In [43]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
```

In [44]:

```
iris_flower_file=pd.read_csv("iris flower file.csv")
```

In [45]:

```
iris_flower_file.head(16)
```

Out[45]:

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

In [46]:

```
iris_flower_file.shape
```

Out[46]:

```
(150, 5)
```

In [47]:

```
iris_flower_file.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 [48]:

```
iris_flower_file.describe()
```

Out[48]:

	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 [49]:

```
iris_flower_file.isnull().sum()
```

Out[49]:

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

In [50]:

```
iris_flower_file.describe()
```

Out[50]:

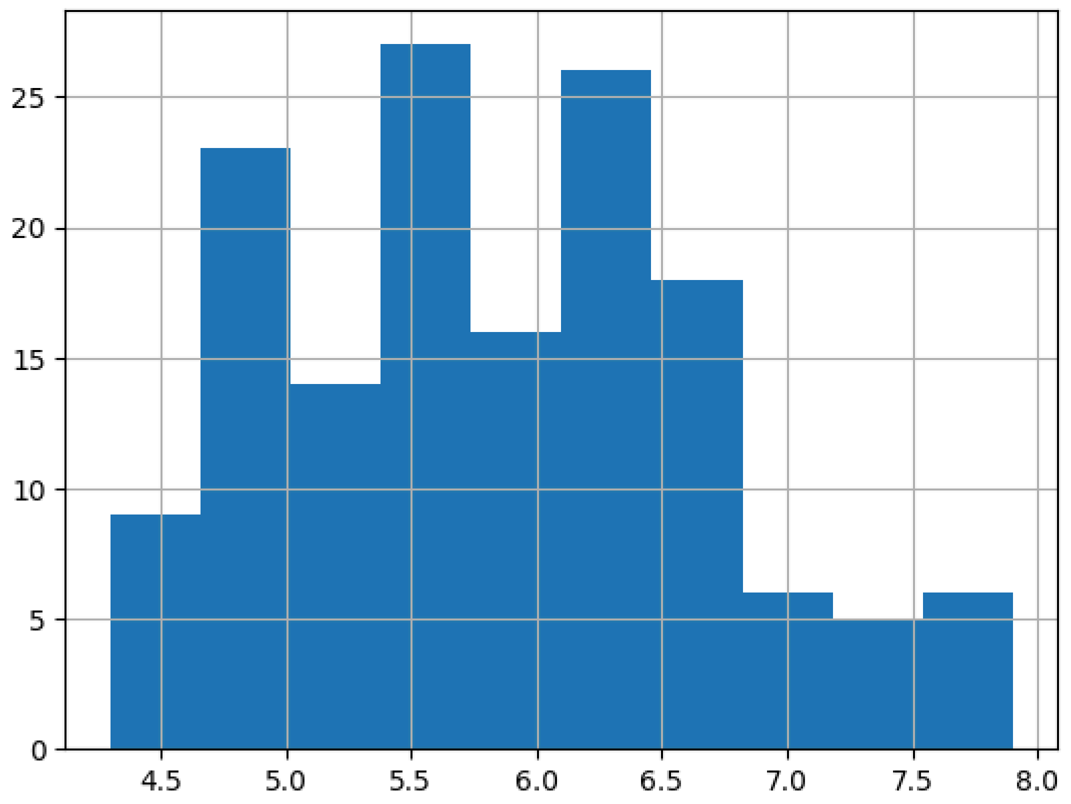
	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

In [54]:

```
iris_flower_file['sepal_length'].hist()
```

Out[54]:

```
<AxesSubplot:>
```

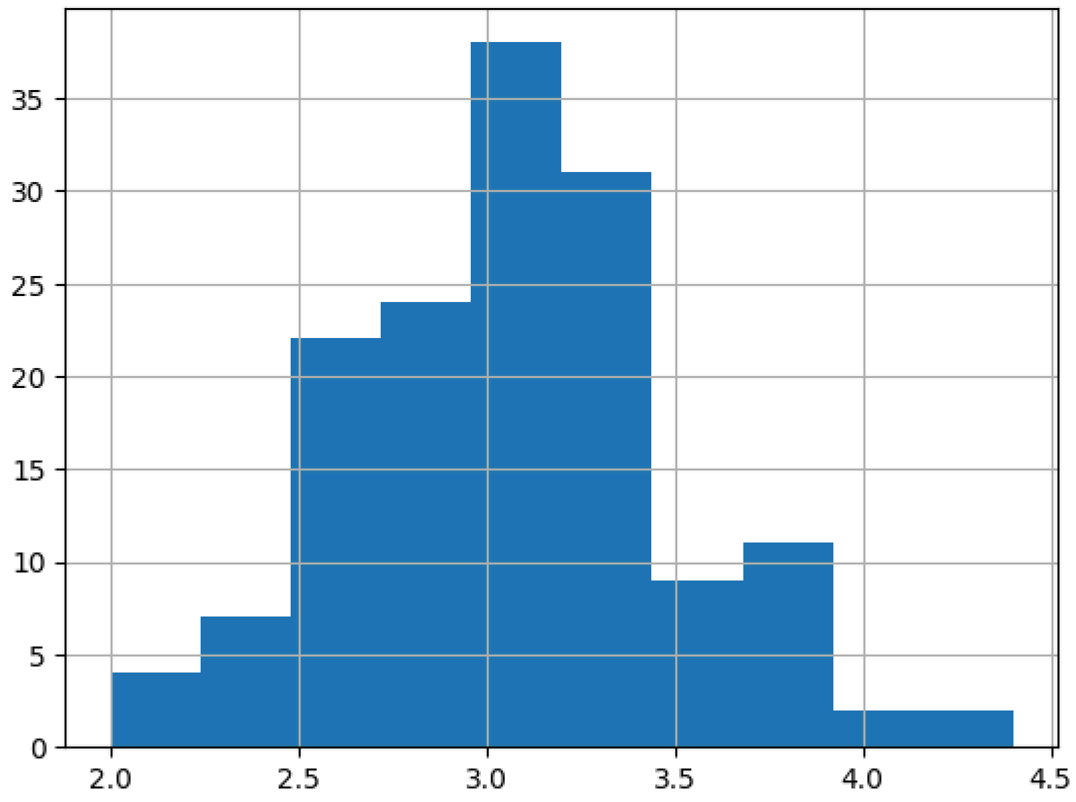


In [57]:

```
iris_flower_file['sepal_width'].hist()
```

Out[57]:

```
<AxesSubplot:>
```

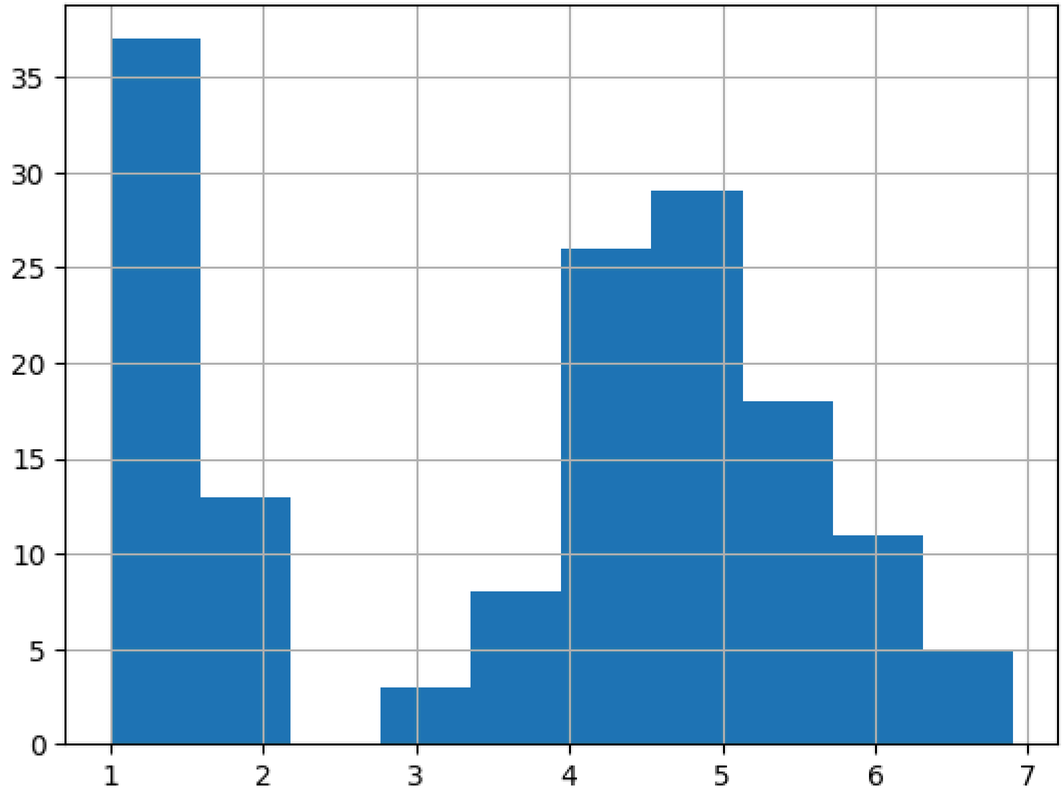


In [61]:

```
iris_flower_file['petal_length'].hist()
```

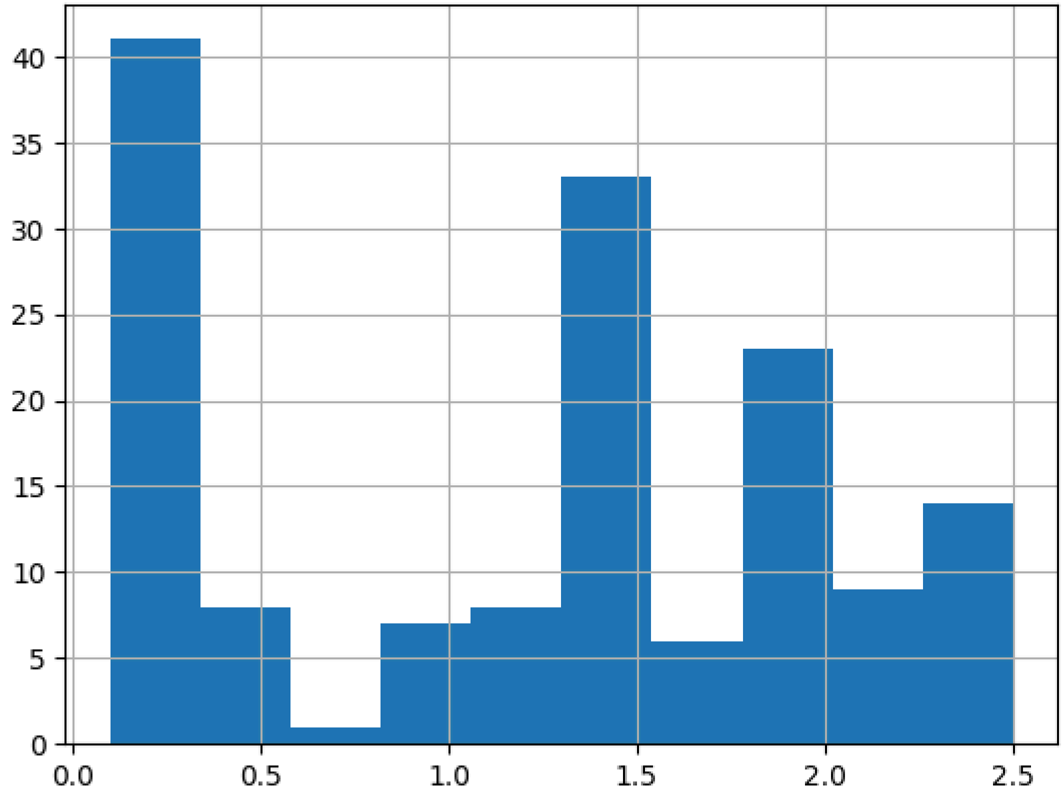
Out[61]:

```
<AxesSubplot:>
```



In [66]:
```
iris_flower_file['petal_width'].hist()
```

Out[66]:
```
<AxesSubplot:>
```



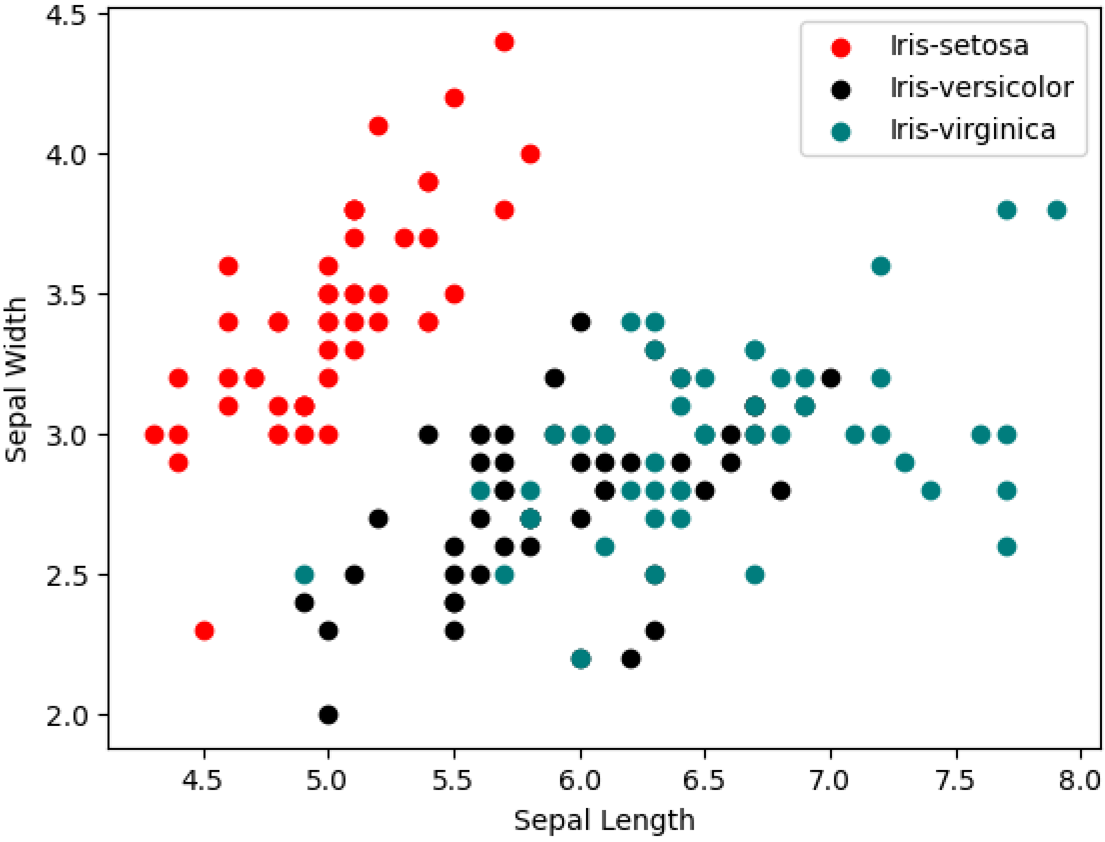
In [67]:
```
colors=['red','Black','teal']
```

In [68]:
```
species=['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
```

In [73]:
```
for i in range(3):
    x=iris_flower_file[iris_flower_file['species']==species[i]]
    plt.scatter(x['sepal_length'],x['sepal_width'],c=colors[i],label=species[i])
plt.xlabel("Sepal Length")
plt.ylabel("Sepal Width")
plt.legend()
```

Out[73]:

```
<matplotlib.legend.Legend at 0x1e26479ff70>
```

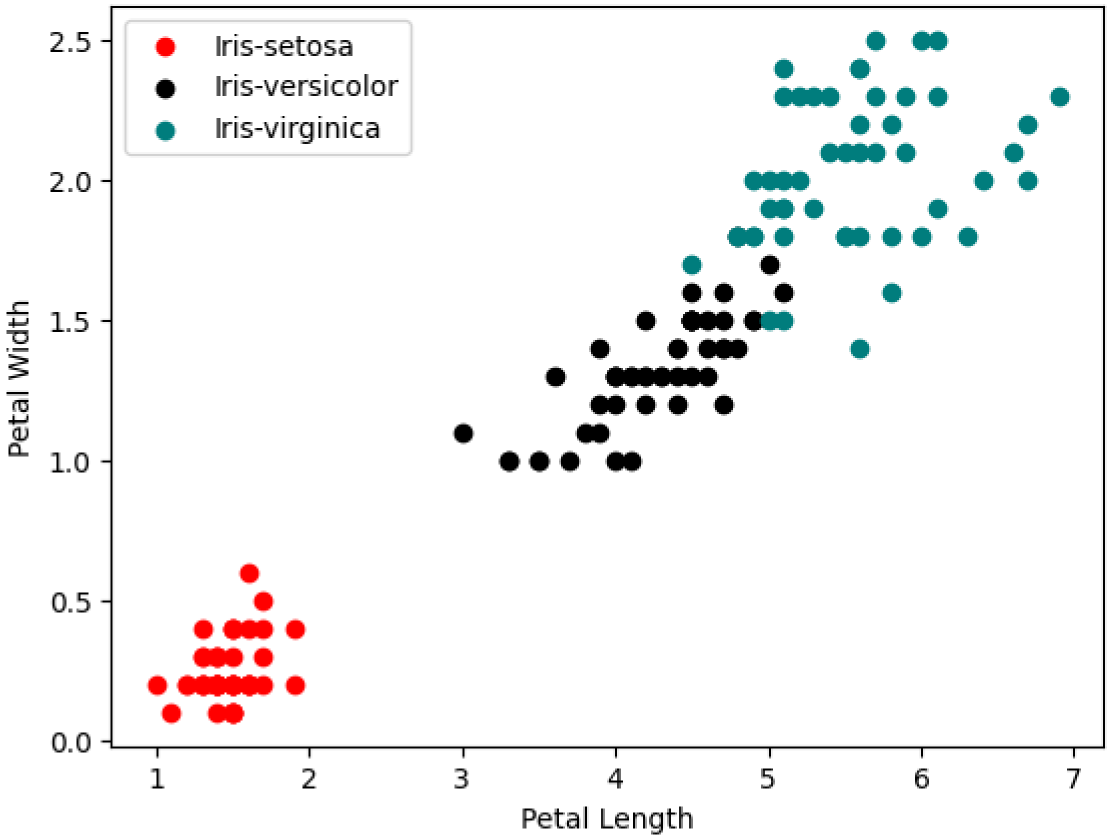


In [76]:

```
for i in range(3):
    x=iris_flower_file[iris_flower_file['species']==species[i]]
    plt.scatter(x['petal_length'],x['petal_width'],c=colors[i],label=species[i])
plt.xlabel("Petal Length")
plt.ylabel("Petal Width")
plt.legend()
```

