Step 1: Import necessary libraries

In [9]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.cluster import KMeans, SpectralClustering, DBSCAN
from sklearn.metrics import silhouette_score
```

Step 2: Load the dataset

In [16]:

```
# Load the Iris dataset
X = pd.read_csv("C:/Users/SMIT YENKAR/OneDrive/Desktop/T.Y .Sub/ML LAB/Iris.
print(X.head())
# Perform encoding for the 'Species' column
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
X['Species'] = label_encoder.fit_transform(X['Species'])
```

```
   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
```

Step 3: Apply K-means clustering

In [17]:

```
# Apply K-means clustering
kmeans = KMeans(n_clusters=3)
kmeans_labels = kmeans.fit_predict(X.drop(columns=['Species']))
```

Step 4: Apply Spectral clustering

In [18]:

```
# Apply Spectral clustering
spectral = SpectralClustering(n_clusters=3, affinity='nearest_neighbors')
spectral_labels = spectral.fit_predict(X.drop(columns=['Species']))
```

Step 5: Apply DBSCAN clustering

In [19]:

```
# Apply DBSCAN clustering
dbscan = DBSCAN(eps=0.5, min_samples=5)
dbscan_labels = dbscan.fit_predict(X.drop(columns=['Species']))
```

Step 6: Evaluate clustering performance using silhouette score

In [27]:

```
from sklearn.metrics import silhouette_score
from sklearn.preprocessing import StandardScaler

# Normalize the data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X.drop(columns=['Species']))

# Adjust hyperparameters and apply DBSCAN clustering
dbscan = DBSCAN(eps=0.4, min_samples=5)
dbscan_labels = dbscan.fit_predict(X_scaled)

# Evaluate clustering performance using silhouette score
unique_labels = np.unique(dbscan_labels)
num_clusters = len(unique_labels) - (1 if -1 in dbscan_labels else 0)  # Exc
if num_clusters > 1:
    silhouette_dbscan = silhouette_score(X_scaled, dbscan_labels)
    print("Silhouette Score (DBSCAN):", silhouette_dbscan)
else:
    print("DBSCAN failed to identify distinct clusters.")

print("\nSilhouette Score:")
print("K-means:", silhouette_kmeans)
print("Spectral Clustering:", silhouette_spectral)
```

```
Silhouette Score (DBSCAN): 0.10816320348375333

Silhouette Score:
K-means: 0.5821934246576435
Spectral Clustering: 0.5514935658161797
```

In [ ]: