```
LetsGrowMore Task 1 Project Iris Flower for more
detail refer the below links
```

Sadhana Chidambaram

LinkedIn | Github |

IRIS FLOWER Classification using Machine Learning

DATASET TAKEN FROM KAGGLE

In [5]:
```
from IPython.display import Image
Image(filename='E:/LETSGROWMORE INTERN/BEGGINERS LEVEL/IRIS DATASET/image.png',width
```

Out[5]:

IMPORT LIBRARIES

In [6]:
```
import numpy as np # to calculate matrix problem we go for it
import pandas as pd # data processing where used for import dataset ect..
import matplotlib.pyplot as plt #used for visualzing the dataset
```

Importing dataset

In [9]:
```
iris = pd.read_csv('E:/LETSGROWMORE INTERN/BEGGINERS LEVEL/IRIS DATASET/archive/IRIS
print(iris)
```

```
     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
..            ...          ...           ...          ...             ...
145           6.7          3.0           5.2          2.3  Iris-virginica
146           6.3          2.5           5.0          1.9  Iris-virginica
147           6.5          3.0           5.2          2.0  Iris-virginica
148           6.2          3.4           5.4          2.3  Iris-virginica
149           5.9          3.0           5.1          1.8  Iris-virginica

[150 rows x 5 columns]
```

In [10]:
```
iris.head() # viewing first 5 dataset headers means values
```

Out[10]:

	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 [11]:
```
iris.shape # num of rows and cols
```

Out[11]: (150, 5)

In [12]:
```
#set rest of data atribute into feature variable
features=iris[['sepal_length','sepal_width','petal_length','petal_width']]
```

In [13]:
```
#display top 5 rows of features
features.head()
```

Out[13]:

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [35]:
```
iris.describe(include='all')
```

Out[35]:

	sepal_length	sepal_width	petal_length	petal_width	species

	sepal_length	sepal_width	petal_length	petal_width	species
count	150.000000	150.000000	150.000000	150.000000	150
unique	NaN	NaN	NaN	NaN	3
top	NaN	NaN	NaN	NaN	Iris-setosa
freq	NaN	NaN	NaN	NaN	50
mean	5.843333	3.054000	3.758667	1.198667	NaN
std	0.828066	0.433594	1.764420	0.763161	NaN
min	4.300000	2.000000	1.000000	0.100000	NaN
25%	5.100000	2.800000	1.600000	0.300000	NaN
50%	5.800000	3.000000	4.350000	1.300000	NaN
75%	6.400000	3.300000	5.100000	1.800000	NaN
max	7.900000	4.400000	6.900000	2.500000	NaN

In [36]:

```
iris.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
```

Finding Missing values and replacing

In [42]:

```
iris.drop(columns="petal_width",inplace=False)
```

Out[42]:

	sepal_length	sepal_width	petal_length	species
0	5.1	3.5	1.4	Iris-setosa
1	4.9	3.0	1.4	Iris-setosa
2	4.7	3.2	1.3	Iris-setosa
3	4.6	3.1	1.5	Iris-setosa
4	5.0	3.6	1.4	Iris-setosa
...	...	...	...	...
145	6.7	3.0	5.2	Iris-virginica
146	6.3	2.5	5.0	Iris-virginica
147	6.5	3.0	5.2	Iris-virginica
148	6.2	3.4	5.4	Iris-virginica

	sepal_length	sepal_width	petal_length	species
149	5.9	3.0	5.1	Iris-virginica

150 rows × 4 columns

In [43]:
```
iris.isnull().sum()
```

Out[43]:
```
sepal_length    0
sepal_width     0
petal_length    0
petal_width     0
species         0
dtype: int64
```

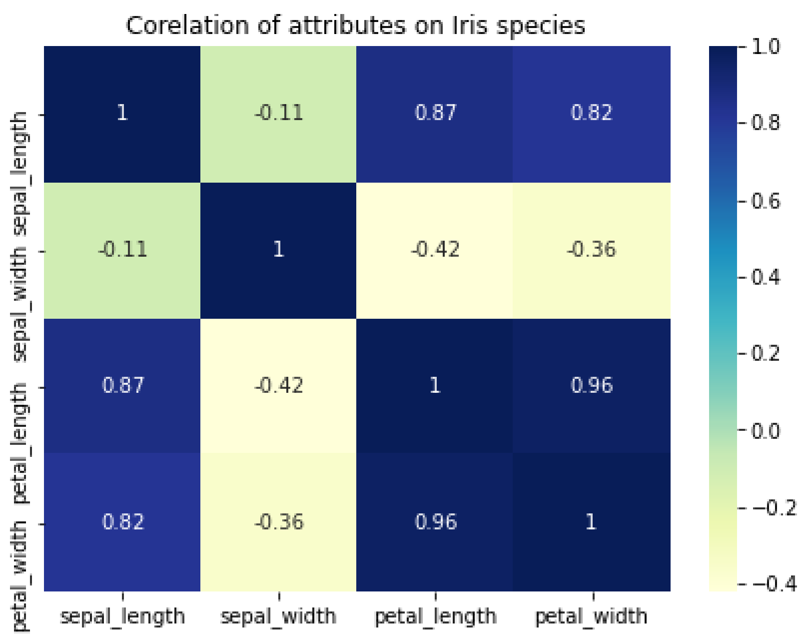
DATA VISUALIZATION

In [44]:
```
#CORRELATION
iris.corr()
```

Out[44]:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.109369	0.871754	0.817954
sepal_width	-0.109369	1.000000	-0.420516	-0.356544
petal_length	0.871754	-0.420516	1.000000	0.962757
petal_width	0.817954	-0.356544	0.962757	1.000000

In [48]:
```
import seaborn as sns
plt.subplots(figsize = (7,5))
sns.heatmap(iris.corr(),annot=True,cmap="YlGnBu").set_title("Corelation of attribute
plt.show()
```



Target Finding for test and train dataset even for predicting

In [18]:
```
#set species column as target column
target=iris['species']
print(target)
```

```
0         Iris-setosa
1         Iris-setosa
2         Iris-setosa
3         Iris-setosa
4         Iris-setosa
             ...
145    Iris-virginica
146    Iris-virginica
147    Iris-virginica
148    Iris-virginica
149    Iris-virginica
Name: species, Length: 150, dtype: object
```

In [19]:
```
#checking unique value counts
target.value_counts()
```

Out[19]:
```
Iris-setosa        50
Iris-versicolor    50
Iris-virginica     50
Name: species, dtype: int64
```

LABLE ENCODING

In [22]:
```
#label encoder from sklearn for label encoding
from sklearn.preprocessing import LabelEncoder
```

In [23]:
```
#coverting text value into numerical value
target=pd.Series(LabelEncoder().fit_transform(target))
```

In [24]:
```
#check unique value counts
target.value_counts()
```

Out[24]:
```
0    50
1    50
2    50
dtype: int64
```

TEST TRAIN SPLIT

In [25]:
```
#import for the sake of split 25% for test
from sklearn.model_selection import train_test_split
```

In [26]:
```
#split data for train and test
x_train,x_test,y_train,y_test=train_test_split(features,target)
```

In [27]:
```
#printing data lengths/number of rows/data
print(len(x_train))
print(len(y_train))
```

```
print(len(x_test))
print(len(y_test))
```

```
112
112
38
38
```

PREDICTION USING NAIVE BAYES ALGORITHM

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values.This is a very strong assumption that is most unlikely in real data, the attributes do not interact.

In [28]:

```
#import naive bayes classifier from sklearn package
from sklearn.naive_bayes import GaussianNB
```

In [29]:

```
#created a object for the classifier
bc=GaussianNB()
```

In [30]:

```
#train usinge train data
bc.fit(x_train,y_train)
```

Out[30]:

```
GaussianNB()
```

In [50]:

```
#predict target data for test feature data and save into pred variable
pred=bc.predict(x_test)
print(pred)
```

```
[0 1 1 2 0 2 0 0 2 2 0 0 1 1 0 2 1 2 1 2 2 2 2 0 0 1 2 2 2 1 2 1 0 0 0 1 1
 1]
```

CLASSIFICATION REPORT ON IRIS FLOWER DATSET

In [32]:

```
#importing classification report function
from sklearn.metrics import classification_report
```

In [33]:

```
print(classification_report(pred,y_test))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        12
           1       1.00      1.00      1.00        12
           2       1.00      1.00      1.00        14

    accuracy                           1.00        38
   macro avg       1.00      1.00      1.00        38
weighted avg       1.00      1.00      1.00        38
```

ACCURACY SCORE 100

Thank you