Out[76]:

```
<matplotlib.legend.Legend at 0x1e264813490>
```

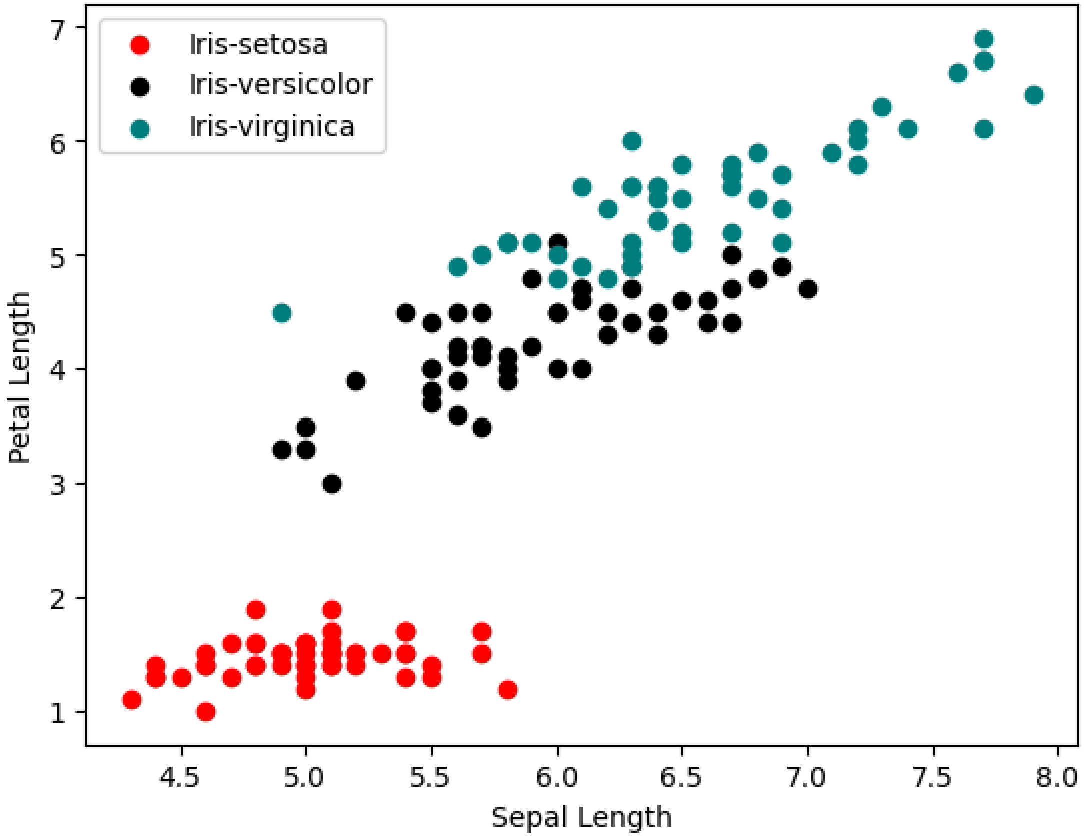


In [78]:

```
for i in range(3):
    x=iris_flower_file[iris_flower_file['species']==species[i]]
    plt.scatter(x['sepal_length'],x['petal_length'],c=colors[i],label=species[i])
plt.xlabel("Sepal Length")
plt.ylabel("Petal Length")
plt.legend()
```

