In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
data = pd.read_csv("Downloads/IRIS.csv")
data
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

In [4]:
```
data.info()
```

```
<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
```

In [5]:
```
data.isnull().sum()
```

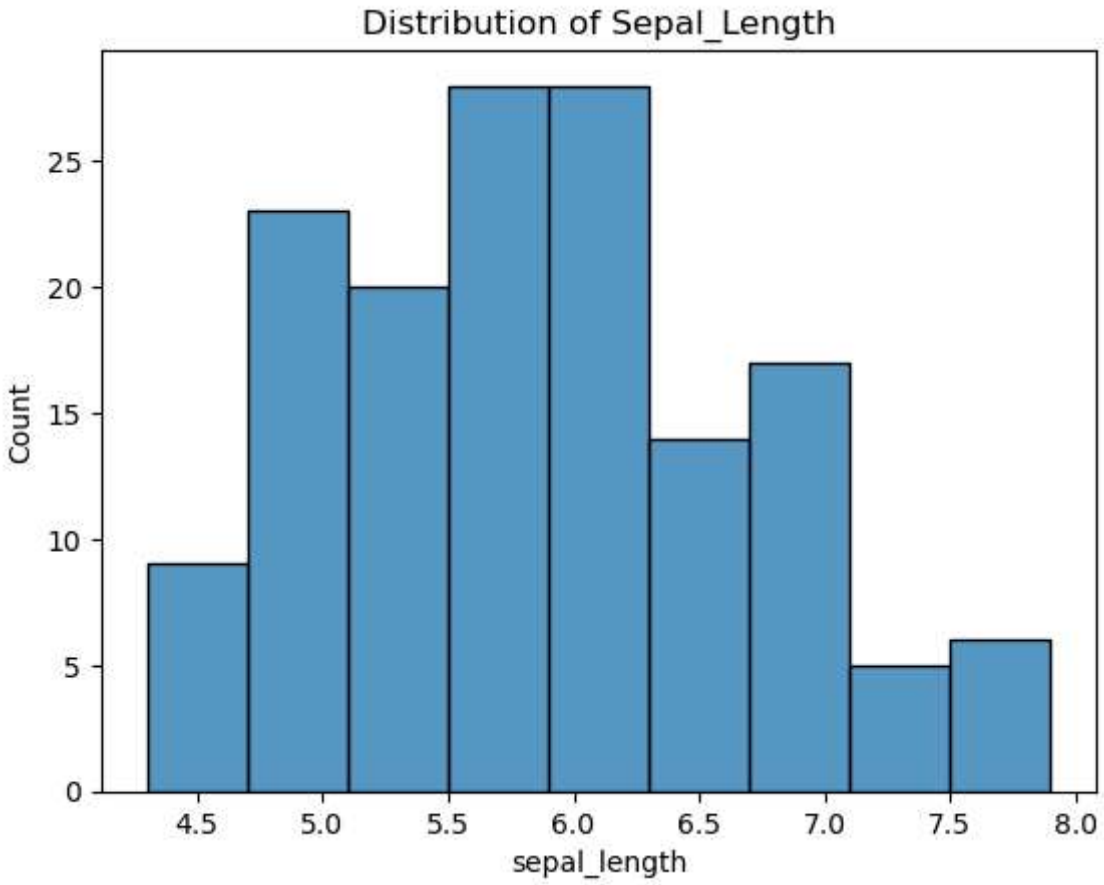
Out[5]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

