Importing Libraries

In [1]:
```
import warnings
warnings.filterwarnings("ignore")

import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
```

Loading Dataset

In [2]:
```
iris = pd.read_csv('iris.csv')
iris
```

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

Data Exploration

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
 5   Species        150 non-null    object 
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

Features Extraction

In [4]:
```
features = iris[['Id','SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWic
```

In [5]:
```
features
```

Out[5]:

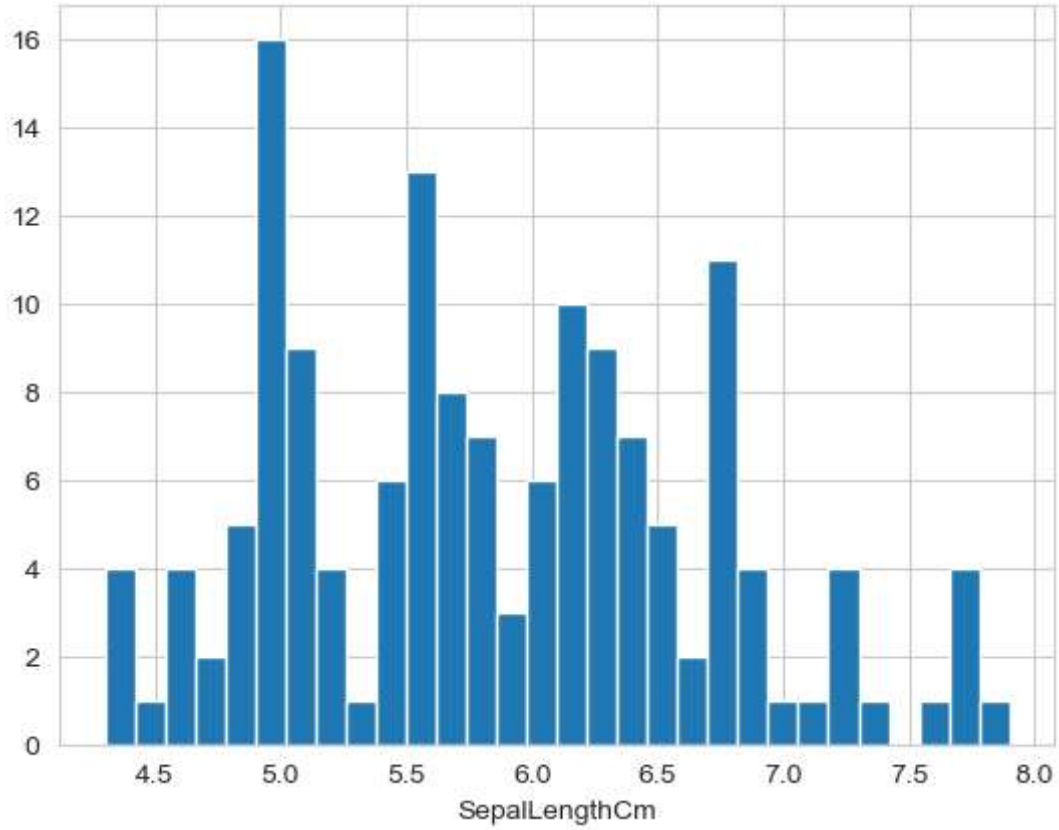
	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	1	5.1	3.5	1.4	0.2
1	2	4.9	3.0	1.4	0.2
2	3	4.7	3.2	1.3	0.2
3	4	4.6	3.1	1.5	0.2
4	5	5.0	3.6	1.4	0.2
...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3
146	147	6.3	2.5	5.0	1.9
147	148	6.5	3.0	5.2	2.0
148	149	6.2	3.4	5.4	2.3
149	150	5.9	3.0	5.1	1.8

150 rows × 5 columns

Data Visualization

In [6]:
```
sns.set_style('whitegrid')
iris['SepalLengthCm'].hist(bins=30)
plt.xlabel('SepalLengthCm')
```

Out[6]:
```
Text(0.5, 0, 'SepalLengthCm')
```

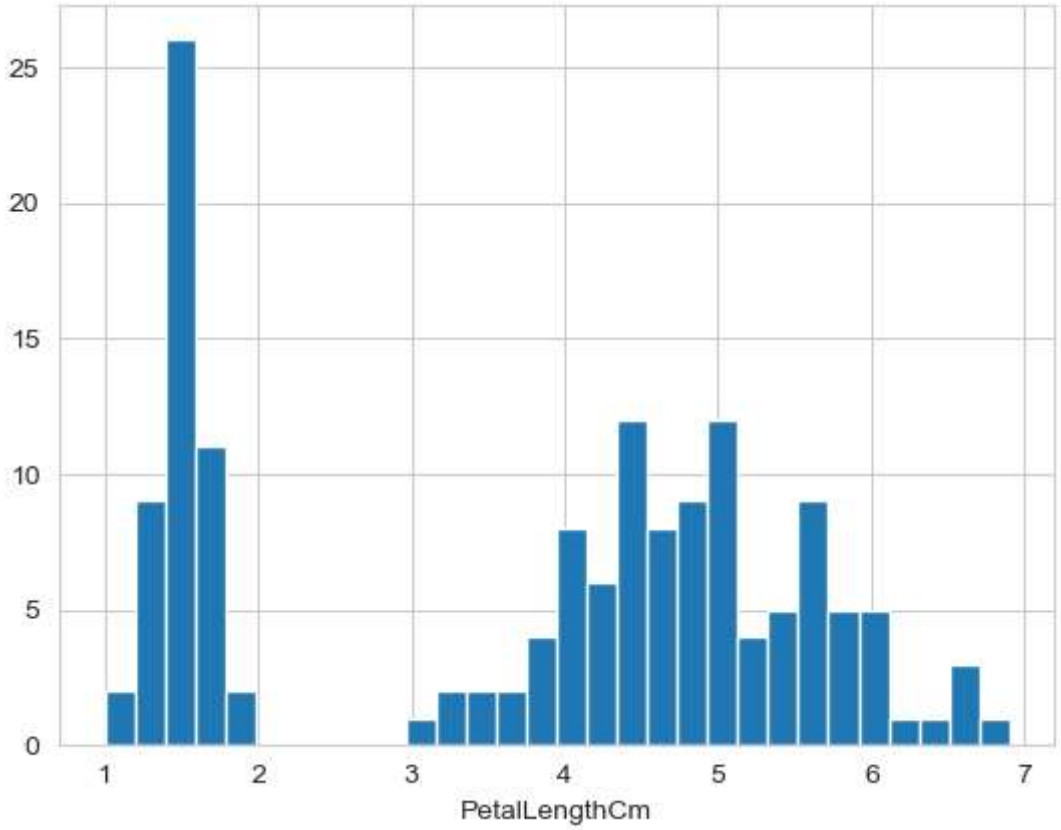


In [7]:
```
sns.set_style('whitegrid')
iris['SepalWidthCm'].hist(bins=30)
plt.xlabel('SepalWidthCm')
```

...

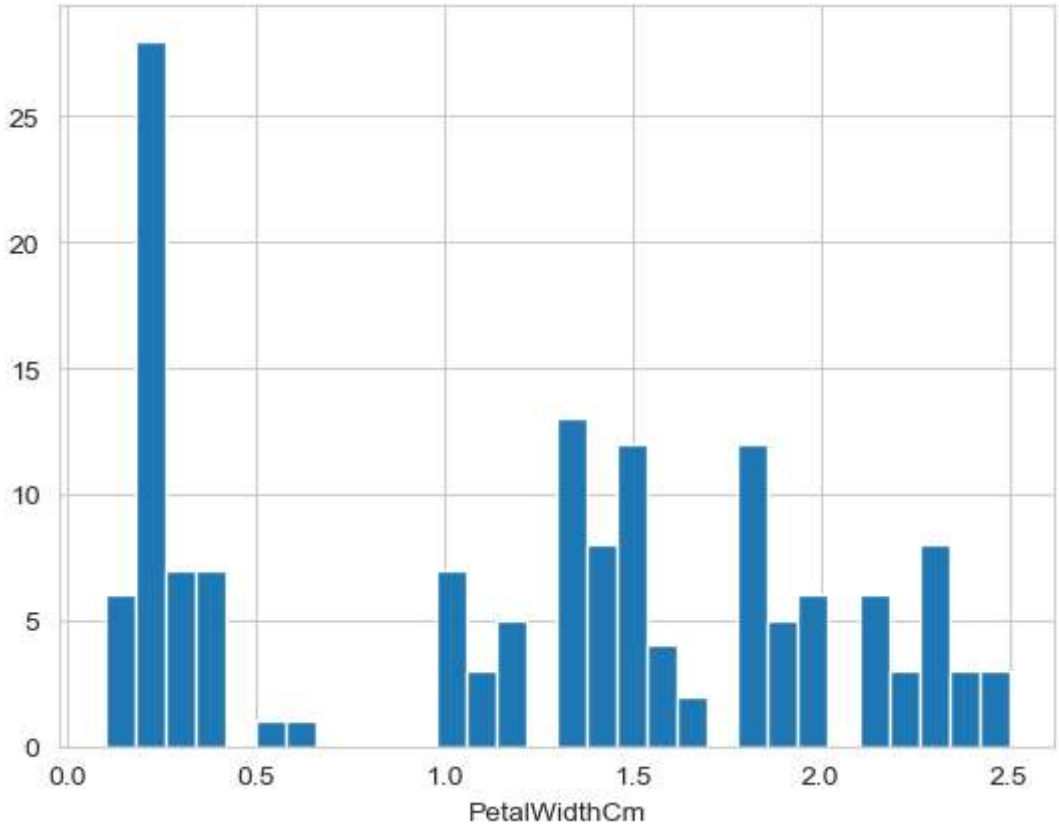
In [8]:
```
sns.set_style('whitegrid')
iris['PetalLengthCm'].hist(bins=30)
plt.xlabel('PetalLengthCm')
```

Out[8]:
```
Text(0.5, 0, 'PetalLengthCm')
```



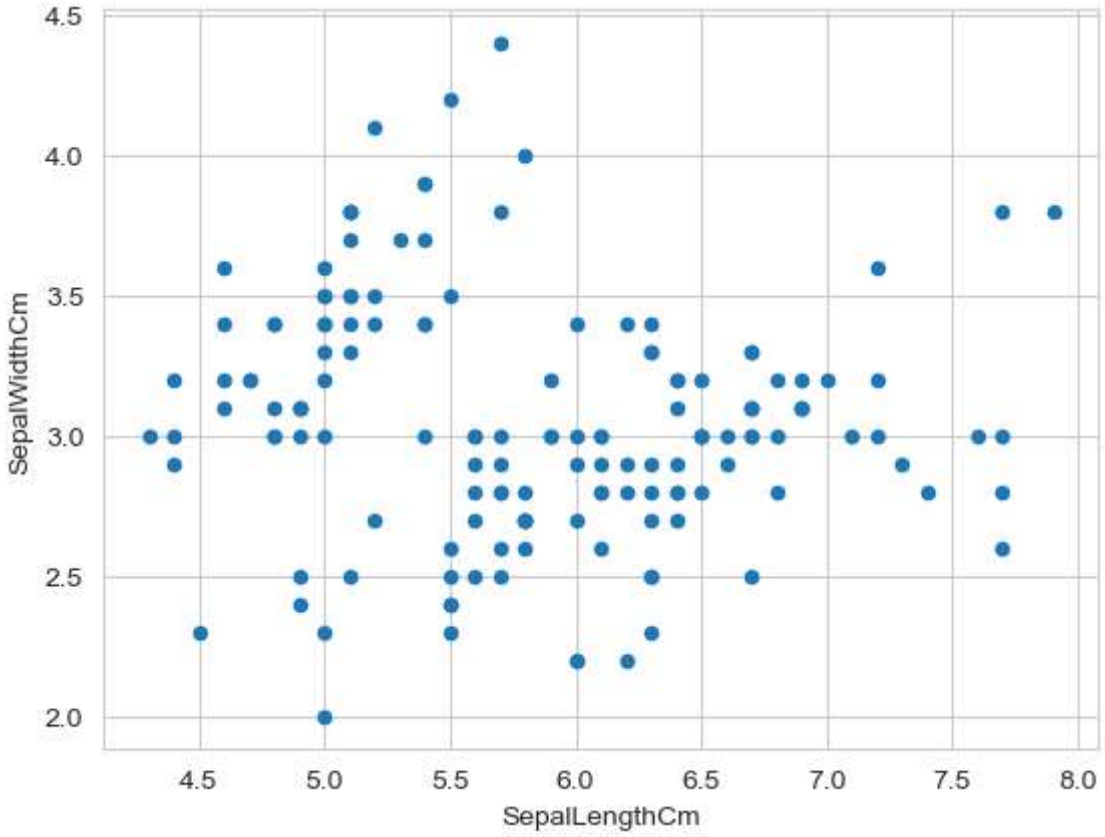
In [9]:
```
sns.set_style('whitegrid')
iris['PetalWidthCm'].hist(bins=30)
plt.xlabel('PetalWidthCm')
```

Out[9]:
```
Text(0.5, 0, 'PetalWidthCm')
```



In [10]:
```
iris.plot.scatter(x = "SepalLengthCm", y = "SepalWidthCm")
```

Out[10]: <Axes: xlabel='SepalLengthCm', ylabel='SepalWidthCm'>



Spliting data for Training & Testing

In [11]:
```
x=features
y=iris['Species']
```

In [12]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33)
```

Model Building By Logistic Regression Algo

In [13]:
```
reg = LogisticRegression()
```

In [14]:
```
reg.fit(x_train,y_train)
```

Out[14]:
```
LogisticRegression
LogisticRegression()
```

Classification Report

In [15]:
```
prediction = reg.predict(x_test)
```

In [16]:
```
print(classification_report(y_test,prediction))
```

```
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        15
Iris-versicolor       1.00      1.00      1.00        21
 Iris-virginica       1.00      1.00      1.00        14

       accuracy                           1.00        50
      macro avg       1.00      1.00      1.00        50
   weighted avg       1.00      1.00      1.00        50
```