Implement Decision Tree Algorithm

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

In [4]:
```
from sklearn.datasets import load_iris
```

In [6]:
```
iris=load_iris()
```

In [8]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(iris.data,iris.target,random_state=100,stratify=iris.target,test_
```

In [10]:
```
from sklearn.metrics import accuracy_score
```

In [12]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [14]:
```
classifier=DecisionTreeClassifier()
```

In [16]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [18]:
```
classifier=DecisionTreeClassifier()
```

In [20]:
```
classifier.fit(x_train,y_train)
```

Out[20]:
```
DecisionTreeClassifier
DecisionTreeClassifier()
```

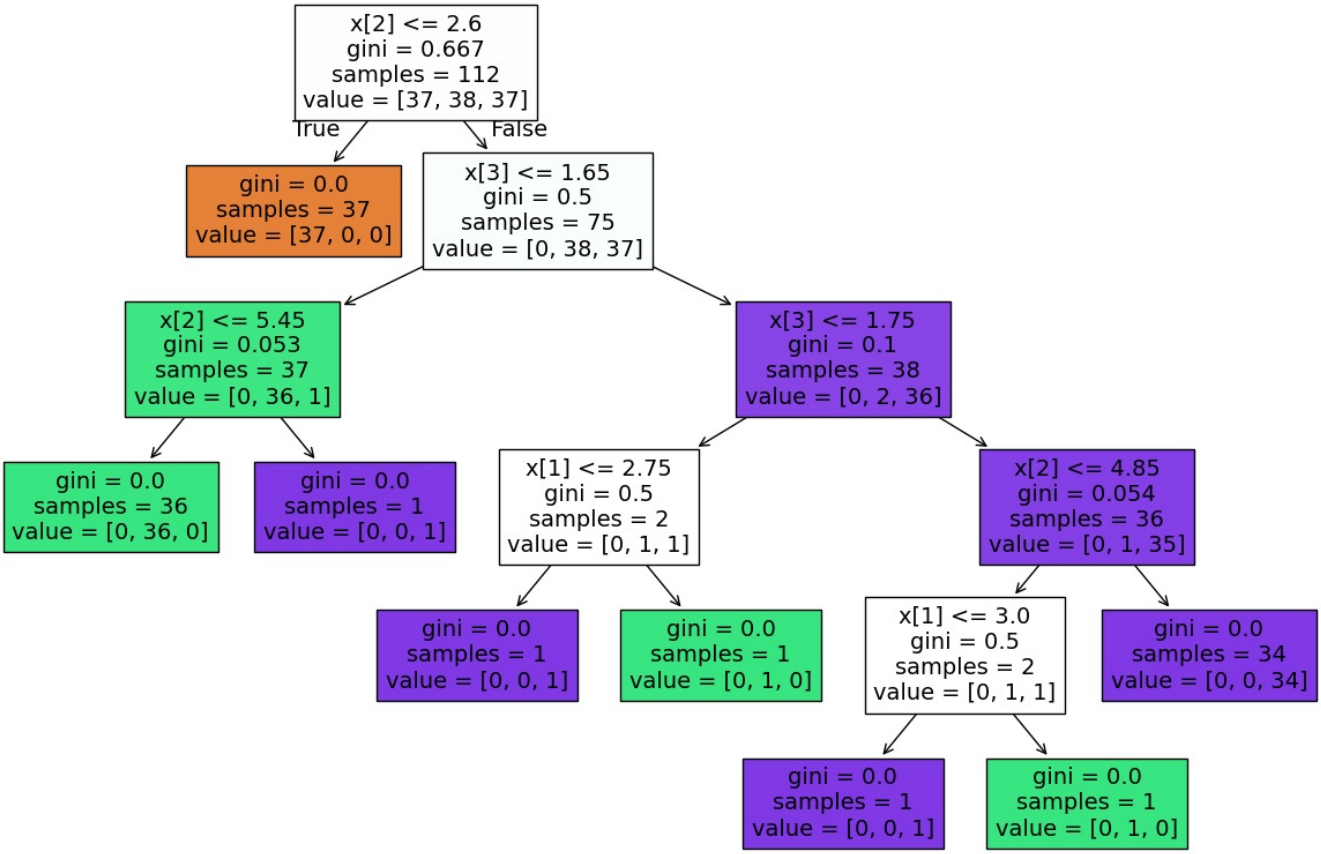
In [22]:
```
pred=classifier.predict(x_test)
accuracy_score(y_test,pred)
```

Out[22]:
```
0.9473684210526315
```

In [24]:
```
from sklearn import tree
plt.figure(figsize=(15,10))
tree.plot_tree(classifier,filled=True)
```

Out[24]:
```
[Text(0.3181818181818182, 0.9166666666666666, 'x[2] <= 2.6\ngini = 0.667\nsamples = 112\nvalue = [37, 38, 37]')
,
 Text(0.22727272727272727, 0.75, 'gini = 0.0\nsamples = 37\nvalue = [37, 0, 0]'),
 Text(0.2727272727272727, 0.8333333333333333, 'True  '),
 Text(0.4090909090909091, 0.75, 'x[3] <= 1.65\ngini = 0.5\nsamples = 75\nvalue = [0, 38, 37]'),
 Text(0.36363636363636365, 0.8333333333333333, '  False'),
 Text(0.18181818181818182, 0.5833333333333334, 'x[2] <= 5.45\ngini = 0.053\nsamples = 37\nvalue = [0, 36, 1]'),
 Text(0.09090909090909091, 0.4166666666666667, 'gini = 0.0\nsamples = 36\nvalue = [0, 36, 0]'),
 Text(0.2727272727272727, 0.4166666666666667, 'gini = 0.0\nsamples = 1\nvalue = [0, 0, 1]'),
 Text(0.6363636363636364, 0.5833333333333334, 'x[3] <= 1.75\ngini = 0.1\nsamples = 38\nvalue = [0, 2, 36]'),
 Text(0.45454545454545453, 0.4166666666666667, 'x[1] <= 2.75\ngini = 0.5\nsamples = 2\nvalue = [0, 1, 1]'),
 Text(0.36363636363636365, 0.25, 'gini = 0.0\nsamples = 1\nvalue = [0, 0, 1]'),
 Text(0.5454545454545454, 0.25, 'gini = 0.0\nsamples = 1\nvalue = [0, 1, 0]'),
 Text(0.8181818181818182, 0.4166666666666667, 'x[2] <= 4.85\ngini = 0.054\nsamples = 36\nvalue = [0, 1, 35]'),
 Text(0.7272727272727273, 0.25, 'x[1] <= 3.0\ngini = 0.5\nsamples = 2\nvalue = [0, 1, 1]'),
 Text(0.6363636363636364, 0.08333333333333333, 'gini = 0.0\nsamples = 1\nvalue = [0, 0, 1]'),
 Text(0.8181818181818182, 0.08333333333333333, 'gini = 0.0\nsamples = 1\nvalue = [0, 1, 0]'),
 Text(0.9090909090909091, 0.25, 'gini = 0.0\nsamples = 34\nvalue = [0, 0, 34]')]
```

In [26]:
```
plt.show()
```



Conclusion : Decision tree learned how to identify iris flowers based on their features like petal length and petal width.Most of leaf node are pure and model got up to 92% accuracy