In [6]:
```
data.describe()
```

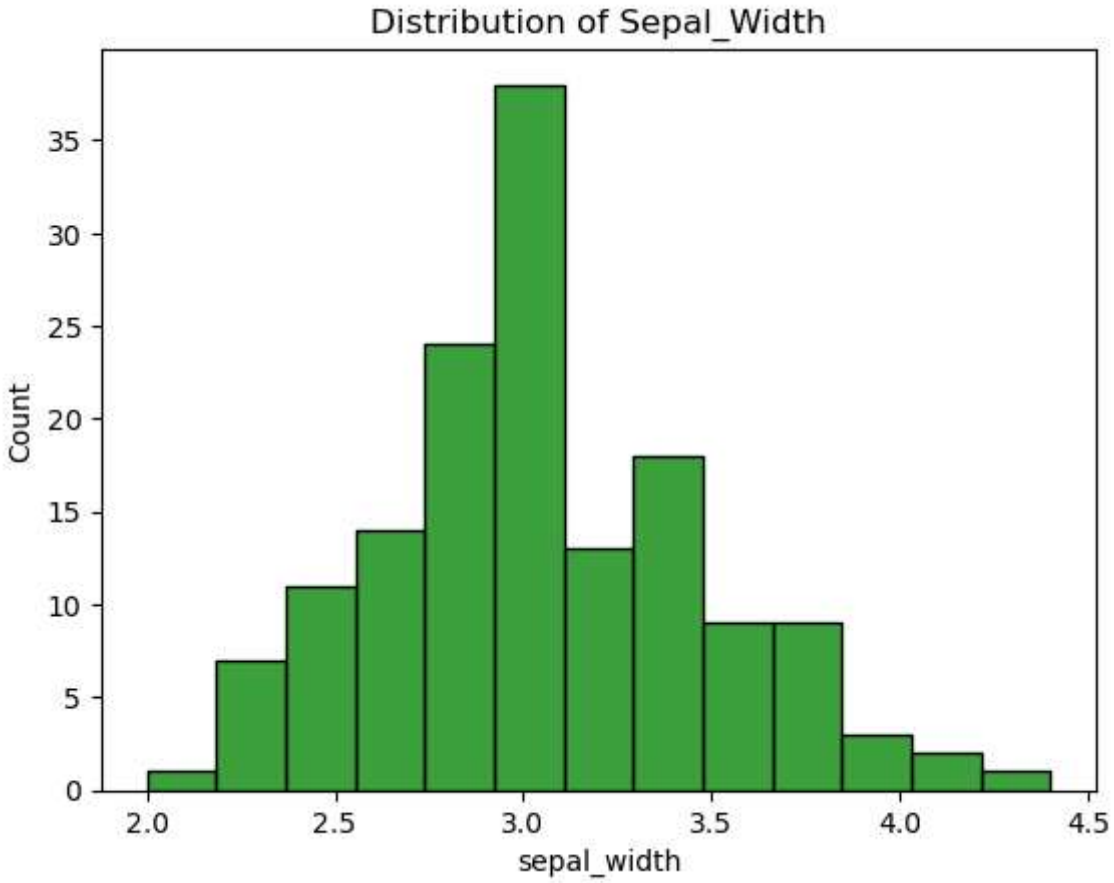
Out[6]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
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

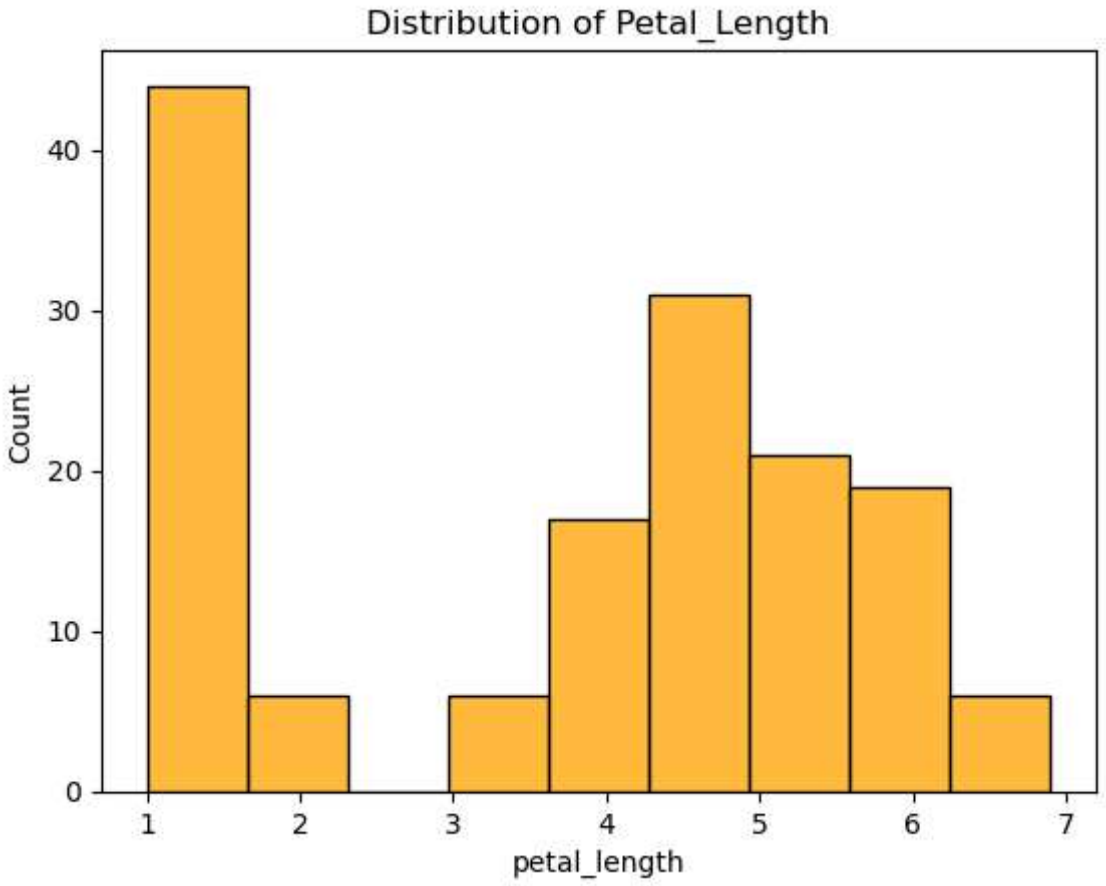
In [7]:
```
sns.histplot(data = data, x= data['sepal_length'])
plt.title("Distribution of Sepal_Length")
plt.show()
```



