######1. load the iris.csv dataset.

1. create a bar plot for count the frequency of the three species.

In [4]:

```
### code here
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv("iris.csv")
df
```

Out[4]:

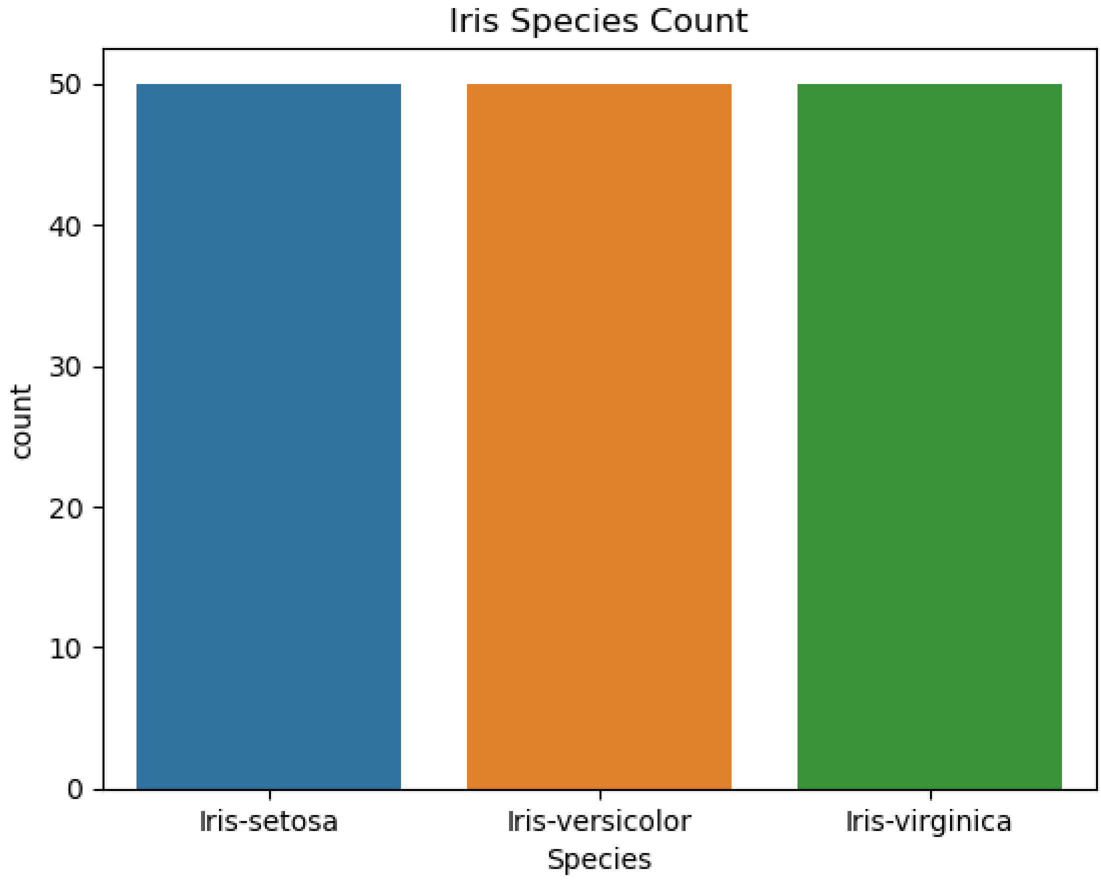
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	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 [9]:

```
sns.countplot('Species',data=df)
plt.title("Iris Species Count")
plt.show()
```

```
C:\Users\pamar\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Future
Warning: Pass the following variable as a keyword arg: x. From version 0.12,
the only valid positional argument will be `data`, and passing other argumen
ts without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



