In [2]:

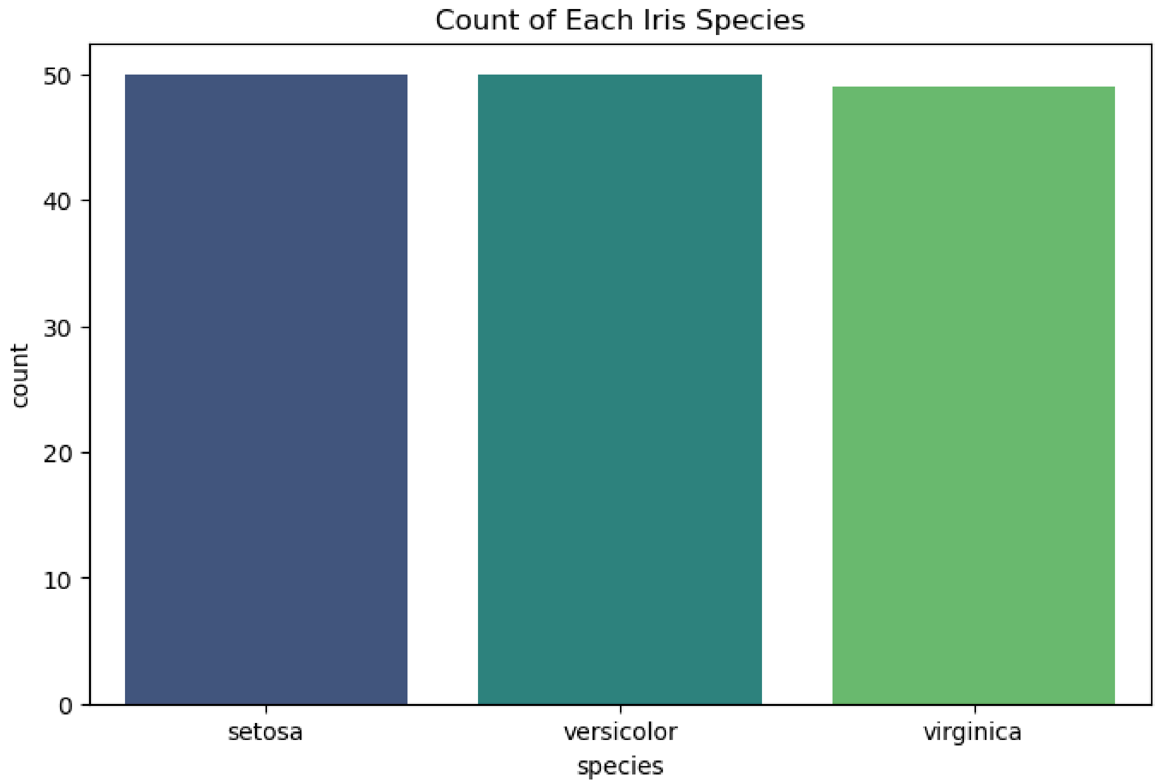
```
# ------------------------------------------------------------
# 📊 DATA ANALYTICS PROJECT: Exploratory Data Analysis on IRIS Dataset
# ------------------------------------------------------------
# Author: Avanish Tripathi
# Course: BCA (Data Science + AI)
# Project Type: Data Analytics / EDA
# ------------------------------------------------------------

# 🧭 PROJECT OBJECTIVE
# The objective of this project is to perform Exploratory Data Analysis (EDA)
# on the Iris dataset to understand the relationship between various features,
# detect patterns, visualize data distributions, and identify any outliers.

# ------------------------------------------------------------
# Step 1: Importing Required Libraries
# ------------------------------------------------------------
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# ------------------------------------------------------------
# Step 2: Fetching Dataset (from online source)
# ------------------------------------------------------------
dataset = pd.read_csv("https://raw.githubusercontent.com/mwaskom/seaborn-data/ma

print(" 🔹 First 5 Rows of the Dataset:")
print(dataset.head())

# ------------------------------------------------------------
# Step 3: Dataset Overview
# ------------------------------------------------------------
print("\n📏 Dataset Shape:", dataset.shape)
print("\nℹ️ Dataset Information:")
print(dataset.info())
print("\n📊 Statistical Summary:")
print(dataset.describe())

# ------------------------------------------------------------
# Step 4: Data Cleaning
# ------------------------------------------------------------
print("\n❓ Missing Values in Dataset:")
print(dataset.isnull().sum())

print("\n🔁 Checking for Duplicate Values:")
print(dataset.duplicated().sum())

# Removing duplicates if any
dataset = dataset.drop_duplicates()
print("✅ Duplicates removed (if present).")