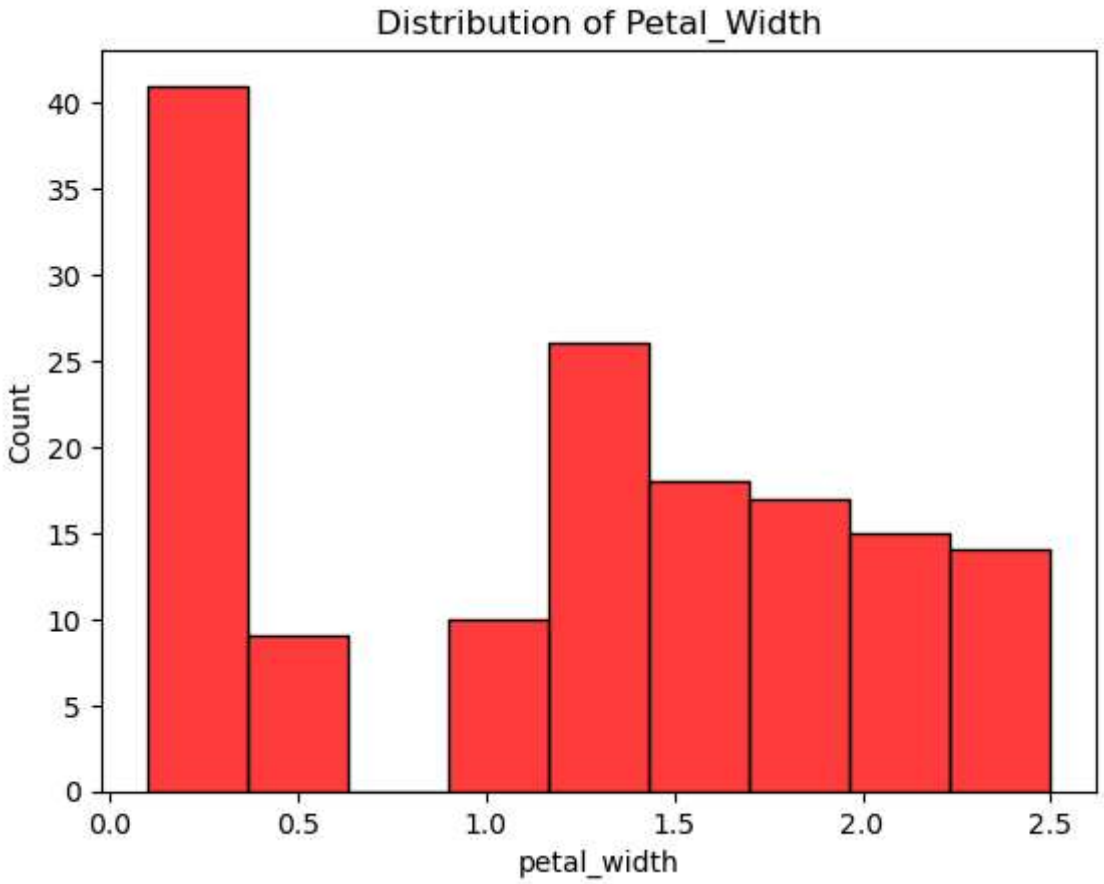
In [8]:
```
sns.histplot(data = data, x = data['sepal_width'], color = 'green')
plt.title("Distribution of Sepal_Width")
plt.show()
```