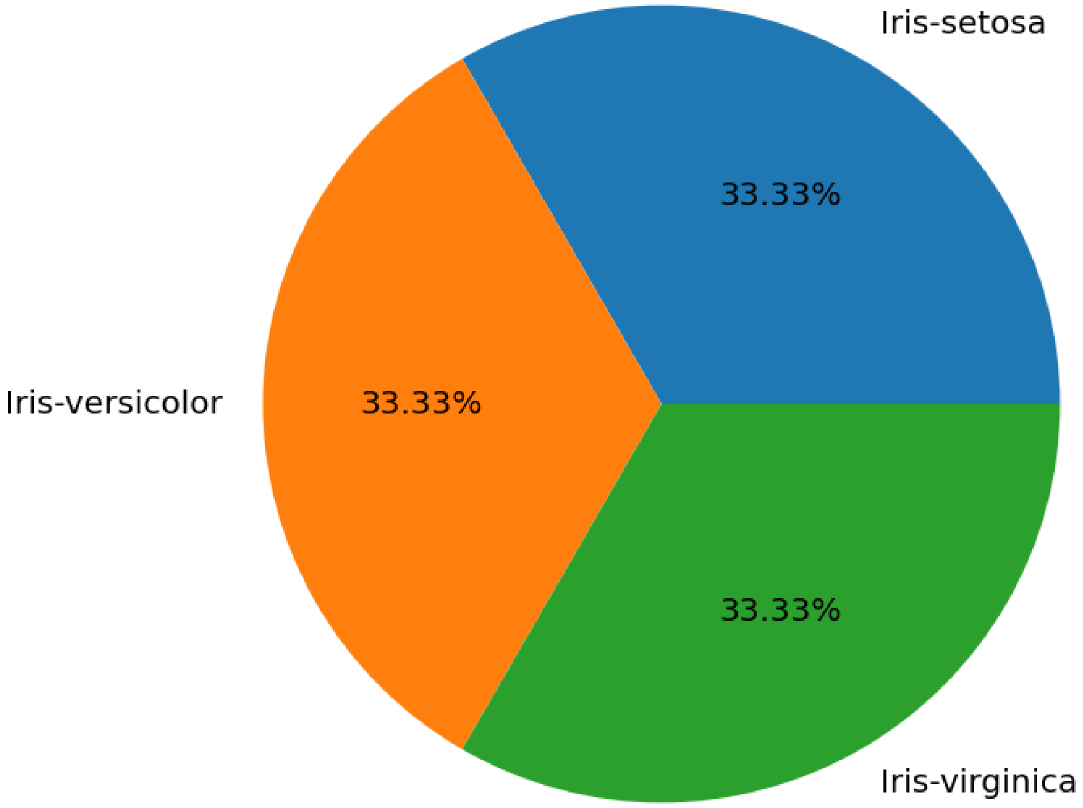
2. Create a pie chart to show the percentage of species.

In [7]:

```
### code here
a=df.Species.value_counts()
plt.figure(figsize=(8,8))
plt.pie(x=a,labels = a.index,autopct='%0.2f%%',textprops = {'size': 'x-large'})
```

Out[7]:

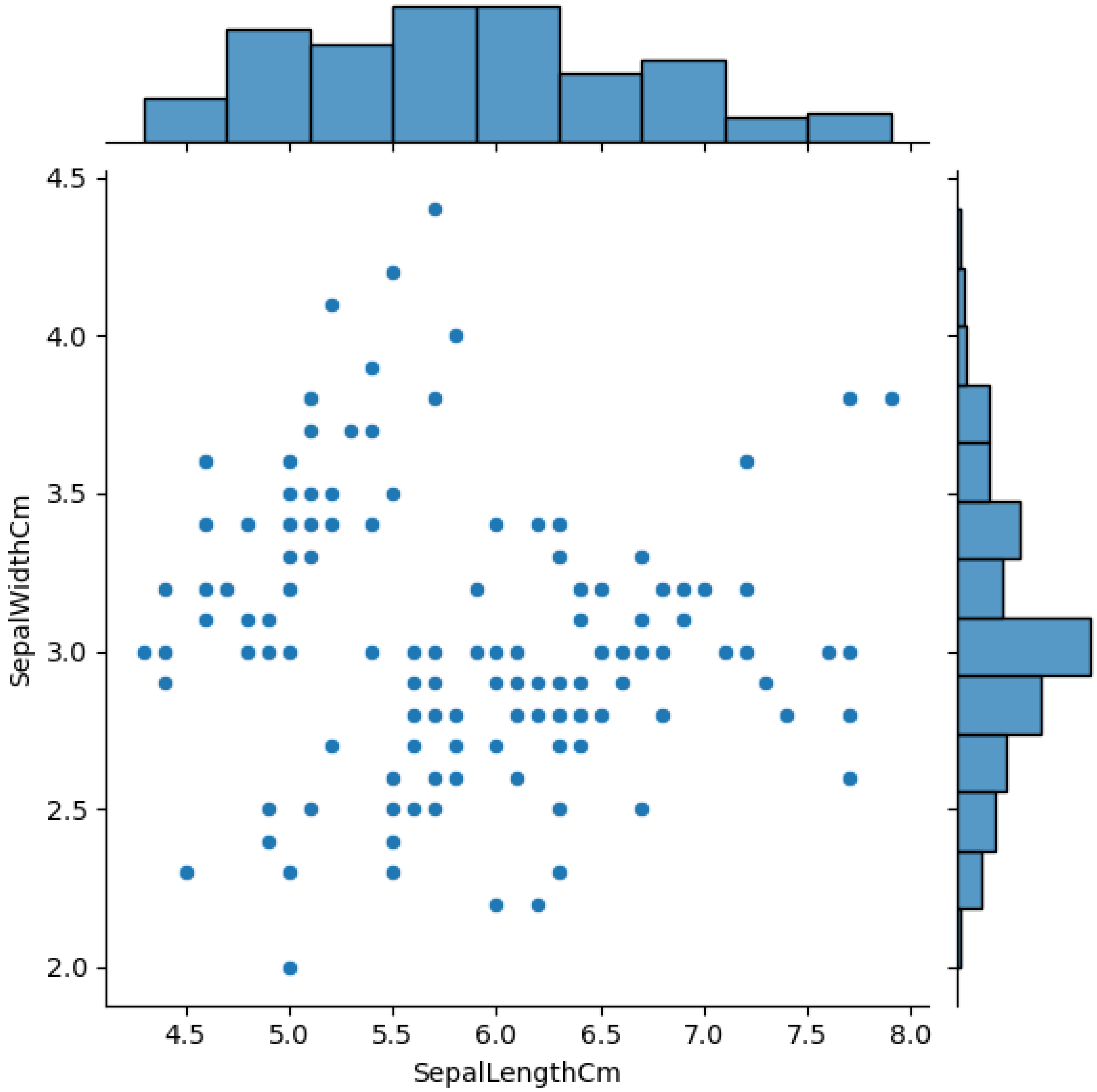
```
([<matplotlib.patches.Wedge at 0x20b76957df0>,
  <matplotlib.patches.Wedge at 0x20b7696b580>,
  <matplotlib.patches.Wedge at 0x20b7696bca0>],
 [Text(0.5499999702695115, 0.9526279613277875, 'Iris-setosa'),
  Text(-1.0999999999999954, -1.0298943258065002e-07, 'Iris-versicolor'),
  Text(0.5500001486524352, -0.9526278583383436, 'Iris-virginica')],
 [Text(0.2999999837833699, 0.5196152516333385, '33.33%'),
  Text(-0.5999999999999974, -5.6176054134900006e-08, '33.33%'),
  Text(0.30000008108314646, -0.5196151954572783, '33.33%')])
```

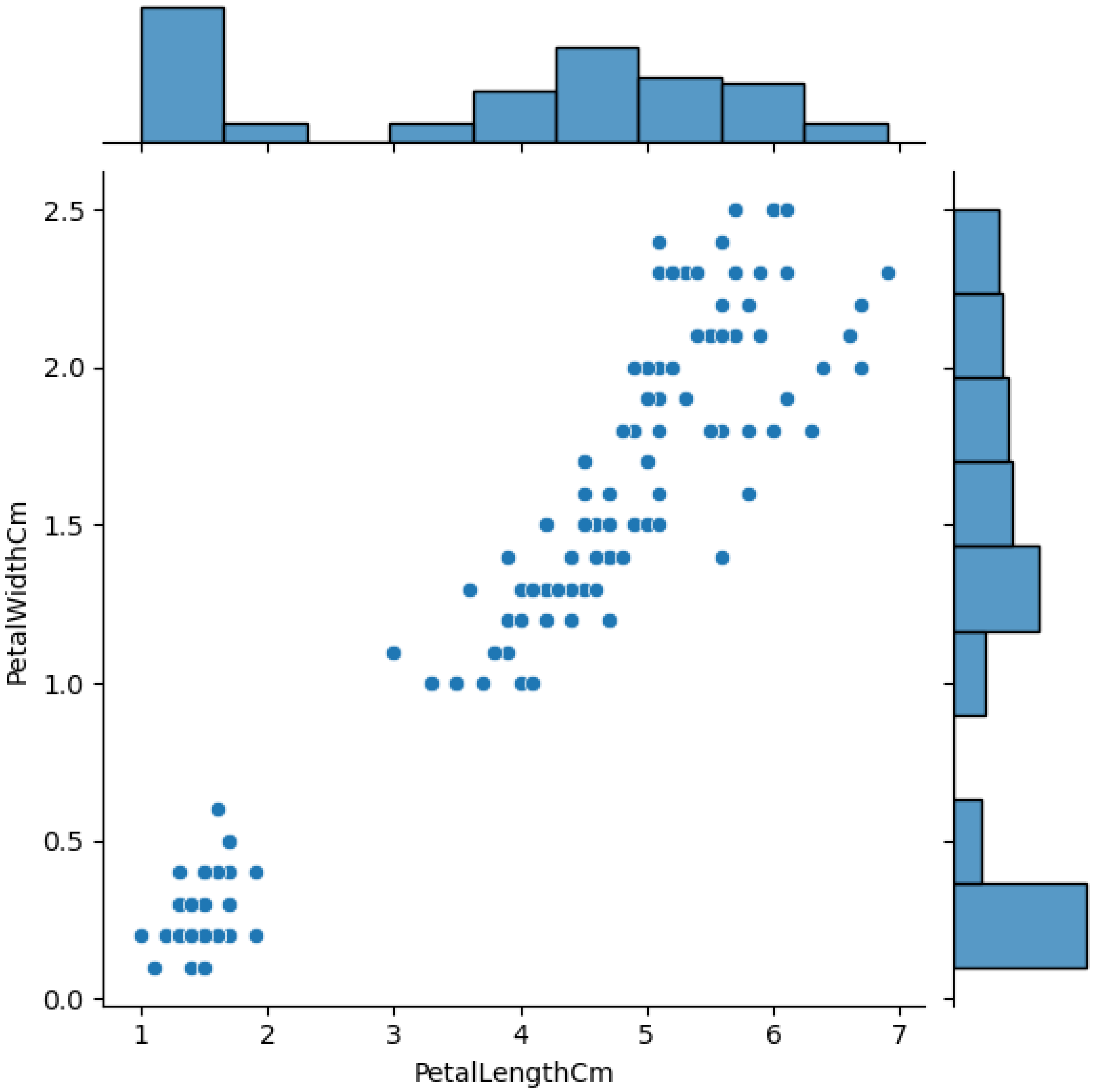


3. Create the joint to all numberical variable for analyze the relationship between two variables and describe their individual distributions on the same plot.

In [16]:

```
### code here
sns.jointplot(x='SepalLengthCm', y='SepalWidthCm', data=df)
sns.jointplot(x='PetalLengthCm', y='PetalWidthCm', data=df)

plt.show()
```

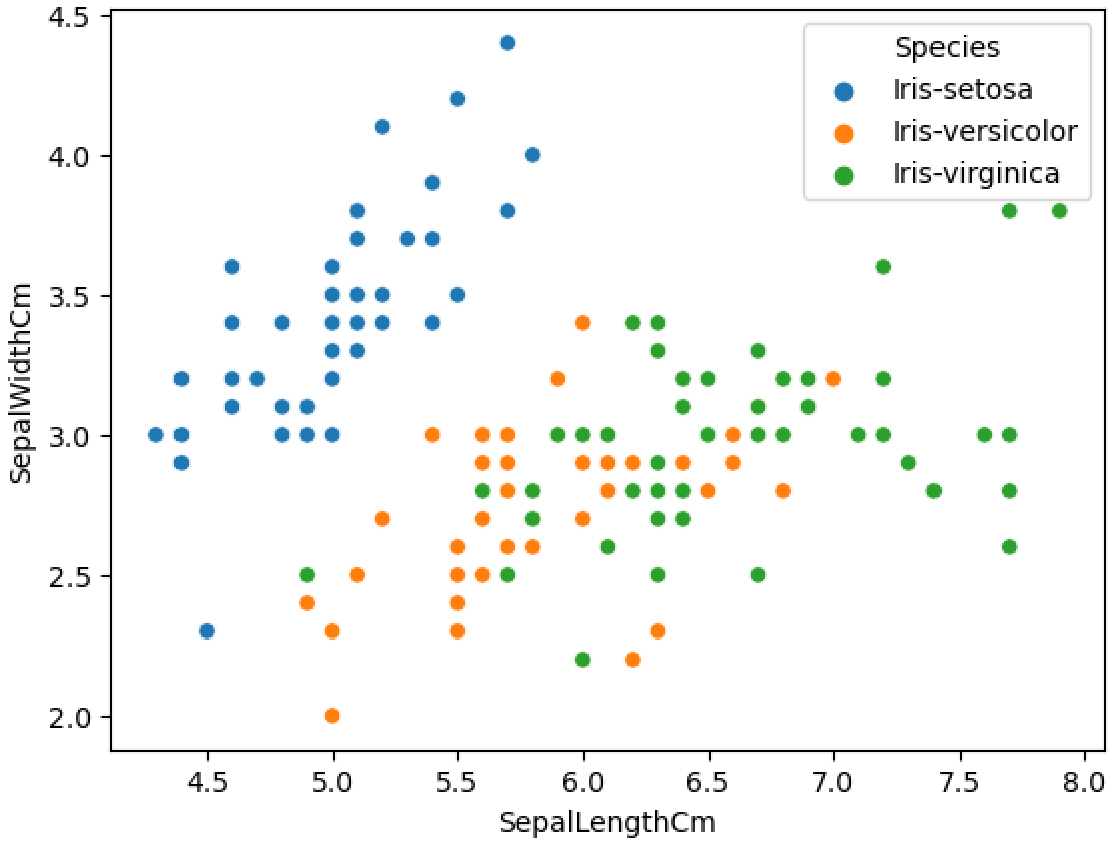


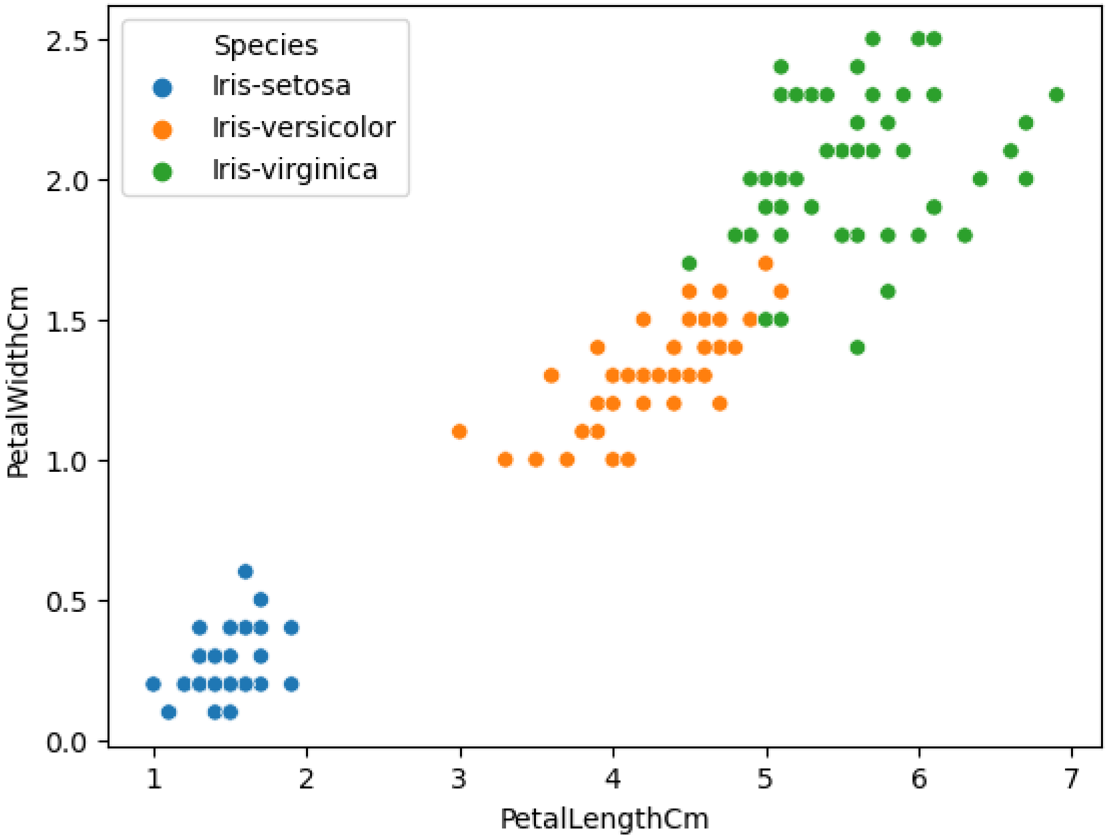


4. Create a scatter plot and take hue = "Species"

In [19]:

```
### code here
sns.scatterplot(x='SepalLengthCm', y='SepalWidthCm', data=df,hue="Species")
