In [1]:

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

In [2]:

```
df1 = pd.read_csv('train.csv')
df2 = pd.read_csv('test.csv')
```

In [3]:

```
df1.head()
```

Out[3]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet_Si
0	FDA15	9.30	Low Fat	0.016047	Dairy	249.8092	OUT049	1999	Mediu
1	DRC01	5.92	Regular	0.019278	Soft Drinks	48.2692	OUT018	2009	Mediu
2	FDN15	17.50	Low Fat	0.016760	Meat	141.6180	OUT049	1999	Mediu
3	FDX07	19.20	Regular	0.000000	Fruits and Vegetables	182.0950	OUT010	1998	Na
4	NCD19	8.93	Low Fat	0.000000	Household	53.8614	OUT013	1987	Hi

In [4]:

```
df1.tail()
```

Out[4]:

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility	Item_Type	Item_MRP	Outlet_Identifier	Outlet_Establishment_Year	Outlet
8518	FDF22	6.865	Low Fat	0.056783	Snack Foods	214.5218	OUT013	1987	
8519	FDS36	8.380	Regular	0.046982	Baking Goods	108.1570	OUT045	2002	
8520	NCJ29	10.600	Low Fat	0.035186	Health and Hygiene	85.1224	OUT035	2004	
8521	FDN46	7.210	Regular	0.145221	Snack Foods	103.1332	OUT018	2009	Me
8522	DRG01	14.800	Low Fat	0.044878	Soft Drinks	75.4670	OUT046	1997	

In [5]:

```
df1.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 [6]:

```
df1.isnull().sum()
```

Out[6]:

```
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
```

In [7]:

```
df1.shape, df2.shape # ((8523, 12), (5681, 11))
```

Out[7]:

```
((8523, 12), (5681, 11))
```

In [8]:
```
df1.isnull().sum()
```

Out[8]:
```
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
```

In [9]:
```
df2.isnull().sum()
```

Out[9]:
```
Item_Identifier                 0
Item_Weight                   976
Item_Fat_Content                0
Item_Visibility                 0
Item_Type                       0
Item_MRP                        0
Outlet_Identifier               0
Outlet_Establishment_Year       0
Outlet_Size                  1606
Outlet_Location_Type            0
Outlet_Type                     0
dtype: int64
```

In [10]:
```
df1.columns
```

Out[10]:
```
Index(['Item_Identifier', 'Item_Weight', 'Item_Fat_Content', 'Item_Visibility',
       'Item_Type', 'Item_MRP', 'Outlet_Identifier',
       'Outlet_Establishment_Year', 'Outlet_Size', 'Outlet_Location_Type',
       'Outlet_Type', 'Item_Outlet_Sales'],
      dtype='object')
```

In [11]:
```
for i in df1.columns:
  print(f"{i} : {df1[i].nunique()}")
```

```
Item_Identifier : 1559
Item_Weight : 415
Item_Fat_Content : 5
Item_Visibility : 7880
Item_Type : 16
Item_MRP : 5938
Outlet_Identifier : 10
Outlet_Establishment_Year : 9
Outlet_Size : 3
Outlet_Location_Type : 3
Outlet_Type : 4
Item_Outlet_Sales : 3493
```

In [12]:
```
for i in df1.columns:
  print(f"{i} : {df1[i].dtype}")
```

```
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
```

In [13]:
```
cat_columns = []
num_columns = []
for i in df1.columns:
    if df1[i].dtype == object:
        cat_columns.append(i)
    else:
        num_columns.append(i)
```

In [14]:
```
cat_columns
```

Out[14]:
```
['Item_Identifier',
 'Item_Fat_Content',
 'Item_Type',
 'Outlet_Identifier',
 'Outlet_Size',
 'Outlet_Location_Type',
 'Outlet_Type']
```

In [15]:
```
num_columns
```

```
In [15]: num_columns
```

```
Out[15]: ['Item_Weight',
          'Item_Visibility',
          'Item_MRP',
          'Outlet_Establishment_Year',
          'Item_Outlet_Sales']
```

```
In [16]: df1['Item_Weight'].fillna(df1['Item_Weight'].mean(), inplace=True)
         df2['Item_Weight'].fillna(df1['Item_Weight'].mean(), inplace=True)
```

```
In [17]: df1.isnull().sum()
```

```
Out[17]: 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                  2410
         Outlet_Location_Type            0
         Outlet_Type                     0
         Item_Outlet_Sales               0
         dtype: int64
```

```
In [18]: df2.isnull().sum()
```

```
Out[18]: 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                  1606
         Outlet_Location_Type            0
         Outlet_Type                     0
         dtype: int64
```

```
In [19]: df1['Outlet_Size'].mode()[0]
```

```
Out[19]: 'Medium'
```

```
In [20]: df1['Outlet_Size'].fillna(df1['Outlet_Size'].mode()[0], inplace=True)
         df2['Outlet_Size'].fillna(df1['Outlet_Size'].mode()[0], inplace=True)
```

```
In [21]: df1.isnull().sum()
```

```
Out[21]: 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 [22]: df2.isnull().sum()
```

```
Out[22]: 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
         dtype: int64
```

```
In [23]: df1[cat_columns].head()
```

Out[23]:

	Item_Identifier	Item_Fat_Content	Item_Type	Outlet_Identifier	Outlet_Size	Outlet_Location_Type	Outlet_Type
0	FDA15	Low Fat	Dairy	OUT049	Medium	Tier 1	Supermarket Type1
1	DRC01	Regular	Soft Drinks	OUT018	Medium	Tier 3	Supermarket Type2
2	FDN15	Low Fat	Meat	OUT049	Medium	Tier 1	Supermarket Type1
3	FDX07	Regular	Fruits and Vegetables	OUT010	Medium	Tier 3	Grocery Store
4	NCD19	Low Fat	Household	OUT013	High	Tier 3	Supermarket Type1

In [24]:

```
df1[cat_columns].nunique()
```

Out[24]:

```
Item_Identifier         1559
Item_Fat_Content           5
Item_Type                 16
Outlet_Identifier         10
Outlet_Size                3
Outlet_Location_Type       3
Outlet_Type                4
dtype: int64
```

In [25]:

```
df1['Item_Fat_Content'].value_counts()
```

Out[25]:

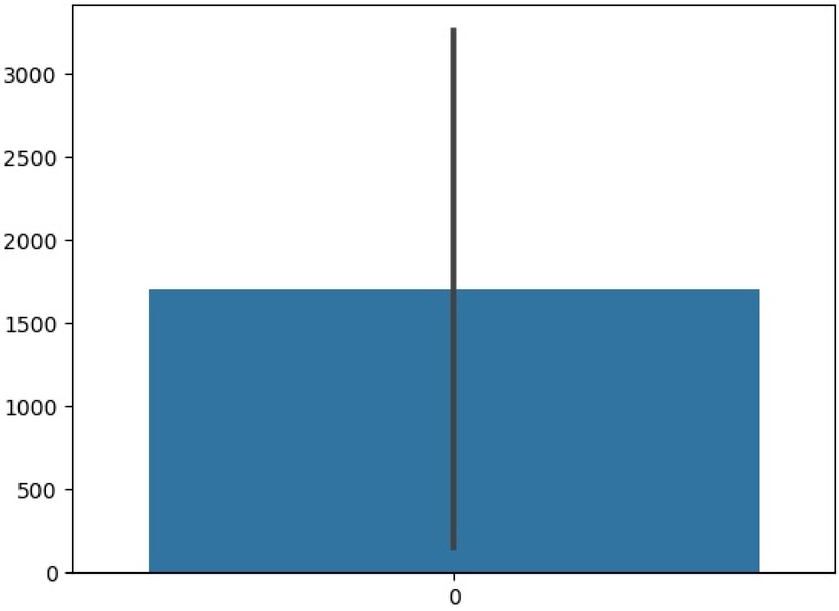
```
Item_Fat_Content
Low Fat    5089
Regular    2889
LF          316
reg         117
low fat     112
Name: count, dtype: int64
```

In [33]:

```
sns.barplot(df1['Item_Fat_Content'].value_counts())
```

Out[33]:

```
<Axes: >
```



In [33]:

```
df1['Item_Fat_Content'].replace({'LF': 'Low Fat', 'low fat': 'Low Fat', 'reg': 'Regular'}, inplace=True)
df2['Item_Fat_Content'].replace({'LF': 'Low Fat', 'low fat': 'Low Fat', 'reg': 'Regular'}, inplace=True)
```

In [34]:

```
df1['Item_Fat_Content'].value_counts()
```

Out[34]:

```
Item_Fat_Content
Low Fat    5517
Regular    3006
Name: count, dtype: int64
```

In [35]:

```
df2['Item_Fat_Content'].value_counts()
```

Out[35]:

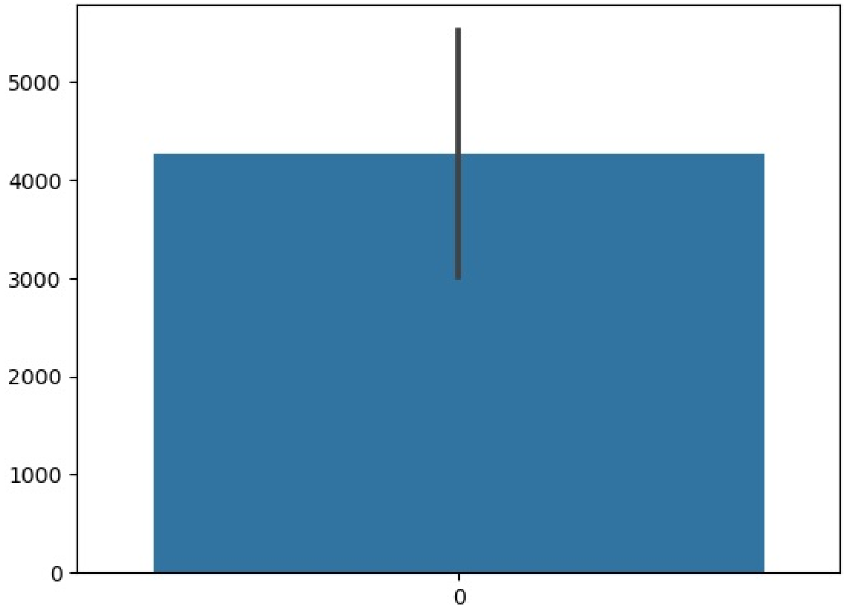
```
Item_Fat_Content
Low Fat    3668
Regular    2013
Name: count, dtype: int64
```

In [36]:

```
sns.barplot(df1['Item_Fat_Content'].value_counts())
```

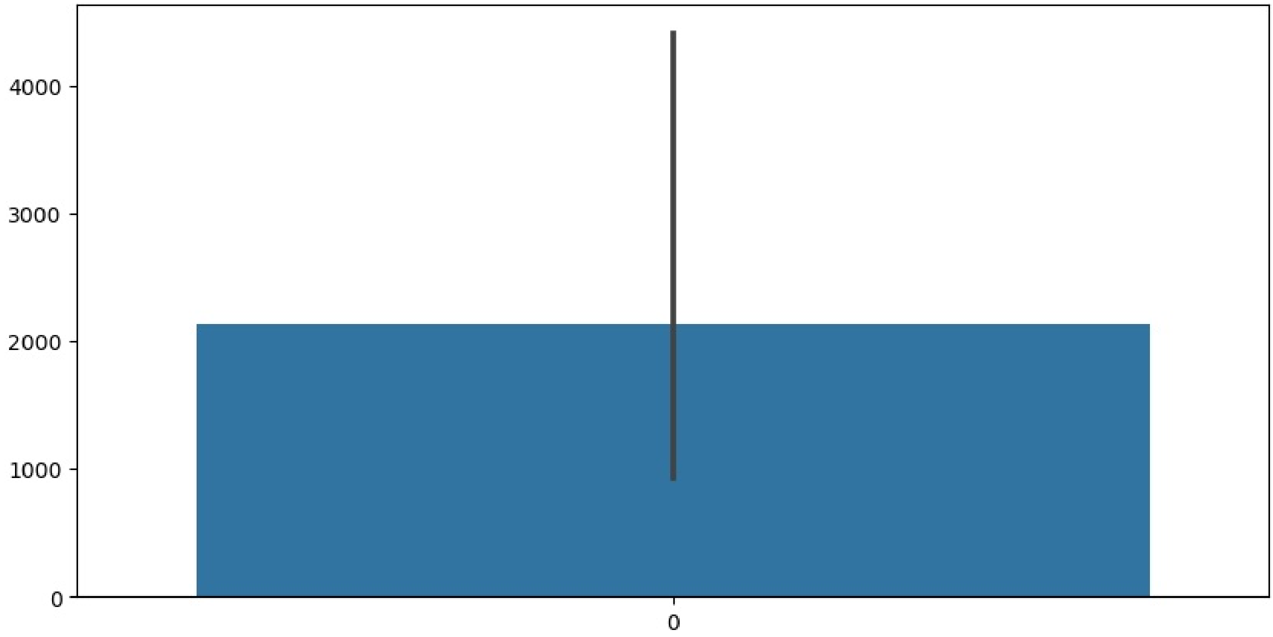
Out[36]:

```
<Axes: >
```



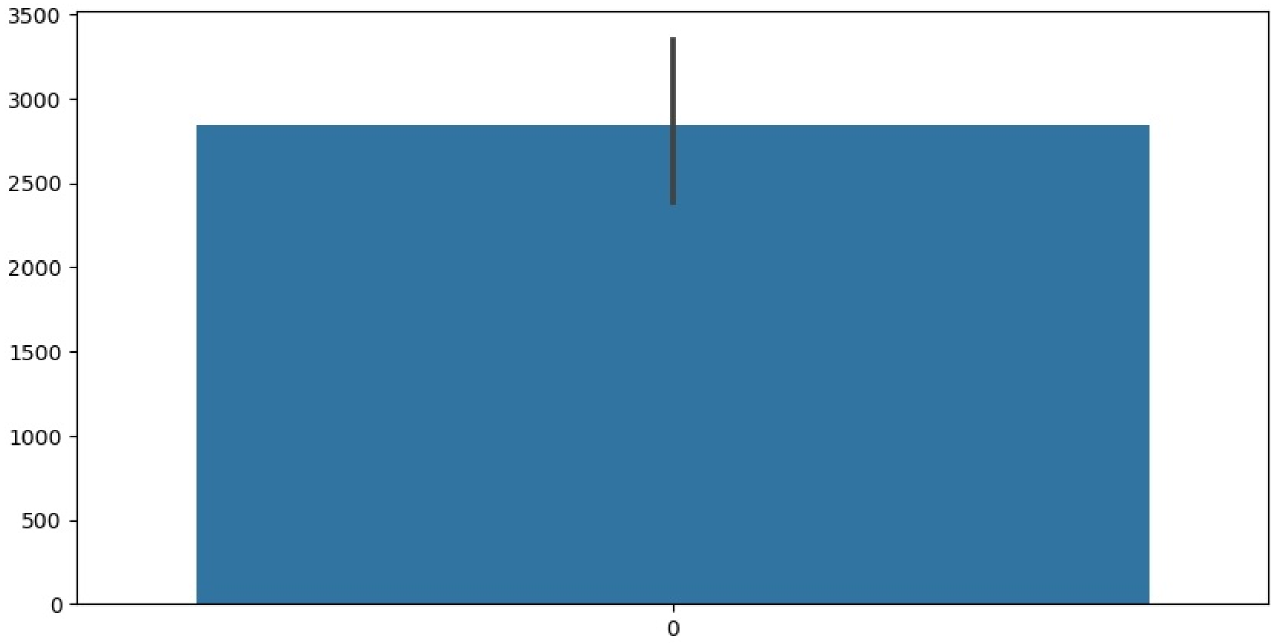
```
In [37]: plt.figure(figsize=(10, 5))
         sns.barplot(df1['Outlet_Type'].value_counts())
```

```
Out[37]: <Axes: >
```



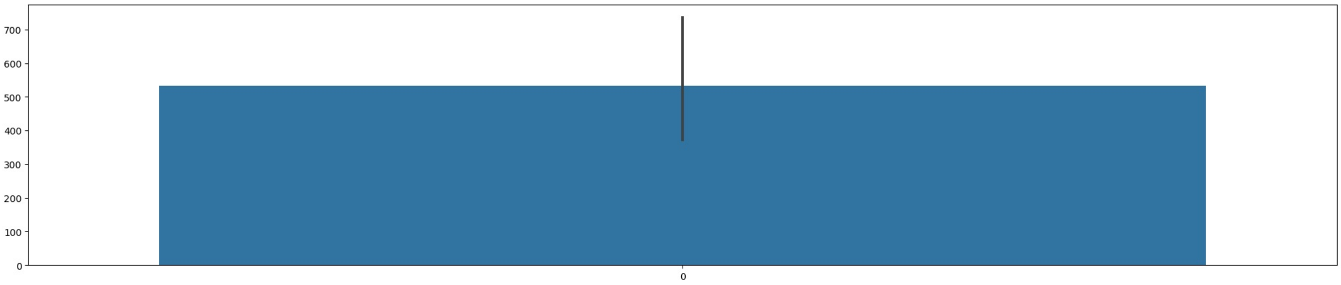
```
In [38]: plt.figure(figsize=(10, 5))
         sns.barplot(df1['Outlet_Location_Type'].value_counts())
```

```
Out[38]: <Axes: >
```



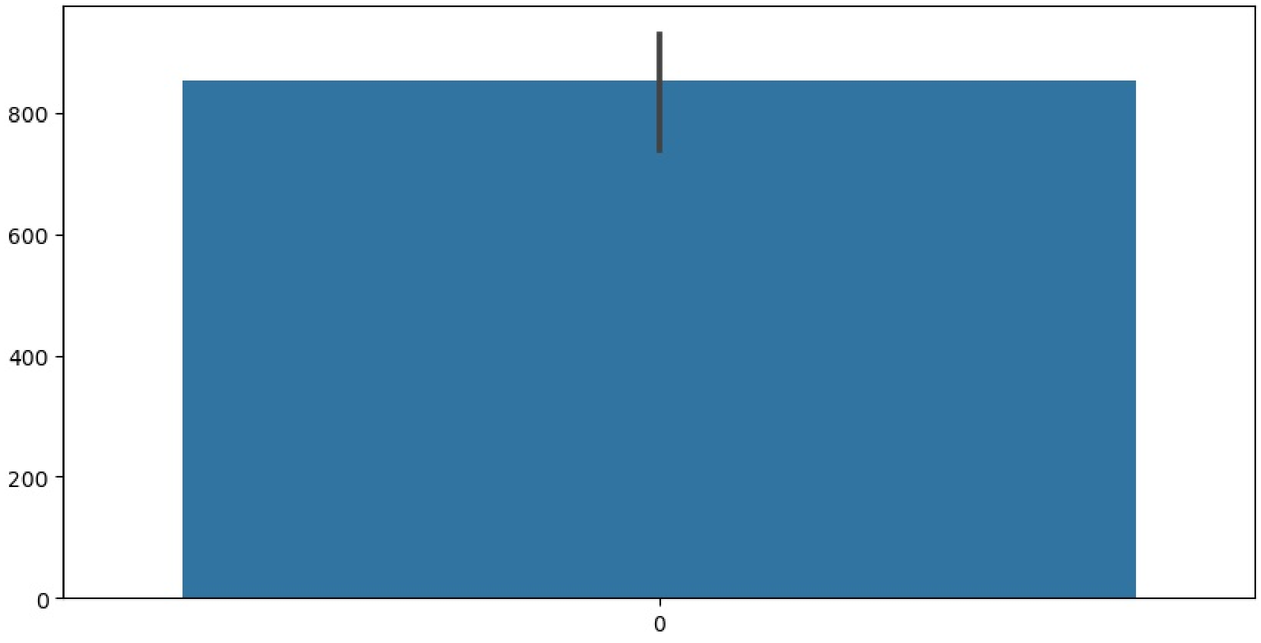
```
In [41]: plt.figure(figsize=(25, 5))
         sns.barplot(df1['Item_Type'].value_counts())

Out[41]: <Axes: >
```



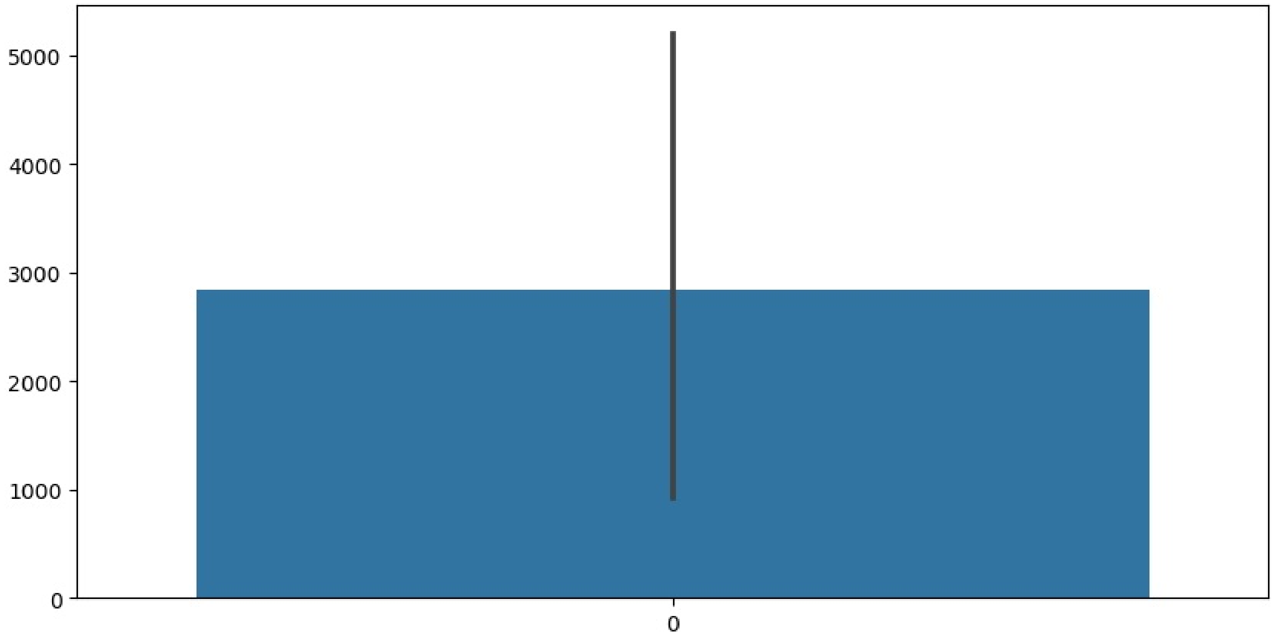
```
In [42]: plt.figure(figsize=(10, 5))
         sns.barplot(df1['Outlet_Identifier'].value_counts())

Out[42]: <Axes: >
```



In [43]:
```
plt.figure(figsize=(10, 5))
sns.barplot(df1['Outlet_Size'].value_counts())
```

Out[43]:
```
<Axes: >
```



In [44]:
```
df1[num_columns]
```

Out[44]:

	Item_Weight	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Item_Outlet_Sales
0	9.300	0.016047	249.8092	1999	3735.1380
1	5.920	0.019278	48.2692	2009	443.4228
2	17.500	0.016760	141.6180	1999	2097.2700
3	19.200	0.000000	182.0950	1998	732.3800
4	8.930	0.000000	53.8614	1987	994.7052
...	...	...	...	...	...
8518	6.865	0.056783	214.5218	1987	2778.3834
8519	8.380	0.046982	108.1570	2002	549.2850
8520	10.600	0.035186	85.1224	2004	1193.1136
8521	7.210	0.145221	103.1332	2009	1845.5976
8522	14.800	0.044878	75.4670	1997	765.6700

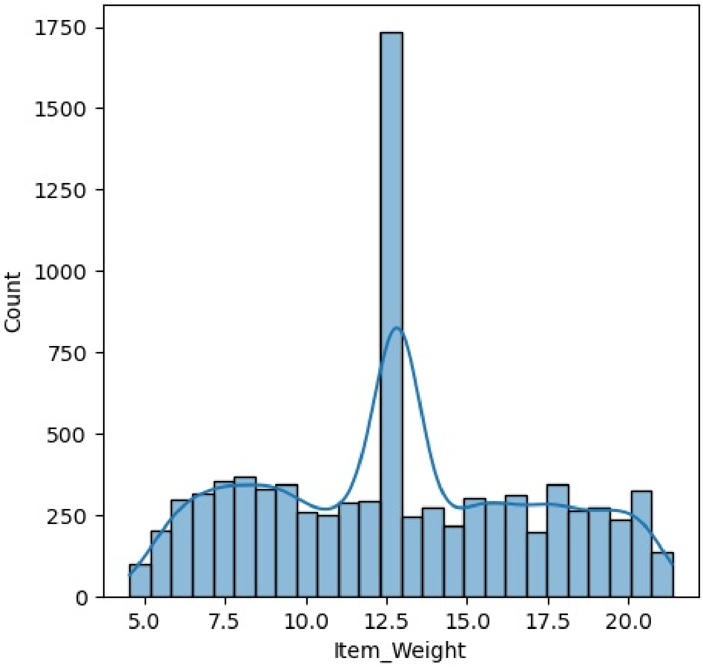
8523 rows × 5 columns

In [45]:

```
plt.figure(figsize=(5, 5))
sns.histplot(df1['Item_Weight'], kde=True)
```

Out[45]:

```
<Axes: xlabel='Item_Weight', ylabel='Count'>
```

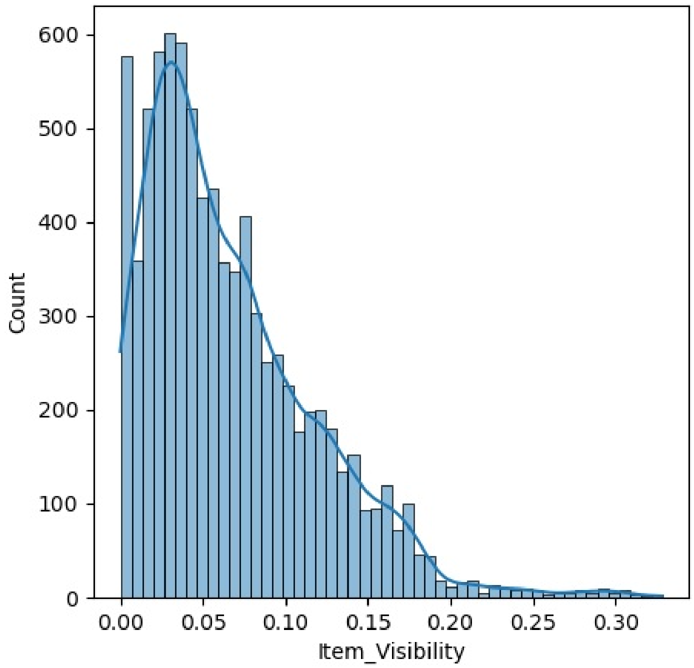


In [46]:

```
plt.figure(figsize=(5, 5))
sns.histplot(df1['Item_Visibility'], kde=True)
```

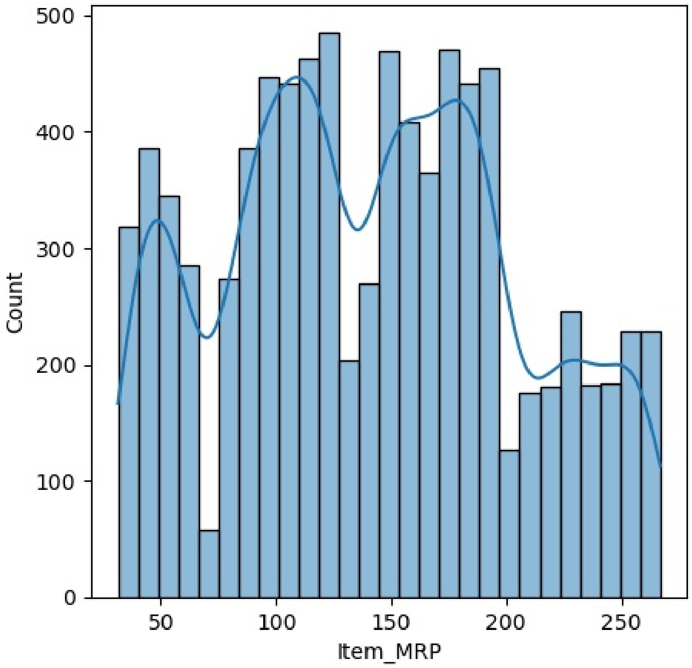
Out[46]:

```
<Axes: xlabel='Item_Visibility', ylabel='Count'>
```



```
In [47]: plt.figure(figsize=(5, 5))
         sns.histplot(df1['Item_MRP'], kde=True)
```

```
Out[47]: <Axes: xlabel='Item_MRP', ylabel='Count'>
```

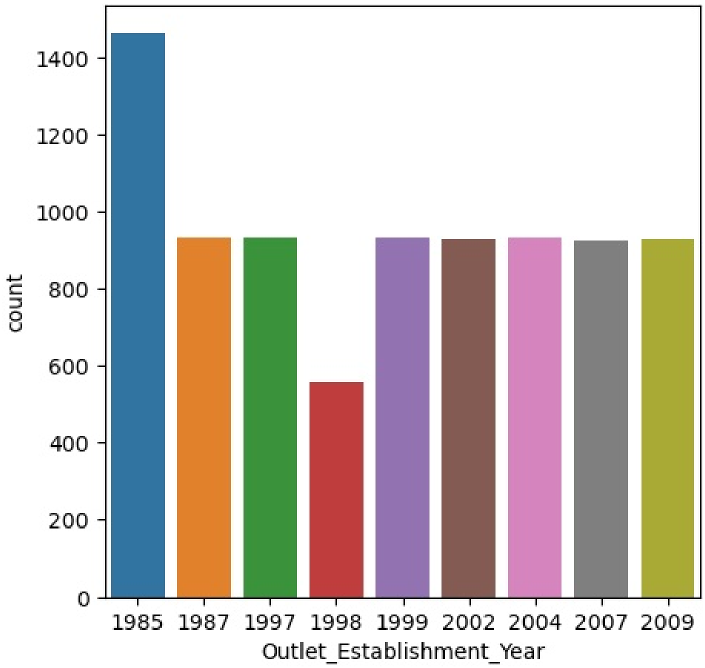


In [48]:

```
plt.figure(figsize=(5, 5))
sns.countplot(x='Outlet_Establishment_Year', data=df1)
```

Out[48]:

```
<Axes: xlabel='Outlet_Establishment_Year', ylabel='count'>
```

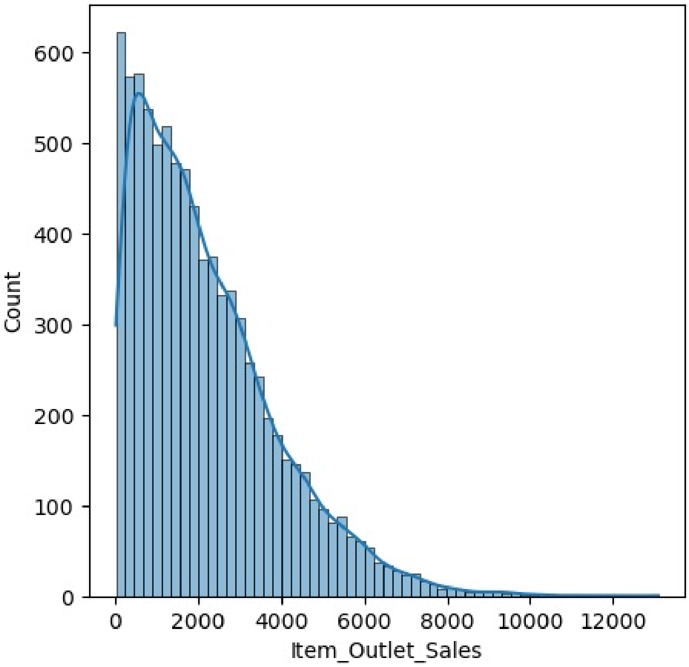


In [49]:

```
plt.figure(figsize=(5, 5))
sns.histplot(df1['Item_Outlet_Sales'], kde=True)
```

Out[49]:

```
<Axes: xlabel='Item_Outlet_Sales', ylabel='Count'>
```



In [50]:

```
df1[cat_columns].nunique()
```

Out[50]:

```
Item_Identifier         1559
Item_Fat_Content           2
Item_Type                 16
Outlet_Identifier         10
Outlet_Size                3
Outlet_Location_Type       3
Outlet_Type                4
dtype: int64
```

In [51]:

```
df1[cat_columns]
```

Out[51]:

	Item_Identifier	Item_Fat_Content	Item_Type	Outlet_Identifier	Outlet_Size	Outlet_Location_Type	Outlet_Type
0	FDA15	Low Fat	Dairy	OUT049	Medium	Tier 1	Supermarket Type1
1	DRC01	Regular	Soft Drinks	OUT018	Medium	Tier 3	Supermarket Type2
2	FDN15	Low Fat	Meat	OUT049	Medium	Tier 1	Supermarket Type1
3	FDX07	Regular	Fruits and Vegetables	OUT010	Medium	Tier 3	Grocery Store
4	NCD19	Low Fat	Household	OUT013	High	Tier 3	Supermarket Type1
...	...	...	...	...	...	...	...
8518	FDF22	Low Fat	Snack Foods	OUT013	High	Tier 3	Supermarket Type1
8519	FDS36	Regular	Baking Goods	OUT045	Medium	Tier 2	Supermarket Type1
8520	NCJ29	Low Fat	Health and Hygiene	OUT035	Small	Tier 2	Supermarket Type1
8521	FDN46	Regular	Snack Foods	OUT018	Medium	Tier 3	Supermarket Type2
8522	DRG01	Low Fat	Soft Drinks	OUT046	Small	Tier 1	Supermarket Type1

8523 rows × 7 columns

In [52]:

```
df1['Item_Identifier'] = df1['Item_Identifier'].str[:2]
df2['Item_Identifier'] = df2['Item_Identifier'].str[:2]
```

In [53]:

```
df1[cat_columns].nunique()
```

Out[53]:

```
Item_Identifier          3
Item_Fat_Content         2
Item_Type               16
Outlet_Identifier       10
Outlet_Size              3
Outlet_Location_Type     3
Outlet_Type              4
dtype: int64
```

In [54]:

```
df1[cat_columns].head()
```

Out[54]:

	Item_Identifier	Item_Fat_Content	Item_Type	Outlet_Identifier	Outlet_Size	Outlet_Location_Type	Outlet_Type
0	FD	Low Fat	Dairy	OUT049	Medium	Tier 1	Supermarket Type1
1	DR	Regular	Soft Drinks	OUT018	Medium	Tier 3	Supermarket Type2
2	FD	Low Fat	Meat	OUT049	Medium	Tier 1	Supermarket Type1
3	FD	Regular	Fruits and Vegetables	OUT010	Medium	Tier 3	Grocery Store
4	NC	Low Fat	Household	OUT013	High	Tier 3	Supermarket Type1

Now it looks good.

I am going to separate the categorical columns into two different categories. As you can see from the above output, we got 7 categorical columns in which 3 columns are ordinal (which means they have a certain order. For example, if you take grade, we know A is first, B is second, and C is third it’s an order those types of columns are also knowns as ordinal columns).

The ordinal columns are ‘Item_Fat_Content’, ‘Outlet_Size’, and ‘Outlet_Location_Type’. The rest of the columns are nominal columns because they don’t any ordering in them.

I am going to apply the Ordinal Encoder technique to ordinal categorical columns. And One-Hot Encoding for nominal categorical columns. Below is the code.

In [55]:

```
ordinal_cat_columns = ['Item_Fat_Content', 'Outlet_Size', 'Outlet_Location_Type']
nominal_cat_columns = ['Item_Identifier', 'Item_Type', 'Outlet_Identifier', 'Outlet_Type']
```

In [56]:

```
# One-Hot Encoding using get_dummies()
df1 = pd.get_dummies(df1, columns=nominal_cat_columns)
df2 = pd.get_dummies(df2, columns=nominal_cat_columns)
```

In [57]:

```
from sklearn.preprocessing import OrdinalEncoder

# Initialize the OrdinalEncoder
ordinal_encoder = OrdinalEncoder()

# Define the ordinal columns
ordinal_cols = ['Item_Fat_Content', 'Outlet_Size', 'Outlet_Location_Type']

# Fit and transform the ordinal columns
df1[ordinal_cols] = ordinal_encoder.fit_transform(df1[ordinal_cols])
df2[ordinal_cols] = ordinal_encoder.fit_transform(df2[ordinal_cols])
```

In [58]:

```
df1.head()
```

Out[58]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Item_Outlet_Sales
0	9.30	0.0	0.016047	249.8092	1999	1.0	0.0	3735.1380
1	5.92	1.0	0.019278	48.2692	2009	1.0	2.0	443.4228
2	17.50	0.0	0.016760	141.6180	1999	1.0	0.0	2097.2700
3	19.20	1.0	0.000000	182.0950	1998	1.0	2.0	732.3800
4	8.93	0.0	0.000000	53.8614	1987	0.0	2.0	994.7052

5 rows × 41 columns

In [60]:

```
y = df1['Item_Outlet_Sales']
X = df1.drop(columns=['Item_Outlet_Sales'])
```

In [61]:

```
y
```

Out[61]:

```
0       3735.1380
1        443.4228
2       2097.2700
3        732.3800
4        994.7052
          ...
8518    2778.3834
8519     549.2850
8520    1193.1136
8521    1845.5976
8522     765.6700
Name: Item_Outlet_Sales, Length: 8523, dtype: float64
```

In [62]:

```
X
```

Out[62]:

	Item_Weight	Item_Fat_Content	Item_Visibility	Item_MRP	Outlet_Establishment_Year	Outlet_Size	Outlet_Location_Type	Item_Identifier_
0	9.300	0.0	0.016047	249.8092	1999	1.0	0.0	Fa
1	5.920	1.0	0.019278	48.2692	2009	1.0	2.0	T
2	17.500	0.0	0.016760	141.6180	1999	1.0	0.0	Fa
3	19.200	1.0	0.000000	182.0950	1998	1.0	2.0	Fa
4	8.930	0.0	0.000000	53.8614	1987	0.0	2.0	Fa
...	...	...	...	...	...	...	...	
8518	6.865	0.0	0.056783	214.5218	1987	0.0	2.0	Fa
8519	8.380	1.0	0.046982	108.1570	2002	1.0	1.0	Fa
8520	10.600	0.0	0.035186	85.1224	2004	2.0	1.0	Fa
8521	7.210	1.0	0.145221	103.1332	2009	1.0	2.0	Fa
8522	14.800	0.0	0.044878	75.4670	1997	2.0	0.0	T

8523 rows × 40 columns

In [63]:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=786)
```

In [64]:

```
from xgboost import XGBRegressor
model = XGBRegressor()
model.fit(X_train, y_train)
model_prediction = model.predict(X_test)
```

Our model is ready and we have also made the prediction using the model which is stored in model_prediction variable. Let’s use the prediction and compare it with the original value and evaluate our model performance using different metrics.

Evaluating The Model

I am going to use 3 metrics. Namely,

1.Mean Absolute Error

2.Mean Squared Error

3.R2 Score

In [65]:

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
model_r2_score = r2_score(y_test, model_prediction)
model_mae_score = mean_absolute_error(y_test, model_prediction)
model_mse_score = mean_squared_error(y_test, model_prediction)

print(model_r2_score, model_mae_score, model_mse_score)
```

```
0.48053856299386644 816.0667807192451 1423839.1058956727
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js