```
TASK-3 : IRIS FLOWER CLASSIFICATION

Author: Bandana Prakash
Batch: June
Domain: Data Science
Aim: to develop a model that can classify iris flowers into different species based on their sepal and
petal measurements.
```

IMPORTING IMPORTANT LIBRARIES

In [1]:
```
import numpy as np
import pandas as pd
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
```

DOWNLOADING DATASETS

In [2]:
```
df = sns.load_dataset('iris')
df.head()
```

Out[2]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

In [3]:
```
df['species'],categories =pd.factorize(df['species'])
df.head()
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

In [4]:
```
df.describe
```

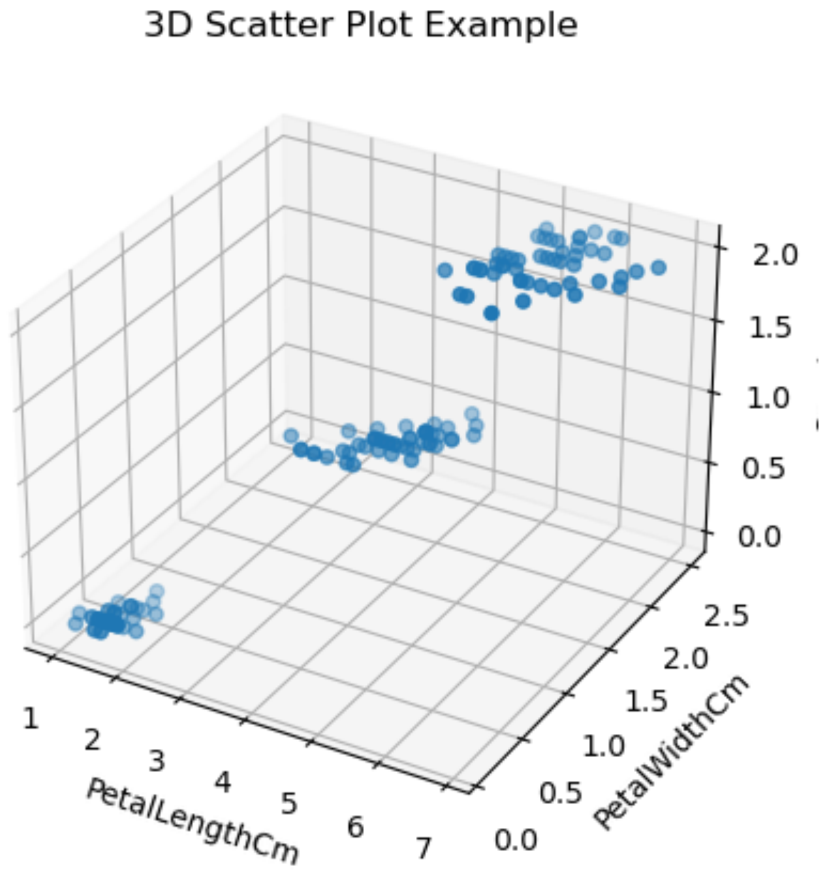
Out[4]:
```
<bound method NDFrame.describe of      sepal_length  sepal_width  petal_length  petal_width  species
0             5.1          3.5           1.4          0.2        0
1             4.9          3.0           1.4          0.2        0
2             4.7          3.2           1.3          0.2        0
3             4.6          3.1           1.5          0.2        0
4             5.0          3.6           1.4          0.2        0
..            ...          ...           ...          ...      ...
145           6.7          3.0           5.2          2.3        2
146           6.3          2.5           5.0          1.9        2
147           6.5          3.0           5.2          2.0        2
148           6.2          3.4           5.4          2.3        2
149           5.9          3.0           5.1          1.8        2

[150 rows x 5 columns]>
```

In [5]:
```
df.isna().sum()
```

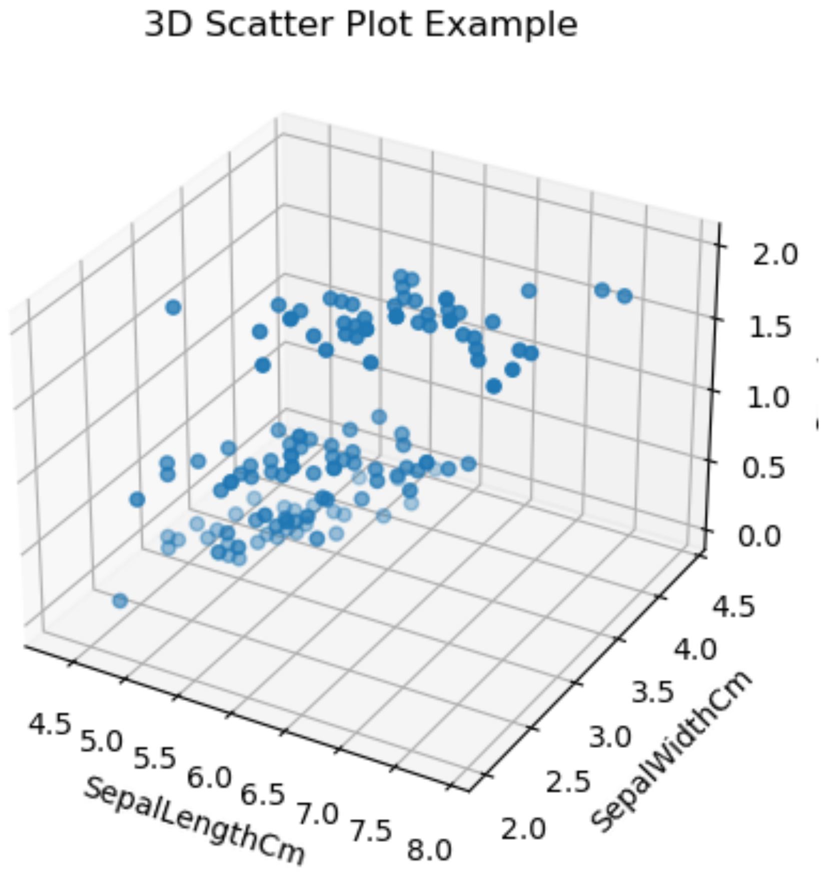
Out[5]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

Hence its time to visualize the data

In [6]:
```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df.petal_length, df.petal_width, df.species)
