TASK:1

Exploratory Data Analysis (EDA) on a IRIS DATASET

In [4]:
```
# Import necessary libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [40]:
```
# Load the Iris dataset
df = sns.load_dataset('iris')
```

In [5]:
```
# Display the first few rows of the dataset
print(df.head())
```

```
   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
```

In [6]:
```
# Step 1: Data Cleaning
# Check for missing values
print(df.isnull().sum())
```

```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

In [7]:
```
# Since there are no missing values, no handling is required here.

# Step 2: Basic Statistical Analysis
# Summary statistics for numerical columns
print(df.describe())
```

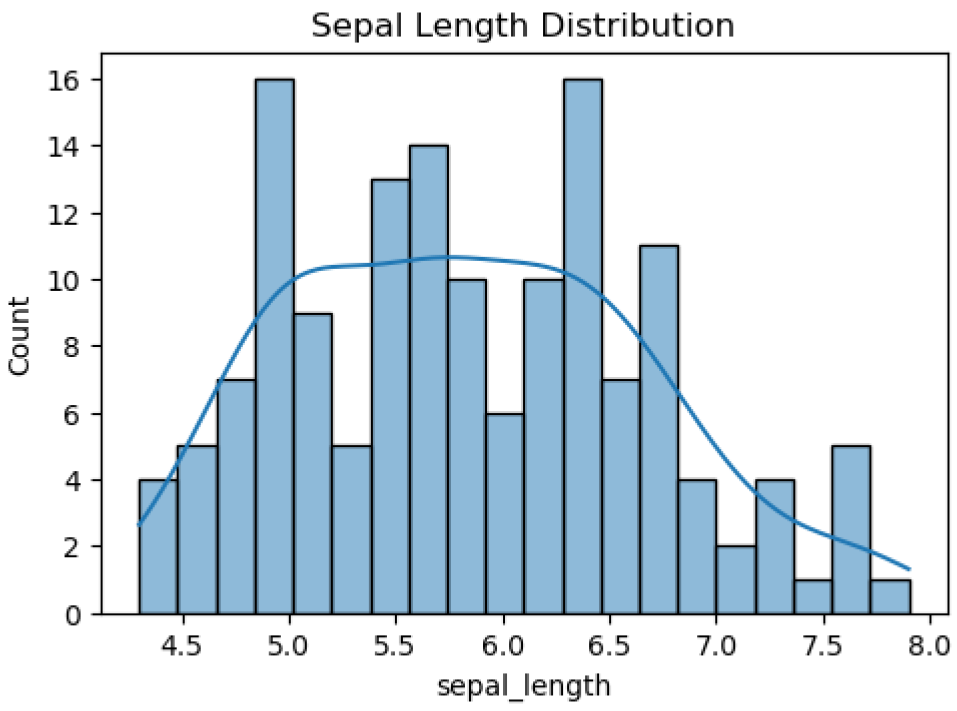
```
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.057333      3.758000     1.199333
std        0.828066     0.435866      1.765298     0.762238
min        4.300000     2.000000      1.000000     0.100000
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
```

```
In [8]: # Distribution of the target variable
        print(df['species'].value_counts())
```

```
setosa        50
versicolor    50
virginica     50
Name: species, dtype: int64
```

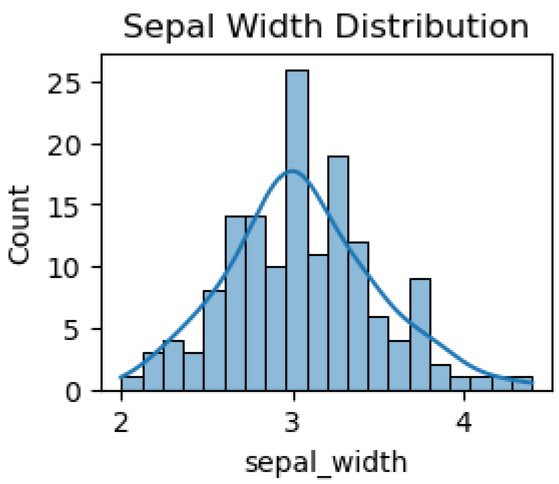
```
In [9]: # Step 3: Data Visualization

        # Histograms for each feature
        plt.figure(figsize=(12, 8))
        plt.subplot(2, 2, 1)
        sns.histplot(df['sepal_length'], bins=20, kde=True)
        plt.title('Sepal Length Distribution')
```

```
Out[9]: Text(0.5, 1.0, 'Sepal Length Distribution')
```



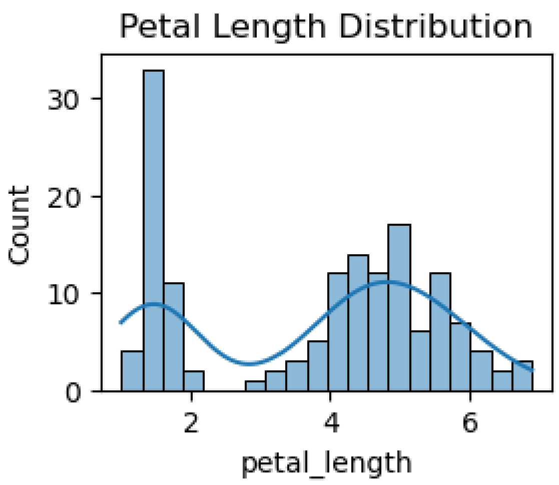
```
In [10]: plt.subplot(2, 2, 2)
         sns.histplot(df['sepal_width'], bins=20, kde=True)
         plt.title('Sepal Width Distribution')
```

```
Out[10]: Text(0.5, 1.0, 'Sepal Width Distribution')
```

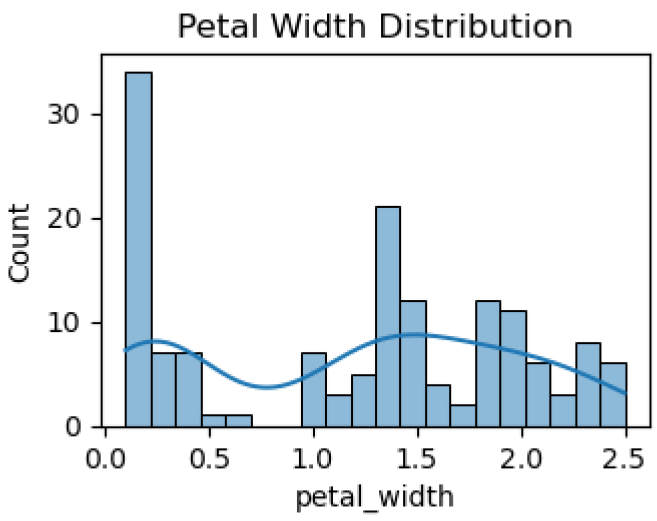


```
In [11]: plt.subplot(2, 2, 3)
         sns.histplot(df['petal_length'], bins=20, kde=True)
         plt.title('Petal Length Distribution')
```

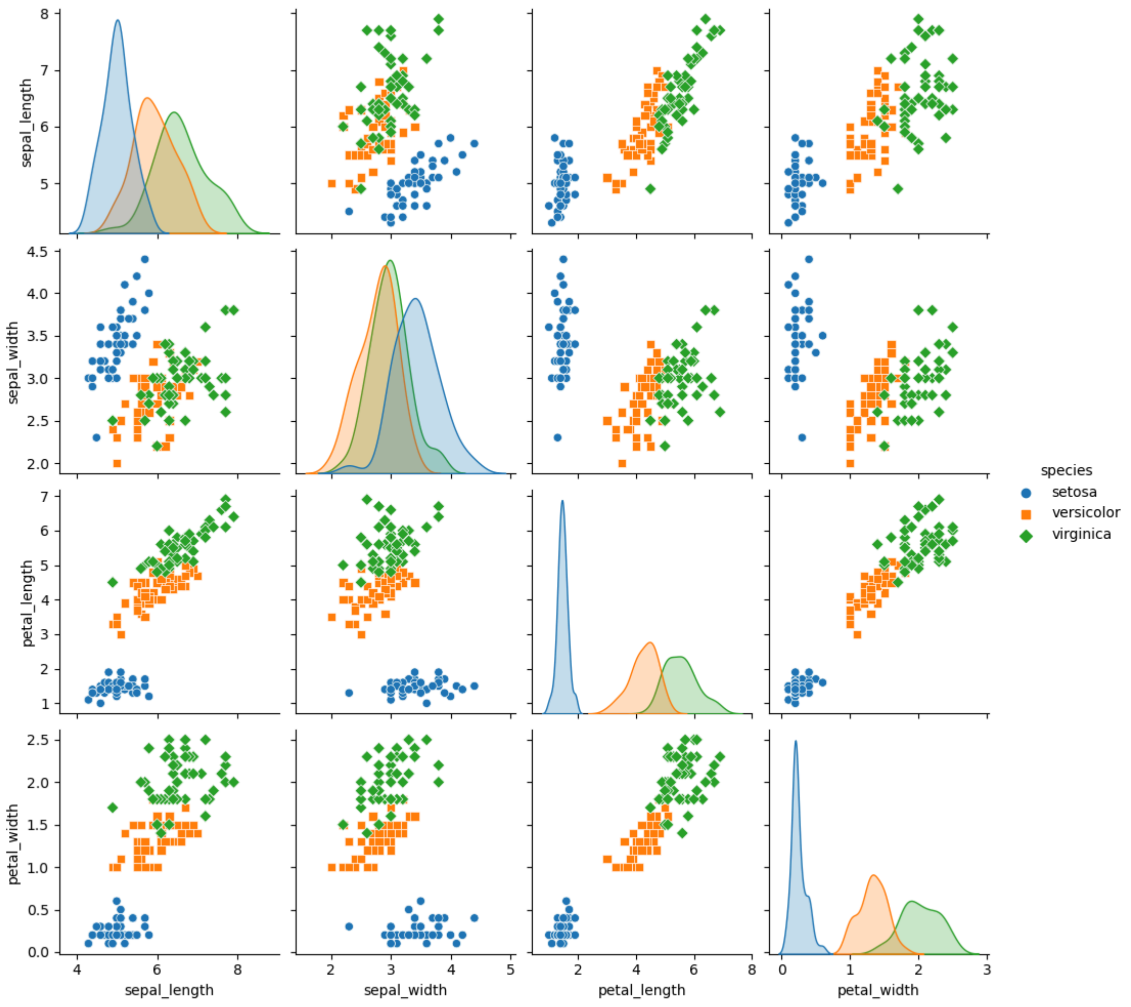
```
Out[11]: Text(0.5, 1.0, 'Petal Length Distribution')
```



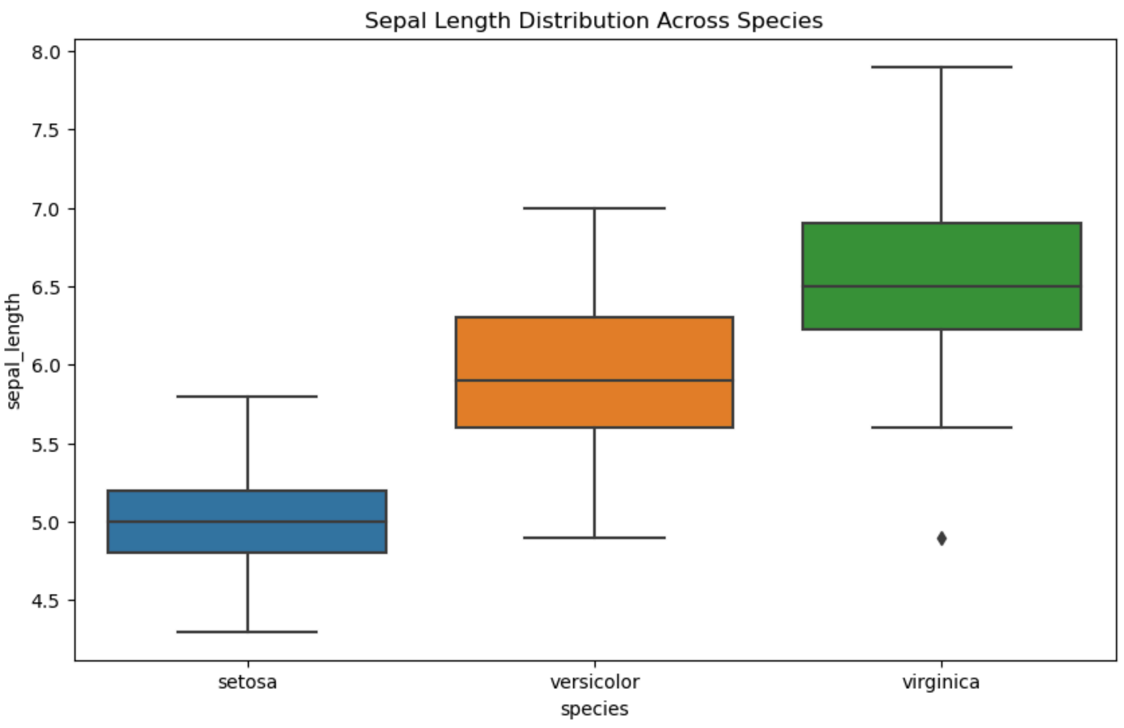
In [12]:
```
plt.subplot(2, 2, 4)
sns.histplot(df['petal_width'], bins=20, kde=True)
plt.title('Petal Width Distribution')

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



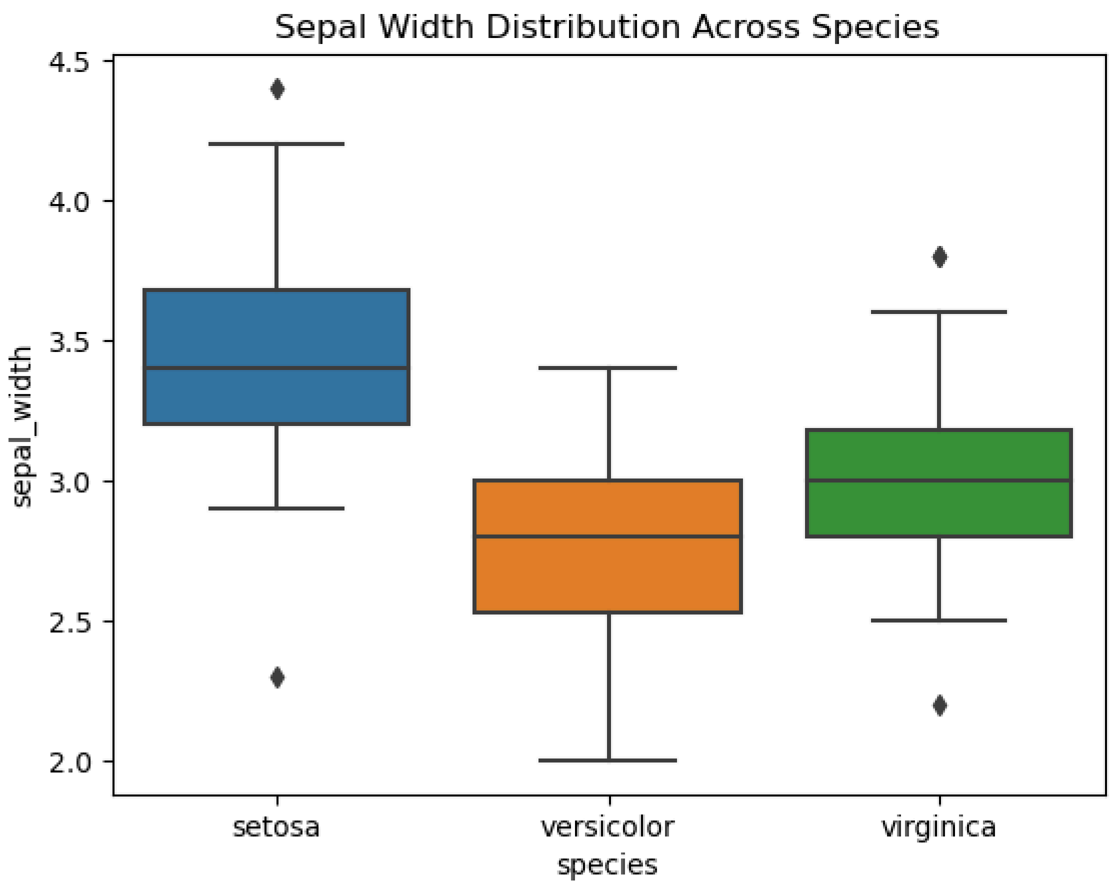
In [13]:
```
# Pairplot to visualize relationships between all features
sns.pairplot(df, hue='species', markers=["o", "s", "D"])
plt.show()
```



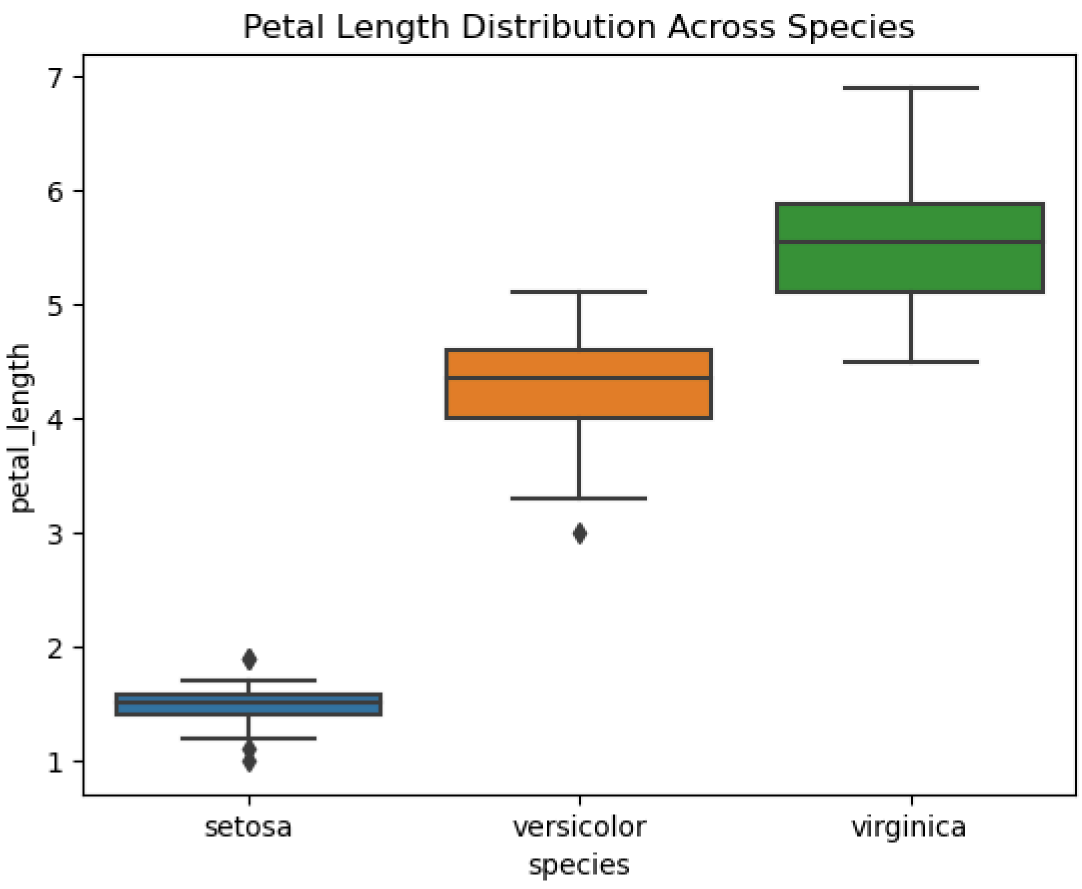
In [14]:
```
# Box Plot for each feature across species
plt.figure(figsize=(10, 6))
sns.boxplot(x='species', y='sepal_length', data=df)
plt.title('Sepal Length Distribution Across Species')
plt.show()
```



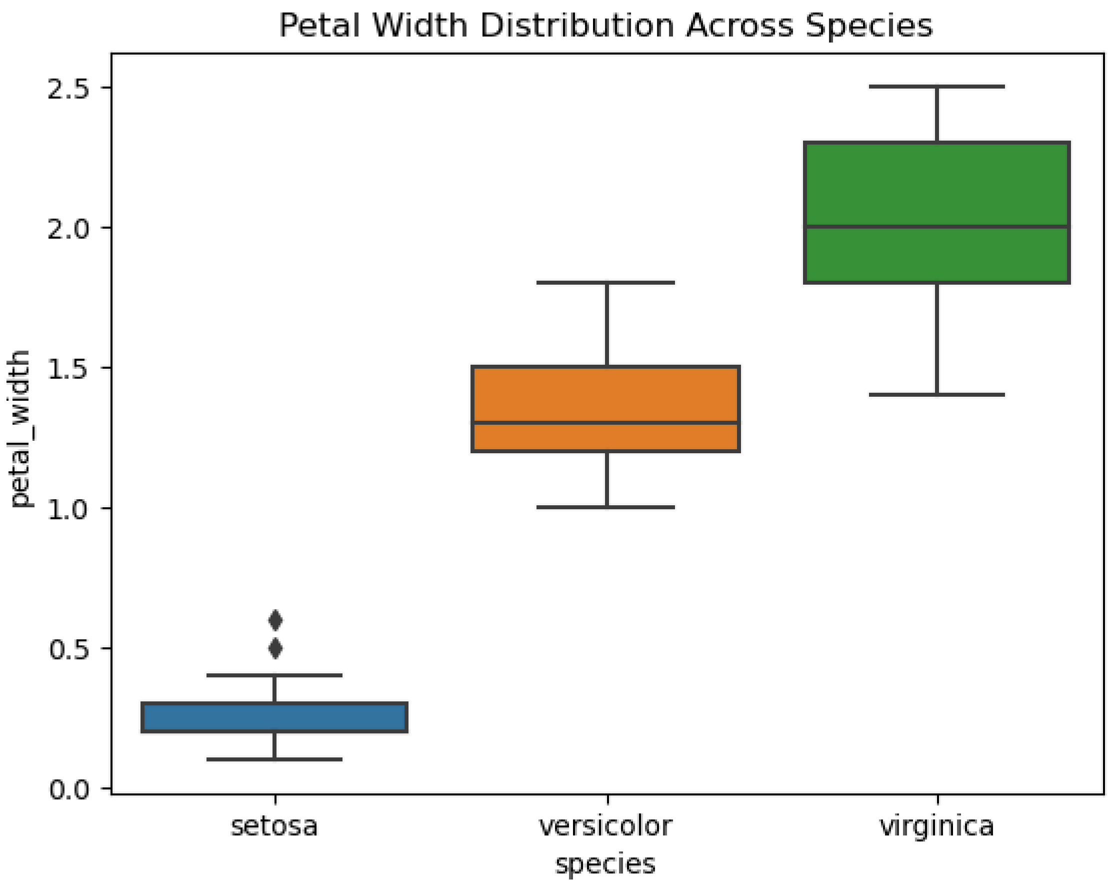
In [15]:
```
sns.boxplot(x='species', y='sepal_width', data=df)
plt.title('Sepal Width Distribution Across Species')
plt.show()
```



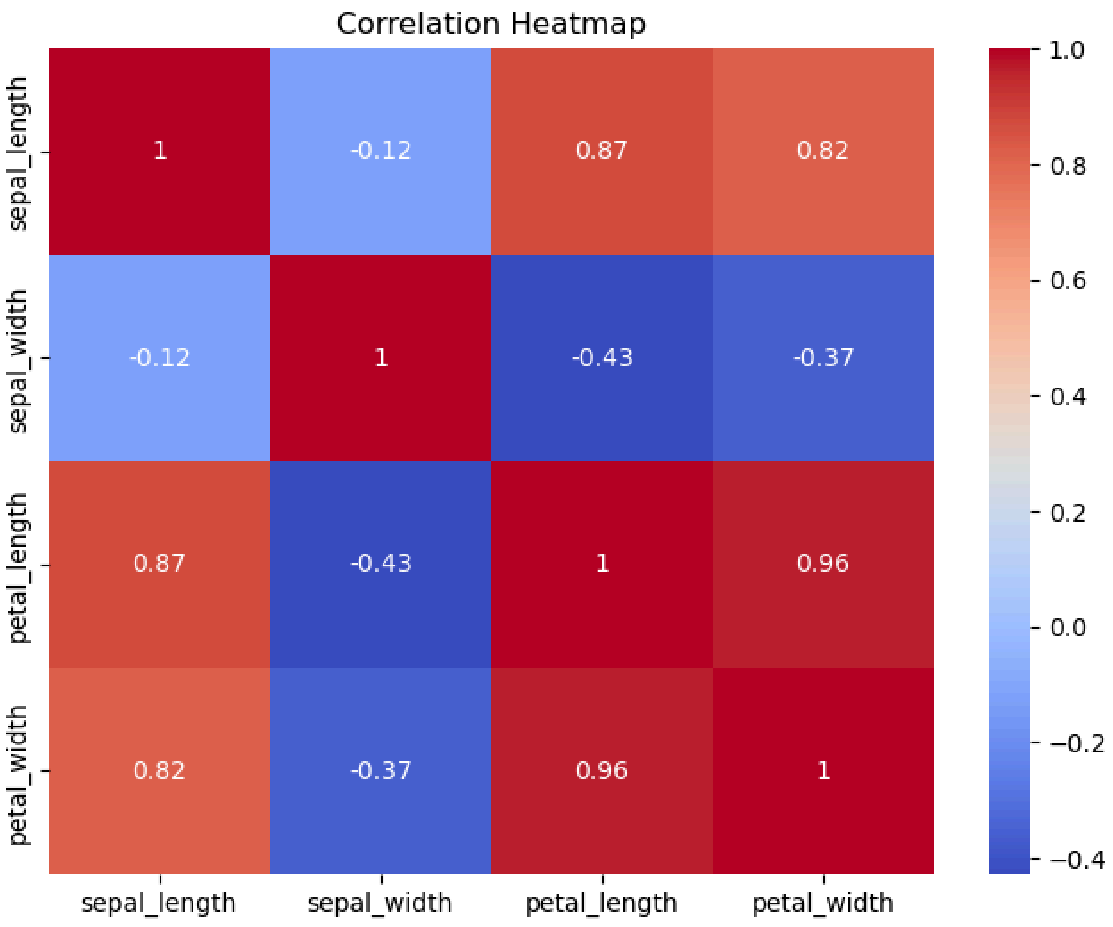
In [16]:
```
sns.boxplot(x='species', y='petal_length', data=df)
plt.title('Petal Length Distribution Across Species')
plt.show()
```



```
In [17]: sns.boxplot(x='species', y='petal_width', data=df)
         plt.title('Petal Width Distribution Across Species')
         plt.show()
```

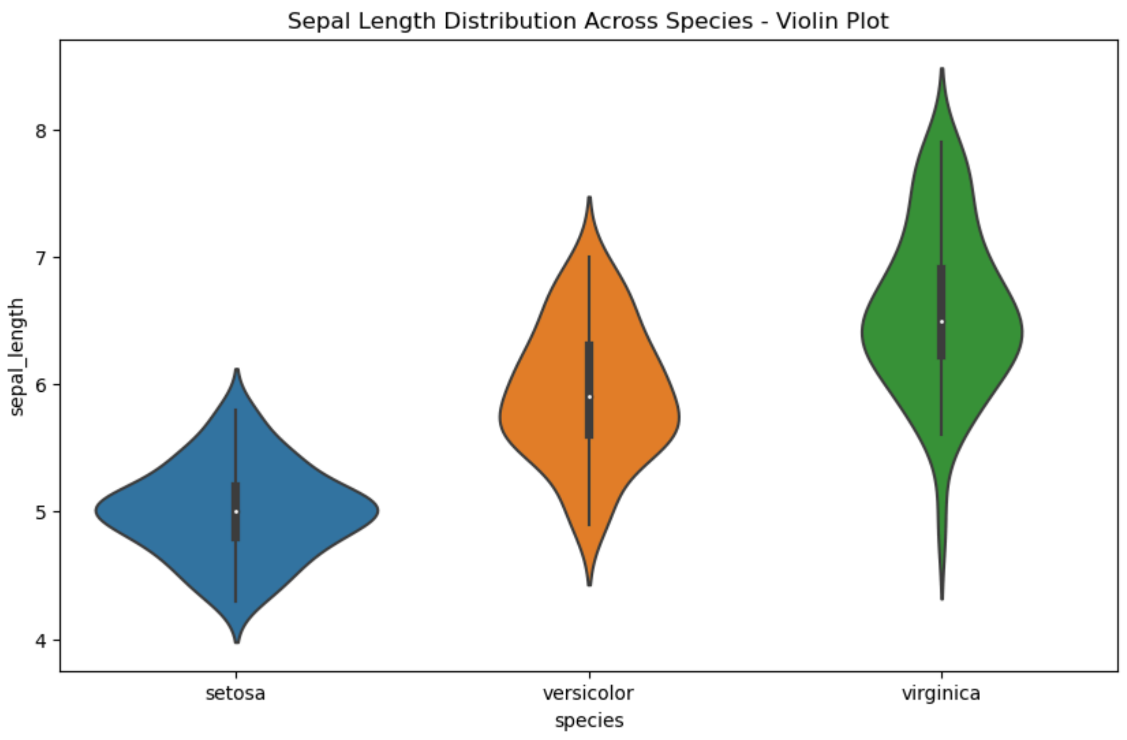


```
In [18]: # Correlation Heatmap
         plt.figure(figsize=(8, 6))
         sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
         plt.title('Correlation Heatmap')
         plt.show()
```

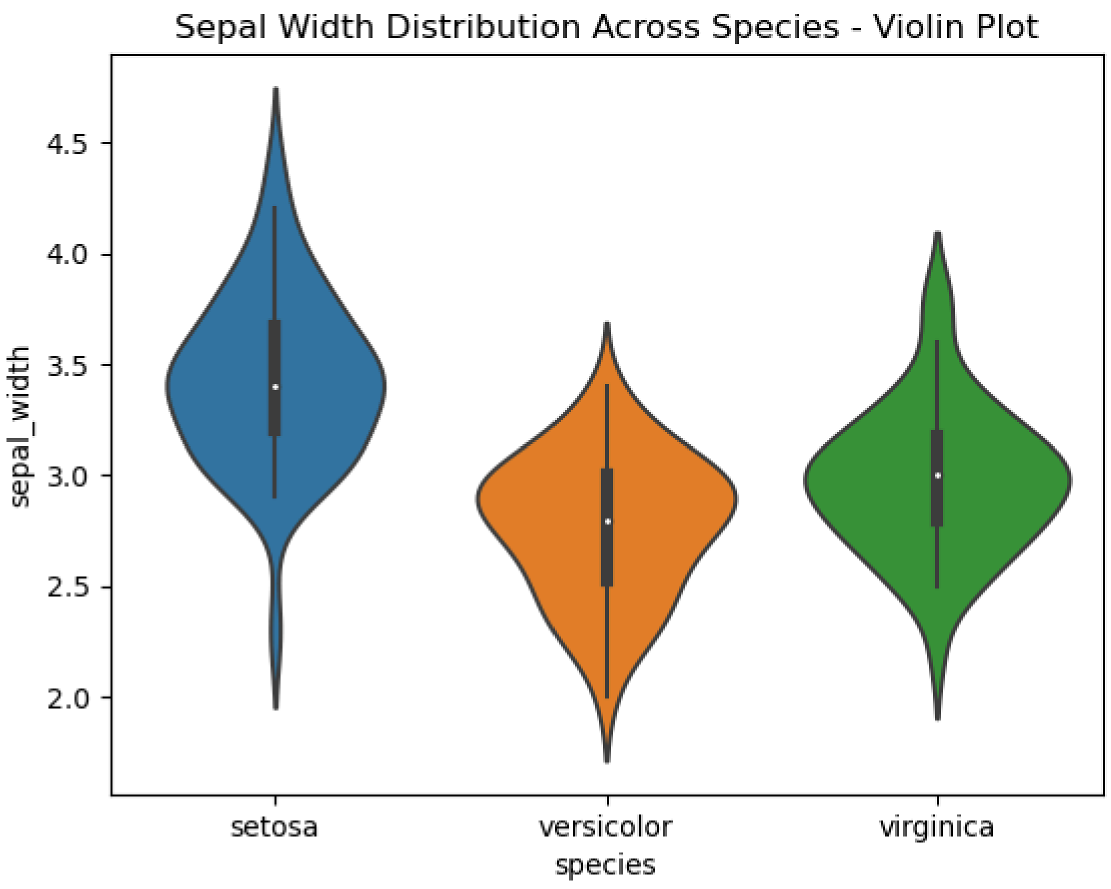


1. Advanced Data Visualizations

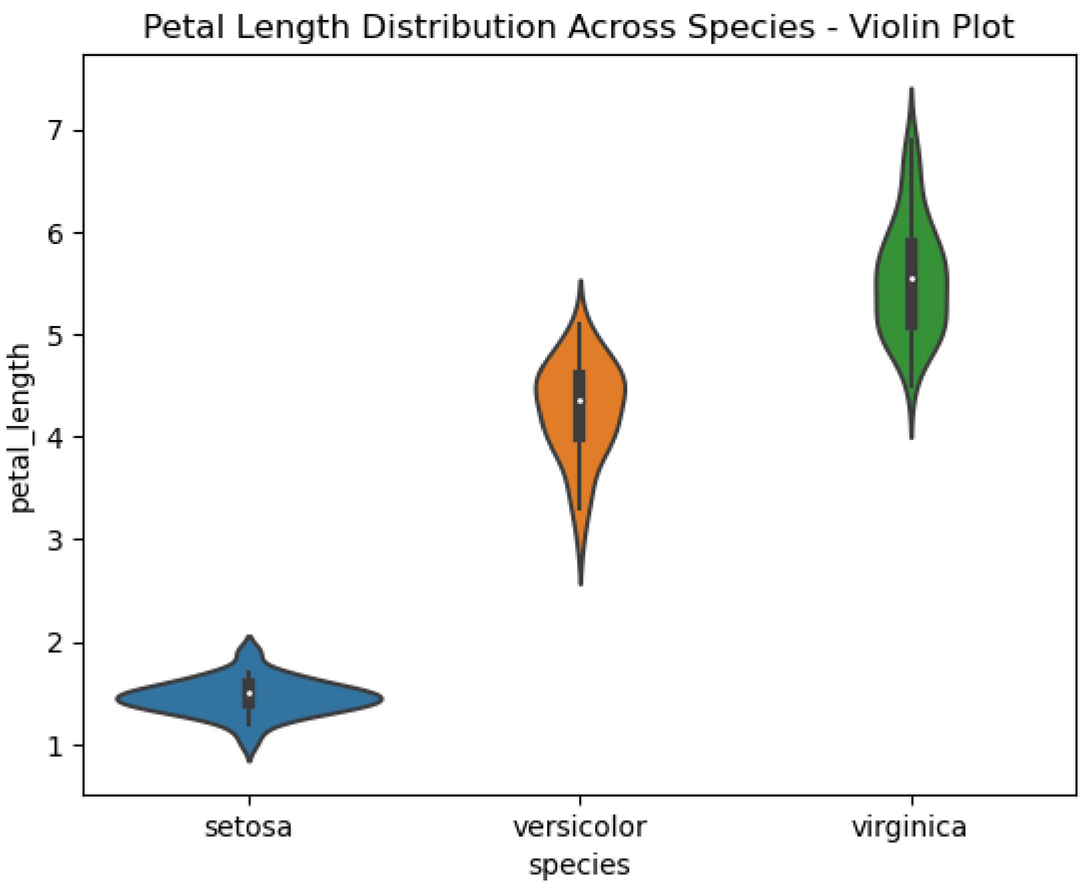
In [19]:
```
# Violin Plots for each feature across species
plt.figure(figsize=(10, 6))
sns.violinplot(x='species', y='sepal_length', data=df)
plt.title('Sepal Length Distribution Across Species - Violin Plot')
plt.show()
```



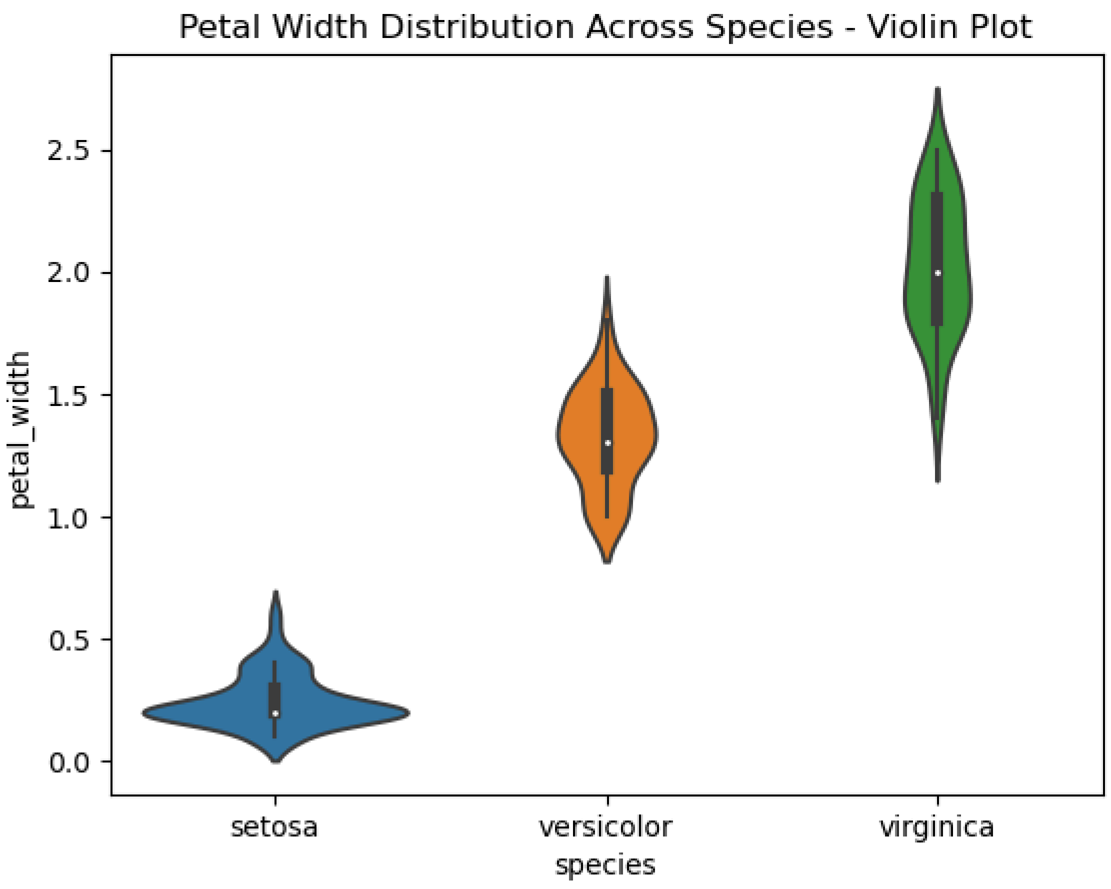
In [20]:
```
sns.violinplot(x='species', y='sepal_width', data=df)
plt.title('Sepal Width Distribution Across Species - Violin Plot')
plt.show()
```



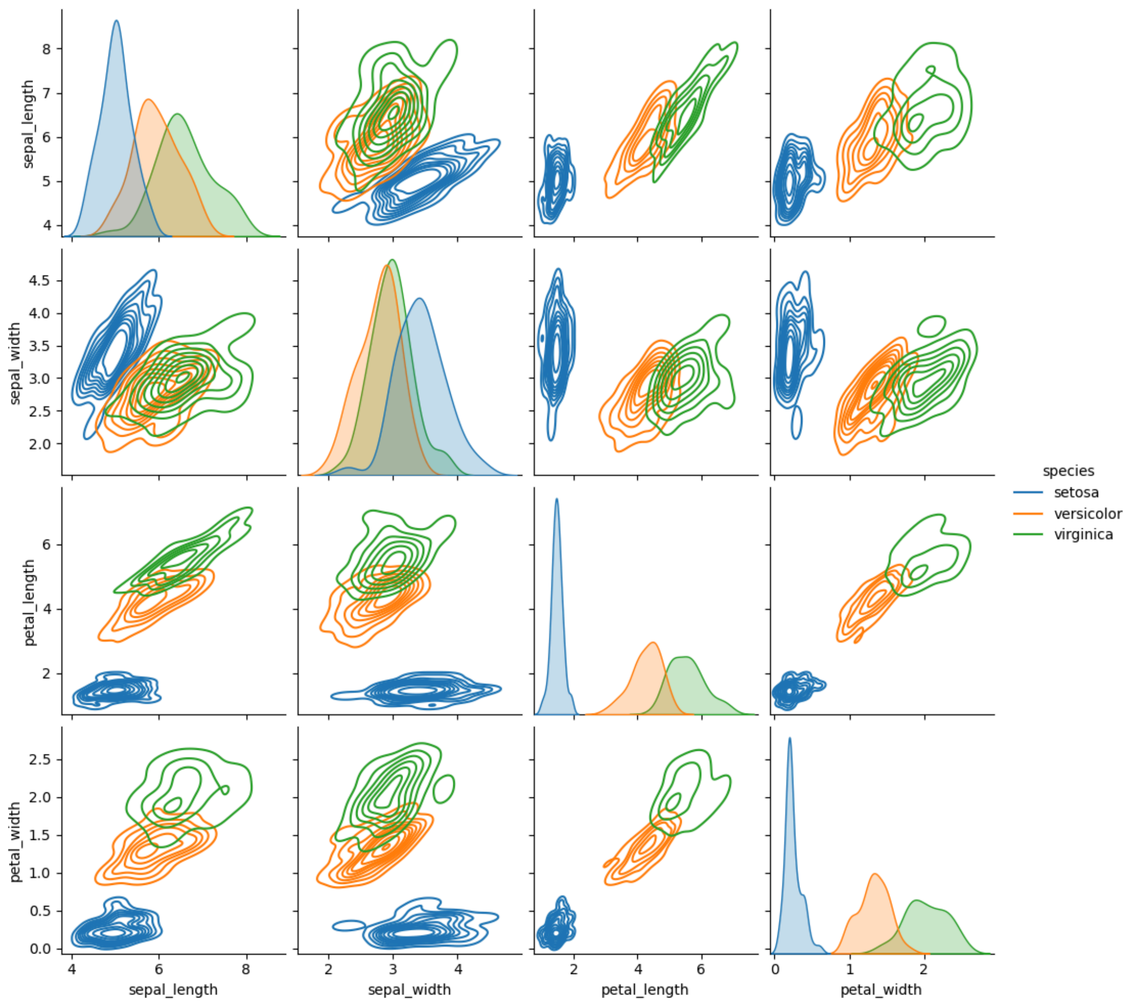
In [21]:
```
sns.violinplot(x='species', y='petal_length', data=df)
plt.title('Petal Length Distribution Across Species - Violin Plot')
plt.show()
```



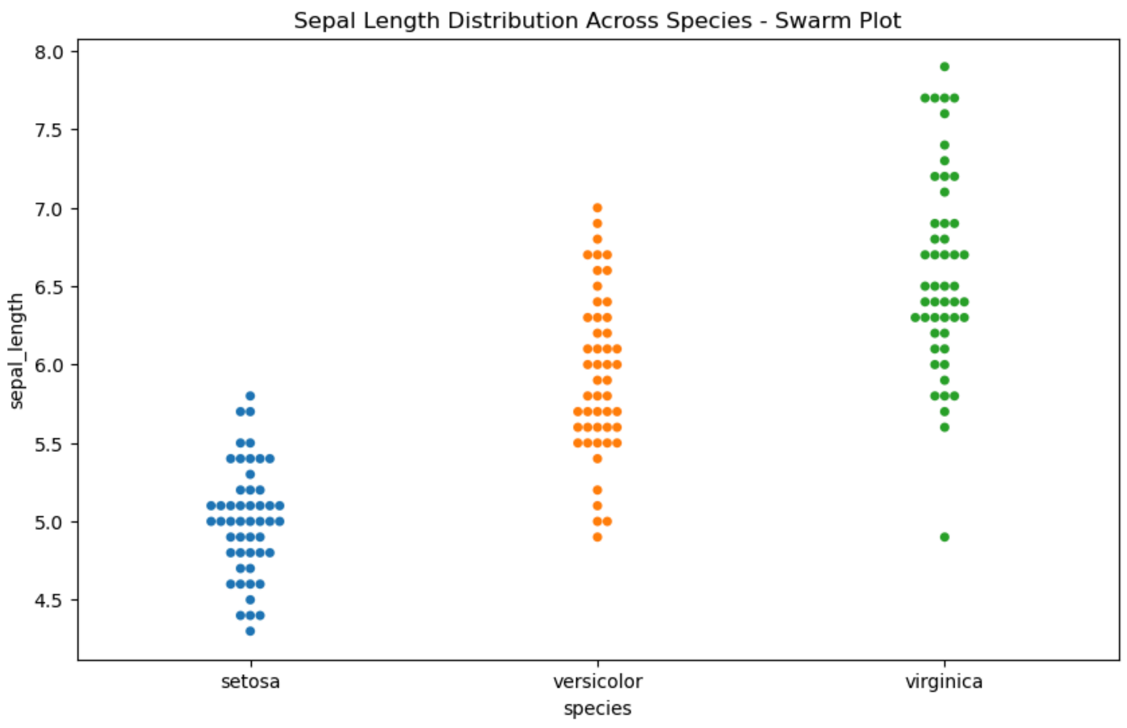
In [22]:
```
sns.violinplot(x='species', y='petal_width', data=df)
plt.title('Petal Width Distribution Across Species - Violin Plot')
plt.show()
```



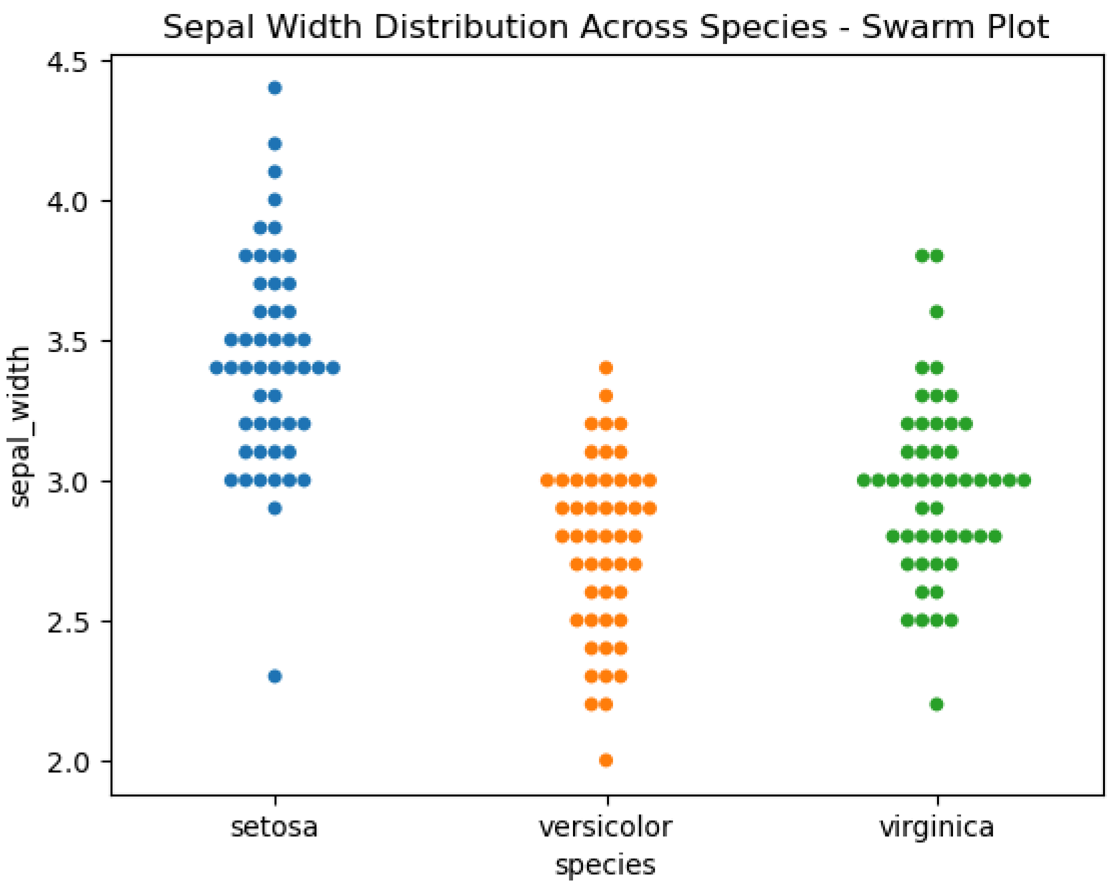
```
In [23]: # Pairwise Kernel Density Estimation (KDE) Plot
         sns.pairplot(df, hue='species', kind='kde', diag_kind='kde')
         plt.show()
```



In [24]:
```
# Swarm Plots for each feature across species
plt.figure(figsize=(10, 6))
sns.swarmplot(x='species', y='sepal_length', data=df)
plt.title('Sepal Length Distribution Across Species - Swarm Plot')
plt.show()
```

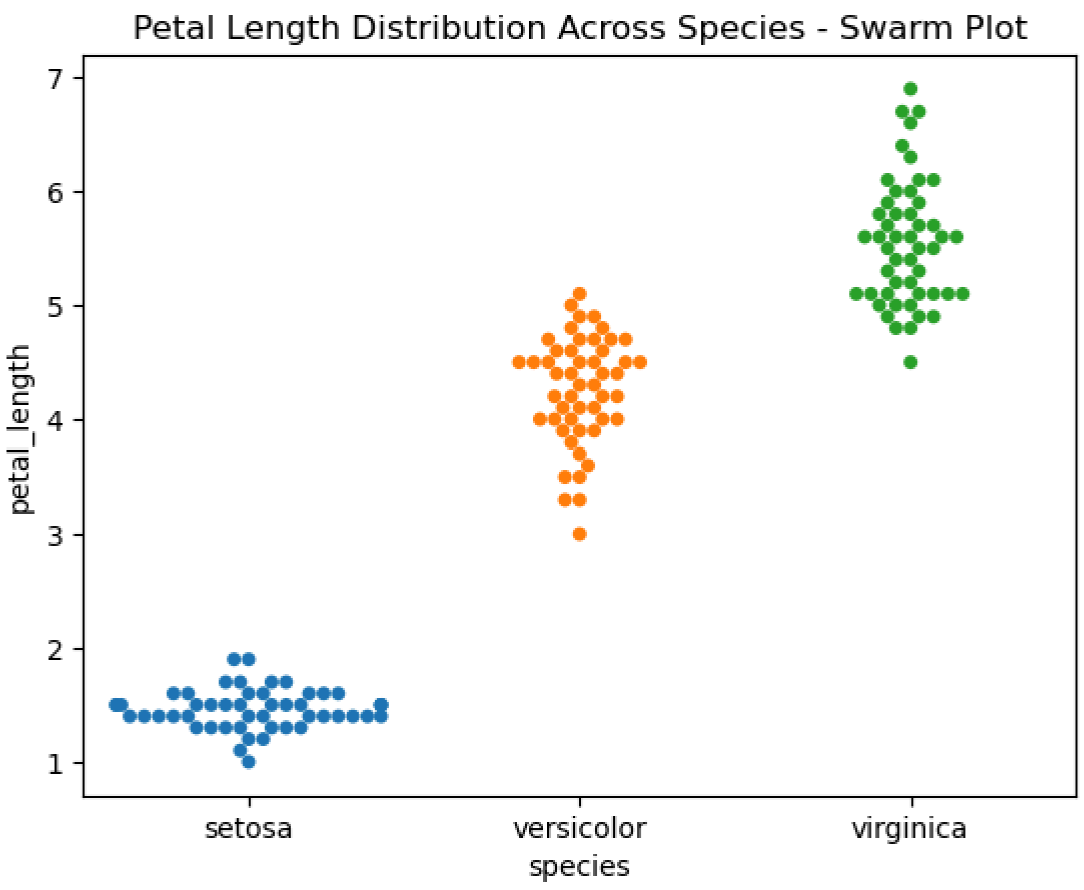


In [25]:
```
sns.swarmplot(x='species', y='sepal_width', data=df)
plt.title('Sepal Width Distribution Across Species - Swarm Plot')
plt.show()
```



In [26]:
```python
sns.swarmplot(x='species', y='petal_length', data=df)
plt.title('Petal Length Distribution Across Species - Swarm Plot')
plt.show()
```

```
C:\Users\chaud\anaconda3\lib\site-packages\seaborn\categorical.py:1296: U
serWarning: 12.0% of the points cannot be placed; you may want to decreas
e the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```



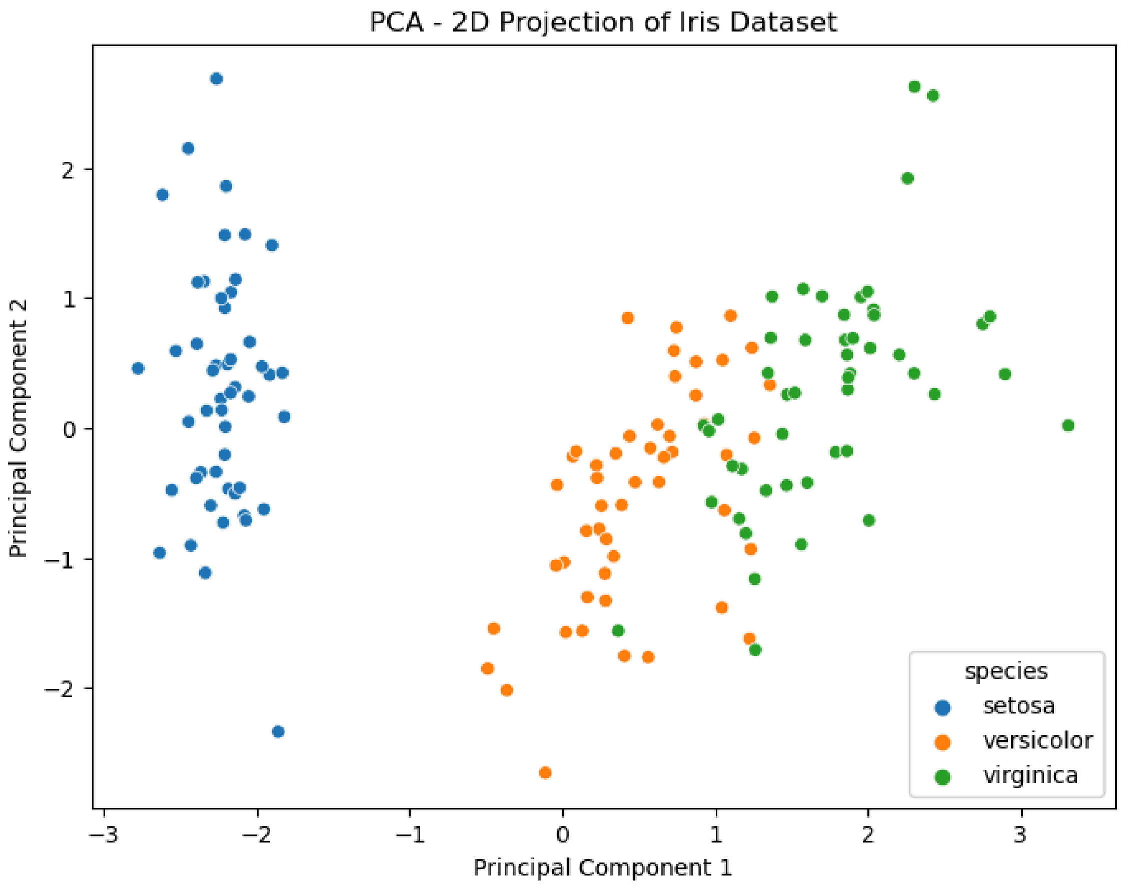
2. Multivariate Analysis with PCA

In [29]:
```python
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
```

In [30]:
```python
# Standardize the data
features = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width']
x = df.loc[:, features].values
y = df.loc[:, ['species']].values
x = StandardScaler().fit_transform(x)
```

In [31]:
```
# Perform PCA
pca = PCA(n_components=2)
principal_components = pca.fit_transform(x)
pca_df = pd.DataFrame(data=principal_components, columns=['Principal Compor
pca_df = pd.concat([pca_df, df[['species']]], axis=1)
```

In [32]:
```
# 2D PCA Plot
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Principal Component 1', y='Principal Component 2', hue='
plt.title('PCA - 2D Projection of Iris Dataset')
plt.show()
```



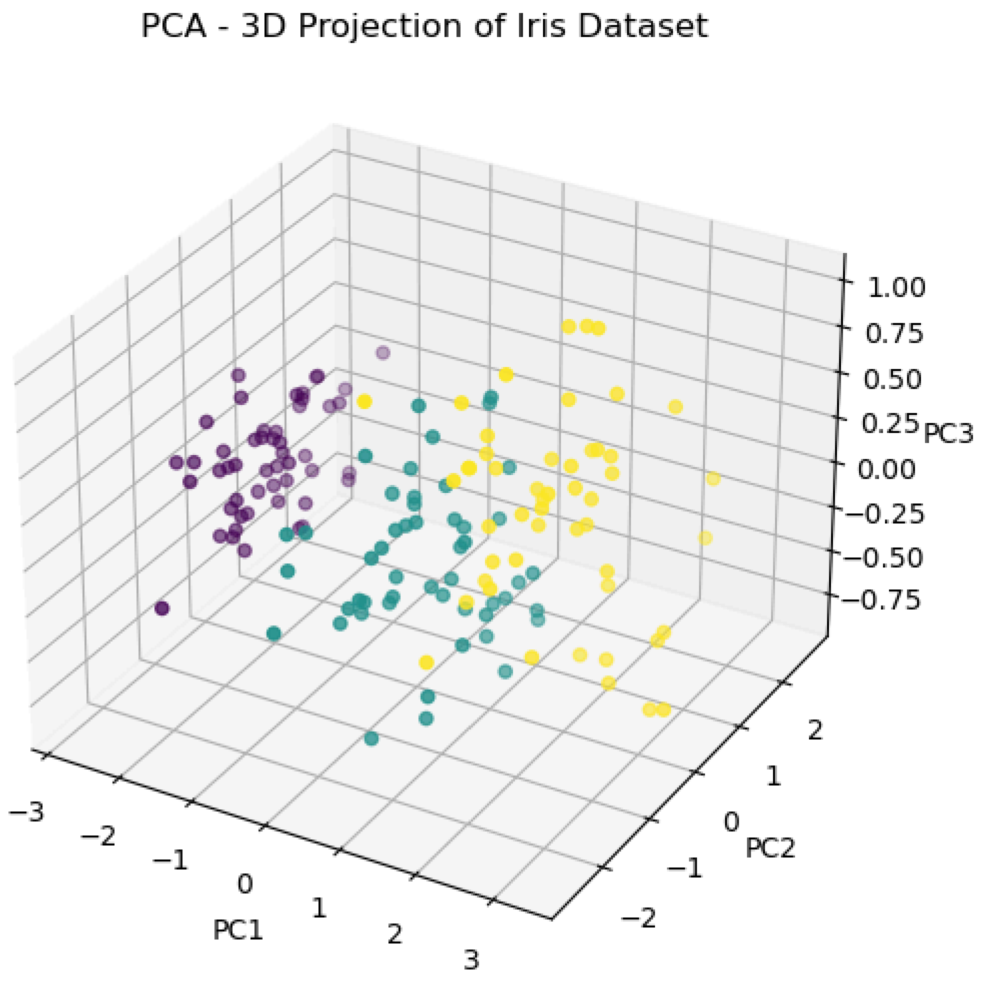
In [33]:
```
# Explained Variance Ratio
print("Explained Variance Ratio by each Principal Component:", pca.explaine
```

```
Explained Variance Ratio by each Principal Component: [0.72962445 0.22850
762]
```

In [34]:
```
# 3D PCA Plot
from mpl_toolkits.mplot3d import Axes3D
```

In [35]:
```
pca_3d = PCA(n_components=3)
principal_components_3d = pca_3d.fit_transform(x)
pca_df_3d = pd.DataFrame(data=principal_components_3d, columns=['PC1', 'PC2
pca_df_3d = pd.concat([pca_df_3d, df[['species']]], axis=1)
```

In [36]:
```
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')
ax.scatter(pca_df_3d['PC1'], pca_df_3d['PC2'], pca_df_3d['PC3'], c=pd.Categ
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.title('PCA - 3D Projection of Iris Dataset')
plt.show()
```


PCA - 3D Projection of Iris Dataset
PC3
PC1
PC2