In [1]:
```
# Importing Important Libraries

import pandas as pd
import numpy as np

# visualization Libraries
import seaborn as sns
import matplotlib.pyplot as plt

# Avoid Warnings
import warnings
warnings.filterwarnings('ignore')

# To print all rows and columns
from IPython import display
pd.set_option('display.max_columns',None)
pd.set_option('display.max_rows',None)


```

Data Collection :

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

Out[2]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Ide
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	O
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	O
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	O
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	O
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	O

Data Preprocessing :

In [3]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8523 entries, 0 to 8522
Data columns (total 12 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Item_Identifier            8523 non-null   object
 1   Item_Weight                7060 non-null   float64
 2   Item_Fat_Content           8523 non-null   object
 3   Item_Visibility            8523 non-null   float64
 4   Item_Type                  8523 non-null   object
 5   Item_MRP                   8523 non-null   float64
 6   Outlet_Identifier          8523 non-null   object
 7   Outlet_Establishment_Year  8523 non-null   int64
 8   Outlet_Size                6113 non-null   object
 9   Outlet_Location_Type       8523 non-null   object
 10  Outlet_Type                8523 non-null   object
 11  Item_Outlet_Sales          8523 non-null   float64
dtypes: float64(4), int64(1), object(7)
memory usage: 799.2+ KB
```

In [3]:
```
1 df.shape
```

Out[3]:
```
(8523, 12)
```

In [4]:
```
1 df.columns
```

Out[4]:
```
Index(['Item_Identifier', 'Item_Weight', 'Item_Fat_Content', 'Item_Visibilit
y',
       'Item_Type', 'Item_MRP', 'Outlet_Identifier',
       'Outlet_Establishment_Year', 'Outlet_Size', 'Outlet_Location_Type',
       'Outlet_Type', 'Item_Outlet_Sales'],
      dtype='object')
```