plt.show()
sns.scatterplot(x='PetalLengthCm', y='PetalWidthCm', data=df,hue="Species")
plt.show()
```





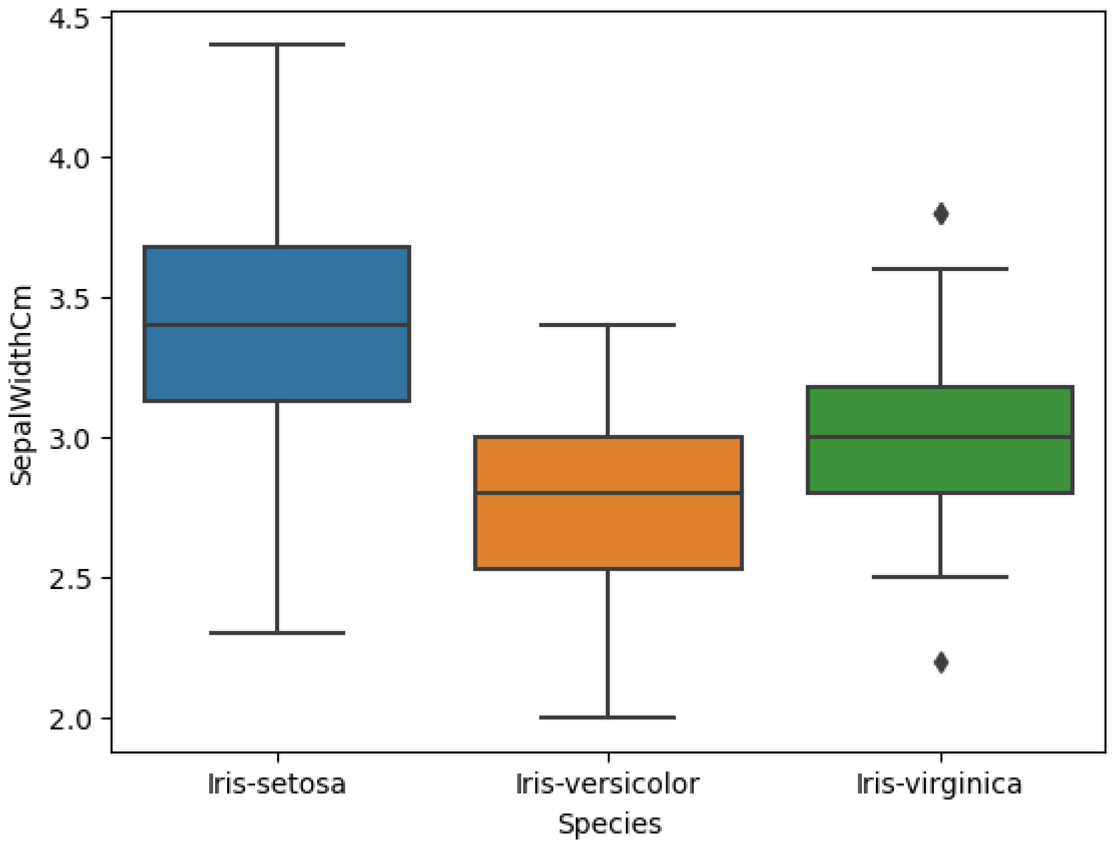
5. Create a Boxplot or Whisker plot as univeriate and bivariate analysis.

In [25]:

```
### code here
sns.boxplot(x='Species', y='SepalWidthCm', data=df)
```

Out[25]:

```
<AxesSubplot:xlabel='Species', ylabel='SepalWidthCm'>
```

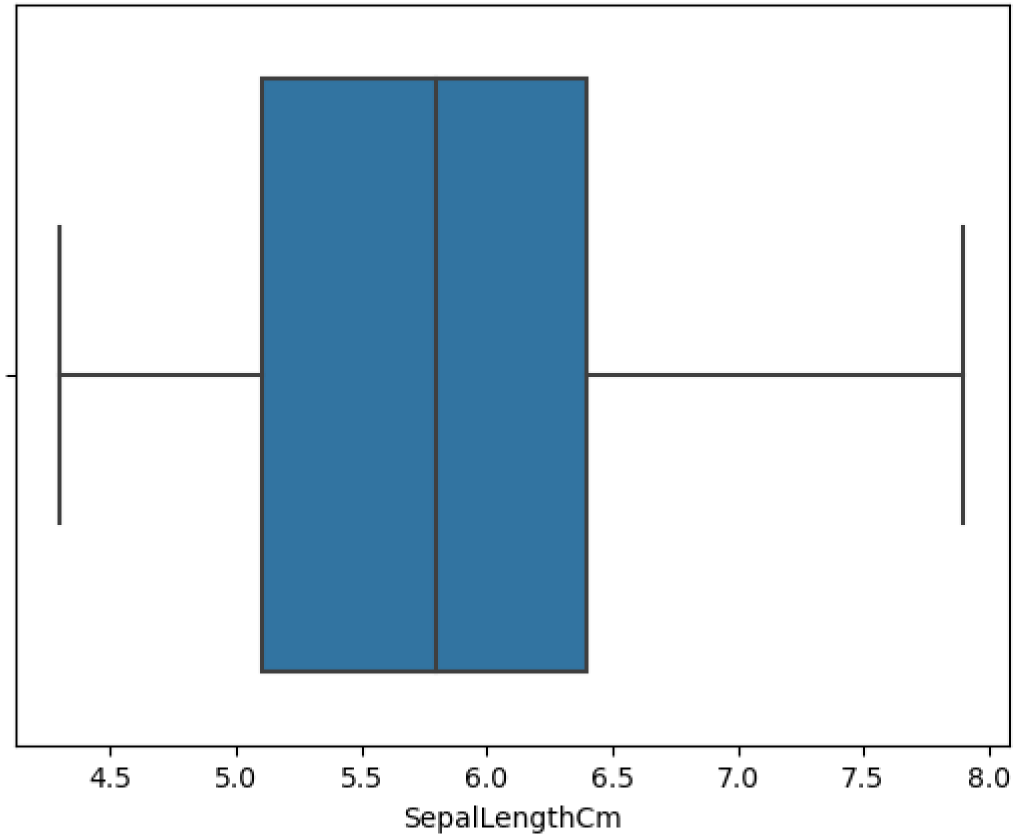


In [23]:

```
sns.boxplot(x=df['SepalLengthCm'])
```

Out[23]:

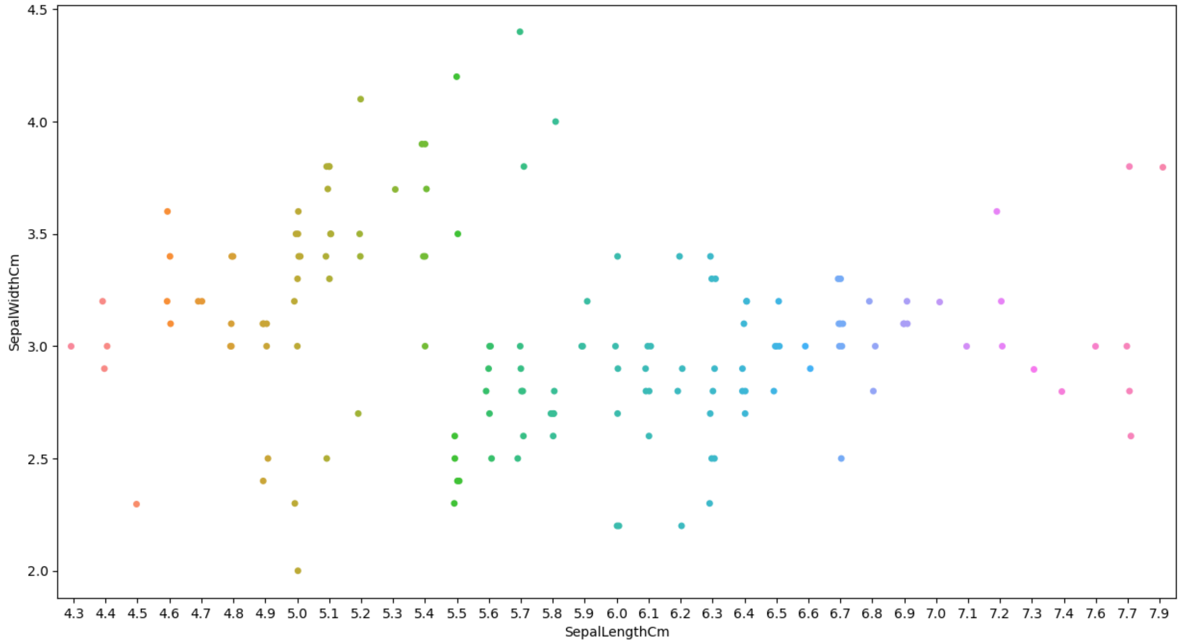
```
<AxesSubplot:xlabel='SepalLengthCm'>
```



6. Create a Strip plot

In [28]:

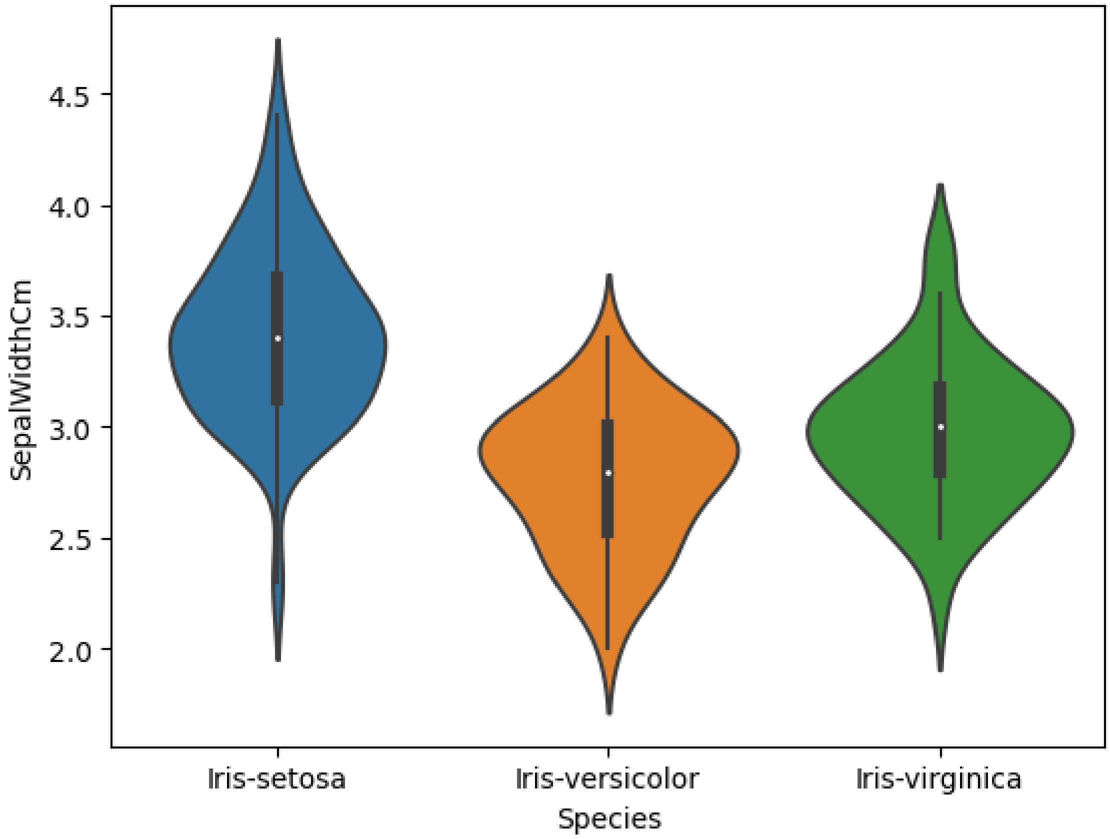
```
### code here
plt.figure(figsize=(15,8))
sns.stripplot(x='SepalLengthCm', y='SepalWidthCm', data=df,jitter=True)

plt.show()
```



7. Create a Violin Plot.

In [29]:

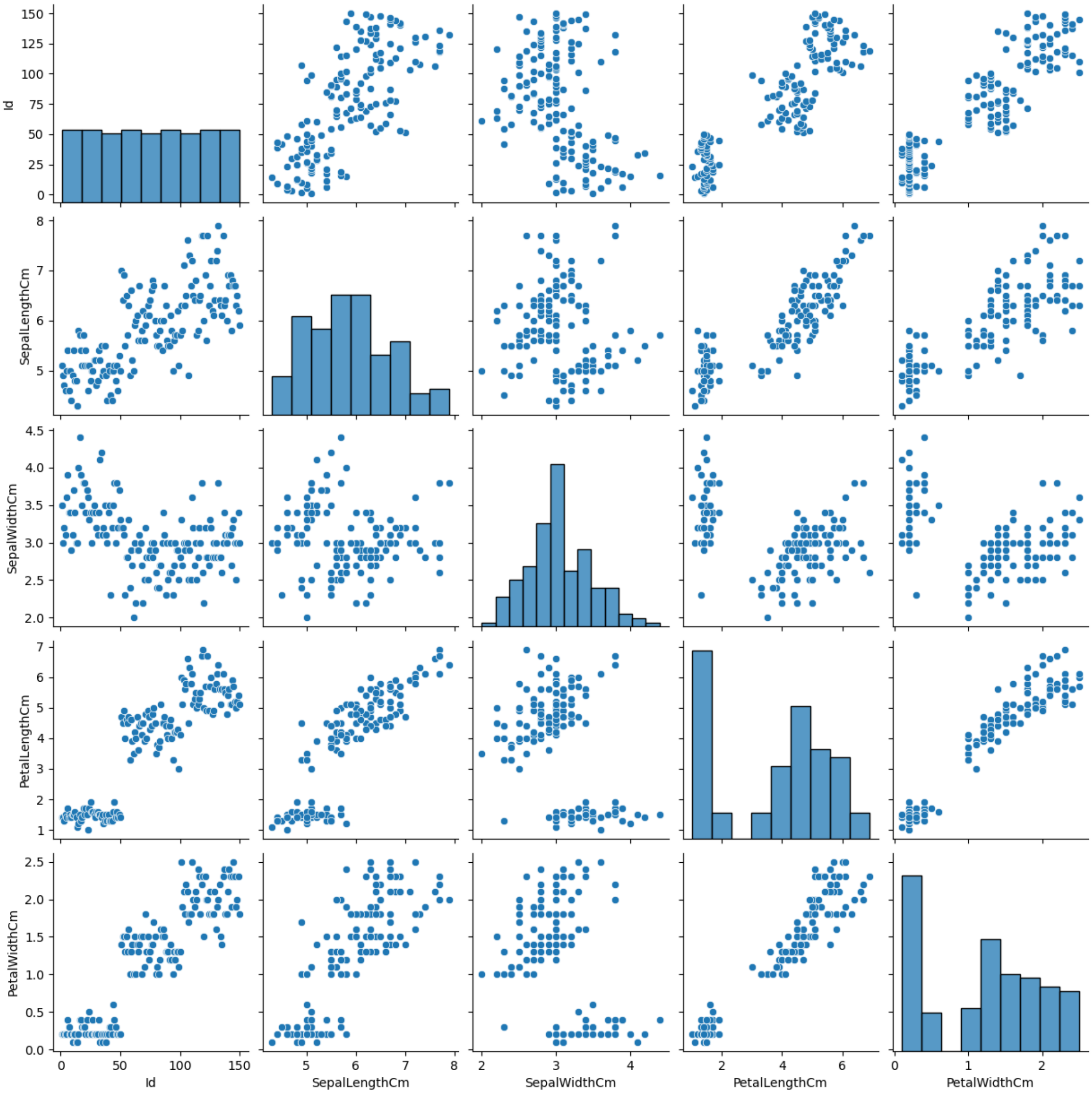
```
### code here
sns.violinplot(x='Species', y='SepalWidthCm', data=df)
plt.show()
```



8. Create a pair chart.

In [30]:

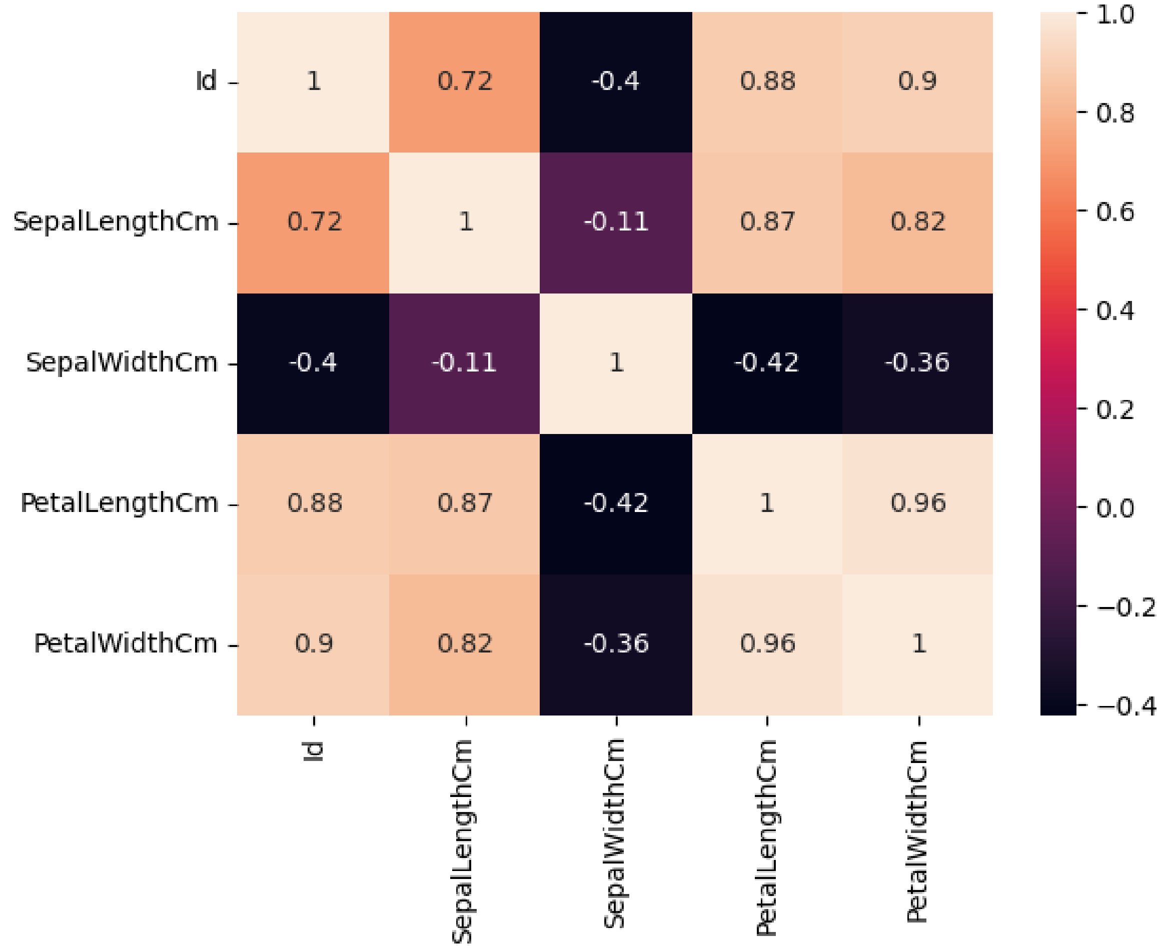
```
### code here
sns.pairplot(df)
plt.show()
```



9. Create a Heat Map

```
In [32]:
```

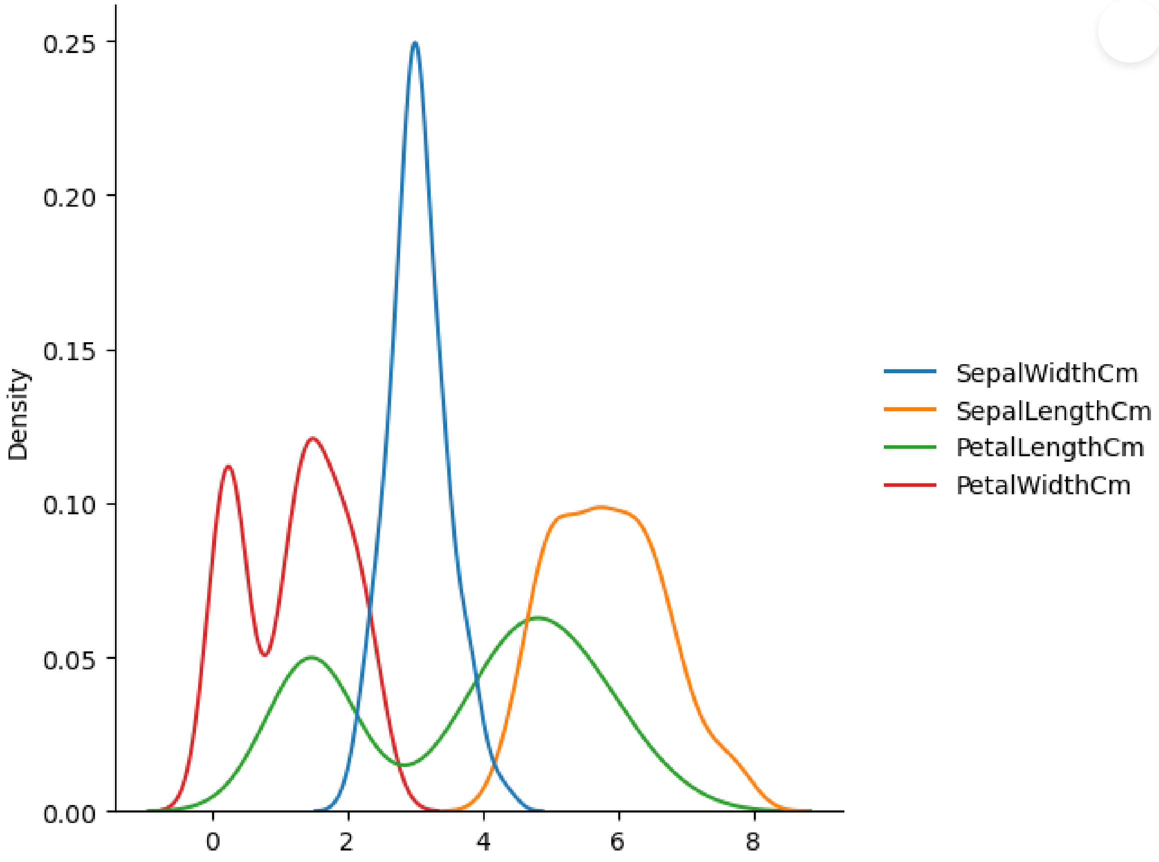
```
### code here
sns.heatmap(data=df.corr(),annot=True)
plt.show()
```



10. Create a distribution plot for all numerical colomns.

In [38]:

```
### code here
sns.displot(df[["SepalWidthCm","SepalLengthCm",'PetalLengthCm','PetalWidthCm']],kind='kde')
plt.show()
```



In [ ]: