In [1]:

```
# Importing Libraries
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

In [2]:

```
data=pd.read_csv('IRIS.csv')
```

In [3]:

```
data.head()
```

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

In [4]:

```
# Giving Proper Heading to Columns
data_header = ['SepalLength','SepalWidth','PetalLength','PetalWidth','Species']
data.to_csv('Iris.csv', header = data_header, index = False)
new_data = pd.read_csv('Iris.csv')
new_data.head()
```

Out[4]:

	SepalLength	SepalWidth	PetalLength	PetalWidth	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

In [5]:

```
# Checking no. of rows and columns
new_data.shape
```

Out[5]:

```
(150, 5)
```

In [6]:

```
# Checking datatypes in dataset
new_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   SepalLength  150 non-null    float64
 1   SepalWidth   150 non-null    float64
 2   PetalLength  150 non-null    float64
 3   PetalWidth   150 non-null    float64
 4   Species      150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

In [7]:

```
# Describing Dataset
new_data.describe()
```

Out[7]:

	SepalLength	SepalWidth	PetalLength	PetalWidth
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 [8]:

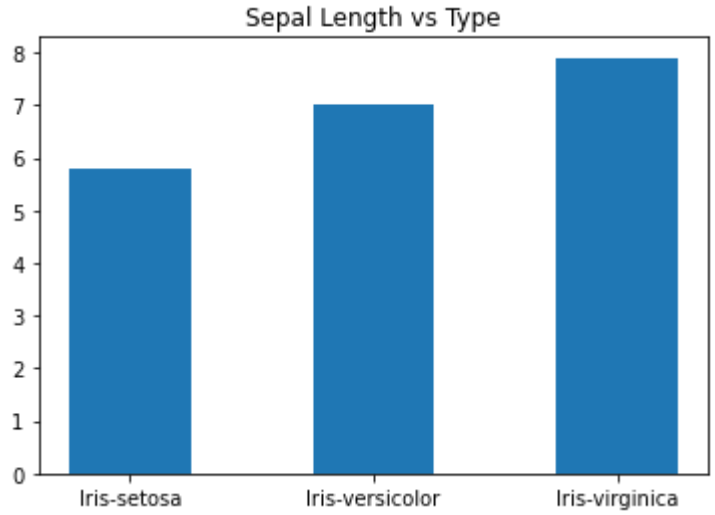
```
# Checking Null Values in DataSet
new_data.isnull().sum()
```

Out[8]:

```
SepalLength    0
SepalWidth     0
PetalLength    0
PetalWidth     0
Species        0
dtype: int64
```

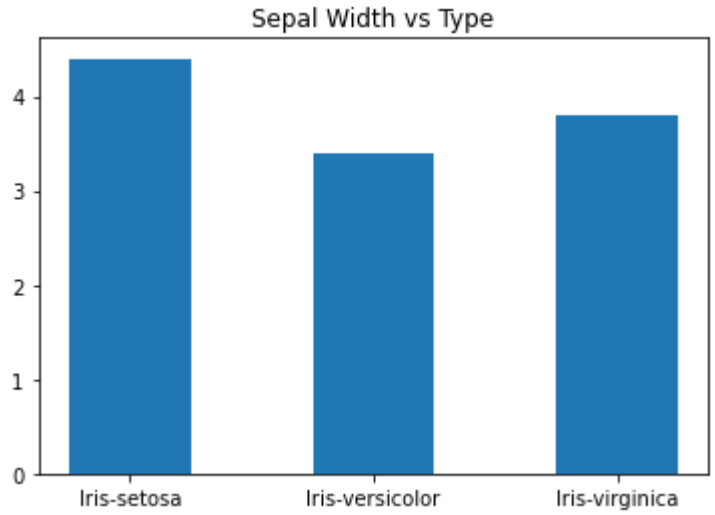
In [9]:

```
# Sepal Length vs Type
plt.bar(new_data['Species'],new_data['SepalLength'], width = 0.5)
plt.title("Sepal Length vs Type")
plt.show()
```



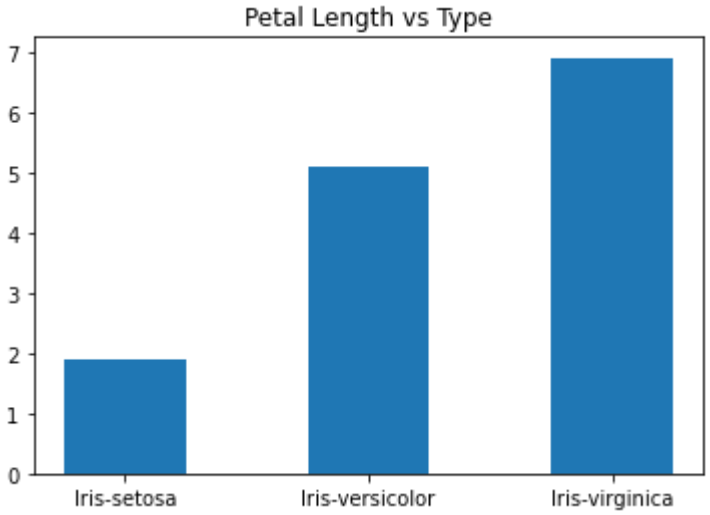
In [10]:

```
# Sepal Width vs Type
plt.bar(new_data['Species'],new_data['SepalWidth'], width = 0.5)
plt.title("Sepal Width vs Type")
plt.show()
```



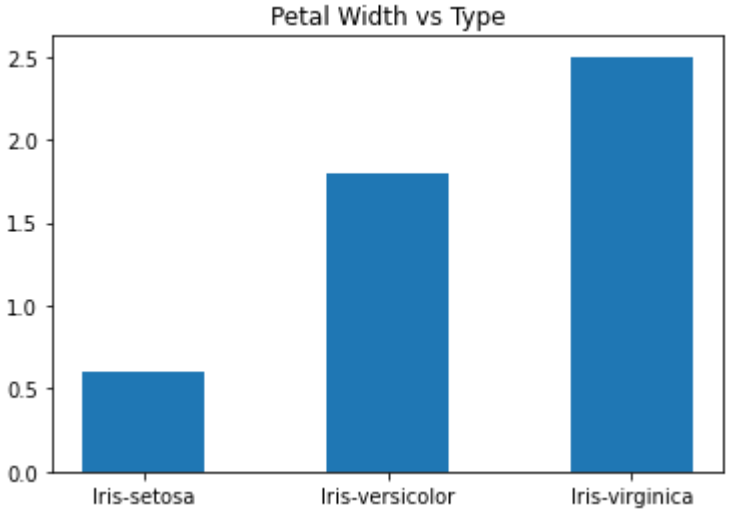
In [11]:

```
# Petal Length vs Type
plt.bar(new_data['Species'],new_data['PetalLength'], width = 0.5)
plt.title("Petal Length vs Type")
plt.show()
```



In [12]:

```
# Petal Width vs Type
plt.bar(new_data['Species'],new_data['PetalWidth'], width = 0.5)
plt.title("Petal Width vs Type")
plt.show()
```