Out[78]:

```
<matplotlib.legend.Legend at 0x1e264731ff0>
```

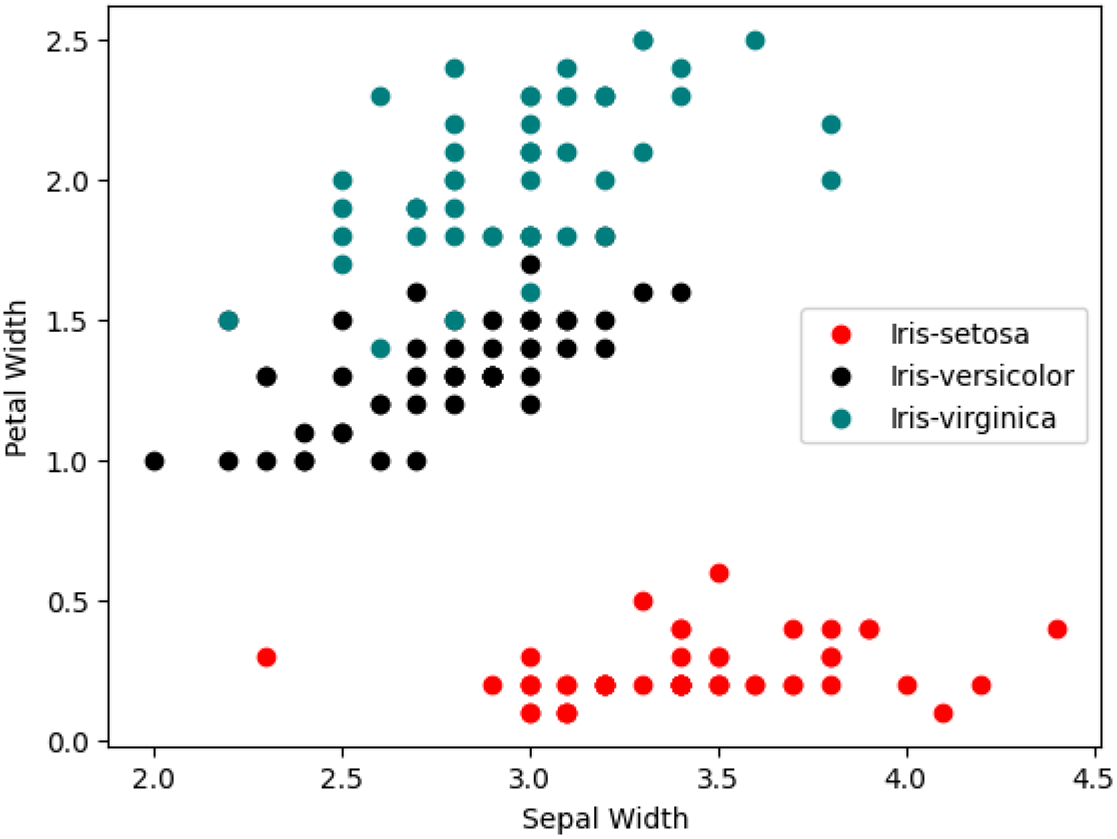


In [80]:

```
for i in range(3):
    x=iris_flower_file[iris_flower_file['species']==species[i]]
    plt.scatter(x['sepal_width'],x['petal_width'],c=colors[i],label=species[i])
plt.xlabel("Sepal Width")
plt.ylabel("Petal Width")
plt.legend()
```

