```
#IRIS FLOWER CLASSIFICATION

#IMPORTING REQUIRED LIBRARIES

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

#LOADING THE DATASET

d=pd.read_csv("/content/Iris.csv")

d.head()
```

	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

```
d.tail()
```

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
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

```
d=d.drop(columns =["Id"])
d.head()
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	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

```
#TO KNOW THE STATS ABOUT DATA
d.describe()
```

```
d.shape
```

```
(150, 5)
```

```
#CHECKING FOR NULL VALUES
```

```
d.isnull().sum()
```

```
SepalLengthCm    0
SepalWidthCm     0
PetalLengthCm    0
PetalWidthCm     0
Species          0
dtype: int64
```

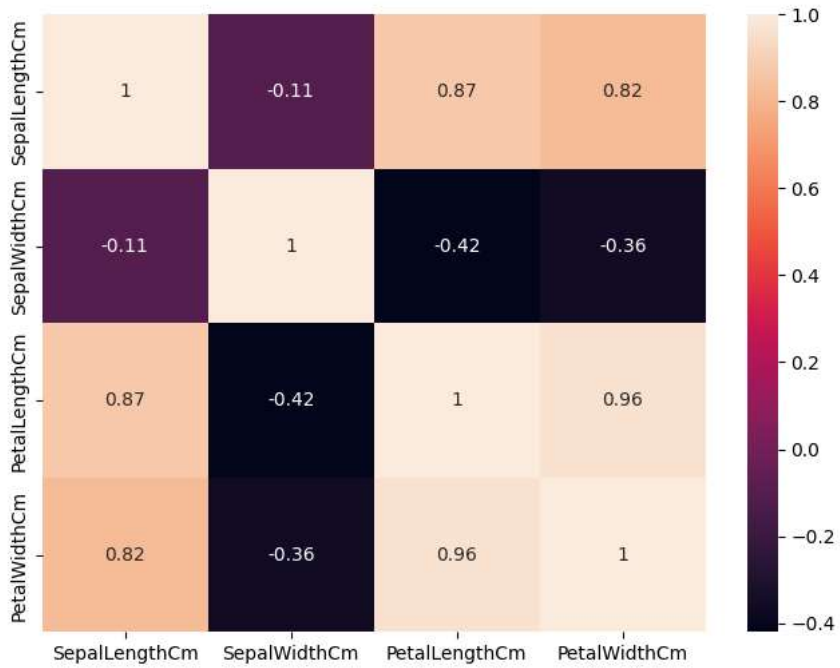
```
#CORRELATION MATRIX
```

```
d.corr()
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
SepalLengthCm	1.000000	-0.109369	0.871754	0.817954
SepalWidthCm	-0.109369	1.000000	-0.420516	-0.356544
PetalLengthCm	0.871754	-0.420516	1.000000	0.962757
PetalWidthCm	0.817954	-0.356544	0.962757	1.000000

```
corr = d.corr()
fig, ax = plt.subplots(figsize = (8,6))
sns.heatmap(corr, annot = True, ax = ax)
```

```
<Axes: >
```




```
#TRAINING THE MODEL
```

```
from sklearn.model_selection import train_test_split
X = d.drop(columns = ["Species"])
Y = d['Species']
x_train, x_test,y_train, y_test = train_test_split(X ,Y ,test_size = 0.30)
```

```
#LOGISTIC_REGRESSION
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
```

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

```
▾ LogisticRegression
LogisticRegression()
```

```
print("Accuracy: ",model.score(x_test, y_test)*100)
```

```
Accuracy:  100.0
```

```
#DECISION TREE CLASSIFIER
```

```
from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
```

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

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

```
print("Accuracy: ",model.score(x_test, y_test)*100)
```

```
Accuracy:  93.33333333333333
```