In [2]:
```
#Experiment No.11
```

In [4]:
```
#Aim: To Perform and Data analysis with Confusion Matrix

#Name: Sakshi Padmakar Yeole
#Class: 3rd yr(B)
#Subject:ET-II
#Roll no.:69
```

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

In [2]:
```
import os
```

In [3]:
```
os.getcwd()
```

Out[3]:
```
'C:\\Users\\hp'
```

In [4]:
```
os.chdir('C:\\Users\\hp\\Downloads')
```

In [5]:
```
data=pd.read_csv("heart.csv")
```

In [6]:
```
data.head()
```

Out[6]:

	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

In [7]:
```
data.tail()
```

Out[7]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
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

In [8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
```

```
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

In [9]:
```
data.describe()
```

Out[9]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	149.114146	0.336585	1.071512	1
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724	0.472772	1.175053	0
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000	0.000000	0.000000	0
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000	0.000000	0.000000	1
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000	0.000000	0.800000	1
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000	1.000000	1.800000	2
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.000000	1.000000	6.200000	2

In [10]:
```
data.shape
```

Out[10]:
```
(1025, 14)
```

In [11]:
```
data.size
```

Out[11]:
```
14350
```

In [12]:
```
data.size
```

Out[12]:
```
14350
```

In [13]:
```
# check Missing Value by record

data.isna()
```

Out[13]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1021	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1022	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1023	False	False	False	False	False	False	False	False	False	False	False	False	False	False
1024	False	False	False	False	False	False	False	False	False	False	False	False	False	False

1025 rows × 14 columns

In [14]:
```
data.isna().any()
```

Out[14]:
```
age         False
sex         False
cp          False
trestbps    False
chol        False
fbs         False
restecg     False
```

```
thalach     False
exang       False
oldpeak     False
slope       False
ca          False
thal        False
target      False
dtype: bool
```

In [15]:
```
data.isna().sum()
```

Out[15]:
```
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
```

In [16]:
```
x=data.drop("target", axis=1)
y=data["target"]
```

In [17]:
```
#splitting the data into training and testing data sets
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)
```

In [20]:
```
x_train
```

Out[20]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
835	49	1	2	118	149	0	0	126	0	0.8	2	3	2
137	64	0	0	180	325	0	1	154	1	0.0	2	0	2
534	54	0	2	108	267	0	0	167	0	0.0	2	0	2
495	59	1	0	135	234	0	1	161	0	0.5	1	0	3
244	51	1	2	125	245	1	0	166	0	2.4	1	0	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
700	41	1	2	130	214	0	0	168	0	2.0	1	0	2
71	61	1	0	140	207	0	0	138	1	1.9	2	1	3
106	51	1	0	140	299	0	1	173	1	1.6	2	0	3
270	43	1	0	110	211	0	1	161	0	0.0	2	0	3
860	52	1	0	112	230	0	1	160	0	0.0	2	1	2

820 rows × 13 columns

In [21]:
```
x_test
```

Out[21]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal
527	62	0	0	124	209	0	1	163	0	0.0	2	0	2
359	53	0	2	128	216	0	0	115	0	0.0	2	0	0
447	55	1	0	160	289	0	0	145	1	0.8	1	1	3
31	50	0	1	120	244	0	1	162	0	1.1	2	0	2
621	48	1	0	130	256	1	0	150	1	0.0	2	2	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
832	68	1	2	118	277	0	1	151	0	1.0	2	1	3
796	41	1	1	135	203	0	1	132	0	0.0	1	0	1

644	44	1	2	120	226	0	1	169	0	0.0	2	0	2
404	61	1	0	140	207	0	0	138	1	1.9	2	1	3
842	58	1	2	112	230	0	0	165	0	2.5	1	1	3

205 rows × 13 columns

In [22]:
```
y_train
```

Out[22]:
```
835    0
137    1
534    1
495    1
244    1
      ..
700    1
71     0
106    0
270    1
860    0
Name: target, Length: 820, dtype: int64
```

In [23]:
```
y_test
```

Out[23]:
```
527    1
359    1
447    0
31     1
621    0
      ..
832    1
796    1
644    1
404    0
842    0
Name: target, Length: 205, dtype: int64
```

In [24]:
```
data.head()
```

Out[24]:

	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

In [25]:
```
from sklearn.linear_model import LogisticRegression
```

In [26]:
```
log = LogisticRegression()
log.fit(x_train, y_train)
```

```
C:\Users\hp\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:763: ConvergenceWarning: lbfgs failed t
o converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
```

Out[26]:
```
LogisticRegression()
```

In [27]:
```
y_pred1=log.predict(x_test)
```

In [28]:
```
from sklearn.metrics import accuracy_score
```

In [29]:
```
accuracy_score (y_test,y_pred1)
```

Out[29]:
```
0.7853658536585366
```

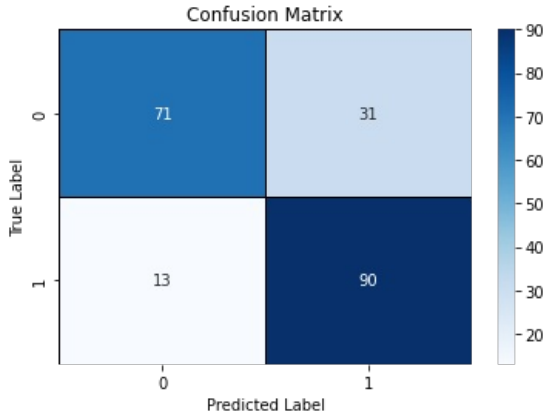
In [30]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix
```

In [31]:
```
cm = confusion_matrix(y_test, y_pred1)

labels = np.unique(y_test)  # Get unique class labels
cm_df = pd.DataFrame(cm, index=labels, columns=labels)

# Plot confusion matrix using seaborn
plt.figure(figsize=(6, 4))
sns.heatmap(cm_df, annot=True, fmt='d', cmap='Blues', linewidths=1, linecolor='black')

plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()
```



In [ ]:
```

```