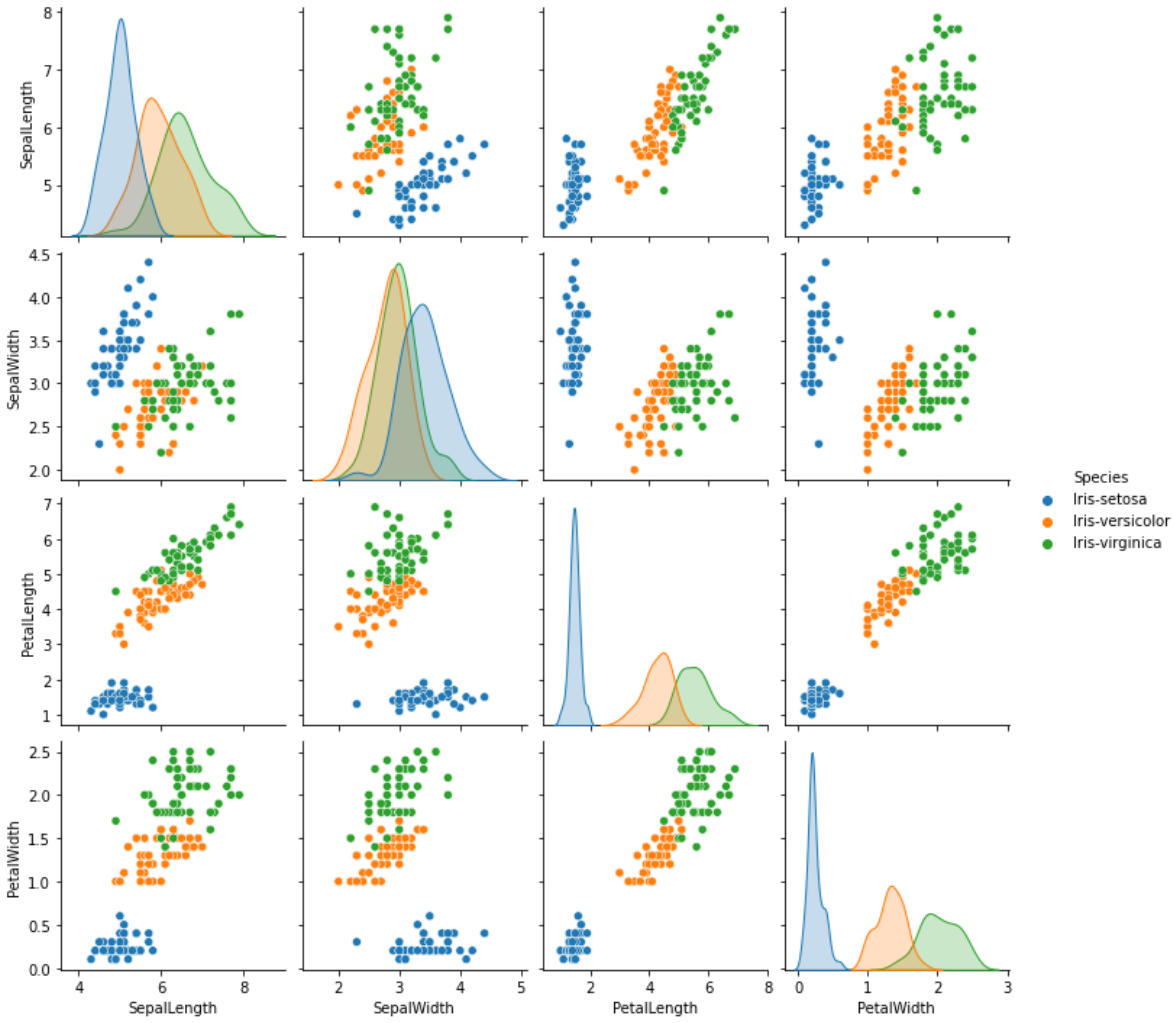


In [13]:

```
sns.pairplot(new_data,hue='Species')
```

Out[13]:

```
<seaborn.axisgrid.PairGrid at 0x16fc85bfb50>
```



In [14]:

```
x = new_data.drop(columns="Species")
y = new_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 = 0.4, random_state = 1
```

In [16]:

```
x_train.head()
```

Out[16]:

	SepalLength	SepalWidth	PetalLength	PetalWidth
11	4.8	3.4	1.6	0.2
113	5.7	2.5	5.0	2.0
123	6.3	2.7	4.9	1.8
12	4.8	3.0	1.4	0.1
2	4.7	3.2	1.3	0.2

In [17]:

```
x_test.head()
```

Out[17]:

	SepalLength	SepalWidth	PetalLength	PetalWidth
14	5.8	4.0	1.2	0.2
98	5.1	2.5	3.0	1.1
75	6.6	3.0	4.4	1.4
16	5.4	3.9	1.3	0.4
131	7.9	3.8	6.4	2.0

In [18]:

```
y_train.head()
```

Out[18]:

```
11        Iris-setosa
113    Iris-virginica
123    Iris-virginica
12        Iris-setosa
2         Iris-setosa
Name: Species, dtype: object
```

In [19]:

```
print("x_train: ", len(x_train))
print("x_test: ", len(x_test))
print("y_train: ", len(y_train))
print("y_test: ", len(y_test))
```

```
x_train:  90
x_test:  60
y_train:  90
y_test:  60
```

In [20]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, roc_curve, roc_auc_score
```

In [21]:

```
model = LogisticRegression()
model.fit(x_train, y_train)
```

Out[21]:

```
▾ LogisticRegression
LogisticRegression()
```

In [22]:

```
predict = model.predict(x_test)
print("Pridicted values on Test Data", predict)
```

```
Pridicted values on Test Data ['Iris-setosa' 'Iris-versicolor' 'Iris-versico
lor' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa'
 'Iris-setosa' 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-versicolor' 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa'
 'Iris-virginica' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-virginica' 'Iris-setosa'
 'Iris-versicolor' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-virginica' 'Iris-setosa' 'Iris-versicolor' 'Iris-virginica'
 'Iris-versicolor' 'Iris-virginica' 'Iris-setosa' 'Iris-setosa'
 'Iris-setosa' 'Iris-versicolor' 'Iris-setosa' 'Iris-setosa'
 'Iris-virginica' 'Iris-virginica' 'Iris-virginica' 'Iris-virginica'
 'Iris-virginica' 'Iris-versicolor' 'Iris-virginica' 'Iris-versicolor']
```

In [23]:

```
y_test_pred = model.predict(x_test)
y_train_pred = model.predict(x_train)
```

In [24]:

```
print("Training Accuracy : ", accuracy_score(y_train, y_train_pred))
print("Test Accuracy : ", accuracy_score(y_test, y_test_pred))
```

```
Training Accuracy :  0.9777777777777777
Test Accuracy :  0.9666666666666667
```

In [ ]:

```

```