PREDICTING THE ENERGY OUTPUT OF WIND TURBINE BASED ON WEATHER CONDITION

ASSIGNMENT - 3

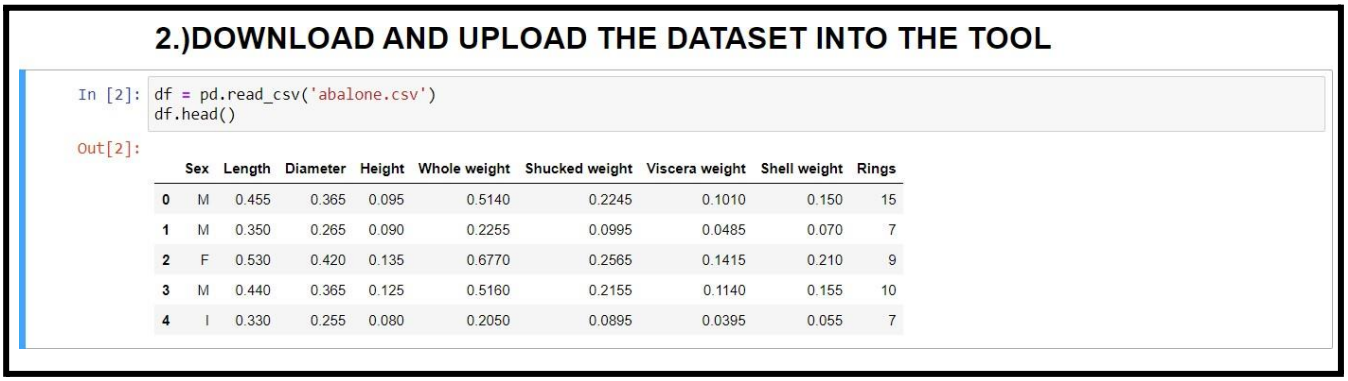
Date	4th October 2022
Team ID	PNT2022TMID54445
Student Name	E.Gokula Krishnan (310619106304)
Domain Name	Education
Project Name	PREDICTING THE ENERGY OUTPUT OF Wind TURBINE BASED ON WEATHER CONDITION
Maximum Marks	2 Marks

1.)IMPORT THE REQUIRED LIBRARIES

1.)IMPORT THE REQUIRED LIBRARIES

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
```

2.)DOWNLOAD AND UPLOAD THE DATASET



2.)DOWNLOAD AND UPLOAD THE DATASET INTO THE TOOL

```
In [2]: df = pd.read_csv('abalone.csv')
        df.head()
```

Out[2]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

3.)HANDLE MISSING VALUES AND DEAL WITH THEM

3.)CHECK FOR MISSING VALUES AND DEAL WITH THEM

```
In [3]: df.isnull().sum()
```

```
Out[3]: Sex               0
        Length            0
        Diameter          0
        Height            0
        Whole weight      0
        Shucked weight    0
        Viscera weight    0
        Shell weight      0
        Rings             0
        dtype: int64
```

4.) PERFORM THE DESCRIPTIVE STATISTICS ON THE DATASET

4.)PERFORM DESCRIPTIVE STATISTICS ON THE DATASET

```
In [4]: df.describe()
```

Out[4]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4177 entries, 0 to 4176
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Sex             4177 non-null   object
 1   Length          4177 non-null   float64
 2   Diameter        4177 non-null   float64
 3   Height          4177 non-null   float64
 4   Whole weight    4177 non-null   float64
 5   Shucked weight  4177 non-null   float64
 6   Viscera weight  4177 non-null   float64
 7   Shell weight    4177 non-null   float64
 8   Rings           4177 non-null   int64
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
```

5.) PERFORM VARIOUS VISUALISATIONS
a.) UNIVARIANTE ANALYSIS

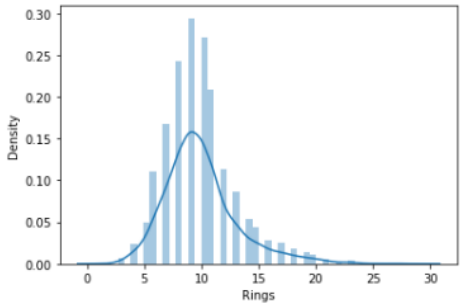
5.)PERFORM VISUALIZATIONS

a.)UNIVARIANTE ANALYSIS

In [6]:
```
sns.distplot(df.Rings)
```

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and
will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexi
bility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

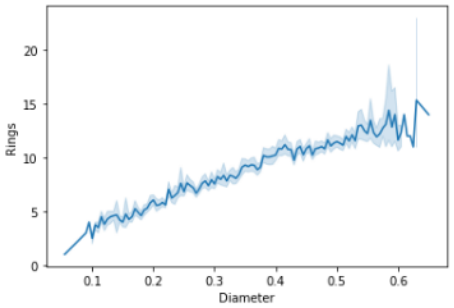
Out[6]:
```
<AxesSubplot:xlabel='Rings', ylabel='Density'>
```



In [7]:
```
sns.lineplot(df.Diameter,df.Rings)
```

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword arg
s: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[7]:
```
<AxesSubplot:xlabel='Diameter', ylabel='Rings'>
```



In [8]:
```
plt.pie(df.Sex.value_coumts(),[0.05,0.05,0.05],colors=['red','green', 'blue'],labels=['Male','Female', 'Infant'],autopct='%1.1fB%',
plt.title('Sex')
plt.shcm()
```

In [9] : sns . barp1ot (d-fi . Rings . va1ue_count s) . Index, df .Rings . va1ue_c ount s ( ))

```
C : \Us ers \Pren\anaconda3\ lib\site— packages \s eaborn\_deco rato rs. py : 36: F uturef4arn ing: Pass the tolls *ing var tables as Keyword arg
s : x, y . F non ver s ion 0. 1,2          the only val id pos itional argumnt will be' data,'          and pass irig ot her argunents without an explicit
keynrd will resu lt in an error or nisinterpretation .
  earnings .earn (
```

Out[9]: ‹AxesSubplot:ylabel='Rimgs'›

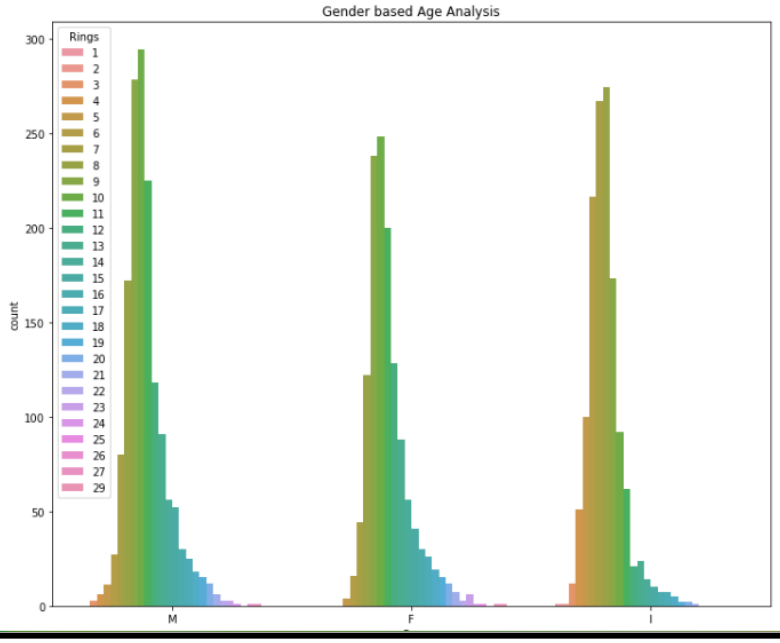
1 Z J •• 5 6 7 0 9]011lZ1M4b16]7B182DJ]EZ5ZtJ5KTJZ9

b.) BI - VARIANTE ANALYSIS

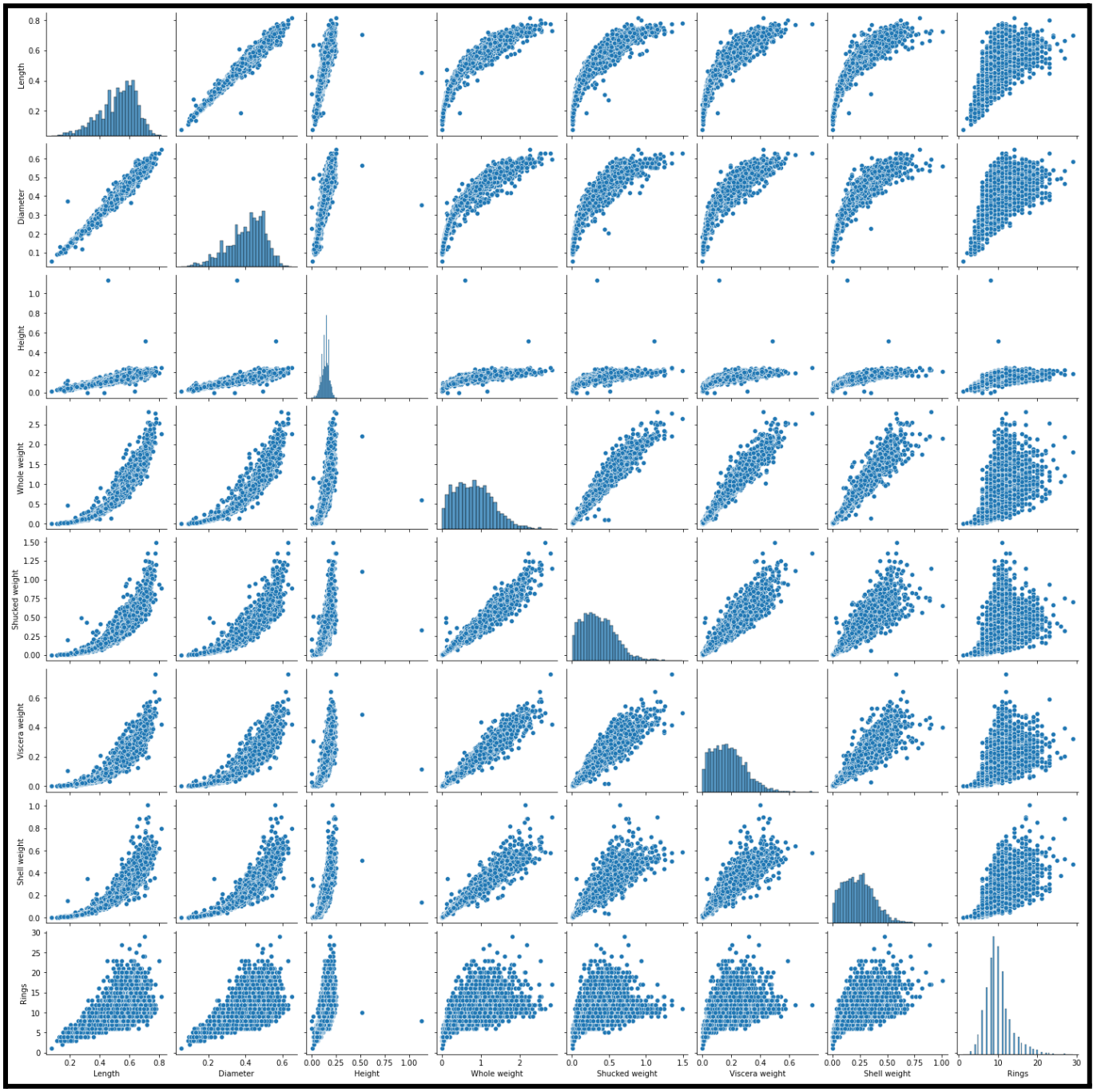
b.)BI-VARIANTE ANALYSIS

In [10]:
```
def countplot_2(x,hue,title=None,figsize=(12,10)):
  plt.figure(figsize=figsize)
  sns.countplot(data=df[[x,hue]],x=x,hue=hue)
  plt.title(title)
  plt.show()
```

In [11]:
```
countplot_2('Sex','Rings','Gender based Age Analysis')
```



c.) MULTI - VARIANTE ANALYSIS

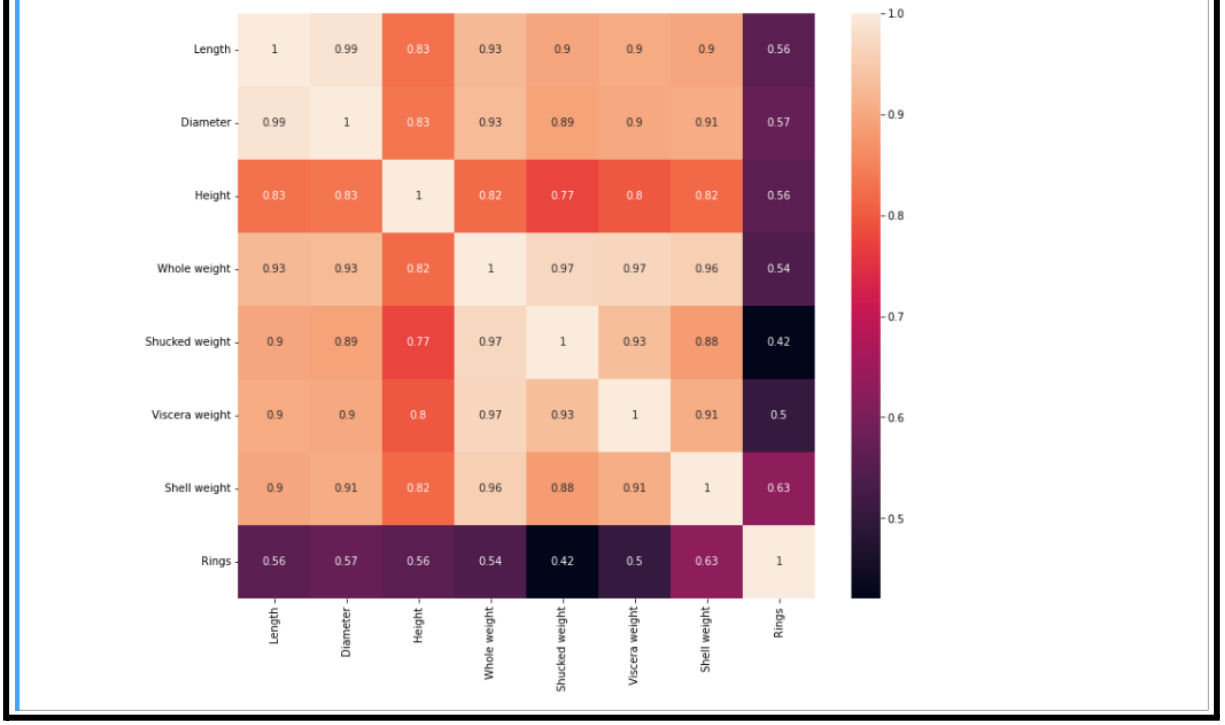


In [13]:
```
df.corr()
```

Out[13]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
Length	1.000000	0.986812	0.827554	0.925261	0.897914	0.903018	0.897706	0.556720
Diameter	0.986812	1.000000	0.833684	0.925452	0.893162	0.899724	0.905330	0.574660
Height	0.827554	0.833684	1.000000	0.819221	0.774972	0.798319	0.817338	0.557467
Whole weight	0.925261	0.925452	0.819221	1.000000	0.969405	0.966375	0.955355	0.540390
Shucked weight	0.897914	0.893162	0.774972	0.969405	1.000000	0.931961	0.882617	0.420884
Viscera weight	0.903018	0.899724	0.798319	0.966375	0.931961	1.000000	0.907656	0.503819
Shell weight	0.897706	0.905330	0.817338	0.955355	0.882617	0.907656	1.000000	0.627574
Rings	0.556720	0.574660	0.557467	0.540390	0.420884	0.503819	0.627574	1.000000

```
In [14]: plt.figure(figsize=(12,10))
         sns.heatmap(df.corr(),annot=True)
         plt.show()
```

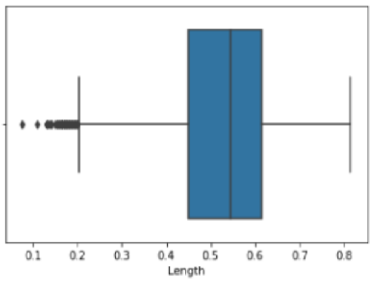


6.) FIND AND REPLACE THE OUTLIERS

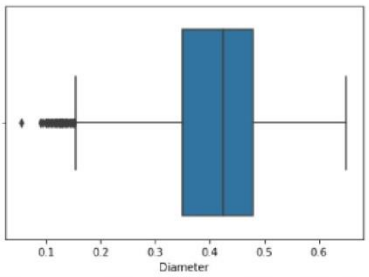
6.)FIND THE OUTLIERS AND REPLACE THE OUTLIERS

In [15]:
```
for i in df.columns.drop('Sex'):
    sns.boxplot(df[i])
    plt.show()
```

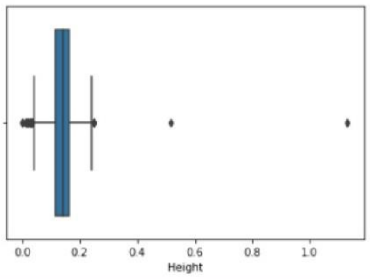
```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



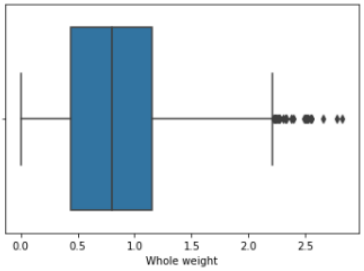

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



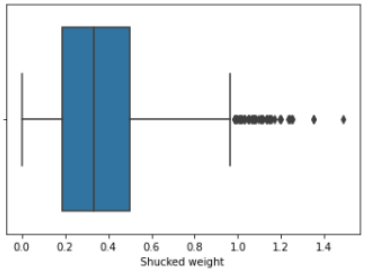

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



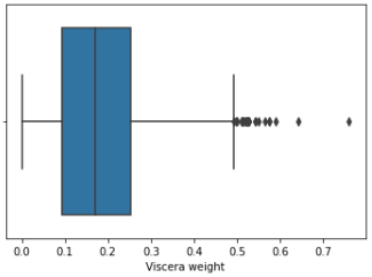
```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



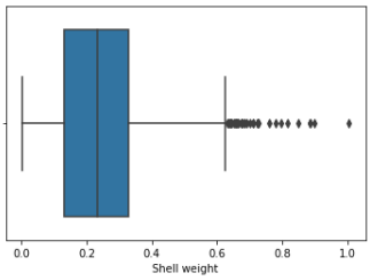

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



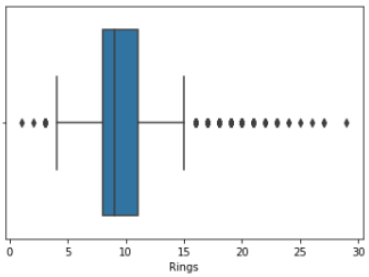
```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```




```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



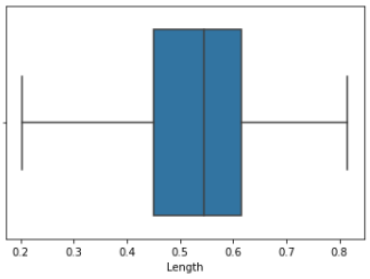

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



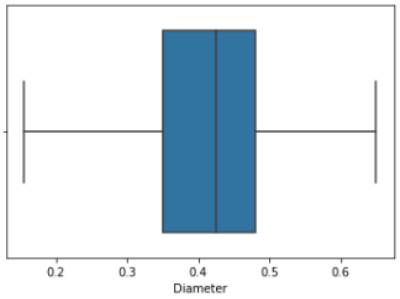
In [16]:
```
for i in df.columns.drop('Sex'):
    Q1 = df[i].quantile(0.25)
    Q3 = df[i].quantile(0.75)
    IQR = Q3-Q1
    upper_limit = Q3 + (1.5*IQR)
    lower_limit = Q1 - (1.5*IQR)
    df[i] = np.where(df[i]>=upper_limit,Q3 + (1.5*IQR),df[i])
    df[i] = np.where(df[i]<=lower_limit,Q1 - (1.5*IQR),df[i])
```

In [17]:
```
for i in  df.columns.drop('Sex'):
    sns.boxplot(df[i])
    plt.show()
```

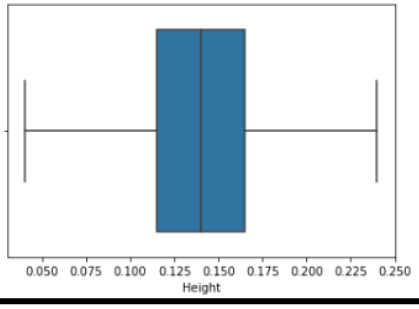
```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



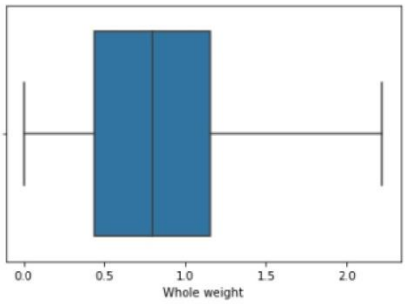

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



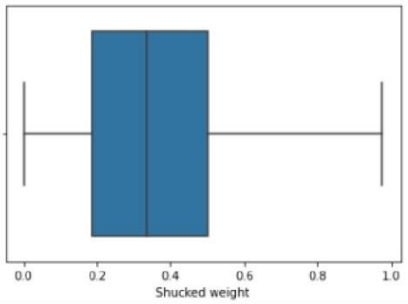

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```




```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



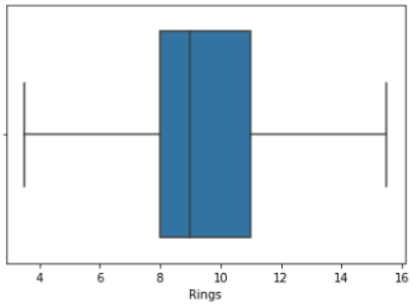
C : \Users\Pren\anaconda3\11b\s1te—packages\seaborn_decorators.py: 36: Futuretgarn1ng: Pass the fo1Irns1ng var lab1e as a keywrd ar
g:x.Frrxaverstonél.12,theon1yva11dposItIona1argument+s111be'data' and passIng other argument s ithout an exp1lc It key
word will result in an error or misinterpretation.
ma rmings .ma rn (

e.o or oz or a‹ es

C : \Us ers \Pren\anaconda3 \lib\s ite— pac kage s \s eaborn_deco rato rs . py: 36: F uturef4arn ing: Pas s the tolls *ing var table as a keyemrd ar
g:x.FromversionB.12,theonlyvalidpositionalargumentwillbe'data',andpassingotherargumentswithoutanexplicitkey
wordwillresultinanerrororinterpretation.
namings.narn(

e.o e.i o.z e.z e.s es es

```
C:\Users\Prem\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword ar
g: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit key
word will result in an error or misinterpretation.
  warnings.warn(
```



7.) CHECK FOR CATEGORICAL COLUMNS AND ENCODE THEM

7.)CHECK FOR CATEGORICAL COLUMNS AND PERFORM ENCODING

In [18]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df.Sex = le.fit_transform(df.Sex)
```

In [19]:
```
df.head()
```

Out[19]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	2	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15.0
1	2	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7.0
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9.0
3	2	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10.0
4	1	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7.0

8.)SPLIT DATA INTO DEPENDENT AND INDEPENDENTVARIABLES

8.)SPLIT THE DATA INTO DEPENDENT AND INDEPENDENT VARIABLES

In [20]:
```
X = df.drop(columns=['Rings'])
X.head()
```

Out[20]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
0	2	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150
1	2	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210
3	2	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155
4	1	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055

In [21]:
```
Y = df.Rings
Y.head()
```

Out[21]:
```
0    15.0
1     7.0
2     9.0
3    10.0
4     7.0
Name: Rings, dtype: float64
```

9.) SCALE THE INDEPENDENT VARIABLES

9.)SCALE THE INDEPENDENT VARIABLES

In [22]:
```
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
X_scaled = pd.DataFrame(scale.fit_transform(X),columns=X.columns)
X_scaled.head()
```

Out[22]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
0	1.0	0.412245	0.424242	0.275	0.230813	0.229231	0.204372	0.237220
1	1.0	0.240816	0.222222	0.250	0.100755	0.101026	0.097611	0.109425
2	0.0	0.534694	0.535354	0.475	0.304294	0.262051	0.286731	0.333067
3	1.0	0.387755	0.424242	0.425	0.231714	0.220000	0.230808	0.245208
4	0.5	0.208163	0.202020	0.200	0.091514	0.090769	0.079309	0.085463

10.)SPLIT THE DATA INTO TRAINING AND TESTING

10.)SPLIT THE DATA INTO TRAINING AND TESTING DATA

In [23]:
```
from sklearn.model_selection import train_test_split
x_train , x_test , y_train , y_test = train_test_split(X_scaled,Y,test_size=0.2,random_state=0)
```

11.) BUILD THEMODEL

11.)BUILD THE MODEL

In [24]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

12.) TRAIN THEMODEL

12.)TRAIN THE MODEL

In [25]:
```
model.fit(x_train,y_train)
```

Out[25]:
```
LinearRegression()
```

13.) TEST THE MODEL

13.)TEST THE MODEL

In [26]:
```
y_predict = model.predict(x_test)
```

In [27]:
```
pd.DataFrame({"Actual":y_test,"Predicted":y_predict.round(0)})
```

Out[27]:

	Actual	Predicted
668	13.0	13.0
1580	8.0	9.0
3784	11.0	10.0
463	5.0	5.0
2615	12.0	10.0
...	...	...
575	11.0	10.0
3231	12.0	9.0
1084	7.0	9.0
290	15.5	12.0
2713	4.0	6.0

836 rows × 2 columns

14.) MEASURE THE PERFORMANCE USING METRICS

14.)MEASURE THE PERFORMANCE USING METRICS

In [28]:
```
from sklearn import metrics
metrics.r2_score(y_test,y_predict)
```

Out[28]:
```
0.58432381444787
```