Data Visualization with Matplotlib and Seaborn using the Iris Dataset

Introduction to Matplotlib :

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. It is widely used for plotting data and offers great flexibility and control over the appearance of plots. Matplotlib can produce a variety of plots such as line plots, scatter plots, histograms, bar charts, and more.

Basic Syntax: To use Matplotlib, first import the pyplot module, which provides a MATLAB-like interface for creating plots:

import matplotlib.pyplot as plt

Common Plot Types in Matplotlib:

Line plots for continuous data.

Scatter plots for visualizing the relationship between two numerical variables.

Histograms for understanding the distribution of data.

Introduction to Seaborn :

Seaborn is a Python data visualization library built on top of Matplotlib. It is designed to provide a high-level interface for drawing attractive and informative statistical graphics. Seaborn works seamlessly with pandas data structures and simplifies the process of creating complex visualizations with just a few lines of code.

Basic Syntax: You can start using Seaborn by importing it as:

import seaborn as sns

Plot Types in Seaborn:

Pair plots to show pairwise relationships in a dataset.

Box plots and **violin plots** for visualizing the distribution of data.

Heatmaps for visualizing the correlation between variables.

Dataset:

The **Iris dataset** consists of 150 observations from 3 species of iris flowers (setosa, versicolor, and virginica). Each observation contains the following features:

sepal length (cm)

sepal width (cm)

petal length (cm)

petal width (cm)

Load the dataset:

Importing Libraries:

We use seaborn for data visualization and matplotlib for creating plots.

Loading the Iris Dataset:

sns.load_dataset('iris') loads a dataset about iris flowers, showing their measurements and species.

Printing the Data:

print(iris) displays the flower data in a table with 150 rows.

In [5]:
```
import seaborn as sns
import matplotlib.pyplot as plt
# Load the Iris dataset
iris=sns.load_dataset('iris')
print(iris)
```

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

[150 rows x 5 columns]
```

1. General Statistics Plot (Matplotlib or Seaborn):

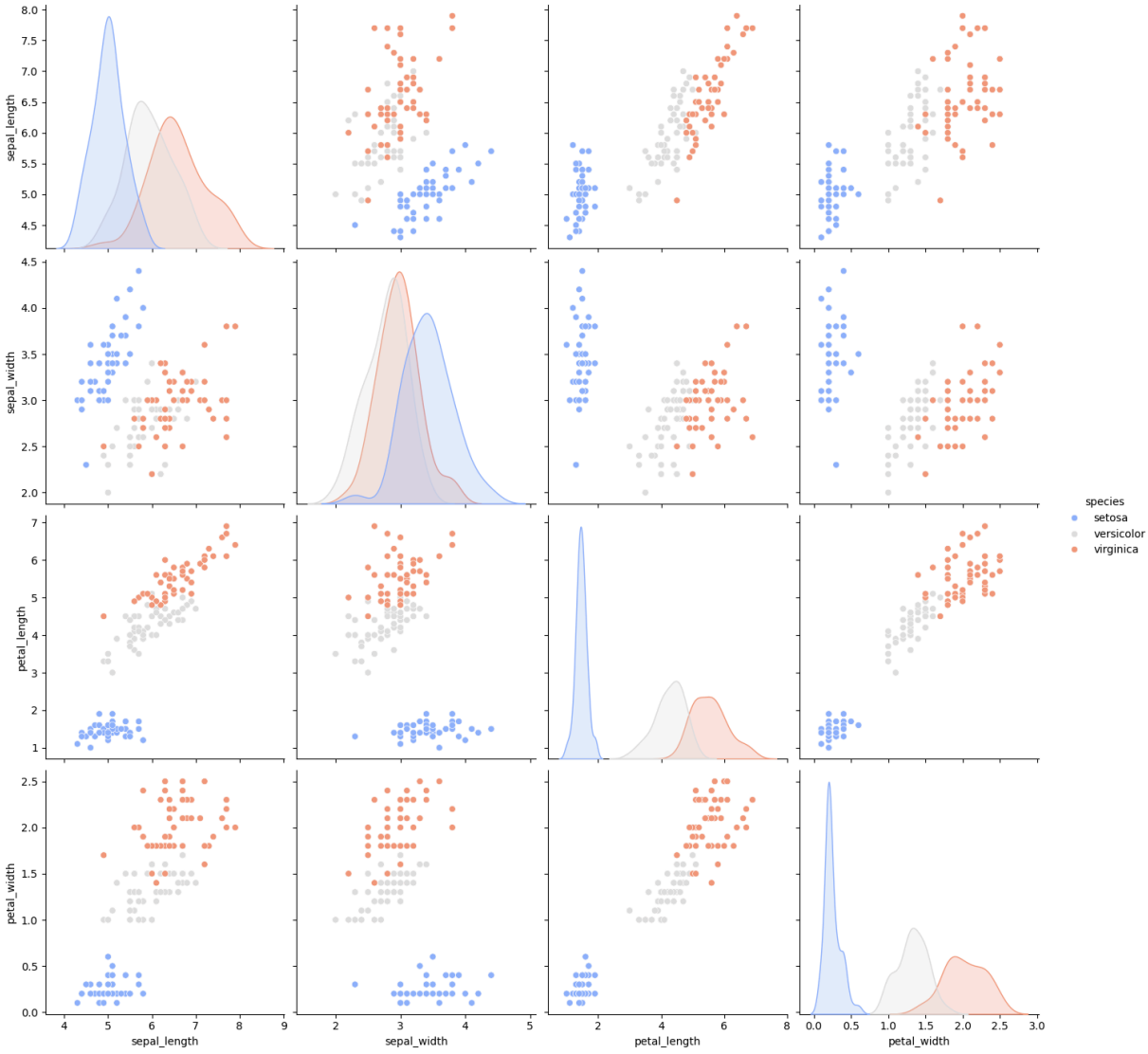
Creating a Pair Plot:

sns.pairplot(iris, hue='species', height=3.5, palette='coolwarm') creates a grid of scatter plots comparing each pair of features (like sepal length vs. sepal width) in the Iris dataset. Each species is shown in a different color using the coolwarm color scheme, and the plot size is set to 3.5.

Displaying the Plot:

plt.show() shows the resulting plot.

In [6]:
```
sns.pairplot(iris, hue='species', height=3.5, palette='coolwarm')
plt.show()
```



2. Pie Plot for Species Frequency: Import Libraries : Use matplotlib and seaborn for plotting.

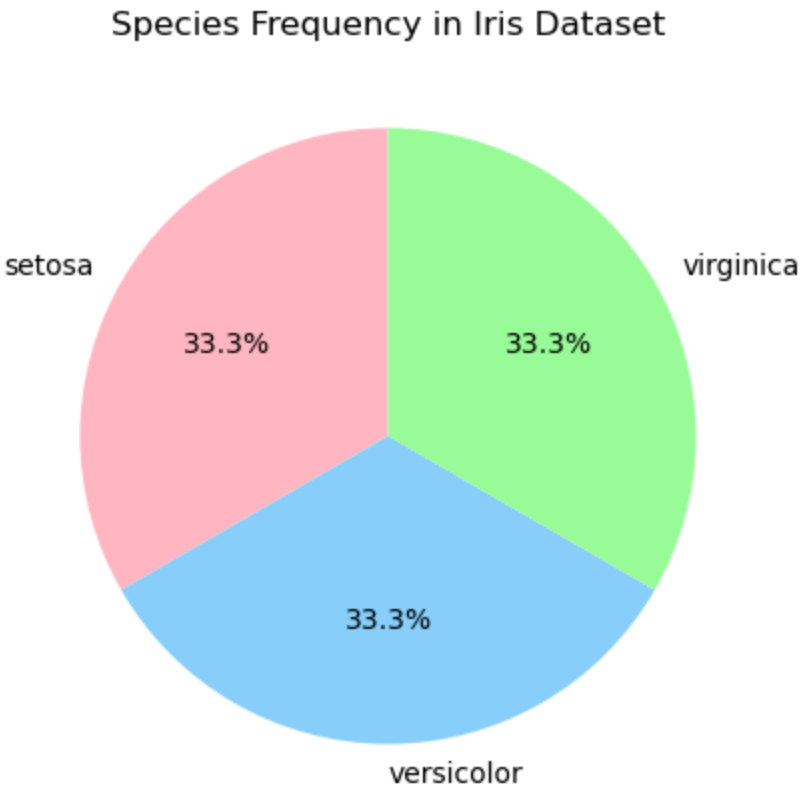
Load Data : Get the Iris dataset with sns.load_dataset('iris').

Count Species : Count how many of each iris species are in the dataset.

Set Colors : Use three colors: Light pink, Light sky blue, Pale green.

Create Pie Chart : Make a pie chart showing species counts with these colors and display it.

In [12]:
```
species_counts = iris['species'].value_counts()
colors = ['#ffb6c1', '#87cefa', '#98fb98']
plt.figure(figsize=(5,5))
plt.pie(species_counts, labels=species_counts.index, autopct='%1.1f%%', star
plt.title('Species Frequency in Iris Dataset')
plt.show()
```



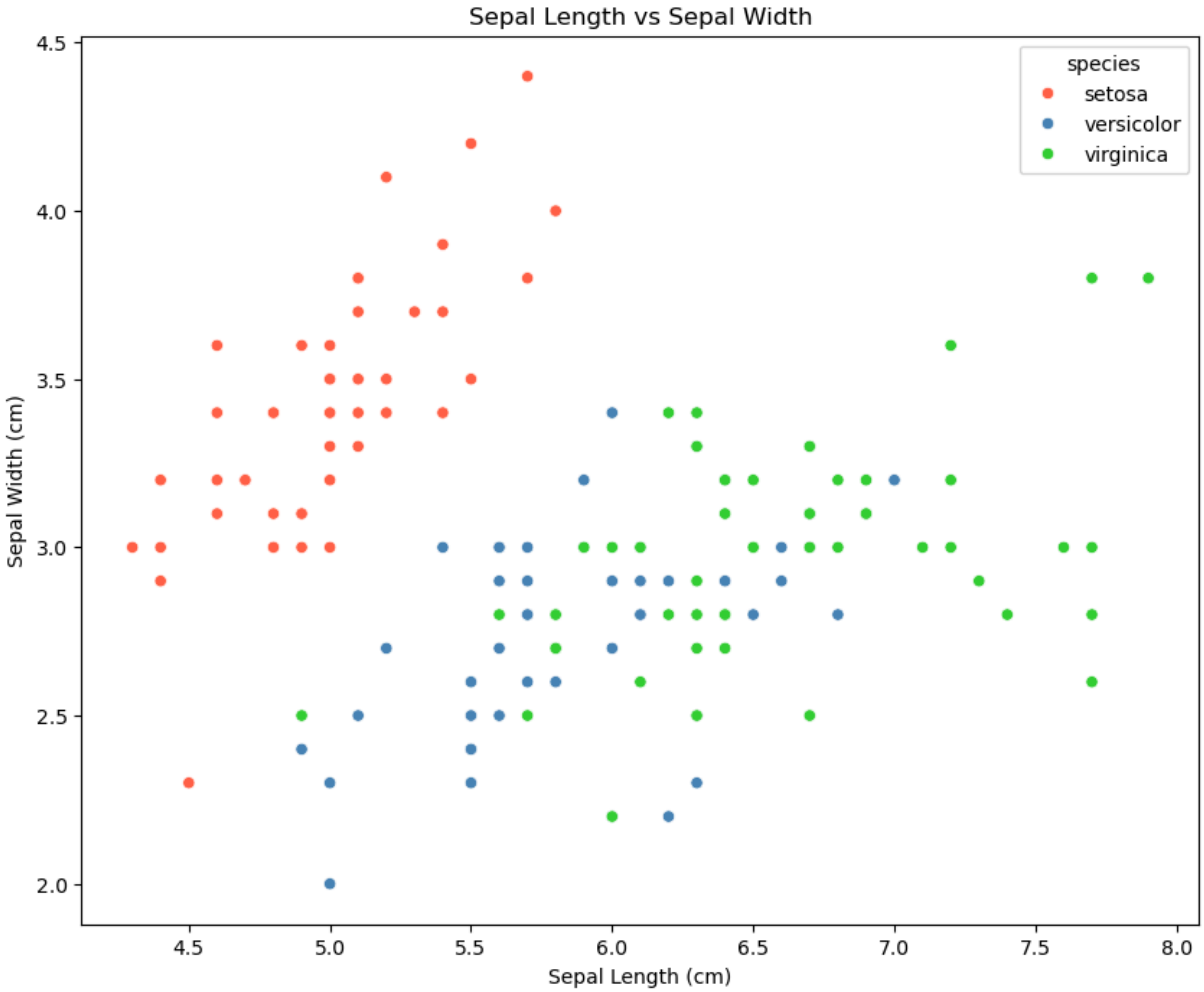
3. Relationship Between Sepal Length and Sepal Width:

Set Up Plot : Create a scatter plot to show sepal_length vs. sepal_width with different colors for each iris species.

Custom Colors : Use three colors: tomato red, steel blue, and lime green.

Show Plot : Display the plot with titles and labels.

In [13]:
```
plt.figure(figsize=(10, 8))
sns.scatterplot(x='sepal_length', y='sepal_width', hue='species',
                data=iris, palette=['#ff6347', '#4682b4', '#32cd32'])
plt.title('Sepal Length vs Sepal Width')
plt.xlabel('Sepal Length (cm)')
plt.ylabel('Sepal Width (cm)')
plt.show()
```



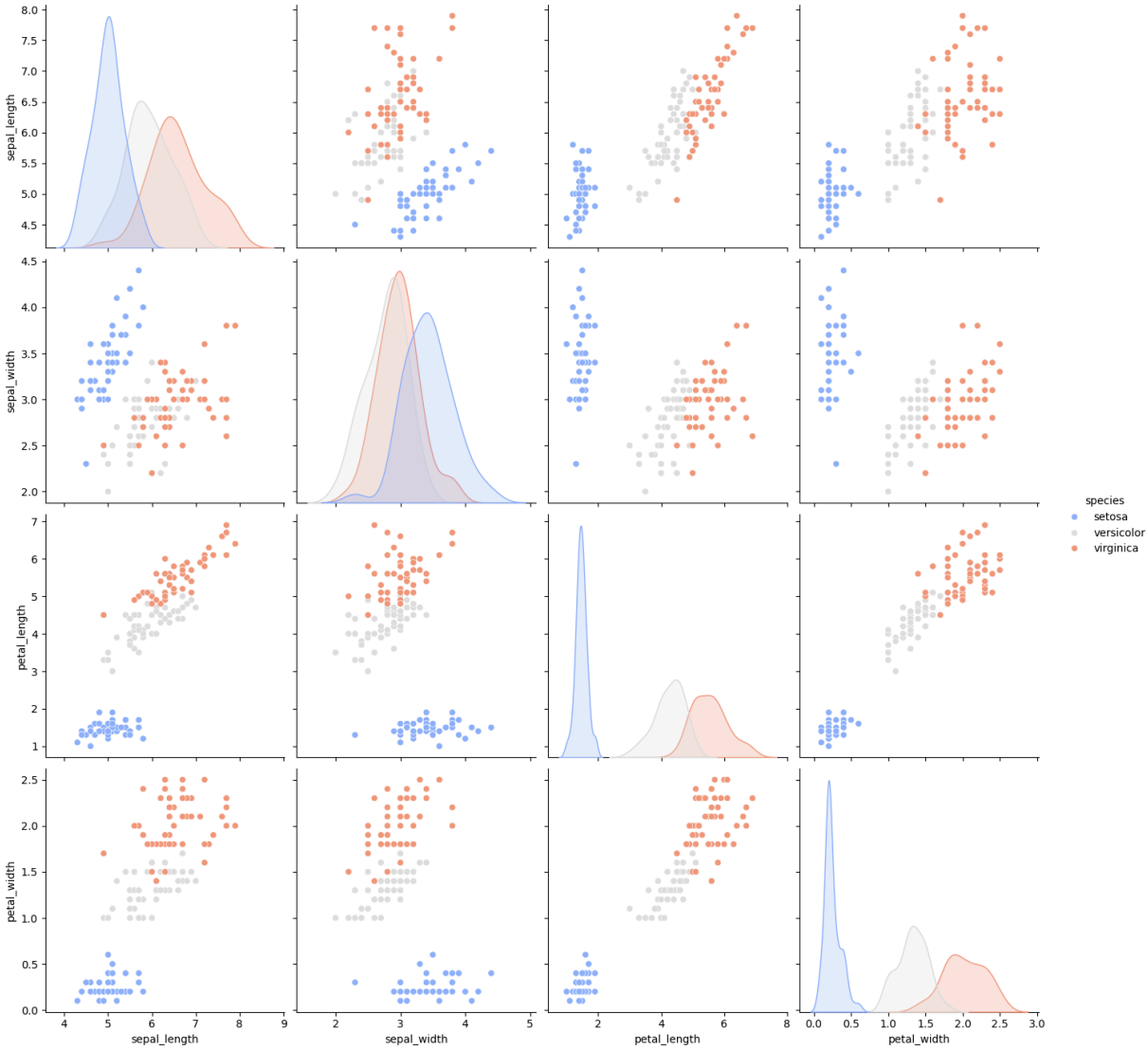
4. Distribution of Sepal and Petal Features:

Create Pair Plot : Make a grid of scatter plots for different flower measurements, with each iris species shown in different colors.

Custom Colors : Use the coolwarm color scheme for the plots.

Show Plot : Display the grid of plots.

In [16]:
```
sns.pairplot(iris, hue='species', height=3.5, palette='coolwarm')
plt.show()
```



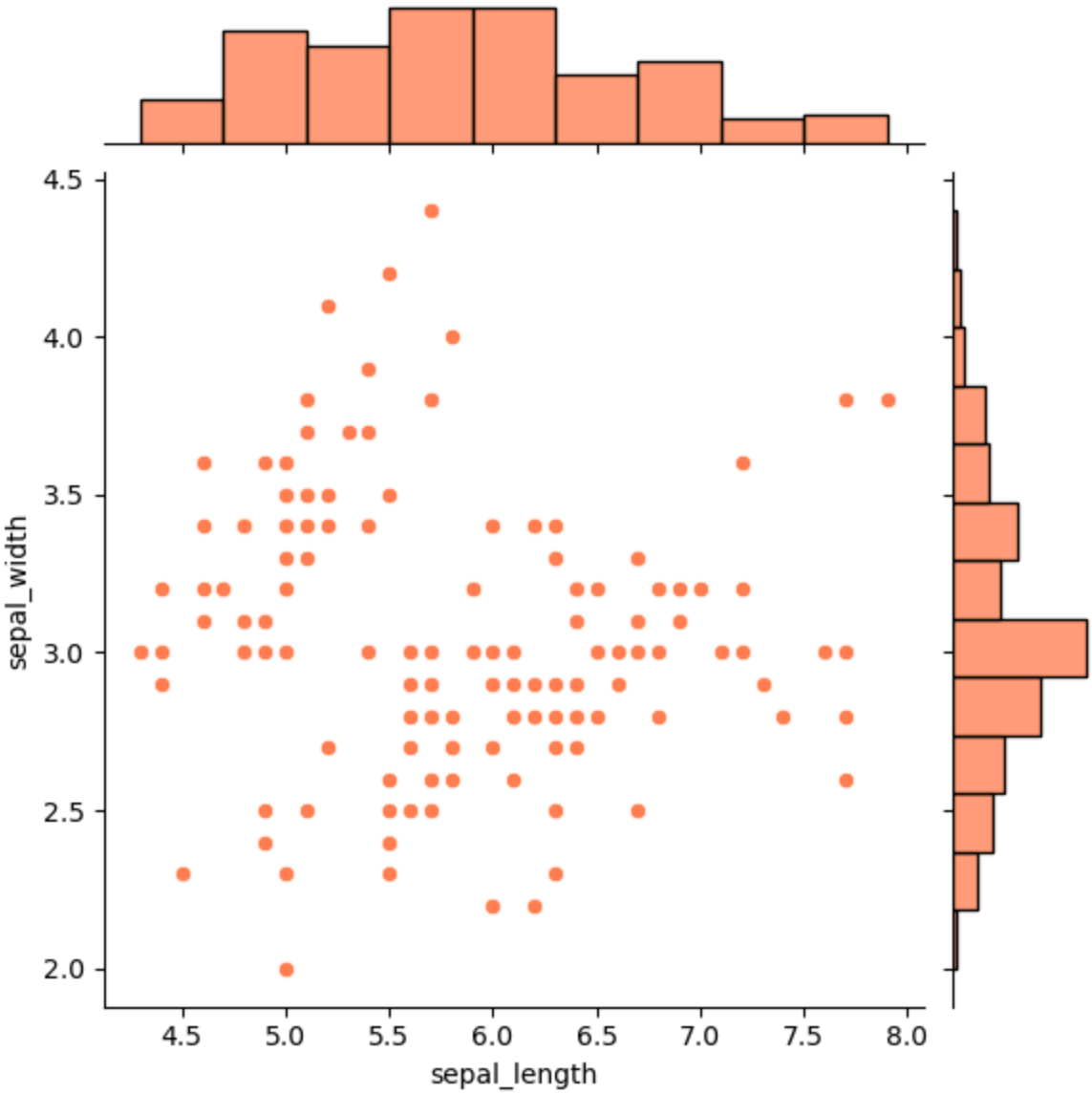
5. Jointplot of Sepal Length vs Sepal Width

Create Joint Plot : Make a scatter plot of sepal_length vs. sepal_width from the Iris dataset.

Custom Color : Use coral color for the points.

Show Plot : Display the scatter plot.

In [26]:
```
sns.jointplot(x='sepal_length', y='sepal_width', data=iris,kind='scatter', c
plt.show()
```



6. KDE Plot for Setosa Species (Sepal Length vs Sepal Width):

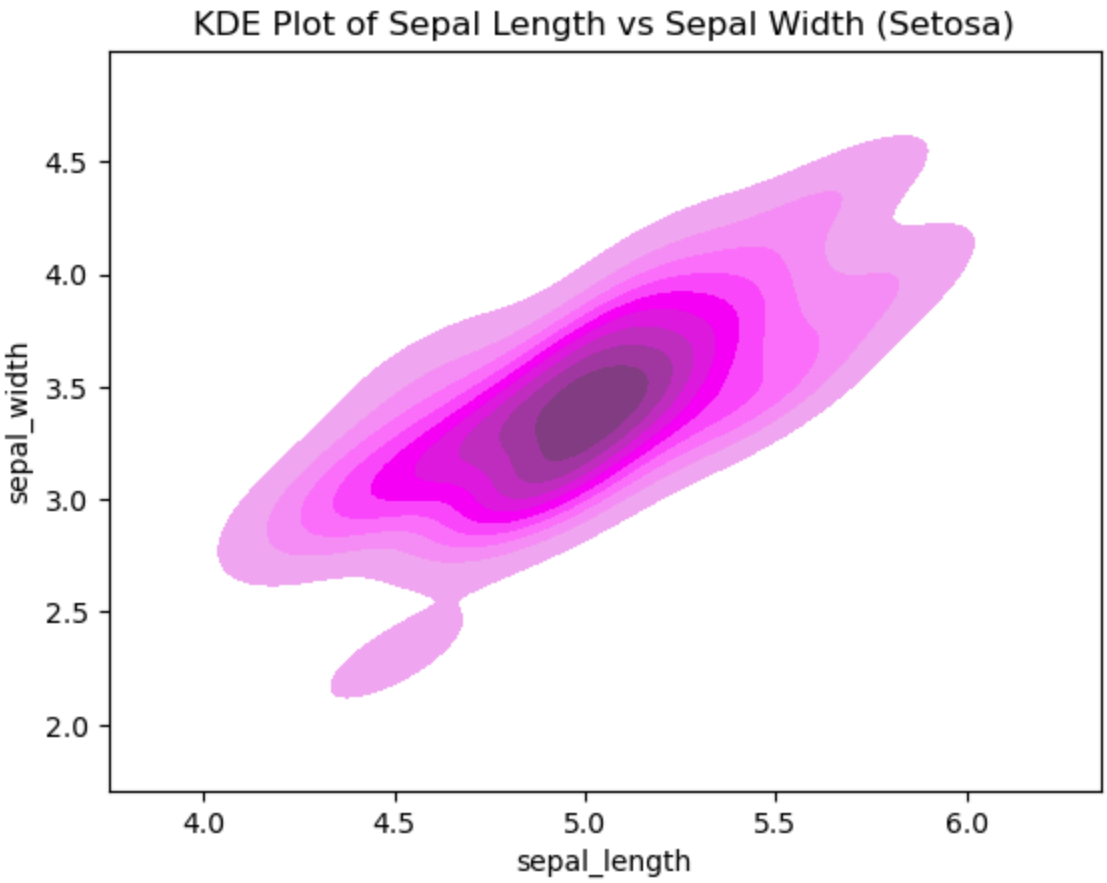
Filter the Dataset : Extract data for the Setosa species from the Iris dataset, focusing on its sepal length and width.

KDE Plot : Generate a Kernel Density Estimate (KDE) plot to visualize the density distribution of sepal length vs. sepal width for Setosa flowers. The plot is shaded (filled) in purple.

Title : The plot is titled "KDE Plot of Sepal Length vs Sepal Width (Setosa)".

Display : The plot is displayed to show the distribution of sepal measurements for Setosa flowers.

In [27]:
```
setosa = iris[iris['species'] == 'setosa']
sns.kdeplot(x='sepal_length', y='sepal_width', data=setosa,fill=True,color='
plt.title('KDE Plot of Sepal Length vs Sepal Width (Setosa)')
plt.show()
```



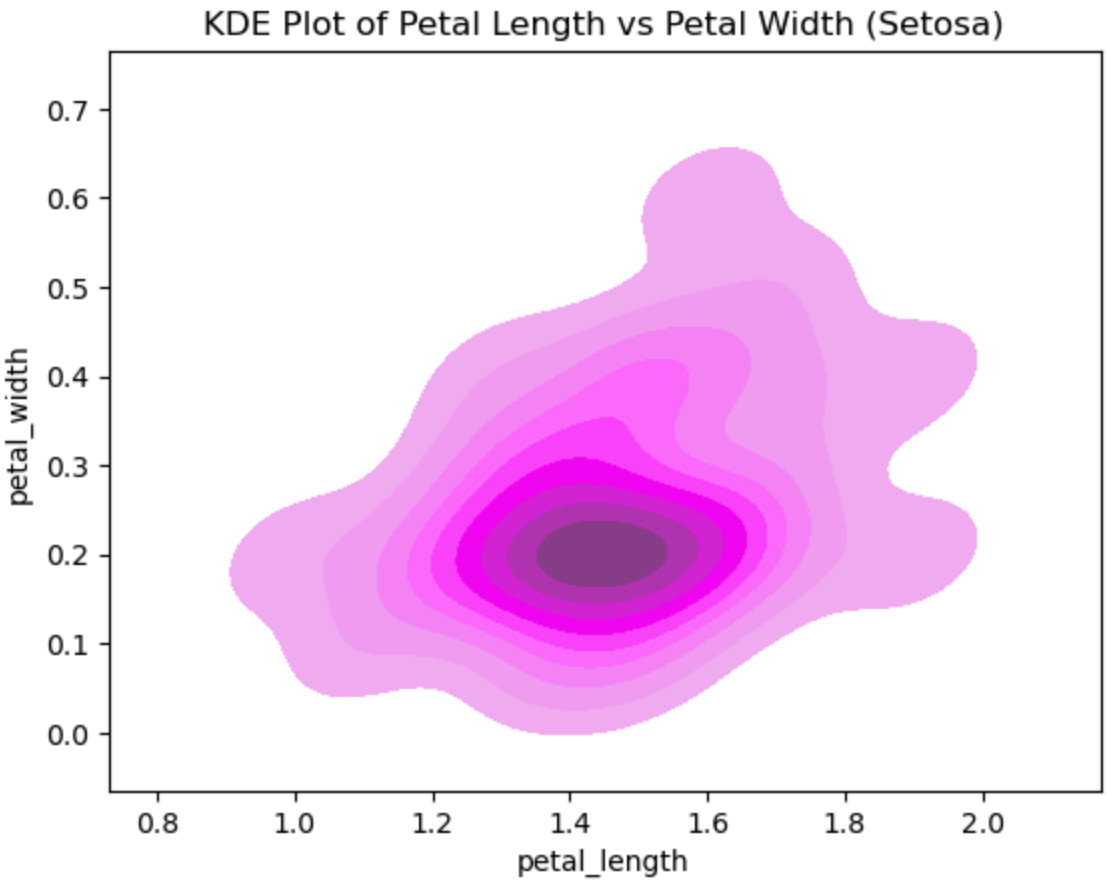
7. KDE Plot for Setosa Species (Petal Length vs Petal Width):

KDE Plot : Create a Kernel Density Estimate (KDE) plot to show the density distribution of petal length vs. petal width for Setosa flowers. The plot is filled with the color purple.

Title : The plot is titled "KDE Plot of Petal Length vs Petal Width (Setosa)".

Display : The plot is displayed to visualize the density of petal measurements for Setosa flowers.

In [28]:
```
sns.kdeplot(x='petal_length', y='petal_width', data=setosa,fill=True, color=
plt.title('KDE Plot of Petal Length vs Petal Width (Setosa)')
plt.show()
```



Conclusion:

Distinct Patterns: The visualizations reveal distinct patterns in flower measurements across different iris species. Setosa, in particular, shows a unique density distribution in petal measurements.

Visualization Benefits: Using scatter plots, pair plots, pie charts, and KDE plots provides a comprehensive view of the dataset, helping to understand relationships and distributions.

Data Insights: These visualizations are valuable for identifying how measurements vary by species and for interpreting the overall distribution of iris flower features.

In [ ]: