Importing necessary libraries

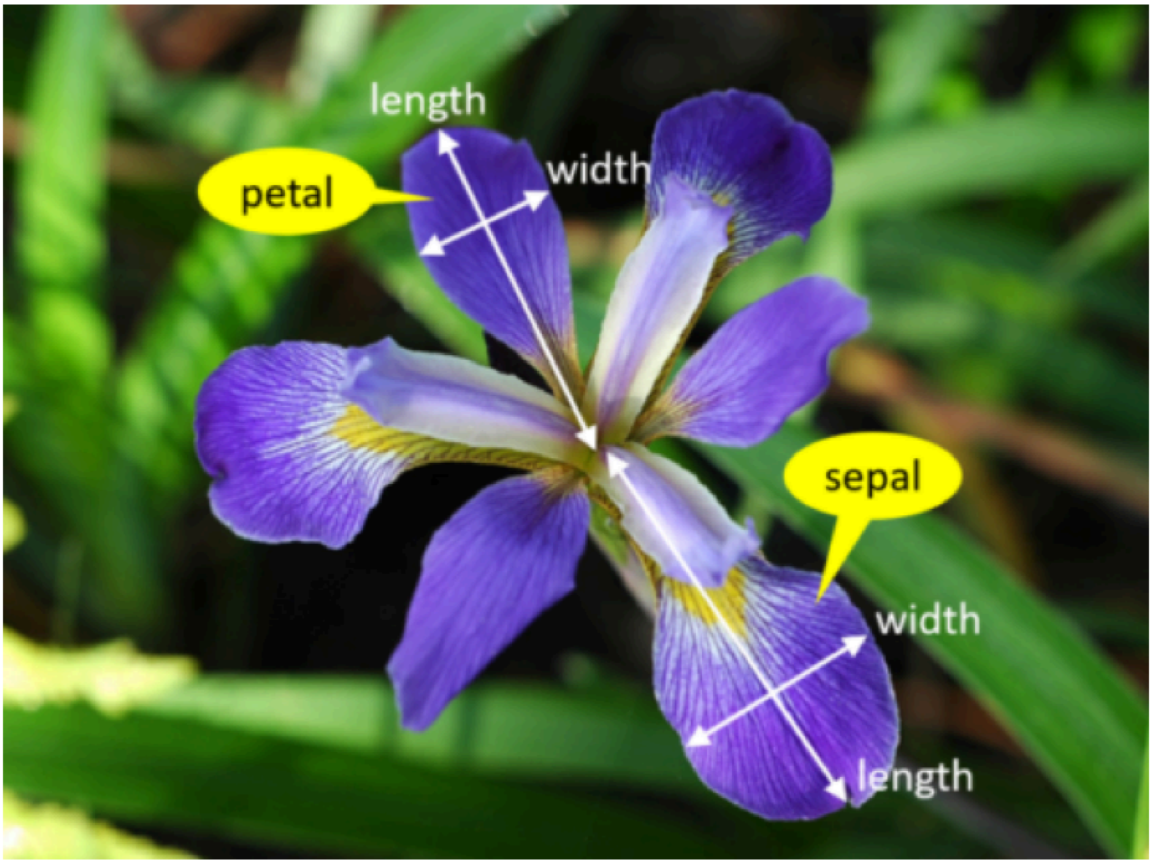
In [1]:
```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from PIL import Image
```

Suppressing warnings

In [2]:
```
warnings.filterwarnings('ignore')
```

Load & Display An Image of IRIS Flower

In [3]:
```
img = Image.open(r'C:\Users\Shriniwas\Desktop\Data Analyst Course\Python Basic\1
fig = plt.gcf()
fig.set_size_inches(10,7)
plt.imshow(img)
plt.axis('off')  # Hide axes
plt.show()
```



Get Current Directory / File Path

In [4]:
```
os.getcwd()
```

Out[4]:
```
'C:\\Users\\Shriniwas'
```

Load the Iris dataset

In [5]:
```
iris = pd.read_csv(r'C:\Users\Shriniwas\Desktop\Data Analyst Course\Python Basic
```

Display the dataset

In [6]:
```
iris
```

Out[6]:

	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
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 6 columns

In [7]:
```
iris.head()
```

Out[7]:

	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 [8]:

```
iris.tail()
```

Out[8]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

Print column names

In [9]:

```
for i in iris.columns:
    print(i)
```

```
Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

Drop the Specific Column i.e - 'Id'

In [10]:

```
iris.drop('Id', axis=1, inplace=True)
```

In [11]:

```
for i in iris.columns:
    print(i)
```

```
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

Check for null and missing values

In [12]:
```
iris.isna()
```

Out[12]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
145	False	False	False	False	False
146	False	False	False	False	False
147	False	False	False	False	False
148	False	False	False	False	False
149	False	False	False	False	False

150 rows × 5 columns

In [13]:
```
iris.isnull()
```

Out[13]:

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
145	False	False	False	False	False
146	False	False	False	False	False
147	False	False	False	False	False
148	False	False	False	False	False
149	False	False	False	False	False

150 rows × 5 columns

In [14]:
```
iris.isnull().sum()
```

Out[14]:
```
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

Count the number of species

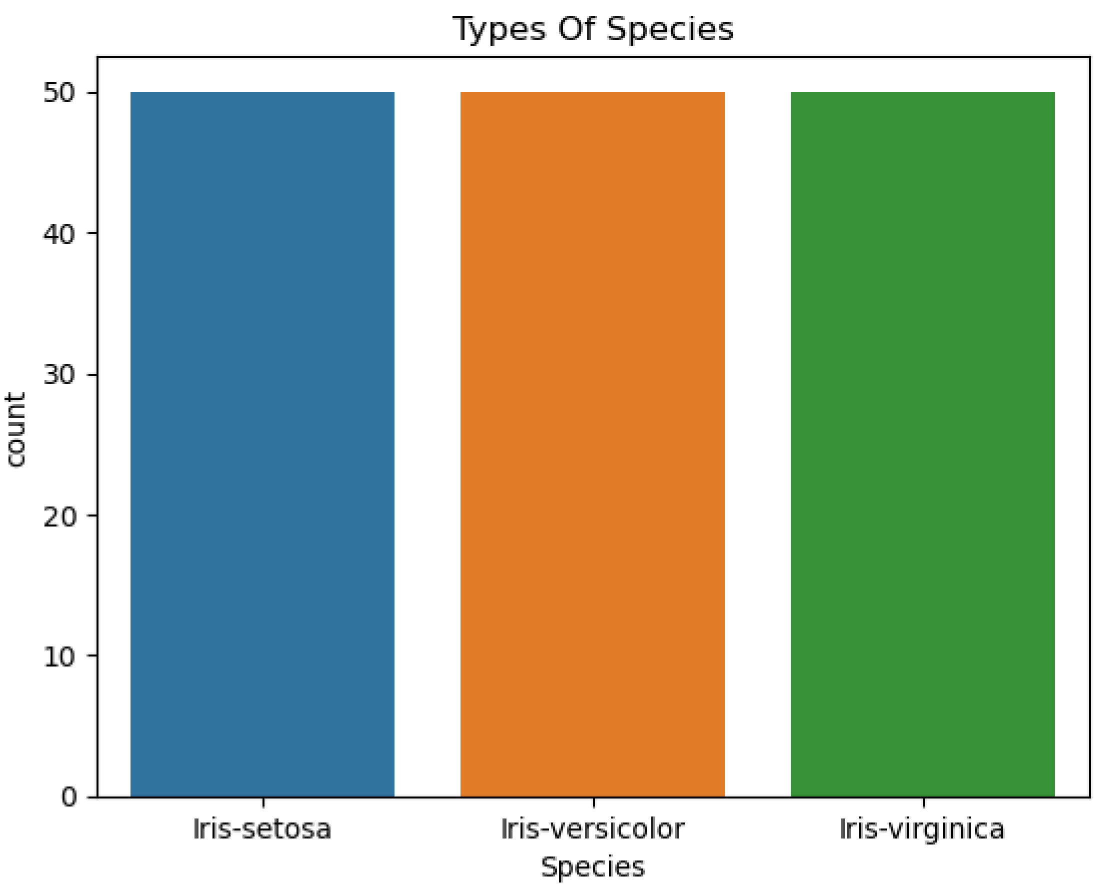
In [15]:
```
iris['Species'].value_counts()
```

Out[15]:
```
Species
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: count, dtype: int64
```

Visualization

Visualization 1: Countplot of species

In [16]:
```
vis1 = sns.countplot(data=iris, x='Species', hue= 'Species')
plt.title('Types Of Species')
plt.show()
```



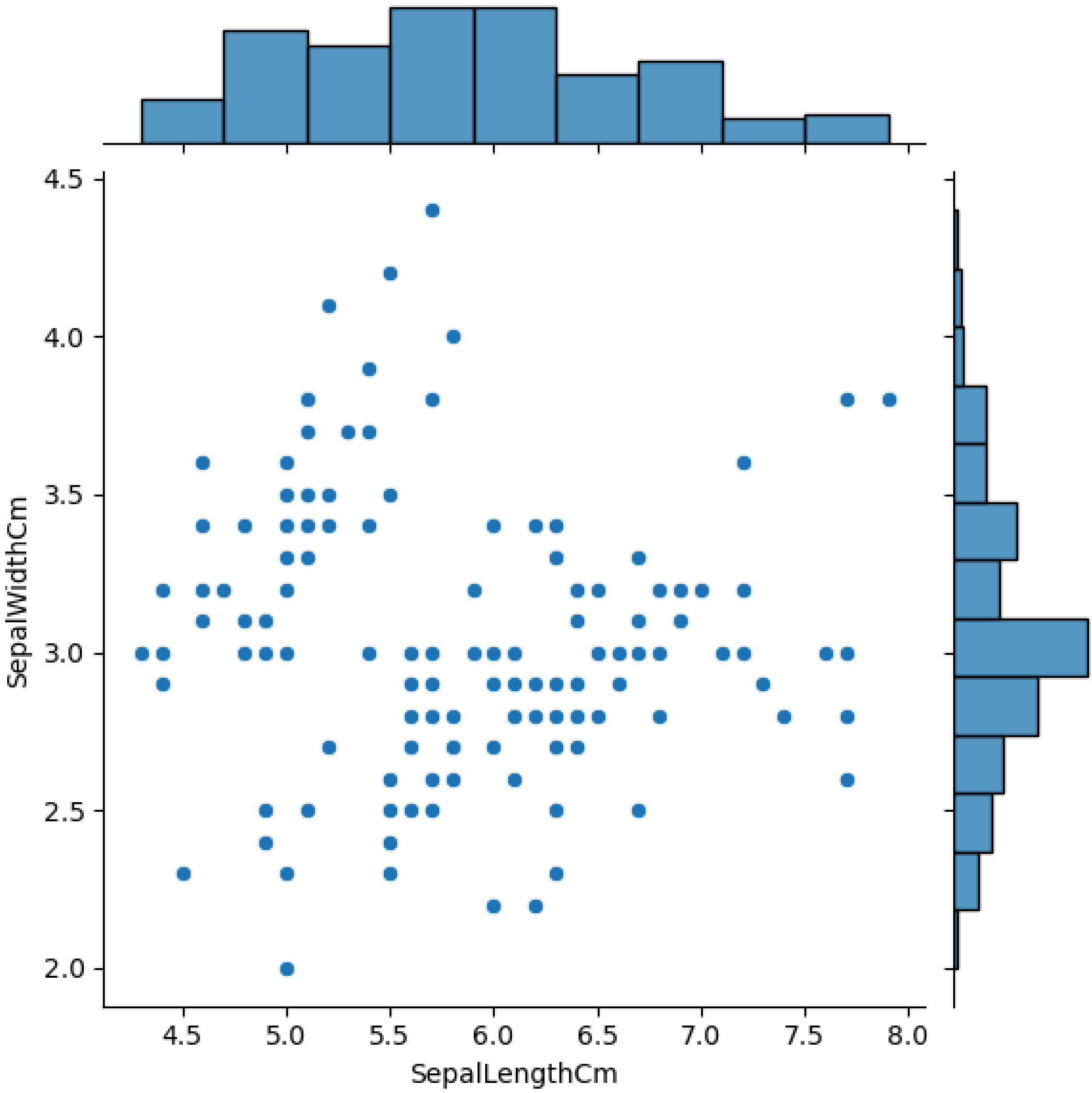
In [17]:
```
iris.head()
```

Out[17]:

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

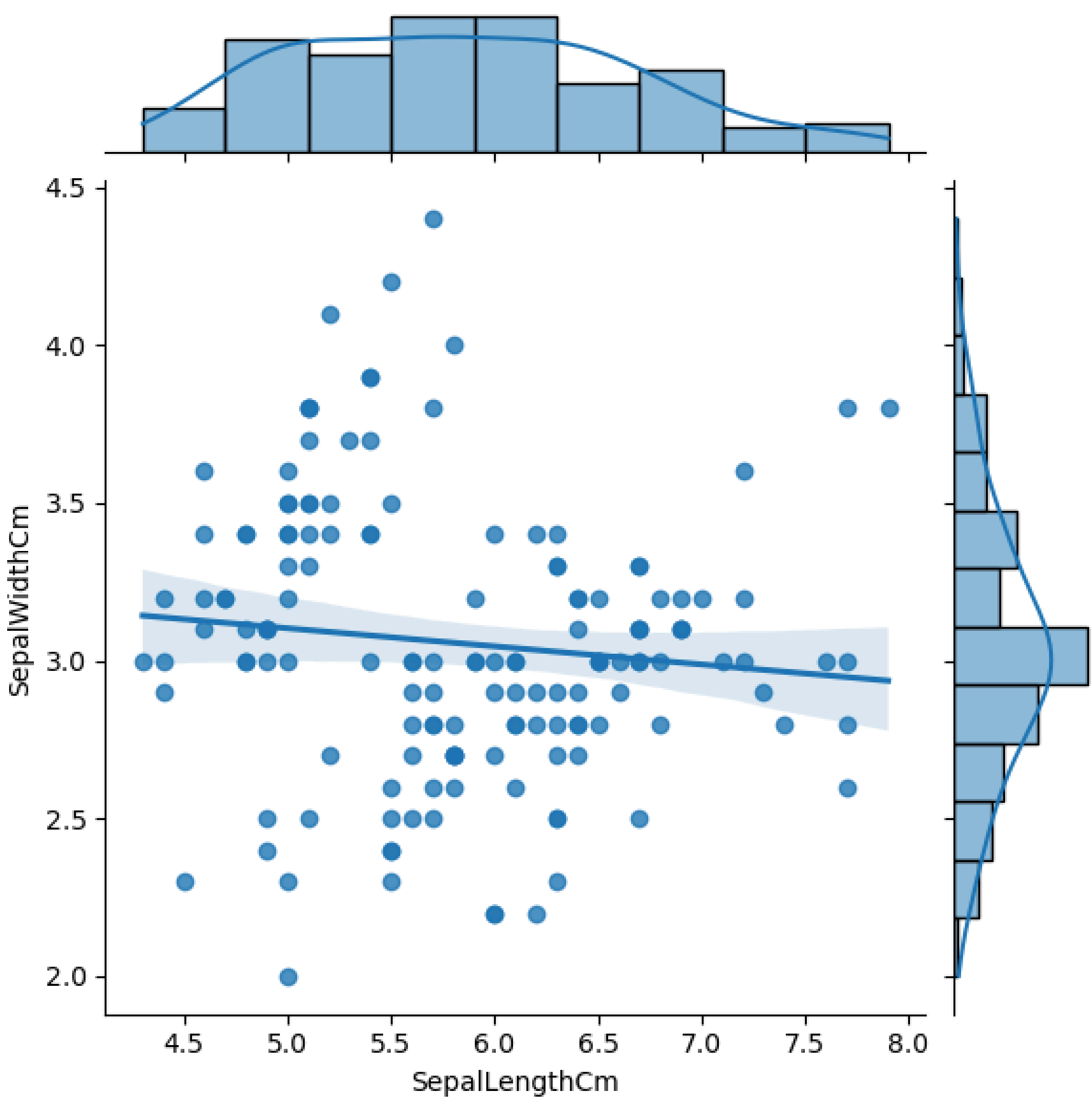
Visualization 2: Jointplot (SepalLengthCm vs SepalWidthCm)

In [18]:
```
vish2 = sns.jointplot(data=iris, x='SepalLengthCm', y='SepalWidthCm')
```



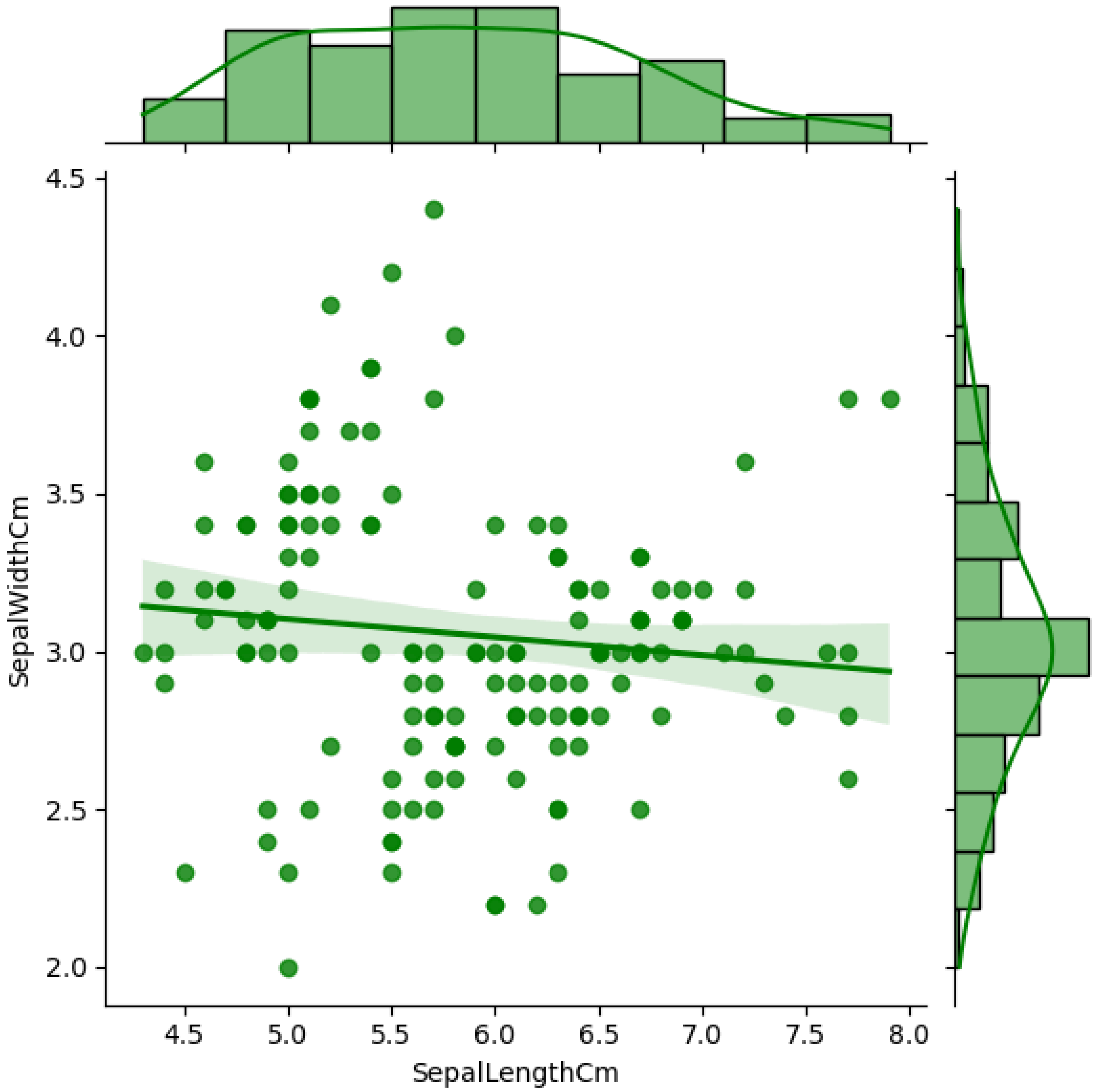
Visualization 3: Jointplot with regression line

In [19]:
```
vish3 = sns.jointplot(data=iris, x='SepalLengthCm', y='SepalWidthCm', kind='reg'
```



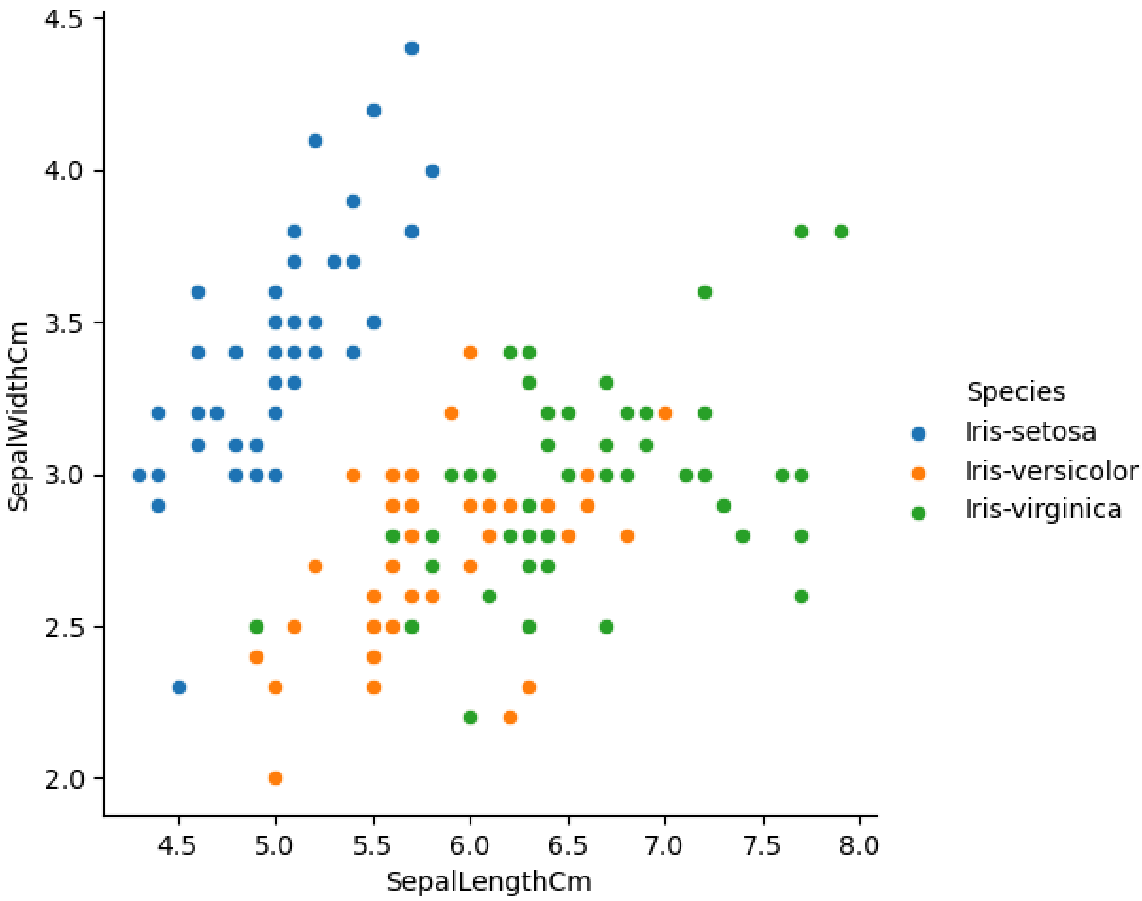
Visualization 4: Scatter plot with species-wise colors

In [20]:
```
vish4 = sns.jointplot(data=iris, x='SepalLengthCm', y='SepalWidthCm', kind='reg'
```



Visualization : FacetGrid (scatter plot of SepalLengthCm vs SepalWidthCm)

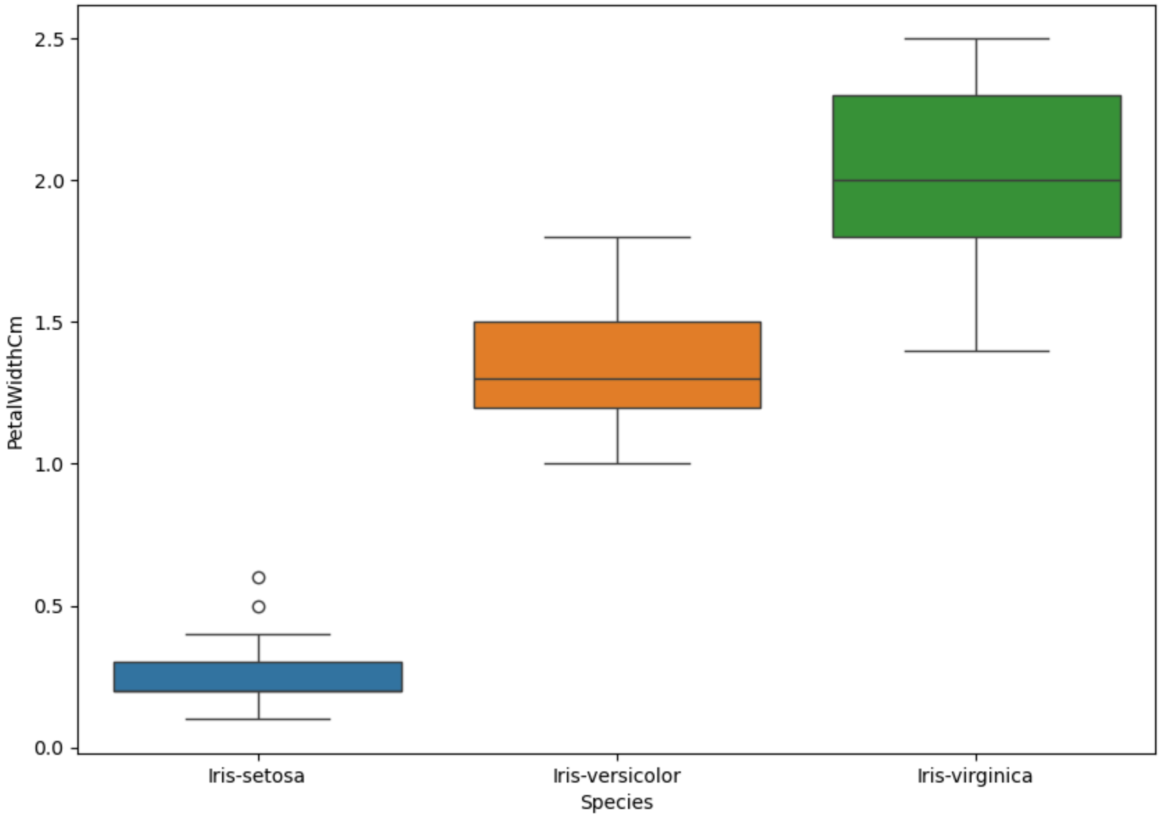
In [21]:
```
f = sns.FacetGrid(data=iris, hue='Species', height=5)
f.map(sns.scatterplot, 'SepalLengthCm', 'SepalWidthCm')
f.add_legend()
plt.show()
```



Visualization 5: Boxplot of PetalWidthCm by species

In [22]:

