True, fmt='.2f')
```

Out[13]:
```
<AxesSubplot:>
```



In [14]:
```
df['ST_Slope'].value_counts()
```

Out[14]:
```
1    460
2    395
0     63
Name: ST_Slope, dtype: int64
```

In [15]:
```
df.drop(['ChestPainType','MaxHR'], inplace = True, axis=1)
```

In [16]:
```
df
```

Out[16]:

	Age	Sex	RestingBP	Cholesterol	FastingBS	RestingECG	ExerciseAngina	Oldpeak	ST_Slope	HeartDisea
0	40	1	140	289	0	1	0	0.0	2	
1	49	0	160	180	0	1	0	1.0	1	
2	37	1	130	283	0	2	0	0.0	2	
3	48	0	138	214	0	1	1	1.5	1	
4	54	1	150	195	0	1	0	0.0	2	
...	...	...	...	...	...	...	...	...	...	...
913	45	1	110	264	0	1	0	1.2	1	
914	68	1	144	193	1	1	0	3.4	1	
915	57	1	130	131	0	1	1	1.2	1	
916	57	0	130	236	0	0	0	0.0	1	
917	38	1	138	175	0	1	0	0.0	2	

918 rows × 10 columns

In [17]:
```
from sklearn.model_selection import train_test_split
X = df.drop('HeartDisease', axis = 1)
y = df['HeartDisease']
```

In [18]:
```
scalar.fit(X)
```

Out[18]:
```
MinMaxScaler()
```

In [19]:
```
standardised_data = scalar.transform(X)
```

In [20]:
```
X = standardised_data
y = df['HeartDisease']
```

In [21]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=
```

In [22]:
```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=20, criterion='entropy', random_state=42)
```

In [23]:
```
rf.fit(X_train, y_train)
```

Out[23]:
```
RandomForestClassifier(criterion='entropy', n_estimators=20, random_state=42)
```

In [24]:
```
y_pred = rf.predict(X_test)
```

In [25]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

Out[25]:
```
0.8586956521739131
```

In [26]:
```
'''RANDOM FOREST CLASSIFIER.'''
```

Out[26]:
```
'RANDOM FOREST CLASSIFIER.'
```