```
import seaborn as sns
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.feature_selection import SelectKBest, f_regression
```

```
data=pd.read_csv("/content/sap-6 - Sheet1.csv")
data.tail(10)
```

	General_Health	Checkup	Exercise	Heart_Disease	Skin_Cancer	Other_Cancer	Depression	Diabetes	Arthritis	Sex	Age_Category
1933	Good	Within the past year	Yes	No	No	No	No	No	Yes	Male	70-74
1934	Fair	Within the past 2 years	Yes	No	No	No	No	No	No	Male	60-64
1935	Good	Within the past year	Yes	No	No	No	Yes	No	Yes	Female	60-64
1936	Excellent	Within the past 5 years	Yes	No	No	No	No	No	No	Female	60-64
1937	Very Good	Within the past 5 years	Yes	No	No	No	No	No	No	Male	18-24
1938	Fair	Within the past year	Yes	No	No	No	Yes	No	No	Female	65-69
1939	Poor	Within the past year	No	No	No	No	Yes	No	Yes	Female	60-64
1940	Excellent	Within the past year	Yes	No	No	No	No	No	No	Female	35-39
1941	Poor	Within the past year	Yes	Yes	No	No	Yes	Yes	Yes	Male	45-49
1942	Very Good	Within the past year	Yes	No	No	Yes	No	No	No	Female	35-39

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1943 entries, 0 to 1942
Data columns (total 19 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   General_Health               1943 non-null   object
 1   Checkup                      1943 non-null   object
 2   Exercise                     1943 non-null   object
 3   Heart_Disease                1943 non-null   object
 4   Skin_Cancer                  1943 non-null   object
 5   Other_Cancer                 1943 non-null   object
 6   Depression                   1943 non-null   object
 7   Diabetes                     1943 non-null   object
 8   Arthritis                    1928 non-null   object
 9   Sex                          1943 non-null   object
 10  Age_Category                 1943 non-null   object
 11  Height_(cm)                  1943 non-null   int64
 12  Weight_(kg)                  1943 non-null   float64
 13  BMI                          1943 non-null   float64
 14  Smoking_History              1943 non-null   object
 15  Alcohol_Consumption          1930 non-null   float64
 16  Fruit_Consumption            1943 non-null   int64
 17  Green_Vegetables_Consumption 1943 non-null   int64
 18  FriedPotato_Consumption      1943 non-null   int64
dtypes: float64(3), int64(4), object(12)
```

```
memory usage: 288.5+ KB
```

```
data.describe()
```

	Height_(cm)	Weight_(kg)	BMI	Alcohol_Consumption	Fruit_Consumption	Green_Vegetables_Consumption	FriedPotato_Consumpti
count	1943.000000	1943.000000	1943.000000	1930.000000	1943.000000	1943.000000	1943.0000
mean	169.588266	83.721544	29.022007	3.201036	24.938240	12.263510	6.3782
std	10.574843	21.391852	6.637578	6.987065	22.310036	12.941783	8.6858
min	135.000000	32.660000	14.060000	0.000000	0.000000	0.000000	0.0000
25%	163.000000	68.040000	24.335000	0.000000	8.000000	4.000000	1.0000
50%	168.000000	81.650000	28.060000	0.000000	20.000000	8.000000	4.0000
75%	178.000000	96.160000	32.345000	2.000000	30.000000	16.000000	8.0000
max	218.000000	181.440000	63.470000	30.000000	120.000000	120.000000	120.0000

```
data.isnull().sum()
```

	0
General_Health	0
Checkup	0
Exercise	0
Heart_Disease	0
Skin_Cancer	0
Other_Cancer	0
Depression	0
Diabetes	0
Arthritis	15
Sex	0
Age_Category	0
Height_(cm)	0
Weight_(kg)	0
BMI	0
Smoking_History	0
Alcohol_Consumption	13
Fruit_Consumption	0
Green_Vegetables_Consumption	0
FriedPotato_Consumption	0

dtype: int64

```
data.isnull().sum()/len(data)*100
```

	0
General_Health	0.000000
Checkup	0.000000
Exercise	0.000000
Heart_Disease	0.000000
Skin_Cancer	0.000000
Other_Cancer	0.000000
Depression	0.000000
Diabetes	0.000000
Arthritis	0.772002
Sex	0.000000
Age_Category	0.000000
Height_(cm)	0.000000
Weight_(kg)	0.000000
BMI	0.000000
Smoking_History	0.000000
Alcohol_Consumption	0.669068
Fruit_Consumption	0.000000
Green_Vegetables_Consumption	0.000000
FriedPotato_Consumption	0.000000

dtype: float64

```
num_data = data.select_dtypes(include="number")
num_data
```

	Height_(cm)	Weight_(kg)	BMI	Alcohol_Consumption	Fruit_Consumption	Green_Vegetables_Consumption	FriedPotato_Consumption
0	150	32.66	14.54	0.0	30	16	12
1	165	77.11	28.29	0.0	30	0	4
2	163	88.45	33.47	4.0	12	3	16
3	180	93.44	28.73	0.0	30	30	8
4	191	88.45	24.37	NaN	8	4	0
...	...	...	...	...	...	...	...
1938	155	77.11	32.12	0.0	28	3	3
1939	168	86.18	30.67	4.0	1	0	30
1940	163	54.43	20.60	20.0	60	4	4
1941	178	89.36	28.27	1.0	5	15	2
1942	178	80.74	25.54	4.0	20	8	4

1943 rows × 7 columns

Next steps: Generate code with num_data | View recommended plots | New interactive sheet

```
num_data.isnull().sum()
```

	0
Height_(cm)	0
Weight_(kg)	0
BMI	0
Alcohol_Consumption	13
Fruit_Consumption	0
Green_Vegetables_Consumption	0
FriedPotato_Consumption	0

dtype: int64

```
num_data.isnull().sum()/len(num_data)*100
```

	0
Height_(cm)	0.000000
Weight_(kg)	0.000000
BMI	0.000000
Alcohol_Consumption	0.669068
Fruit_Consumption	0.000000
Green_Vegetables_Consumption	0.000000
FriedPotato_Consumption	0.000000

dtype: float64

```
num_data.skew()
```

	0
Height_(cm)	0.304509
Weight_(kg)	0.823313
BMI	1.043876
Alcohol_Consumption	2.704986
Fruit_Consumption	1.618124
Green_Vegetables_Consumption	2.988103
FriedPotato_Consumption	4.500676

dtype: float64

```
from sklearn.impute import SimpleImputer


# Reshape the data to fit the imputer
data['Alcohol_Consumption'] = SimpleImputer(strategy='median').fit_transform(data[['Alcohol_Consumption']])



cat_data = data.select_dtypes(include="object")
cat_data
```

	General_Health	Checkup	Exercise	Heart_Disease	Skin_Cancer	Other_Cancer	Depression	Diabetes	Arthritis	Sex	Age_Category
0	Poor	Within the past 2 years	No	No	No	No	No	No	Yes	Female	70-74
1	Very Good	Within the past year	No	Yes	No	No	No	Yes	No	Female	70-74
2	Very Good	Within the past year	Yes	No	No	No	No	Yes	No	Female	60-64
3	Poor	Within the past year	Yes	Yes	No	No	No	Yes	No	Male	75-79
4	Good	Within the past year	No	No	No	No	No	No	No	Male	80+
...	...	...	...	...	...	...	...	...	...	...	...

Next steps: Generate code with `cat_data` View recommended plots New interactive sheet

```
cat_data.isnull().sum()
```

	0
General_Health	0
Checkup	0
Exercise	0
Heart_Disease	0
Skin_Cancer	0
Other_Cancer	0
Depression	0
Diabetes	0
Arthritis	15
Sex	0
Age_Category	0
Smoking_History	0

dtype: int64

```
data
```

	General_Health	Checkup	Exercise	Heart_Disease	Skin_Cancer	Other_Cancer	Depression	Diabetes	Arthritis	Sex	Age_Category
0	Poor	Within the past 2 years	No	No	No	No	No	No	Yes	Female	70-74
1	Very Good	Within the past year	No	Yes	No	No	No	Yes	No	Female	70-74
2	Very Good	Within the past year	Yes	No	No	No	No	Yes	No	Female	60-64
3	Poor	Within the past year	Yes	Yes	No	No	No	Yes	No	Male	75-79
4	Good	Within the past year	No	No	No	No	No	No	No	Male	80+
...	...	...	...	...	...	...	...	...	...	...	...
1938	Fair	Within the past year	Yes	No	No	No	Yes	No	No	Female	65-69
1939	Poor	Within the past year	No	No	No	No	Yes	No	Yes	Female	60-64
1940	Excellent	Within the past year	Yes	No	No	No	No	No	No	Female	35-39
1941	Poor	Within the past year	Yes	Yes	No	No	Yes	Yes	Yes	Male	45-49
1942	Very Good	Within the past year	Yes	No	No	Yes	No	No	No	Female	35-39

1943 rows × 19 columns

Next steps: Generate code with data | View recommended plots | New interactive sheet

```
arthritis_imputer = SimpleImputer(strategy='most_frequent')
data['Arthritis'] = arthritis_imputer.fit_transform(data[['Arthritis']]).flatten()
```

```
data.isnull().sum()
```

	0
General_Health	0
Checkup	0
Exercise	0
Heart_Disease	0
Skin_Cancer	0
Other_Cancer	0
Depression	0
Diabetes	0
Arthritis	0
Sex	0
Age_Category	0
Height_(cm)	0
Weight_(kg)	0
BMI	0
Smoking_History	0
Alcohol_Consumption	0
Fruit_Consumption	0
Green_Vegetables_Consumption	0
FriedPotato_Consumption	0

dtype: int64

```
num_data
```

	Height_(cm)	Weight_(kg)	BMI	Alcohol_Consumption	Fruit_Consumption	Green_Vegetables_Consumption	FriedPotato_Consumption
0	150	32.66	14.54	0.0	30	16	12
1	165	77.11	28.29	0.0	30	0	4
2	163	88.45	33.47	4.0	12	3	16
3	180	93.44	28.73	0.0	30	30	8
4	191	88.45	24.37	NaN	8	4	0
...	...	...	...	...	...	...	...
1938	155	77.11	32.12	0.0	28	3	3
1939	168	86.18	30.67	4.0	1	0	30
1940	163	54.43	20.60	20.0	60	4	4
1941	178	89.36	28.27	1.0	5	15	2
1942	178	80.74	25.54	4.0	20	8	4

1943 rows × 7 columns

Next steps: Generate code with num_data | View recommended plots | New interactive sheet

```
numeric_columns = num_data.columns
numeric_columns
```

```
Index(['Height_(cm)', 'Weight_(kg)', 'BMI', 'Alcohol_Consumption',
       'Fruit_Consumption', 'Green_Vegetables_Consumption',
       'FriedPotato_Consumption'],
      dtype='object')
```

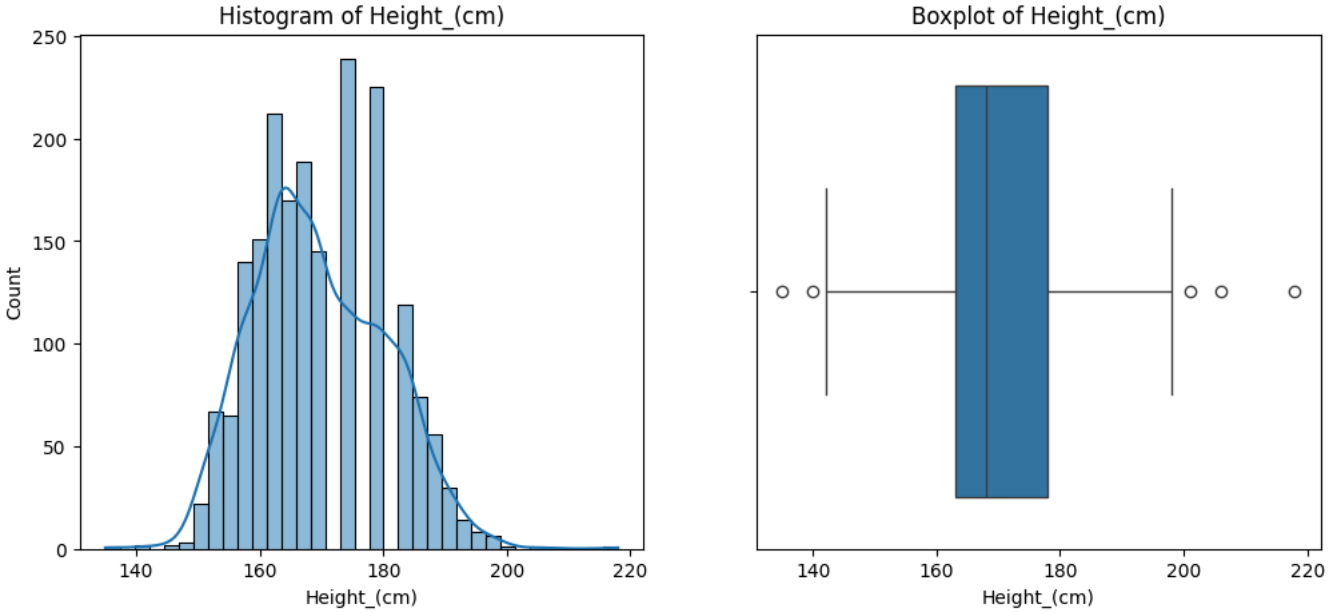
```
num_data.skew()
```

	0
Height_(cm)	0.304509
Weight_(kg)	0.823313
BMI	1.043876
Alcohol_Consumption	2.704986
Fruit_Consumption	1.618124
Green_Vegetables_Consumption	2.988103
FriedPotato_Consumption	4.500676

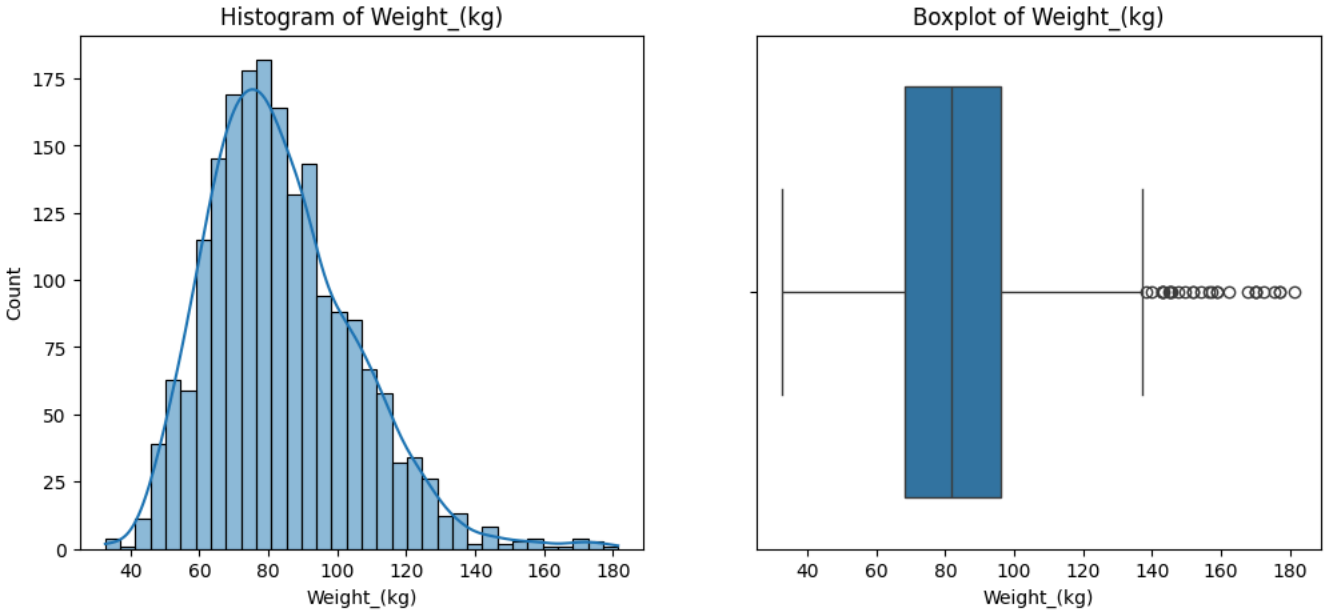
dtype: float64

```
for column in numeric_columns:
    plt.figure(figsize=(12, 5))

    # Histogram
    plt.subplot(1, 2, 1)
    sns.histplot(data[column], kde=True)
    plt.title(f'Histogram of {column}')

    # Boxplot
    plt.subplot(1, 2, 2)
    sns.boxplot(x=data[column])
    plt.title(f'Boxplot of {column}')

    plt.show()
```

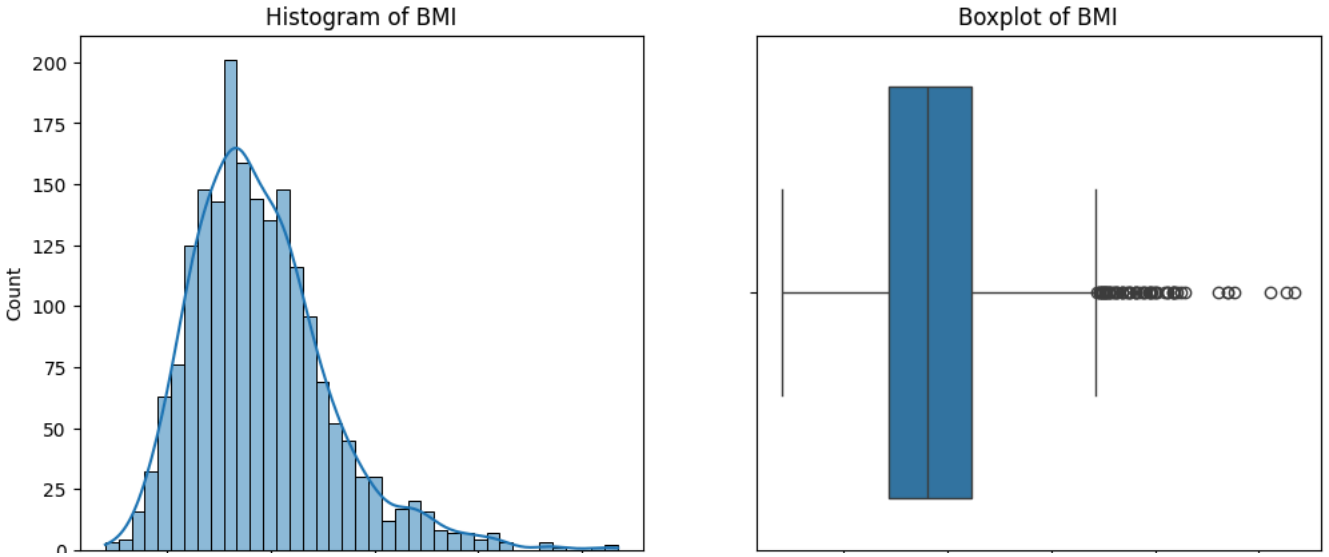
```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



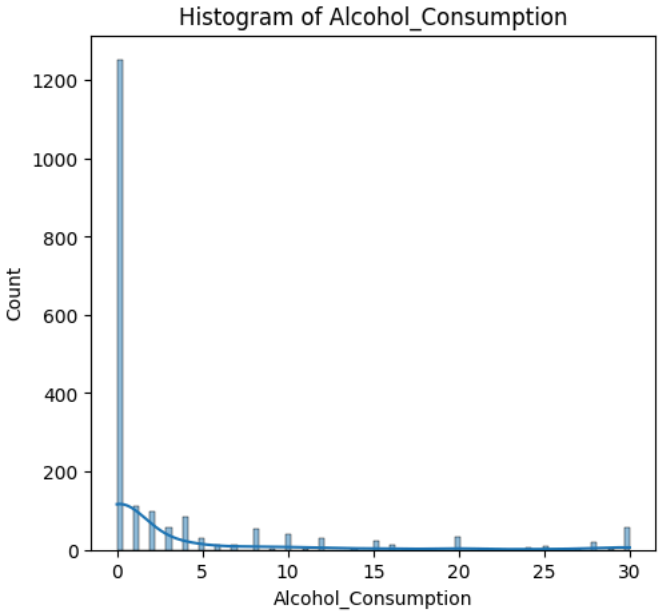

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

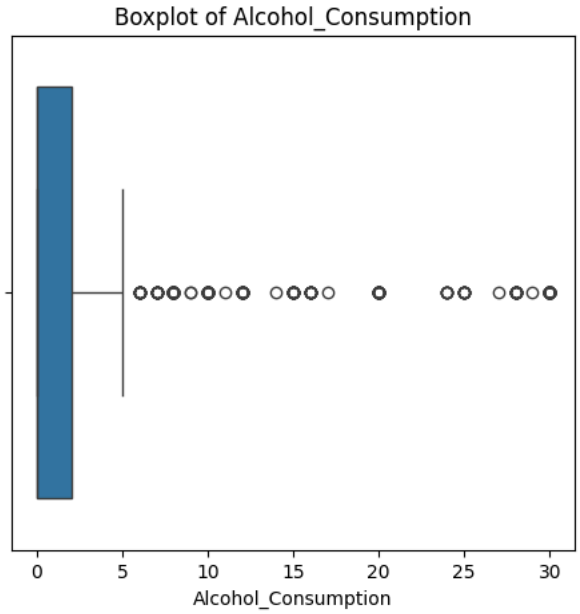



```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

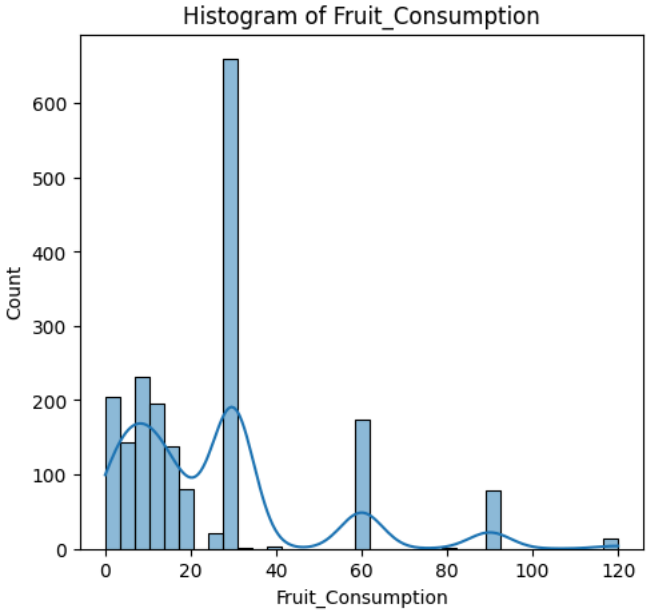


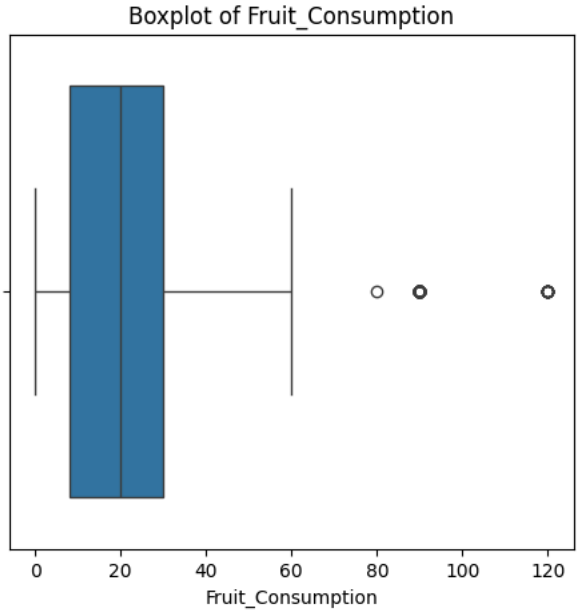

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



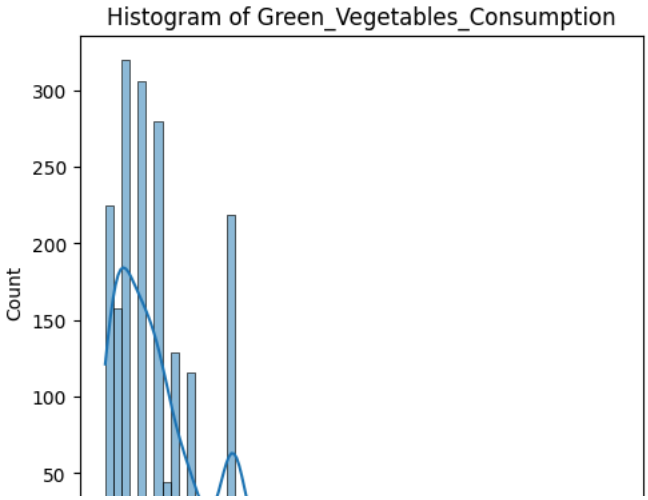


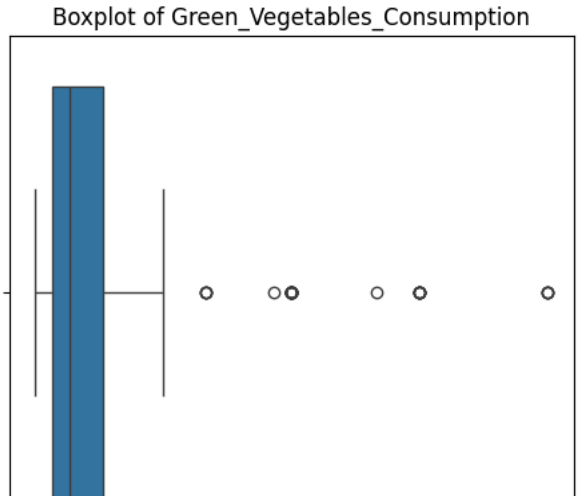

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

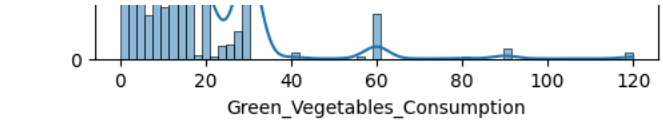


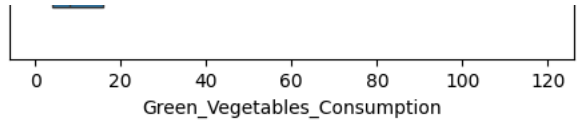



```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

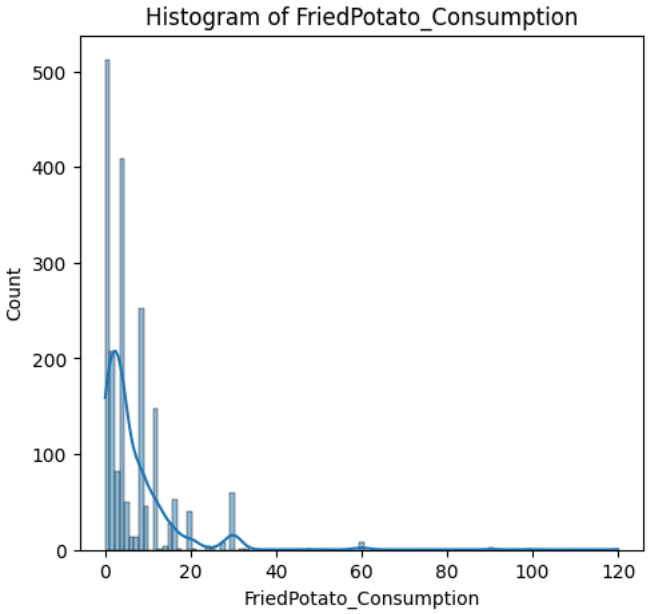


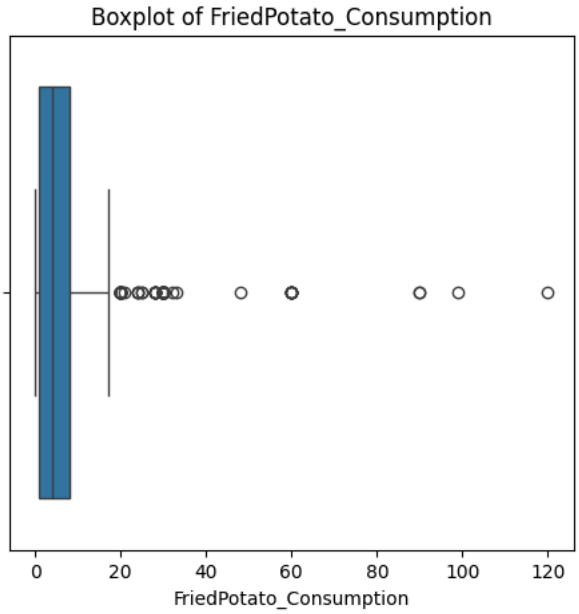






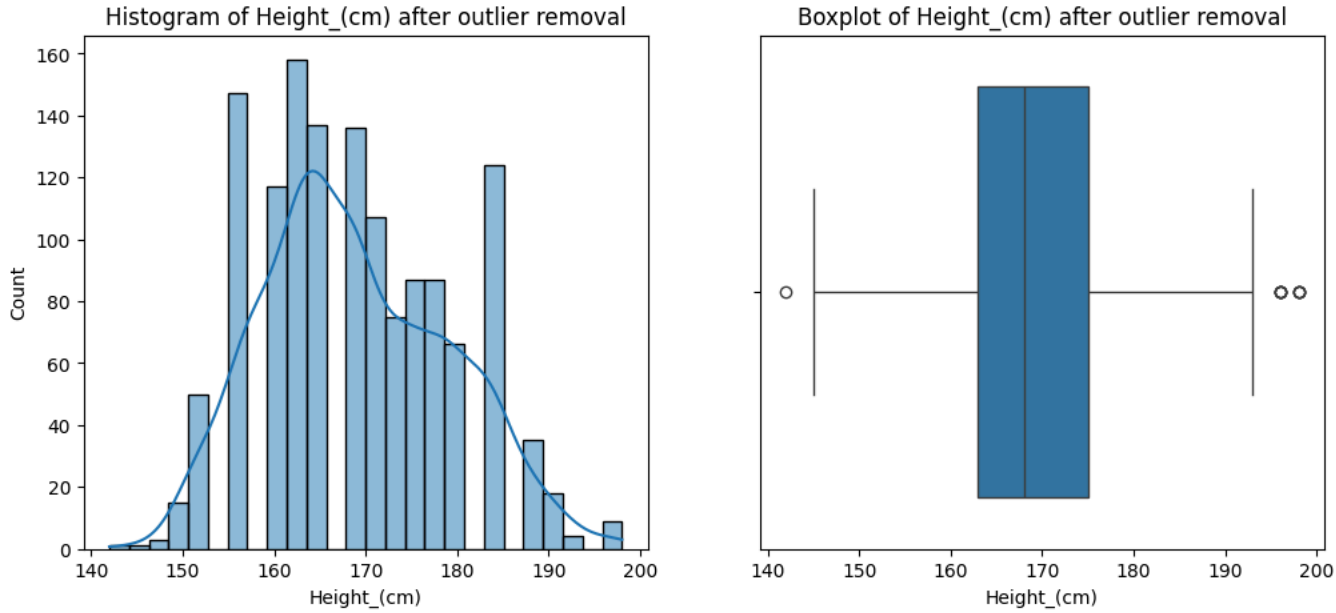

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



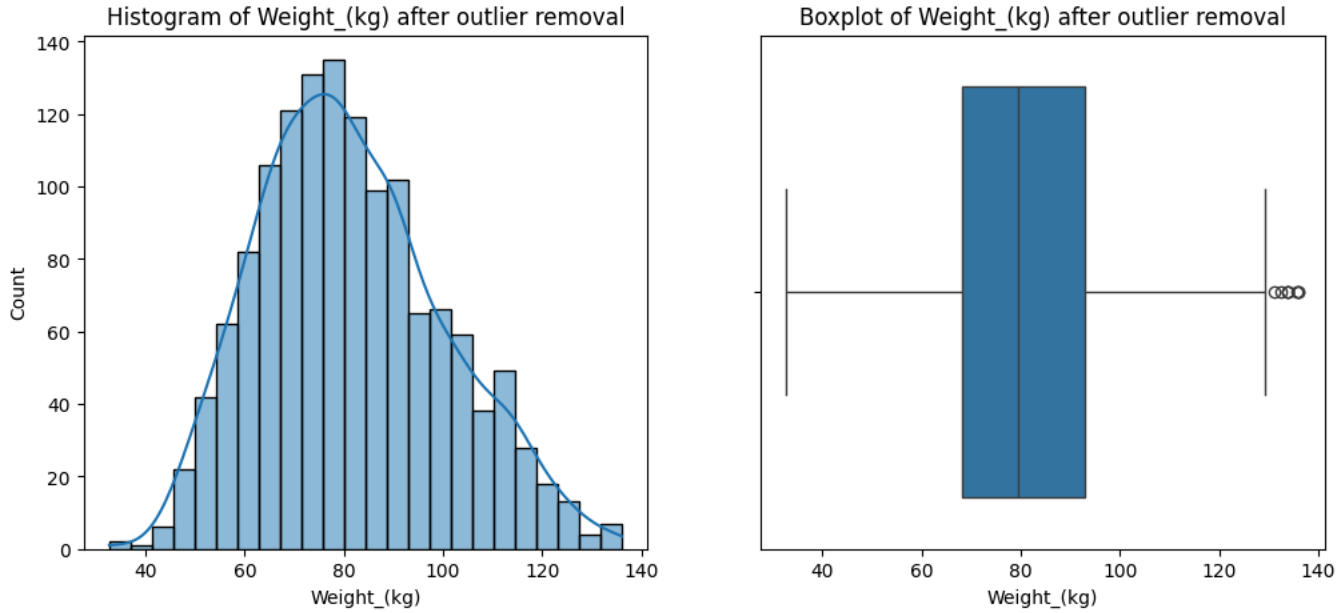


```
def remove_outliers_iqr(data, column):
    Q1 = data[column].quantile(0.25)
    Q3 = data[column].quantile(0.75)
    IQR = Q3 - Q1

    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Return a filtered DataFrame that excludes outliers
    return data[(data[column] >= lower_bound) &  (data[column] <= upper_bound)]

# Start with the original data
cleaned_data = data.copy()  # Create a copy for cleaning

# Remove outliers for each numeric column
for column in numeric_columns:
    cleaned_data = remove_outliers_iqr(cleaned_data, column)

# Visualize the cleaned data again to confirm outliers are removed
for column in numeric_columns:
    plt.figure(figsize=(12, 5))

    # Histogram
    plt.subplot(1, 2, 1)
    sns.histplot(cleaned_data[column], kde=True)
    plt.title(f'Histogram of {column} after outlier removal')

    # Boxplot
    plt.subplot(1, 2, 2)
    sns.boxplot(x=cleaned_data[column])
    plt.title(f'Boxplot of {column} after outlier removal')

    plt.show()
```

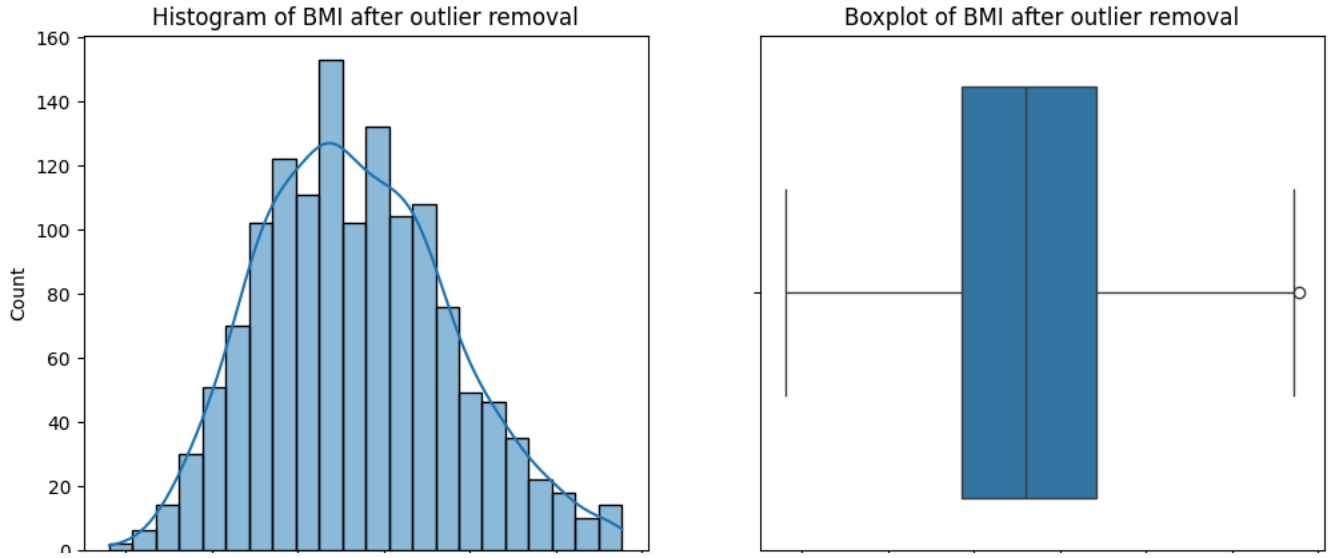
```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



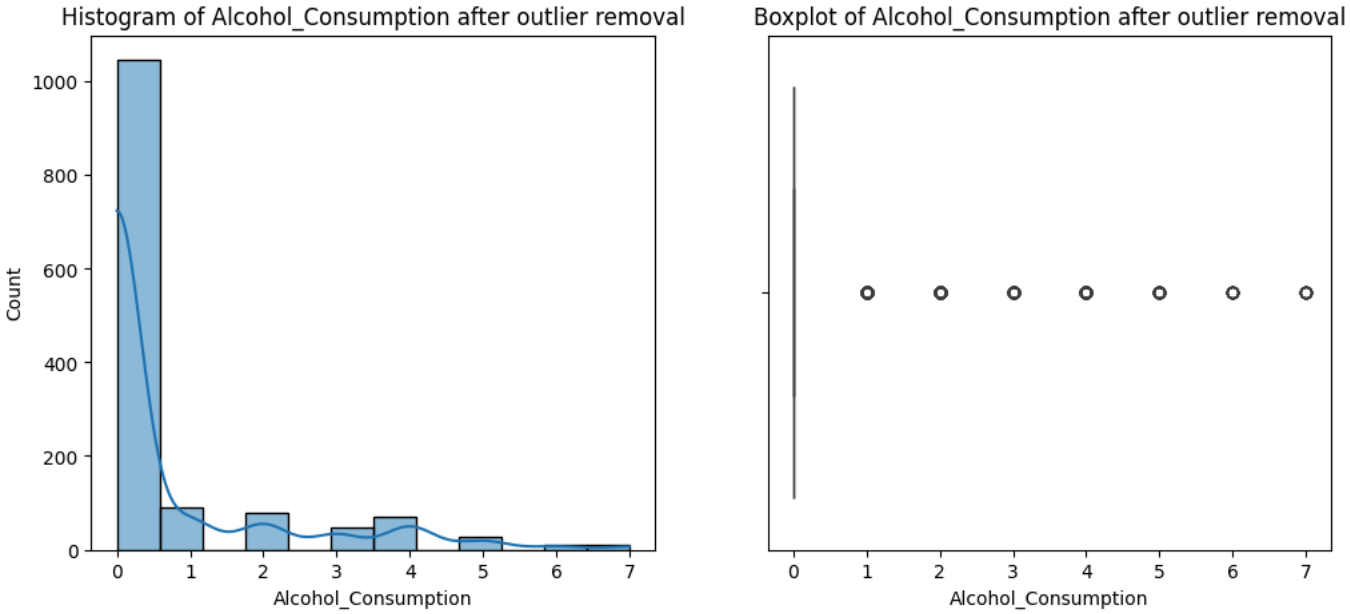

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



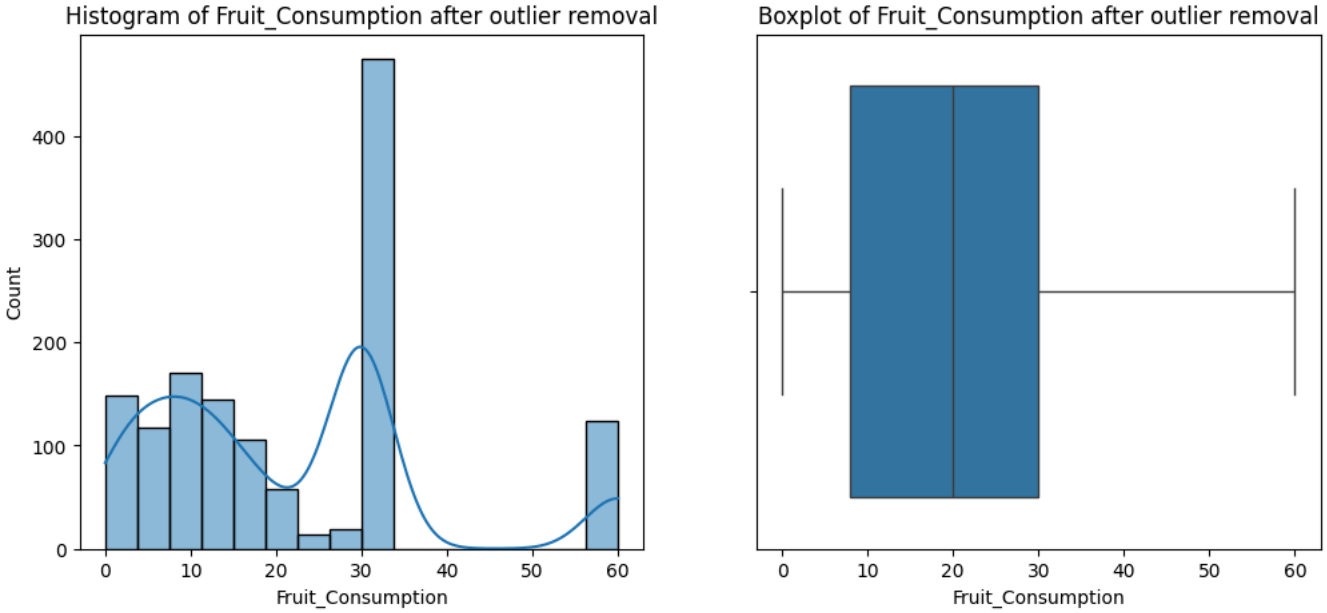

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



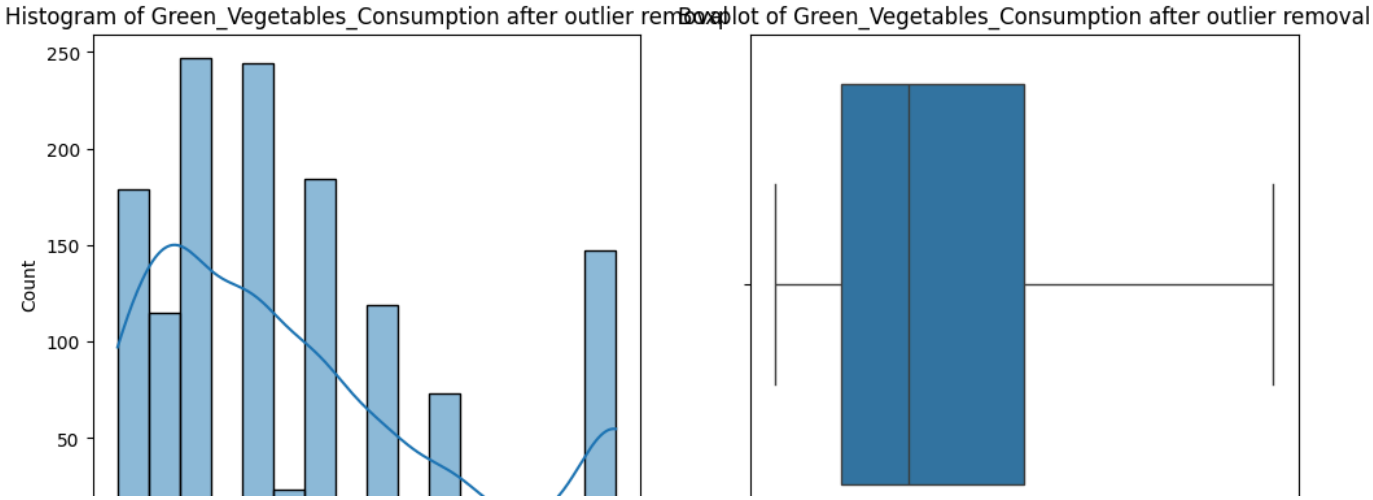

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

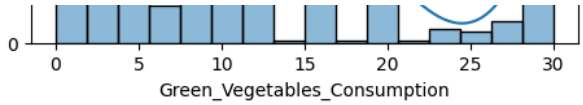


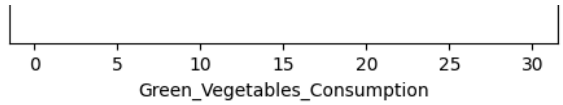

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```



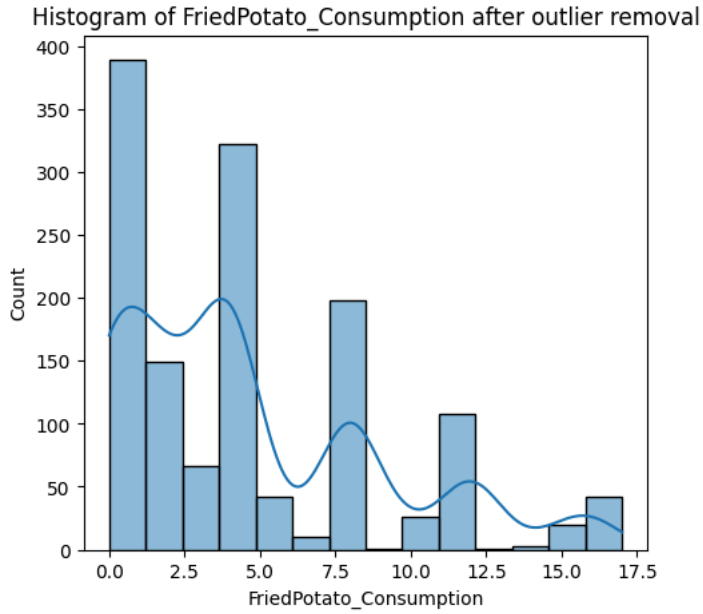

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

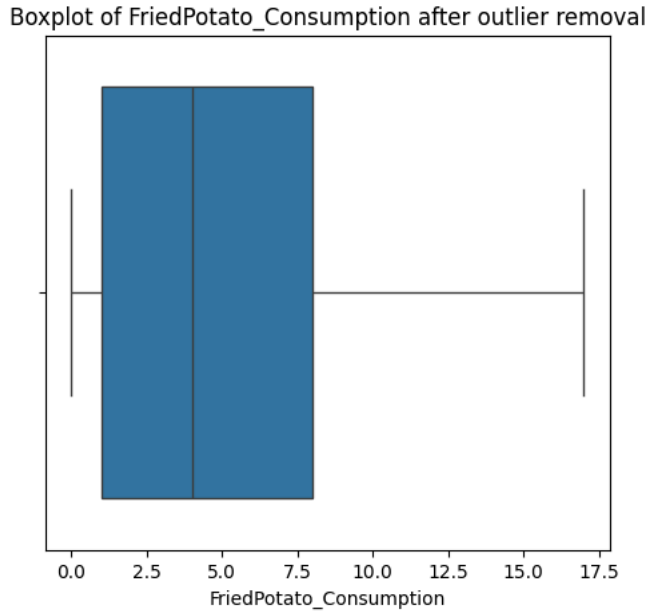







```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```





```
cleaned_data.select_dtypes("number").skew()
```

	0
Height_(cm)	0.319420
Weight_(kg)	0.395749
BMI	0.324142
Alcohol_Consumption	2.254803
Fruit_Consumption	0.908701
Green_Vegetables_Consumption	0.995776
FriedPotato_Consumption	0.985051

dtype: float64


```
cleaned_data.select_dtypes("number").skew()
```

	0
Height_(cm)	0.319420
Weight_(kg)	0.395749
BMI	0.324142
Alcohol_Consumption	2.254803
Fruit_Consumption	0.908701
Green_Vegetables_Consumption	0.995776
FriedPotato_Consumption	0.985051

dtype: float64


```
cleaned_data.shape
```

```
(1377, 19)
```

```
cleaned_data["Heart_Disease"].value_counts()
```

Heart_Disease	count
No	1202
Yes	175

dtype: int64

```
x=cleaned_data.drop("Heart_Disease",axis=1)
x
```

	General_Health	Checkup	Exercise	Skin_Cancer	Other_Cancer	Depression	Diabetes	Arthritis	Sex	Age_Category	Height_(cm)	W
0	Poor	Within the past 2 years	No	No	No	No	No	Yes	Female	70-74	150	
1	Very Good	Within the past year	No	No	No	No	Yes	No	Female	70-74	165	
2	Very Good	Within the past year	Yes	No	No	No	Yes	No	Female	60-64	163	
3	Poor	Within the past year	Yes	No	No	No	Yes	No	Male	75-79	180	
4	Good	Within the past year	No	No	No	No	No	No	Male	80+	191	
...	...	...	...	...	...	...	...	...	...	...	...	
1936	Excellent	Within the past 5 years	Yes	No	No	No	No	No	Female	60-64	155	
1937	Very Good	Within the past 5 years	Yes	No	No	No	No	No	Male	18-24	178	
1938	Fair	Within the past year	Yes	No	No	Yes	No	No	Female	65-69	155	
1941	Poor	Within the past year	Yes	No	No	Yes	Yes	Yes	Male	45-49	178	
1942	Very Good	Within the past year	Yes	No	Yes	No	No	No	Female	35-39	178	

1377 rows × 18 columns

Next steps: Generate code with x | View recommended plots | New interactive sheet

```
y=cleaned_data["Heart_Disease"]
y
```

	Heart_Disease
0	No
1	Yes
2	No
3	Yes
4	No
...	...
1936	No
1937	No
1938	No
1941	Yes
1942	No

1377 rows × 1 columns

dtype: object

```
cat_cleaned_data=x.select_dtypes(include="object")
cat_cleaned_data
```

	General_Health	Checkup	Exercise	Skin_Cancer	Other_Cancer	Depression	Diabetes	Arthritis	Sex	Age_Category	Smoking_Histor
0	Poor	Within the past 2 years	No	No	No	No	No	Yes	Female	70-74	Ye
1	Very Good	Within the past year	No	No	No	No	Yes	No	Female	70-74	N
2	Very Good	Within the past year	Yes	No	No	No	Yes	No	Female	60-64	N
3	Poor	Within the past year	Yes	No	No	No	Yes	No	Male	75-79	N
4	Good	Within the past year	No	No	No	No	No	No	Male	80+	Ye
...	...	...	...	...	...	...	...	...	...	...	

Next steps: Generate code with `cat_cleaned_data` View recommended plots New interactive sheet

```
cat_cleaned_data_columns=list(cat_cleaned_data)
cat_cleaned_data_columns
```

```
['General_Health',
 'Checkup',
 'Exercise',
 'Skin_Cancer',
 'Other_Cancer',
 'Depression',
 'Diabetes',
 'Arthritis',
 'Sex',
 'Age_Category',
 'Smoking_History']
```

```
cat_cleaned_data=x.select_dtypes(include="object")
cat_cleaned_data
```