```
fig=plt.gcf()
fig.set_size_inches(10,7)

vish5= sns.boxplot(data=iris,  x='Species', y='PetalWidthCm', hue='Species', ori
```

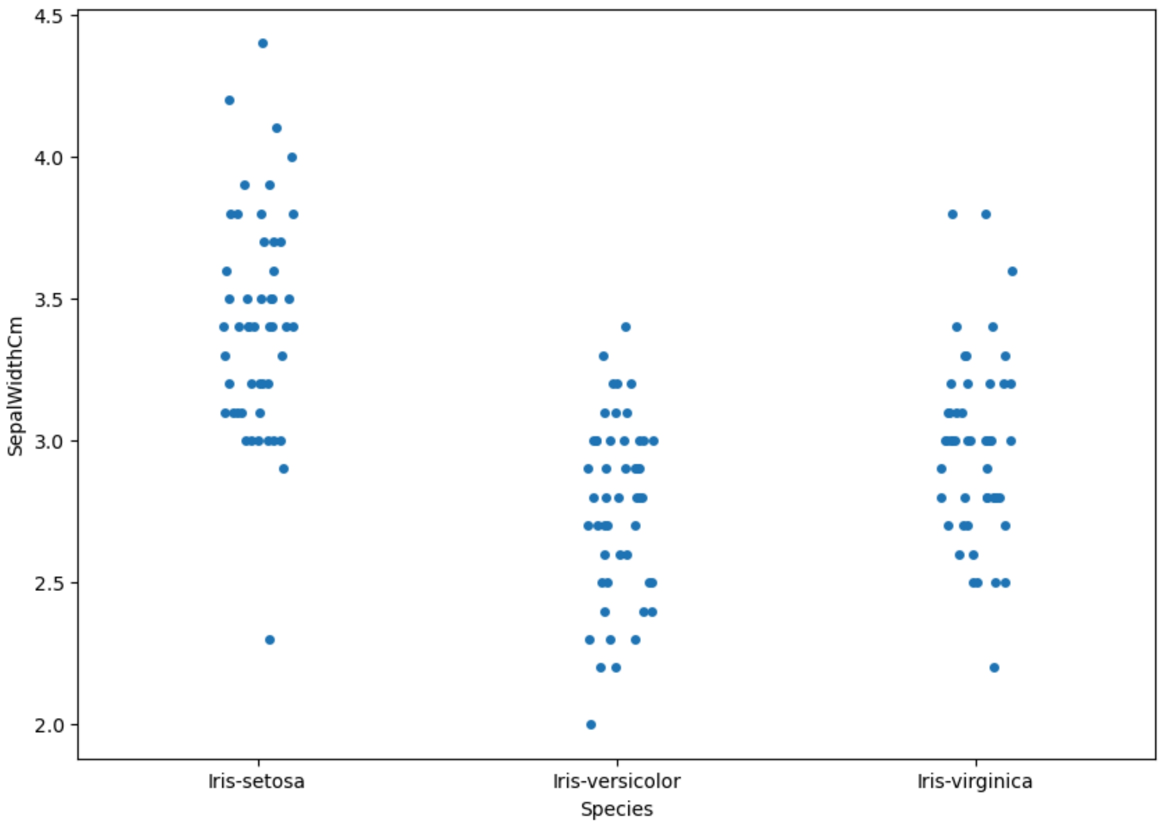


In [23]:
```
for i in iris.columns:
    print(i)
```

```
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

Visualization 6: Stripplot for different features

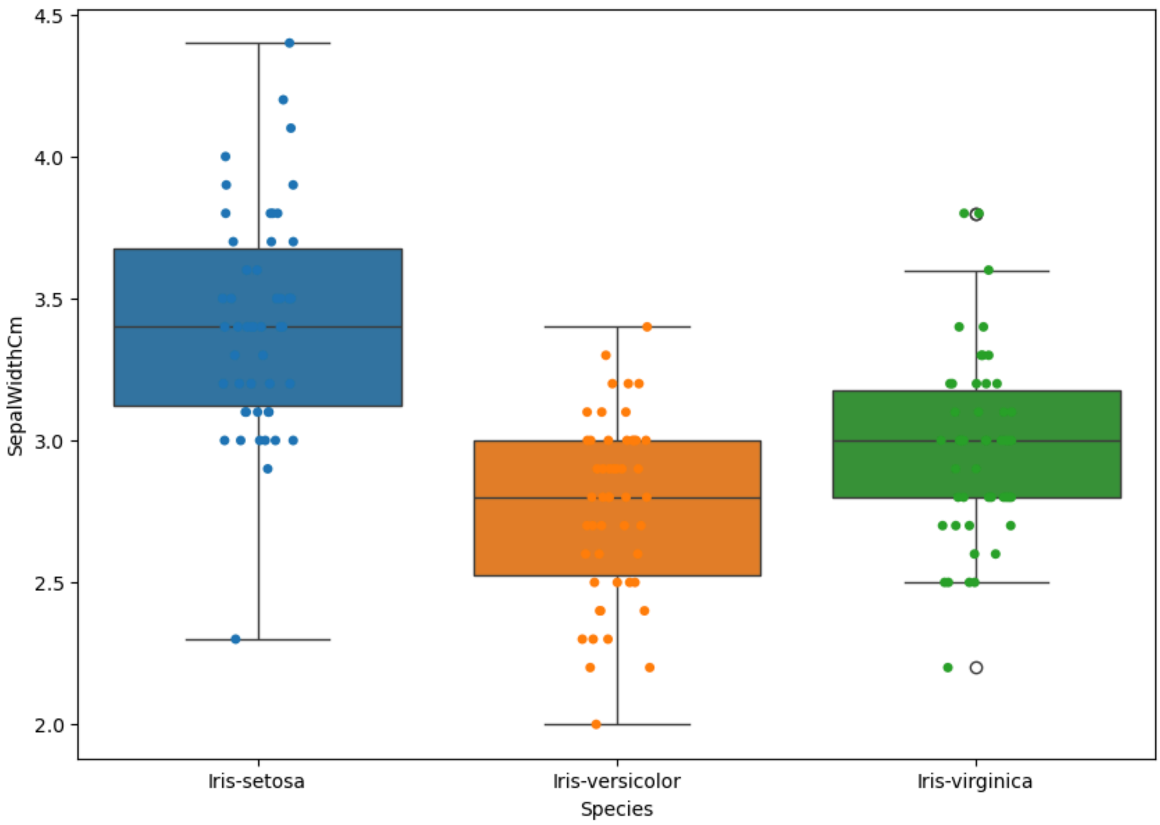
In [24]:
```
vish6=sns.stripplot(data=iris, x='Species', y='SepalWidthCm')
fig=plt.gcf()
fig.set_size_inches(10,7)
plt.show()
```



Visualization : Combination with Boxplot & Stripplot for different features

In [25]:

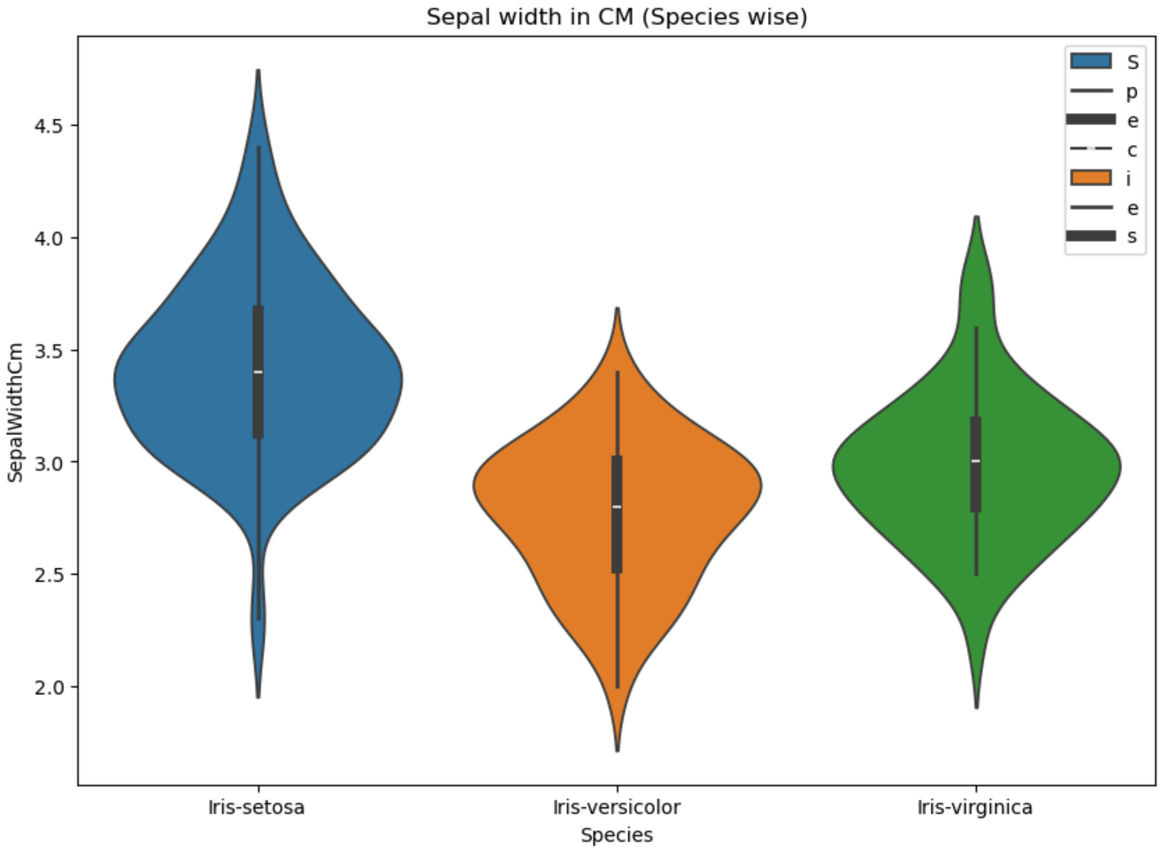
```
fig=plt.gcf()
fig.set_size_inches(10,7)

# Create the box plot
fig2 = sns.boxplot(data=iris, x='Species', y='SepalWidthCm', hue='Species')

# Overlay the strip plot for SepalWidthCm
fig2 = sns.stripplot(x='Species', y='SepalWidthCm',hue='Species', data=iris, jit

# Display the plot
plt.show()
```



Visualization 7: Violin plots for different features

In [26]:

```
vish7= sns.violinplot(data=iris, x='Species', y ='SepalWidthCm',hue='Species')

vish7=plt.gcf()
vish7.set_size_inches(10,7)


plt.title('Sepal width in CM (Species wise)')
plt.legend('Species')
plt.show()
```



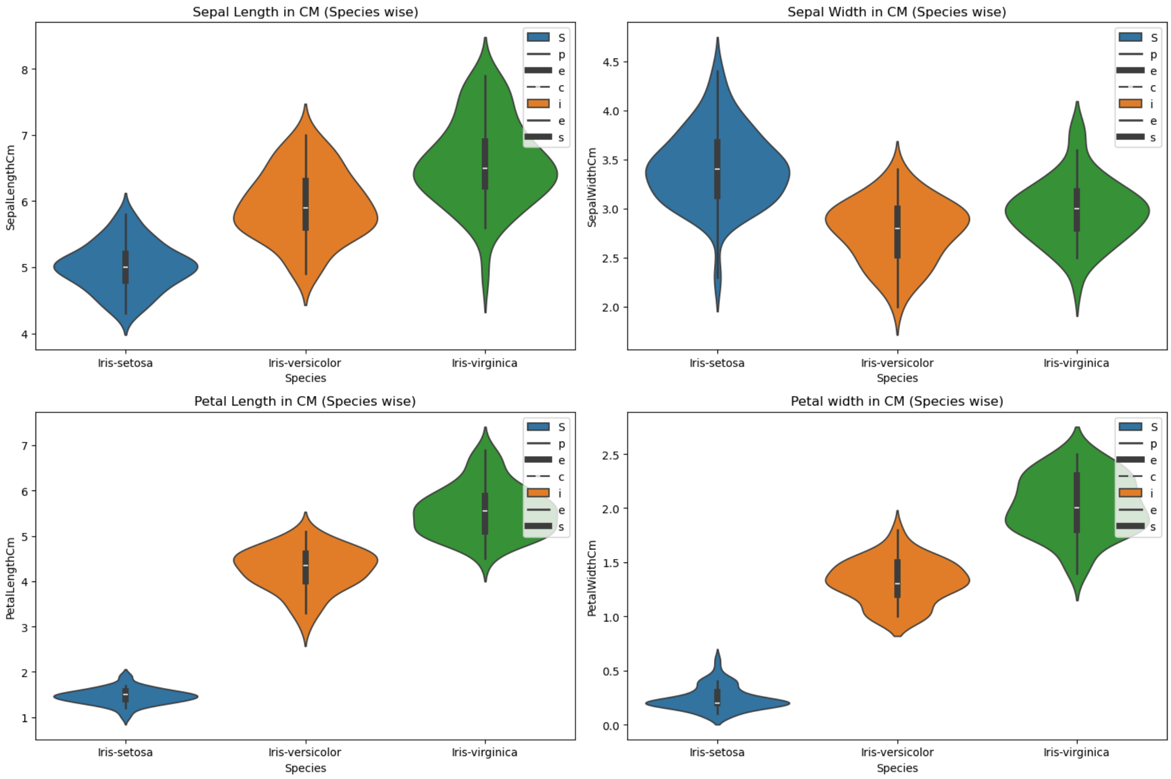
In [27]:
```
for i in iris.columns:
    print(i)
```

```
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

Visualization 8 : Combination with Subplot & Violinplot for different features

In [28]:
```
vish8= plt.figure(figsize=(15,10))

vish8= plt.subplot(2,2,1)
vish8= sns.violinplot(data=iris, x='Species', y ='SepalLengthCm',hue='Species')
plt.title('Sepal Length in CM (Species wise)')
plt.legend('Species')

vish8= plt.subplot(2,2,2)
vish8= sns.violinplot(data=iris, x='Species', y ='SepalWidthCm',hue='Species')
plt.title('Sepal Width in CM (Species wise)')
plt.legend('Species')

vish8= plt.subplot(2,2,3)
vish8= sns.violinplot(data=iris, x='Species', y ='PetalLengthCm',hue='Species')
plt.title('Petal Length in CM (Species wise)')
plt.legend('Species')

vish8= plt.subplot(2,2,4)
vish8= sns.violinplot(data=iris, x='Species', y ='PetalWidthCm',hue='Species')
plt.title('Petal width in CM (Species wise)')
```

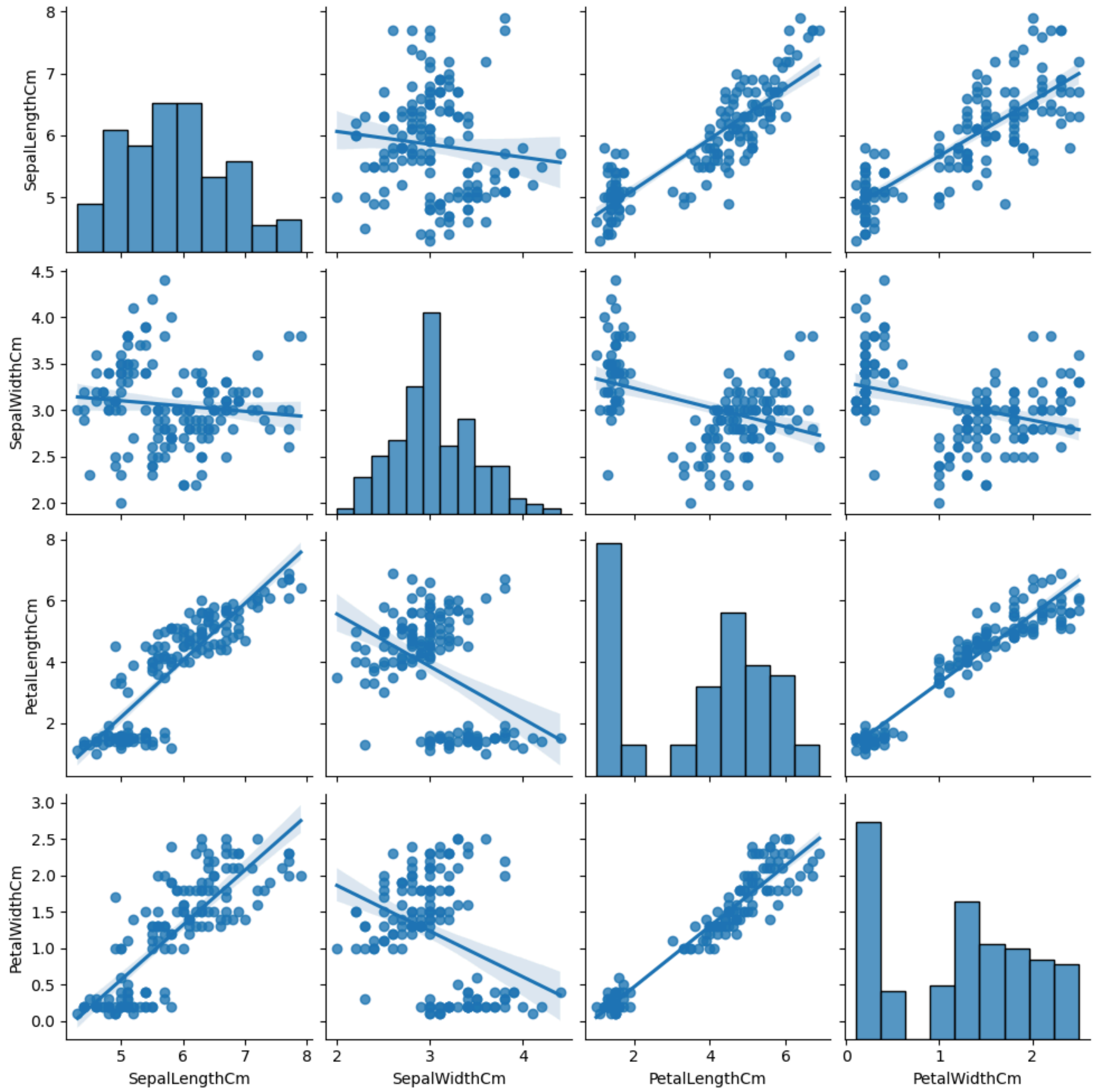
```
plt.legend('Species')

plt.tight_layout()
plt.show()
```



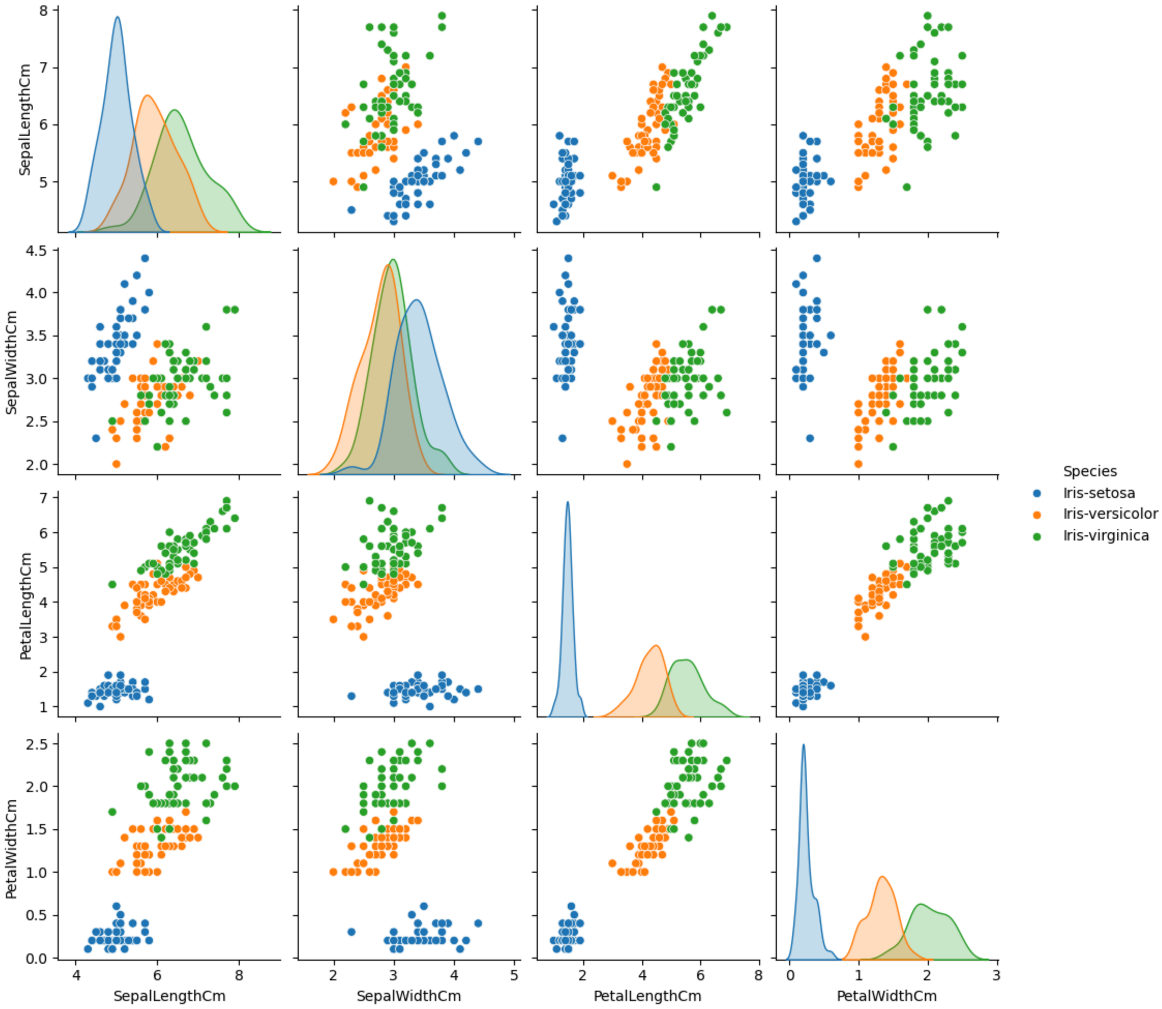
Visualization 9: Pairplot

In [29]:

```
vish9= sns.pairplot(data  = iris, kind='reg')
```



In [30]:
```
vish9= sns.pairplot(data  = iris, kind='scatter', hue='Species')
```



In [31]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

Visualization 10: Heatmap of correlation matrix

- CHECK & CONVERT ALL COLUMNS DATATYPE TO INT BEFORE USING HEAT MAP
- Import LABELENCODER

In [32]:
```
from sklearn.preprocessing import LabelEncoder
```

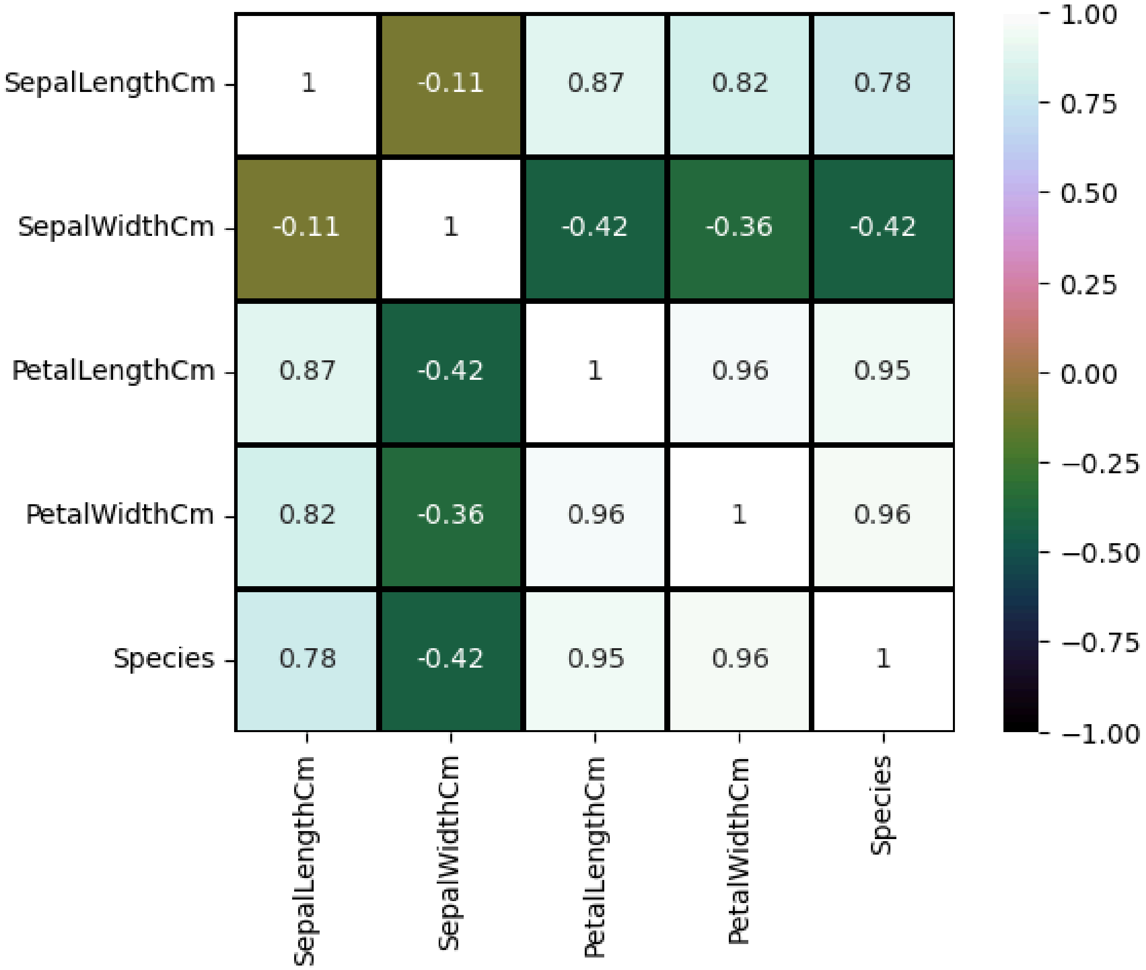
In [33]:
```
le= LabelEncoder()
```

In [34]:
```
iris['Species']=le.fit_transform(iris['Species'])
```

In [35]:
```
iris.dtypes
```

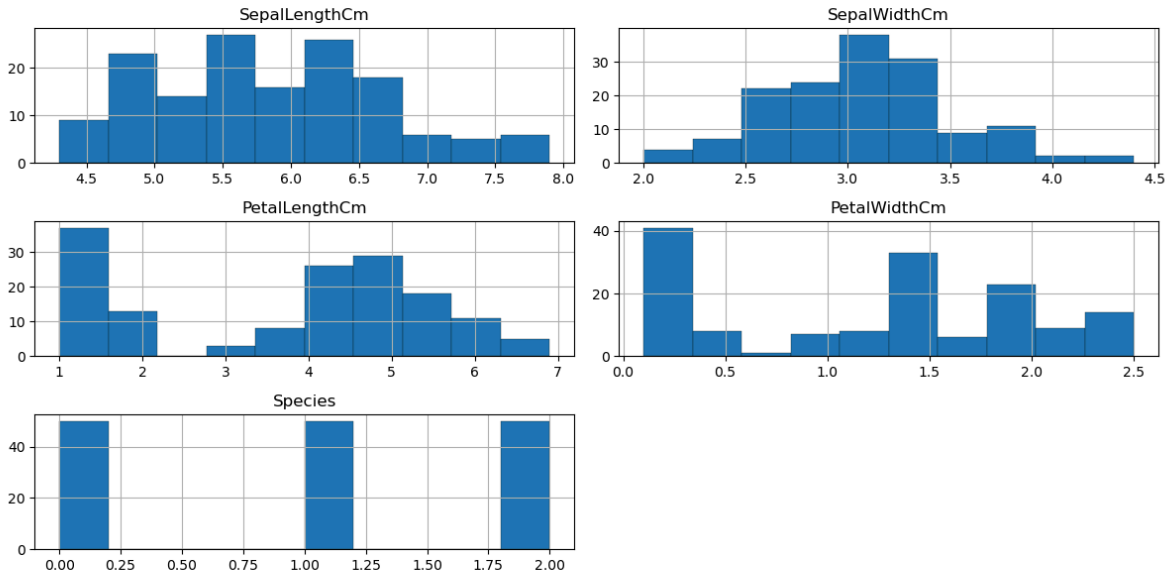
Out[35]:
```
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species            int32
dtype: object
```

In [36]:
```
vish10 = sns.heatmap(iris.corr(), annot=True,cmap='cubehelix', linewidth=1, line
```



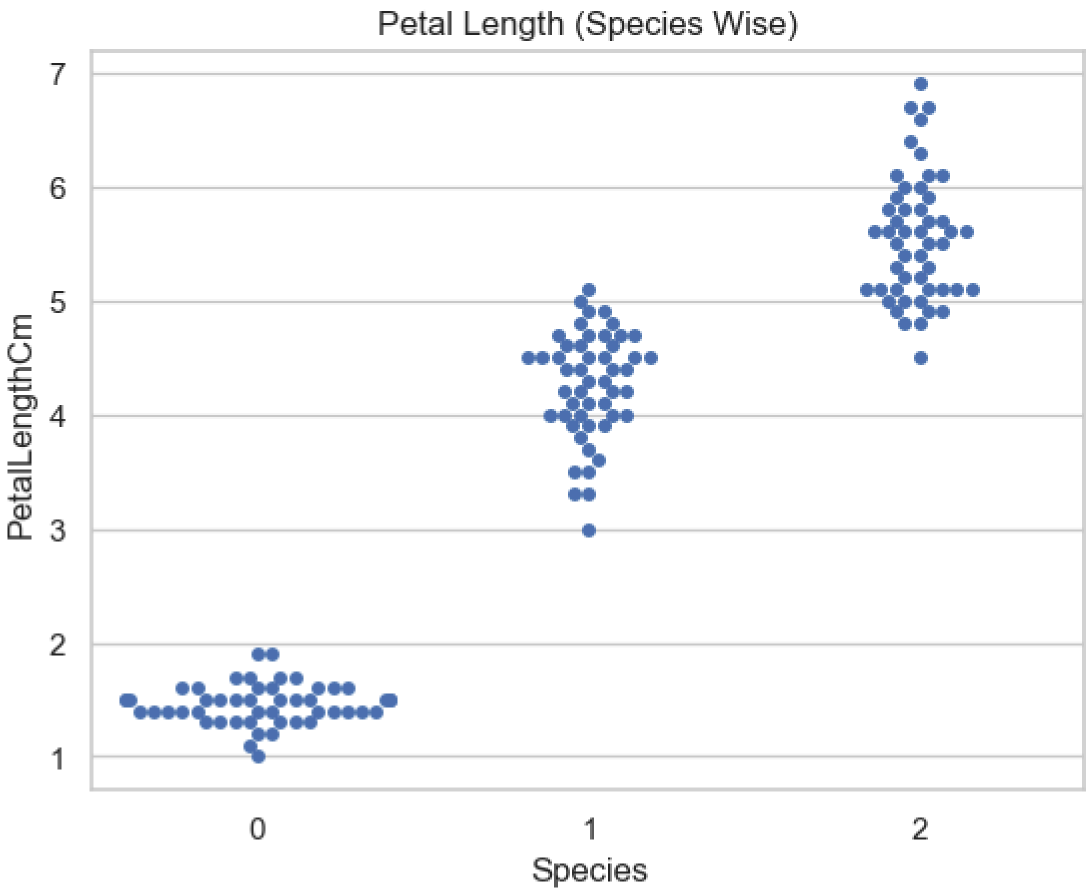
Visualization 11: Plot histograms for all columns in the Iris dataset

In [37]:
```
vish11=iris.hist(edgecolor='black',linewidth=0.2)
fig= plt.gcf()
fig.set_size_inches(12,6)
plt.tight_layout()
plt.show()
```



Visualization 12: Swarm plot to visualize Petal Length distribution by Species

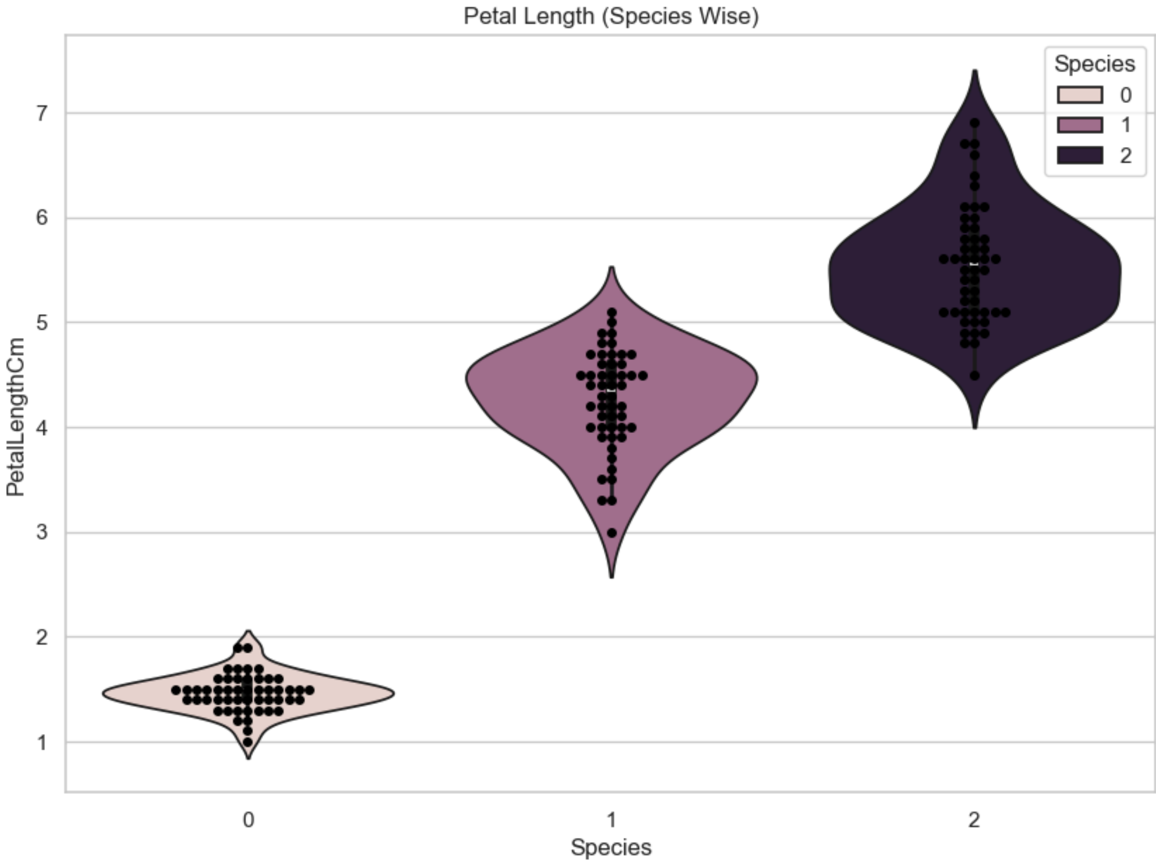
In [38]:
```
vish12 = sns.set(style='whitegrid')
vish12 = sns.swarmplot(x="Species", y="PetalLengthCm", data=iris)
plt.title('Petal Length (Species Wise)')
plt.show()
```



Visualization 13: Combining Violin plot & Swarm plot with custom color for Petal Length

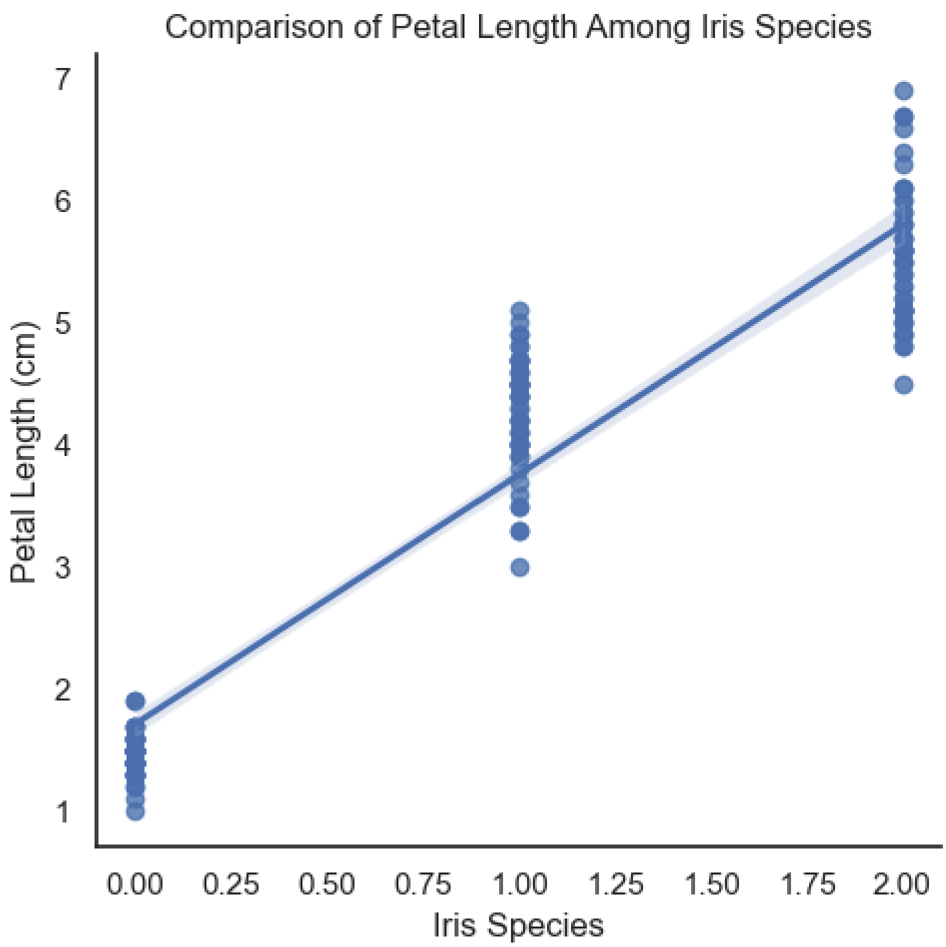
distribution

In [39]:
```
vish13 = sns.set(style='whitegrid')
vish13 = sns.swarmplot(x="Species", y="PetalLengthCm", data=iris, color= 'Black'
vish13 = sns.violinplot(x="Species", y="PetalLengthCm", data=iris,hue= 'Species'
fig= plt.gcf()
fig.set_size_inches(10,7)
plt.title('Petal Length (Species Wise)')
plt.show()
```



Visualization 14: Linear model plot to compare Petal Length among Iris species

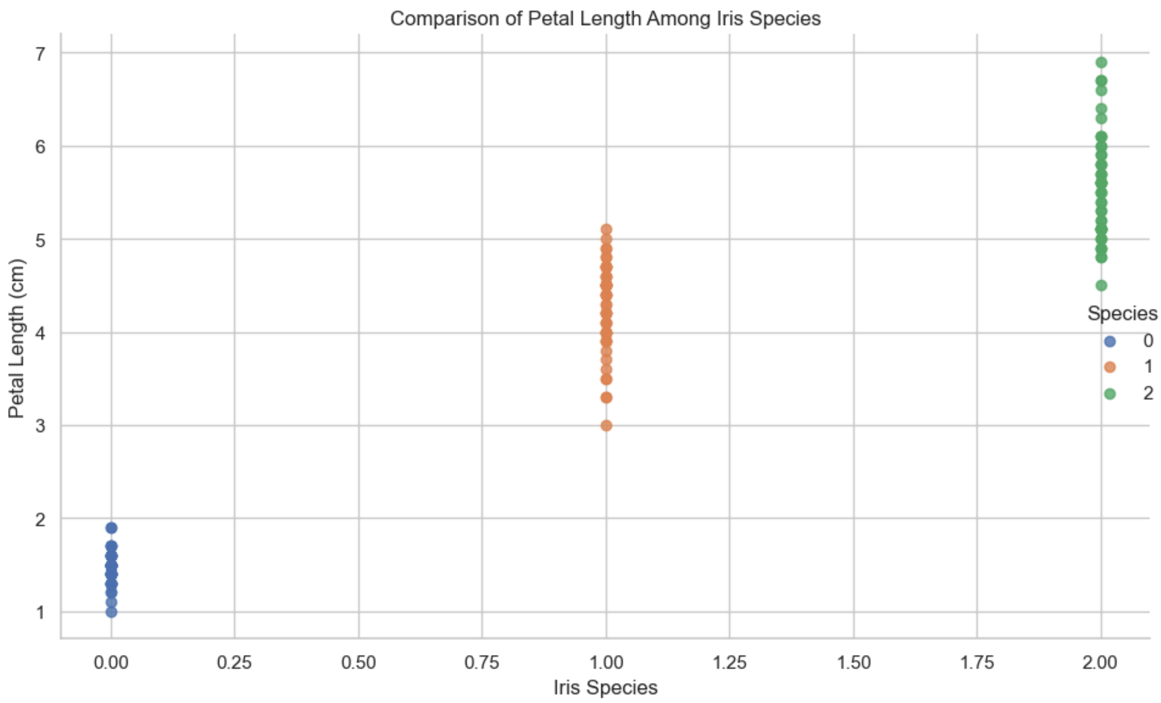
In [40]:
```
vish14 = sns.set(style='white')
vish14 = sns.lmplot(data= iris, x= 'Species', y='PetalLengthCm')

plt.title('Comparison of Petal Length Among Iris Species')
plt.xlabel('Iris Species')  # X-axis label
plt.ylabel('Petal Length (cm)')  # Y-axis label
plt.tight_layout()

plt.show()
```



Visualization 15: Linear model plot to compare Petal Length among Iris species

In [41]:

```
vish15 = sns.set(style='whitegrid')
vish15 = sns.lmplot(data= iris, x= 'Species', y='PetalLengthCm',fit_reg= False,

plt.title('Comparison of Petal Length Among Iris Species')
plt.xlabel('Iris Species')  # X-axis label
plt.ylabel('Petal Length (cm)')  # Y-axis label
plt.tight_layout()

plt.show()
```

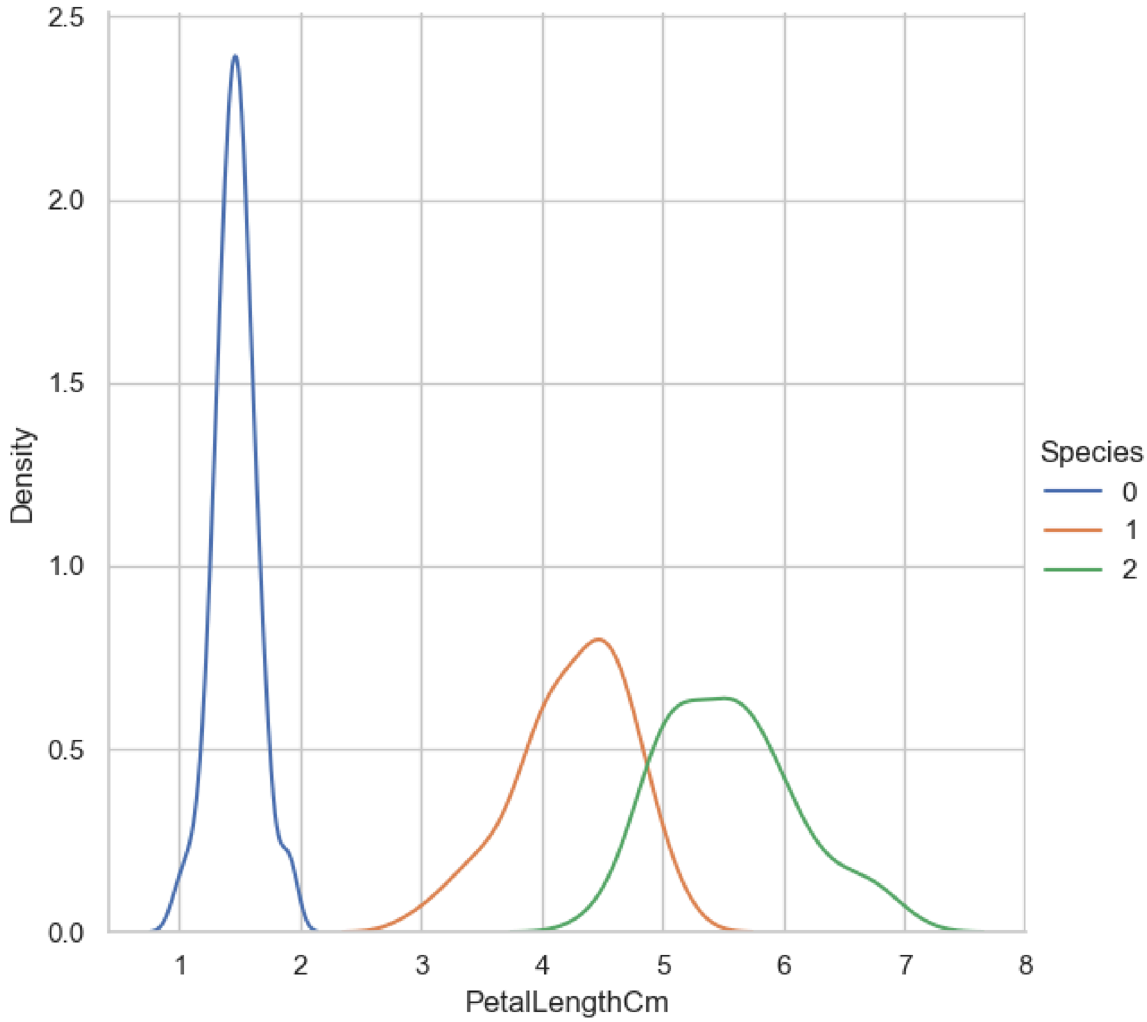


In [42]:
```
for i in iris.columns:
    print(i)
```

```
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

Visualization 16: Facegrid with KDE plot for Petal Length distribution among species

In [43]:
```
# Create the FacetGrid and apply KDE plot
vish16 = sns.FacetGrid(iris, hue="Species", height=6)
vish16.map(sns.kdeplot, "PetalLengthCm")
vish16.add_legend()

# Turn off interactive mode (optional in Jupyter)
plt.ioff()

# Show the plot
plt.show()
```

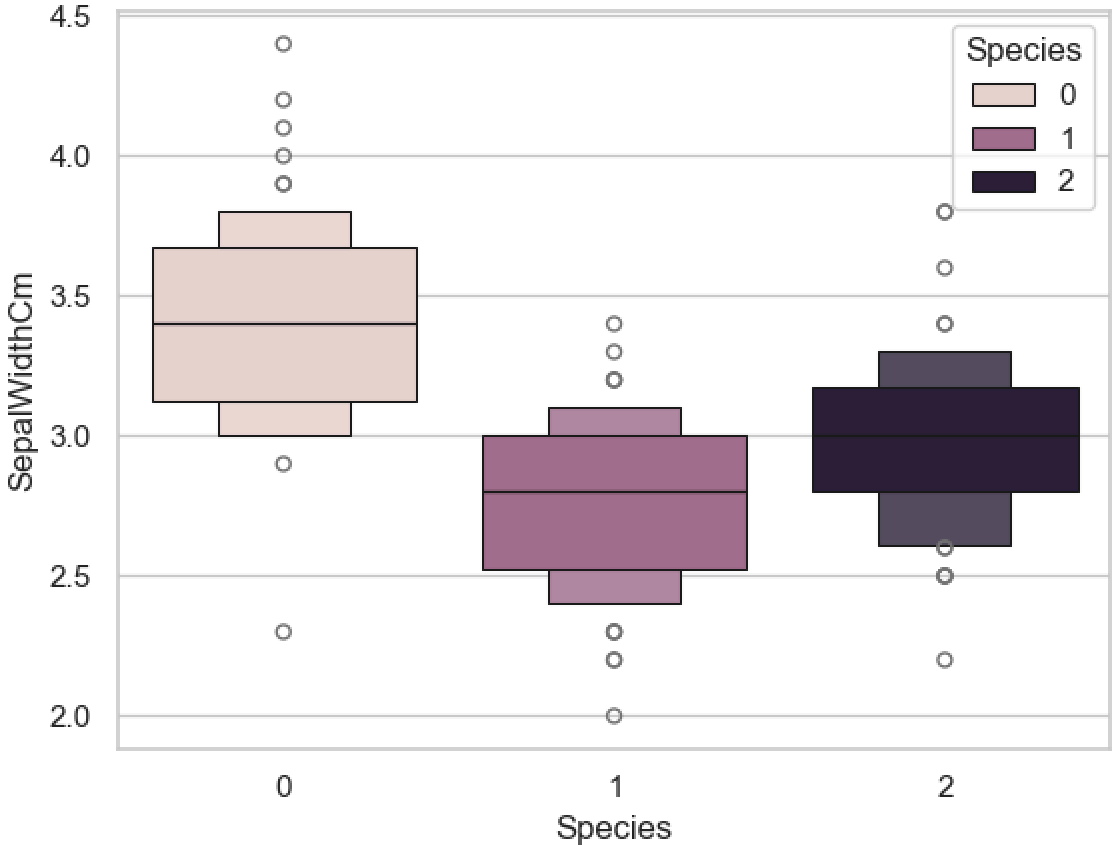


In [44]:
```
for i in iris.columns:
    print(i)
```

```
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

Visualization 17: Boxen plot for Sepal Width distribution among species

In [45]:
```
vish17 = sns.boxenplot(data=iris, x='Species', y='SepalWidthCm', hue='Species')
plt.show()
```



Create a kde plot of sepal_length versus sepal width for setosa species of flower.

In [46]:
```
iris = pd.read_csv(r'C:\Users\Shriniwas\Desktop\Data Analyst Course\Python Basic
for i in iris.columns:
    print(i)
```

```
Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
```

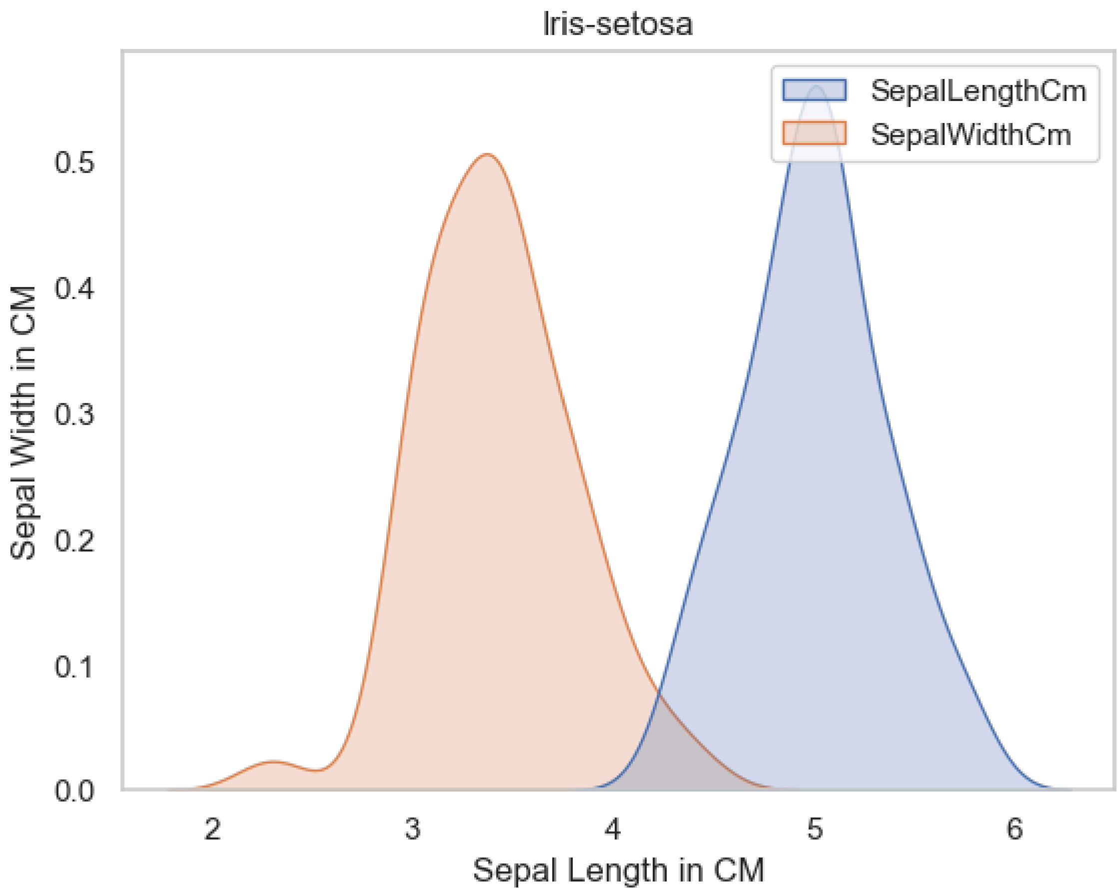
In [47]:
```
for i in iris['Species'].unique():
    print(i)
```

```
Iris-setosa
Iris-versicolor
Iris-virginica
```

Visualization 18: KDE plot of Sepal Length vs. Sepal Width for Iris-setosa species

In [48]:
```
copy = iris[iris['Species']=='Iris-setosa']
vish18 =sns.kdeplot(data=copy[['SepalLengthCm', 'SepalWidthCm']], shade= True, s
plt.title('Iris-setosa')
plt.xlabel('Sepal Length in CM')
plt.ylabel('Sepal Width in CM')
```

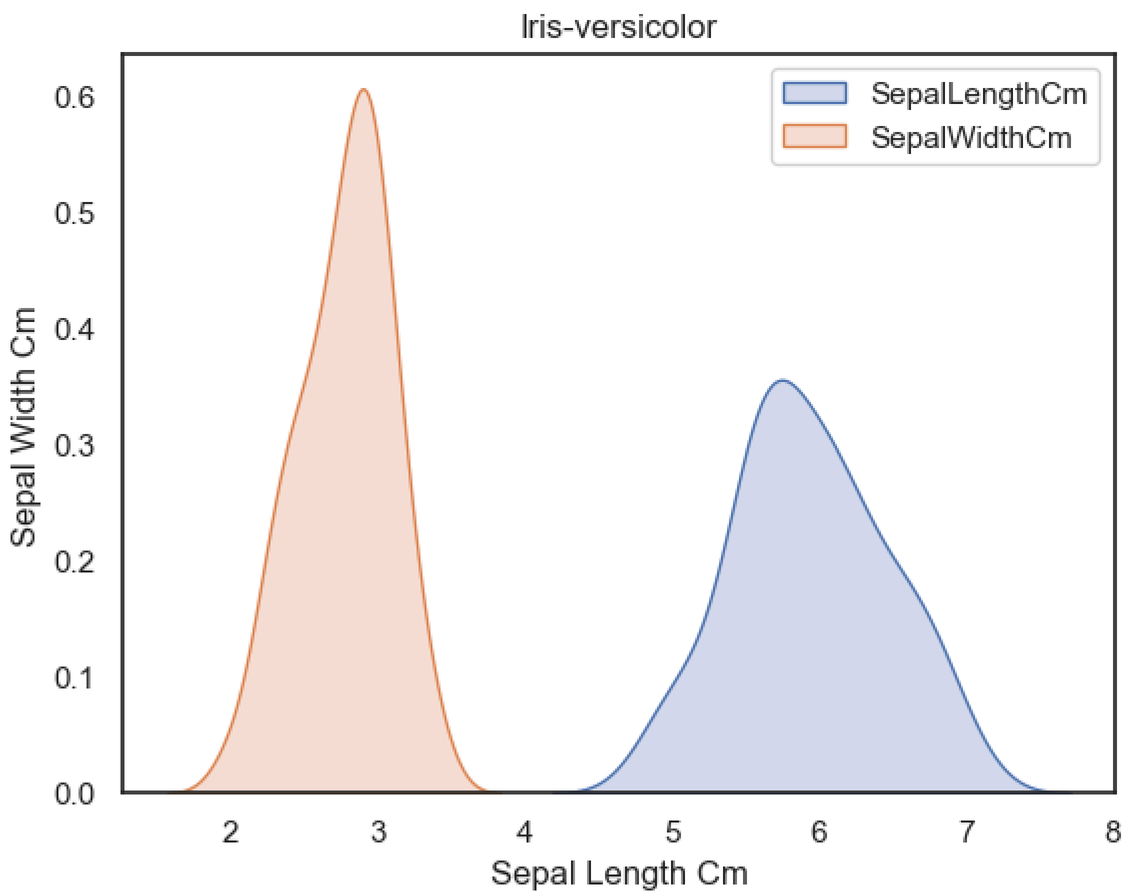
```
plt.grid(False)
plt.show()
```



Visualization 19: KDE plot of Sepal Length vs. Sepal Width for Iris-versicolor species

In [49]:

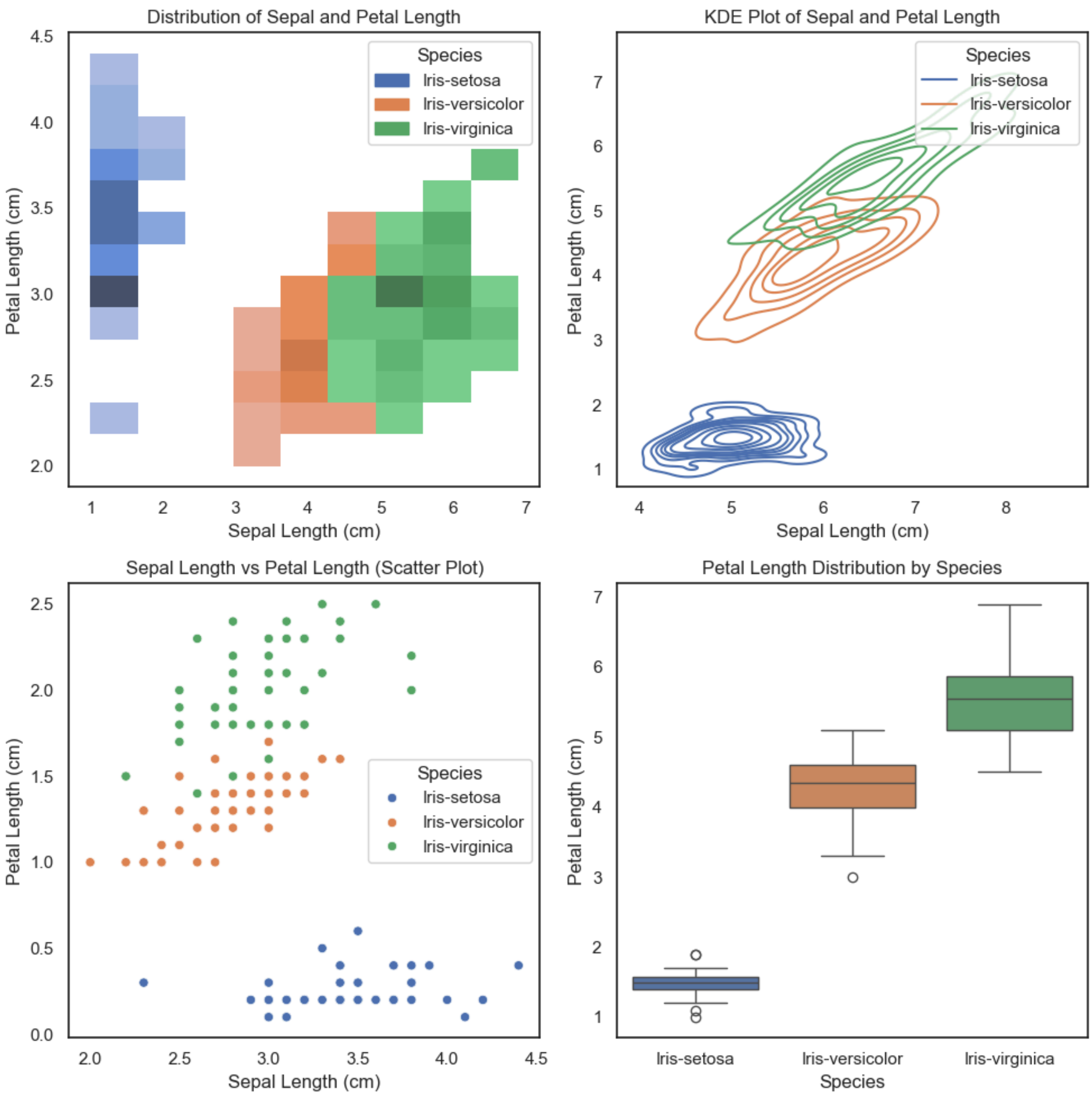
```
# Create a kde plot of sepal_length versus sepal width for setosa species of flo
sns.set_style('white')
sub=iris[iris['Species']=='Iris-versicolor']
vish19 = sns.kdeplot(data=sub[['SepalLengthCm','SepalWidthCm']], shade=True, sha
plt.title('Iris-versicolor')
plt.xlabel('Sepal Length Cm')
plt.ylabel('Sepal Width Cm')

plt.grid(False)
plt.show()
```



Visualization 20: Subplot to visualize various features of Iris dataset

In [50]:

```
f, axes = plt.subplots(2,2, figsize=(10,10))
col1 = sns.histplot(data=iris,x= 'PetalLengthCm', y='SepalWidthCm', hue='Species
col1.set_title("Distribution of Sepal and Petal Length")
col1.set_xlabel("Sepal Length (cm)")
col1.set_ylabel("Petal Length (cm)")

col2 = sns.kdeplot(data=iris,x= 'SepalLengthCm', y='PetalLengthCm', hue='Species
col2.set_title("KDE Plot of Sepal and Petal Length")
col2.set_xlabel("Sepal Length (cm)")
col2.set_ylabel("Petal Length (cm)")

col3 = sns.scatterplot(data=iris,x= 'SepalWidthCm', y='PetalWidthCm', hue='Speci
col3.set_title("Sepal Length vs Petal Length (Scatter Plot)")
col3.set_xlabel("Sepal Length (cm)")
col3.set_ylabel("Petal Length (cm)")

col4 = sns.boxplot(data=iris,x= 'Species', y='PetalLengthCm', hue='Species', ax=
col4.set_title("Petal Length Distribution by Species")
col4.set_xlabel("Species")
col4.set_ylabel("Petal Length (cm)")

plt.tight_layout()
plt.show()
```



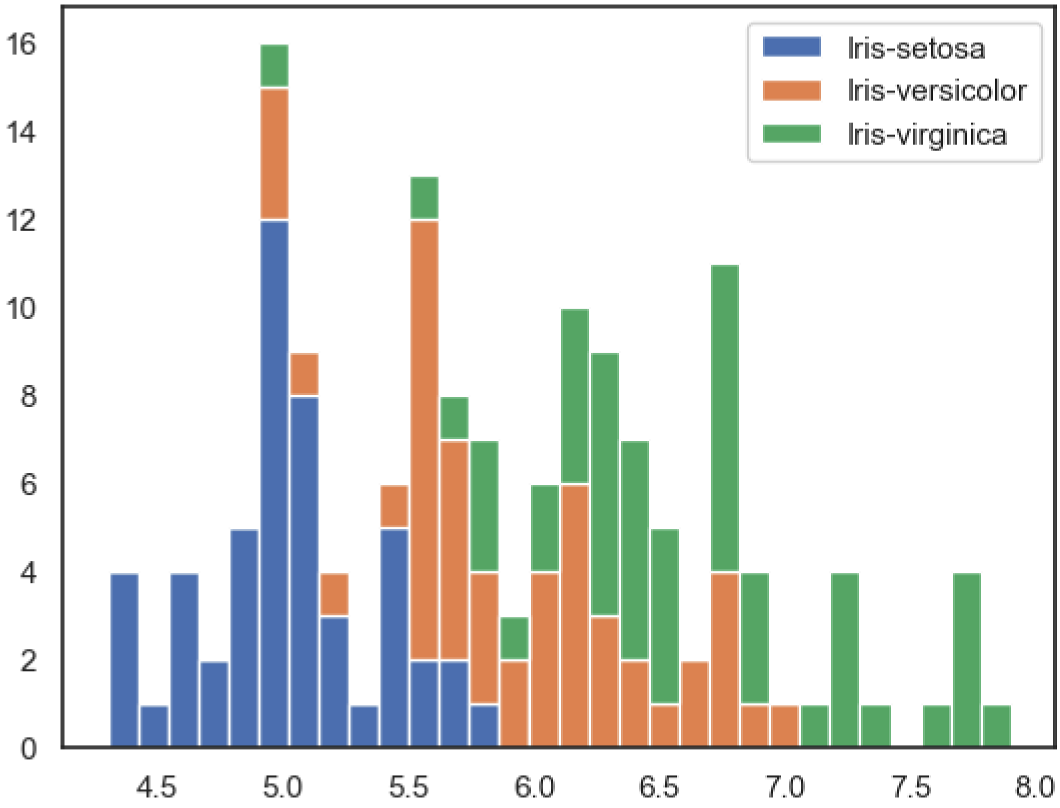
Visualization 21: Stacked histogram for Sepal Length by species

In [51]:
```
iris['Species'] = iris['Species'].astype('category')
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype   
---  ------         --------------  -----   
 0   Id             150 non-null    int64   
 1   SepalLengthCm  150 non-null    float64 
 2   SepalWidthCm   150 non-null    float64 
 3   PetalLengthCm  150 non-null    float64 
 4   PetalWidthCm   150 non-null    float64 
 5   Species        150 non-null    category
dtypes: category(1), float64(4), int64(1)
memory usage: 6.3 KB
```

In [52]:
```
list1=list()
mylabels=list()
for gen in iris.Species.cat.categories:
```

```
    list1.append(iris[iris.Species==gen].SepalLengthCm)
    mylabels.append(gen)

vish21=plt.hist(list1,bins=30,stacked=True,rwidth=1,label=mylabels)
plt.legend()
plt.show()
```



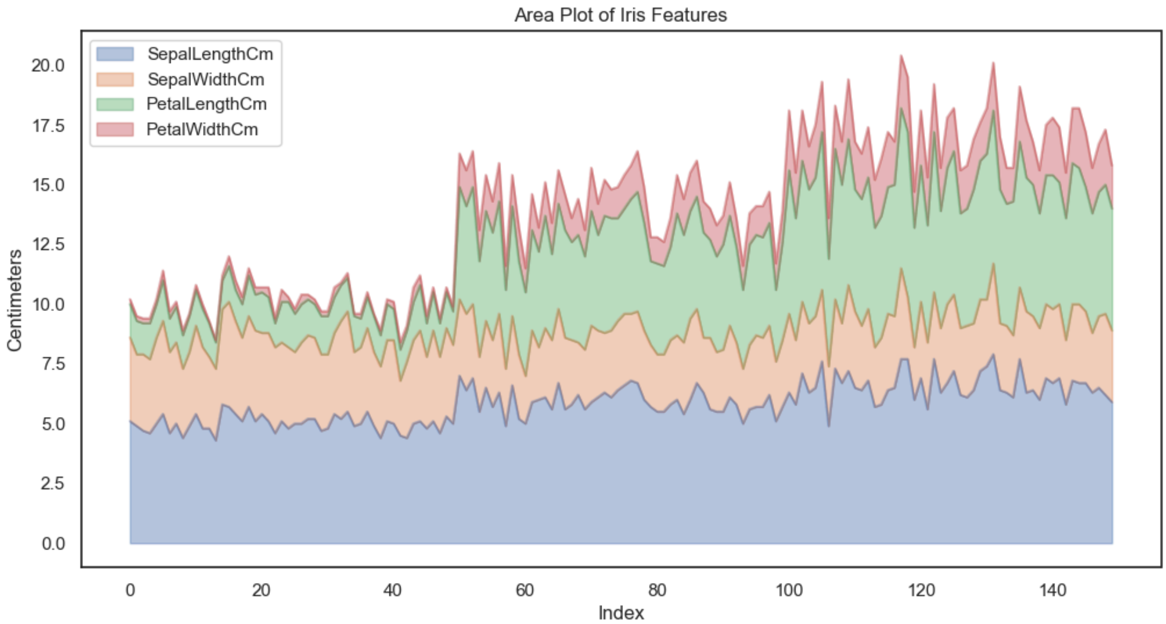
Visualization 22: Area plot for Iris features

In [53]:
```
iris['Species']=le.fit_transform(iris['Species'])

iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             150 non-null    int64
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    int32
dtypes: float64(4), int32(1), int64(1)
memory usage: 6.6 KB
```

In [54]:
```
vish22 =iris.plot.area(y=['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'Pet

# Add title and labels
plt.title('Area Plot of Iris Features')
plt.xlabel('Index')
plt.ylabel('Centimeters')
```

```
# Show the plot
plt.show()
```

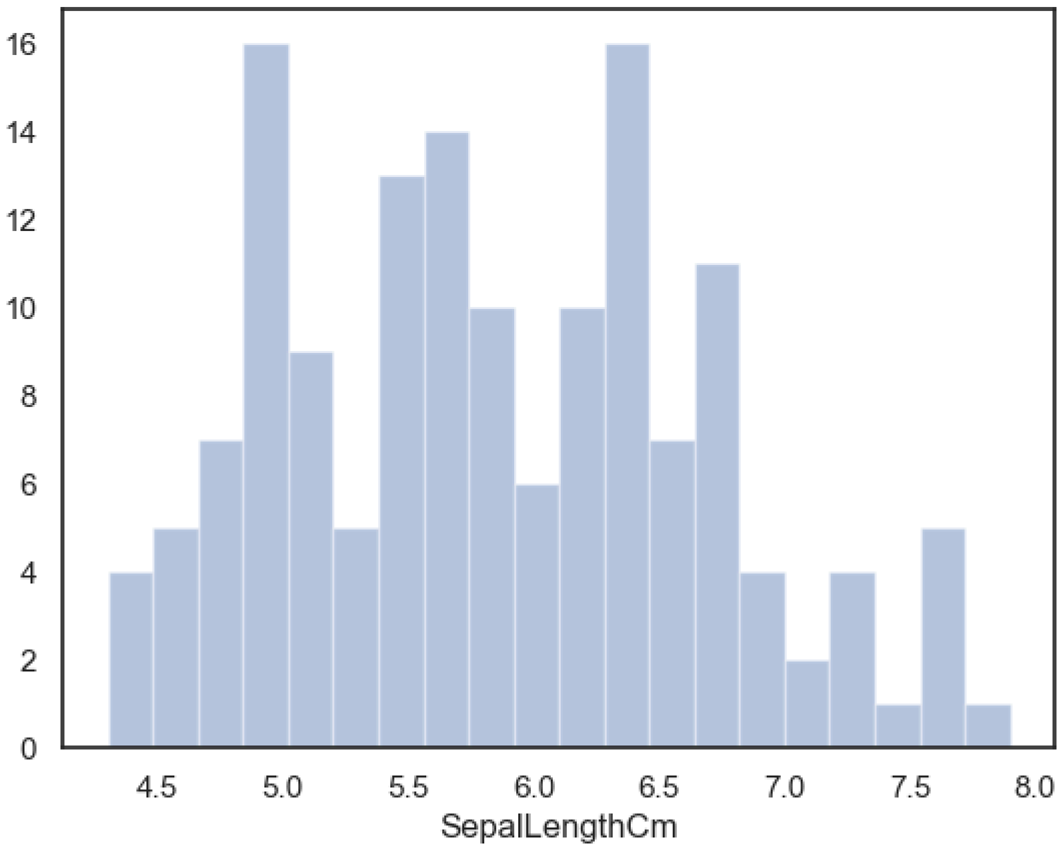


In [55]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    int32  
dtypes: float64(4), int32(1), int64(1)
memory usage: 6.6 KB
```

Visualization 23: Distplot for Sepal Length with specified bins

In [56]:
```
vish23 = sns.distplot(iris['SepalLengthCm'],kde=False,bins=20)
plt.show()
```



Iris Dataset Analysis Summary

Overview

The Iris dataset is a classic dataset in machine learning and data analysis. It contains measurements of sepals and petals for three species of Iris flowers:

1. **Setosa**
2. **Versicolor**
3. **Virginica**

This project aimed to explore, visualize, and analyze the dataset to uncover patterns and relationships among the features.

Key Findings

1. **Species Distribution**: The dataset contains balanced samples for all three species.
2. **Feature Relationships**:
    - Sepal length and sepal width show moderate correlation across species.
    - Petal length and petal width exhibit strong positive correlations.
3. **Distinct Patterns**:
    - Iris-Setosa is easily separable from the other two species based on petal dimensions.

- Versicolor and Virginica have overlapping characteristics, making them harder to distinguish.

4. **Feature Importance**:
    - Petal dimensions (length and width) are key differentiators among the species.

Visualizations

The following visualizations were created to aid understanding:

- **Histograms**: To observe the distribution of individual features.
- **Pair Plots**: To visualize relationships and separations between species.
- **Violin Plots**: To analyze feature distribution across species.
- **Heatmap**: To identify correlations among features.
- **KDE Plots**: To explore density distributions for specific species.

Conclusion

This analysis demonstrates the power of visualization and statistical methods in exploring datasets. The Iris dataset highlights how features can be leveraged to distinguish between classes in classification tasks.

Thank you for exploring the Iris dataset with us!