Out[80]:

```
<matplotlib.legend.Legend at 0x1e2649e3eb0>
```

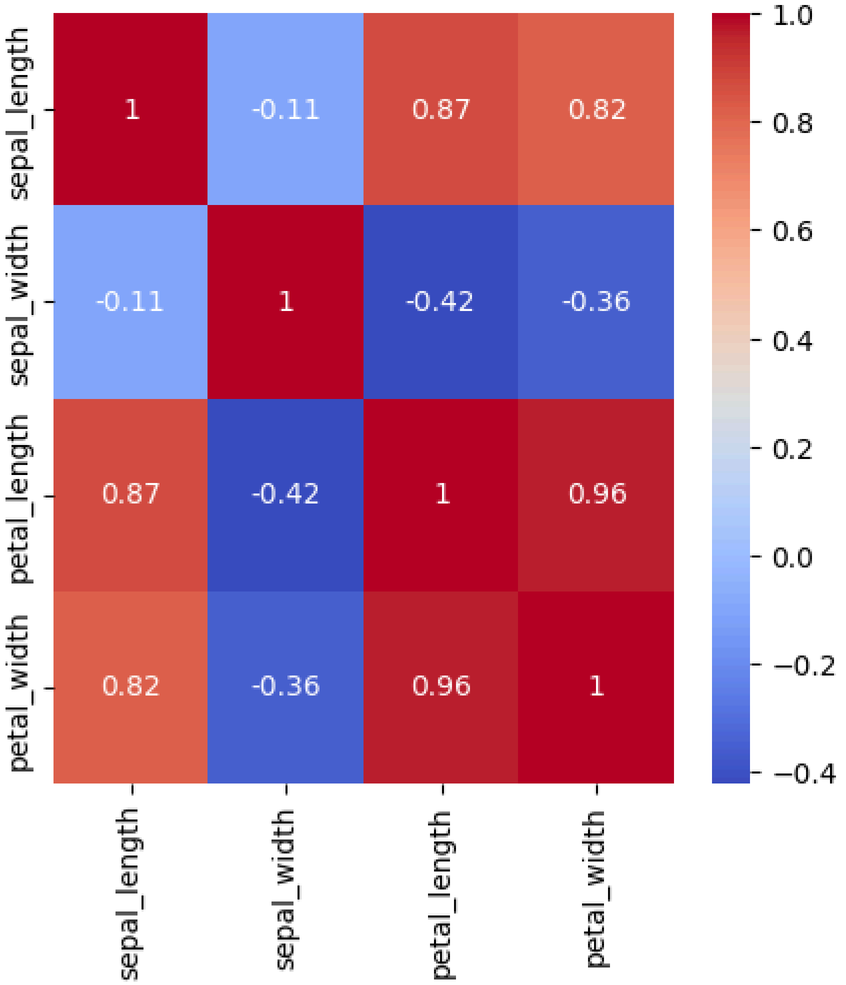


In [83]:

```
numeric_columns=iris_flower_file.drop(columns='species')
corr=numeric_columns.corr()
fig,axis=plt.subplots(figsize=(5,5))
sns.heatmap(corr,annot=True,ax=axis,cmap='coolwarm')
```

Out[83]:

```
<AxesSubplot:>
```



```
In [84]: le=LabelEncoder()
```

```
In [86]: iris_flower_file['species']=le.fit_transform(iris_flower_file['species'])
```

```
In [87]: iris_flower_file.head(16)
```

Out[87]:

	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
5	5.4	3.9	1.7	0.4	0
6	4.6	3.4	1.4	0.3	0
7	5.0	3.4	1.5	0.2	0
8	4.4	2.9	1.4	0.2	0
9	4.9	3.1	1.5	0.1	0
10	5.4	3.7	1.5	0.2	0
11	4.8	3.4	1.6	0.2	0
12	4.8	3.0	1.4	0.1	0
13	4.3	3.0	1.1	0.1	0
14	5.8	4.0	1.2	0.2	0
15	5.7	4.4	1.5	0.4	0

In [88]:

```
x=iris_flower_file.drop(columns='species')
```

In [89]:

```
y=iris_flower_file['species']
```

In [91]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [92]:

```
LR=LogisticRegression()
```

In [93]:

```
LR.fit(x_train,y_train)
```

Out[93]:

```
▾ LogisticRegression
LogisticRegression()
```

In [94]:

```
KNN=KNeighborsClassifier()
```

In [95]:

```
KNN.fit(x_train,y_train)
```

Out[95]:

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

In [96]:

```
DT=DecisionTreeClassifier()
```

In [97]:

```
DT.fit(x_train,y_train)
```

Out[97]:

```
▾ DecisionTreeClassifier

DecisionTreeClassifier()
```

In [98]:

```
LR_accuracy=LR.score(x_test,y_test)*100
KNN_accuracy=KNN.score(x_test,y_test)*100
DT_accuracy=DT.score(x_test,y_test)*100
```

In [99]:

```
print(f"Accuracy by using Logistic Regression: {LR_accuracy}%")
```

```
Accuracy by using Logistic Regression: 95.55555555555556%
```

In [100…

```
print(f"Accuracy by using K Nearest Neighbors Algorithm: {KNN_accuracy}%")
```

```
Accuracy by using K Nearest Neighbors Algorithm: 100.0%
```

In [101…

```
print(f"Accuracy by using Decision Tree Classifier: {DT_accuracy}%")
```

```
Accuracy by using Decision Tree Classifier: 93.33333333333333%
```