In [9]:
```
sns.histplot(data = data, x = data['petal_length'], color = 'orange')
plt.title("Distribution of Petal_Length")
plt.show()
```

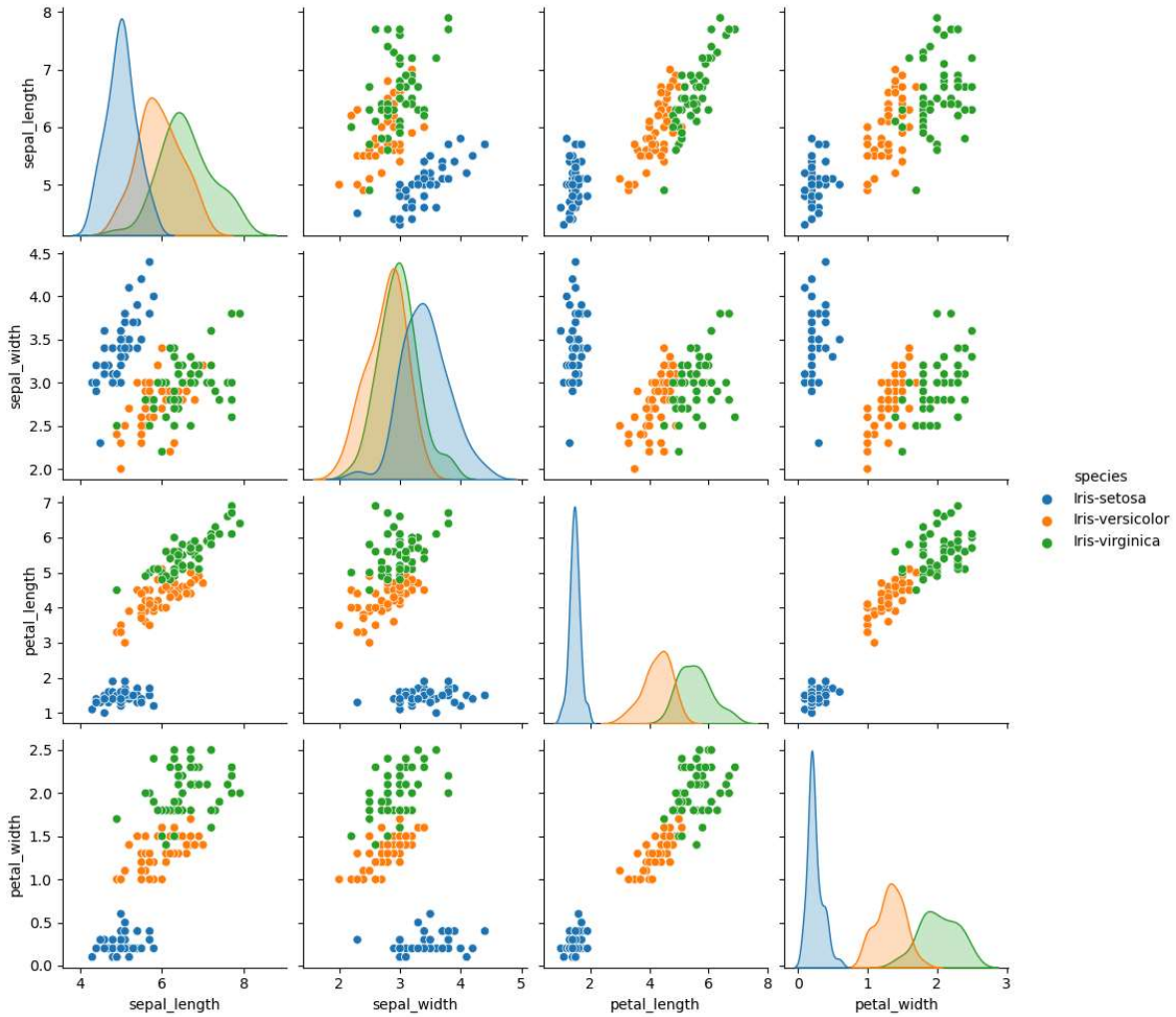


In [10]:
```
sns.histplot(data = data, x = data['petal_width'],color = "red")
plt.title("Distribution of Petal_Width")
plt.show()
```



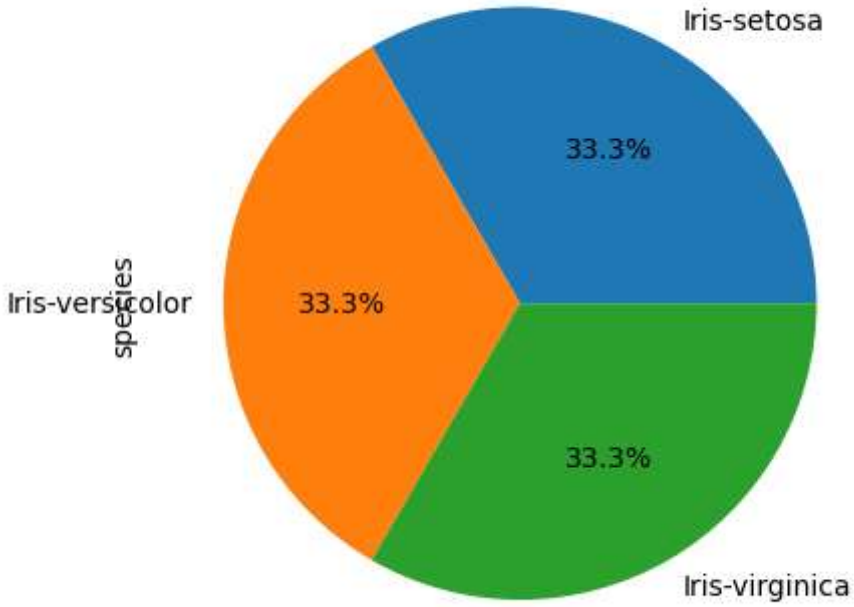
In [11]:
```
sns.pairplot(data, hue = 'species')
```

Out[11]: <seaborn.axisgrid.PairGrid at 0x19be6da3110>



In [12]:
```
data['species'].value_counts().plot(kind = 'pie',autopct='%1.1f%%')
```

Out[12]: <Axes: ylabel='species'>



In [13]:
```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
data['species'] = encoder.fit_transform(data['species'])
data.head()
```

Out[13]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

In [14]:
```
x = data.drop(columns = 'species')
Y = data['species']
```

In [15]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, Y_train, Y_test = train_test_split(x,Y, test_size = 0.3, rand
```

In [16]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, Y_train)
```

Out[16]:
```
LogisticRegression()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [17]:
```
pred = model.predict(x_test)
pred
```

Out[17]:
```
array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0, 2,
       0, 2, 2, 2, 2, 2, 0, 0, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 2, 1, 1, 0,
       0])
```

In [18]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
print("Confusion Matrix : \n",confusion_matrix(Y_test,pred))
print("Accuracy Score :", accuracy_score(Y_test,pred))
```

```
Confusion Matrix :
 [[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
Accuracy Score : 1.0
```

In [19]:
```
from sklearn.neighbors import  KNeighborsClassifier
knn =  KNeighborsClassifier()
knn.fit(x_train, Y_train)
```

Out[19]:
```
KNeighborsClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [20]:
```
knn_pred = knn.predict(x_test)
knn_pred
```

Out[20]:
```
array([1, 0, 2, 1, 1, 0, 1, 2, 1, 1, 2, 0, 0, 0, 0, 1, 2, 1, 1, 2, 0, 2,
       0, 2, 2, 2, 2, 2, 0, 0, 0, 0, 1, 0, 0, 2, 1, 0, 0, 0, 2, 1, 1, 0,
       0])
```

In [21]:
```
print("Confusion Matrix : \n",confusion_matrix(Y_test, knn_pred))
print("Accuracy Score :", accuracy_score(Y_test,knn_pred))
```

```
Confusion Matrix :
 [[19  0  0]
 [ 0 13  0]
 [ 0  0 13]]
Accuracy Score : 1.0
```

In [ ]:
```

```