In [5]:
```
1 df.index
```

Out[5]:
```
RangeIndex(start=0, stop=8523, step=1)
```

In [6]:
```
1 df.axes
```

Out[6]:
```
[RangeIndex(start=0, stop=8523, step=1),
 Index(['Item_Identifier', 'Item_Weight', 'Item_Fat_Content', 'Item_Visibilit
y',
        'Item_Type', 'Item_MRP', 'Outlet_Identifier',
        'Outlet_Establishment_Year', 'Outlet_Size', 'Outlet_Location_Type',
        'Outlet_Type', 'Item_Outlet_Sales'],
       dtype='object')]
```

In [7]:
```
1 df.dtypes
```

Out[7]:
```
Item_Identifier              object
Item_Weight                 float64
Item_Fat_Content              object
Item_Visibility             float64
Item_Type                     object
Item_MRP                    float64
Outlet_Identifier             object
Outlet_Establishment_Year      int64
Outlet_Size                   object
Outlet_Location_Type          object
Outlet_Type                   object
Item_Outlet_Sales           float64
dtype: object
```

In [8]:
```
1 df.describe()
```

Out[8]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
count	7060.000000	8523.000000	8523.000000	8523.000000	8523.000000
mean	12.857645	0.066132	140.992782	1997.831867	2181.288914
std	4.643456	0.051598	62.275067	8.371760	1706.499616
min	4.555000	0.000000	31.290000	1985.000000	33.290000
25%	8.773750	0.026989	93.826500	1987.000000	834.247400
50%	12.600000	0.053931	143.012800	1999.000000	1794.331000
75%	16.850000	0.094585	185.643700	2004.000000	3101.296400
max	21.350000	0.328391	266.888400	2009.000000	13086.964800

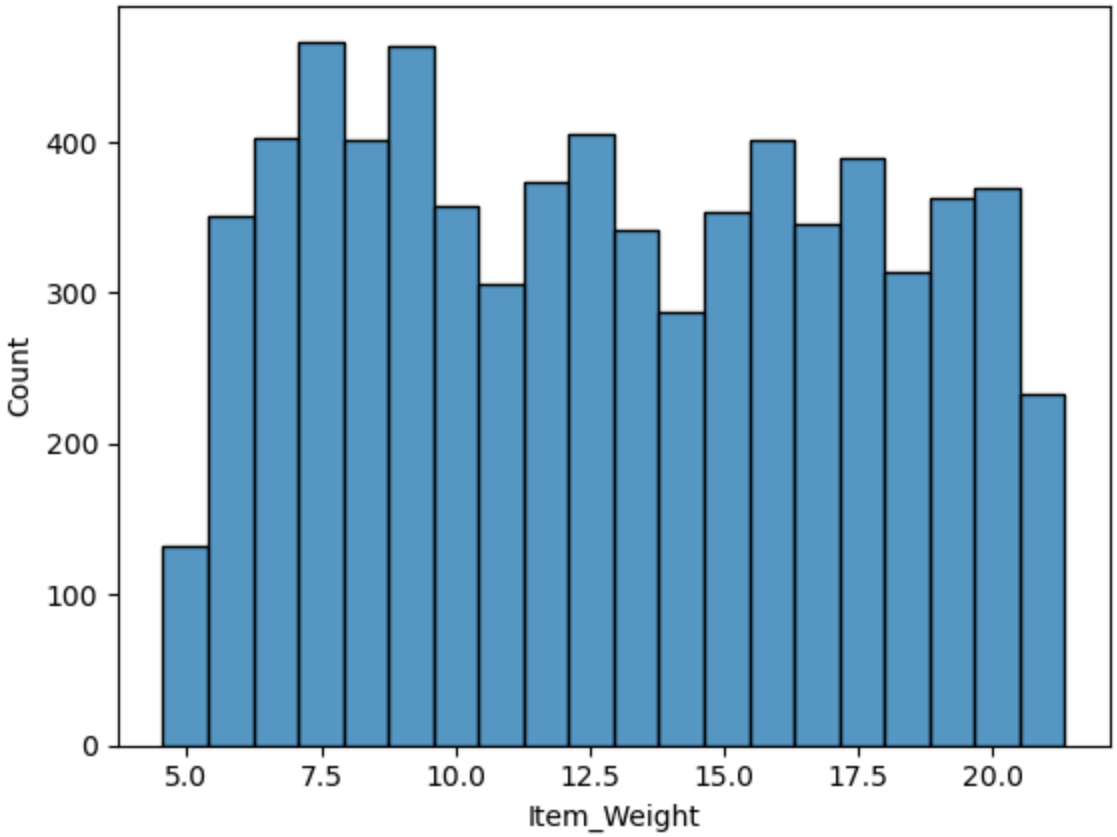
In [9]:
```
1 df.isna().sum()
```

Out[9]:
```
Item_Identifier               0
Item_Weight                1463
Item_Fat_Content              0
Item_Visibility               0
Item_Type                     0
Item_MRP                      0
Outlet_Identifier             0
Outlet_Establishment_Year     0
Outlet_Size                2410
Outlet_Location_Type          0
Outlet_Type                   0
Item_Outlet_Sales             0
dtype: int64
```

Data Visualization :

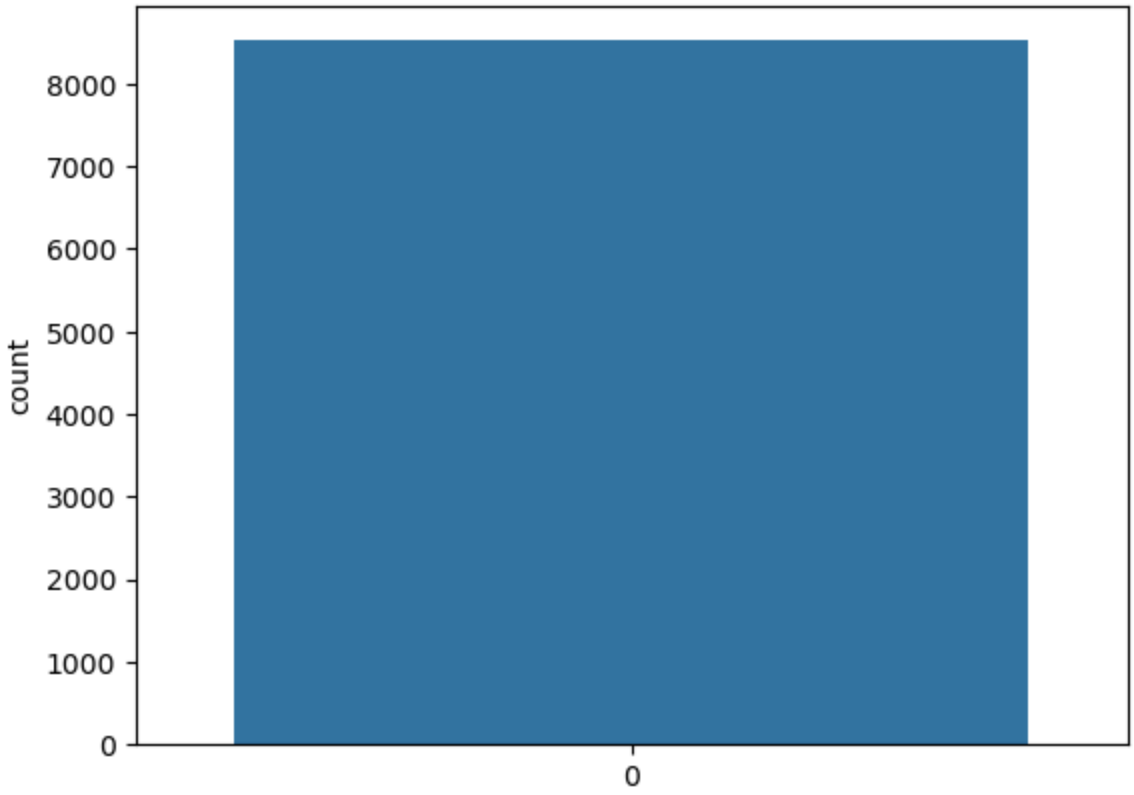
In [21]:
```
1 sns.histplot(df['Item_Weight'])
```

Out[21]:
```
<Axes: xlabel='Item_Weight', ylabel='Count'>
```



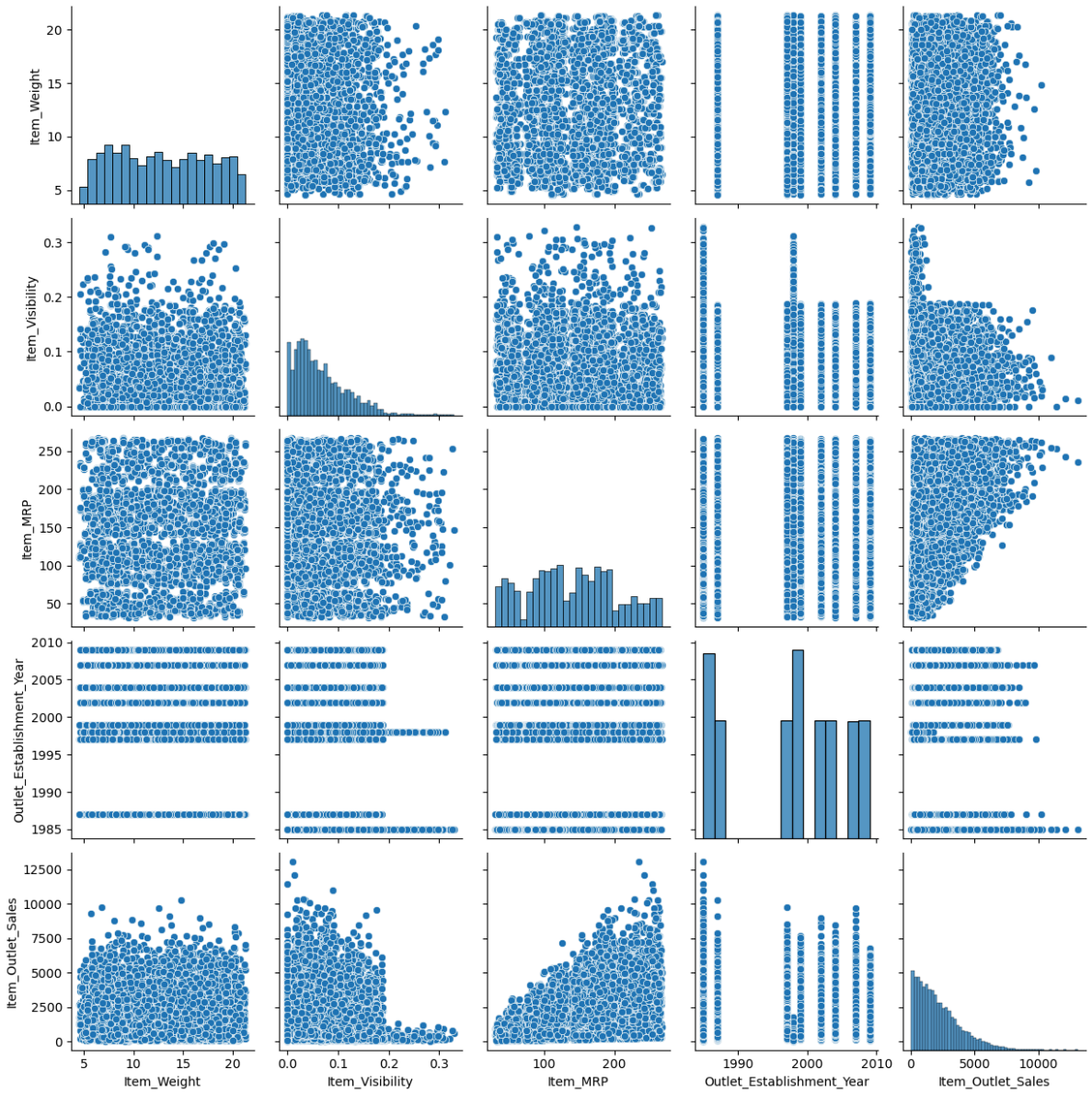
In [22]:
```
1 sns.countplot(df['Item_Weight'])
```

Out[22]: <Axes: ylabel='count'>



In [23]:
```
1 sns.pairplot(df)
```

Out[23]: <seaborn.axisgrid.PairGrid at 0x14da4166a40>



1) Item_Weight :

In [20]:
```
1 df['Item_Weight'].unique()
```

Out[20]:
```
array([ 9.3  ,  5.92 , 17.5  , 19.2  ,  8.93 , 10.395, 13.65 ,    nan,
       16.2  , 11.8  , 18.5  , 15.1  , 17.6  , 16.35 ,  9.   , 13.35 ,
       18.85 , 14.6  , 13.85 , 13.   ,  7.645, 11.65 ,  5.925, 19.25 ,
       18.6  , 18.7  , 17.85 , 10.   ,  8.85 ,  9.8  , 13.6  , 21.35 ,
       12.15 ,  6.42 , 19.6  , 15.85 ,  7.39 , 10.195,  9.895, 10.895,
        7.905,  9.195,  8.365,  7.97 , 17.7  , 19.35 ,  8.645, 15.6  ,
       18.25 ,  7.855,  7.825,  8.39 , 12.85 , 19.   ,  5.905,  7.76 ,
       16.75 , 15.5  ,  6.055,  6.305, 20.85 , 20.75 ,  8.895, 19.7  ,
        8.75 , 13.3  ,  8.31 , 19.75 , 17.1  , 10.5  ,  6.635, 14.15 ,
        8.89 ,  9.1  ,  7.5  , 16.85 ,  7.485, 11.6  , 12.65 , 20.25 ,
        8.6  , 12.6  ,  8.88 , 20.5  , 13.5  ,  7.235,  6.92 ,  8.02 ,
       12.8  , 16.6  , 14.   , 16.   , 21.25 ,  7.365, 18.35 ,  5.465,
        7.27 ,  6.155, 19.5  , 15.2  , 14.5  , 13.1  , 12.3  , 11.1  ,
       11.3  ,  5.75 , 11.35 ,  6.525, 10.3  ,  5.78 , 11.85 , 18.75 ,
        5.26 , 16.1  ,  9.5  , 13.8  , 14.65 ,  6.67 ,  6.11 , 17.2  ,
        6.32 ,  4.88 ,  5.425, 14.1  ,  7.55 , 17.25 , 12.   , 10.1  ,
        7.785, 13.15 ,  8.5  ,  7.63 ,  9.285,  7.975, 15.7  ,  8.985,
       20.35 ,  6.59 , 19.85 ,  6.26 , 18.2  ,  8.695,  7.075,  8.195,
        7.09 ,  6.095,  6.15 ,  9.395, 15.75 ,  7.475,  6.445, 19.1  ,
       15.   , 16.7  ,  7.07 ,  6.48 ,  9.695, 11.15 ,  9.6  , 20.7  ,
        5.5  ,  7.895, 17.35 ,  7.285,  6.17 , 11.395,  7.71 , 12.1  ,
       14.35 ,  8.1  ,  8.05 , 16.5  ,  6.785,  7.575,  7.47 , 15.25 ,
        7.605, 18.   , 21.2  ,  8.97 , 10.6  ,  6.865, 10.8  , 15.15 ,
       18.1  ,  6.655, 20.1  ,  7.935, 15.35 , 12.35 ,  6.85 ,  8.775,
       14.85 ,  7.84 , 12.5  ,  8.325,  5.765,  5.985, 14.3  ,  6.135,
        8.51 ,  6.65 ,  5.695,  6.36 ,  8.3  ,  7.56 ,  8.71 ,  6.695,
       14.8  , 17.75 ,  8.575,  6.57 ,  8.68 ,  5.63 ,  9.13 ,  6.715,
        5.82 ,  7.93 ,  5.   ,  7.445,  6.675,  8.18 ,  6.98 ,  7.435,
       20.6  ,  8.355,  8.975, 20.2  ,  5.655,  5.175, 20.   ,  7.67 ,
        4.785,  8.395,  6.175,  8.21 ,  5.845,  7.17 ,  8.785,  7.89 ,
        5.32 ,  5.03 ,  8.945,  6.28 ,  7.565,  9.31 ,  7.02 ,  5.46 ,
        6.13 ,  6.55 , 17.   , 16.25 ,  5.15 ,  7.865,  6.575,  7.06 ,
        5.785,  7.42 ,  6.235,  6.75 ,  5.86 ,  5.035,  6.38 ,  5.675,
       11.5  , 21.   , 21.1  ,  4.61 ,  7.   ,  6.405, 14.7  ,  7.68 ,
        8.185,  8.655, 10.85 ,  8.42 ,  7.85 ,  4.59 ,  7.51 , 11.   ,
        6.71 , 14.75 ,  7.59 ,  5.155,  6.365,  8.235,  5.365,  8.485,
        7.535,  4.92 ,  6.385,  8.26 ,  7.945,  8.63 ,  9.21 ,  6.965,
        8.905,  7.21 ,  7.3  ,  9.27 , 10.695,  6.215,  7.405,  7.72 ,
        6.115,  6.035,  6.78 , 15.3  ,  7.105,  7.52 ,  4.805,  6.425,
        7.6  ,  6.61 ,  7.325,  8.115,  5.94 ,  5.635,  7.35 ,  5.825,
        6.63 ,  7.05 ,  8.315,  6.8  ,  7.035,  8.96 ,  5.51 ,  8.43 ,
        8.27 ,  7.81 ,  6.885,  5.44 ,  5.405,  4.635, 10.65 ,  5.735,
        6.465,  8.84 ,  7.75 ,  6.765,  9.065,  7.655,  4.615,  8.76 ,
        7.1  ,  6.195,  5.615,  8.52 ,  7.26 ,  6.825,  5.325,  5.59 ,
        5.88 ,  5.19 ,  6.985,  9.06 ,  6.69 ,  8.615,  7.275,  6.96 ,
        9.17 ,  8.155,  5.73 ,  8.935,  8.92 ,  7.36 ,  7.64 ,  5.34 ,
        7.22 ,  6.615,  6.76 ,  6.3  ,  5.98 ,  8.06 ,  6.44 ,  5.095,
        8.8  ,  6.325,  7.31 ,  9.035,  9.105,  7.145,  4.905,  4.555,
        8.   ,  7.315,  6.89 ,  5.945,  6.86 ,  6.935,  6.03 ,  7.725,
        5.885,  7.155,  6.46 ,  5.48 ,  8.01 ,  5.8  ,  5.305,  6.905,
        7.96 ,  5.11 ,  8.77 ,  7.685,  8.275,  8.38 ,  8.35 ,  9.42 ,
        6.775,  6.4  ,  6.895,  5.485,  6.52 ,  8.67 ,  5.21 ,  5.4  ])
```

In [19]:
```
1 df['Item_Weight'].nunique()
```

Out[19]: 415

In [18]:
```
1 df['Item_Weight'].value_counts()
```

Out[18]:
```
12.150    86
17.600    82
13.650    77
11.800    76
15.100    68
9.300     68
10.500    66
16.700    66
19.350    63
20.700    62
16.000    62
9.800     61
17.700    60
17.750    60
18.850    59
15.850    59
15.000    59
16.750    58
18.250    58
```

In [24]:
```
1 df['Item_Weight'].dtypes
```

Out[24]: dtype('float64')

In [17]:
```
1 df['Item_Weight'].isna().sum()
```

Out[17]: 1463

In [4]:
```
1 mean = df['Item_Weight'].mean()
2 mean
```

Out[4]: 12.857645184135976

In [5]:
```
1 df['Item_Weight'].fillna(mean,inplace=True)
```

In [29]:
```
1 df['Item_Weight'].isna().sum()
```

Out[29]: 0

2) Outlet_Size :

In [30]:
```
1 df['Outlet_Size'].unique()
```

Out[30]: array(['Medium', nan, 'High', 'Small'], dtype=object)

In [31]:
```
df['Outlet_Size'].nunique()
```

Out[31]:
```
3
```

In [32]:
```
df['Outlet_Size'].value_counts()
```

Out[32]:
```
Medium    2793
Small     2388
High       932
Name: Outlet_Size, dtype: int64
```

In [33]:
```
df['Outlet_Size'].dtypes
```

Out[33]:
```
dtype('O')
```

In [34]:
```
df['Outlet_Size'].isna().sum()
```

Out[34]:
```
2410
```

In [6]:
```
mode = df['Outlet_Size'].mode()[0]
mode
```

Out[6]:
```
'Medium'
```

In [7]:
```
df['Outlet_Size'].fillna(mode,inplace = True)
```

In [39]:
```
df['Outlet_Size'].isna().sum()
```

Out[39]:
```
0
```

In [55]:
```
df.isna().sum()
```

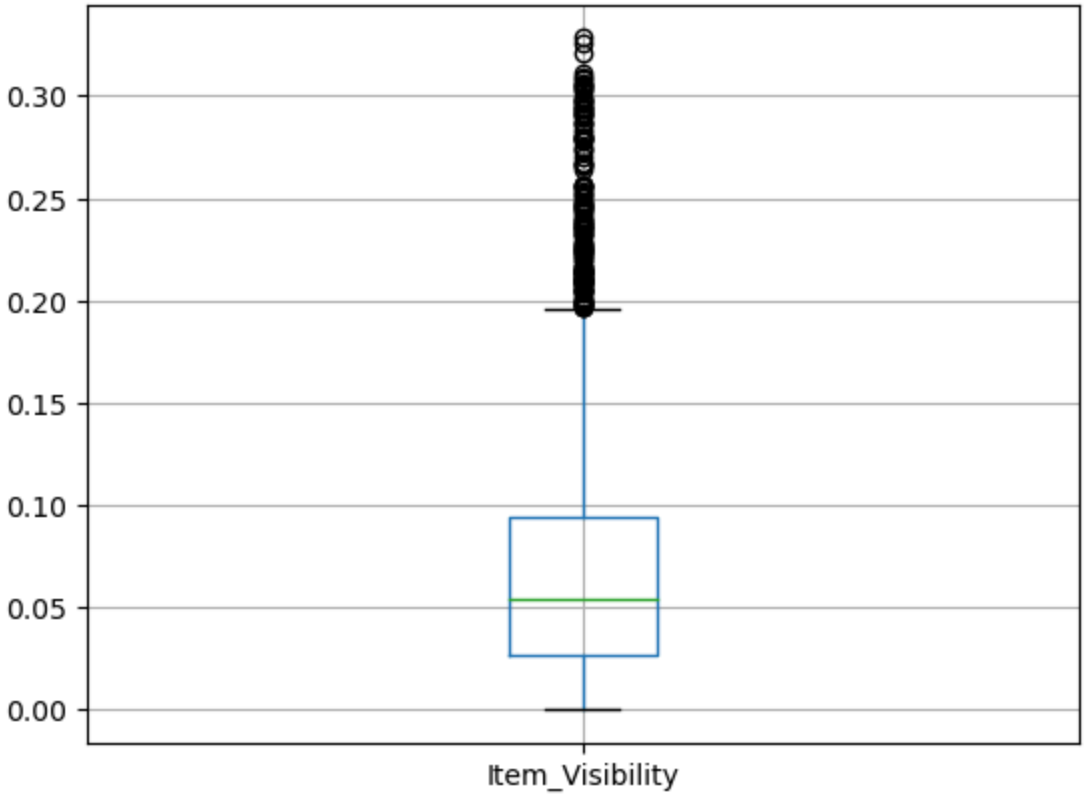
Out[55]:
```
Item_Identifier              0
Item_Weight                  0
Item_Fat_Content              0
Item_Visibility              0
Item_Type                    0
Item_MRP                     0
Outlet_Identifier            0
Outlet_Establishment_Year    0
Outlet_Size                  0
Outlet_Location_Type         0
Outlet_Type                  0
Item_Outlet_Sales            0
dtype: int64
```

In [ ]:
```
# All the Missing values are successfully removed
```

Handling Outliers :

In [44]:
```
1 df[['Item_Visibility']].boxplot()
```

Out[44]: <Axes: >



In [8]:
```
q1 = df['Item_Visibility'].quantile(0.25)
q3 = df['Item_Visibility'].quantile(0.75)

median = df['Item_Visibility'].median()
IQR = q3 - q1

upper_tail = q3 + 1.5 * IQR
lower_tail = q3 - 1.5 * IQR

print('First Quantile : ',q1)
print('Third Quantile : ',q3)

print('IQR :-->',IQR)

print('Upper_tail :-->',upper_tail)
print('Lower tail:',lower_tail)


```

```
First Quantile :  0.0269894775
Third Quantile :  0.0945852925
IQR :--> 0.067595815
Upper_tail :--> 0.195979015
Lower tail: -0.006808430000000004
```

In [9]:
```
df['Item_Visibility'].loc[df['Item_Visibility']> upper_tail]
```

Out[9]:
```
49      0.255395
83      0.293418
108     0.278974
174     0.291865
334     0.204700
434     0.264125
502     0.228993
521     0.297884
532     0.233040
680     0.210376
847     0.220226
854     0.328391
966     0.205295
1159    0.247321
1225    0.214140
1272    0.227190
1291    0.223440
1311    0.267353
1324    0.256375
```

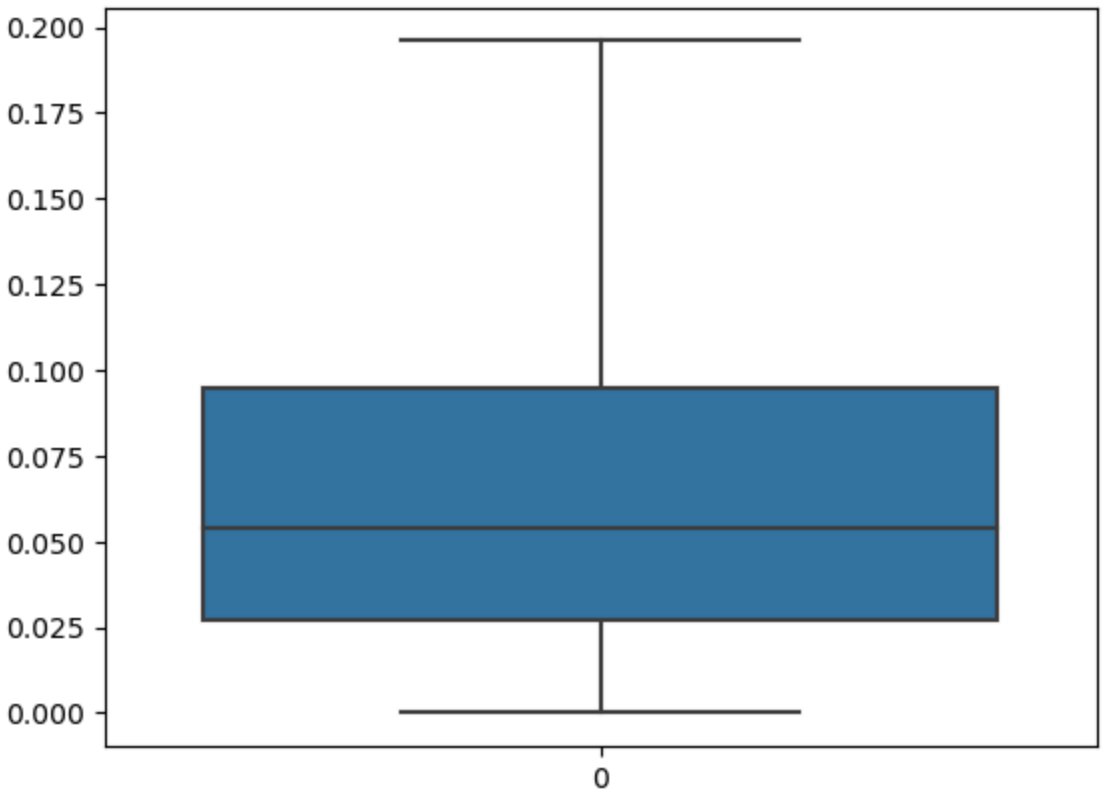
In [10]:
```
df['Item_Visibility'].loc[df['Item_Visibility'] > upper_tail] = upper_tail
```

In [11]:
```
df['Item_Visibility'].loc[df['Item_Visibility'] > upper_tail]
```

Out[11]:
```
Series([], Name: Item_Visibility, dtype: float64)
```

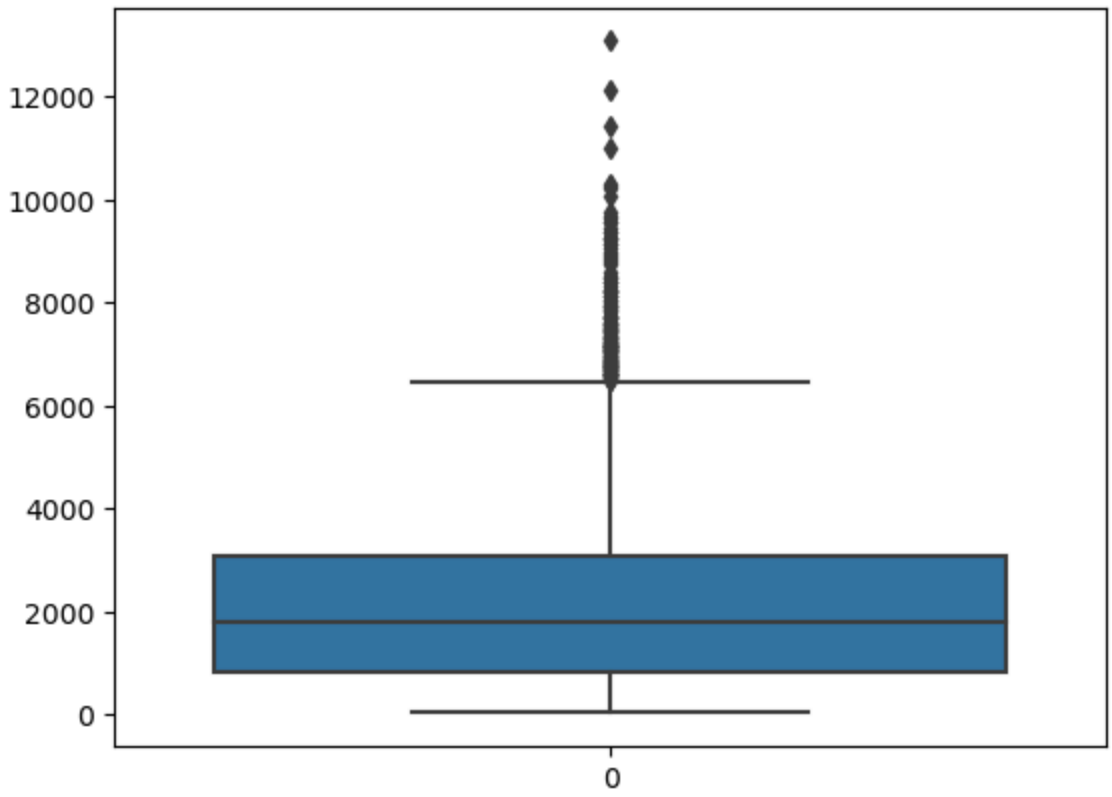
In [51]:
```
1 sns.boxplot(df['Item_Visibility'])
```

Out[51]: <Axes: >



In [54]:
```
1 sns.boxplot(df['Item_Outlet_Sales'])
```

Out[54]: <Axes: >



In [12]:
```
1  q1 = df['Item_Outlet_Sales'].quantile(0.25)
2  q3 = df['Item_Outlet_Sales'].quantile(0.75)
3
4  median = df['Item_Outlet_Sales'].median()
5  IQR = q3 - q1
6
7  upper_tail = q3 + 1.5 * IQR
8  lower_tail = q3 - 1.5 * IQR
9
10 print('First Quantile : ',q1)
11 print('Third Quantile : ',q3)
12
13 print('IQR :-->',IQR)
14
15 print('Upper_tail :-->',upper_tail)
16 print('Lower tail:',lower_tail)
17
18
```

```
First Quantile :  834.2474
Third Quantile :  3101.2964
IQR :--> 2267.049
Upper_tail :--> 6501.8699
Lower tail: -299.2770999999998
```

In [14]:
```
1 df['Item_Outlet_Sales'].loc[df['Item_Outlet_Sales'] > upper_tail]
```

Out[14]:
```
43      6768.5228
130     7968.2944
132     6976.2524
145     7370.4060
203     6704.6060
240     6795.1548
243     7222.5984
275     7298.4996
276     7452.9652
304     7696.6480
333     9267.9360
373     7763.2280
402     6911.0040
424     6687.9610
456     9158.0790
472     8114.7704
497     7094.7648
640     7192.6374
641     6611.3940
```

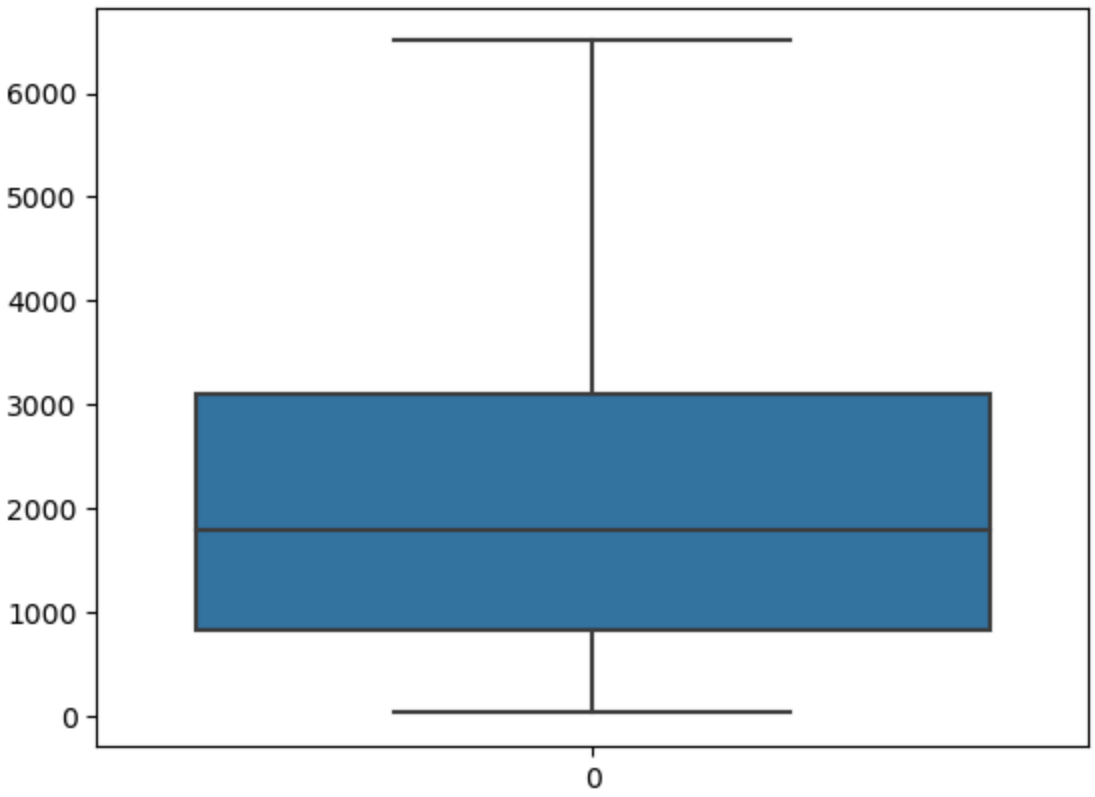
In [15]:
```
1 df['Item_Outlet_Sales'].loc[df['Item_Outlet_Sales'] > upper_tail] = upper_
```

In [16]:
```
1 df['Item_Outlet_Sales'].loc[df['Item_Outlet_Sales'] > upper_tail]
```

Out[16]:
```
Series([], Name: Item_Outlet_Sales, dtype: float64)
```

In [59]:
```
1 sns.boxplot(df['Item_Outlet_Sales'])
```

Out[59]: <Axes: >



Groupby function :

In [62]:
```
1 df.groupby('Item_Identifier').first().head()
```

Out[62]:

Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identi
DRA12	11.600	Low Fat	0.041178	Soft Drinks	140.3154	OUT(
DRA24	19.350	Regular	0.040154	Soft Drinks	164.6868	OUT(
DRA59	8.270	Regular	0.127928	Soft Drinks	184.8924	OUT(
DRB01	7.390	Low Fat	0.082367	Soft Drinks	187.7530	OUT(
DRB13	6.115	Regular	0.007084	Soft Drinks	191.1530	OUT(

Crosstab Function :

In [17]:
```
1 pd.crosstab(df['Item_Weight'],df['Item_Identifier']).head()
```

Out[17]:

Item_Identifier	DRA12	DRA24	DRA59	DRB01	DRB13	DRB24	DRB25	DRB48	DRC01	DRC12
Item_Weight										
4.555	0	0	0	0	0	0	0	0	0	0
4.590	0	0	0	0	0	0	0	0	0	0
4.610	0	0	0	0	0	0	0	0	0	0
4.615	0	0	0	0	0	0	0	0	0	0
4.635	0	0	0	0	0	0	0	0	0	0

Feature Selection :

i) Linearity :

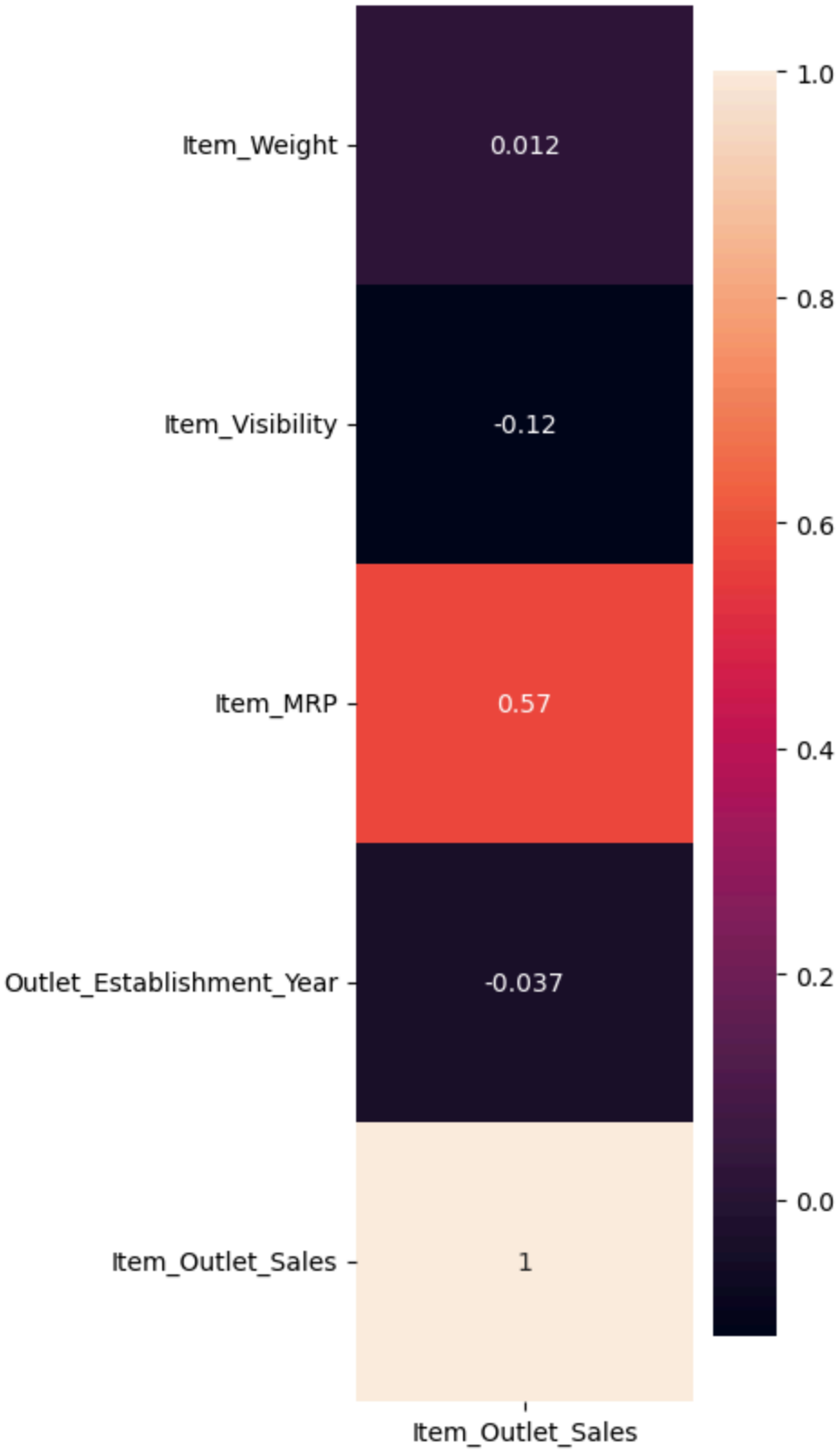
In [19]:
```
1 r = df.corr()[['Item_Outlet_Sales']]
2 r
```

Out[19]:

	Item_Outlet_Sales
Item_Weight	0.012370
Item_Visibility	-0.120418
Item_MRP	0.574554
Outlet_Establishment_Year	-0.037133
Item_Outlet_Sales	1.000000

In [20]:
```
1 plt.figure(figsize = (3,10))
2 sns.heatmap(r,annot = True)
```

Out[20]: <Axes: >



ii) Multicollinearity :

In [25]:
```
df1 = df.drop('Item_Outlet_Sales',axis = 1)
df1.head()
```

Out[25]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Ide
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	O
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	O
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	O
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	O
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	O

In [28]:
```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

```
vif_list = []

for i in range(df1.shape[1]) :
    vif = variance_inflation_factor(df1,i)
    vif_list.append(vif)

print(vif_list)
```

Model Building :

In [35]:
```
from sklearn.linear_model import LinearRegression
```

In [31]:
```
x = df.drop('Item_Outlet_Sales',axis = 1)
y = df['Item_Outlet_Sales']
```

In [32]:
```
from sklearn.model_selection import train_test_split
```

In [38]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size =0.2 , rand
```

In [36]:
```
model = LinearRegression()
```

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

```
y_pred = model.predict(x_test)
```

```
2 y_pred_train = model.predict(x_train)
```

In [ ]:

```
1
```