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

In [2]:
```
tele_df=pd.read_csv("telecom_churn_data.csv")
```

In [3]:
```
tele_df
```

Out[3]:

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	409-8743	Female	36	62	no
1	2015	100643	340-5930	Female	39	149	no
2	2015	100756	372-3750	Female	65	126	no
3	2015	101595	331-4902	Female	24	131	no
4	2015	101653	351-8398	Female	40	191	no
...	...	...	...	...	...	...	...
1995	2015	997132	385-7387	Female	54	75	no
1996	2015	998086	383-9255	Male	45	127	no
1997	2015	998474	353-2080	NaN	53	94	no
1998	2015	998934	359-7788	Male	40	94	no
1999	2015	999961	414-1496	Male	37	73	no

2000 rows × 16 columns

Data Frame Size

In [4]:
```
tele_df.size
```

Out[4]:
```
32000
```

In [ ]:
```

```

Data Frame Shape

In [5]:
```
print(f" the total columns is {tele_df.shape[1]}")
print(f" the total rows {tele_df.shape[0]}")
```

```
the total columns is 16
the total rows 2000
```

In [6]:
```
tele_df.shape
```

Out[6]:
```
(2000, 16)
```

In [ ]:
```

```

Data Types

In [7]:
```
tele_df.dtypes
```

Out[7]:
```
year                      int64
customer_id               int64
phone_no                 object
gender                   object
age                       int64
no_of_days_subscribed     int64
multi_screen             object
mail_subscribed          object
weekly_mins_watched     float64
minimum_daily_mins      float64
maximum_daily_mins      float64
weekly_max_night_mins     int64
videos_watched            int64
maximum_days_inactive   float64
customer_support_calls    int64
churn                   float64
dtype: object
```

In [ ]:
```

```

Filter categorical columns and numerical columns

In [8]:
```
cat_clm=tele_df.select_dtypes(include="object").columns
num_clm=tele_df.select_dtypes(exclude="object").columns
```

In [9]:
```
cat_clm
```

Out[9]:
```
Index(['phone_no', 'gender', 'multi_screen', 'mail_subscribed'], dtype='objec
t')
```

In [10]:
```
num_clm
```

Out[10]:
```
Index(['year', 'customer_id', 'age', 'no_of_days_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls', 'churn'],
      dtype='object')
```

In [ ]:
```

```

Check null values

In [11]:
```
tele_df.isnull()
```

Out[11]:

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
1995	False	False	False	False	False	False	False
1996	False	False	False	False	False	False	False
1997	False	False	False	True	False	False	False
1998	False	False	False	False	False	False	False
1999	False	False	False	False	False	False	False

2000 rows × 16 columns

In [12]:
```
tele_df.isnull().sum()
```

Out[12]:
```
year                      0
customer_id               0
phone_no                  0
gender                   24
age                       0
no_of_days_subscribed     0
multi_screen              0
mail_subscribed           0
weekly_mins_watched       0
minimum_daily_mins        0
maximum_daily_mins        0
weekly_max_night_mins     0
videos_watched            0
maximum_days_inactive    28
customer_support_calls    0
churn                    35
dtype: int64
```

In [ ]:
```

```

FILL THE NULL VALUES WITH MODE FOR CATEGORICAL COLUMNS

In [14]:
```
mode_data=tele_df["gender"].mode()
md=mode_data.values[0]
tele_df["gender"]=tele_df["gender"].fillna(md)

tele_df.isnull().sum()
```

Out[14]:
```
year                       0
customer_id                0
phone_no                   0
gender                     0
age                        0
no_of_days_subscribed      0
multi_screen               0
mail_subscribed            0
weekly_mins_watched        0
minimum_daily_mins         0
maximum_daily_mins         0
weekly_max_night_mins      0
videos_watched             0
maximum_days_inactive     28
customer_support_calls     0
churn                     35
dtype: int64
```

In [15]:
```
mode=tele_df["maximum_days_inactive"].mode()
md=mode.values[0]
tele_df["maximum_days_inactive"]=tele_df["maximum_days_inactive"].fillna(md)

tele_df.isnull().sum()
```

Out[15]:
```
year                       0
customer_id                0
phone_no                   0
gender                     0
age                        0
no_of_days_subscribed      0
multi_screen               0
mail_subscribed            0
weekly_mins_watched        0
minimum_daily_mins         0
maximum_daily_mins         0
weekly_max_night_mins      0
videos_watched             0
maximum_days_inactive      0
customer_support_calls     0
churn                     35
dtype: int64
```

In [16]:
```
mode=tele_df["churn"].mode()
md=mode.values[0]
tele_df["churn"]=tele_df["churn"].fillna(md)
```

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

Out[17]:
```
year                    0
customer_id             0
phone_no                0
gender                  0
age                     0
no_of_days_subscribed   0
multi_screen            0
mail_subscribed         0
weekly_mins_watched     0
minimum_daily_mins      0
maximum_daily_mins      0
weekly_max_night_mins   0
videos_watched          0
maximum_days_inactive   0
customer_support_calls  0
churn                   0
dtype: int64
```

In [18]:
```
tele_df
```

Out[18]:

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	409-8743	Female	36	62	no
1	2015	100643	340-5930	Female	39	149	no
2	2015	100756	372-3750	Female	65	126	no
3	2015	101595	331-4902	Female	24	131	no
4	2015	101653	351-8398	Female	40	191	no
...	...	...	...	...	...	...	...
1995	2015	997132	385-7387	Female	54	75	no
1996	2015	998086	383-9255	Male	45	127	no
1997	2015	998474	353-2080	Male	53	94	no
1998	2015	998934	359-7788	Male	40	94	no
1999	2015	999961	414-1496	Male	37	73	no

2000 rows × 16 columns

In [ ]:
```
```

Find the Numerical values and categorycal columns

In [20]:
```
cat_clm=tele_df.select_dtypes(include="object").columns
num_clm=tele_df.select_dtypes(exclude="object").columns
```

In [21]:
```
cat_clm
```

Out[21]:
```
Index(['phone_no', 'gender', 'multi_screen', 'mail_subscribed'], dtype='objec
t')
```

In [22]:
```
num_clm
```

Out[22]:
```
Index(['year', 'customer_id', 'age', 'no_of_days_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls', 'churn'],
      dtype='object')
```

In [23]:
```
data=tele_df["gender"].value_counts()
data
```

Out[23]:
```
gender
Male      1077
Female     923
Name: count, dtype: int64
```

In [ ]:
```

```

Find the keys and values

In [25]:
```
keys=tele_df["gender"].value_counts().keys()
keys
```

Out[25]:
```
Index(['Male', 'Female'], dtype='object', name='gender')
```

In [26]:
```
values=tele_df["gender"].value_counts().values
values
```

Out[26]:
```
array([1077,  923], dtype=int64)
```

In [27]:
```
gender_df=pd.DataFrame(zip(keys,values),columns=["Label","counts"])
gender_df
```

Out[27]:

	Label	counts
0	Male	1077
1	Female	923

In [ ]:
```

```

Frequency table of categorical columns

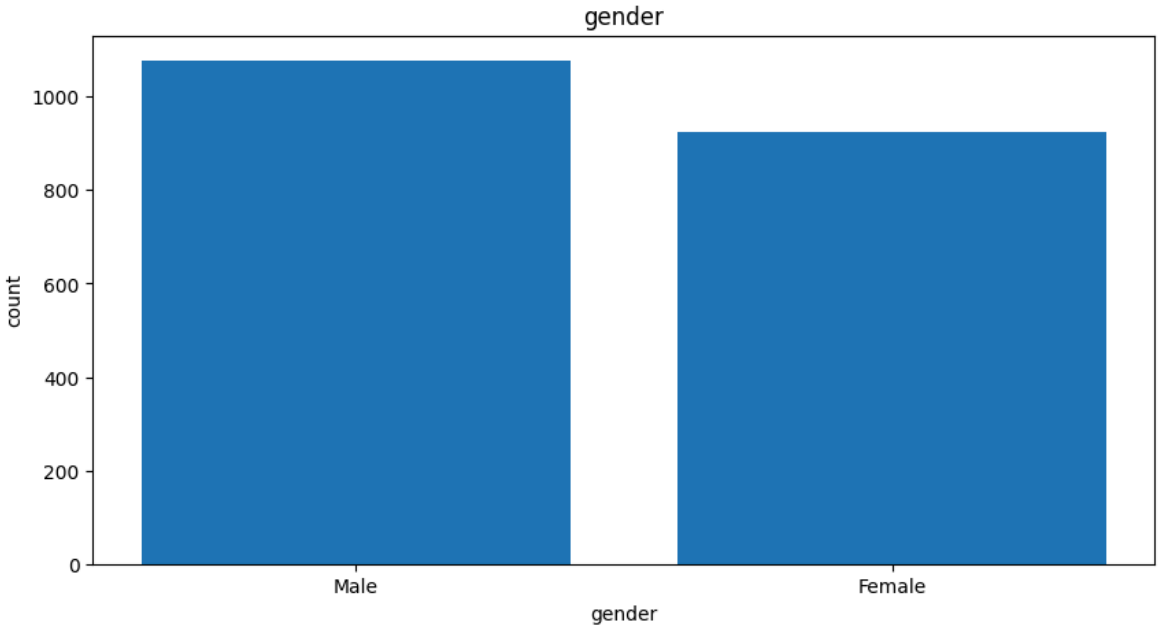
In [29]:
```
cat_clm=tele_df.select_dtypes(include="object").columns
for i in cat_clm[1:]:
    print("============>>>>>>>>>")
    keys=tele_df[i].value_counts().keys()
    values=tele_df[i].value_counts().values

    gender_df=pd.DataFrame(zip(keys,values),columns=["Label","counts"])
    print(gender_df)
```

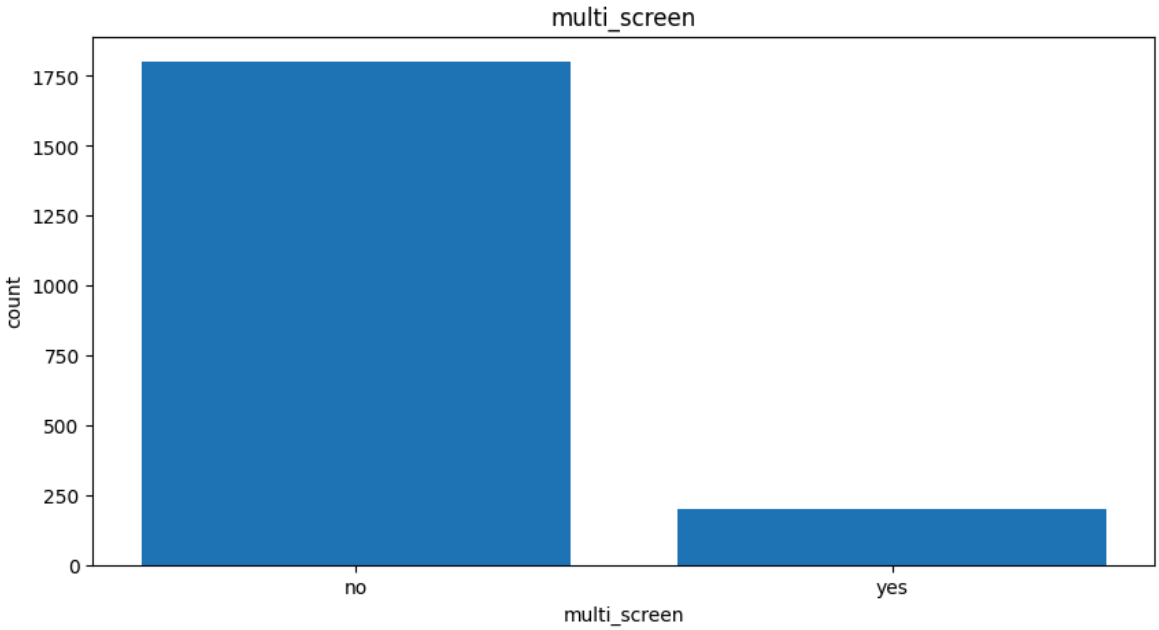
```
============>>>>>>>>>
    Label  counts
0    Male    1077
1  Female     923
============>>>>>>>>>
  Label  counts
0    no    1802
1   yes     198
============>>>>>>>>>
  Label  counts
0    no    1430
1   yes     570
```

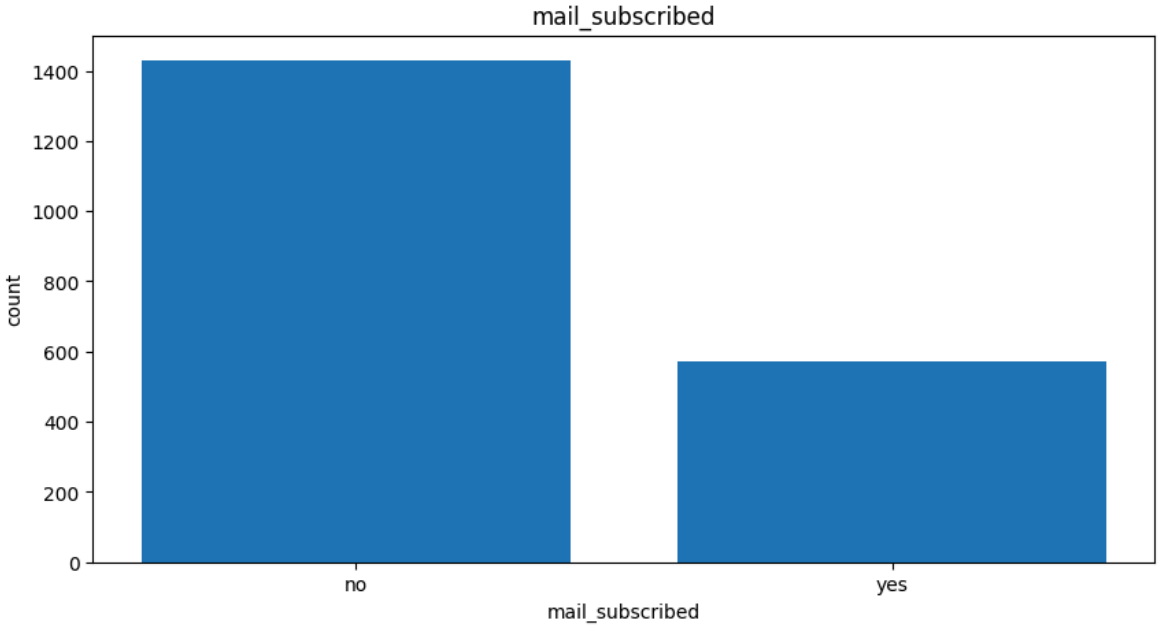
In [ ]:

BAR CHART

In [31]:
```
for i in cat_clm[1:]:
    counts=tele_df[i].value_counts()
    keys=counts.keys()
    values=counts.values

    plt.figure(figsize=(10,5))
    plt.bar(keys,values)
    plt.xlabel(i)
    plt.title(i)
    plt.ylabel("count")
```





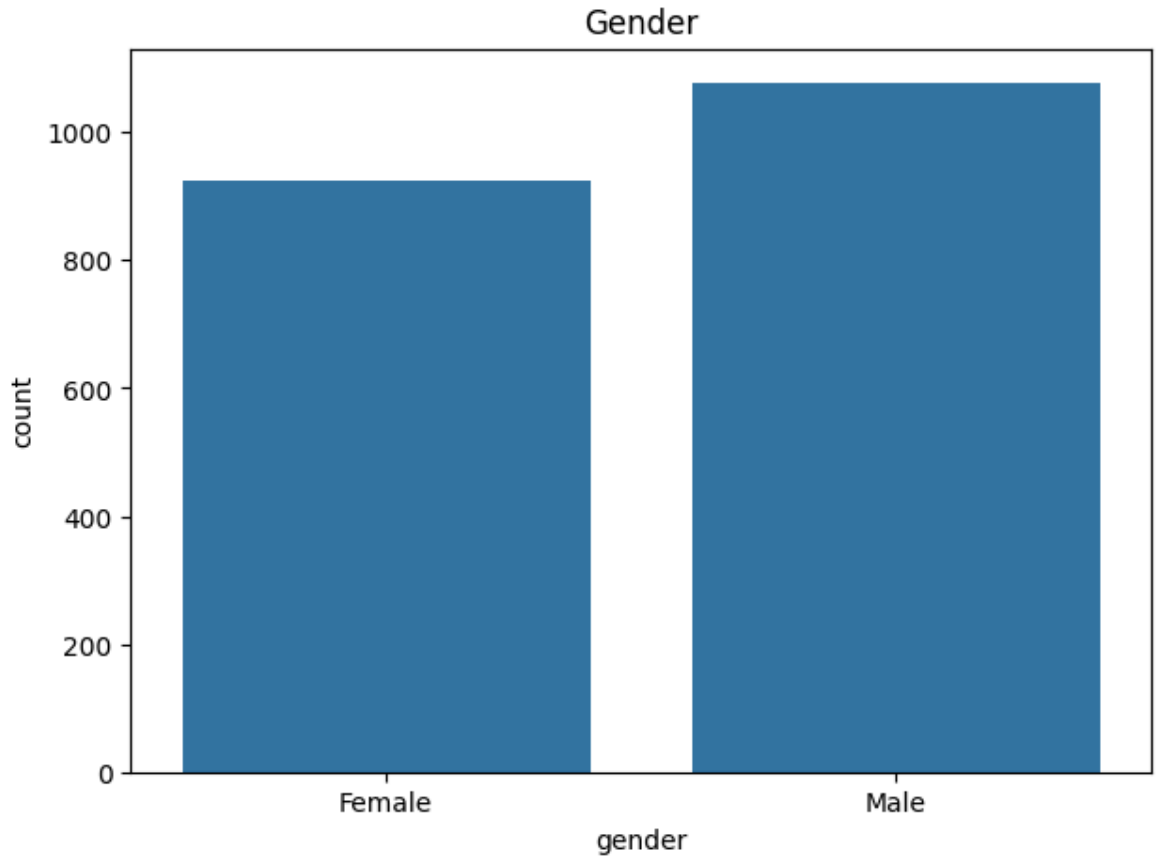


In [ ]:

BAR CHART USING COUNTPLOT METHOD

In [33]:
```
plt.figure(figsize=(7,5))
plt.title("Gender")
sns.countplot(data=tele_df,x="gender")
```

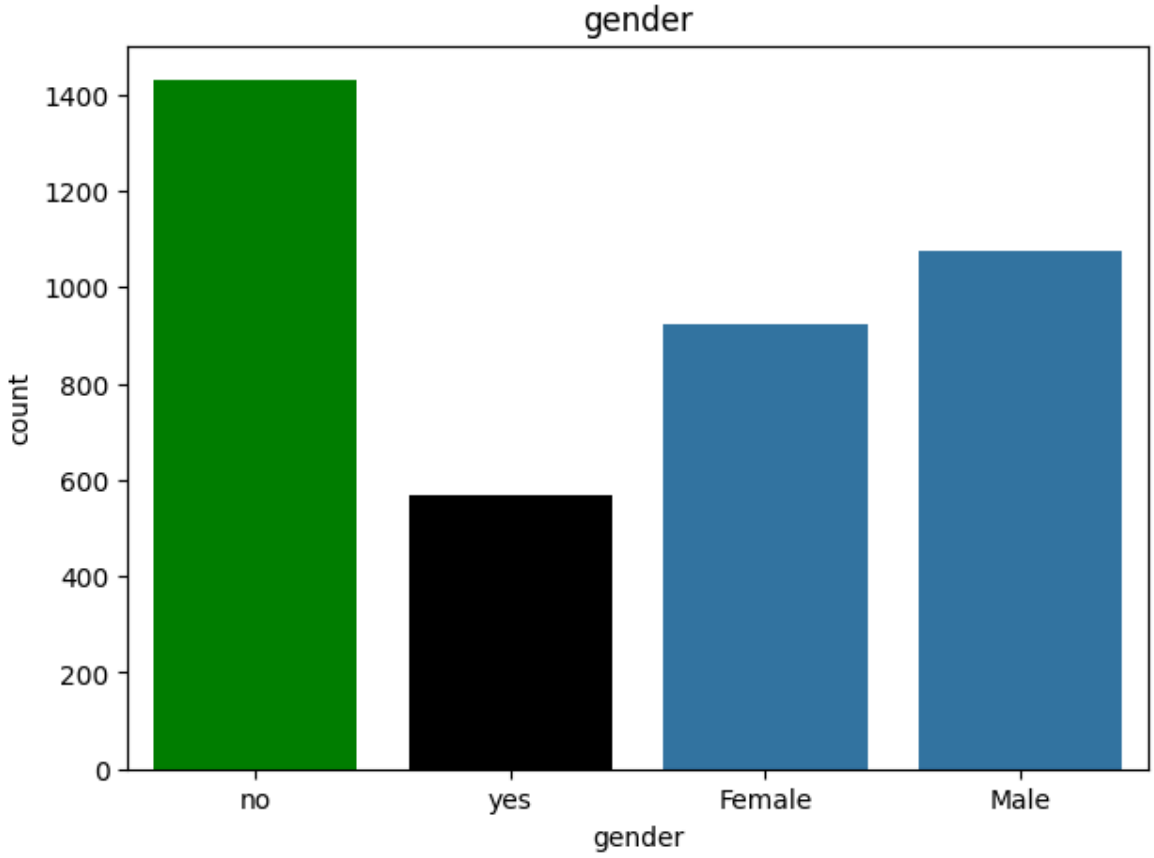
Out[33]: <Axes: title={'center': 'Gender'}, xlabel='gender', ylabel='count'>

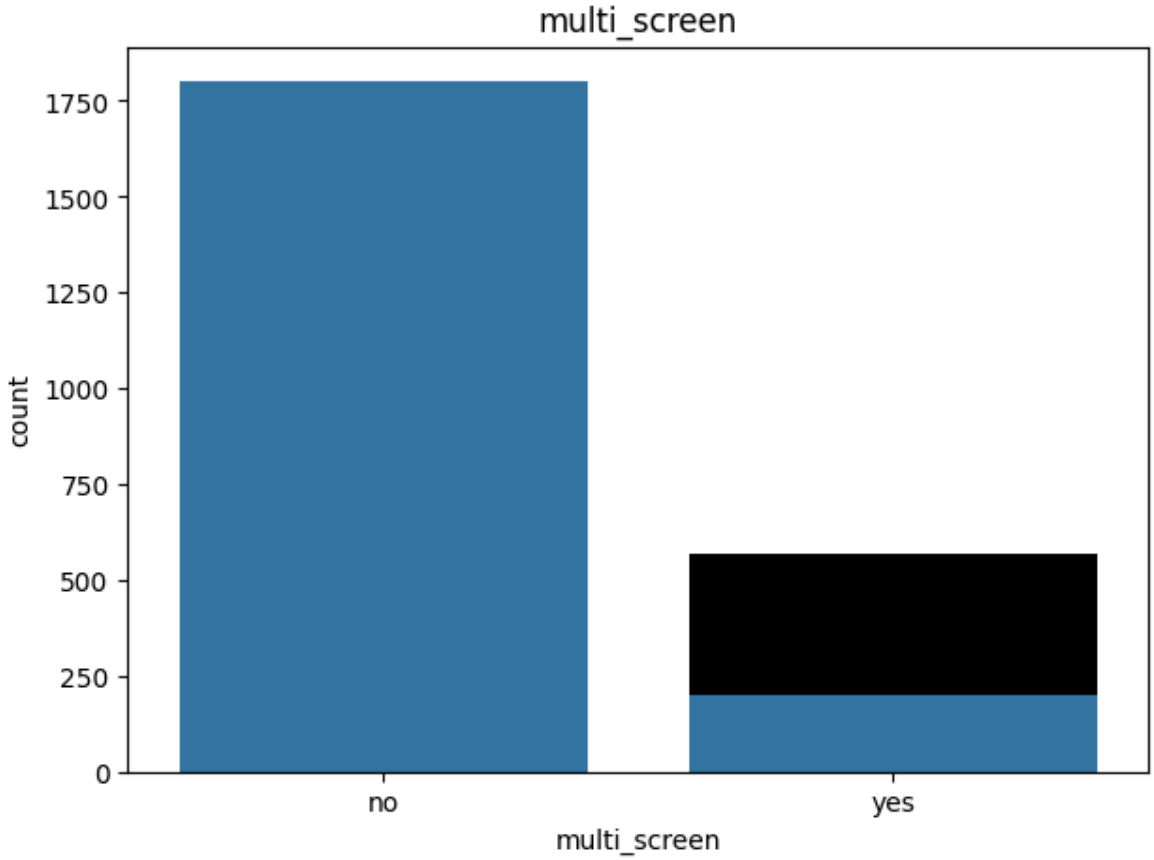


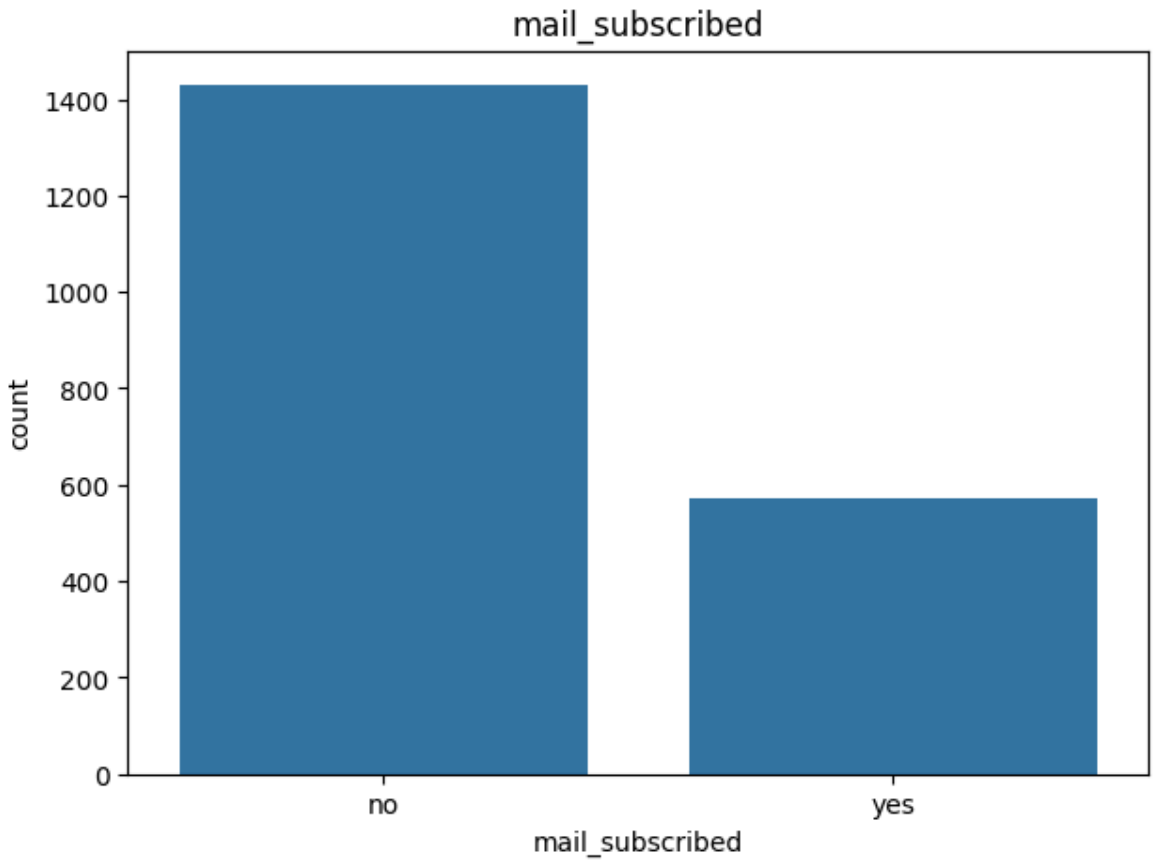
In [ ]:

BAR CHART USING COUNTPLOT METHOD IN FOR-LOOP FUNCTION

In [35]:
```
colors = ['green', 'black']
for i in cat_clm[1:]:
    plt.figure(figsize=(7,5))
    plt.bar(keys, values, color=colors[:len(keys)])
    plt.title(f"{i}")
    sns.countplot(data=tele_df,
                  x=i)
```


gender
1400
1200
1000
800
600
400
200
0
count
no
yes
Female
Male
gender

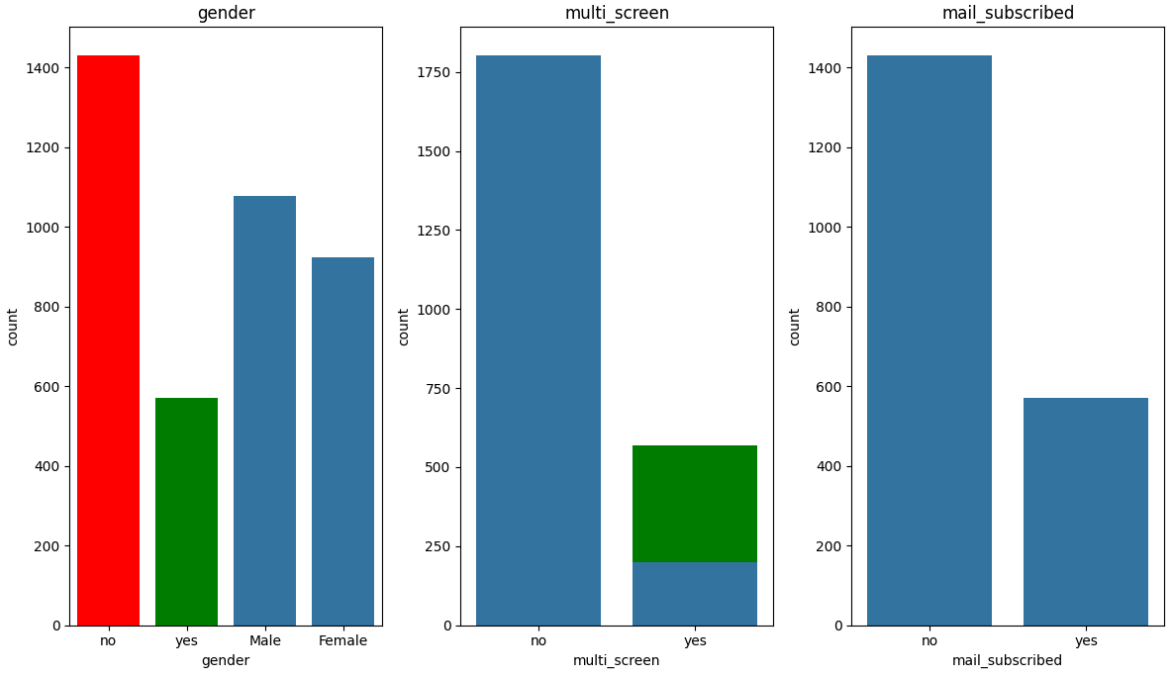

multi_screen
1750
1500
1250
1000
750
500
250
0
count
no
yes
multi_screen



In [ ]:

SUBPLOT

In [37]:
```
import matplotlib.pyplot as plt
import seaborn as sns
colors = ['red', 'green', 'blue']

plt.figure(figsize=(12, 7))
for i in range(1, len(cat_clm)):
    keys1 = tele_df[cat_clm[i]].value_counts().keys()
    plt.subplot(1, 3, i)
    plt.bar(keys, values, color=colors[:len(keys)])
    plt.title(f"{cat_clm[i]}")
    sns.countplot(data=tele_df, x=cat_clm[i], order=keys1)


plt.tight_layout()
plt.show()
```



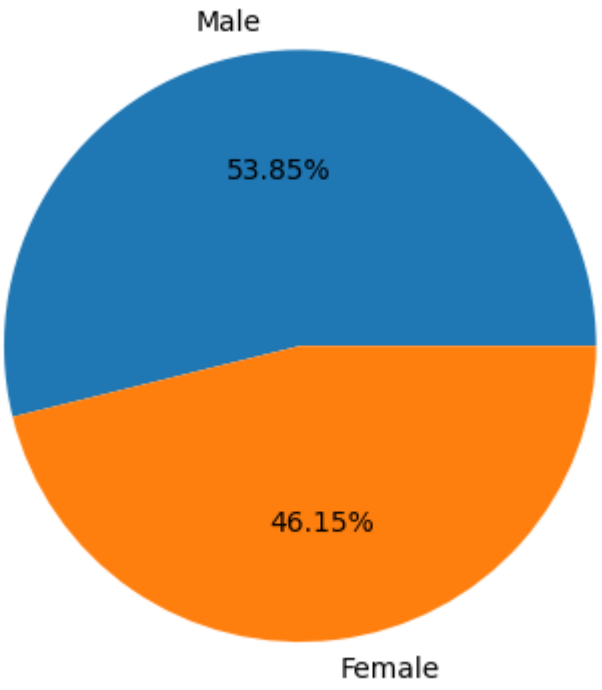
In [ ]:

PIE CHART

In [39]:
```
cat_clm
```

Out[39]:
```
Index(['phone_no', 'gender', 'multi_screen', 'mail_subscribed'], dtype='objec
t')
```

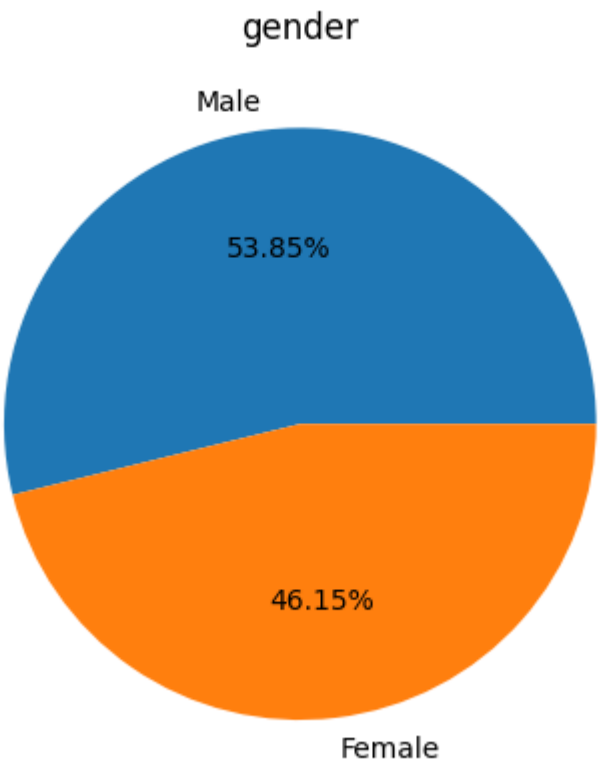
In [40]:
```
keys=tele_df["gender"].value_counts().keys()
values=tele_df["gender"].value_counts().values
plt.pie(values,labels=keys,autopct="%0.2f%%")
plt.show()
```

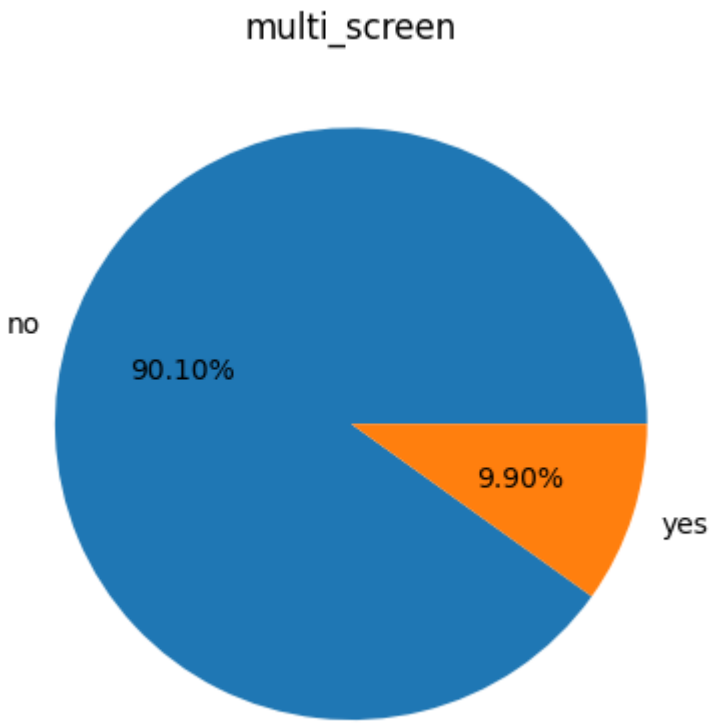


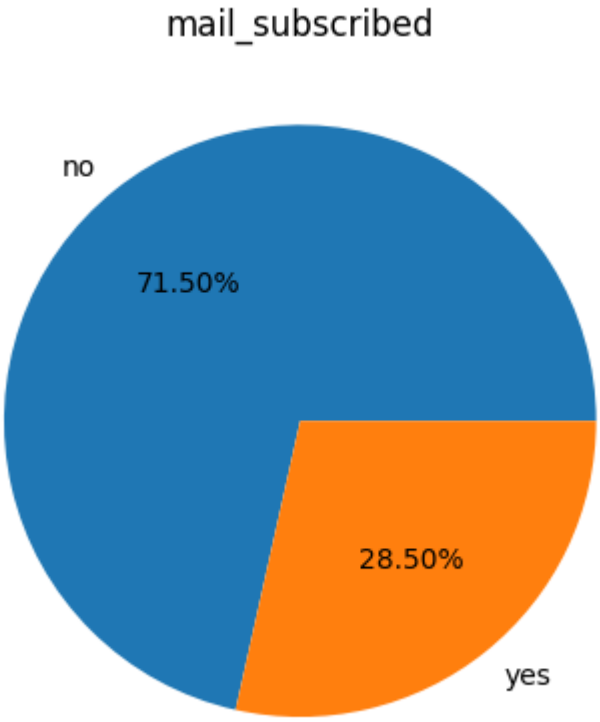
In [ ]:

USING FOR-LOOP

In [43]:
```
for i in cat_clm[1:]:
    keys=tele_df[i].value_counts().keys()
    values=tele_df[i].value_counts().values
    plt.title(i)
    plt.pie(values,labels=keys,autopct="%0.2f%%")
    plt.show()
```







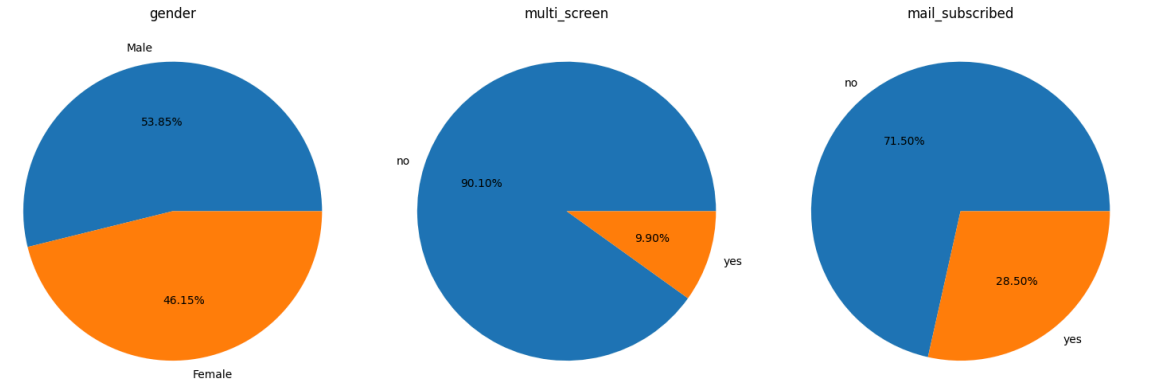
In [ ]:

In [ ]:

USING FOR-LOOP IN SUBPLOT

In [45]:
```
plt.figure(figsize=(15, 5))
for idx, i in enumerate(cat_clm[1:], start=1):
    plt.subplot(1, 3, idx)
```

```
    keys=tele_df[i].value_counts().keys()
    values=tele_df[i].value_counts().values
    plt.title(i)
    plt.pie(values, labels=keys, autopct="%0.2f%%")

plt.tight_layout()
plt.show()
```



In [ ]:

NUMERICAL COLUMNS ANALYSIS

In [47]:
```
tele_df
```

Out[47]:

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	409-8743	Female	36	62	no
1	2015	100643	340-5930	Female	39	149	no
2	2015	100756	372-3750	Female	65	126	no
3	2015	101595	331-4902	Female	24	131	no
4	2015	101653	351-8398	Female	40	191	no
...	...	...	...	...	...	...	...
1995	2015	997132	385-7387	Female	54	75	no
1996	2015	998086	383-9255	Male	45	127	no
1997	2015	998474	353-2080	Male	53	94	no
1998	2015	998934	359-7788	Male	40	94	no
1999	2015	999961	414-1496	Male	37	73	no

2000 rows × 16 columns

In [48]:
```
num_clm
```

Out[48]:
```
Index(['year', 'customer_id', 'age', 'no_of_days_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls', 'churn'],
      dtype='object')
```

In [49]:
```
# First we have to drop the churn columns

num_clm_2=num_clm.drop("churn")
num_clm_2
```

Out[49]:
```
Index(['year', 'customer_id', 'age', 'no_of_days_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls'],
      dtype='object')
```

In [50]:
```
# frequency table for numerial column
# We will use sescribe method here
```

In [51]:
```
tele_df.describe()
```

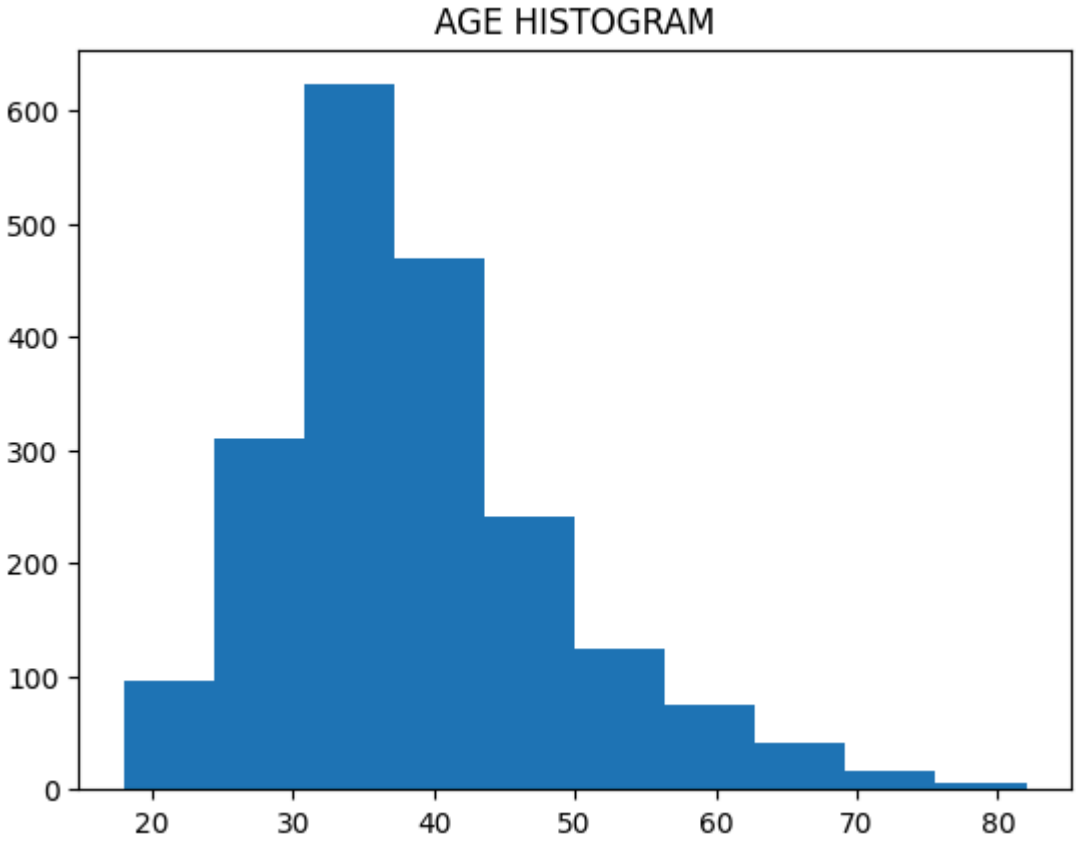
Out[51]:

	year	customer_id	age	no_of_days_subscribed	weekly_mins_watched
count	2000.0	2000.000000	2000.00000	2000.000000	2000.000000
mean	2015.0	554887.157500	38.69050	99.750000	270.178425
std	0.0	261033.690318	10.20641	39.755386	80.551627
min	2015.0	100198.000000	18.00000	1.000000	0.000000
25%	2015.0	328634.750000	32.00000	73.000000	218.212500
50%	2015.0	567957.500000	37.00000	99.000000	269.925000
75%	2015.0	773280.250000	44.00000	127.000000	324.675000
max	2015.0	999961.000000	82.00000	243.000000	526.200000

In [ ]:
```

```

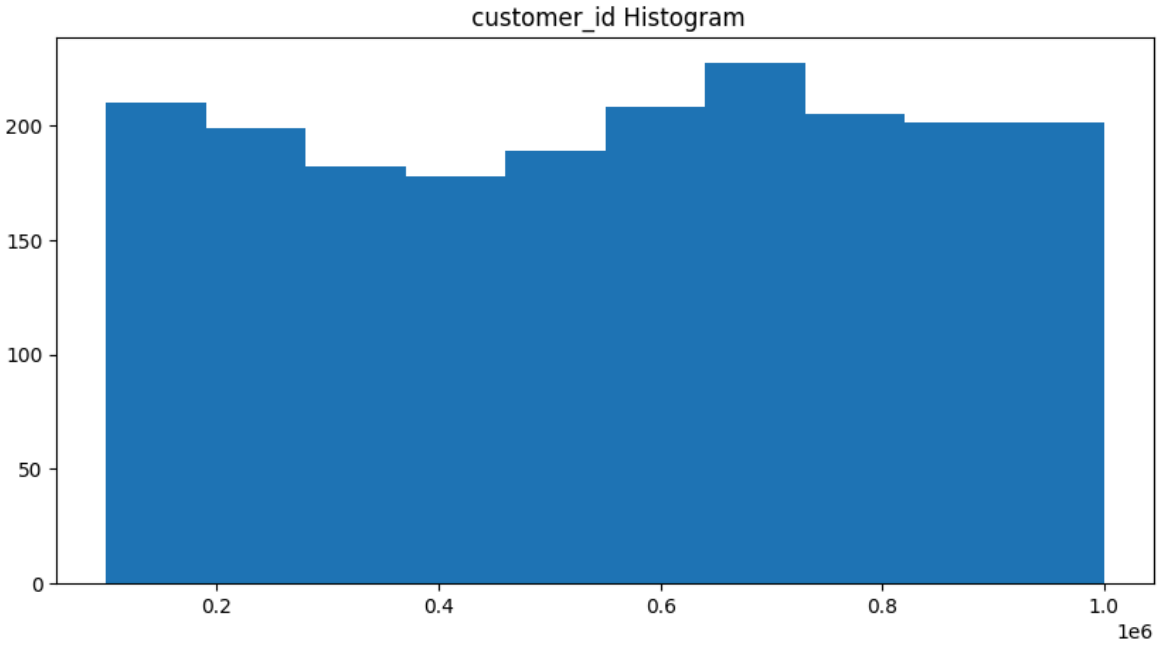
HOSTOGRAM

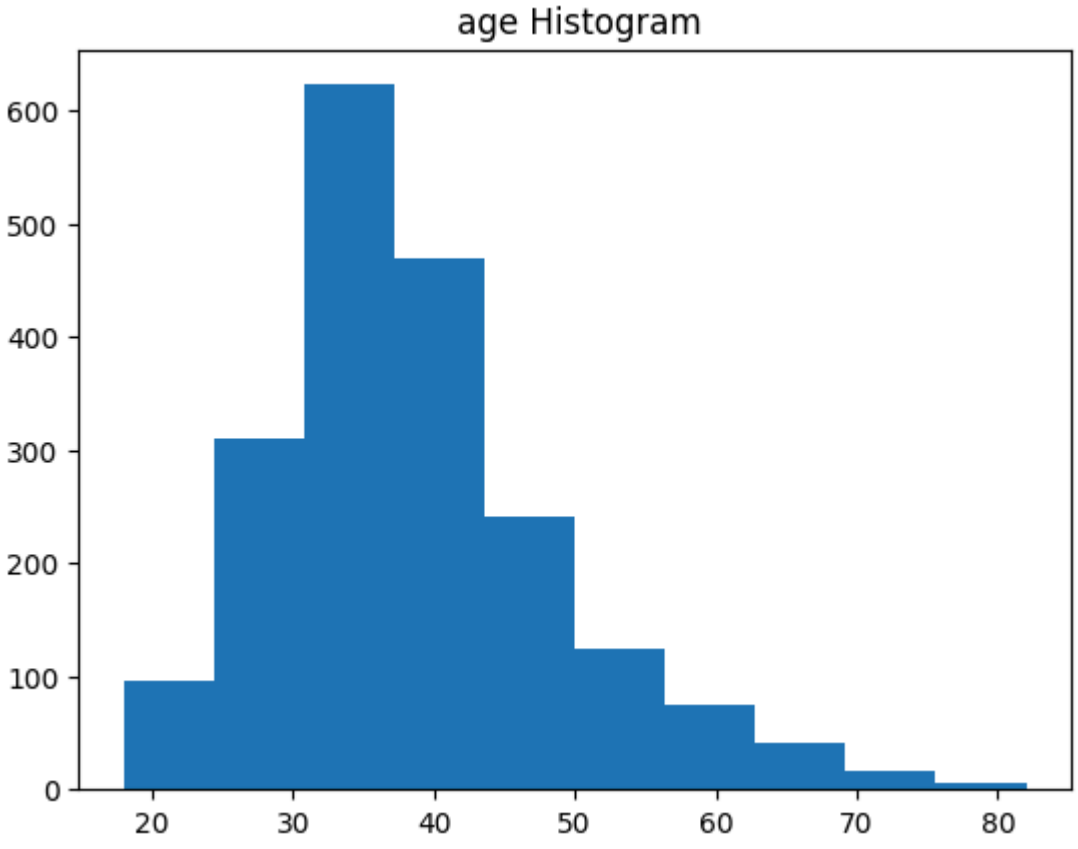
In [53]:
```
data=tele_df["age"]
plt.hist(data)
plt.title("AGE HISTOGRAM")
plt.show()
```

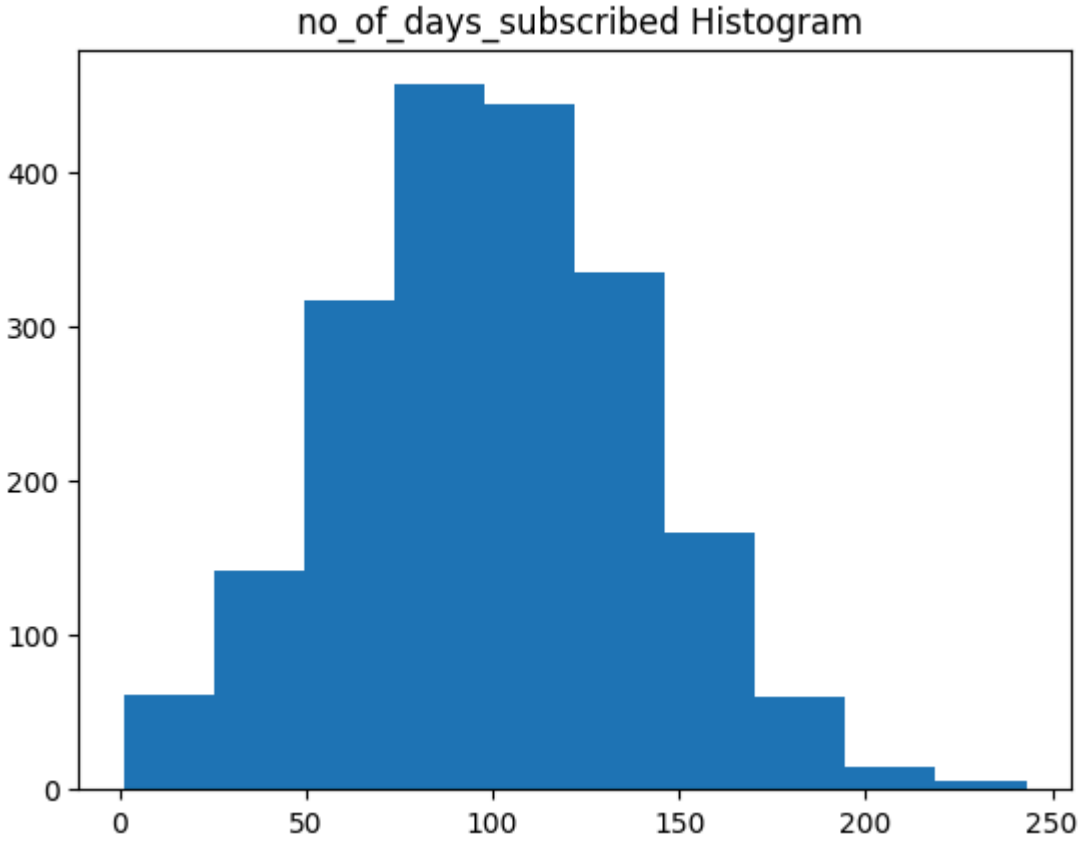


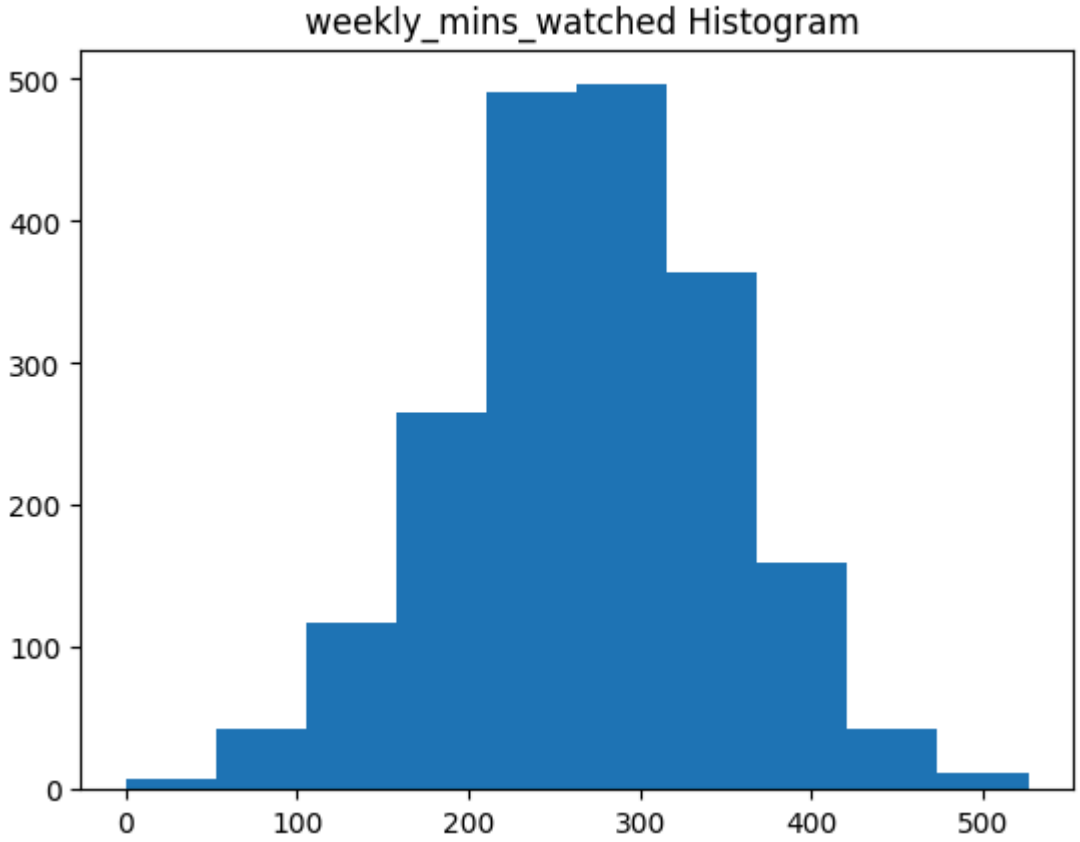
USING FOR-LOOP

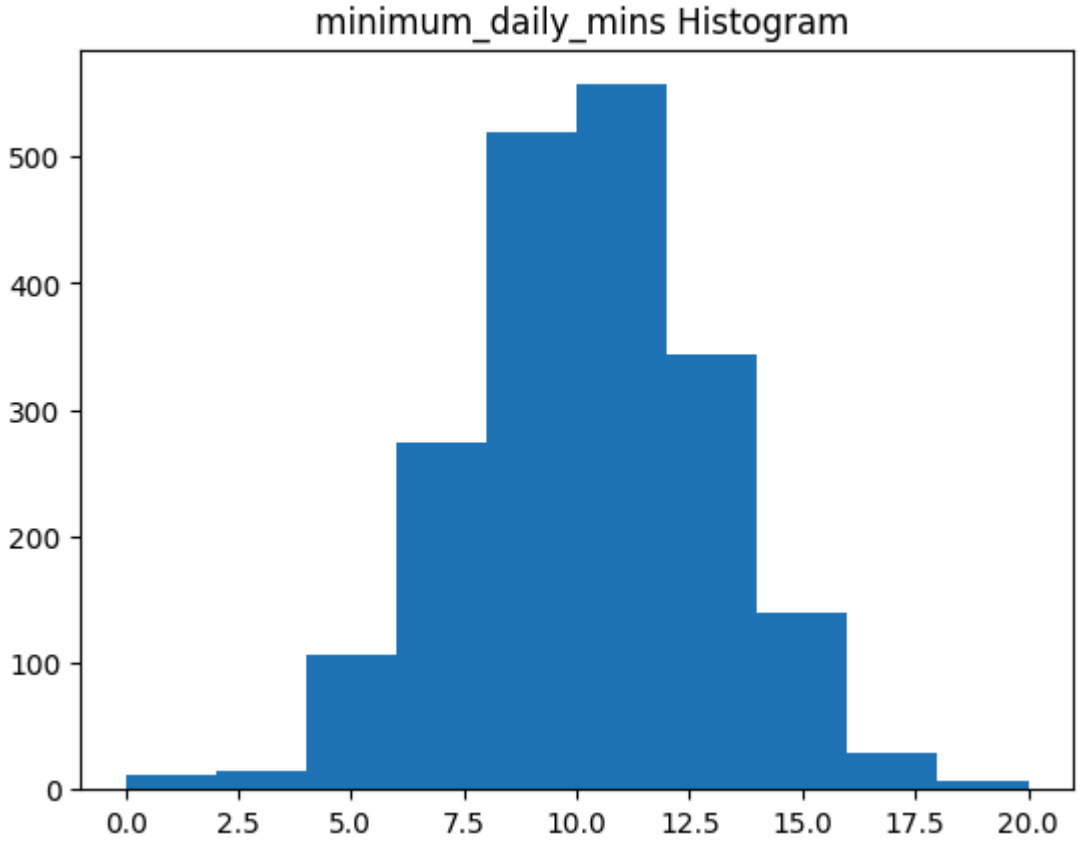
In [54]:
```
plt.figure(figsize=(10,5))
for i in num_clm_2[1:]:
    plt.hist(tele_df[i])
    plt.title(f" {i} Histogram")
    plt.show()
```

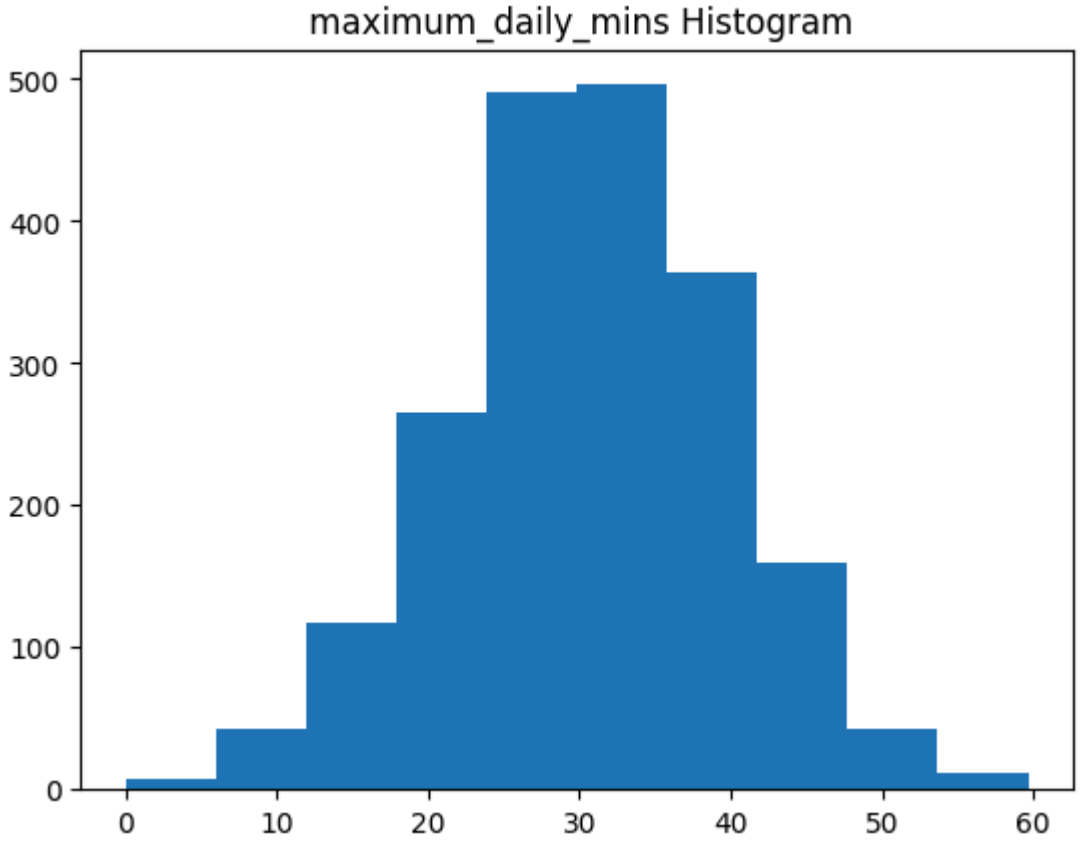


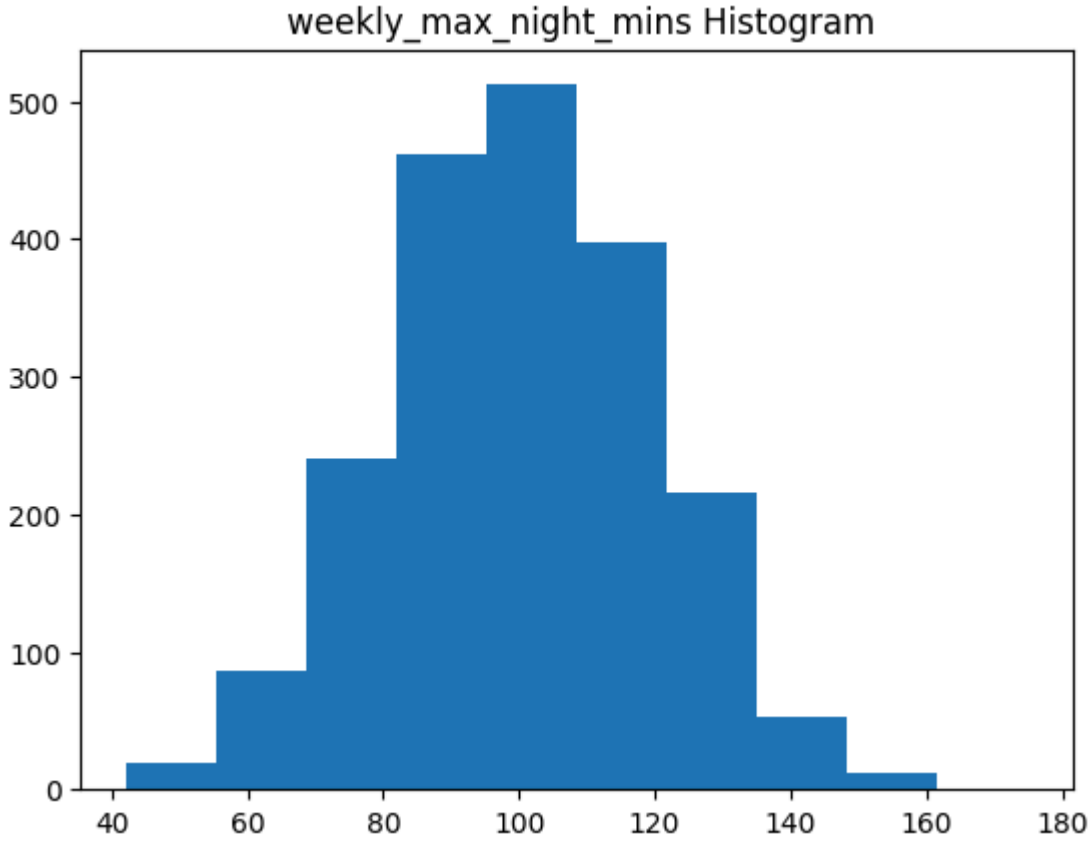

age Histogram
600
500
400
300
200
100
0
20
30
40
50
60
70
80

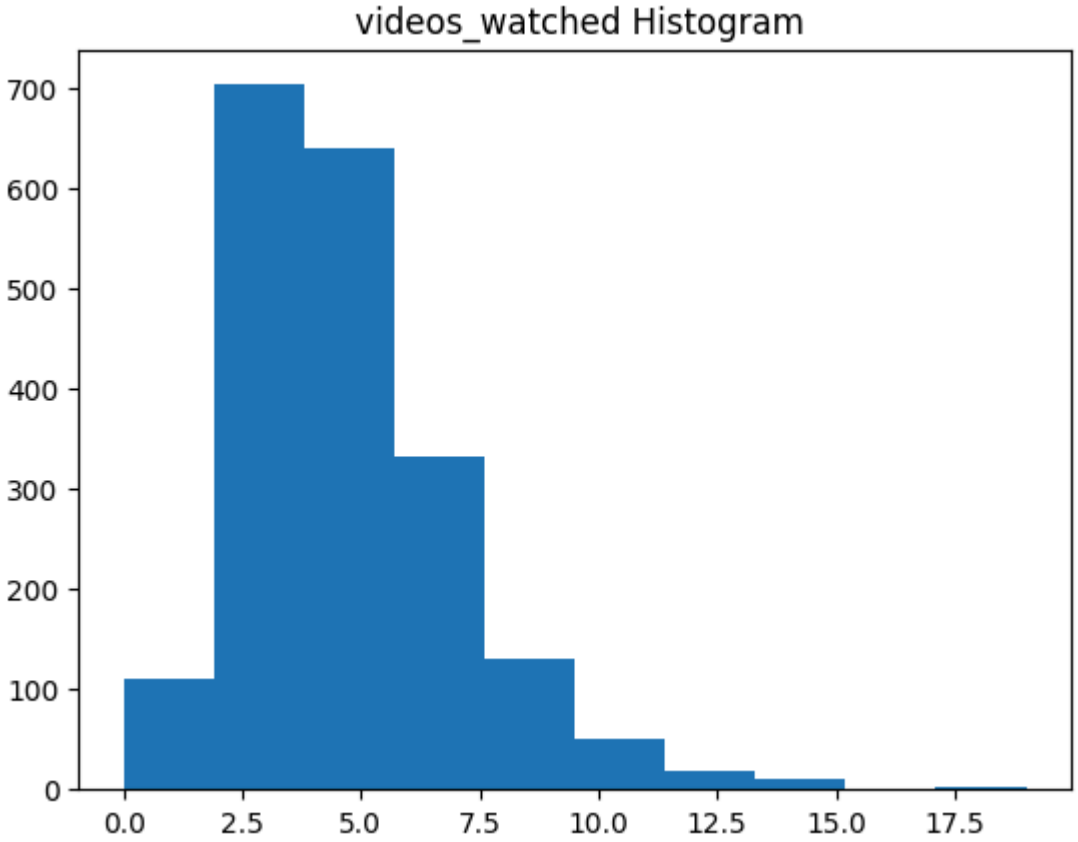

no_of_days_subscribed Histogram
400
300
200
100
0
0
50
100
150
200
250

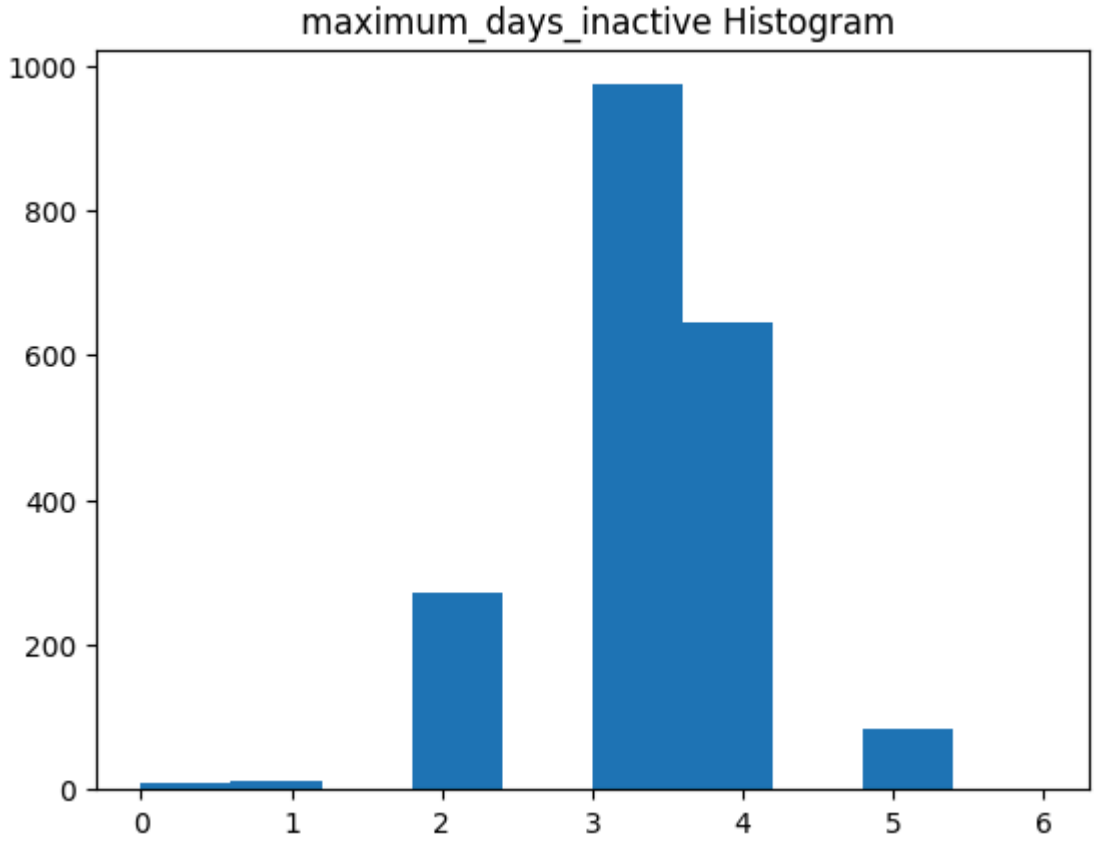

weekly_mins_watched Histogram
500
400
300
200
100
0
0
100
200
300
400
500

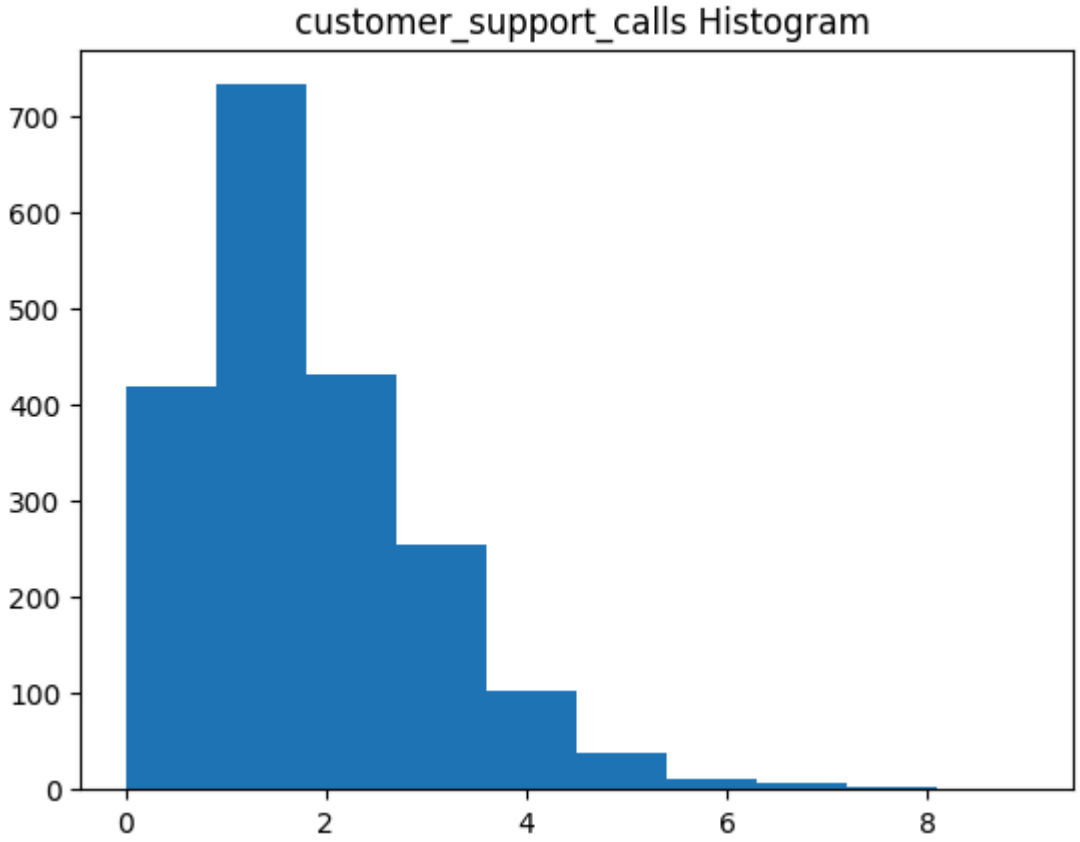

minimum_daily_mins Histogram
500
400
300
200
100
0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0


maximum_daily_mins Histogram
500
400
300
200
100
0
0
10
20
30
40
50
60


weekly_max_night_mins Histogram
500
400
300
200
100
0
40
60
80
100
120
140
160
180


videos_watched Histogram
700
600
500
400
300
200
100
0
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

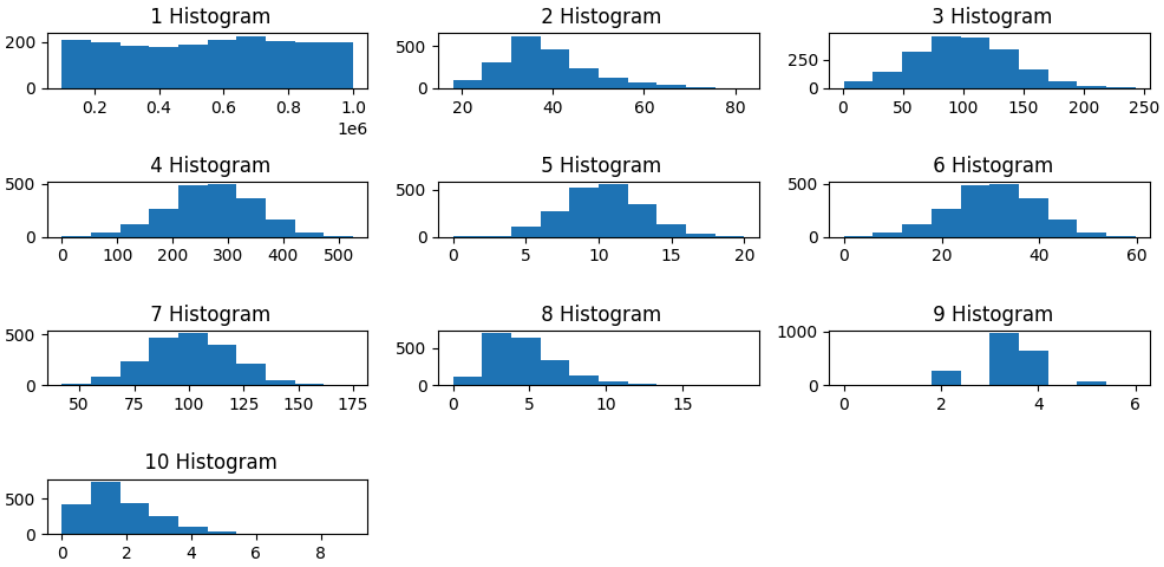

maximum_days_inactive Histogram
1000
800
600
400
200
0
0
1
2
3
4
5
6



In [ ]:

SUBPLOT

In [56]:
```
plt.figure(figsize=(10,5))
for i in range(1,len(num_clm_2)):

    plt.subplot(4,3,i)
    plt.hist(tele_df[num_clm_2[i]])
    plt.title(f" {i} Histogram")

plt.tight_layout()
plt.show()
```



In [ ]:

DATA DISTRIBUTION

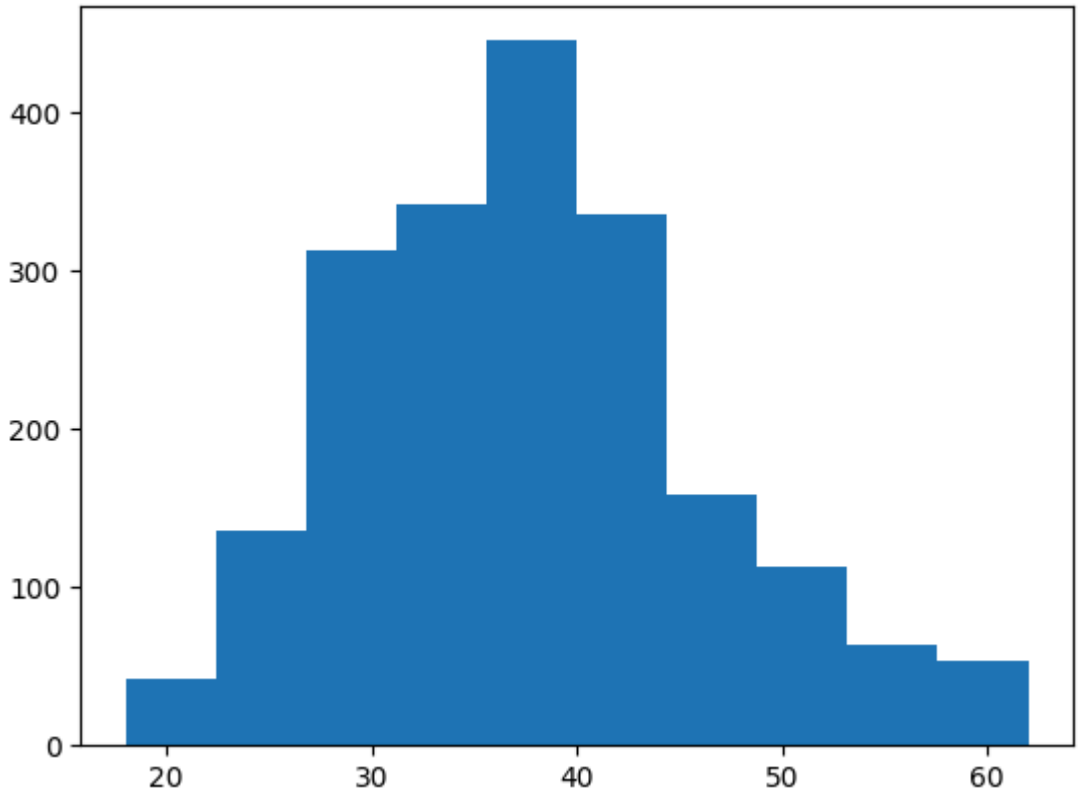
In [58]:
```
tele_df=pd.read_csv("telecom_churn_data.csv")
tele_df
```

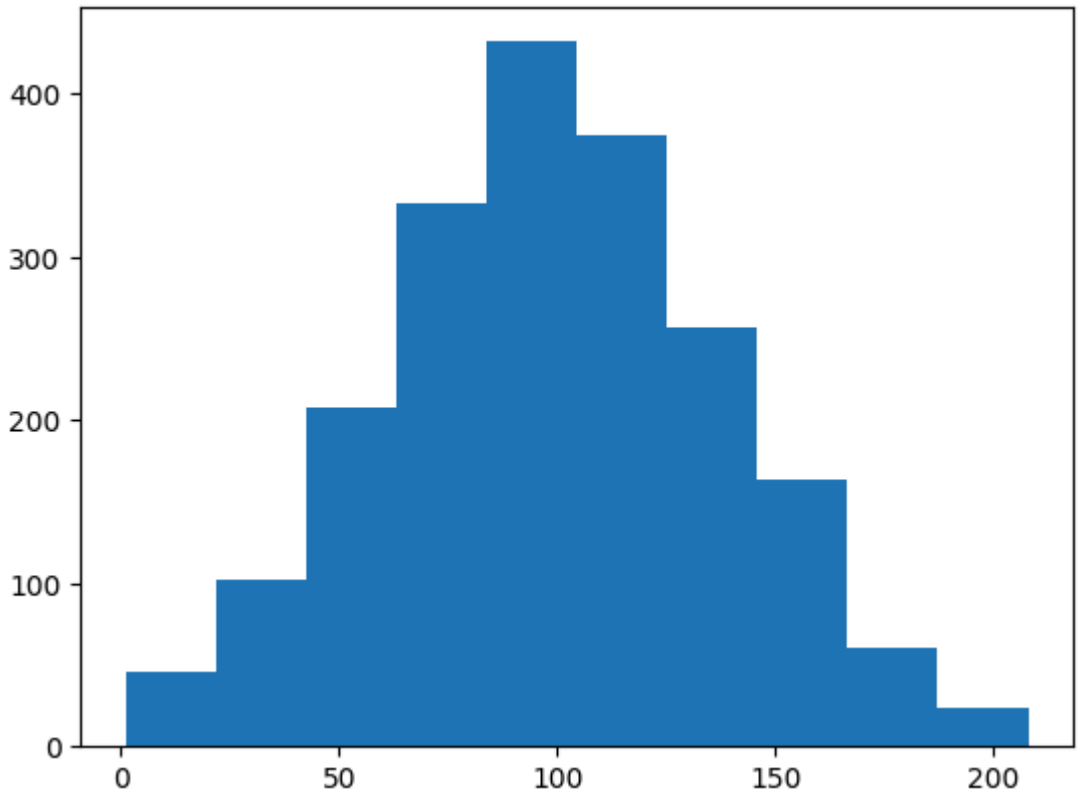
Out[58]:

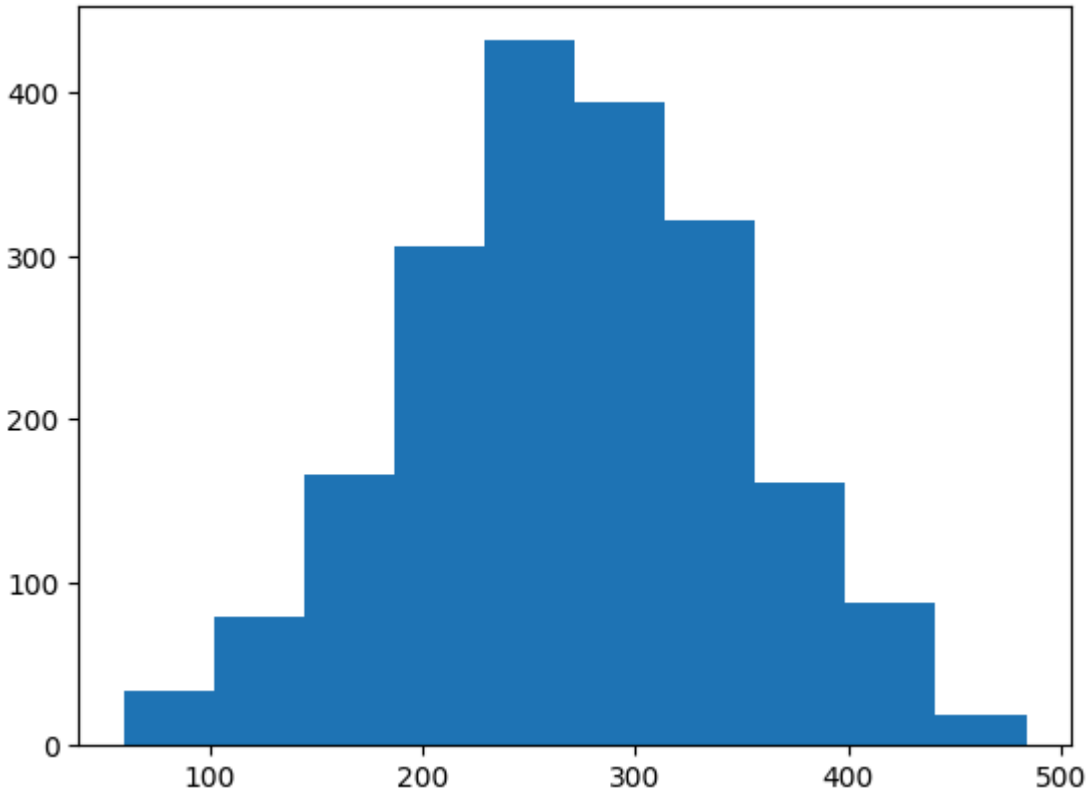
	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	409-8743	Female	36	62	no
1	2015	100643	340-5930	Female	39	149	no
2	2015	100756	372-3750	Female	65	126	no
3	2015	101595	331-4902	Female	24	131	no
4	2015	101653	351-8398	Female	40	191	no
...	...	...	...	...	...	...	...
1995	2015	997132	385-7387	Female	54	75	no
1996	2015	998086	383-9255	Male	45	127	no
1997	2015	998474	353-2080	NaN	53	94	no
1998	2015	998934	359-7788	Male	40	94	no
1999	2015	999961	414-1496	Male	37	73	no

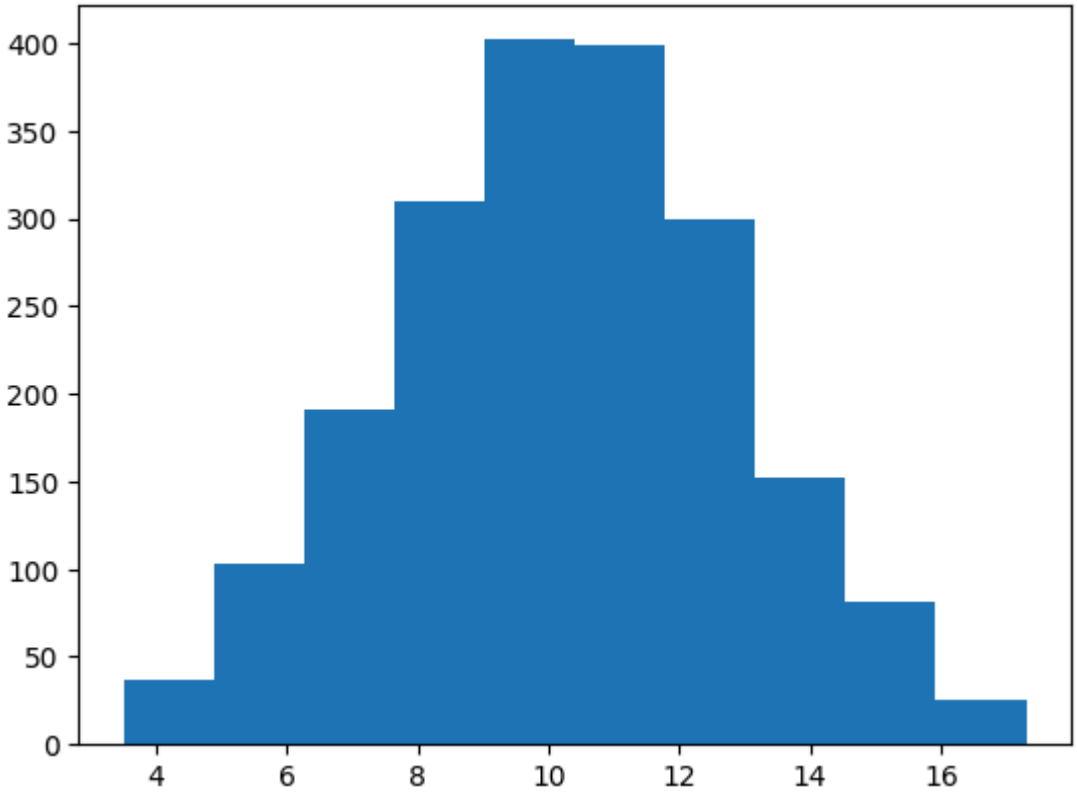
2000 rows × 16 columns

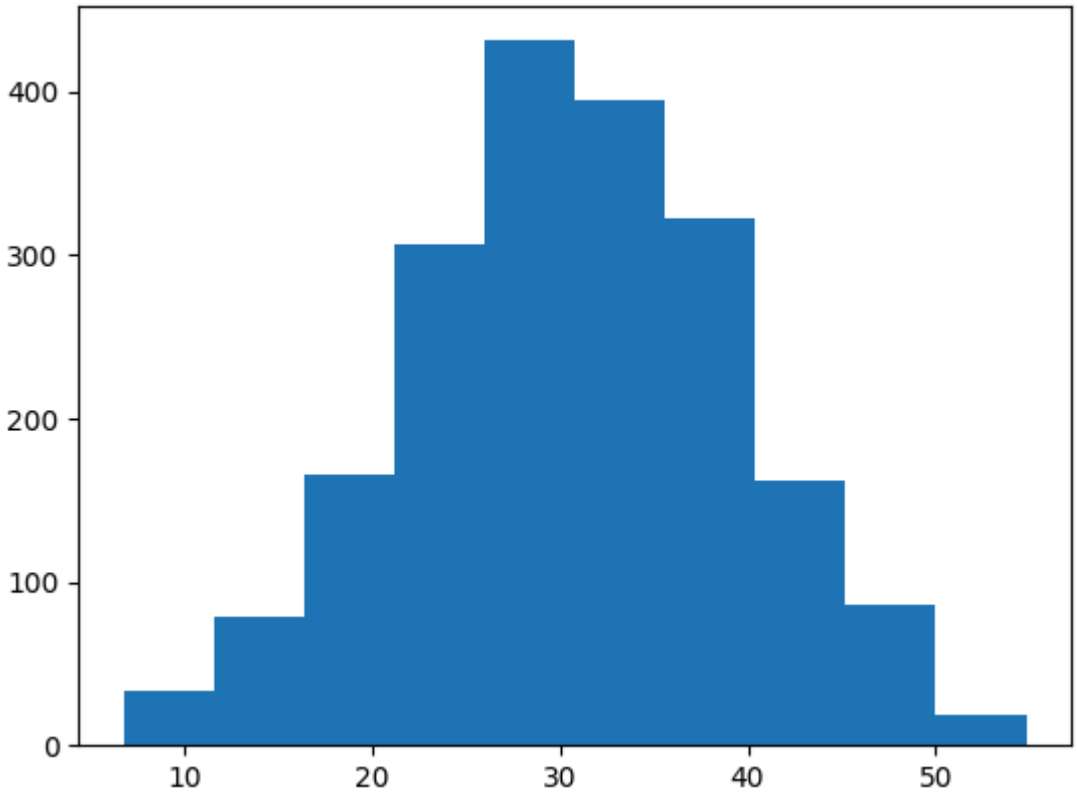
In [59]:
```
for i in  num_clm_2[2:]:

    q1=np.quantile(tele_df[i],q=0.25)
    q2=np.quantile(tele_df[i],q=0.50)
    q3=np.quantile(tele_df[i],q=0.75)


    iqr=q3-q1
    lb=q1-1.5*iqr
    ub=q3+1.5*iqr
    cont=(tele_df[i]>ub)| (tele_df[i]<lb)
    true=np.median(tele_df[i])
    false=tele_df[i]

    v=np.where(cont,true,false)
    tele_df[i]=v


    plt.hist(v)
    plt.show()
```

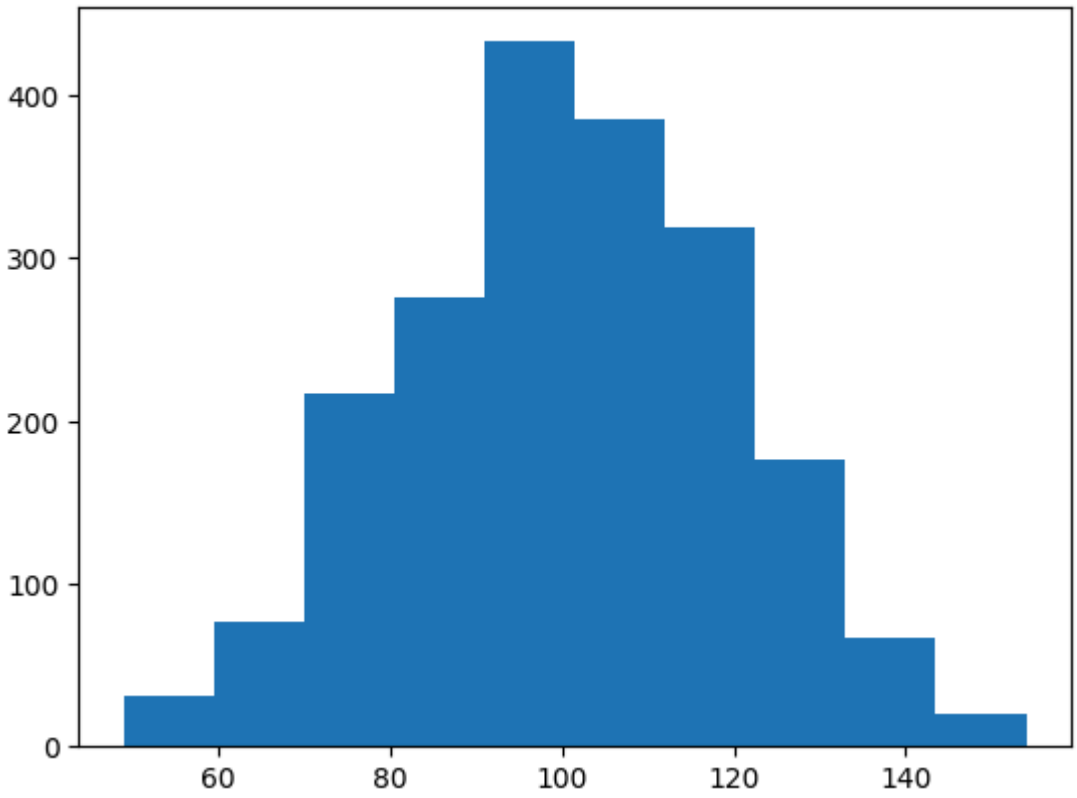

400
300
200
100
0
20
30
40
50
60

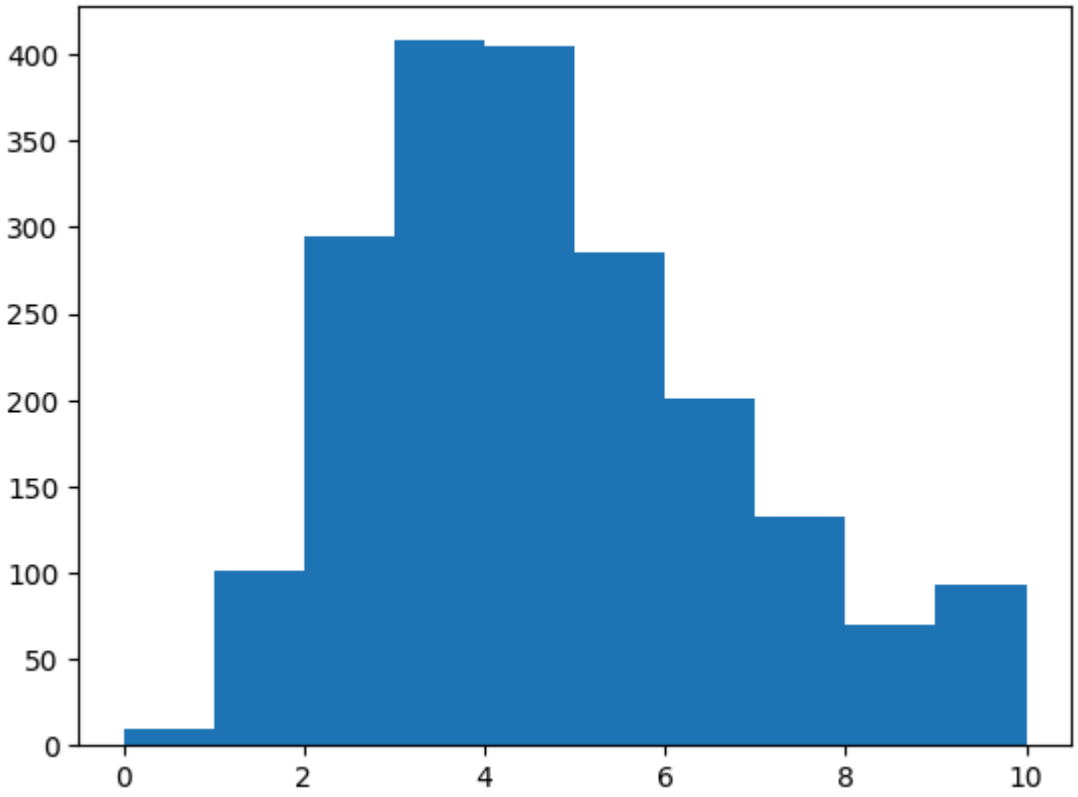

400
300
200
100
0
0
50
100
150
200

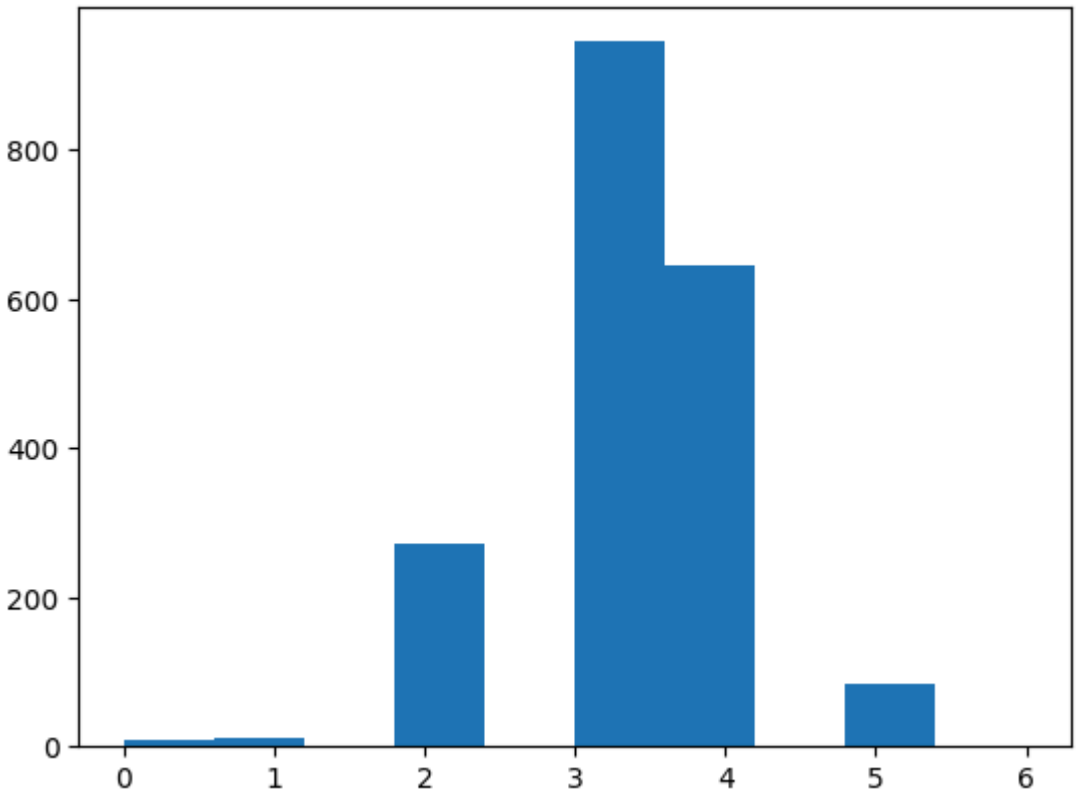

400
300
200
100
0
100
200
300
400
500

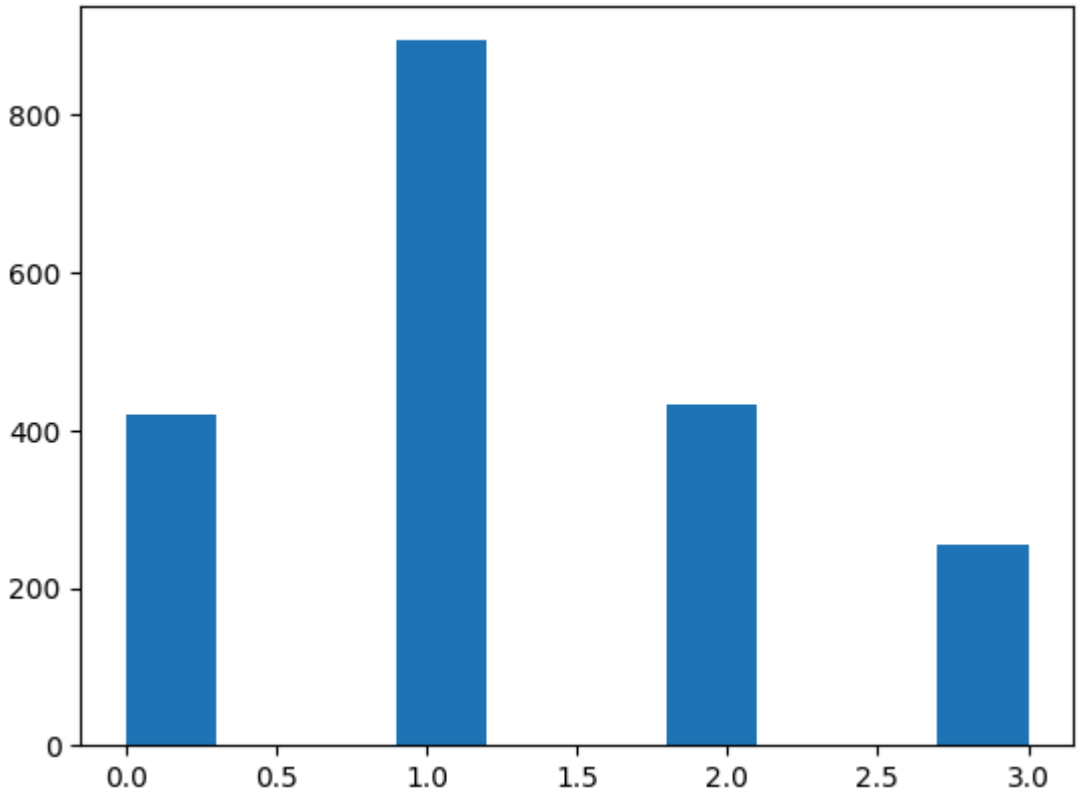

400
350
300
250
200
150
100
50
0
4
6
8
10
12
14
16


400
300
200
100
0
10
20
30
40
50


400
300
200
100
0
60
80
100
120
140


400
350
300
250
200
150
100
50
0
0
2
4
6
8
10

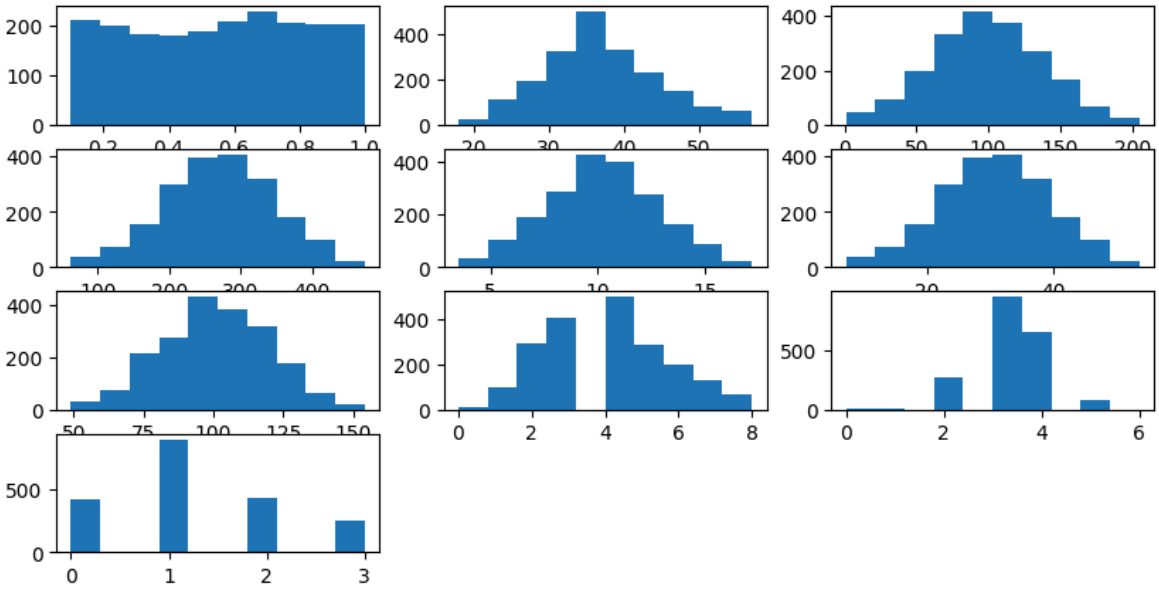

800
600
400
200
0
0
1
2
3
4
5
6



In [ ]:

DATA DISTRIBUTION WITH SUBPLOT

In [61]:
```
plt.figure(figsize=(10,5))
for i in range(1,len(num_clm_2)):
    plt.subplot(4,3,i)

    q1=np.quantile(tele_df[num_clm[i]],q=0.25)
    q2=np.quantile(tele_df[num_clm[i]],q=0.50)
    q3=np.quantile(tele_df[num_clm[i]],q=0.75)


    iqr=q3-q1
    lb=q1-1.5*iqr
    ub=q3+1.5*iqr
    cont=(tele_df[num_clm[i]]>ub)| (tele_df[num_clm[i]]<lb)
    true=np.median(tele_df[num_clm[i]])
    false=tele_df[num_clm[i]]

    v=np.where(cont,true,false)
    tele_df[i]=v


    plt.hist(v)
plt.show()
```

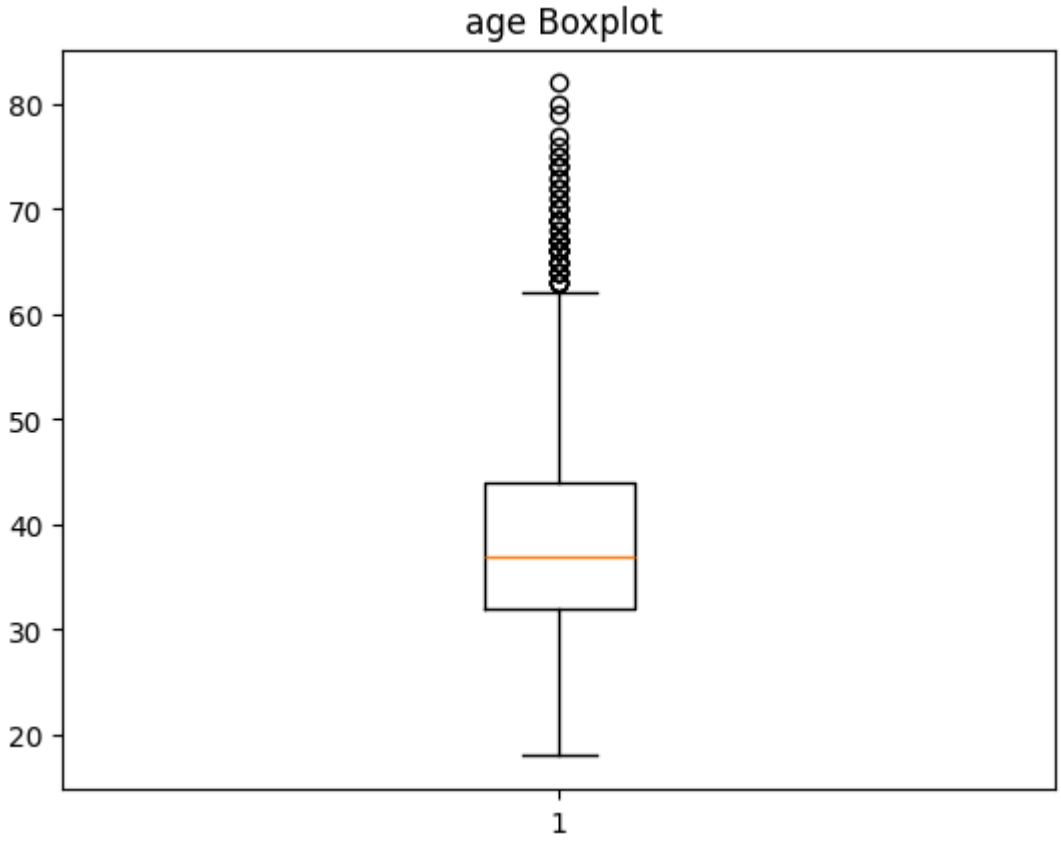


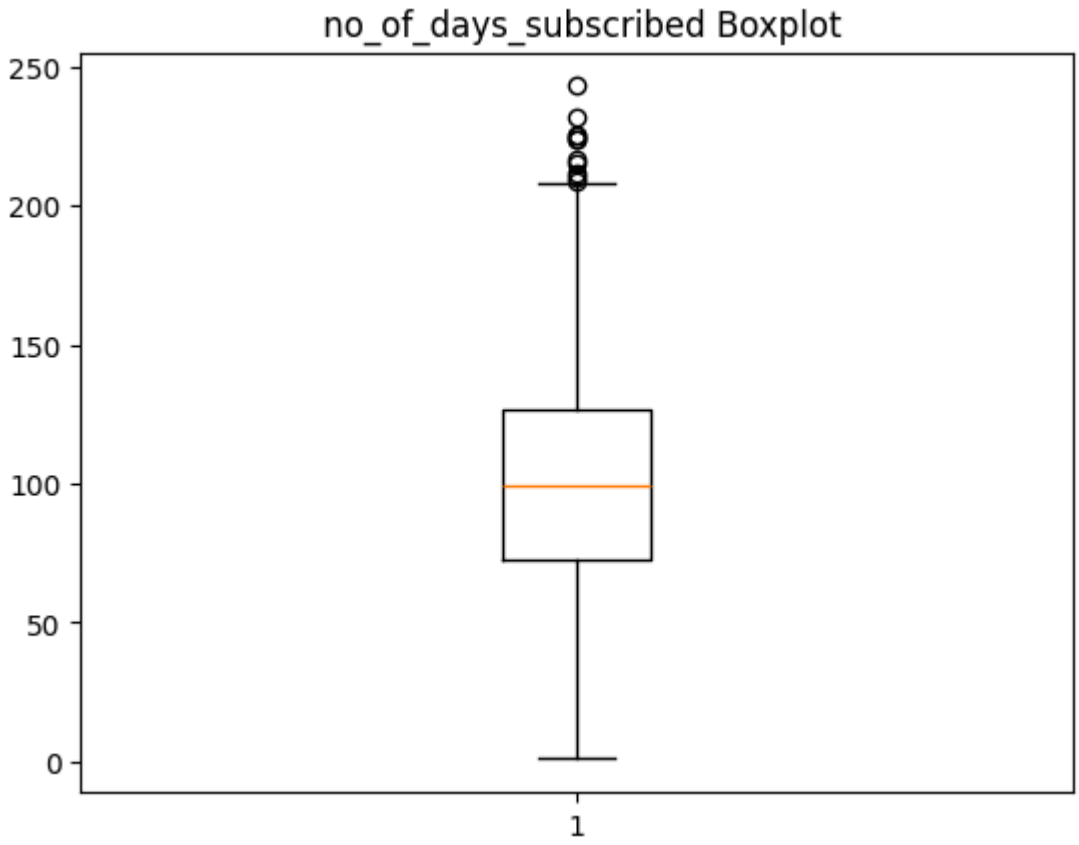
In [ ]:

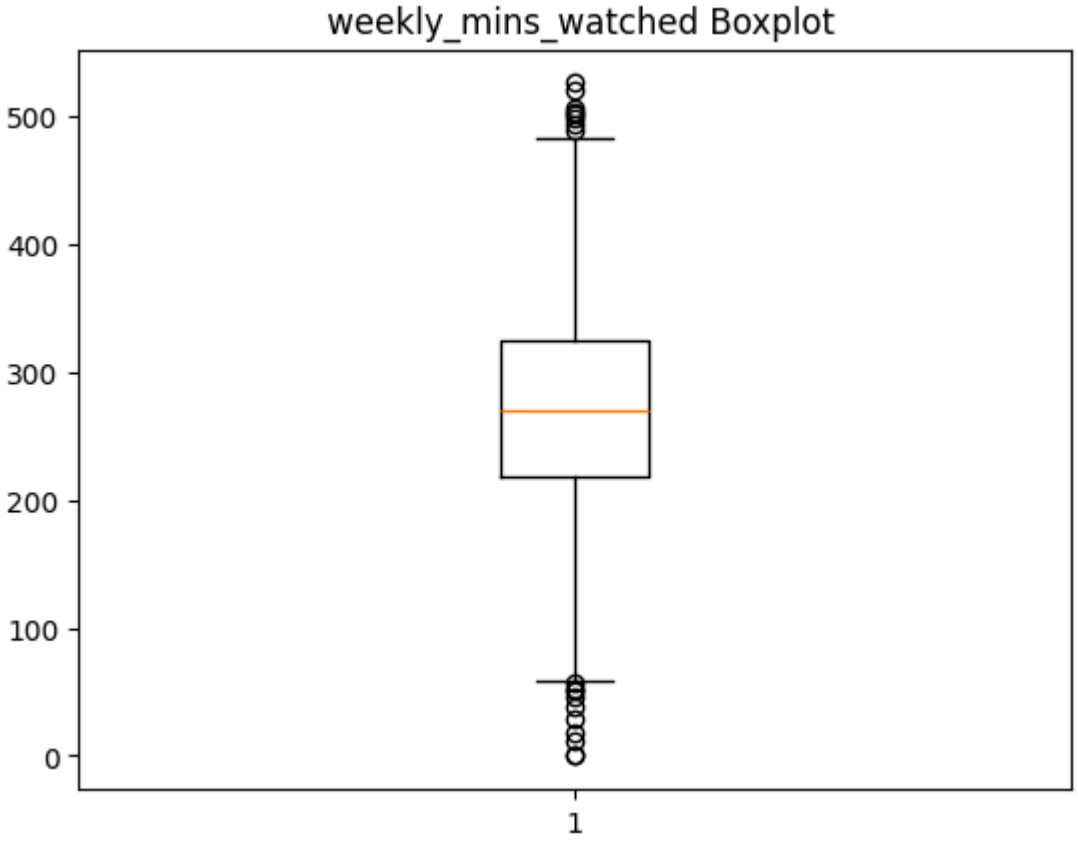
BOXPLOT

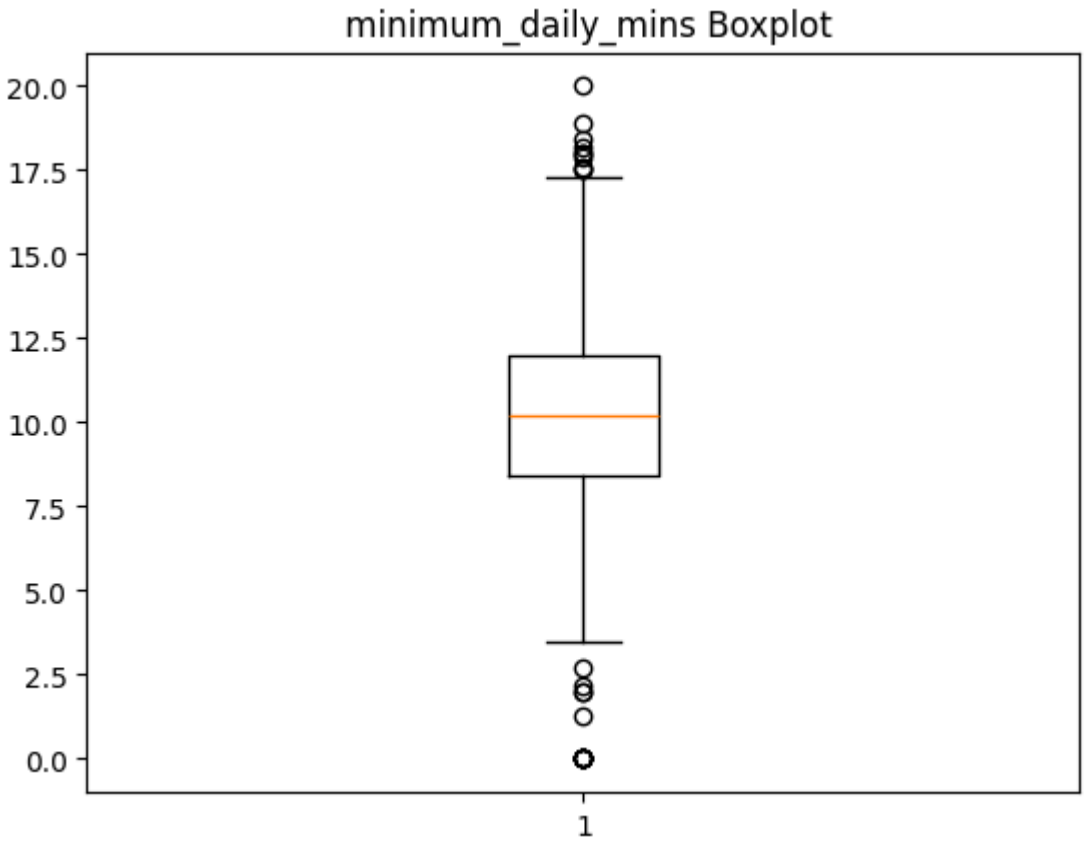
- box plot is used to identify the ouliers
- Outliers is an observtion having huge positive value or huge negative value

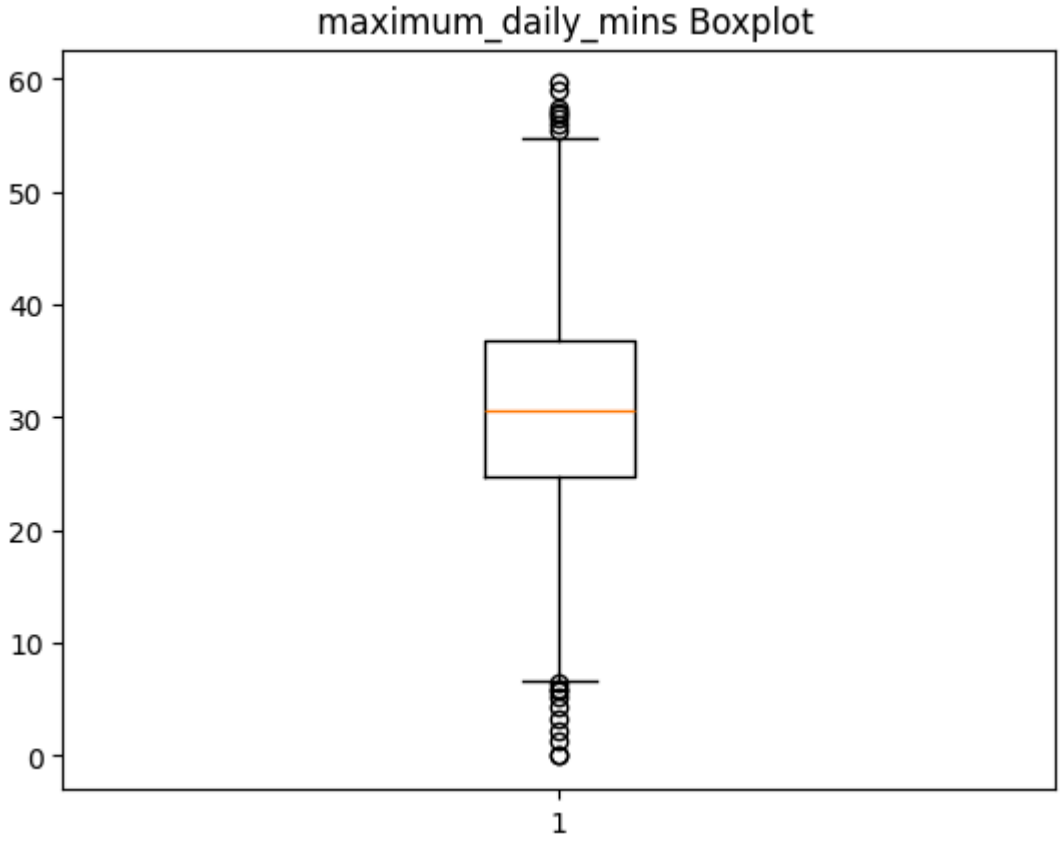
In [64]:
```
def boxplot():
    tele_df=pd.read_csv("telecom_churn_data.csv")

    for i in num_clm_2[2:]:
        box_data=tele_df[i]
        plt.boxplot(box_data)
        plt.title(f" {i} Boxplot")
        plt.show()
boxplot()
```

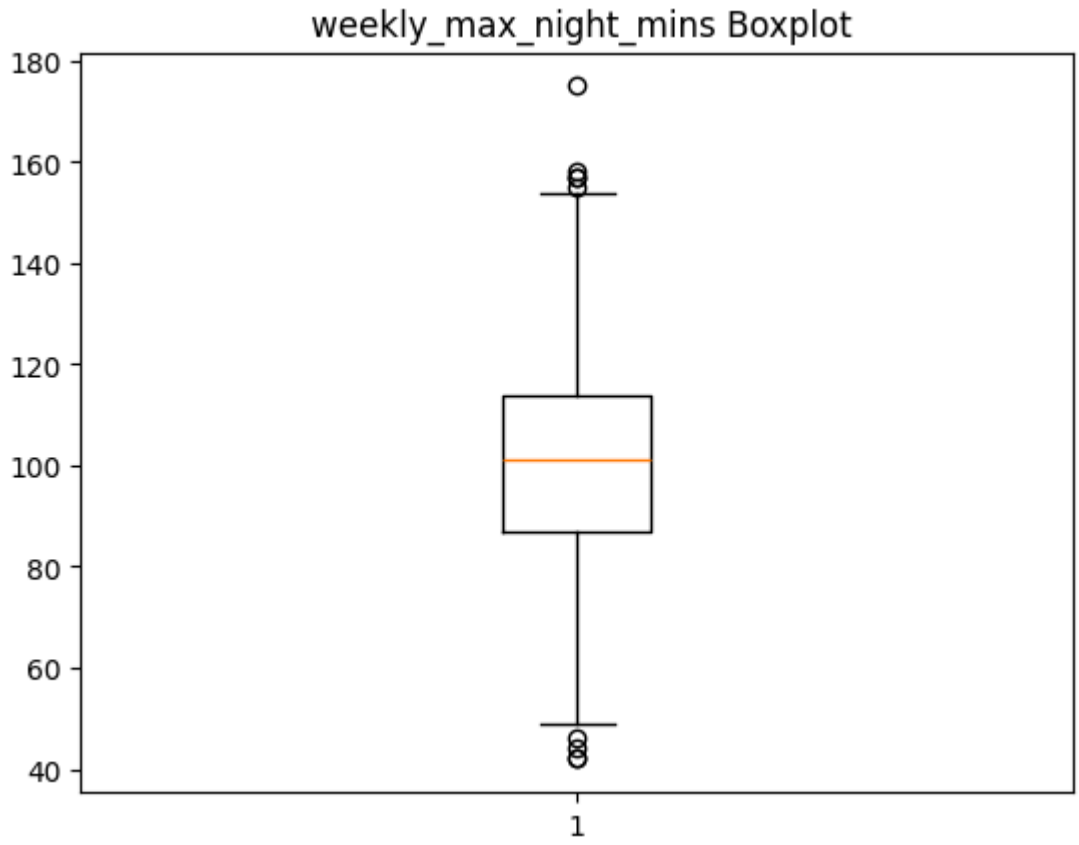

age Boxplot
80
70
60
50
40
30
20
1

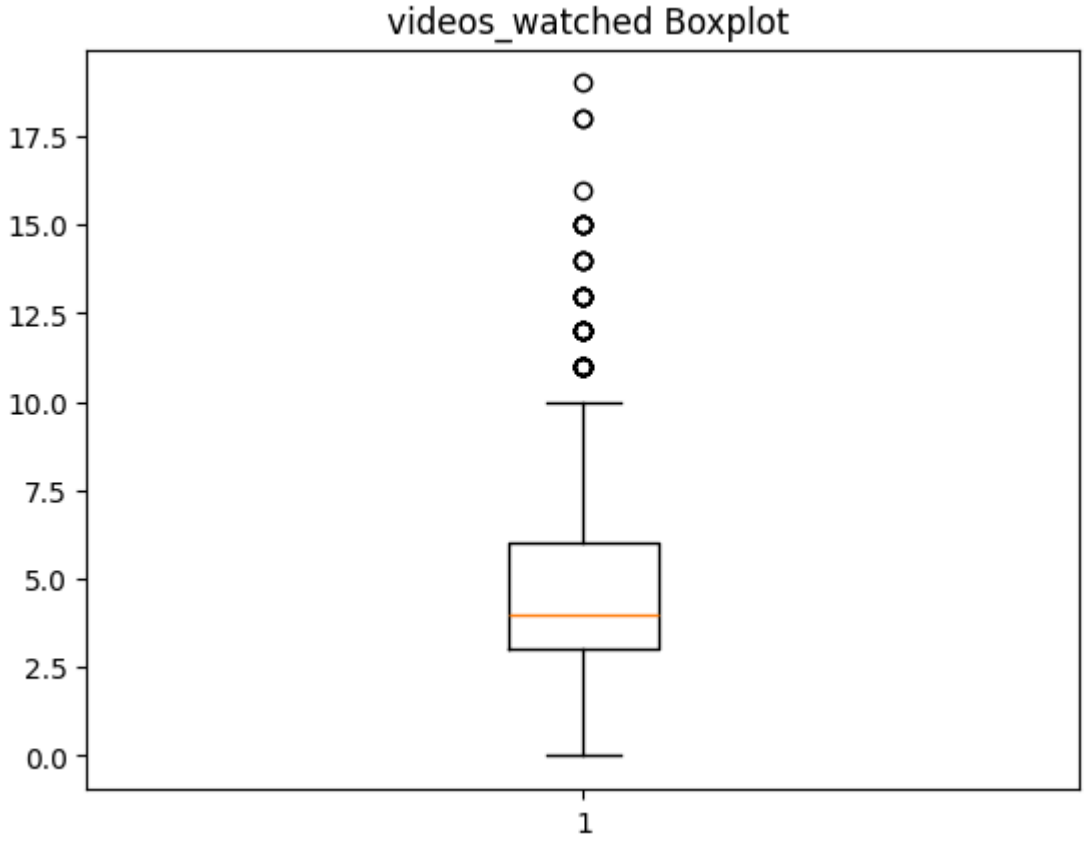

no_of_days_subscribed Boxplot
250
200
150
100
50
0
1

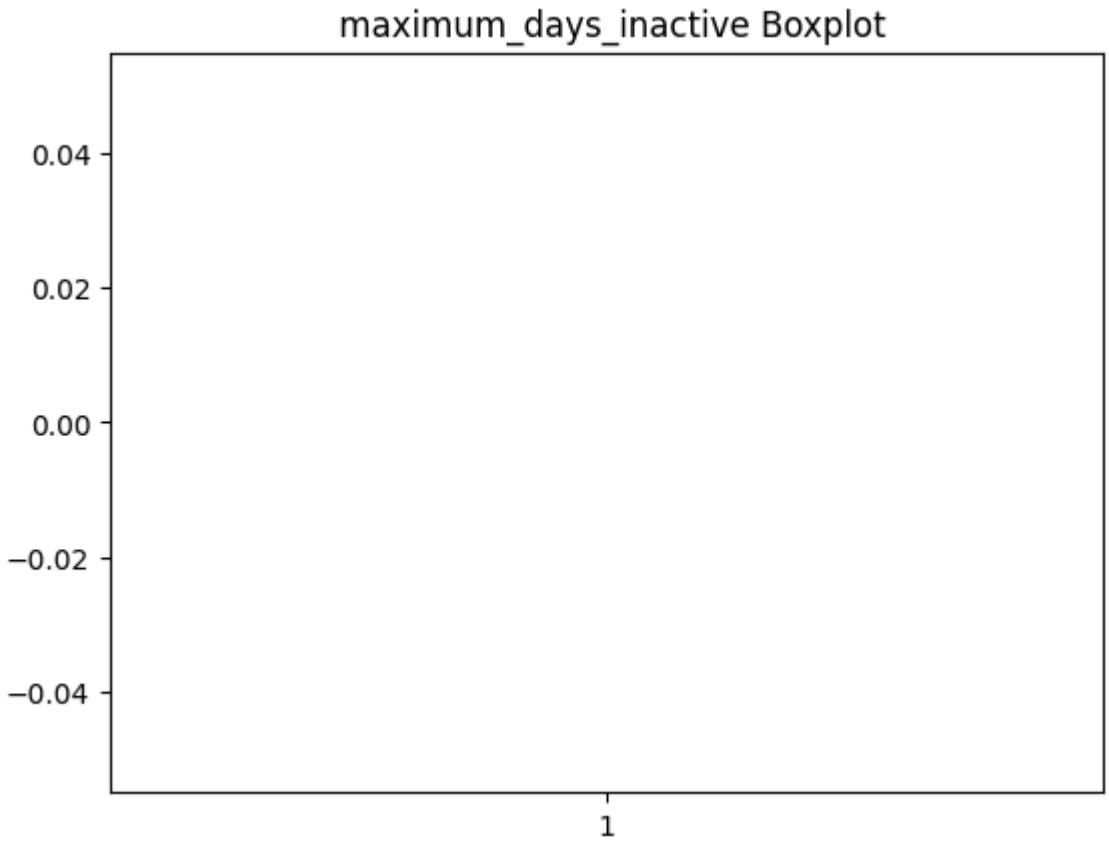

weekly_mins_watched Boxplot
500
400
300
200
100
0
1

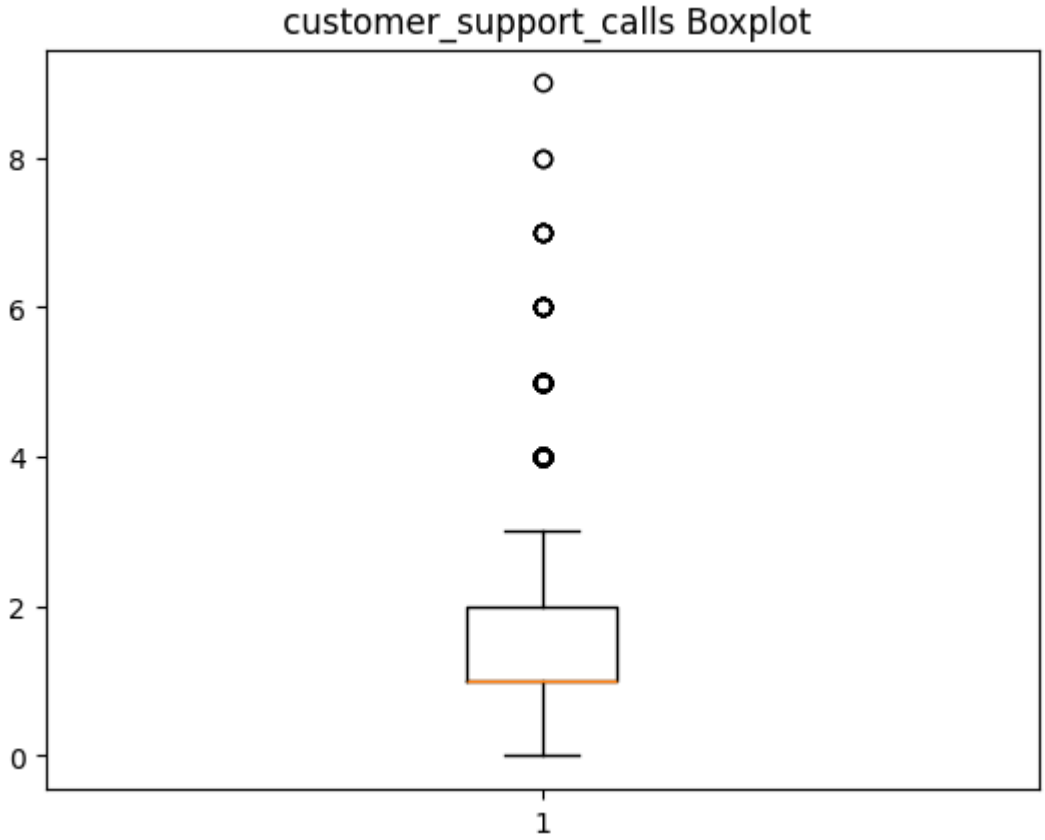

minimum_daily_mins Boxplot
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1


maximum_daily_mins Boxplot
60
50
40
30
20
10
0
1


weekly_max_night_mins Boxplot
180
160
140
120
100
80
60
40
1


videos_watched Boxplot
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1


maximum_days_inactive Boxplot
0.04
0.02
0.00
−0.02
−0.04
1



In [ ]:

OUTLIERS ANALYSIS

- We know that Outliers available less than Q1-1.5IQR and Q3+1.5IQR
- step-1:
  - Calculate Q1=25p
  - Calculate Q2=50p
  - Calculate Q3=75p
- step-2:
  - Calculate IQR = Q3-Q1
- step-3 :
  - Calculate lb=Q1-1.5IQR
  - calculate ub=Q1+1.5IQR
- step-4 :
  - con1 = data < lb
  - con2 = data > ub
  - con3 = con1 or-and con2 (bitwise operator)
- step-5
  - data[con3]

In [ ]:

OUTLIERS DATA

In [68]:
```
tele_df = pd.read_csv("telecom_churn_data.csv")
outliers_data = pd.DataFrame()

for i in num_clm_2:
    q1 = np.quantile(tele_df[i], q=0.25)
    q2 = np.quantile(tele_df[i], q=0.50)
    q3 = np.quantile(tele_df[i], q=0.75)

    iqr = q3 - q1

    lb = q1 - 1.5 * iqr
    ub = q3 + 1.5 * iqr


    con1 = tele_df[i] > ub
    con2 = tele_df[i] < lb
    con3 = con1 | con2


    # Append outliers for this column
    outliers_data = pd.concat([outliers_data, tele_df[con3]])
```

In [69]:
```
outliers_data
```

Out[69]:

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
2	2015	100756	372-3750	Female	65	126	no
30	2015	111252	410-9633	Female	63	106	no
71	2015	124837	413-2241	Male	67	163	no
87	2015	131070	334-9182	Male	64	21	no
154	2015	165319	403-9733	Female	66	68	no
...	...	...	...	...	...	...	...
1884	2015	950907	412-2520	Male	69	73	no
1887	2015	952646	332-5521	Male	37	122	no
1926	2015	968500	345-1524	Male	36	101	no
1931	2015	969732	354-3237	Female	34	98	no
1933	2015	970012	368-3078	Male	47	102	no

356 rows × 16 columns

In [ ]:
```
```

NONOUTLIERS DATA

In [71]:
```
tele_df=pd.read_csv(r"C:\Users\HP\Desktop\Naresh IT\Data Set Analysis\Telecom_Ch
Nonoutliers_data=pd.DataFrame()
for i in num_clm_2:
```

```
    q1=np.percentile(tele_df[i],q=0.25)
    q2=np.percentile(tele_df[i],q=0.50)
    q3=np.percentile(tele_df[i],q=0.75)

    iqr=q3-q1

    lb=q1-1.5*iqr
    ub=q3+1.5*iqr

    con1=tele_df[i]>lb
    con2=tele_df[i]<ub
    con3=con1 & con2
    #Append outliers for this column
    Nonoutliers_data=pd.concat([Nonoutliers_data,tele_df[con3]])
```

In [72]:
```
Nonoutliers_data
```

Out[72]:

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	409-8743	Female	36	62	no
1	2015	100643	340-5930	Female	39	149	no
2	2015	100756	372-3750	Female	65	126	no
3	2015	101595	331-4902	Female	24	131	no
4	2015	101653	351-8398	Female	40	191	no
...	...	...	...	...	...	...	...
1956	2015	979406	366-7360	Male	47	129	no
1963	2015	981937	380-5873	Male	37	46	no
1973	2015	986715	353-7461	Male	37	128	no
1974	2015	987122	364-8981	Female	26	54	no
1983	2015	989511	346-4216	Female	43	154	no

1291 rows × 16 columns

In [ ]:
```
```

COMPARISON BOTH OUTLIERS DATA AND AFTER FILLING OUTLIERS DATA USING BOXPLOT

Outliers

In [73]:
```
tele_df=pd.read_csv("telecom_churn_data.csv")
num_clm=tele_df.select_dtypes(exclude="object").columns
num_clm
```

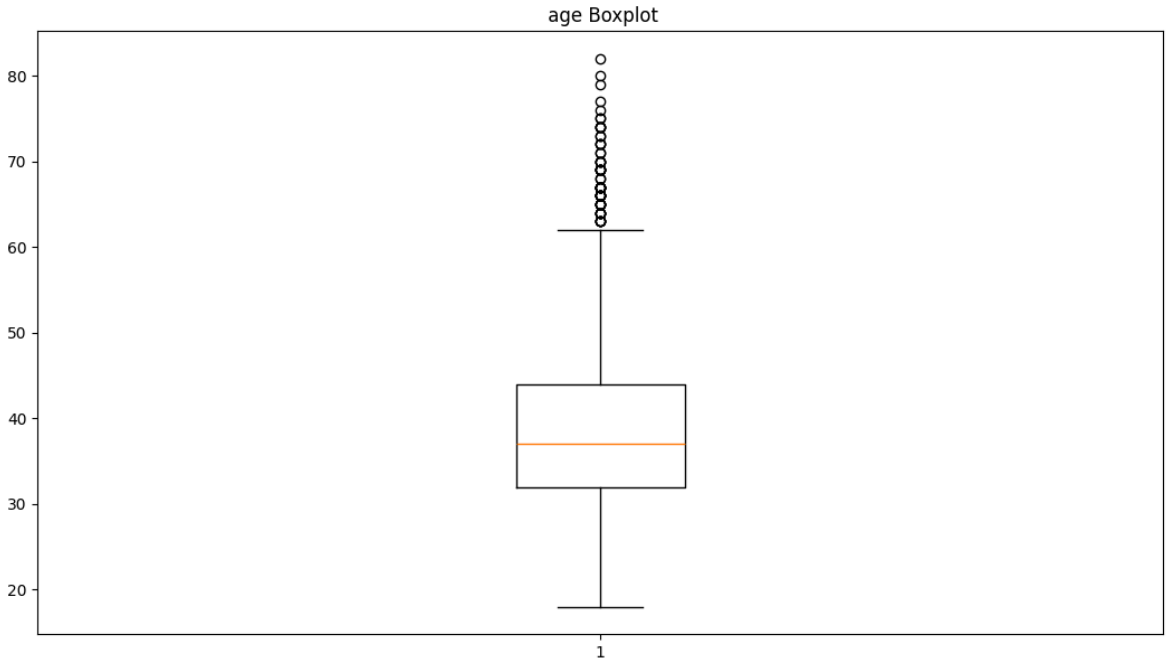
Out[73]:
```
Index(['year', 'customer_id', 'age', 'no_of_days_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls', 'churn'],
      dtype='object')
```

In [74]:
```
num_clm_2=num_clm.drop("churn")
```

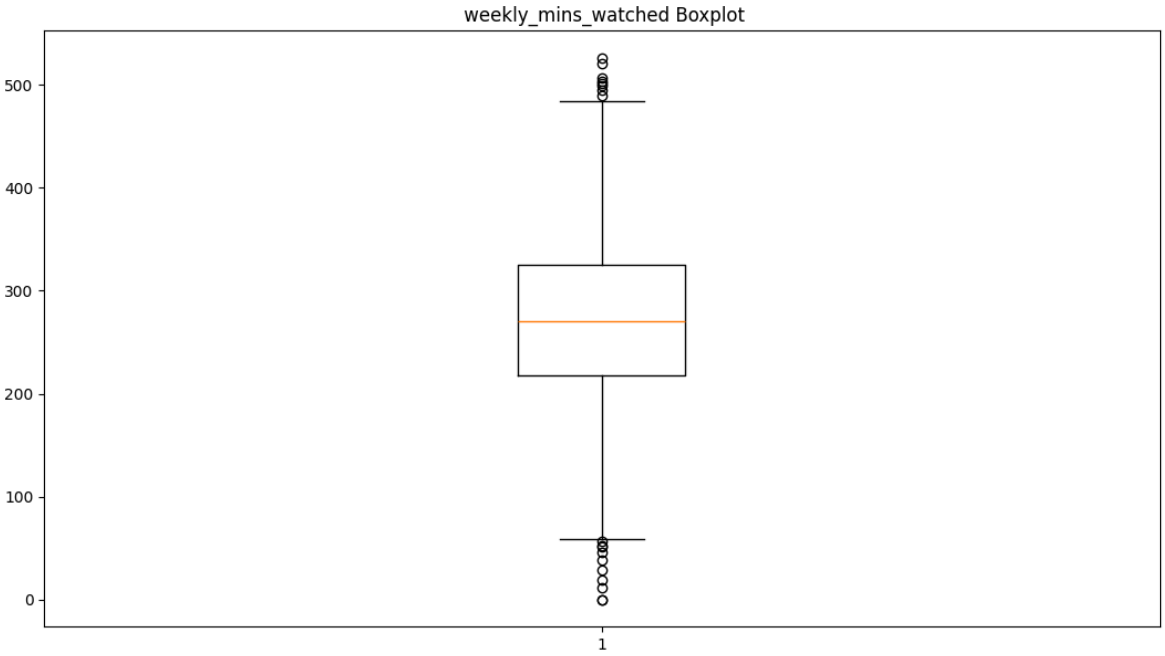
In [75]:
```
num_clm_2
```

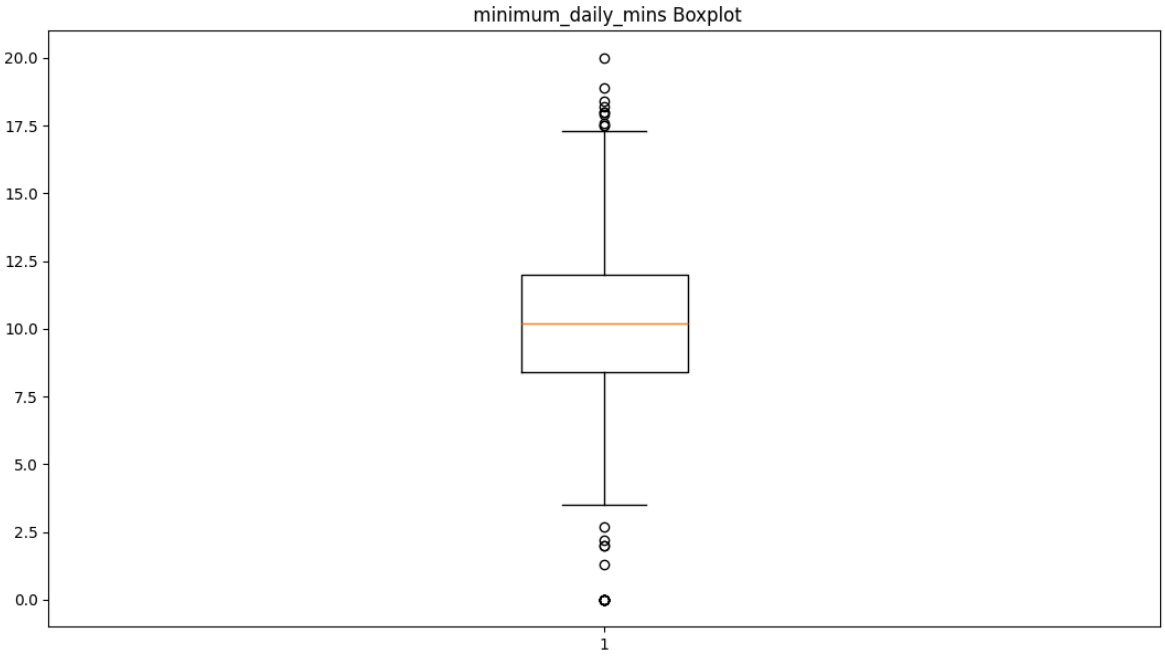
Out[75]:
```
Index(['year', 'customer_id', 'age', 'no_of_days_subscribed',
       'weekly_mins_watched', 'minimum_daily_mins', 'maximum_daily_mins',
       'weekly_max_night_mins', 'videos_watched', 'maximum_days_inactive',
       'customer_support_calls'],
      dtype='object')
```

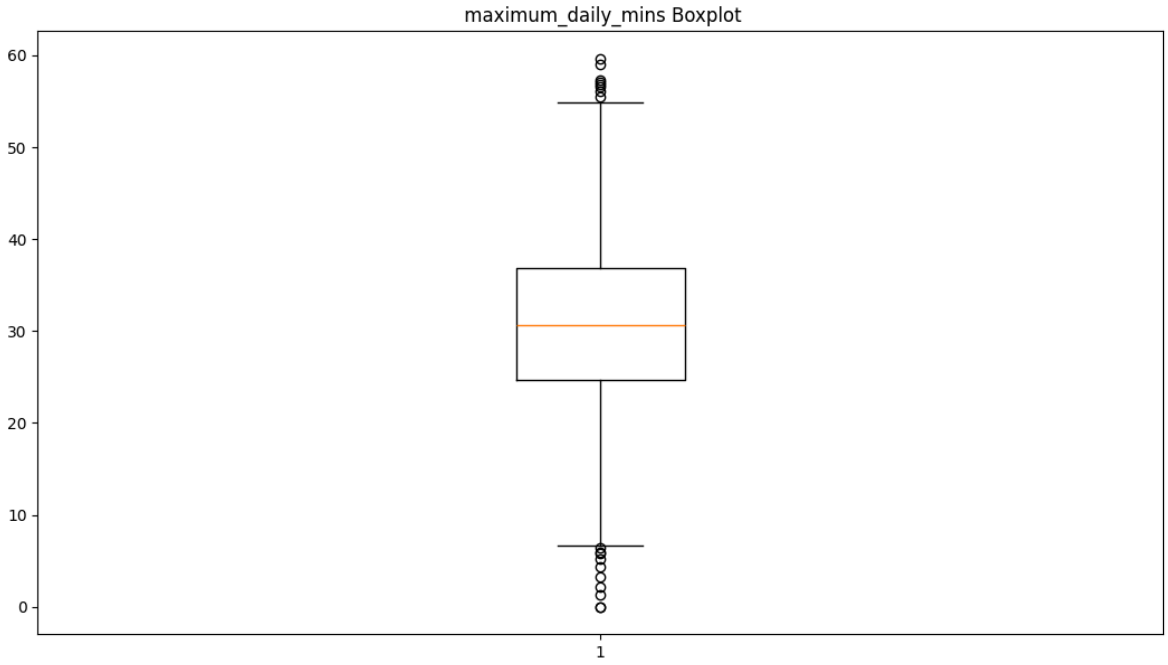
In [76]:
```
tele_df=pd.read_csv("telecom_churn_data.csv")
for i in num_clm_2[2:]:
    plt.figure(figsize=(13,7))

    plt.boxplot(tele_df[i])
    plt.title(f" {i} Boxplot")
```