# ------------------------------------------------------------
# Step 5: Species Distribution
# ------------------------------------------------------------
print("\n🌸 Species Count:")
print(dataset['species'].value_counts())

plt.figure(figsize=(8,5))
```

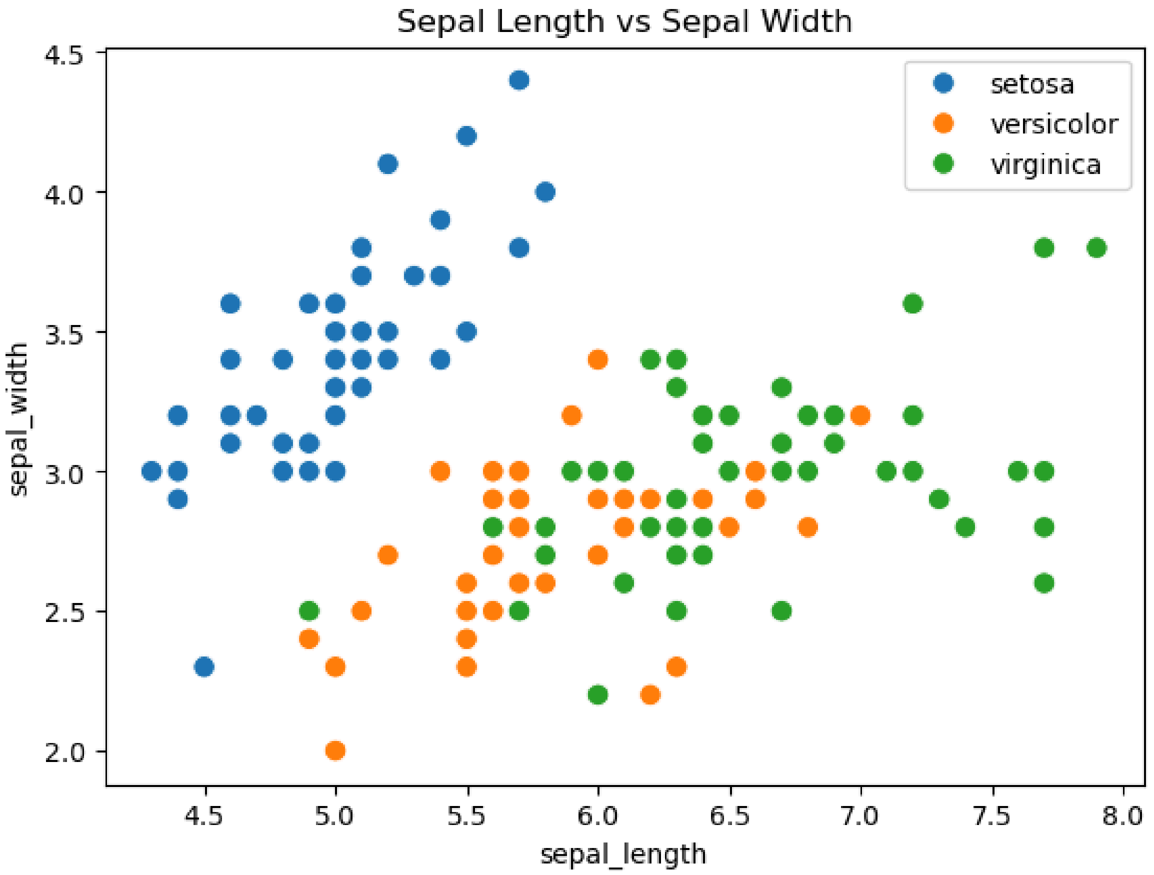
```
sns.countplot(x='species', data=dataset, palette='viridis')
plt.title("Count of Each Iris Species")
plt.show()

# ------------------------------------------------------------
# Step 6: Relationship Between Variables
# ------------------------------------------------------------
# Sepal Length vs Sepal Width
plt.figure(figsize=(7,5))
sns.scatterplot(x='sepal_length', y='sepal_width', hue='species', data=dataset,
plt.title("Sepal Length vs Sepal Width")
plt.legend(bbox_to_anchor=(1,1))
plt.show()

# Petal Length vs Petal Width
plt.figure(figsize=(7,5))
sns.scatterplot(x='petal_length', y='petal_width', hue='species', data=dataset,
plt.title("Petal Length vs Petal Width")
plt.legend(bbox_to_anchor=(1,1))
plt.show()

# Pairplot for Multivariate Relationships
sns.pairplot(dataset, hue='species', diag_kind='hist')
plt.show()

# ------------------------------------------------------------
# Step 7: Histograms (Distribution of Features)
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 2, figsize=(10,8))
axes[0,0].hist(dataset['sepal_length'], bins=10, color='skyblue')
axes[0,0].set_title("Sepal Length")
axes[0,1].hist(dataset['sepal_width'], bins=10, color='orange')
axes[0,1].set_title("Sepal Width")
axes[1,0].hist(dataset['petal_length'], bins=10, color='green')
axes[1,0].set_title("Petal Length")
axes[1,1].hist(dataset['petal_width'], bins=10, color='red')
axes[1,1].set_title("Petal Width")
plt.tight_layout()
plt.show()

# ------------------------------------------------------------
# Step 8: Correlation & Heatmap
# ------------------------------------------------------------
corr = dataset.select_dtypes(include=['float64']).corr()
print("\n📈 Correlation Matrix:\n", corr)

plt.figure(figsize=(6,5))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title("Heatmap of Feature Correlations")
plt.show()

# ------------------------------------------------------------
# Step 9: Boxplots (Detect Outliers)
# ------------------------------------------------------------
def box_graph(y):
    sns.boxplot(x="species", y=y, data=dataset, palette="Set2")

plt.figure(figsize=(10,10))
plt.subplot(221)
box_graph('sepal_length')
```

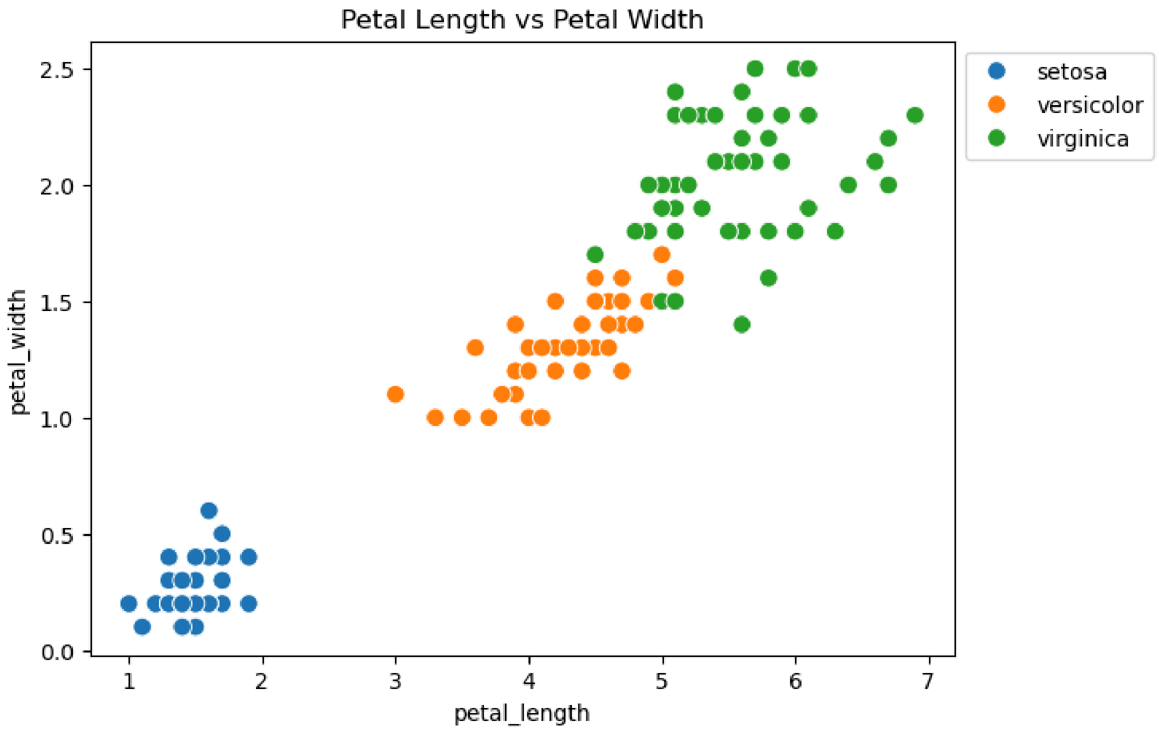
```
plt.subplot(222)
box_graph('sepal_width')
plt.subplot(223)
box_graph('petal_length')
plt.subplot(224)
box_graph('petal_width')
plt.tight_layout()
plt.show()

# -------------------------------------------------------------
# Step 10: Outlier Detection & Removal (Using IQR)
# -------------------------------------------------------------
col = 'sepal_width'
Q1 = np.percentile(dataset[col], 25)
Q3 = np.percentile(dataset[col], 75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

print(f"\n🔍 Detecting Outliers in {col}:")
print(f"Lower Bound = {lower_bound}, Upper Bound = {upper_bound}")

print("Old Shape:", dataset.shape)
filtered_data = dataset[(dataset[col] >= lower_bound) & (dataset[col] <= upper_b
print("New Shape after removing outliers:", filtered_data.shape)

plt.figure(figsize=(7,5))
sns.boxplot(x=filtered_data[col])
plt.title(f"Boxplot of {col} After Removing Outliers")
plt.show()

# -------------------------------------------------------------
# Step 11: Insights & Conclusion
# -------------------------------------------------------------
print("\n📘 PROJECT CONCLUSION:")
print("""
✅ The Iris dataset contains 3 species — Setosa, Versicolor, Virginica.
✅ Petal measurements are the strongest differentiators among species.
✅ Petal length and petal width show high correlation.
✅ Setosa flowers have the smallest petals, Virginica the largest.
✅ Few outliers detected in Sepal Width were successfully removed.
""")

print("\n The dataset is now clean, well-understood, and ready for Machine Learn

# -------------------------------------------------------------
# END OF PROJECT
# -------------------------------------------------------------
```

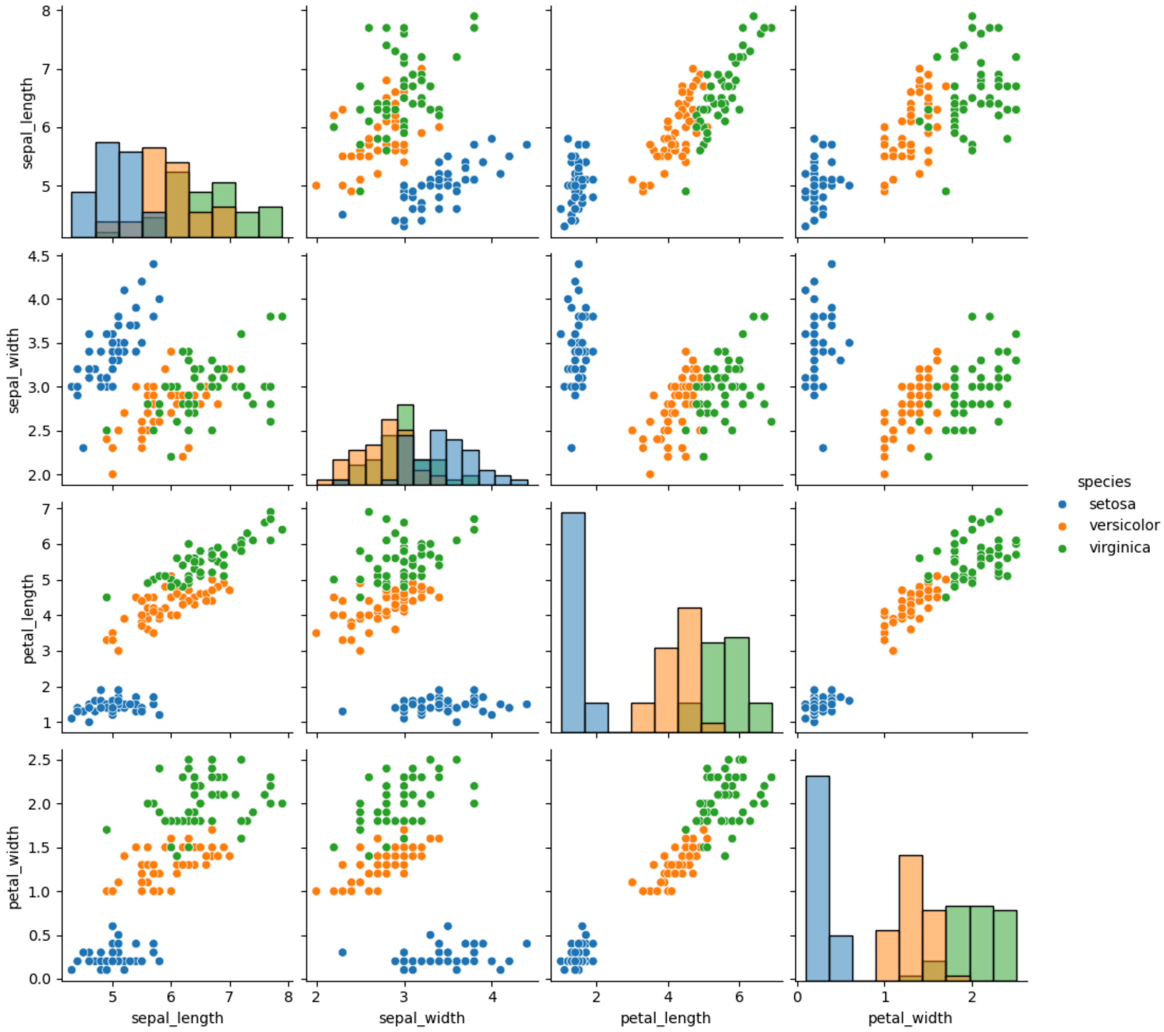
```
🔹 First 5 Rows of the Dataset:
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa

📏 Dataset Shape: (150, 5)

ℹ️ Dataset Information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
None

📊 Statistical Summary:
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.057333      3.758000     1.199333
std        0.828066     0.435866      1.765298     0.762238
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000

❓ Missing Values in Dataset:
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64

🔁 Checking for Duplicate Values:
1
✅ Duplicates removed (if present).

🌸 Species Count:
species
setosa        50
versicolor    50
virginica     49
Name: count, dtype: int64
```

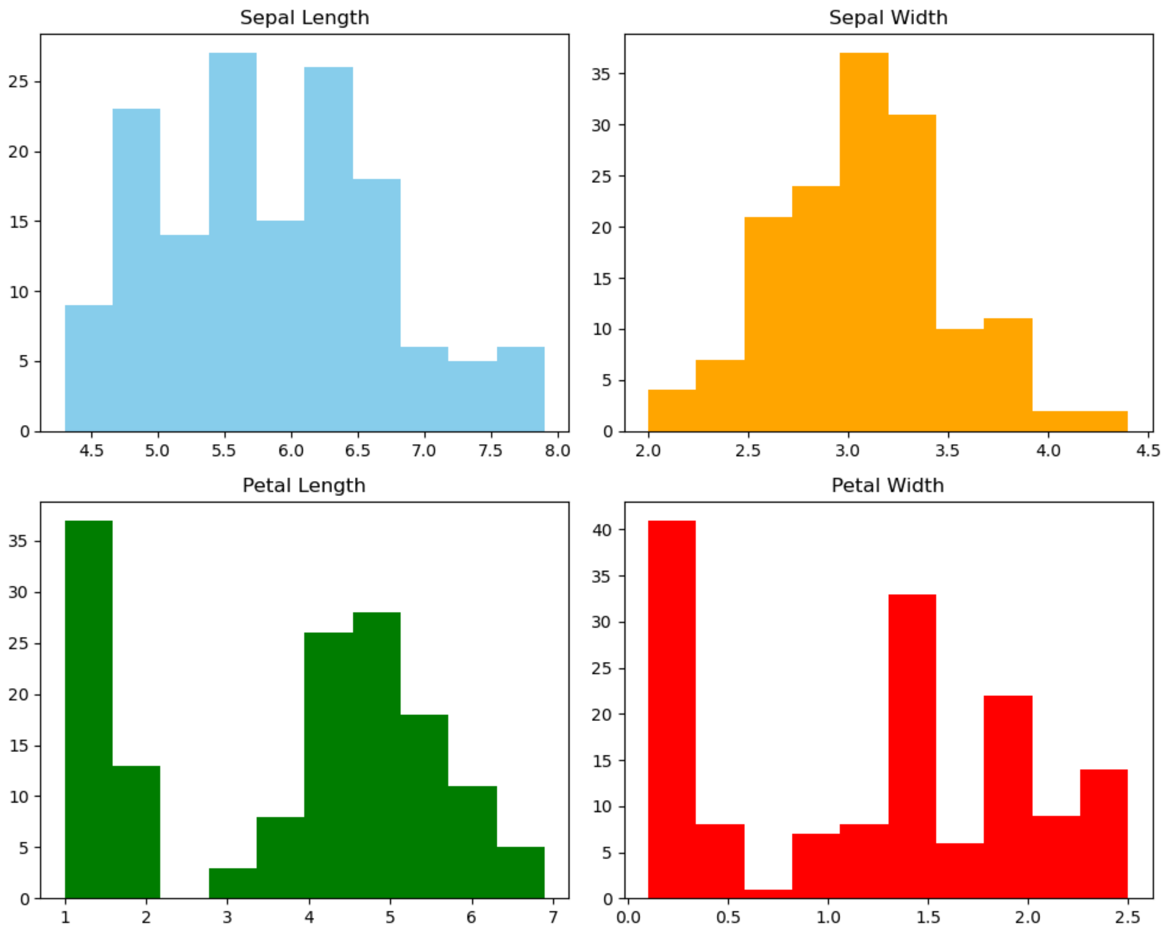
```
C:\Users\avani\AppData\Local\Temp\ipykernel_16832\133258735.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.countplot(x='species', data=dataset, palette='viridis')
```



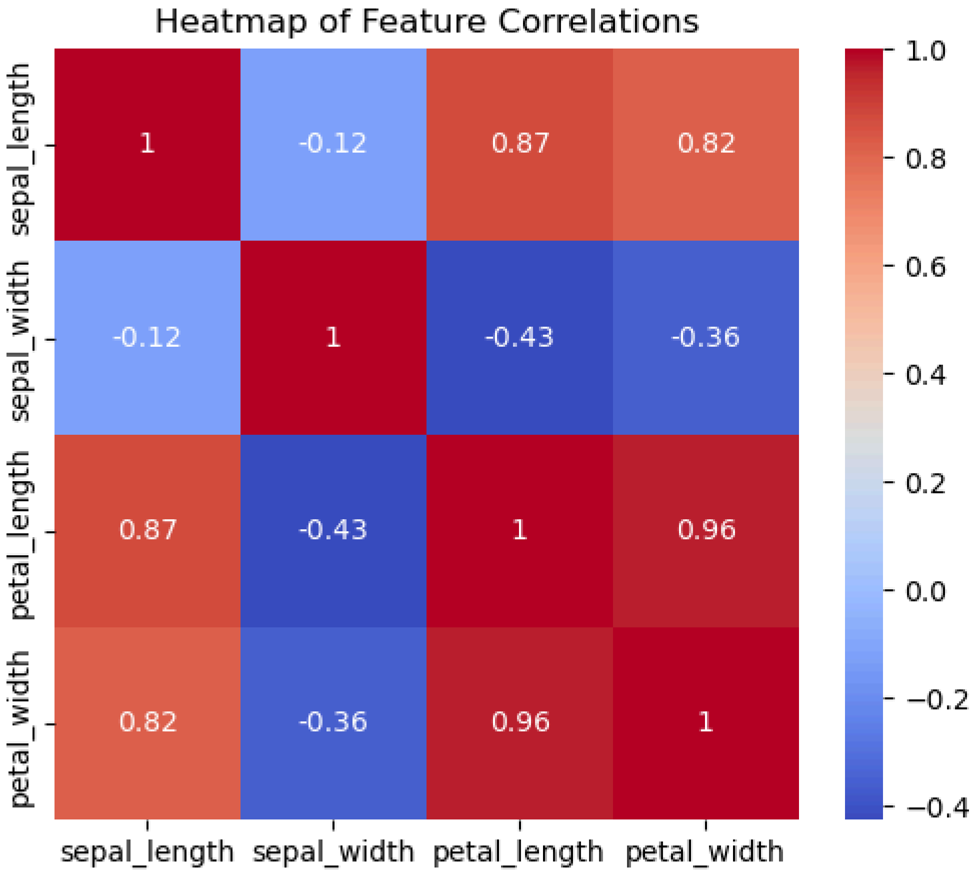



Petal Length vs Petal Width
petal_width
0.0
0.5
1.0
1.5
2.0
2.5
petal_length
1
2
3
4
5
6
7
setosa
versicolor
virginica

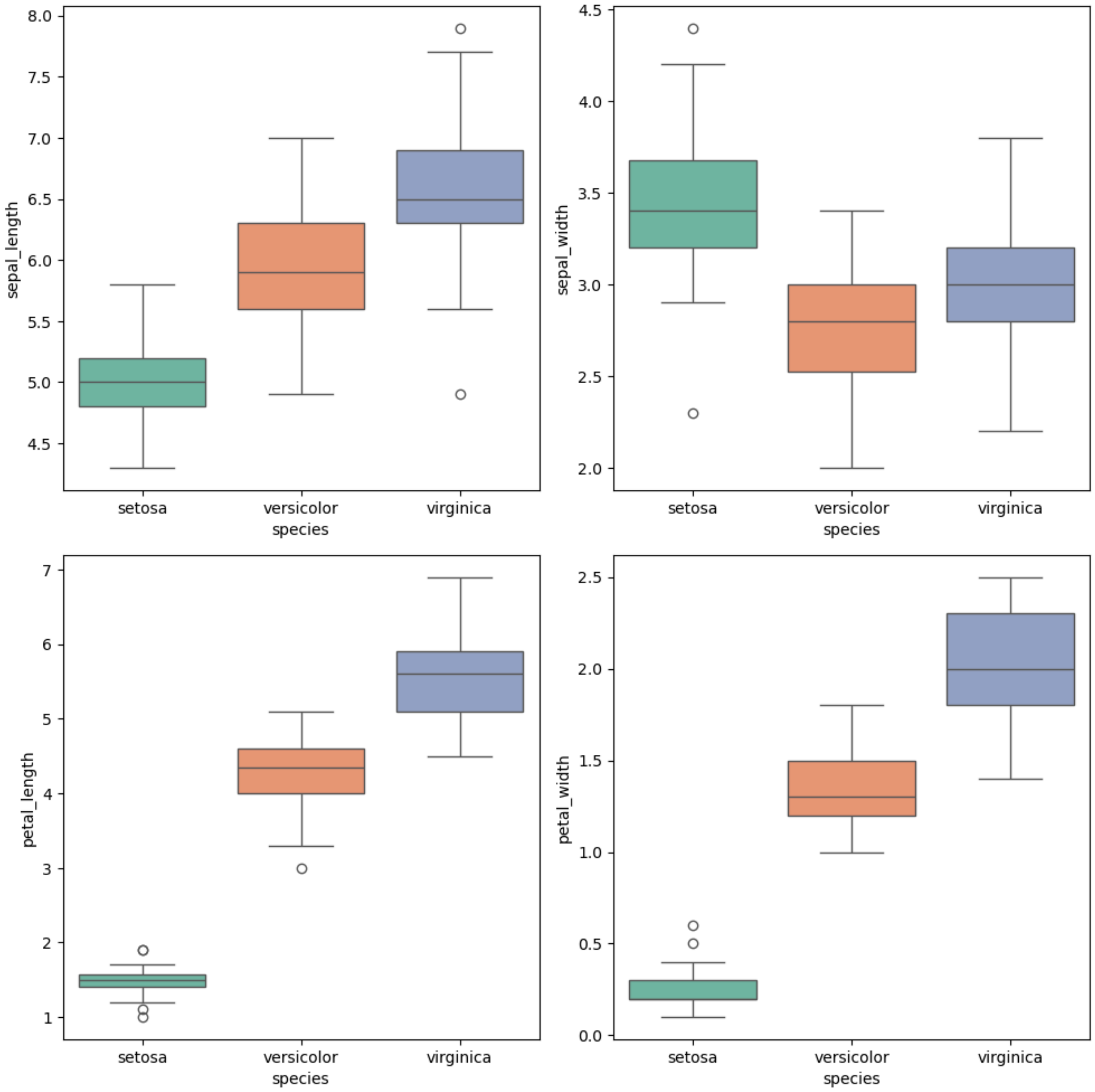

sepal_length
sepal_width
petal_length
petal_width
8
7
6
5
4.5
4.0
3.5
3.0
2.5
2.0
7
6
5
4
3
2
1
2.5
2.0
1.5
1.0
0.5
0.0
5
6
7
8
2
3
4
2
4
6
0
1
2
sepal_length
sepal_width
petal_length
petal_width
species
setosa
versicolor
virginica



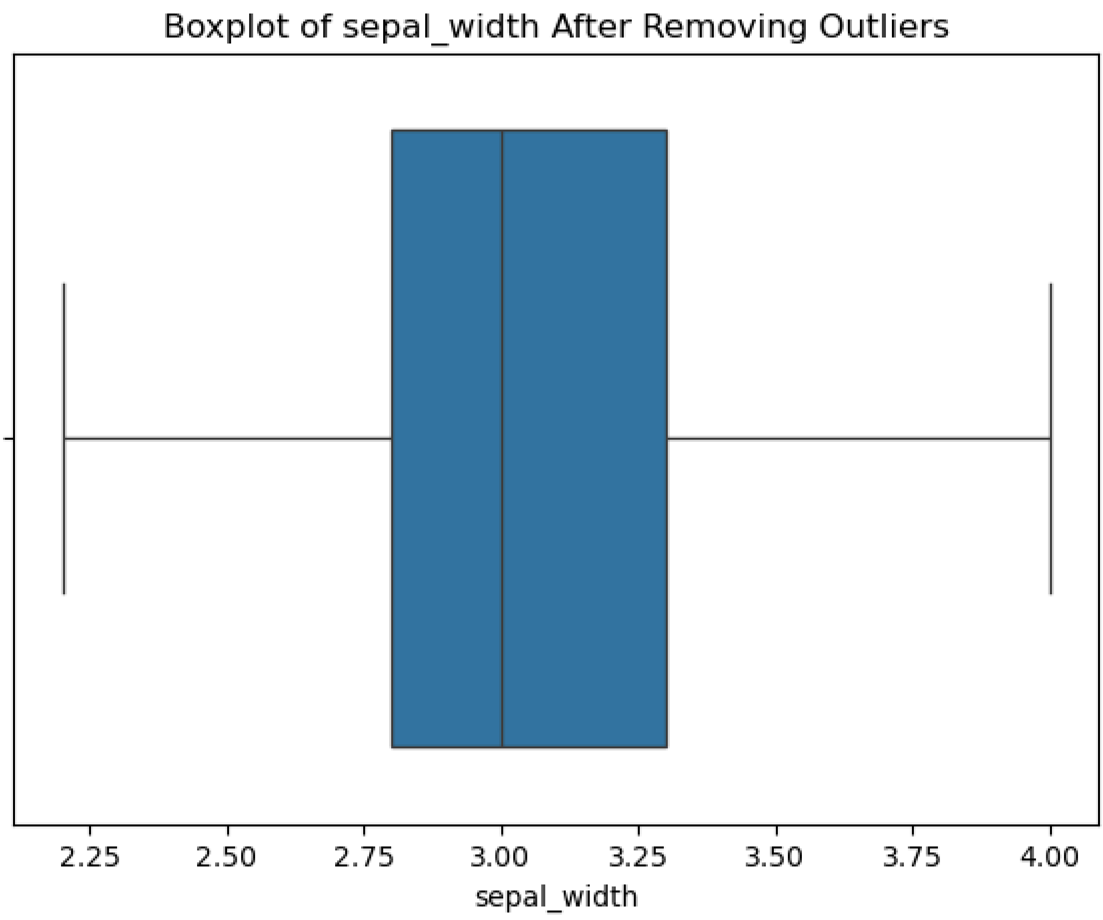
```
📈 Correlation Matrix:
              sepal_length  sepal_width  petal_length  petal_width
sepal_length      1.000000    -0.118129      0.873738     0.820620
sepal_width      -0.118129     1.000000     -0.426028    -0.362894
petal_length      0.873738    -0.426028      1.000000     0.962772
petal_width       0.820620    -0.362894      0.962772     1.000000
```



```
C:\Users\avani\AppData\Local\Temp\ipykernel_16832\133258735.py:114: FutureWarnin
g:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.boxplot(x="species", y=y, data=dataset, palette="Set2")
C:\Users\avani\AppData\Local\Temp\ipykernel_16832\133258735.py:114: FutureWarnin
g:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.boxplot(x="species", y=y, data=dataset, palette="Set2")
C:\Users\avani\AppData\Local\Temp\ipykernel_16832\133258735.py:114: FutureWarnin
g:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.boxplot(x="species", y=y, data=dataset, palette="Set2")
C:\Users\avani\AppData\Local\Temp\ipykernel_16832\133258735.py:114: FutureWarnin
g:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.boxplot(x="species", y=y, data=dataset, palette="Set2")
```



```
🔍 Detecting Outliers in sepal_width:
Lower Bound = 2.05, Upper Bound = 4.05
Old Shape: (149, 5)
New Shape after removing outliers: (145, 5)
```



```
📘 PROJECT CONCLUSION:

✅ The Iris dataset contains 3 species — Setosa, Versicolor, Virginica.
✅ Petal measurements are the strongest differentiators among species.
✅ Petal length and petal width show high correlation.
✅ Setosa flowers have the smallest petals, Virginica the largest.
✅ Few outliers detected in Sepal Width were successfully removed.


 The dataset is now clean, well-understood, and ready for Machine Learning tasks
such as classification!
```

In [ ]:

In [ ]: