In [36]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import confusion_matrix, accuracy_score, precision_s
```

In [10]:

```
df = pd.read_csv("Desktop/datasets/Iris.csv")
df.head()
```

Out[10]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

In [12]:

```
X = df.iloc[:, :-1]   # all columns except species
y = df.iloc[:, -1]    # species column
```

In [14]:

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)
```

In [16]:

```
model = GaussianNB()
model.fit(X_train, y_train)
```

Out[16]:

```
▾ GaussianNB ⓘ ?
GaussianNB()
```

In [18]:

```
y_pred = model.predict(X_test)
```

In [20]:

```
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
```

In [22]:

```
TP = np.diag(cm)
FP = cm.sum(axis=0) - TP
FN = cm.sum(axis=1) - TP
TN = cm.sum() - (FP + FN + TP)
```

In [24]:

```
for i, cls in enumerate(model.classes_):
    print(f"\nClass: {cls}")
    print("TP:", TP[i])
    print("FP:", FP[i])
    print("FN:", FN[i])
    print("TN:", TN[i])
```

```
Class: Iris-setosa
TP: 19
FP: 0
FN: 0
TN: 26

Class: Iris-versicolor
TP: 13
FP: 0
FN: 0
TN: 32

Class: Iris-virginica
TP: 13
FP: 0
FN: 0
TN: 32
```

In [26]:

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 1.0
```

In [28]:

```
error_rate = 1 - accuracy
print("Error Rate:", error_rate)
```

```
Error Rate: 0.0
```

In [30]:

```
precision = precision_score(y_test, y_pred, average='macro')
print("Precision:", precision)
```

```
Precision: 1.0
```

In [32]:

```
recall = recall_score(y_test, y_pred, average='macro')
print("Recall:", recall)
```

```
Recall: 1.0
```

In [34]:

```
print("\n--- Model Performance ---")
print("Accuracy :", accuracy)
print("Error Rate :", error_rate)
print("Precision :", precision)
print("Recall :", recall)
```

```
--- Model Performance ---
Accuracy : 1.0
Error Rate : 0.0
Precision : 1.0
Recall : 1.0
```

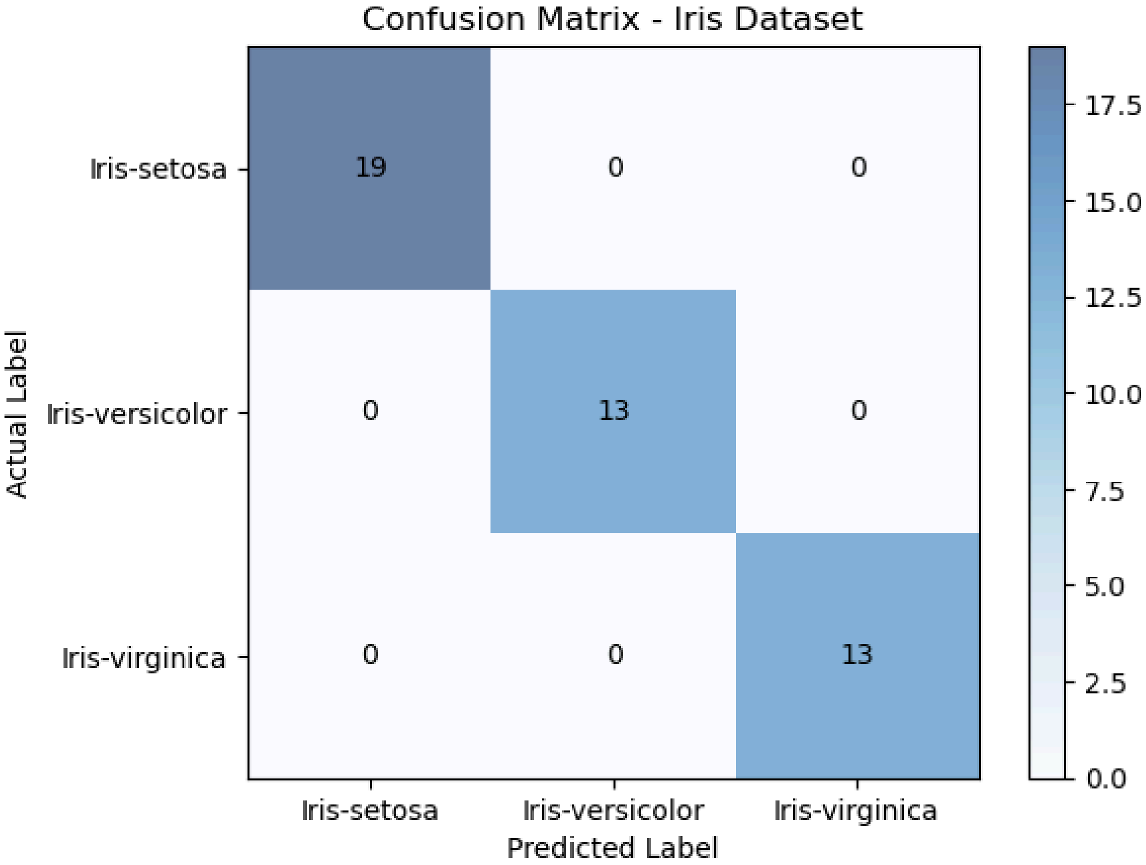
In [42]:

```
plt.figure()
plt.imshow(cm, cmap="Blues", alpha=0.6)  # lighter color
plt.title("Confusion Matrix - Iris Dataset")
plt.xlabel("Predicted Label")
plt.ylabel("Actual Label")
plt.colorbar()

classes = model.classes_
plt.xticks(range(len(classes)), classes)
plt.yticks(range(len(classes)), classes)

for i in range(len(classes)):
    for j in range(len(classes)):
        plt.text(j, i, cm[i, j], ha="center", va="center")

plt.show()
```



```
In [ ]:
```