In [5]:
```
import  pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [7]:
```
df=pd.read_csv("home.csv")
df
```

Out[7]:

	Unix Timestamp	Transaction_ID	Television	Dryer	Oven	Refrigerator	Microwave	Line Voltage	Voltage	Apparent Power	Energy Consumption (kWh)	Offloading Decision
0	1577836800	1	0	0	0	1	0	237	233	1559	24.001763	0
1	1577839322	2	0	1	0	0	1	232	230	1970	31.225154	1
2	1577841845	3	0	1	0	0	0	223	222	1684	70.460700	1
3	1577844368	4	1	0	1	1	0	225	224	1694	32.264043	1
4	1577846891	5	1	0	0	1	0	222	214	1889	32.728111	0
...	...	...	...	...	...	...	...	...	...	...	...	...
48967	1701377135	48968	1	0	1	0	0	234	230	1815	20.161006	1
48968	1701379658	48969	0	0	1	0	1	238	235	1692	91.965343	0
48969	1701382181	48970	1	0	0	1	1	235	229	1686	40.224097	0
48970	1701384704	48971	1	1	1	1	1	237	230	1754	37.366360	0
48971	1701387227	48972	0	0	0	0	1	220	215	1660	66.239065	0

48972 rows × 13 columns

understand the business problem

In [ ]:
```
# TITLE:-Smart Home for saving the electric energy
```

In [ ]:
```
# DOMAIN:-“Electricity Board”
```

In [ ]:
```
#Description:-Identify the most and least used appliances in smart homes.
            #Measure energy consumption across different devices (like Television, Oven, Refrigerator, etc.)
```

In [ ]:
```
# Objective:-
# how to save the electrictiy
# how to safe the electronics devices
```

data understanding and exploration

In [9]:
```
df.head()
```

Out[9]:

	Unix Timestamp	Transaction_ID	Television	Dryer	Oven	Refrigerator	Microwave	Line Voltage	Voltage	Apparent Power	Energy Consumption (kWh)	Offloading Decision	Ban
0	1577836800	1	0	0	0	1	0	237	233	1559	24.001763	0	
1	1577839322	2	0	1	0	0	1	232	230	1970	31.225154	1	
2	1577841845	3	0	1	0	0	0	223	222	1684	70.460700	1	
3	1577844368	4	1	0	1	1	0	225	224	1694	32.264043	1	
4	1577846891	5	1	0	0	1	0	222	214	1889	32.728111	0	

In [ ]:
```
# observe:- by default we see top 5 rows
```

In [23]:
```
df.tail()
```

Out[23]:

	Unix Timestamp	Transaction_ID	Television	Dryer	Oven	Refrigerator	Microwave	Line Voltage	Voltage	Apparent Power	Energy Consumption (kWh)	Offloading Decision
48967	1701377135	48968	1	0	1	0	0	234	230	1815	20.161006	1
48968	1701379658	48969	0	0	1	0	1	238	235	1692	91.965343	0
48969	1701382181	48970	1	0	0	1	1	235	229	1686	40.224097	0
48970	1701384704	48971	1	1	1	1	1	237	230	1754	37.366360	0
48971	1701387227	48972	0	0	0	0	1	220	215	1660	66.239065	0

In [ ]:
```
# observe:- by default we see last 5 rows
```

In [15]:
```
df.shape
```

Out[15]: (48972, 13)

In [ ]:
```
# observe:-i observe that there are 48972 rows and 13 columns
```

In [17]:
```
df.ndim
```

Out[17]: 2

In [ ]:
```
# observe:-i observe that this Dataset is a two dimensional.
```

In [11]:
```
df.size
```

Out[11]: 636636

In [ ]:
```
# observe:- i observe that there are 636636 data available  in this Dataset
```

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 48972 entries, 0 to 48971
Data columns (total 13 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Unix Timestamp            48972 non-null  int64  
 1   Transaction_ID            48972 non-null  int64  
 2   Television                48972 non-null  int64  
 3   Dryer                     48972 non-null  int64  
 4   Oven                      48972 non-null  int64  
 5   Refrigerator              48972 non-null  int64  
 6   Microwave                 48972 non-null  int64  
 7   Line Voltage              48972 non-null  int64  
 8   Voltage                   48972 non-null  int64  
 9   Apparent Power            48972 non-null  int64  
 10  Energy Consumption (kWh)  48972 non-null  float64
 11  Offloading Decision       48972 non-null  int64  
 12  Bandwidth                 48972 non-null  int64  
dtypes: float64(1), int64(12)
memory usage: 4.9 MB
```

In [ ]:
```
# Obesreve:  there is no missing value in  any columns
```

In [11]:
```
df.dtypes
```

Out[11]:
```
Unix Timestamp               int64
Transaction_ID               int64
Television                   int64
Dryer                        int64
Oven                         int64
Refrigerator                 int64
Microwave                    int64
Line Voltage                 int64
Voltage                      int64
Apparent Power               int64
Energy Consumption (kWh)   float64
Offloading Decision          int64
Bandwidth                    int64
dtype: object
```

In [ ]:
```
# observe:- here i observe that there are all continuous variable
```

In [27]:
```
df.duplicated().sum()
```

Out[27]:
```
0
```

In [ ]:
```
# Obesreve: there is no duplicates value available in this Dataset
```

In [13]:
```
df.keys().to_list()
```

Out[13]:
```
['Unix Timestamp',
 'Transaction_ID',
 'Television',
 'Dryer',
 'Oven',
 'Refrigerator',
 'Microwave',
 'Line Voltage',
 'Voltage',
 'Apparent Power',
 'Energy Consumption (kWh)',
 'Offloading Decision',
 'Bandwidth']
```

In [ ]:
```
# Explain each column
# Unix Timestamp - Time of data recording in Unix format.
# Transaction_ID - Unique ID for each record.
# Television - Power usage or status of the television.
# Dryer - Power usage or status of the dryer.
```

```
# Oven - Power usage or status of the oven.
# Refrigerator - Power usage or status of the refrigerator.
# Microwave - Power usage or status of the microwave.
# Line Voltage - Voltage supplied to the system.
# Voltage - Voltage measured at the appliance/system.
# Apparent Power - Total power (active + reactive) used.
# Energy Consumption (kWh) - Energy used in kilowatt-hours.
# Offloading Decision - Whether to offload tasks or not (0/1).
# Bandwidth - Available or used network bandwidth
```

In [19]:
```
df.isnull().sum()
```

Out[19]:
```
Unix Timestamp             0
Transaction_ID             0
Television                 0
Dryer                      0
Oven                       0
Refrigerator               0
Microwave                  0
Line Voltage               0
Voltage                    0
Apparent Power             0
Energy Consumption (kWh)   0
Offloading Decision        0
Bandwidth                  0
dtype: int64
```

In [ ]:
```
# observe : there is no null value in any column.
```

Explore each column

column=1 Unix Timestamp

In [25]:
```
# this is the continuous variable and correct data types and there is not any null value
# but there is no use of this column in the analysis simply drop this column
df["Unix Timestamp"].dtypes
```

Out[25]:
```
dtype('int64')
```

In [27]:
```
df["Unix Timestamp"].isnull().sum()
```

Out[27]:
```
0
```

column=2 Transaction_ID

In [42]:
```
# this is the continuous variable and correct data types and there is not any null value
# but there is no use of this column in the analysis simply drop this column
df["Transaction_ID"].dtypes
```

Out[42]:
```
dtype('int64')
```

In [29]:
```
df["Transaction_ID"].isnull().sum()
```

Out[29]:
```
0
```

column=3 Television

In [31]:
```
# this is the count variable and correct data types and there is not any null value
df["Television"].dtypes
```

Out[31]:
```
dtype('int64')
```

In [33]:
```
df["Television"].unique()
```

Out[33]:
```
array([0, 1], dtype=int64)
```

In [35]:
```
df["Television"].value_counts()
```

Out[35]:
```
Television
1    24655
0    24317
Name: count, dtype: int64
```

In [37]:
```
df["Television"].isnull().sum()
```

Out[37]:
```
0
```

column 4 Dryer

In [54]:
```
# this is the count variable and correct data types and there is not any null value
df["Dryer"].dtypes
```

Out[54]:
```
dtype('int64')
```

In [56]:
```
df["Dryer"].unique()
```

Out[56]:
```
array([0, 1], dtype=int64)
```

In [39]:
```
df["Dryer"].value_counts()
```

Out[39]:
```
Dryer
1    24490
0    24482
Name: count, dtype: int64
```

In [60]:
```
df["Dryer"].isnull().sum()
```

Out[60]:
```
0
```

column=5 Oven

In [41]:
```
# this is the count variable and correct data types and there is not any null value
df["Oven"].dtypes
```

Out[41]:
```
dtype('int64')
```

In [43]:
```
df["Oven"].unique()
```

Out[43]:
```
array([0, 1], dtype=int64)
```

In [45]:
```
df["Oven"].value_counts()
```

Out[45]:
```
Oven
1    24583
0    24389
Name: count, dtype: int64
```

In [68]:
```
df["Oven"].isnull().sum()
```

Out[68]:
```
0
```

column=7 Refrigerator

In [70]:
```
# this is the count variable and correct data types and there is not any null value
df["Refrigerator"].dtypes
```

Out[70]:
```
dtype('int64')
```

In [47]:
```
df["Refrigerator"].unique()
```

Out[47]:
```
array([1, 0], dtype=int64)
```

In [72]:
```
df["Refrigerator"].value_counts()
```

Out[72]:
```
Refrigerator
1    24601
0    24371
Name: count, dtype: int64
```

In [49]:
```
df["Refrigerator"].isnull().sum()
```

Out[49]:
```
0
```

column=8 Microwave

In [78]:
```
# this is the count variable and correct data types and there is not any null value
df["Microwave"].dtypes
```

Out[78]:
```
dtype('int64')
```

In [80]:
```
df["Microwave"].unique()
```

Out[80]:
```
array([0, 1], dtype=int64)
```

In [51]:
```
df["Microwave"].value_counts()
```

Out[51]:
```
Microwave
1    24542
0    24430
Name: count, dtype: int64
```

In [84]:
```
df["Microwave"].isnull().sum()
```

Out[84]:
```
0
```

column=9 Line Voltage

In [53]:
```
# this is the continuous variable and correct data types and there is not any null value
df["Line Voltage"].dtypes
```

Out[53]:
```
dtype('int64')
```

In [90]:
```
df["Line Voltage"].isnull().sum()
```

Out[90]:
```
0
```

column=10 Voltage

In [92]:
```
# this is the continuous variable and correct data types and there is not any null value
df["Voltage"].dtypes
```

Out[92]:
```
dtype('int64')
```

In [96]:
```
df["Voltage"].isnull().sum()
```

Out[96]:
```
0
```

column=11 Energy Consumption (kWh)

In [98]:
```
# this is the continuous variable and correct data types and there is not any null value
df["Energy Consumption (kWh)"].dtypes
```

Out[98]:
```
dtype('float64')
```

In [55]:
```
df["Energy Consumption (kWh)"].isnull().sum()
```

Out[55]:
```
0
```

column= 12 Apparent Power

In [102…
```
# this is the continuous variable and correct data types and there is not any null value
df["Apparent Power"].dtypes
```

Out[102… dtype('int64')

In [106…
```
df["Apparent Power"].isnull().sum()
```

Out[106… 0

column= 13 Bandwidth

In [108…
```
# this is the continuous variable and correct data types and there is not any null value
df["Bandwidth"].dtypes
```

Out[108… dtype('int64')

In [110…
```
df["Bandwidth"].isnull().sum()
```

Out[110… 0

data preprocessing

In [ ]:
```
# Data cleaning
# demension reduction
# Data transformation
```

In [33]:
```
df.drop(columns=["Unix Timestamp","Transaction_ID"],inplace=True)
```

In [35]:
```
df
```

Out[35]:

	Television	Dryer	Oven	Refrigerator	Microwave	Line Voltage	Voltage	Apparent Power	Energy Consumption (kWh)	Offloading Decision	Bandwidth
0	0	0	0	1	0	237	233	1559	24.001763	0	30741
1	0	1	0	0	1	232	230	1970	31.225154	1	35070
2	0	1	0	0	0	223	222	1684	70.460700	1	44253
3	1	0	1	1	0	225	224	1694	32.264043	1	20911
4	1	0	0	1	0	222	214	1889	32.728111	0	1708
...	...	...	...	...	...	...	...	...	...	...	...
48967	1	0	1	0	0	234	230	1815	20.161006	1	22742
48968	0	0	1	0	1	238	235	1692	91.965343	0	22466
48969	1	0	0	1	1	235	229	1686	40.224097	0	10004
48970	1	1	1	1	1	237	230	1754	37.366360	0	48012
48971	0	0	0	0	1	220	215	1660	66.239065	0	4712

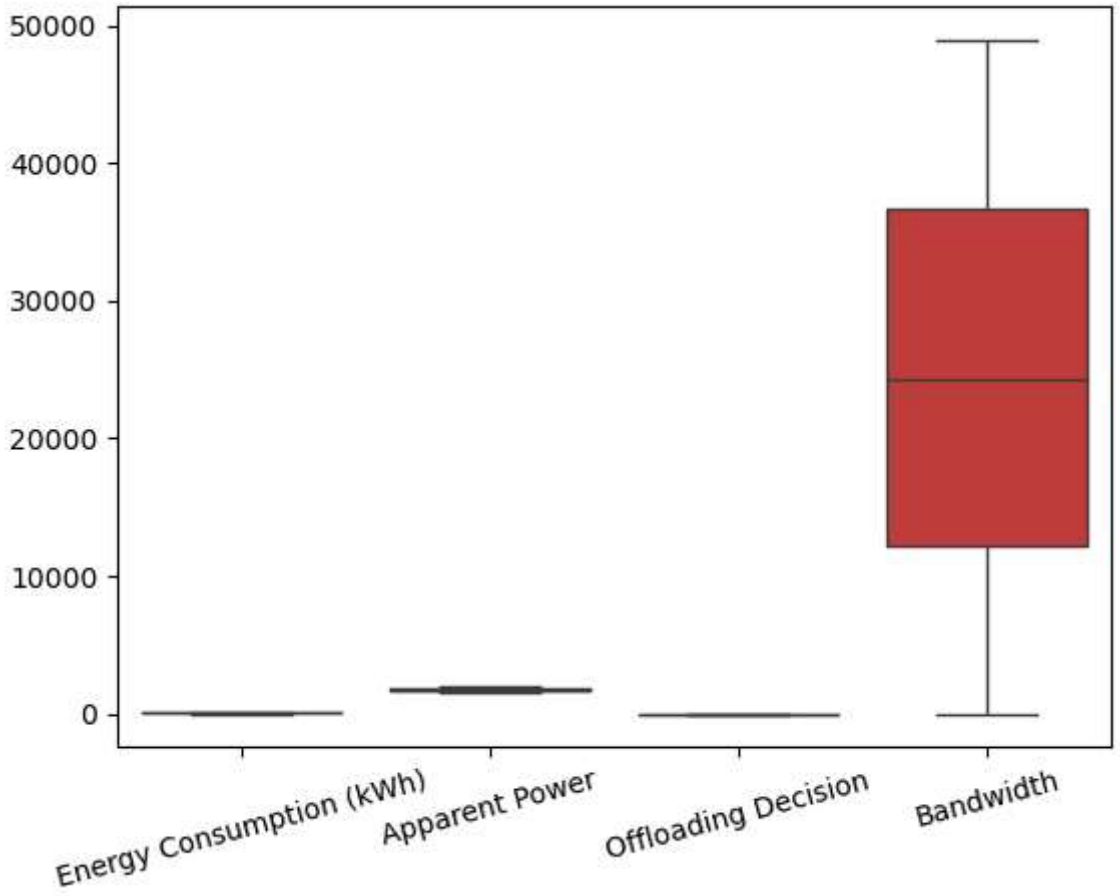
48972 rows × 11 columns

In [235…
```
conti=["Bandwidth","Voltage","Line Voltage","Energy Consumption","Offloading Decision",
      "Apparent Power"]
conti
discrete=["Television","Dryer","Oven","Refrigerator","Microwave"]
discrete
```

Out[235…
```
['Television', 'Dryer', 'Oven', 'Refrigerator', 'Microwave']
```

treat the outliers

In [39]:
```
sns.boxplot(df[["Energy Consumption (kWh)","Apparent Power","Offloading Decision","Bandwidth"]])
plt.xticks(rotation=15)
plt.show()
```



In [ ]:
```
# there is no any outliers present.
```

data analysis and visualization

In [13]:
```
df.head(2)
```

Out[13]:

	Unix Timestamp	Transaction_ID	Television	Dryer	Oven	Refrigerator	Microwave	Line Voltage	Voltage	Apparent Power	Energy Consumption (kWh)	Offloading Decision	Ban
0	1577836800	1	0	0	0	1	0	237	233	1559	24.001763	0	
1	1577839322	2	0	1	0	0	1	232	230	1970	31.225154	1	

In [15]:
```
df
```

Out[15]:

	Unix Timestamp	Transaction_ID	Television	Dryer	Oven	Refrigerator	Microwave	Line Voltage	Voltage	Apparent Power	Energy Consumption (kWh)	Offloading Decision
0	1577836800	1	0	0	0	1	0	237	233	1559	24.001763	0
1	1577839322	2	0	1	0	0	1	232	230	1970	31.225154	1
2	1577841845	3	0	1	0	0	0	223	222	1684	70.460700	1
3	1577844368	4	1	0	1	1	0	225	224	1694	32.264043	1
4	1577846891	5	1	0	0	1	0	222	214	1889	32.728111	0
...	...	...	...	...	...	...	...	...	...	...	...	...
48967	1701377135	48968	1	0	1	0	0	234	230	1815	20.161006	1
48968	1701379658	48969	0	0	1	0	1	238	235	1692	91.965343	0
48969	1701382181	48970	1	0	0	1	1	235	229	1686	40.224097	0
48970	1701384704	48971	1	1	1	1	1	237	230	1754	37.366360	0
48971	1701387227	48972	0	0	0	0	1	220	215	1660	66.239065	0

48972 rows × 13 columns

In [ ]:
```
# q 1 what is the average  energy consumption across all
```

In [242…
```
df["Energy Consumption (kWh)"].mean()
```

Out[242…
```
55.10126995297966
```

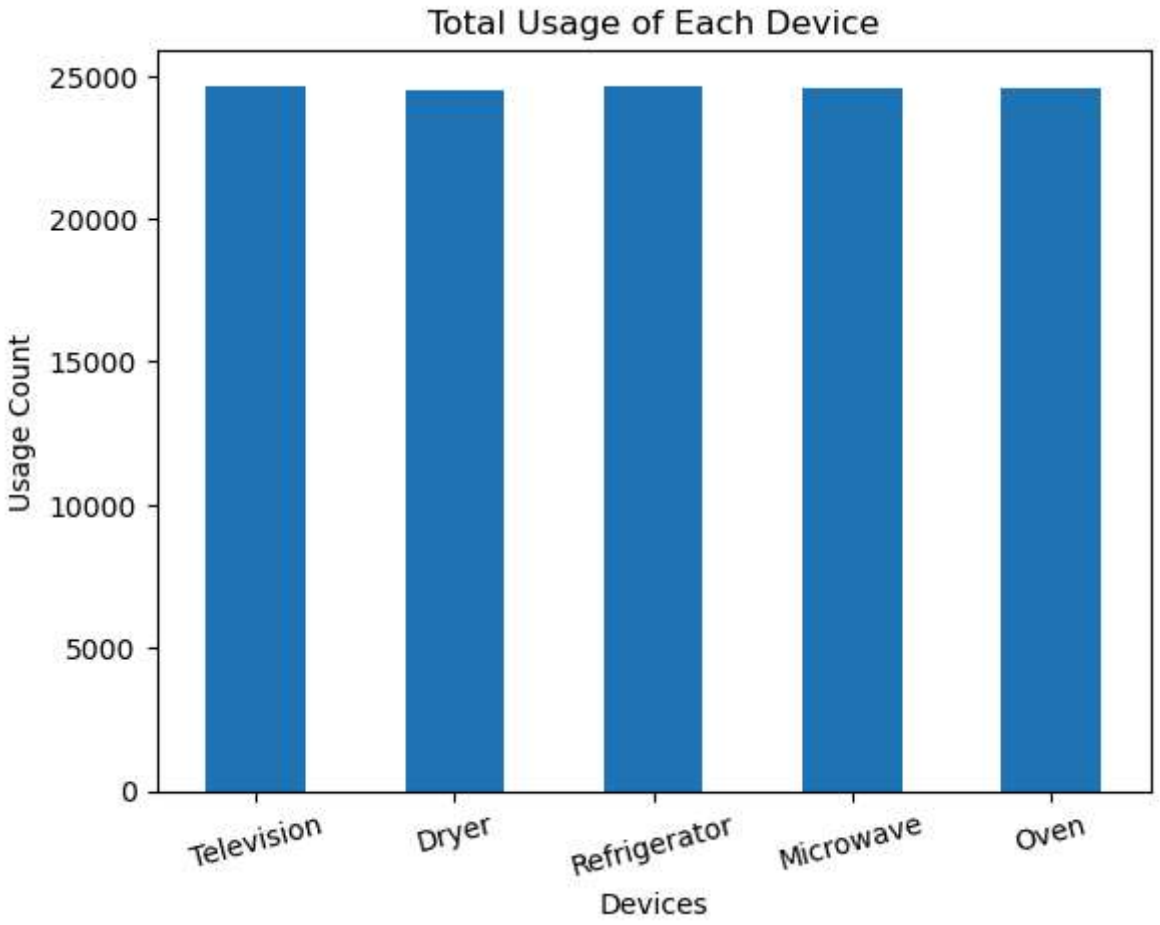
In [ ]:
```
# question 2 which device used most (by frequency)
# ans= television=24655
```

In [277…
```
total=df[["Television","Dryer","Refrigerator","Microwave","Oven"]].sum()
total
```

Out[277…
```
Television      24655
Dryer           24490
Refrigerator    24601
Microwave       24542
Oven            24583
dtype: int64
```

In [405…

```
total.plot(kind='bar')
plt.title('Total Usage of Each Device')
plt.ylabel('Usage Count')
plt.xlabel('Devices')
plt.xticks(rotation=15)
plt.savefig("Total Usage of Each Device")
plt.show()
```



In [ ]:

```
# q 3 what is the average  energy consumption each type of devices
```

In [315…

```
avg_energy = {}
for device in device_columns:
    avg = df[df[device] == 1]['Energy Consumption (kWh)'].mean()
    avg_energy[device] = avg
print(avg_energy)
```

```
{'Television': 55.13789495582316, 'Dryer': 55.06501663212863, 'Oven': 55.162551922169385, 'Refrigerator': 55.22102349917767, 'Micro
wave': 55.280956186982316}
```

In [119…
```
df[df['Television'] == 1]['Energy Consumption (kWh)'].mean()
```

Out[119…
```
55.13789495582316
```

In [127…
```
df[df["Dryer"]==1]['Energy Consumption (kWh)'].mean()
```

Out[127…
```
55.06501663212863
```

In [129…
```
df[df["Oven"]==1]['Energy Consumption (kWh)'].mean()
```

Out[129…
```
55.162551922169385
```

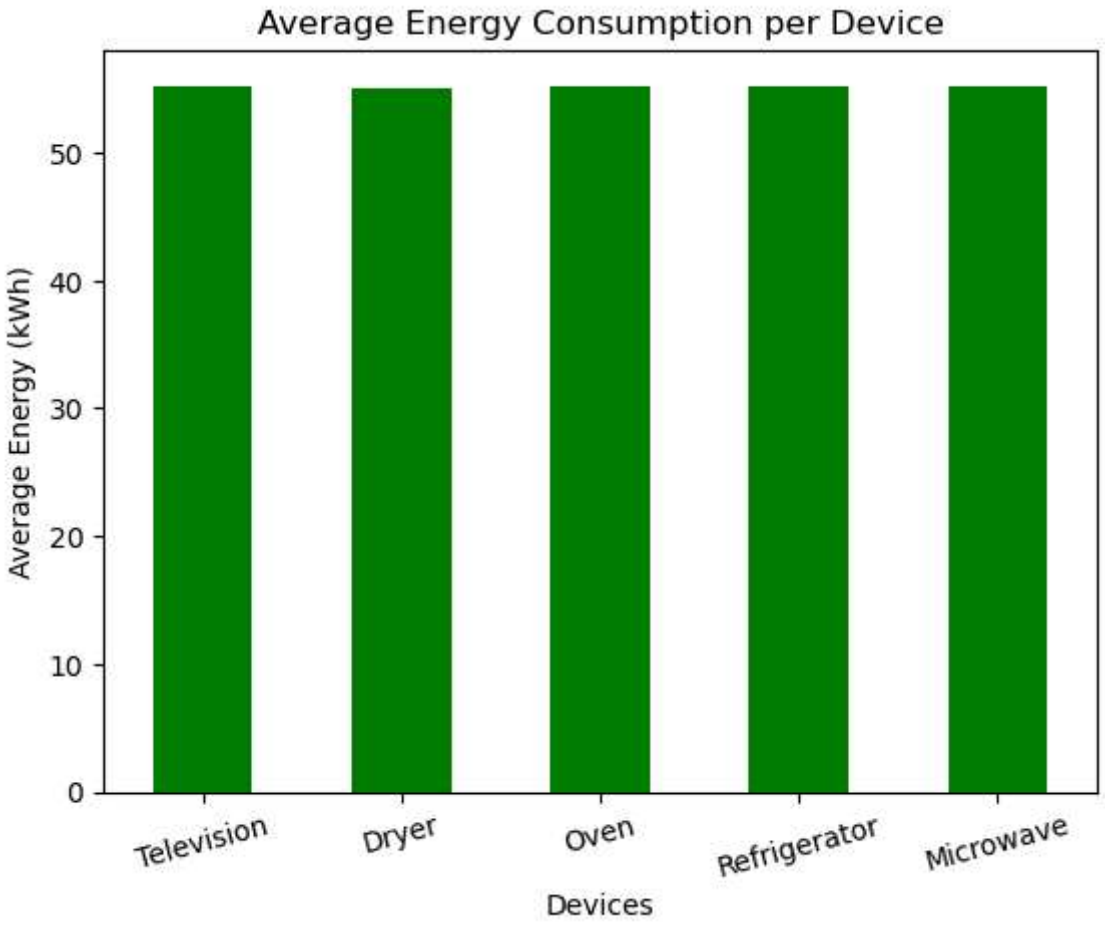
In [131…
```
df[df[""]==1]['Energy Consumption (kWh)'].mean()
```

Out[131…
```
55.22102349917767
```

In [133…
```
df[df["Microwave"]==1]['Energy Consumption (kWh)'].mean()
```

Out[133…
```
55.280956186982316
```

In [407…
```
pd.Series(avg_energy).plot(kind='bar', color='green')
plt.title('Average Energy Consumption per Device')
plt.ylabel('Average Energy (kWh)')
plt.xlabel('Devices')
plt.xticks(rotation=15)
plt.savefig("Average Energy Consumption per Device")
plt.show()
```



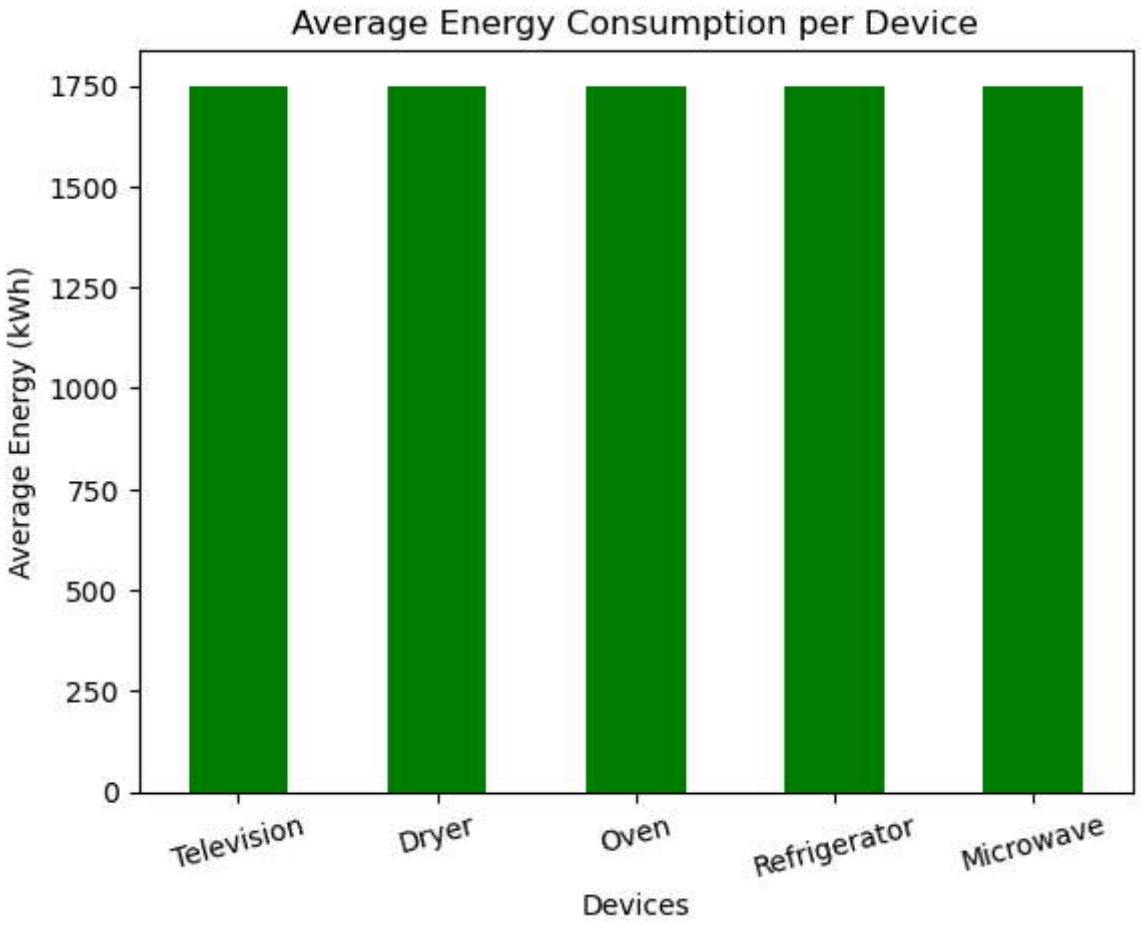
In [421…

```
avg_energy = {}
for device in device_columns:
    avg = df[df[device] == 1]["Apparent Power"].mean()
    avg_energy[device] = avg
print(avg_energy)
```

```
{'Television': 1749.2316365848712, 'Dryer': 1750.1506329113924, 'Oven': 1749.8859781149574, 'Refrigerator': 1748.983862444616, 'Microwave': 1750.0119794637765}
```

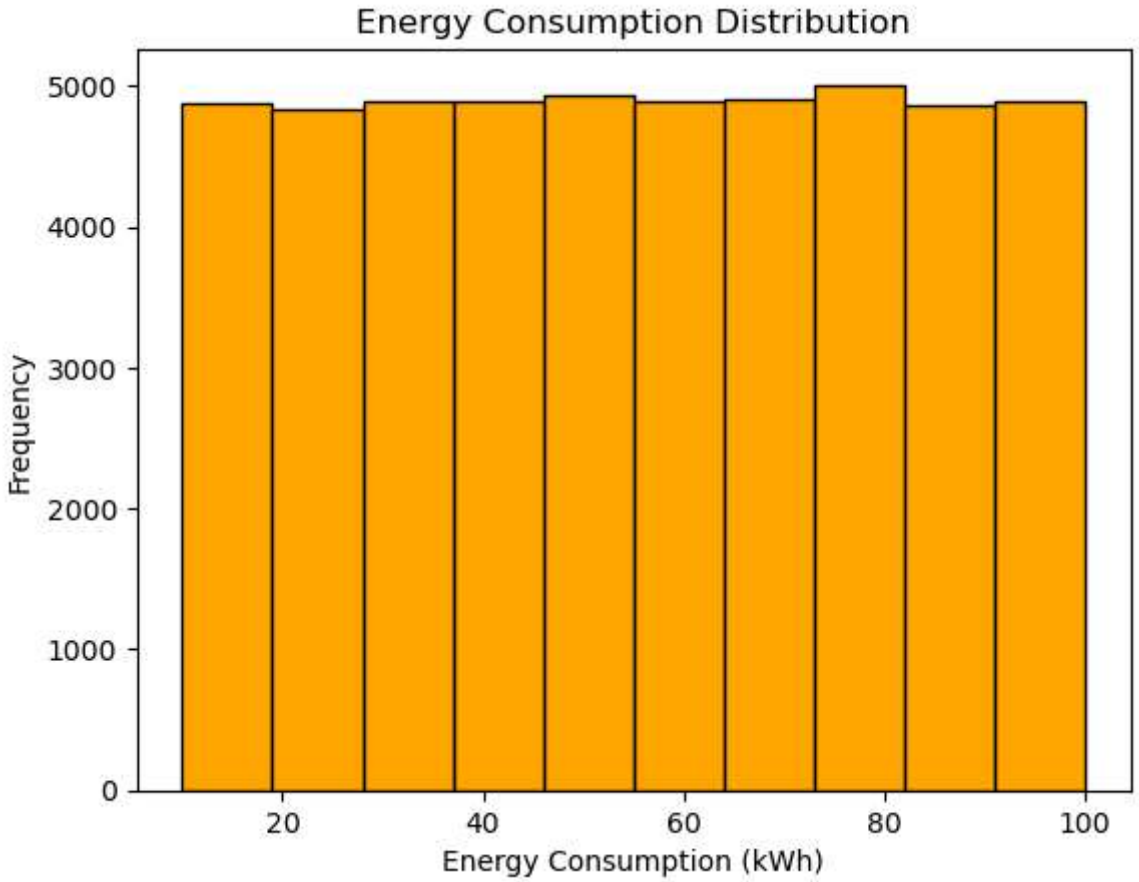
In [423…

```
pd.Series(avg_energy).plot(kind='bar', color='green')
plt.title('Average Energy Consumption per Device')
plt.ylabel('Average Energy (kWh)')
plt.xlabel('Devices')
plt.xticks(rotation=15)
plt.savefig("Average Energy Consumption per Device")
plt.show()
```



In [ ]:
```
# question 4 energy consumption distribution
```

In [413…
```
df['Energy Consumption (kWh)'].plot(kind='hist', color='orange', edgecolor='black')
plt.title('Energy Consumption Distribution')
plt.xlabel('Energy Consumption (kWh)')
plt.ylabel('Frequency')
plt.savefig("Energy Consumption Distribution1")
plt.show()
```



In [ ]:
```
# question 5 correlation b/w the devices
```

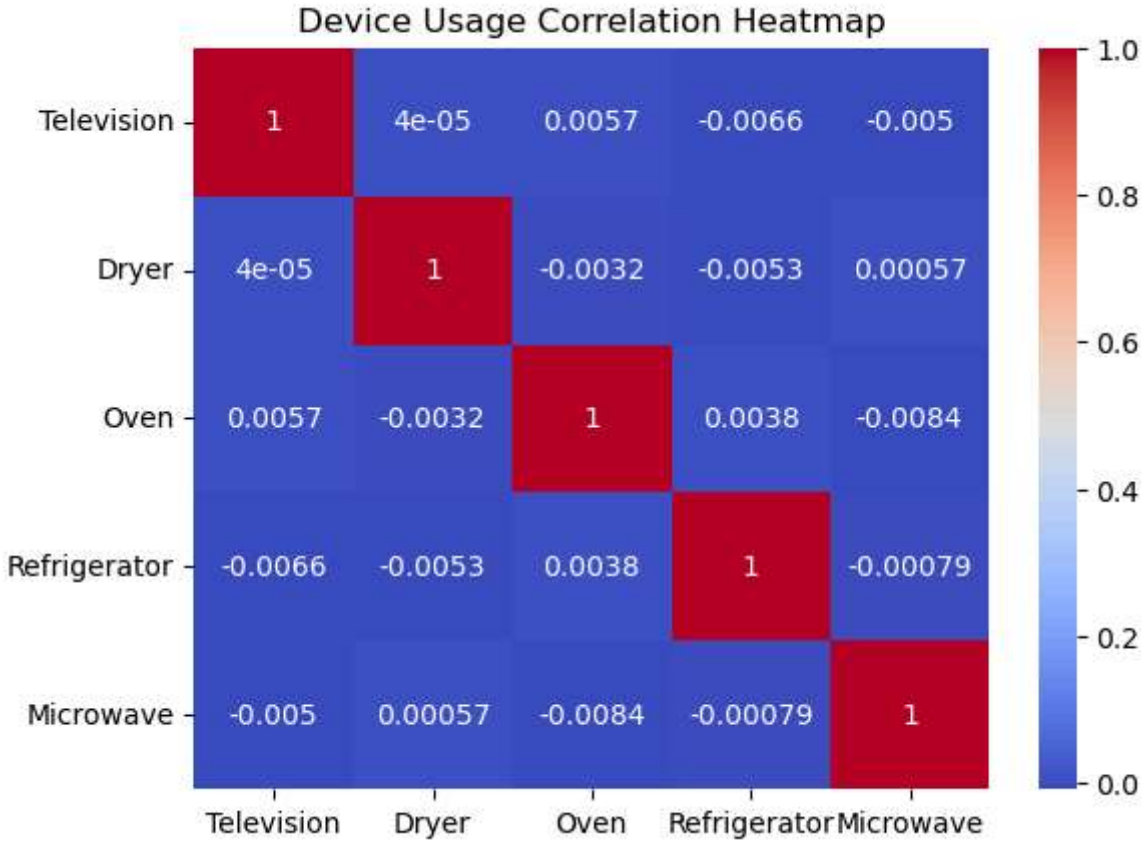
In [327…
```
device_corr = df[['Television', 'Dryer', 'Oven', 'Refrigerator', 'Microwave']].corr()
device_corr
```

Out[327…

	Television	Dryer	Oven	Refrigerator	Microwave
Television	1.000000	0.000040	0.005690	-0.006567	-0.005039
Dryer	0.000040	1.000000	-0.003227	-0.005269	0.000571
Oven	0.005690	-0.003227	1.000000	0.003820	-0.008381
Refrigerator	-0.006567	-0.005269	0.003820	1.000000	-0.000787
Microwave	-0.005039	0.000571	-0.008381	-0.000787	1.000000

In [409…
```
sns.heatmap(device_corr, annot=True, cmap='coolwarm')
plt.title('Device Usage Correlation Heatmap')
```

```
plt.savefig("Device Usage Correlation Heatmap")
plt.show()
```

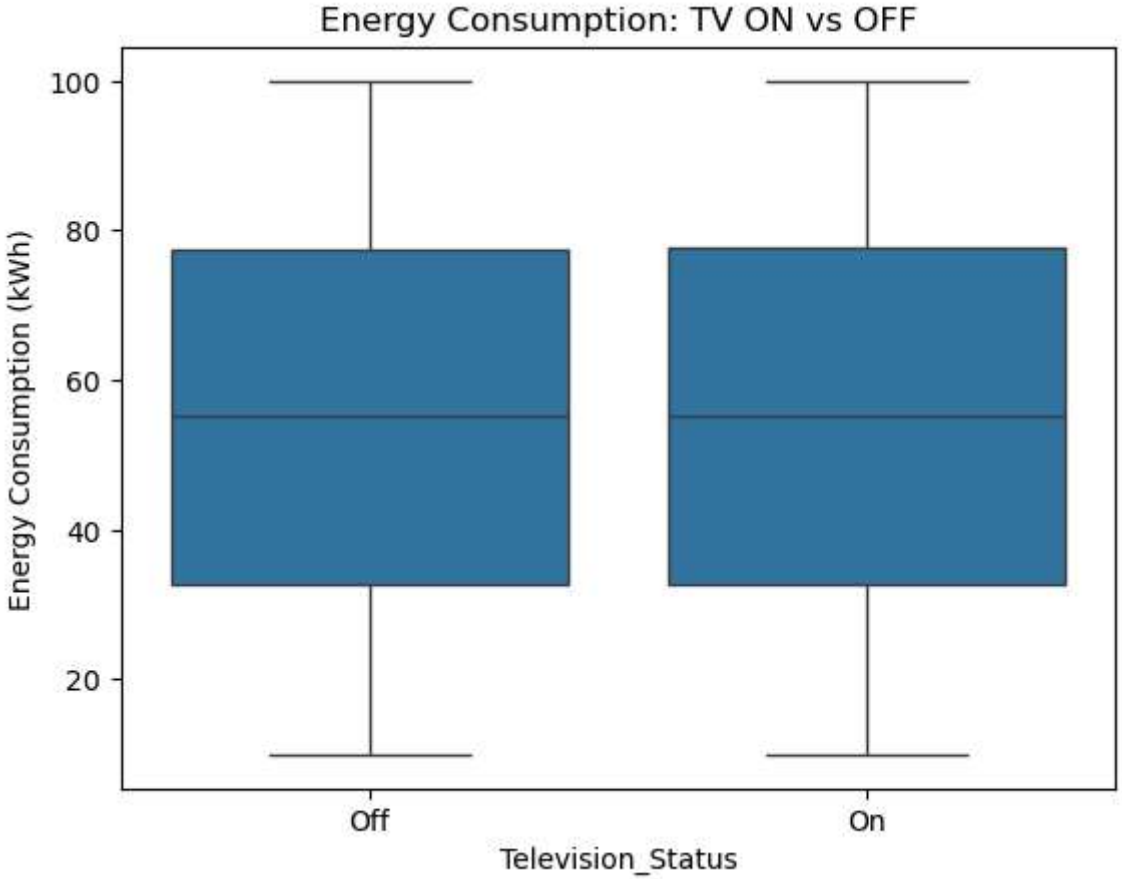


In [ ]:
```
# question 6 Compare average energy consumption when Television is ON vs OFF.
```

In [401…
```
df['Television_Status'] = df['Television'].replace({1: 'On', 0: 'Off'})
df.groupby("Television")["Energy Consumption (kWh)"].mean()
```

Out[401…
```
Television
0    55.064136
1    55.137895
Name: Energy Consumption (kWh), dtype: float64
```

In [417…
```
sns.boxplot(x='Television_Status', y='Energy Consumption (kWh)', data=df)
plt.title('Energy Consumption: TV ON vs OFF')
plt.savefig("Energy ConsumptiV ON vs OFF12")
plt.show()
```



In [349…
```
df[['Television']].value_counts(normalize=True)*100
```

Out[349…
```
Television
1             50.345095
0             49.654905
Name: proportion, dtype: float64
```