In [11]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_iris
from sklearn.cluster import KMeans, AgglomerativeClustering
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.preprocessing import StandardScaler
```

Load Iris Dataset

In [2]:

```
data = load_iris()
df = pd.DataFrame(data.data, columns=data.feature_names)
```

Display dataset information ¶

In [3]:

```
print(df.info())
print(df.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 4 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   sepal length (cm)  150 non-null    float64
 1   sepal width (cm)   150 non-null    float64
 2   petal length (cm)  150 non-null    float64
 3   petal width (cm)   150 non-null    float64
dtypes: float64(4)
memory usage: 4.8 KB
None
       sepal length (cm)  sepal width (cm)  petal length (cm)  \
count         150.000000        150.000000         150.000000   
mean            5.843333          3.057333           3.758000   
std             0.828066          0.435866           1.765298   
min             4.300000          2.000000           1.000000   
25%             5.100000          2.800000           1.600000   
50%             5.800000          3.000000           4.350000   
75%             6.400000          3.300000           5.100000   
max             7.900000          4.400000           6.900000   

       petal width (cm)  
count        150.000000  
mean           1.199333  
std            0.762238  
min            0.100000  
25%            0.300000  
50%            1.300000  
75%            1.800000  
max            2.500000  
```

Feature Scaling

In [4]:

```
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df)
```

KMeans Clustering

In [14]:

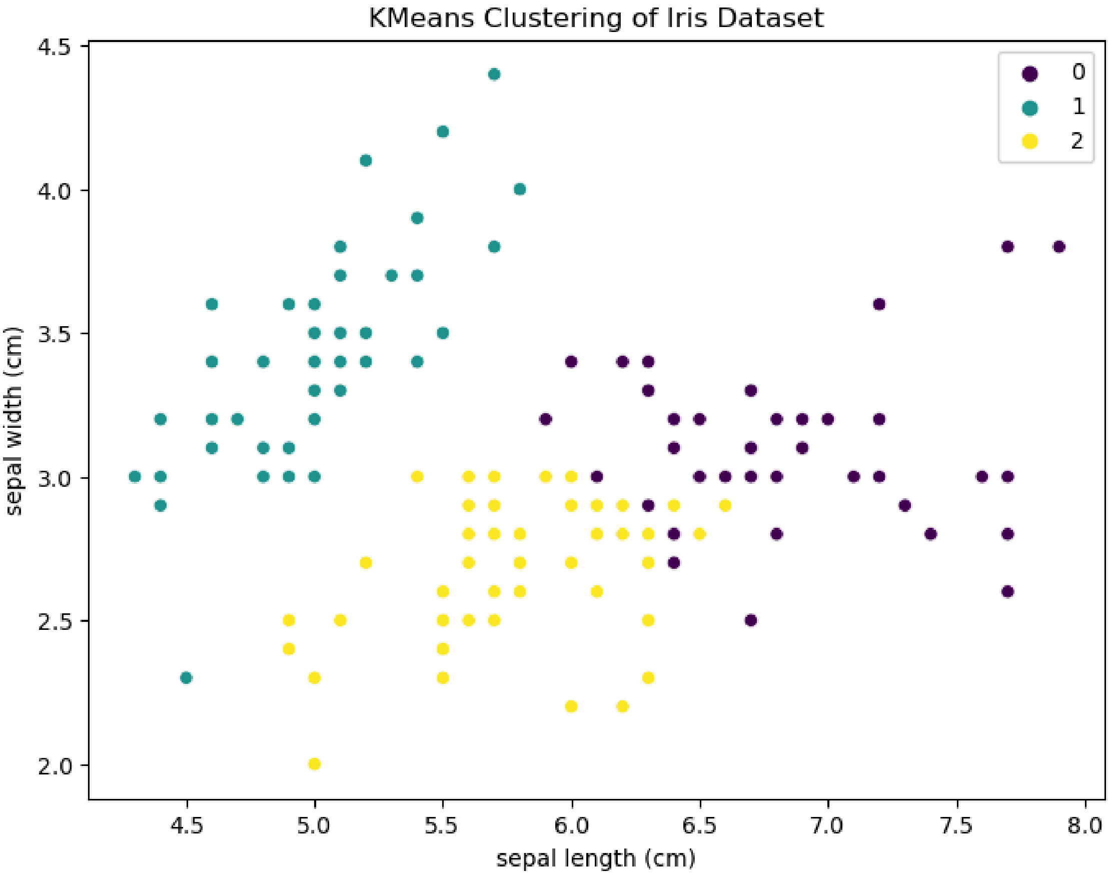
```
import os
os.environ["OMP_NUM_THREADS"] = "1"
kmeans = KMeans(n_clusters=3, n_init=10, random_state=42)
kmeans_labels = kmeans.fit_predict(df_scaled)
df['KMeans_Cluster'] = kmeans_labels
```

```
C:\Users\athir\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:138
2: UserWarning: KMeans is known to have a memory leak on Windows with MK
L, when there are less chunks than available threads. You can avoid it by
setting the environment variable OMP_NUM_THREADS=1.
  warnings.warn(
```

Visualizing KMeans Clusters

In [7]:

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x=df.iloc[:, 0], y=df.iloc[:, 1], hue=kmeans_labels, palet
plt.title("KMeans Clustering of Iris Dataset")
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
plt.show()
```



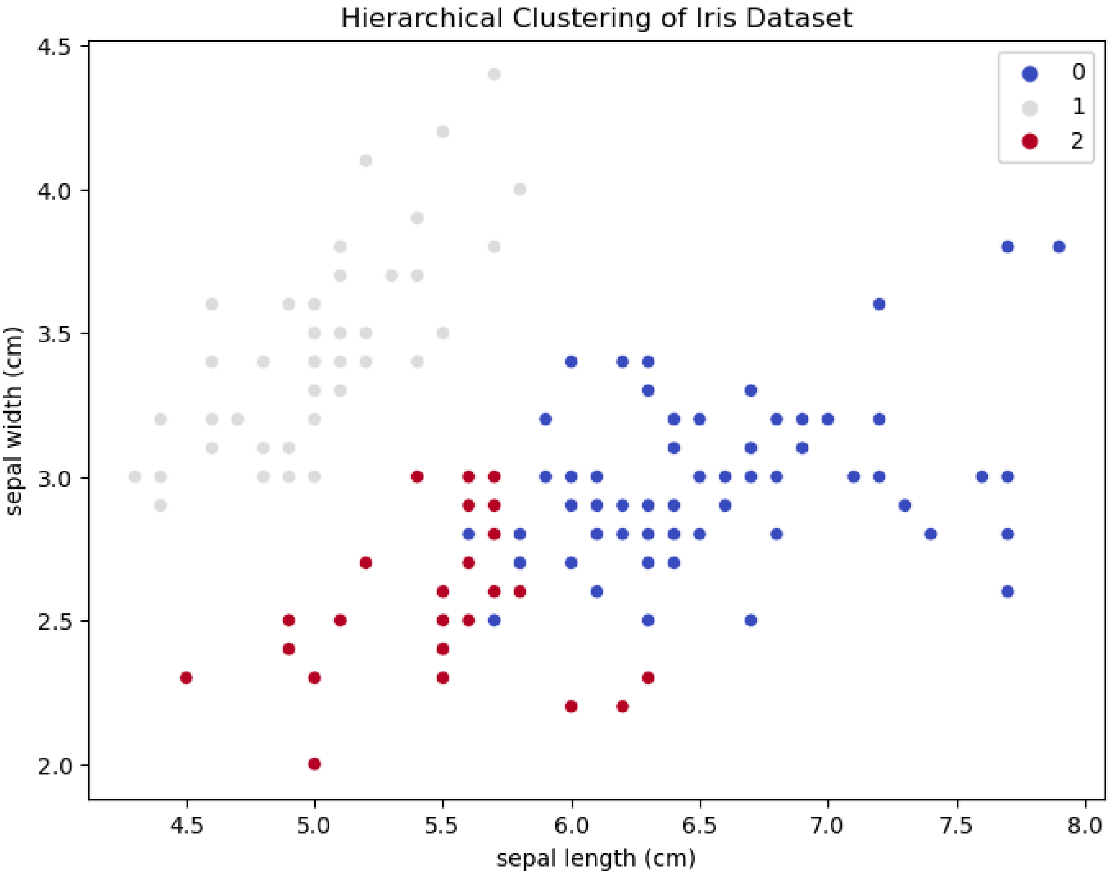
Hierarchical Clustering

In [8]:

```
hierarchical = AgglomerativeClustering(n_clusters=3)
hierarchical_labels = hierarchical.fit_predict(df_scaled)
df['Hierarchical_Cluster'] = hierarchical_labels
```

Visualizing Hierarchical Clusters

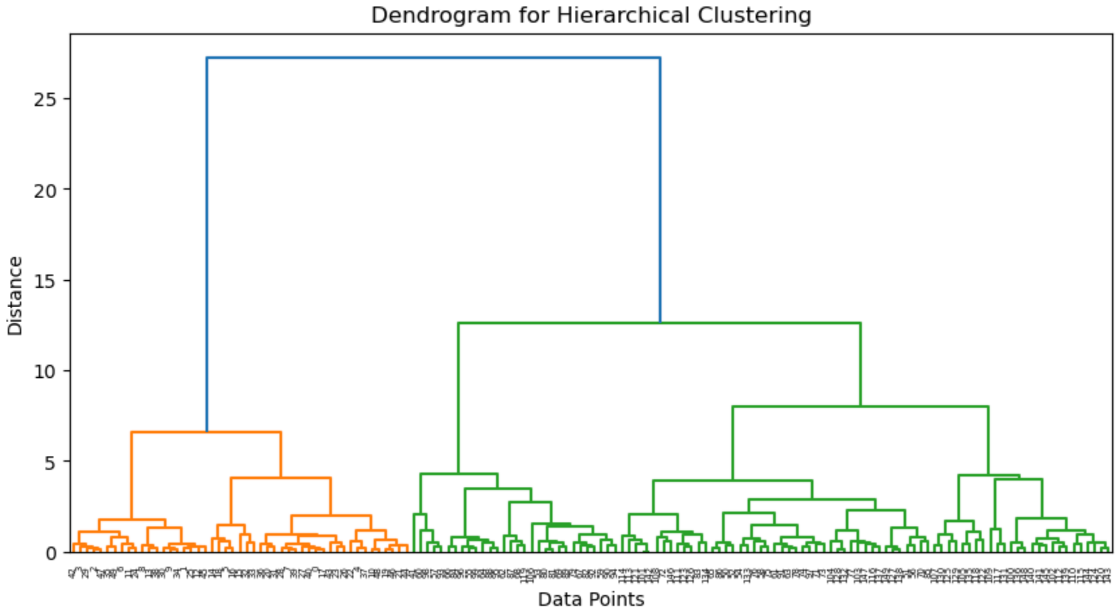
In [9]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(x=df.iloc[:, 0], y=df.iloc[:, 1], hue=hierarchical_labels,
plt.title("Hierarchical Clustering of Iris Dataset")
plt.xlabel(data.feature_names[0])
plt.ylabel(data.feature_names[1])
plt.show()
```



Dendrogram

In [10]:

```
plt.figure(figsize=(10, 5))
linkage_matrix = linkage(df_scaled, method='ward')
dendrogram(linkage_matrix)
plt.title("Dendrogram for Hierarchical Clustering")
plt.xlabel("Data Points")
plt.ylabel("Distance")
plt.show()
```



In [ ]: