In [1]:
```
# Importing Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:
```
iris = pd.read_csv('Iris.csv')
iris
# The code imports the necessary libraries: pandas, numpy, seaborn, and matplotlib.pyplot.
# It then reads the Iris dataset from a CSV file using the pd.read_csv() function and stores it in a
```

Out[2]:

	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(20)
```

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
5	6	5.4	3.9	1.7	0.4	Iris-setosa
6	7	4.6	3.4	1.4	0.3	Iris-setosa
7	8	5.0	3.4	1.5	0.2	Iris-setosa
8	9	4.4	2.9	1.4	0.2	Iris-setosa
9	10	4.9	3.1	1.5	0.1	Iris-setosa
10	11	5.4	3.7	1.5	0.2	Iris-setosa
11	12	4.8	3.4	1.6	0.2	Iris-setosa
12	13	4.8	3.0	1.4	0.1	Iris-setosa
13	14	4.3	3.0	1.1	0.1	Iris-setosa
14	15	5.8	4.0	1.2	0.2	Iris-setosa
15	16	5.7	4.4	1.5	0.4	Iris-setosa
16	17	5.4	3.9	1.3	0.4	Iris-setosa
17	18	5.1	3.5	1.4	0.3	Iris-setosa
18	19	5.7	3.8	1.7	0.3	Iris-setosa
19	20	5.1	3.8	1.5	0.3	Iris-setosa

In [3]:
```
iris.describe()
```

Out[3]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

In [4]:
```
iris.isnull().sum()
```

Out[4]:
```
Id               0
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

In [5]:
```
iris.columns
```

Out[5]:
```
Index(['Id', 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm',
       'Species'],
      dtype='object')
```

SepalLengthCm : Numerical variable

SepalWidthCm: Numerical variable

PetalLengthCm: Numerical variable

PetalWidthCm: Numerical variable

Species: Categorial Variable

```
Categories: Iris-setosa
            Iris-versicolor
            Iris-virginica
```

In [6]:
```
iris.dtypes
```

Out[6]:
```
Id               int64
SepalLengthCm    float64
SepalWidthCm     float64
PetalLengthCm    float64
PetalWidthCm     float64
Species           object
dtype: object
```

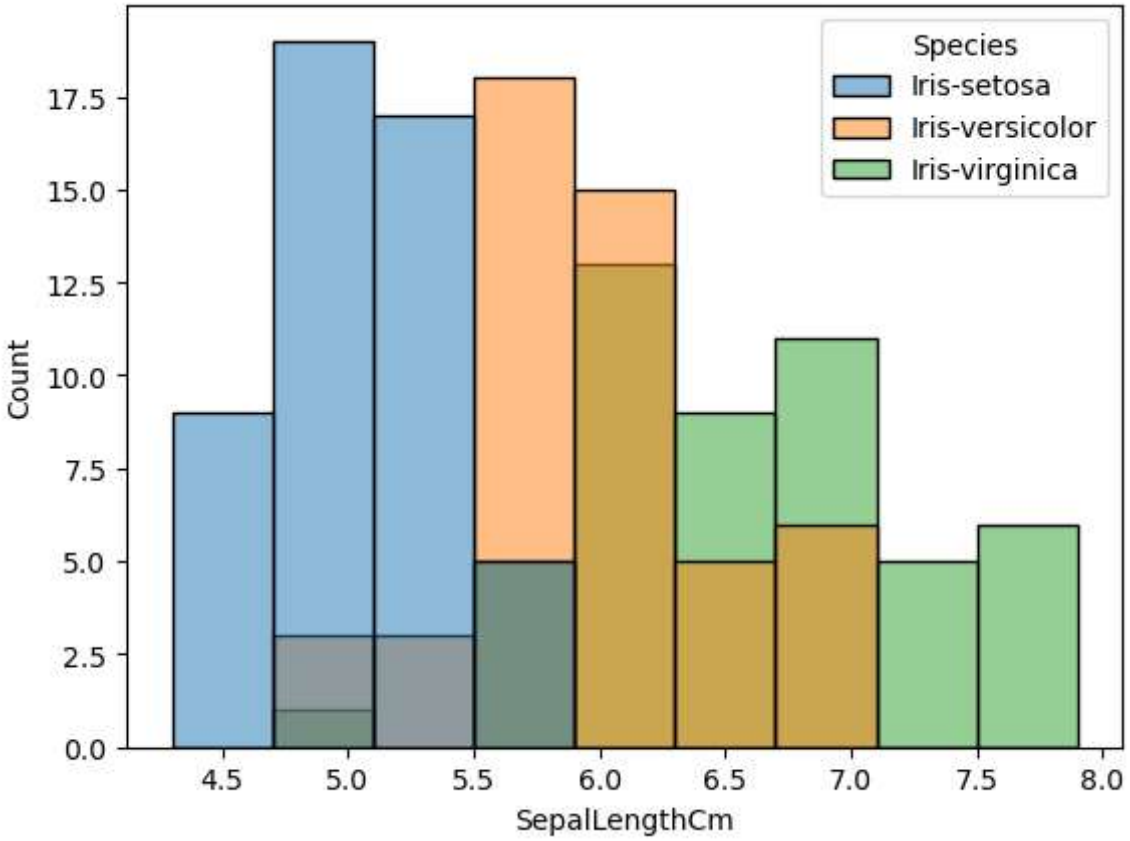
In [7]:
```
sns.countplot(iris['Species'])
# bar plot showing the count of each species in the Iris dataset.
# It will display the number of samples for each species on the y-axis and the species names on
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[7], line 1
----> 1 sns.countplot(iris['Species'])

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\categorical.py:2943, in countplot(data, x, y, hue, order, hue_order, orient, color, palette, saturation, width, dodge, ax, **kwargs)
   2940 elif x is not None and y is not None:
   2941     raise ValueError("Cannot pass values for both `x` and `y`")
-> 2943 plotter = _CountPlotter(
   2944     x, y, hue, data, order, hue_order,
   2945     estimator, errorbar, n_boot, units, seed,
   2946     orient, color, palette, saturation,
   2947     width, errcolor, errwidth, capsize, dodge
   2948 )
   2950 plotter.value_label = "count"
   2952 if ax is None:

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\categorical.py:1530, in _BarPlotter.__init__(self, x, y, hue, data, order, hue_order, estimator, errorbar, n_boot, units, seed, orient, color, palette, saturation, width, errcolor, errwidth, capsize, dodge)
   1525 def __init__(self, x, y, hue, data, order, hue_order,
   1526              estimator, errorbar, n_boot, units, seed,
   1527              orient, color, palette, saturation, width,
   1528              errcolor, errwidth, capsize, dodge):
   1529     """Initialize the plotter."""
-> 1530     self.establish_variables(x, y, hue, data, orient,
   1531                              order, hue_order, units)
   1532     self.establish_colors(color, palette, saturation)
   1533     self.estimate_statistic(estimator, errorbar, n_boot, seed)

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\categorical.py:516, in _CategoricalPlotter.establish_variables(self, x, y, hue, data, orient, order, hue_order, units)
    513     plot_data = data
    515 # Convert to a list of arrays, the common representation
--> 516 plot_data = [np.asarray(d, float) for d in plot_data]
    518 # The group names will just be numeric indices
    519 group_names = list(range(len(plot_data)))

File C:\ProgramData\anaconda3\lib\site-packages\seaborn\categorical.py:516, in <listcomp>(.0)
    513     plot_data = data
    515 # Convert to a list of arrays, the common representation
--> 516 plot_data = [np.asarray(d, float) for d in plot_data]
    518 # The group names will just be numeric indices
    519 group_names = list(range(len(plot_data)))

File C:\ProgramData\anaconda3\lib\site-packages\pandas\core\series.py:893, in Series.__array__(self, dtype)
    846 def __array__(self, dtype: npt.DTypeLike | None = None) -> np.ndarray:
    847     """
    848     Return the values as a NumPy array.
    849
   (...)
    891           dtype='datetime64[ns]')
    892     """
--> 893     return np.asarray(self._values, dtype)

