In [21]:

```
import pandas as pd
import numpy as np
import seaborn as sns
df = pd.read_csv(r"C:\SEM 5\Dataset\Heart.csv")
df
```

Out[21]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Old
0	1	63	1	typical	145	233	1	2	150	0	
1	2	67	1	asymptomatic	160	286	0	2	108	1	
2	3	67	1	asymptomatic	120	229	0	2	129	1	
3	4	37	1	nonanginal	130	250	0	0	187	0	
4	5	41	0	nontypical	130	204	0	2	172	0	
...	...	...	...	...	...	...	...	...	...	...	
298	299	45	1	typical	110	264	0	0	132	0	
299	300	68	1	asymptomatic	144	193	1	0	141	0	
300	301	57	1	asymptomatic	130	131	0	0	115	1	
301	302	57	0	nontypical	130	236	0	2	174	0	
302	303	38	1	nonanginal	138	175	0	0	173	0	

303 rows × 15 columns

In [22]:

```
df.columns
```

Out[22]:

```
Index(['Unnamed: 0', 'Age', 'Sex', 'ChestPain', 'RestBP', 'Chol', 'Fbs',
       'RestECG', 'MaxHR', 'ExAng', 'Oldpeak', 'Slope', 'Ca', 'Thal', 'AH
D'],
      dtype='object')
```

In [23]:

```
df.head()
```

Out[23]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpe
0	1	63	1	typical	145	233	1	2	150	0	2
1	2	67	1	asymptomatic	160	286	0	2	108	1	1
2	3	67	1	asymptomatic	120	229	0	2	129	1	2
3	4	37	1	nonanginal	130	250	0	0	187	0	3
4	5	41	0	nontypical	130	204	0	2	172	0	1

In [24]:

```
df.shape
```

Out[24]:

```
(303, 15)
```

In [25]:

```
df.dtypes
```

Out[25]:

```
Unnamed: 0      int64
Age             int64
Sex             int64
ChestPain      object
RestBP          int64
Chol            int64
Fbs             int64
RestECG         int64
MaxHR           int64
ExAng           int64
Oldpeak       float64
Slope           int64
Ca            float64
Thal           object
AHD            object
dtype: object
```

In [26]:

```
df.describe()
```

Out[26]:

	Unnamed: 0	Age	Sex	RestBP	Chol	Fbs	RestECG
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000
mean	152.000000	54.438944	0.679868	131.689769	246.693069	0.148515	0.990099
std	87.612784	9.038662	0.467299	17.599748	51.776918	0.356198	0.994971
min	1.000000	29.000000	0.000000	94.000000	126.000000	0.000000	0.000000
25%	76.500000	48.000000	0.000000	120.000000	211.000000	0.000000	0.000000
50%	152.000000	56.000000	1.000000	130.000000	241.000000	0.000000	1.000000
75%	227.500000	61.000000	1.000000	140.000000	275.000000	0.000000	2.000000
max	303.000000	77.000000	1.000000	200.000000	564.000000	1.000000	2.000000

In [27]:

```
df.isna().sum()
```

Out[27]:

```
Unnamed: 0    0
Age           0
Sex           0
ChestPain     0
RestBP        0
Chol          0
Fbs           0
RestECG       0
MaxHR         0
ExAng         0
Oldpeak       0
Slope         0
Ca            4
Thal          2
AHD           0
dtype: int64
```

In [34]:

```
df['Ca'].nunique()
```

Out[34]:

```
4
```

In [35]:

```
df['Thal'].nunique()
```

Out[35]:

```
3
```

In [36]:

```
df.nunique()
```

Out[36]:

```
Unnamed: 0    303
Age            41
Sex             2
ChestPain       4
RestBP         50
Chol          152
Fbs             2
RestECG         3
MaxHR          91
ExAng           2
Oldpeak        40
Slope           3
Ca              4
Thal            3
AHD             2
dtype: int64
```

In [38]:

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

Out[38]:

```
Unnamed: 0    0
Age           0
Sex           0
ChestPain     0
RestBP        0
Chol          0
Fbs           0
RestECG       0
MaxHR         0
ExAng         0
Oldpeak       0
Slope         0
Ca            4
Thal          2
AHD           0
dtype: int64
```

In [39]:

```
df['Ca'].fillna(method="ffill",inplace=True)
```

In [41]:

```
df['Thal'].fillna(method="ffill",inplace=True)
```

In [42]:

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

Out[42]:

```
Unnamed: 0    0
Age           0
Sex           0
ChestPain     0
RestBP        0
Chol          0
Fbs           0
RestECG       0
MaxHR         0
ExAng         0
Oldpeak       0
Slope         0
Ca            0
Thal          0
AHD           0
dtype: int64
```

In [43]:

```
duplicate=df.duplicated().sum()
if duplicate:
    print("Duplicated row{}".format(duplicate))
else:
    print("No duplicate")
```

```
No duplicate
```

In [44]:

```
df['Ca'].nunique()
```

Out[44]:

```
4
```

In [45]:

```
df['Ca']=df['Ca'].astype('object')
```

In [46]:

```
df.dtypes
```

Out[46]:

```
Unnamed: 0      int64
Age             int64
Sex             int64
ChestPain      object
RestBP          int64
Chol            int64
Fbs             int64
RestECG         int64
MaxHR           int64
ExAng           int64
Oldpeak       float64
Slope           int64
Ca             object
Thal           object
AHD            object
dtype: object
```

In [47]:

```
print(df[df.duplicated()])
```

```
Empty DataFrame
Columns: [Unnamed: 0, Age, Sex, ChestPain, RestBP, Chol, Fbs, RestECG, Max
HR, ExAng, Oldpeak, Slope, Ca, Thal, AHD]
Index: []
```

In [48]:

```
df.isna().sum()
df=df.fillna(df.median())
df.isnull().sum()
```

Out[48]:

```
Unnamed: 0    0
Age           0
Sex           0
ChestPain     0
RestBP        0
Chol          0
Fbs           0
RestECG       0
MaxHR         0
ExAng         0
Oldpeak       0
Slope         0
Ca            0
Thal          0
AHD           0
dtype: int64
```

In [49]:

```
subset1=df[df['Sex']==0]
subset1.shape
```

Out[49]:

```
(97, 15)
```

In [50]:

```
subset2=df[df['Sex']==1]
subset2.shape
```

Out[50]:

```
(206, 15)
```

In [51]:

```
combine=[subset1,subset2]
result=pd.concat (combine)
result.shape
```

Out[51]:

```
(303, 15)
```

In [52]:

```
subset3=df[['Age','Sex','RestBP']]
subset3
```

Out[52]:

	Age	Sex	RestBP
0	63	1	145
1	67	1	160
2	67	1	120
3	37	1	130
4	41	0	130
...	...	...	...
298	45	1	110
299	68	1	144
300	57	1	130
301	57	0	130
302	38	1	138

303 rows × 3 columns

In [53]:

```
subset4=df[['Chol','Fbs']]
subset4
```

Out[53]:

```
     Chol  Fbs
  0   233    1
  1   286    0
  2   229    0
  3   250    0
  4   204    0
...   ...  ...
298   264    0
299   193    1
300   131    0
301   236    0
302   175    0
```

303 rows × 2 columns

In [54]:

```
combine1=[subset3,subset4]
result1=pd.concat(combine)
result1.shape
```

Out[54]:

```
(303, 15)
```

In [ ]:

```
```

In [119]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
classifier= KNeighborsClassifier(n_neighbors=7)
classifier.fit(x_train, y_train)
y_pred= classifier.predict(x_test)
# Convert y_pred and y_test to NumPy arrays if needed
y_pred = np.array(y_pred)
y_test = np.array(y_test)

# Flatten y_pred and y_test
y_pred = y_pred.flatten()
y_test = y_test.flatten()

confusion_matrix(y_pred,y_test)
```

Out[119]:

```
array([[14,  0, 18,  0],
       [ 0, 19, 16,  0],
       [15, 12, 27,  1],
       [ 0,  0,  0,  0]], dtype=int64)
```

In [108]:

```
result1.head()
```

Out[108]:

	Unnamed: 0	Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldp
4	5	41	0	nontypical	130	204	0	2	172	0	
6	7	62	0	asymptomatic	140	268	0	2	160	0	
7	8	57	0	asymptomatic	120	354	0	0	163	1	
11	12	56	0	nontypical	140	294	0	2	153	0	
18	19	48	0	nonanginal	130	275	0	0	139	0	

In [109]:

```
from sklearn.model_selection import train_test_split
x=df[['Age','Sex','RestBP']]
y=df[['Slope','RestECG']]
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.20,random_state=0)
```

In [110]:

```
x_train
```

Out[110]:

	Age	Sex	RestBP
74	44	1	110
153	55	1	160
64	54	1	120
296	59	1	164
287	58	1	125
...	...	...	...
251	58	1	146
192	43	1	132
117	35	0	138
47	50	1	150
172	59	0	174

242 rows × 3 columns

In [111]:

```
df[df['Ca']==0]
df.loc[df['Ca']==0,'Ca']=np.NaN
df['Ca'].unique()
```

Out[111]:

```
array([nan,  3.,  2.,  1.])
```

In [112]:

```
y_train
```

Out[112]:

	Slope	RestECG
74	1	2
153	2	2
64	2	0
296	2	2
287	2	0
...	...	...
251	2	0
192	2	2
117	1	0
47	2	2
172	2	0

242 rows × 2 columns

In [113]:

```
df[df['Ca']==0]
df.loc[df['Ca']==0,'Ca']=np.NaN
df['Ca'].unique()
```

Out[113]:

```
array([nan,  3.,  2.,  1.])
```

In [ ]:

```

```

In [114]:

```
from sklearn.tree import DecisionTreeClassifier
model=DecisionTreeClassifier(criterion='entropy',max_depth=2)
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
feature_names=df.columns[0:7]
print(feature_names,end='')
```

```
Index(['Unnamed: 0', 'Age', 'Sex', 'ChestPain', 'RestBP', 'Chol', 'Fbs'],
dtype='object')
```

In [115]:

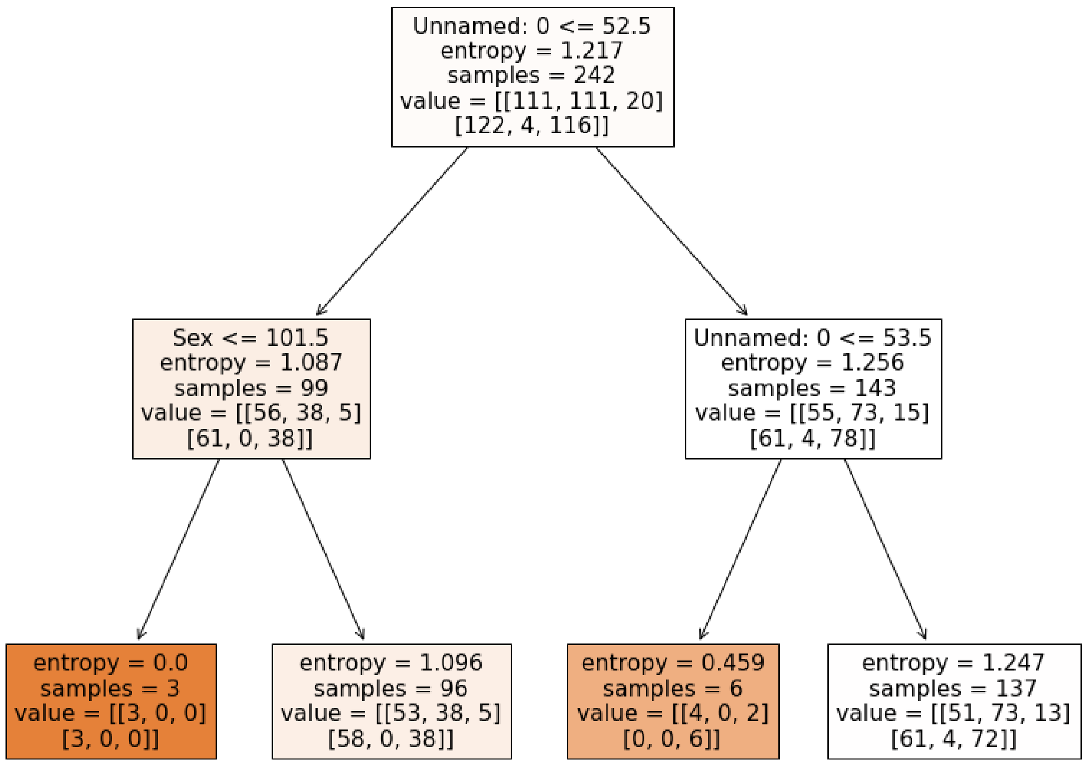
```
class_names=[str(x) for x in model.classes_]
class_names
```

Out[115]:

```
['[1 2 3]', '[0 1 2]']
```

In [116]:

```
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree
fig=plt.figure(figsize=(14,12))
plot_tree(model,feature_names=feature_names,class_names=class_names,filled=True)
plt.savefig("true visualization.png")
```



In [ ]:

In [ ]: