Importing Neccesary Libraries

In [1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
```

Loading the DataSet

In [12]:
```
file_path = "C:/Users/Aniket/Downloads/Cipherbyte/Iris Flowers.xlsx"
iris_df = pd.read_excel(file_path)
iris_df.head()
```

Out[12]:

	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

Encode the 'Species' column into numerical values

In [25]:
```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
iris_df['Species'] = label_encoder.fit_transform(iris_df['Species'])
```

Split data into features (X) and target (y)

In [24]:
```
X = iris_df.drop(columns=['Species'])
y = iris_df['Species']
```

Split data into training and testing sets (80% train, 20% test)

In [26]:
```
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)
```

Train a Random Forest Classifier

In [27]:
```
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)
```

Out[27]:
```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Make predictions

In [28]:
```
y_pred = clf.predict(X_test)
```

Evaluate the model

In [33]:
```
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

# Print evaluation results
print("Model Accuracy:", accuracy)
print("\nClassification Report:\n", report)
print("\nConfusion Matrix:\n", conf_matrix)
```

```
Model Accuracy: 1.0

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30


Confusion Matrix:
 [[10  0  0]
 [ 0  9  0]
 [ 0  0 11]]
```