ValueError: could not convert string to float: 'Iris-setosa'
```

In [8]:
```
sns.histplot(data=iris, x="SepalLengthCm",hue="Species")
# histogram plot showing the distribution of sepal lengths for each species in the Iris dataset.
```

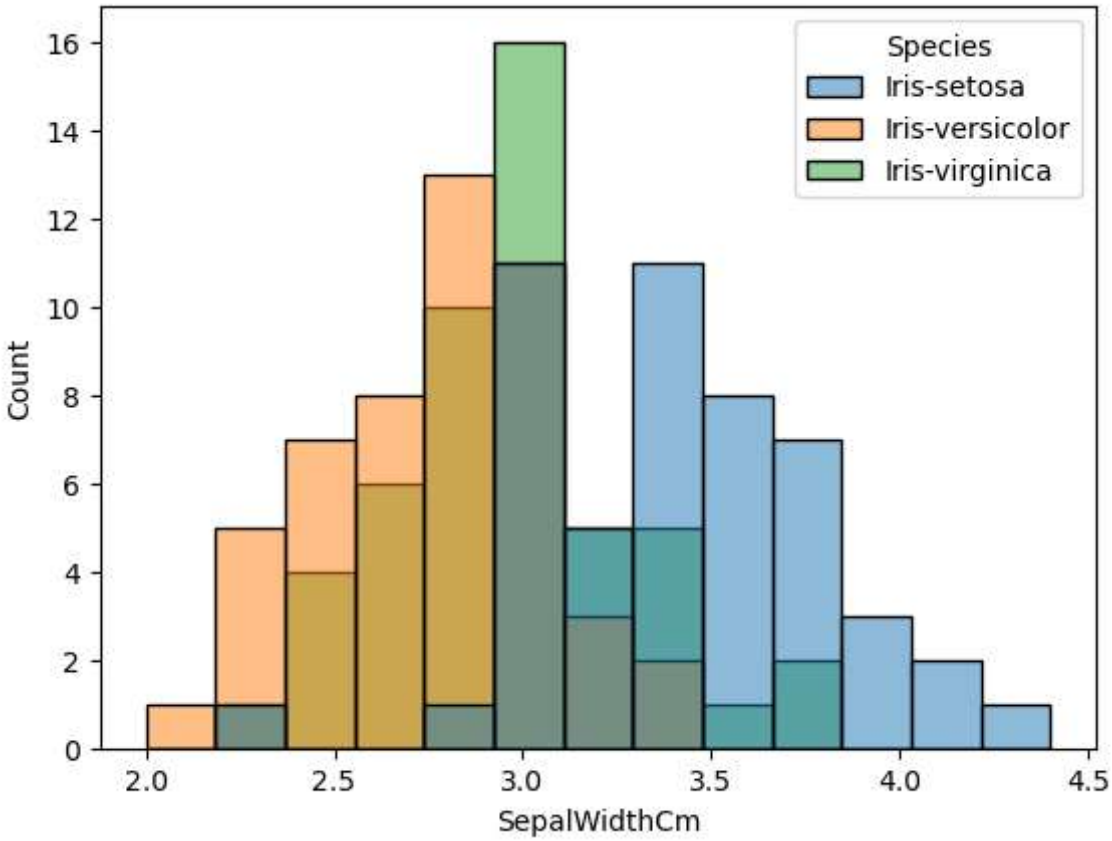
```
# The plot will have the sepal length values on the x-axis and the count of samples on the y-axis
# The histogram bars will be colored differently based on the species they belong to.
```

Out[8]: <Axes: xlabel='SepalLengthCm', ylabel='Count'>



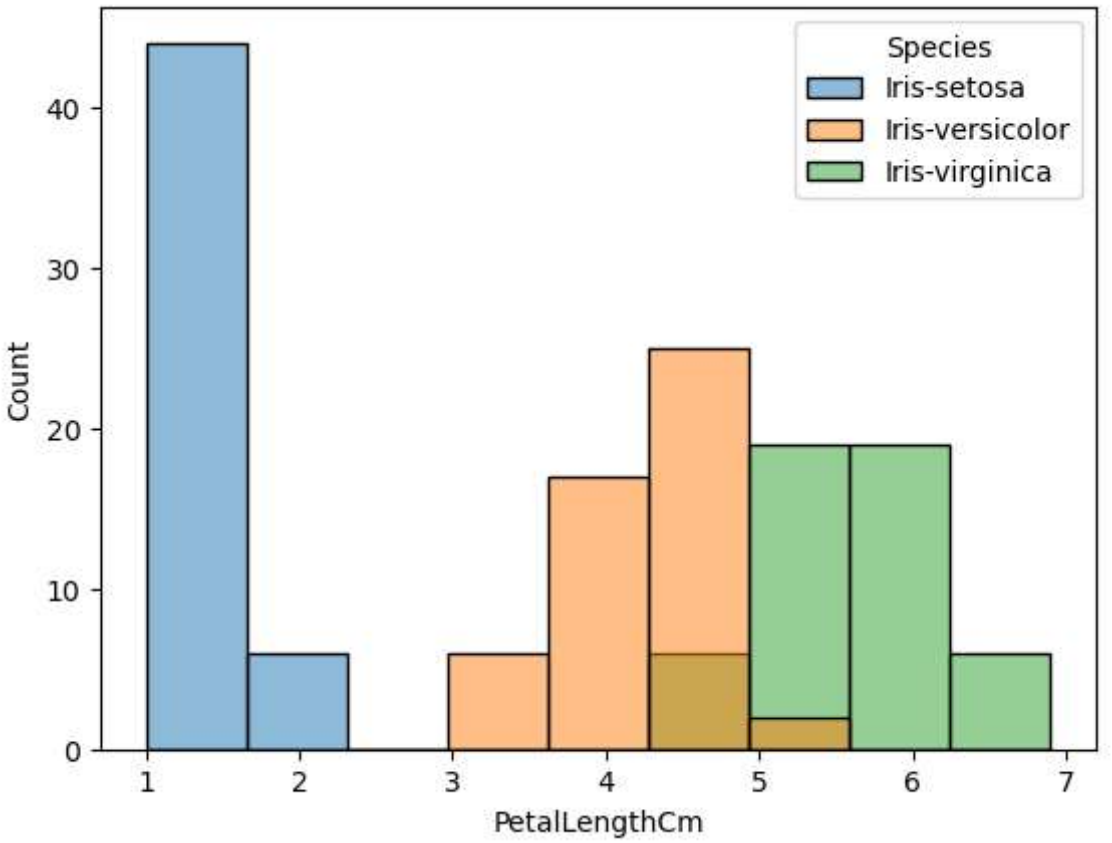
In [9]:
```
sns.histplot(data=iris, x="SepalWidthCm",hue="Species")
# histogram plot showing the distribution of sepal widths for each species in the Iris dataset.
# The plot will have the sepal width values on the x-axis and the count of samples on the y-axis.
# The histogram bars will be colored differently based on the species they belong to.
```

Out[9]: <Axes: xlabel='SepalWidthCm', ylabel='Count'>



In [10]:
```
sns.histplot(data=iris, x="PetalLengthCm",hue="Species")
#  histogram plot of the 'PetalLengthCm' column in the iris dataframe, which contains the petal l
#  Additionally, we are using the hue parameter to differentiate between the three species of the
```

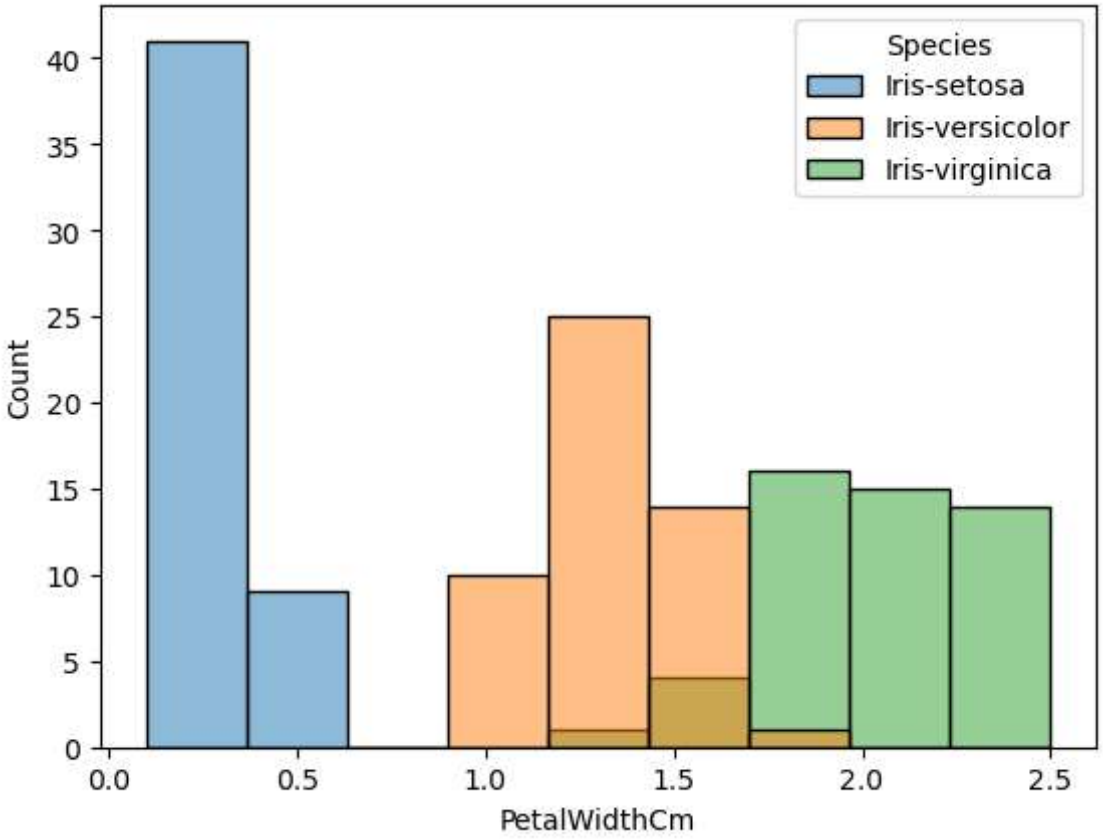
Out[10]: <Axes: xlabel='PetalLengthCm', ylabel='Count'>



In [11]:
```
sns.histplot(data=iris, x="PetalWidthCm",hue="Species")
# histogram plot showing the distribution of petal widths for each species in the Iris dataset.
```

```
# The plot will have the petal width values on the x-axis and the count of samples on the y-axis.
# The histogram bars will be colored differently based on the species they belong to.
```

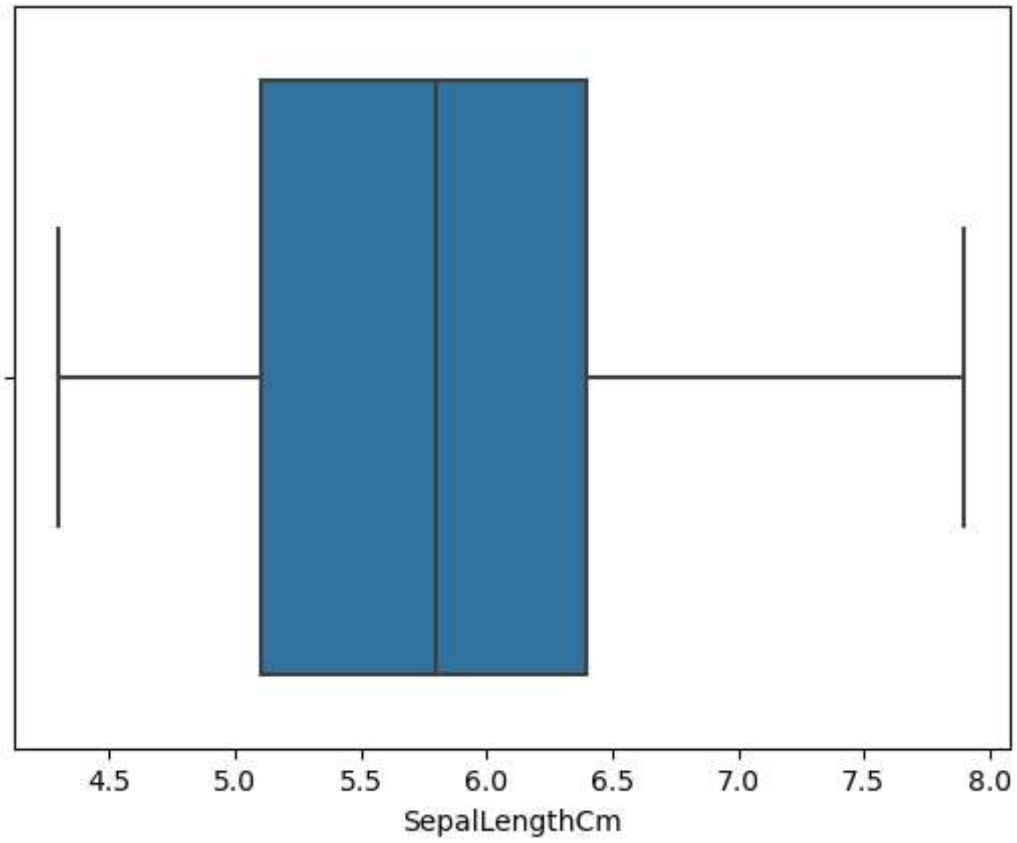
Out[11]: <Axes: xlabel='PetalWidthCm', ylabel='Count'>



Create]ing a BOX plot for each feature in the dataset.

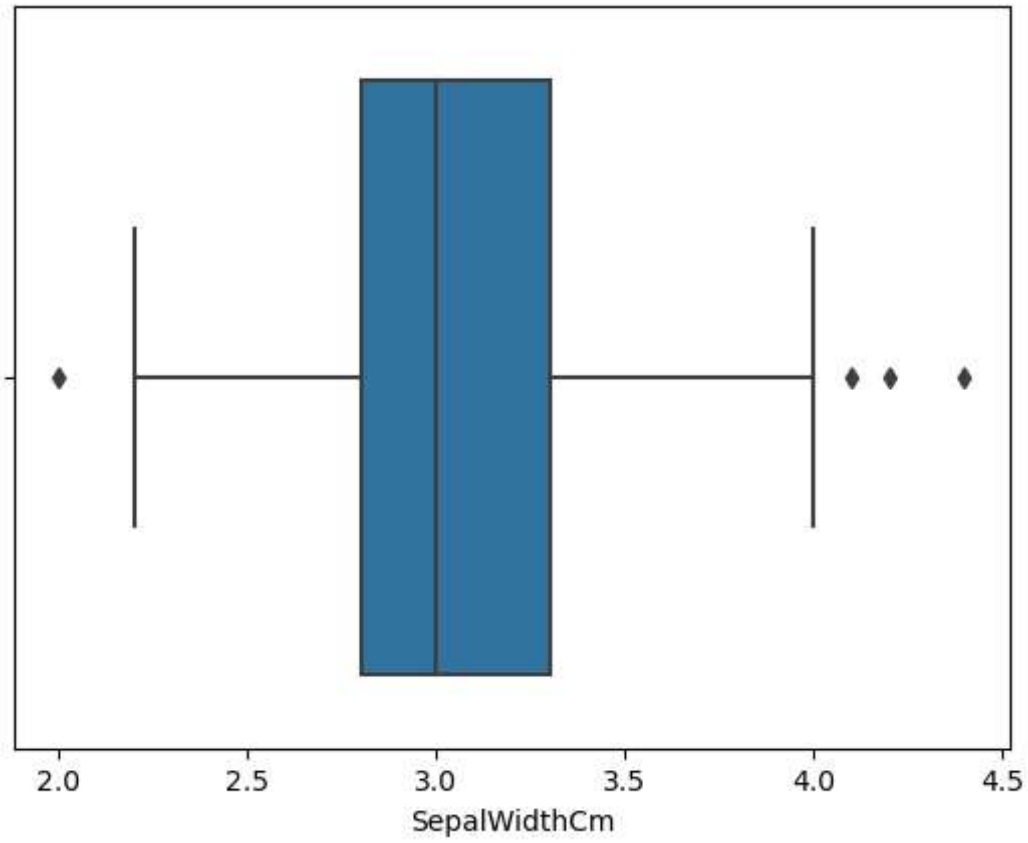
In [12]:
```
sns.boxplot(x=iris["SepalLengthCm"])
# we are creating a box plot of the 'SepalLengthCm' column in the iris dataframe, which contains
```

Out[12]: <Axes: xlabel='SepalLengthCm'>



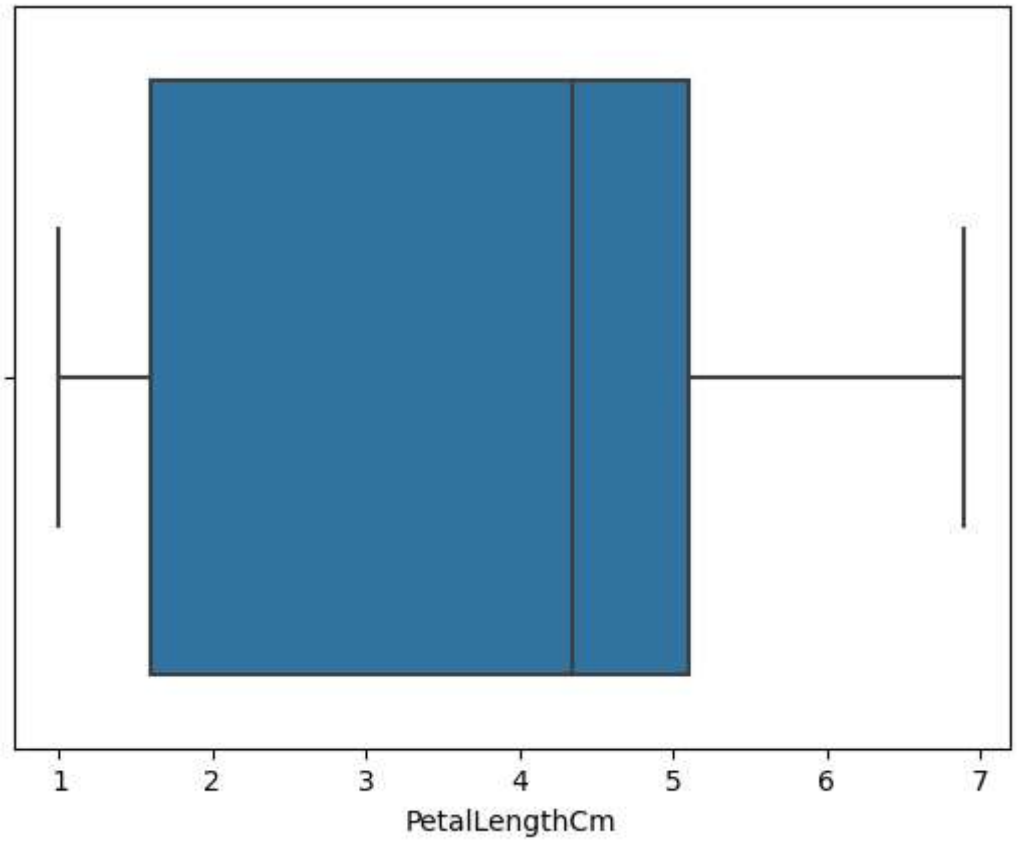
In [13]:
```
sns.boxplot(x=iris["SepalWidthCm"])
```

Out[13]: <Axes: xlabel='SepalWidthCm'>



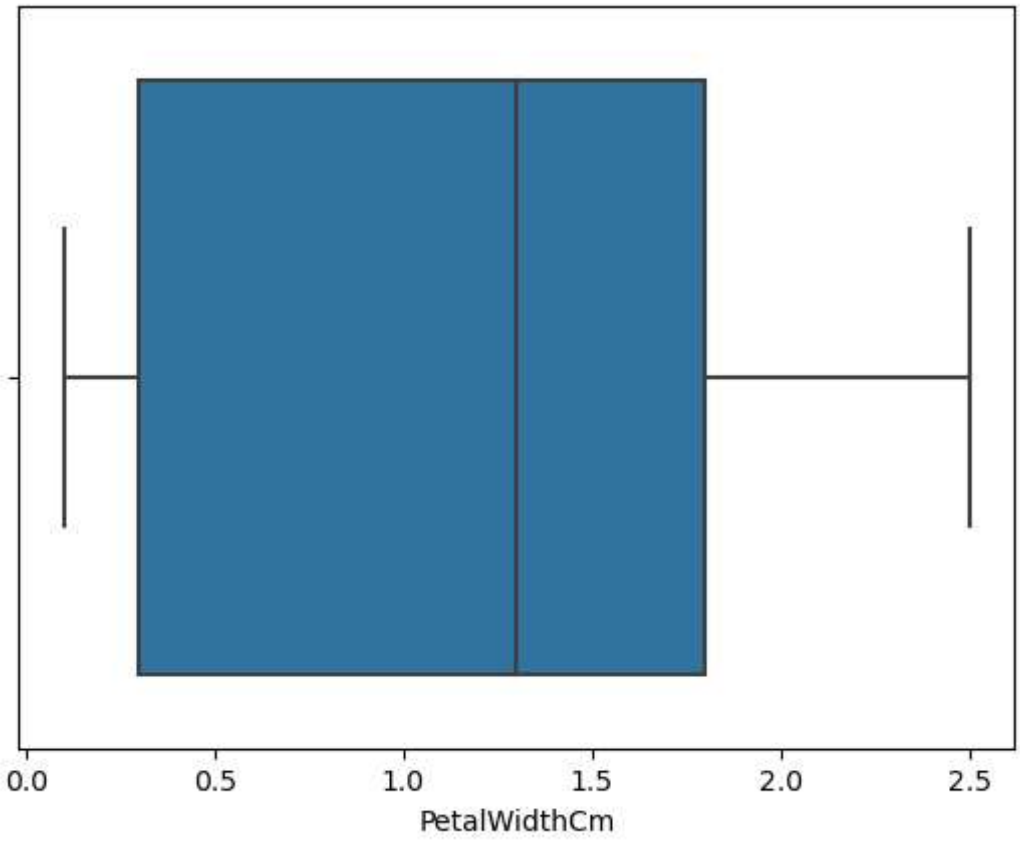
In [14]:
```
sns.boxplot(x=iris["PetalLengthCm"])
```

Out[14]: <Axes: xlabel='PetalLengthCm'>



In [15]:
```
sns.boxplot(x=iris["PetalWidthCm"])
```

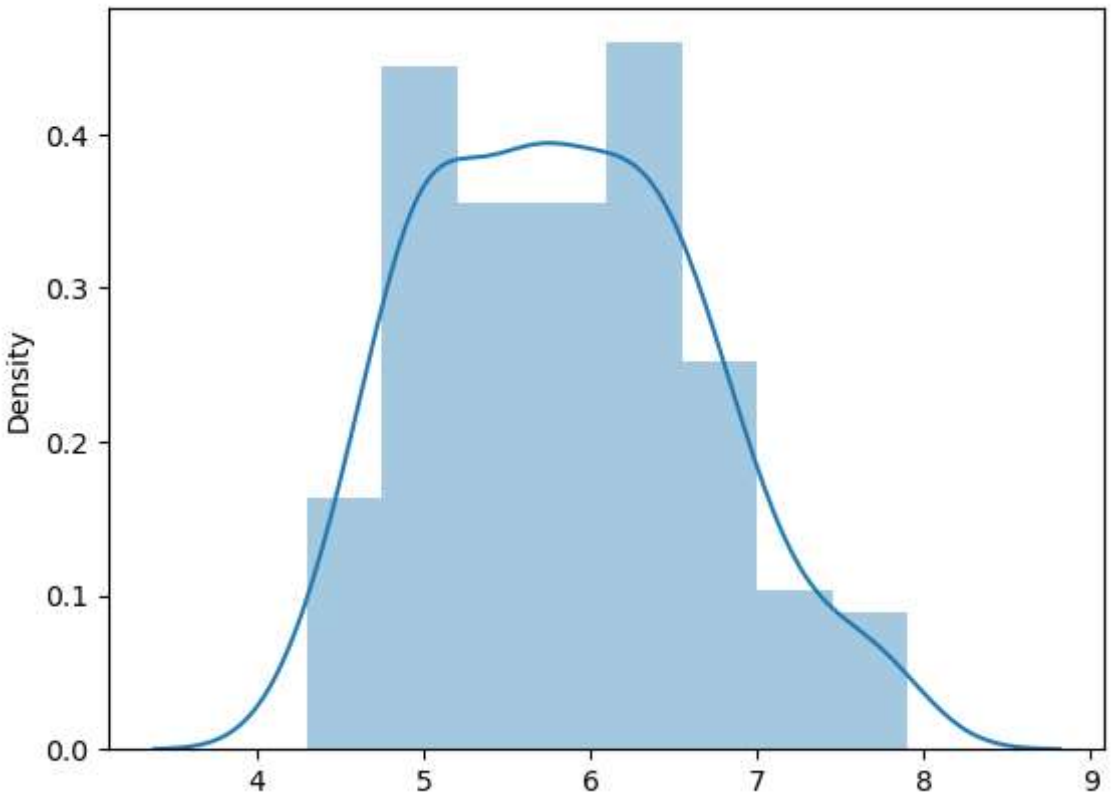
Out[15]: <Axes: xlabel='PetalWidthCm'>



In [16]:
```
sns.distplot(x=iris.SepalLengthCm)
# density plot showing the distribution of sepal lengths in the Iris dataset.
# The plot will have the sepal length values on the x-axis and the density of observations on the
```

```
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\3282725837.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.SepalLengthCm)
```

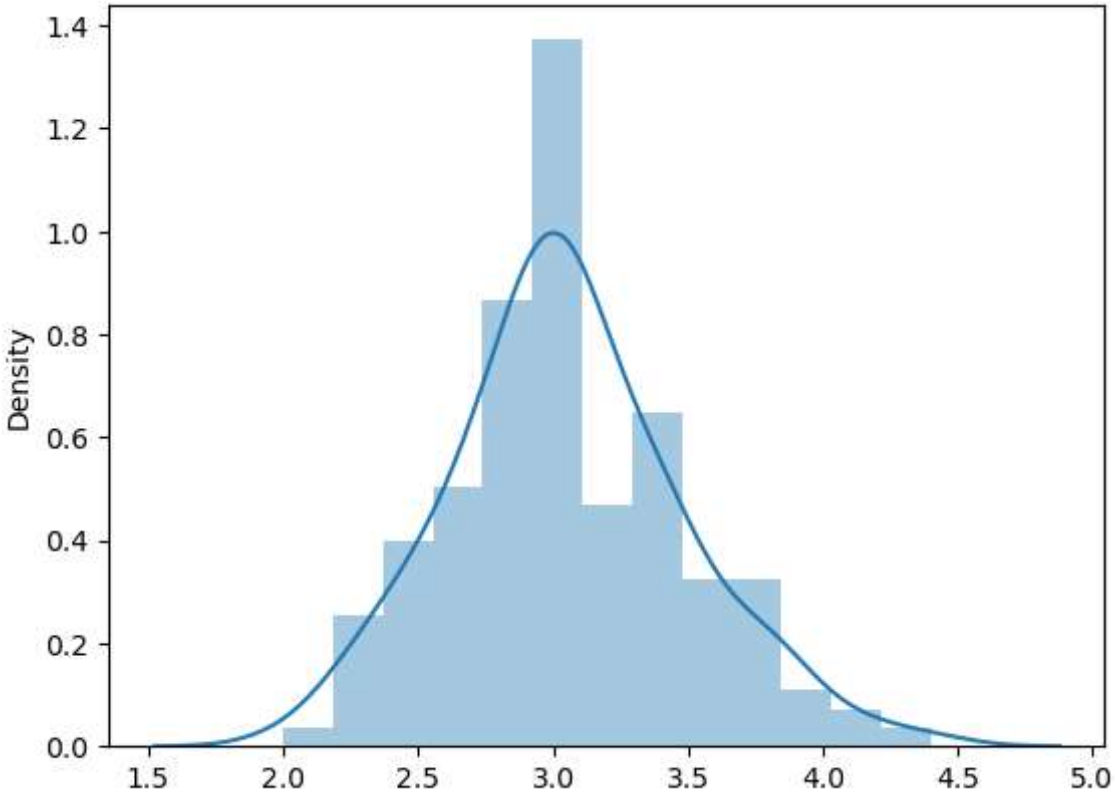
Out[16]: <Axes: ylabel='Density'>



In [17]:
```
sns.distplot(x=iris.SepalWidthCm)
```

```
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\254127868.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.SepalWidthCm)
```

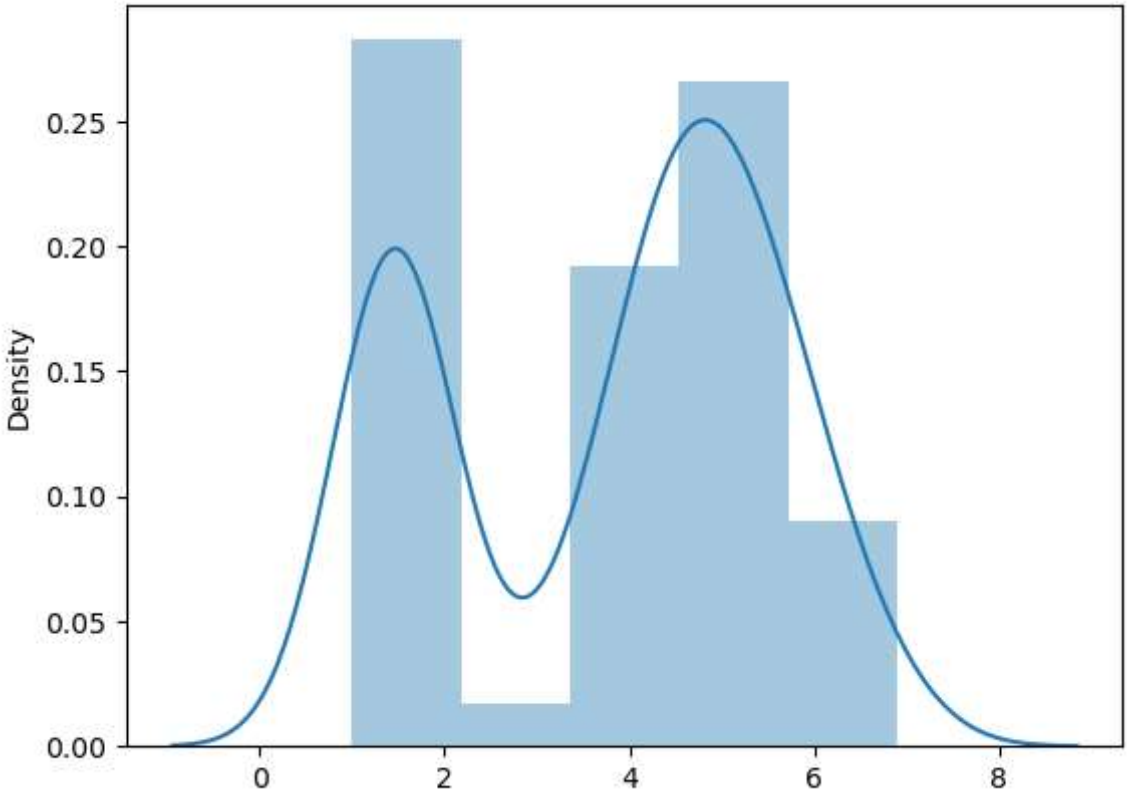
Out[17]: <Axes: ylabel='Density'>



In [18]:
```
sns.distplot(x=iris.PetalLengthCm)
```

```
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\2623398589.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.PetalLengthCm)
```

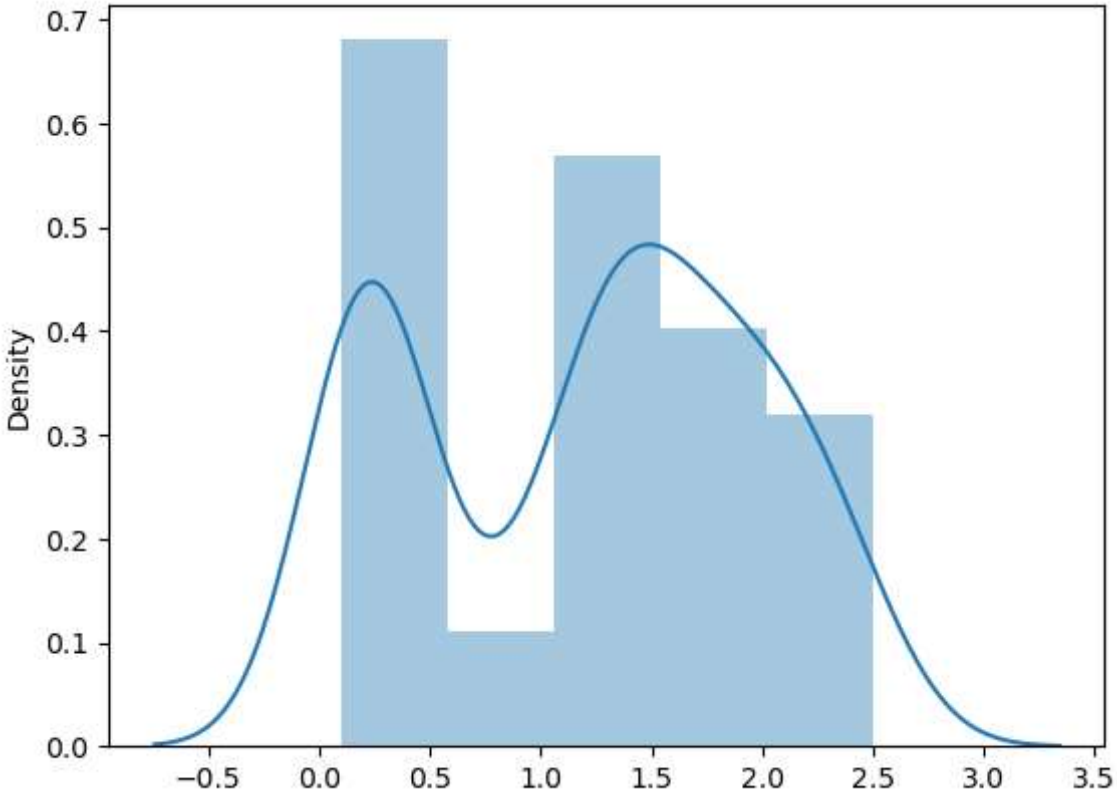
Out[18]: <Axes: ylabel='Density'>



In [19]:
```
sns.distplot(x=iris.PetalWidthCm)
```

```
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\64046386.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.PetalWidthCm)
```

Out[19]:
```
<Axes: ylabel='Density'>
```



In [20]:
```
Q1=iris['SepalWidthCm'].quantile(0.25)
Q3=iris['SepalWidthCm'].quantile(0.75)
IQR=Q3-Q1
print("IQR(", IQR, ") =", "Q3(", Q3, ")- Q1(", Q1, ")")
# Calculating the 25th percentile and 75th percentile of the 'SepalWidthCm' column using the qu
# We then subtract the 25th percentile from the 75th percentile to obtain the IQR.
# Finally, we print out the value of the IQR along with the corresponding Q1 and Q3 values.
```

```
IQR( 0.5 ) = Q3( 3.3 )- Q1( 2.8 )
```

In [21]:
```
lower_limit=Q1-IQR
upper_limit=Q3+IQR
lower_limit,upper_limit
# calculating the lower limit and upper limit using the Q1 and IQR values calculated in the previou
# We then assign these values to the variables lower_limit and upper_limit, respectively.
```

Out[21]:
```
(2.3, 3.8)
```

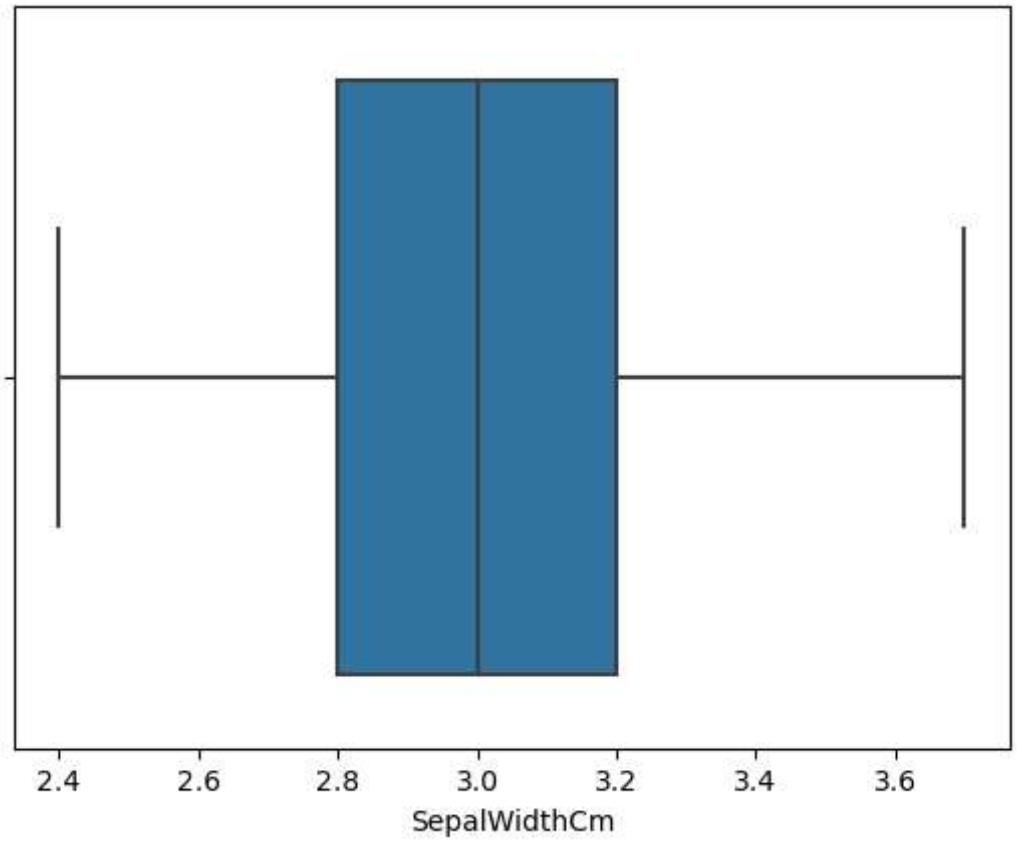
In [22]:
```
df_without_outliers=iris[(iris['SepalWidthCm']>lower_limit)&(iris['SepalWidthCm']<upper_limit)]
df_without_outliers
# we are creating a new dataframe called df_without_outliers that contains only the rows of the i
# calculated in the previous code snippets. We are using boolean indexing to select the rows wh
# limit. The resulting dataframe contains only the rows that meet these criteria, effectively remo
```

Out[22]:

	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

130 rows × 6 columns

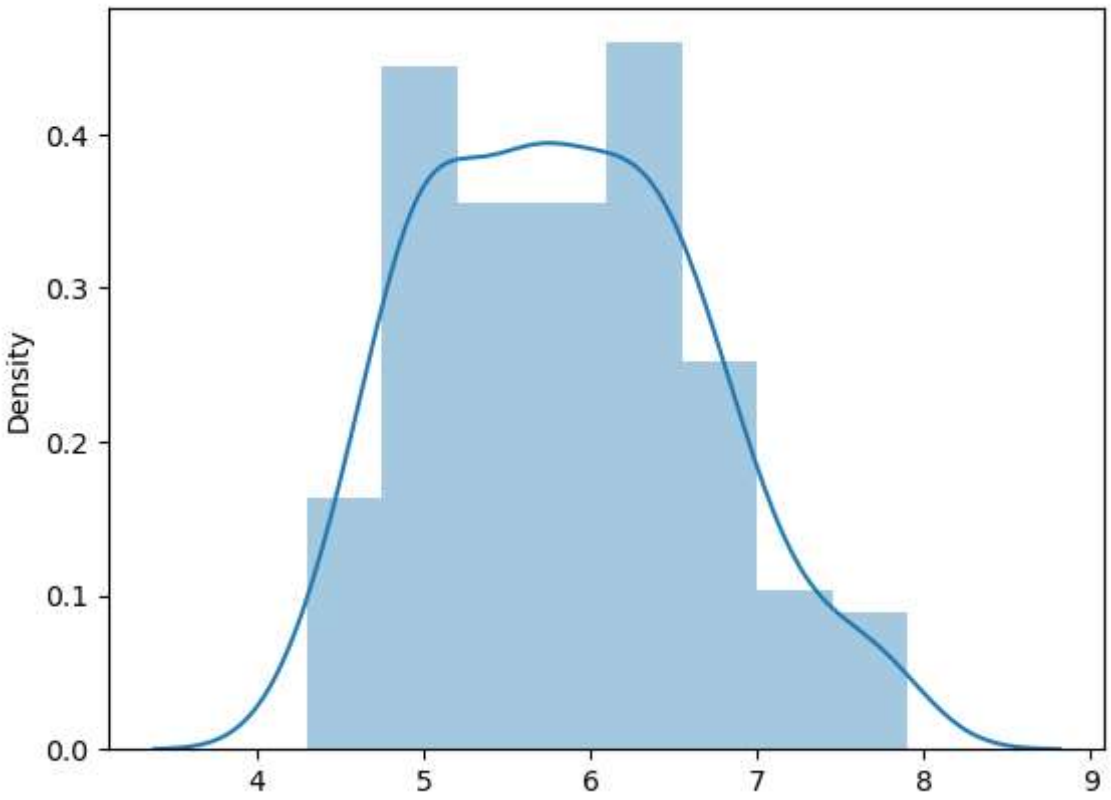
In [23]:
```
ax = sns.boxplot(x=df_without_outliers["SepalWidthCm"])
# box plot of the 'SepalWidthCm' column in the df_without_outliers dataframe.
# The box plot will show the median, quartiles, and any outliers that fall outside the whiskers.
```



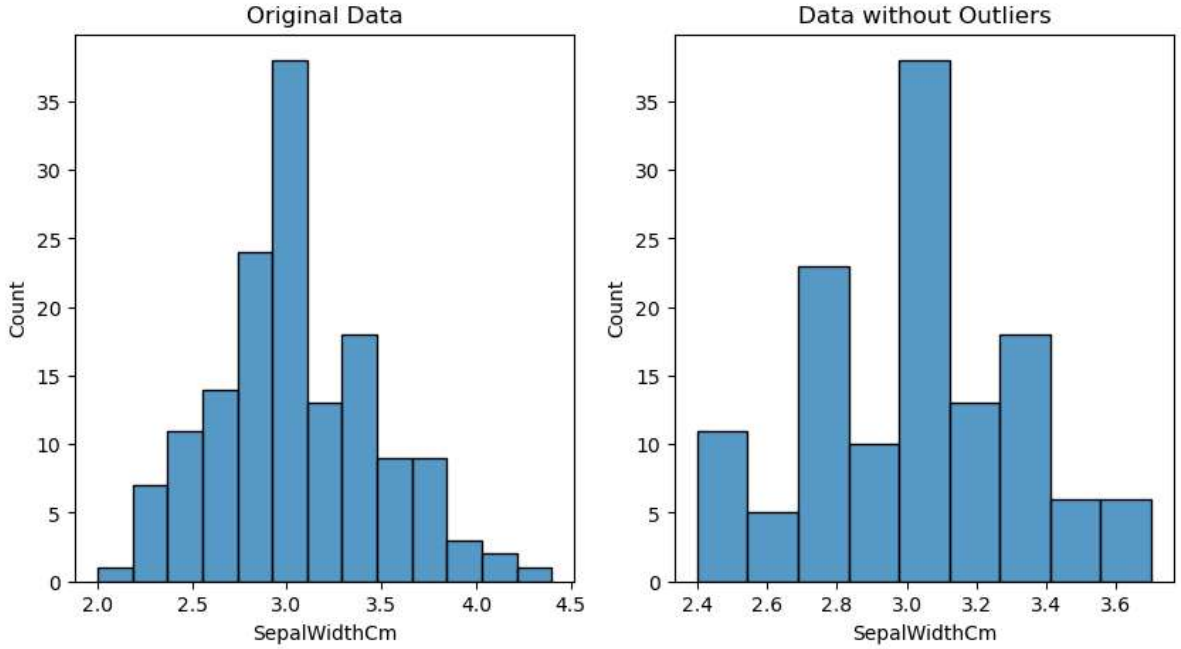
In [24]:
```
sns.distplot(x=iris.SepalLengthCm)
```

```
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\672948272.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.SepalLengthCm)
```

Out[24]: <Axes: ylabel='Density'>



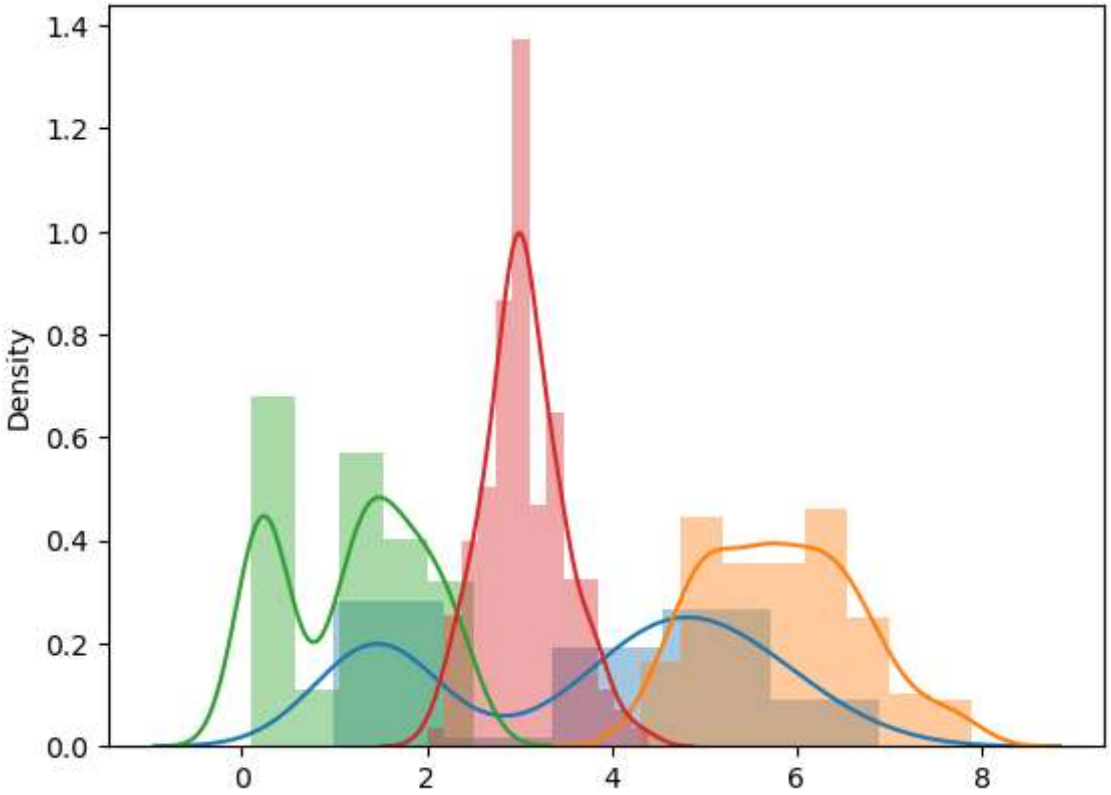
In [25]:
```
import matplotlib.pyplot as plt
# Create a figure with two subplots
fig, axs = plt.subplots(ncols=2, figsize=(10, 5))
sns.histplot(data=iris, x="SepalWidthCm", ax=axs[0])
axs[0].set_title('Original Data')
sns.histplot(data=df_without_outliers, x="SepalWidthCm", ax=axs[1])
axs[1].set_title('Data without Outliers')

plt.show()
# two histograms
# The left histogram shows the distribution of the 'SepalWidthCm' column in the original iris data
# while the right histogram shows the distribution of the 'SepalWidthCm' column in the df_withou
# histogram on the right has a slightly different shape than the one on the left, as the outliers ha
```



In [26]:
```
sns.distplot(x=iris.PetalLengthCm)
sns.distplot(x=iris.SepalLengthCm)
sns.distplot(x=iris.PetalWidthCm)
sns.distplot(x=iris.SepalWidthCm)
# sns.distplot(x=iris.PetalLengthCm) creates a histogram of the 'PetalLengthCm' column in the ir
# sns.distplot(x=iris.SepalLengthCm) creates a histogram of the 'SepalLengthCm' column in the
# sns.distplot(x=iris.PetalWidthCm) creates a histogram of the 'PetalWidthCm' column in the iris
# sns.distplot(x=iris.SepalWidthCm) creates a histogram of the 'SepalWidthCm' column in the iri
# When you run these code snippets, you should see four histograms stacked on top of each otl
# Each histogram shows the distribution of the respective column in the iris dataframe.
```

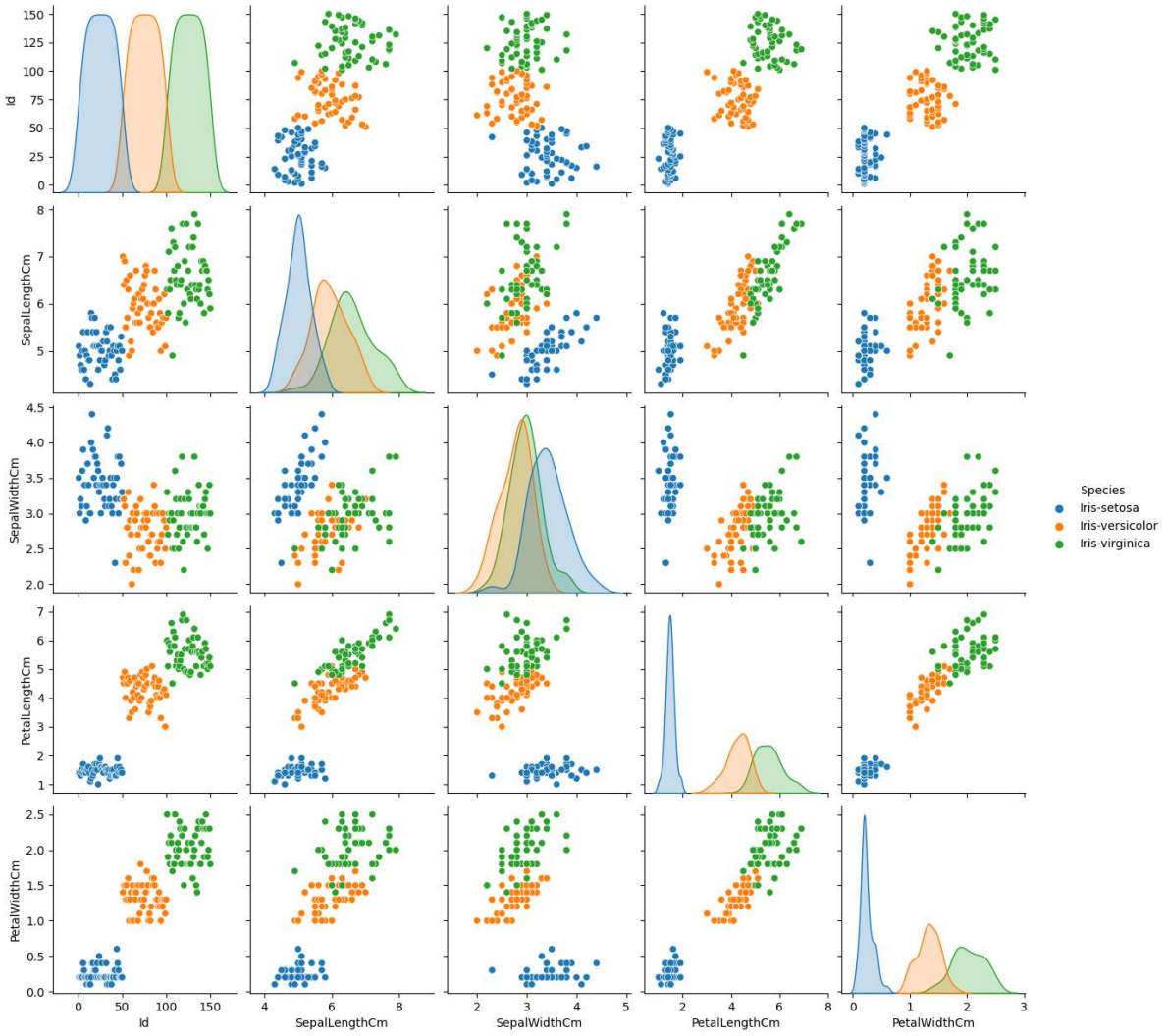
```
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\1629669816.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.PetalLengthCm)
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\1629669816.py:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.SepalLengthCm)
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\1629669816.py:3: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.PetalWidthCm)
C:\Users\SKY_NET\AppData\Local\Temp\ipykernel_14928\1629669816.py:4: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(x=iris.SepalWidthCm)
```

Out[26]: <Axes: ylabel='Density'>



In [27]:
```
sns.pairplot(data= iris, hue='Species')
# grid of scatter plots, with each row and column representing a different variable in the iris data
# The diagonal of the grid shows a histogram of each variable, and the off-diagonal elements sh
# The points in each scatter plot are colored according to the 'Species' column, allowing you to s
```

Out[27]:
```
<seaborn.axisgrid.PairGrid at 0x1bcfb33ff70>
```


Id
SepalLengthCm
SepalWidthCm
PetalLengthCm
PetalWidthCm
Species
Iris-setosa
Iris-versicolor
Iris-virginica