ax.set_xlabel('PetalLengthCm')
ax.set_ylabel('PetalWidthCm')
ax.set_zlabel('Species')
plt.title('3D Scatter Plot Example')
plt.show()
```

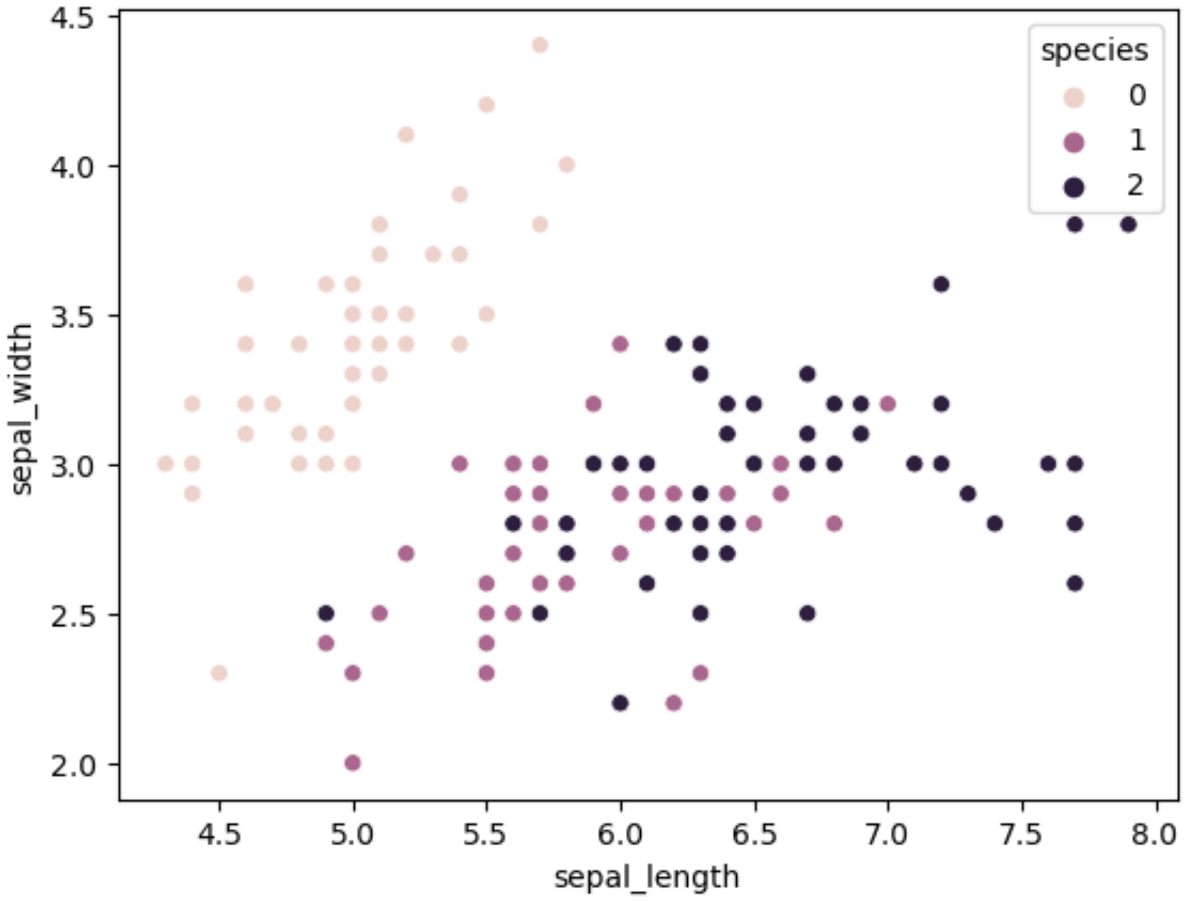


In [7]:
```
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(df.sepal_length, df.sepal_width, df.species)
ax.set_xlabel('SepalLengthCm')
ax.set_ylabel('SepalWidthCm')
ax.set_zlabel('Species')
plt.title('3D Scatter Plot Example')
plt.show()
```

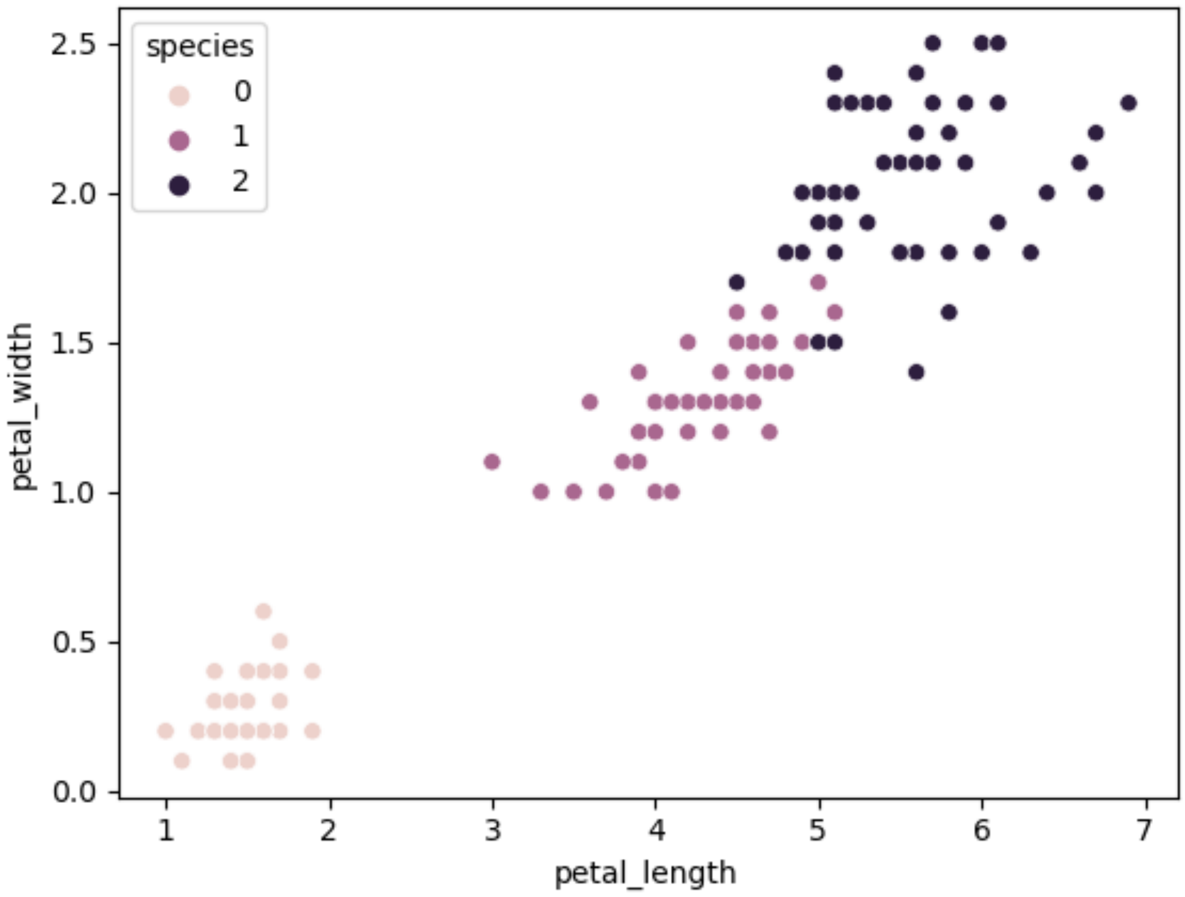


Thus 3-D plot gives us the glimpse of species of iris flower is more inclined towards the variables petal length and petal width.

```
In [8]: sns.scatterplot(data=df, x="sepal_length", y="sepal_width",hue="species");
```



```
In [9]: sns.scatterplot(data=df, x="petal_length", y="petal_width",hue="species");
```



Applying Elbow Technique

In [10]:

```
k_rng = range(1,10)
sse=[]

for k in k_rng:
  km = KMeans(n_clusters=k)
  km.fit(df[[ 'petal_length', 'petal_width']])
  sse.append(km.inertia_)
```

```
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

In [11]:

```
sse
```

Out[11]:

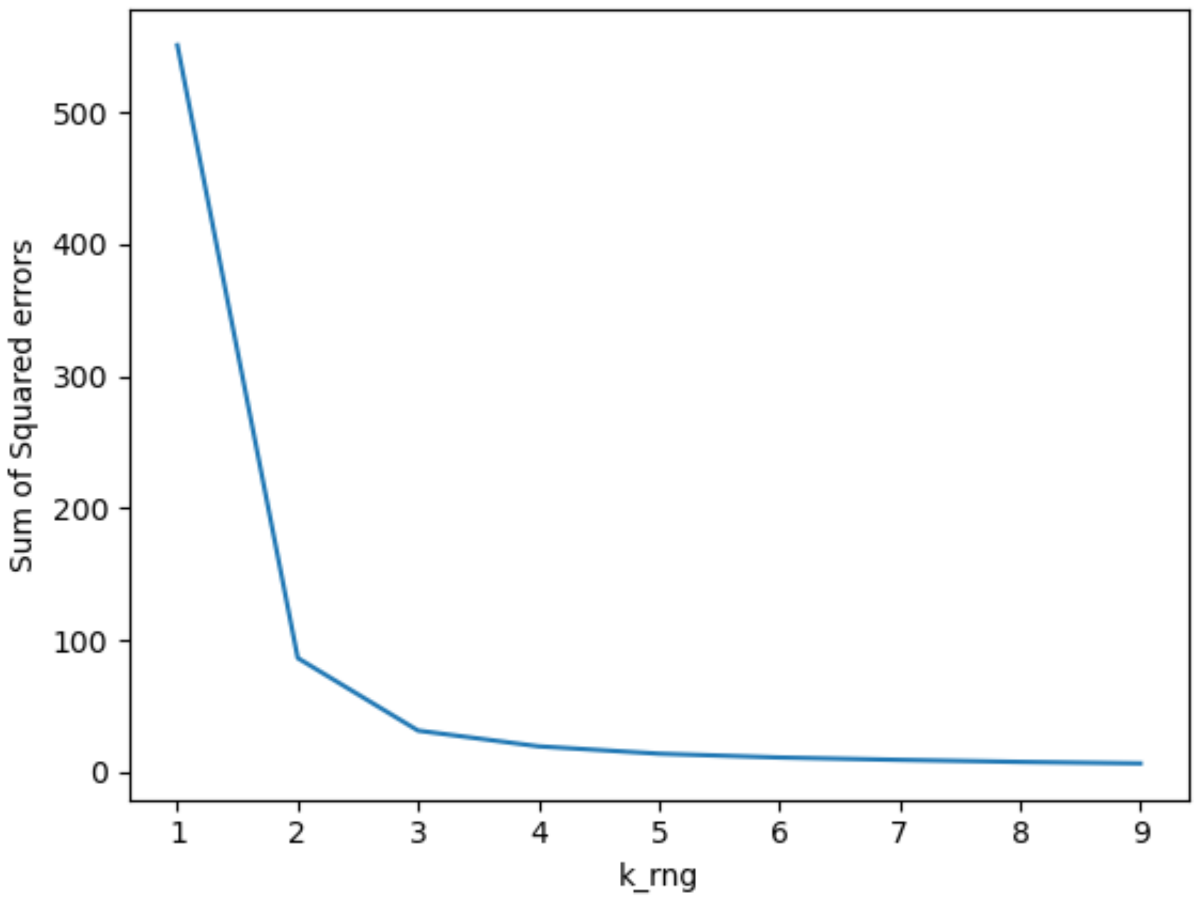
```
[550.8953333333333,
 86.39021984551397,
 31.371358974358973,
 19.483000899685113,
 13.91690875790876,
 11.025145110250373,
 9.206861111111111,
 7.667019523446297,
 6.541584461432288]
```

In [12]:

```
plt.xlabel('k_rng')
plt.ylabel("Sum of Squared errors")
plt.plot(k_rng, sse)
```

Out[12]:

```
[<matplotlib.lines.Line2D at 0x139ff0390>]
```



Applying KMean Algorithm

In [13]:

```
km = KMeans(n_clusters=3,random_state=0,)
y_predicted = km.fit_predict(df[['petal_length','petal_width']])
y_predicted
```

```
/Volumes/Prototype/anaconda3/lib/python3.11/site-packages/sklearn/cluster/_kmeans.py:1412: FutureWarnin
g: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explic
itly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

Out[13]:

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, 2, 2, 2, 2, 2, 0, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 0, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 0, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2], dtype=int32)
```

In [14]:

```
df['cluster']=y_predicted
df.head(150)
```

Out[14]:

	sepal_length	sepal_width	petal_length	petal_width	species	cluster
0	5.1	3.5	1.4	0.2	0	1
1	4.9	3.0	1.4	0.2	0	1
2	4.7	3.2	1.3	0.2	0	1
3	4.6	3.1	1.5	0.2	0	1
4	5.0	3.6	1.4	0.2	0	1
...	...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	2	2
146	6.3	2.5	5.0	1.9	2	2
147	6.5	3.0	5.2	2.0	2	2
148	6.2	3.4	5.4	2.3	2	2
149	5.9	3.0	5.1	1.8	2	2

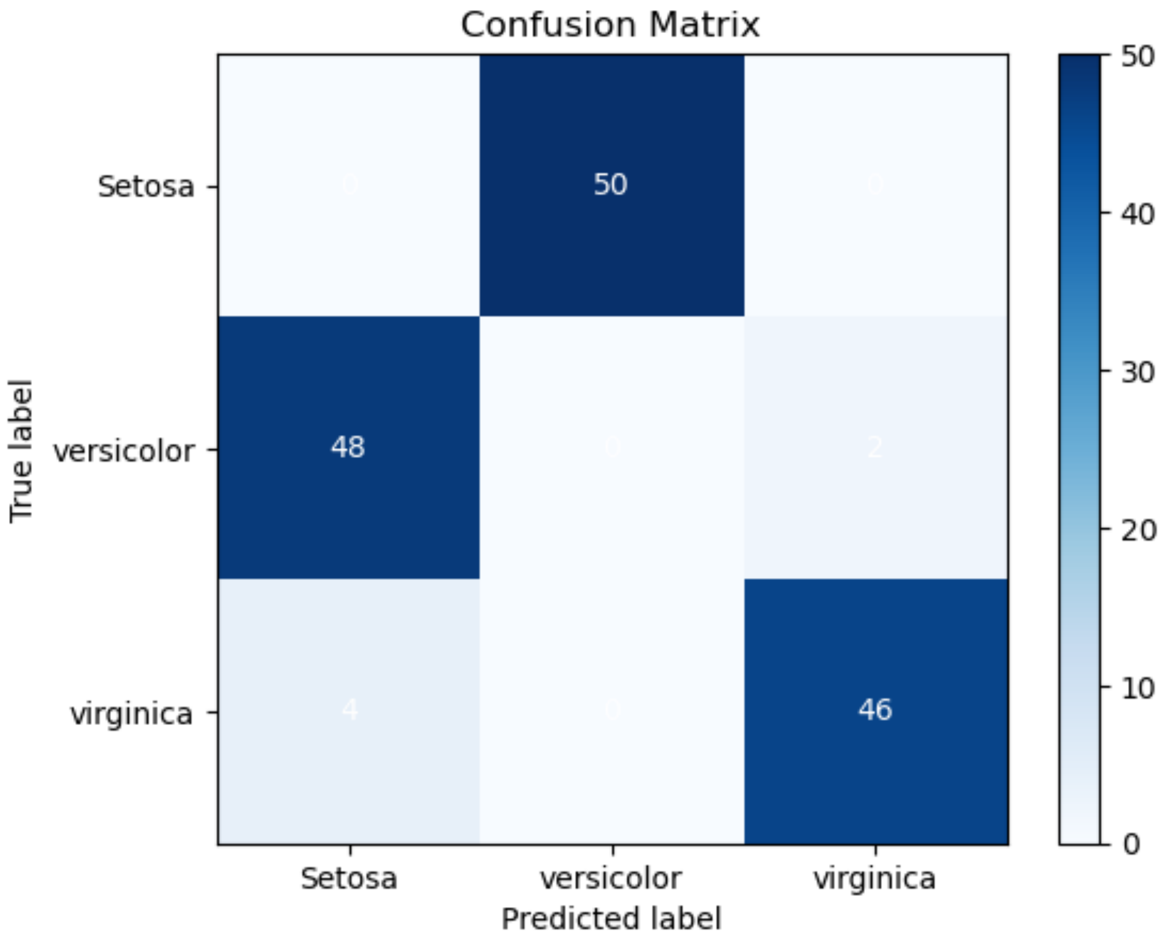
150 rows × 6 columns

Accuracy measure

In [15]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(df.species, df.cluster)
cm
```

Out[15]:
```
array([[ 0, 50,  0],
       [48,  0,  2],
       [ 4,  0, 46]])
```

In [16]:
```
true_labels = df.species
predicted_labels= df.cluster

cm = confusion_matrix(true_labels, predicted_labels)
class_labels = ['Setosa', 'versicolor', 'virginica']

# Plot confusion matrix
plt.imshow(cm, interpolation='nearest', cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
tick_marks = np.arange(len(class_labels))
plt.xticks(tick_marks, class_labels)
plt.yticks(tick_marks, class_labels)

# Fill matrix with values
for i in range(len(class_labels)):
    for j in range(len(class_labels)):
        plt.text(j, i, str(cm[i][j]), ha='center', va='center', color='white')

plt.xlabel('Predicted label')
plt.ylabel('True label')
plt.show()
```


Confusion Matrix
Setosa
versicolor
virginica
True label
Setosa
versicolor
virginica
Predicted label