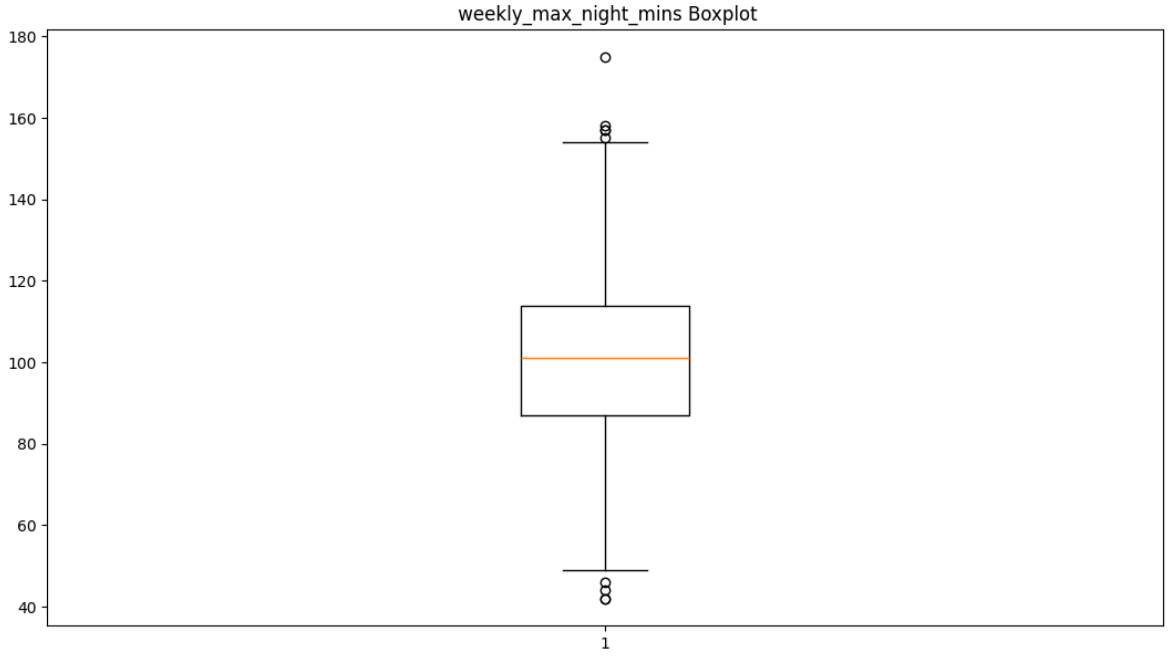


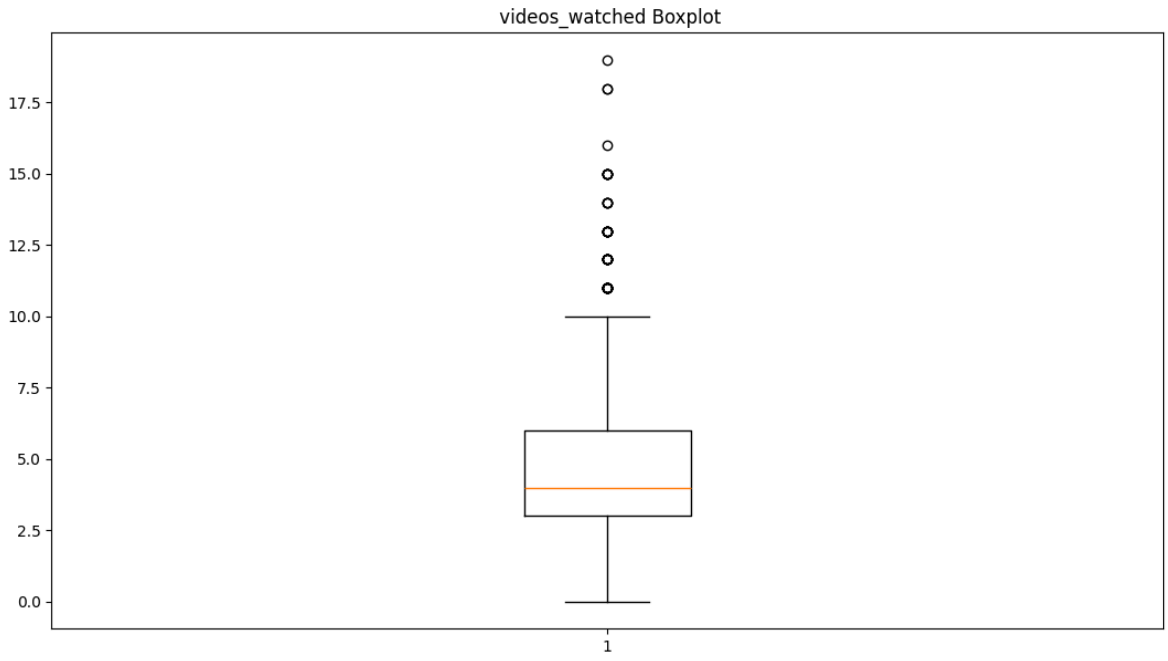

no_of_days_subscribed Boxplot
250
200
150
100
50
0
1

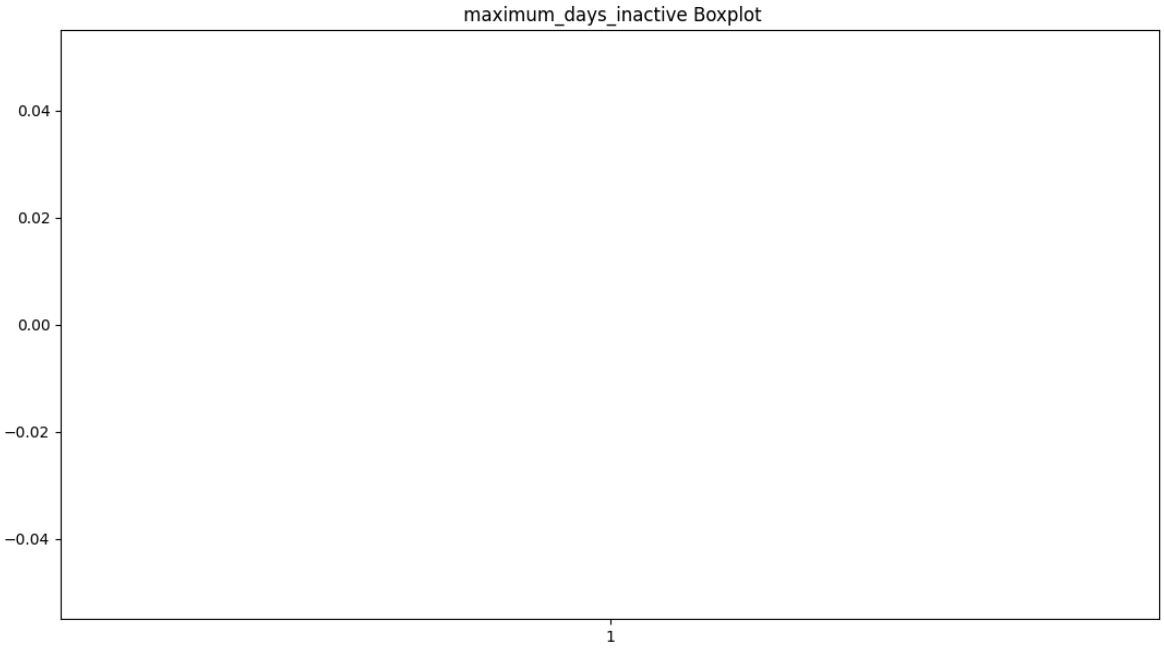

weekly_mins_watched Boxplot
500
400
300
200
100
0
1

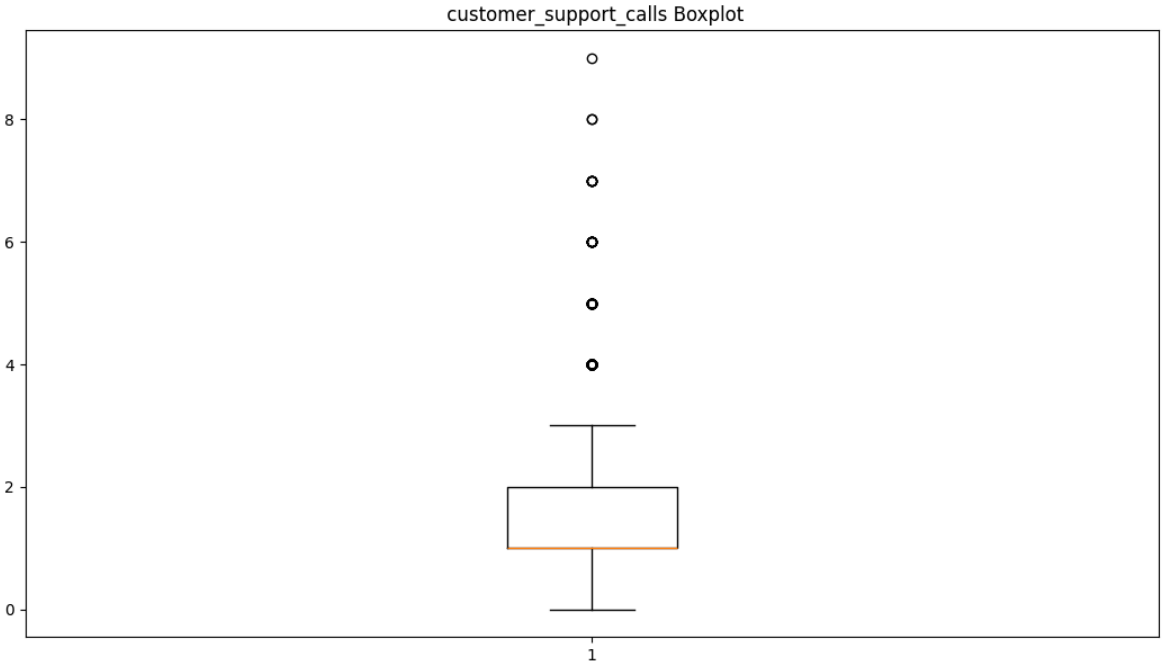

minimum_daily_mins Boxplot
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1


maximum_daily_mins Boxplot
60
50
40
30
20
10
0
1


weekly_max_night_mins Boxplot
180
160
140
120
100
80
60
40
1


videos_watched Boxplot
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1



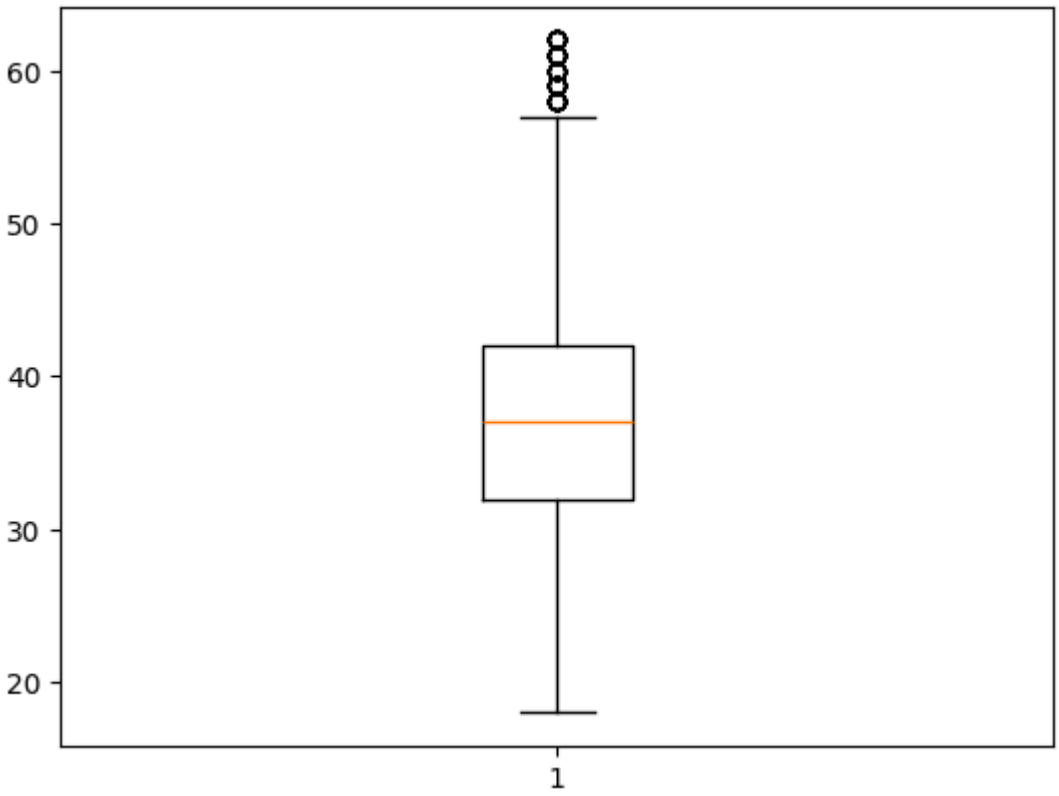


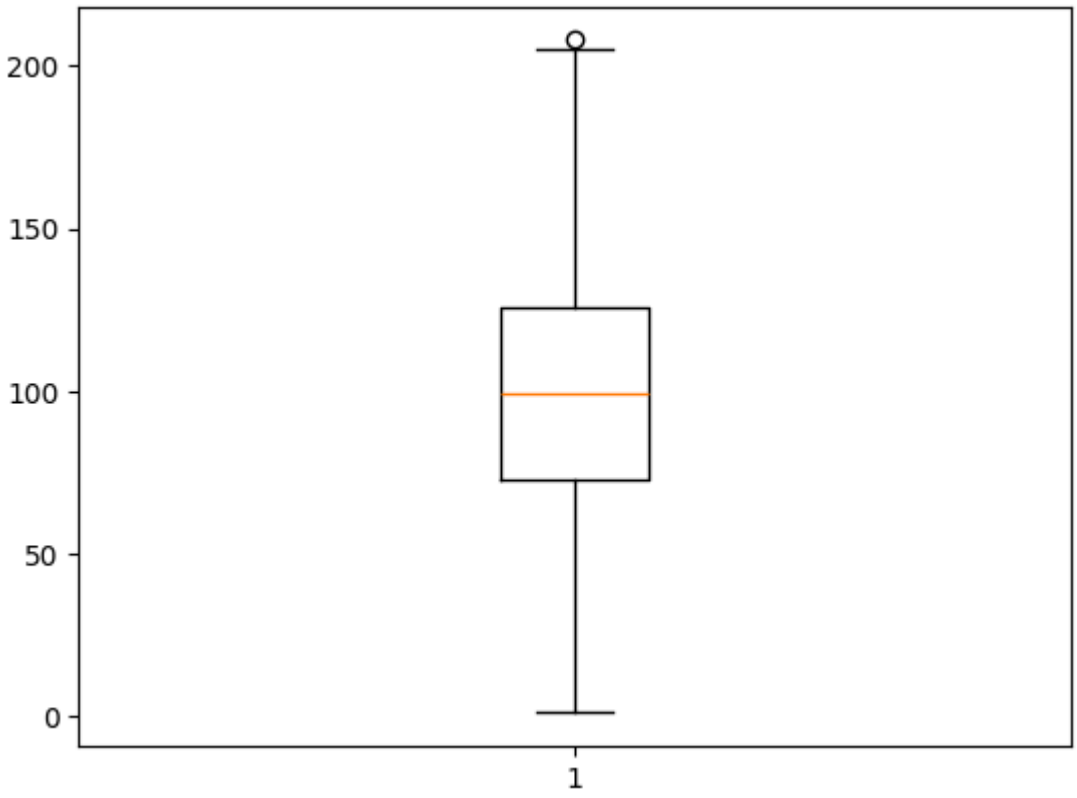
In [ ]:

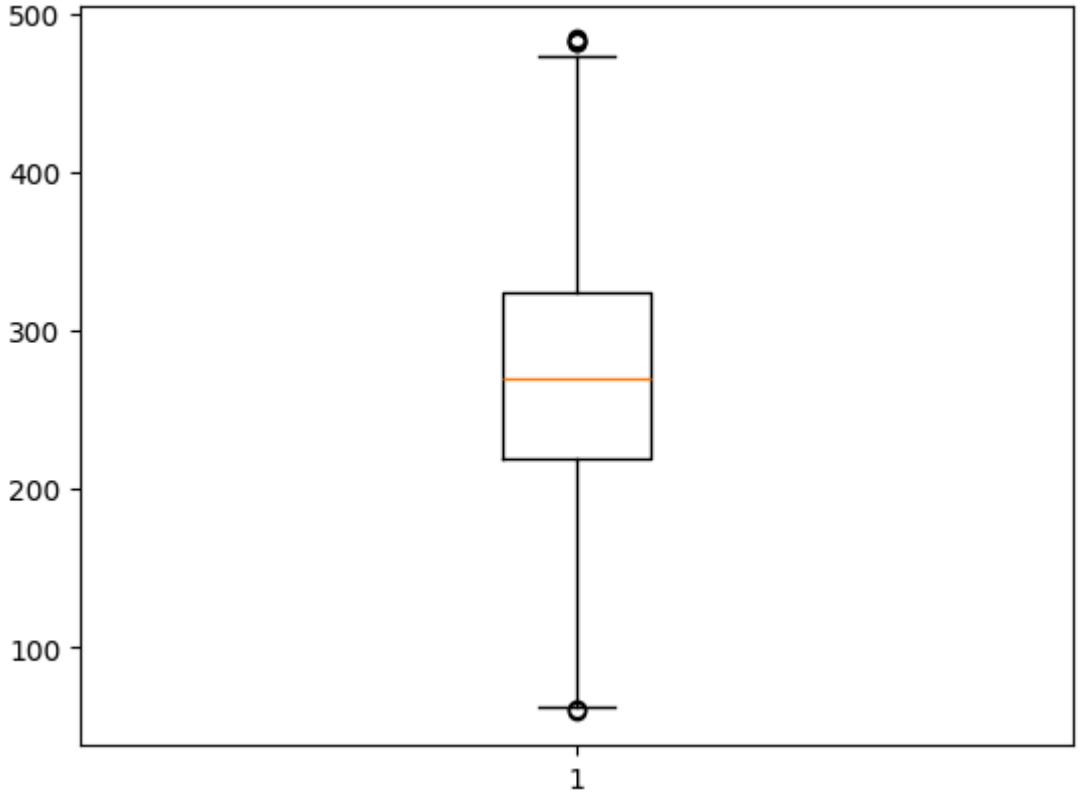
FILLING OUTLIERS DATA

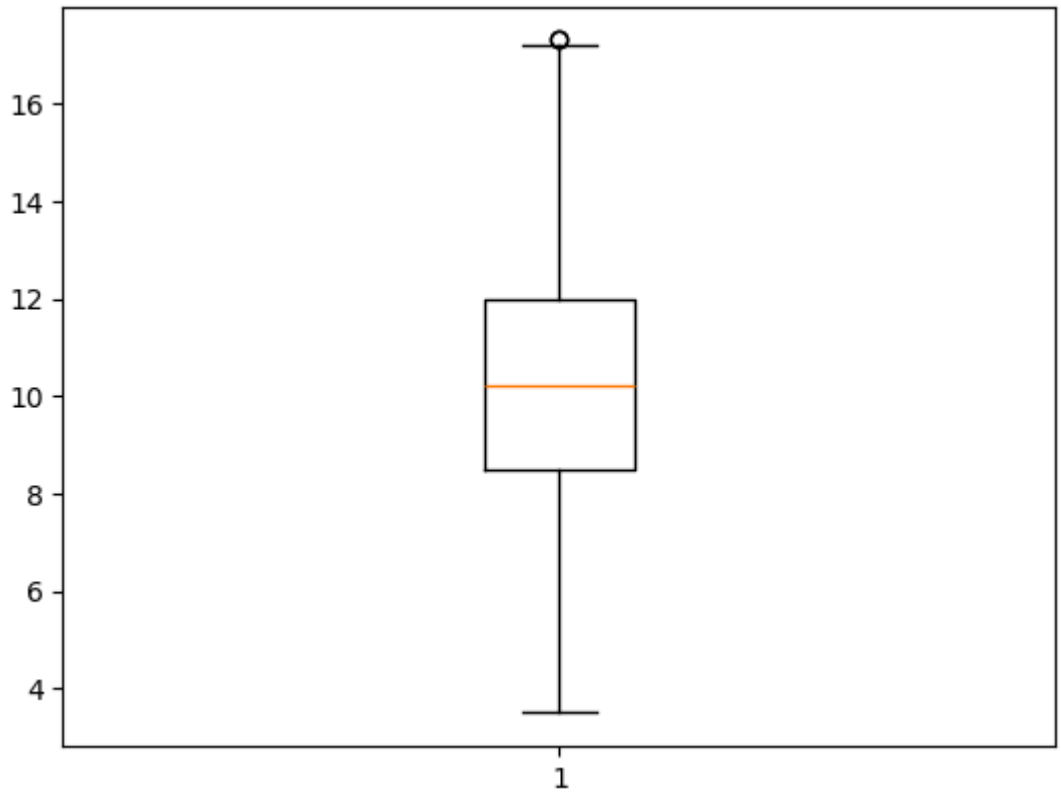
In [77]:
```
for i in num_clm_2[2:]:
    q1=np.quantile(tele_df[i],q=0.25)
    q2=np.quantile(tele_df[i],q=0.50)
    q3=np.quantile(tele_df[i],q=0.75)


    iqr=q3-q1

    lb=q1-1.5*iqr
    ub=q3+1.5*iqr

    con1=tele_df[i]<lb
    con2=tele_df[i]>ub
    con3=con1 | con2
```

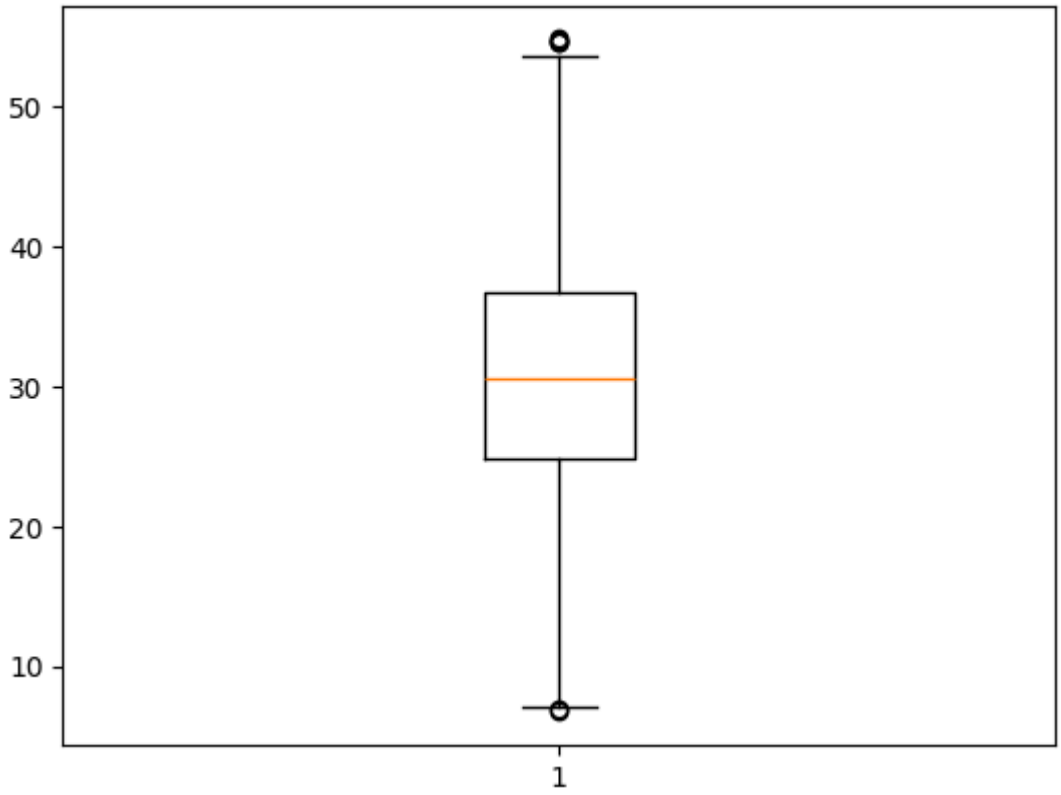
```
true=np.median(tele_df[i])
false=tele_df[i]

v=np.where(con3,true,false)
tele_df[i]=v

plt.boxplot(v)
plt.show()
```

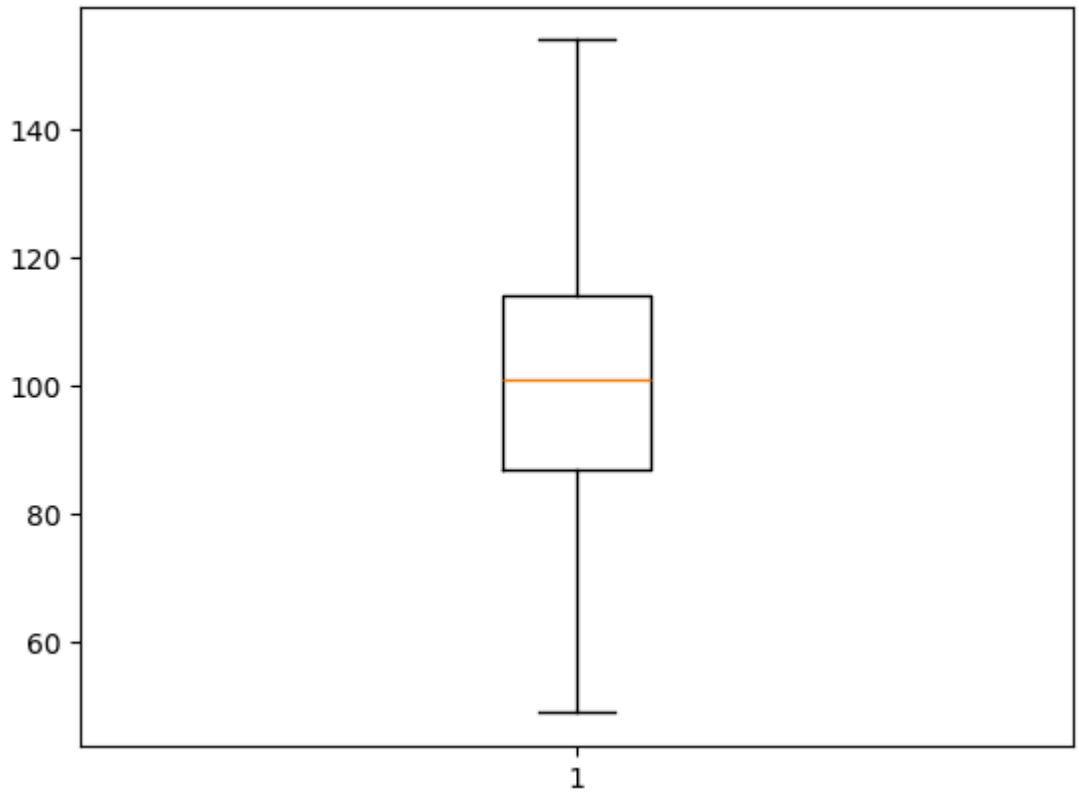


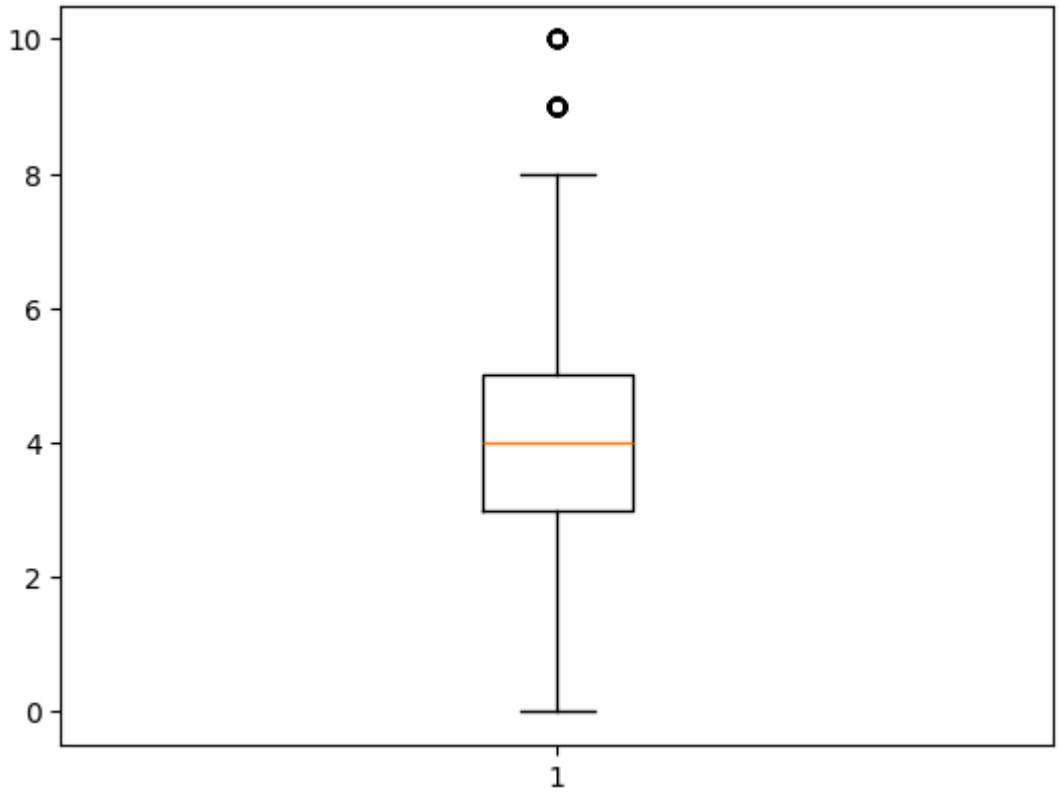


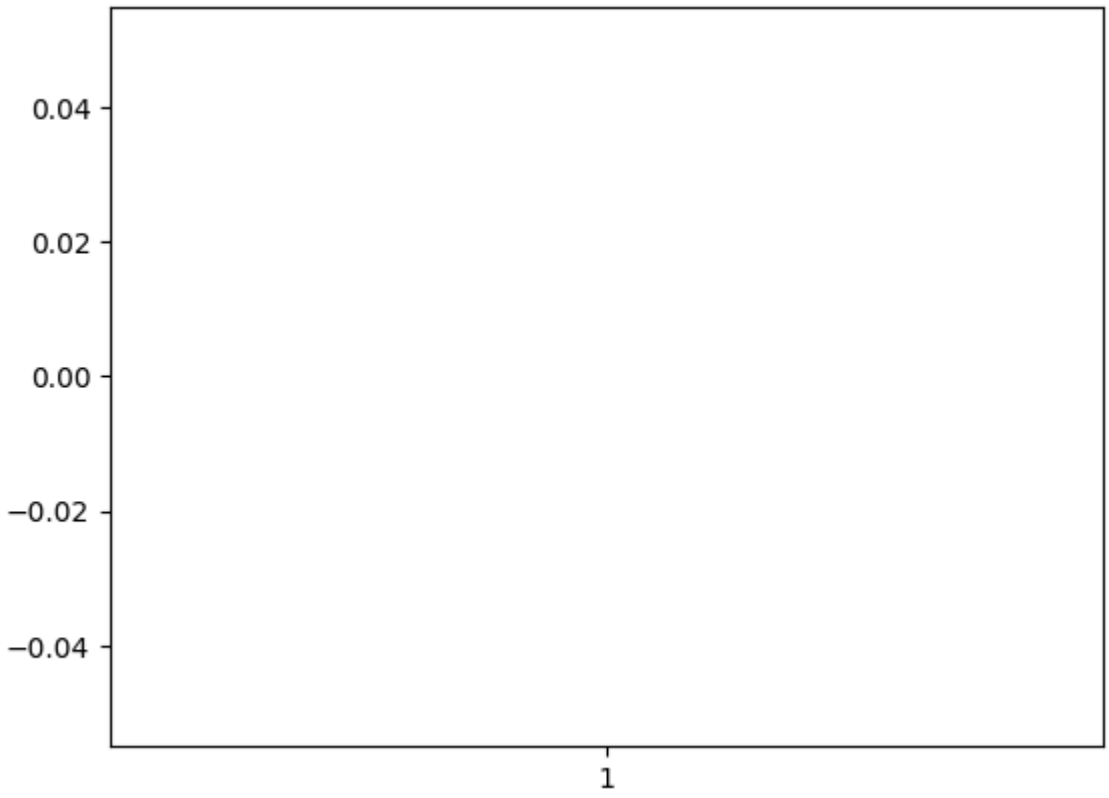

500
400
300
200
100
1

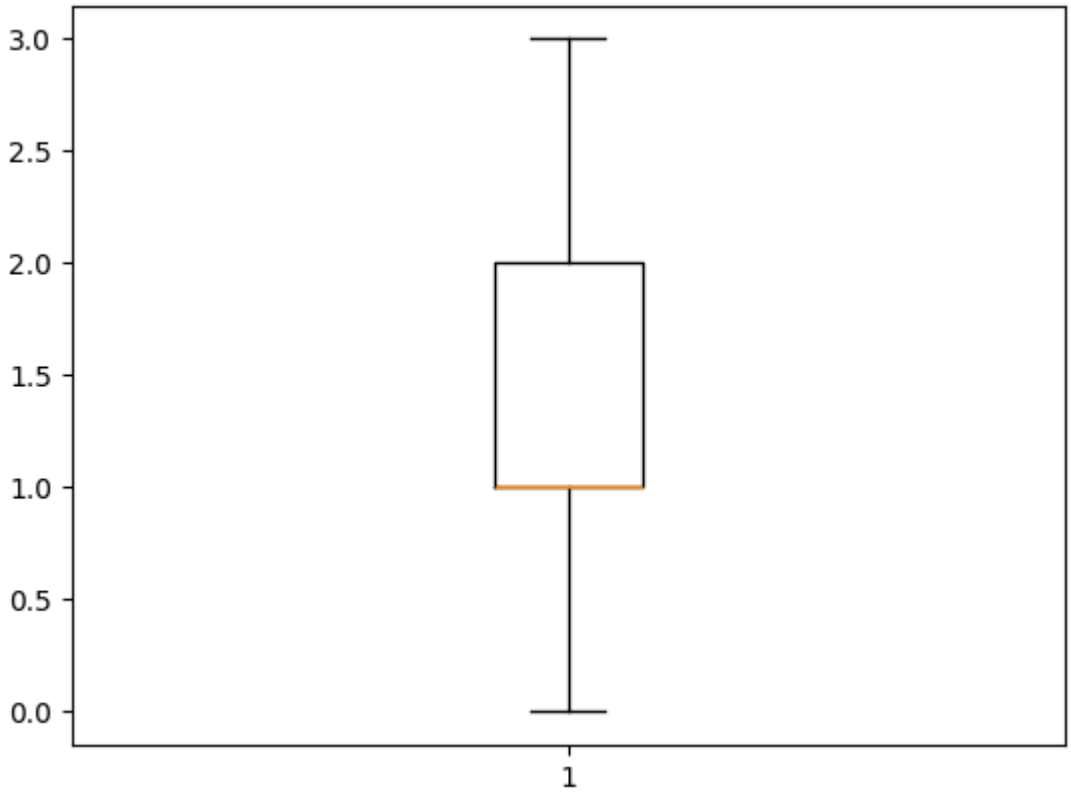

16
14
12
10
8
6
4
1


50
40
30
20
10
1


140
120
100
80
60
1


10
8
6
4
2
0
1


0.04
0.02
0.00
−0.02
−0.04
1



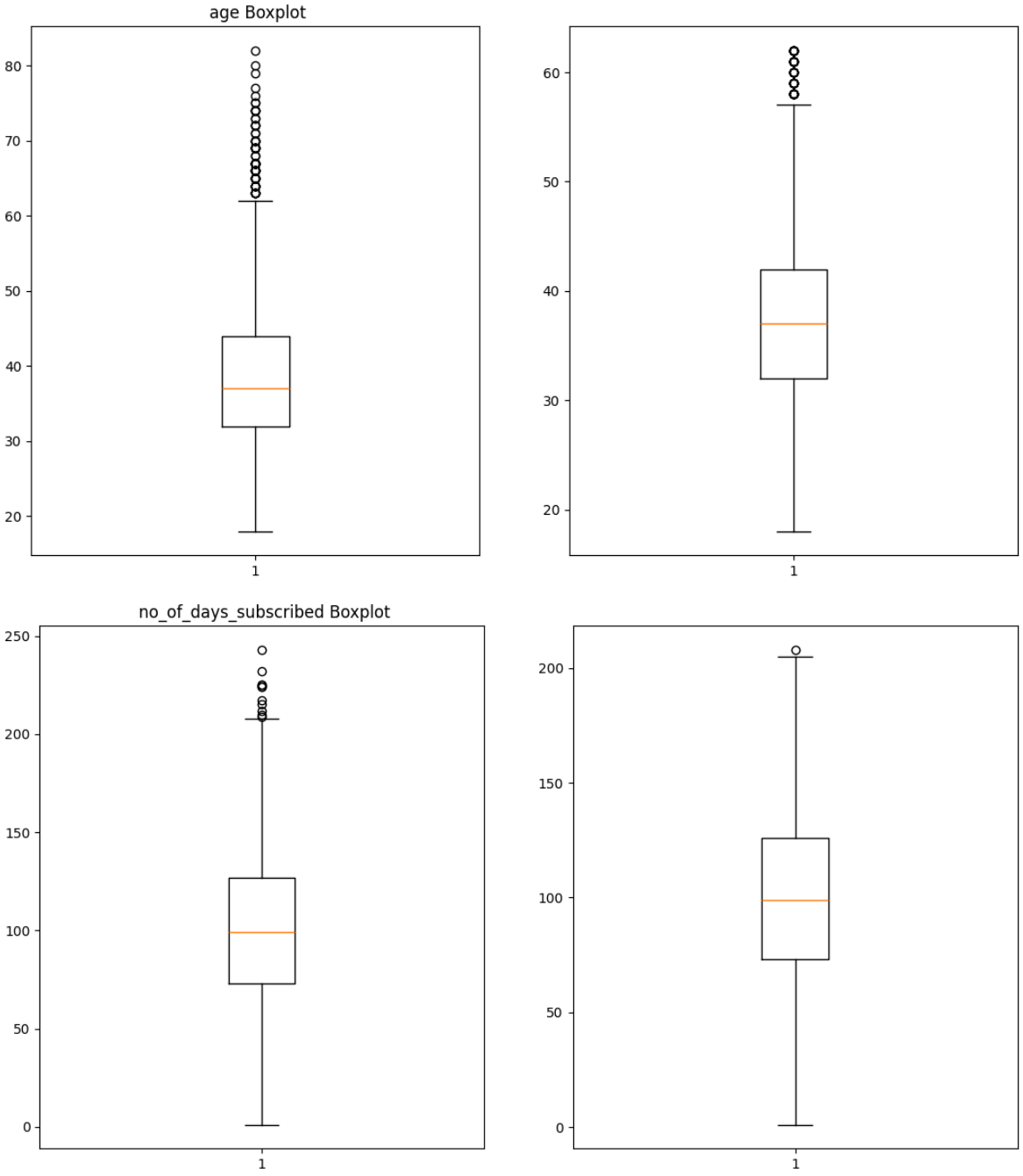
In [ ]:

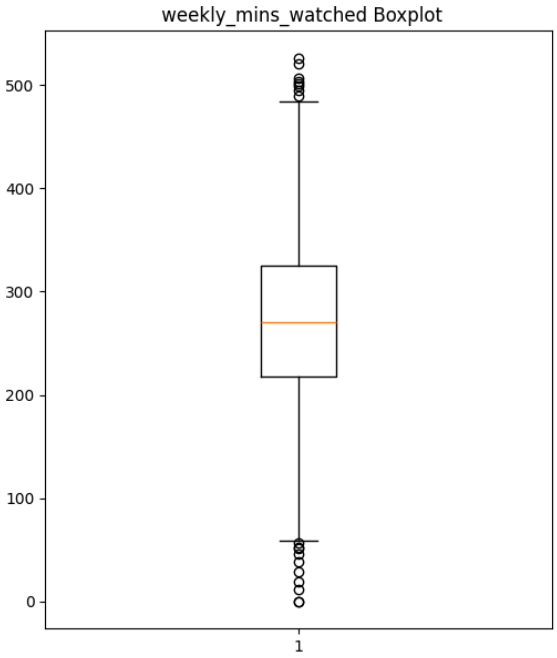
COMPARISON BOTH

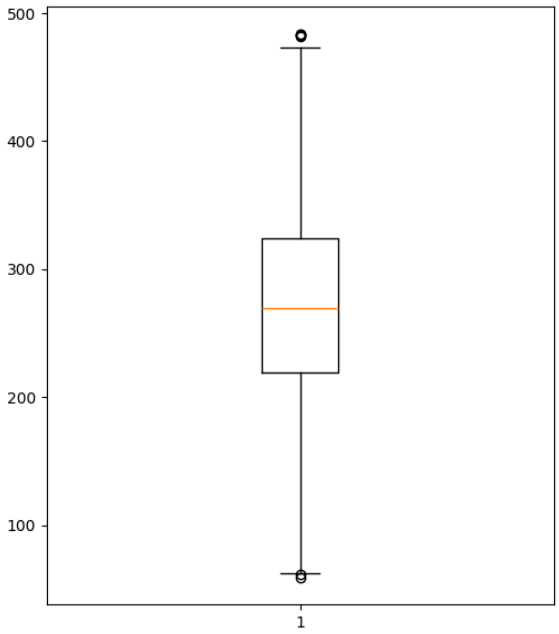
In [152…

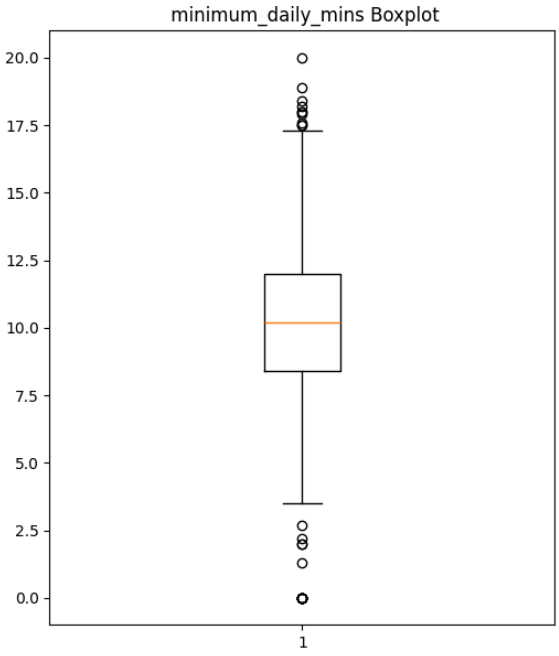
```
tele_df=pd.read_csv("telecom_churn_data.csv")
for i in num_clm_2[2:]:
    # with outliers
    plt.figure(figsize=(13,7))
    plt.subplot(1,2,1)
    plt.boxplot(tele_df[i])
    plt.title(f" {i} Boxplot")



    # fill the outliers
    plt.subplot(1,2,2)
    q1=np.quantile(tele_df[i],q=0.25)
    q2=np.quantile(tele_df[i],q=0.50)
    q3=np.quantile(tele_df[i],q=0.75)


    iqr=q3-q1

    lb=q1-1.5*iqr
    ub=q3+1.5*iqr

    con1=tele_df[i]<lb
    con2=tele_df[i]>ub
    con3=con1 | con2

    true=np.median(tele_df[i])
    false=tele_df[i]

    v=np.where(con3,true,false)
```

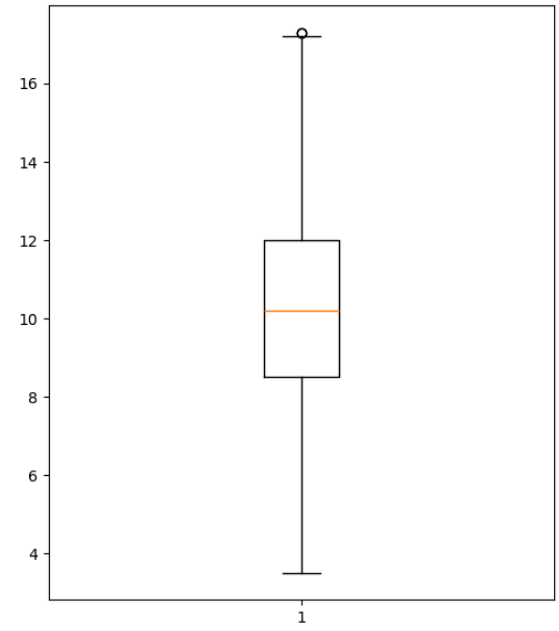
```
    tele_df[i]=v

    plt.boxplot(v)
    plt.show()
```

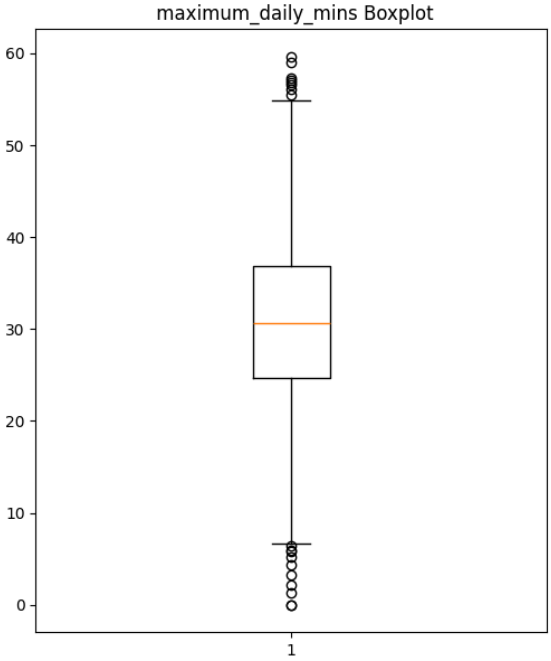


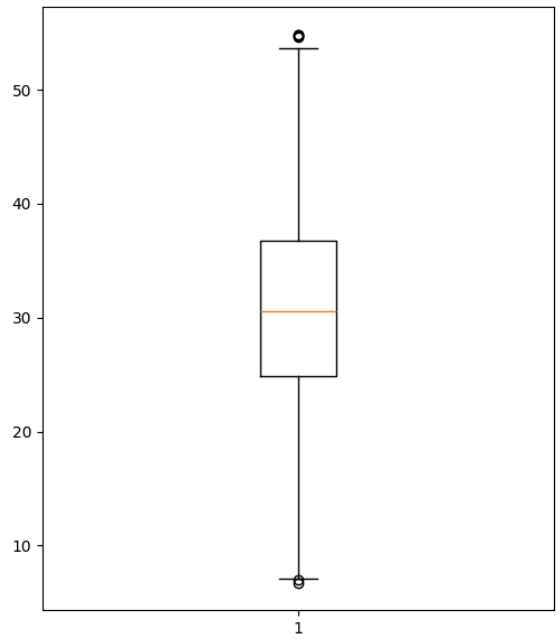

weekly_mins_watched Boxplot
500
400
300
200
100
0
1

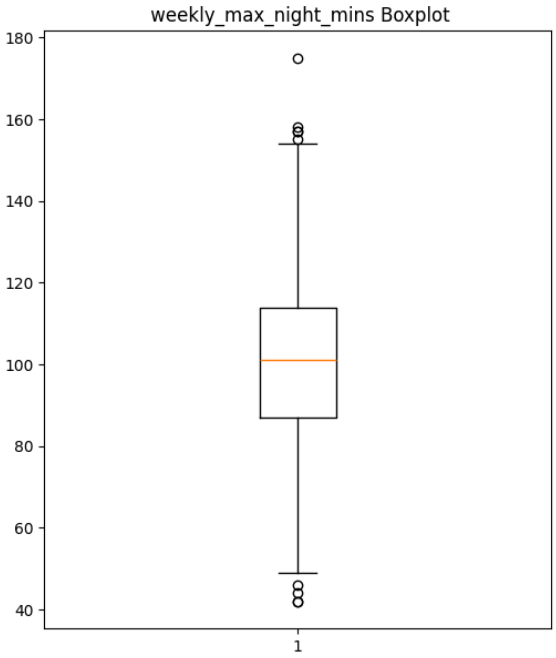

500
400
300
200
100
1

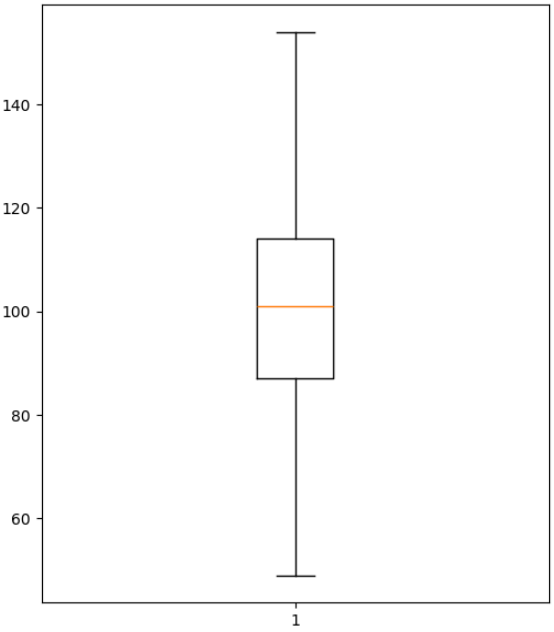

minimum_daily_mins Boxplot
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1

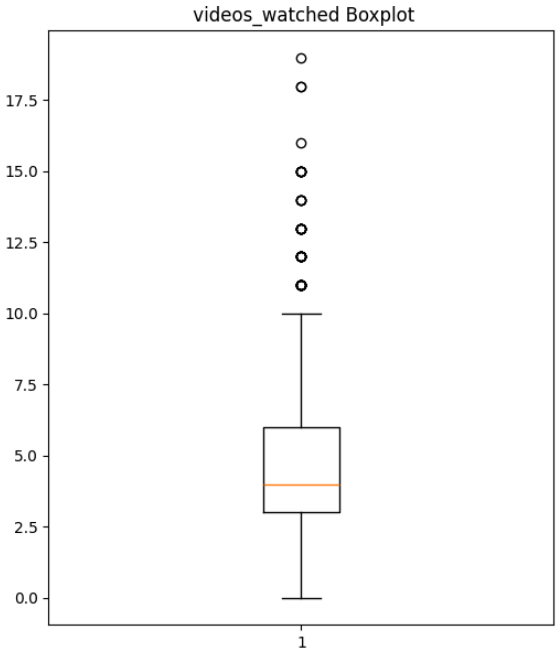

16
14
12
10
8
6
4
1

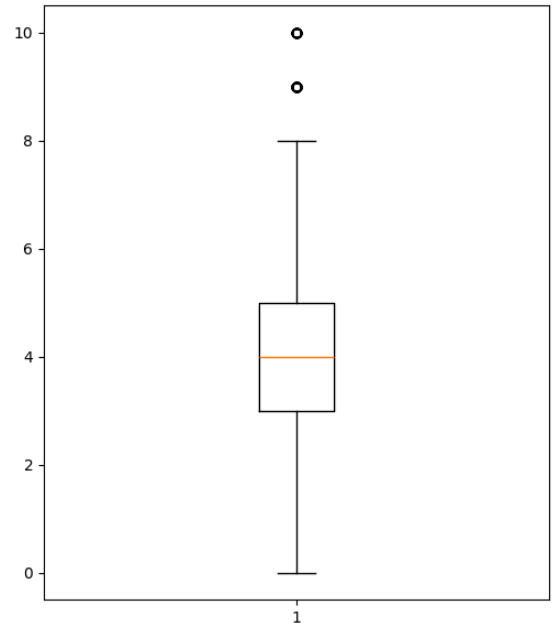

maximum_daily_mins Boxplot
60
50
40
30
20
10
0
1

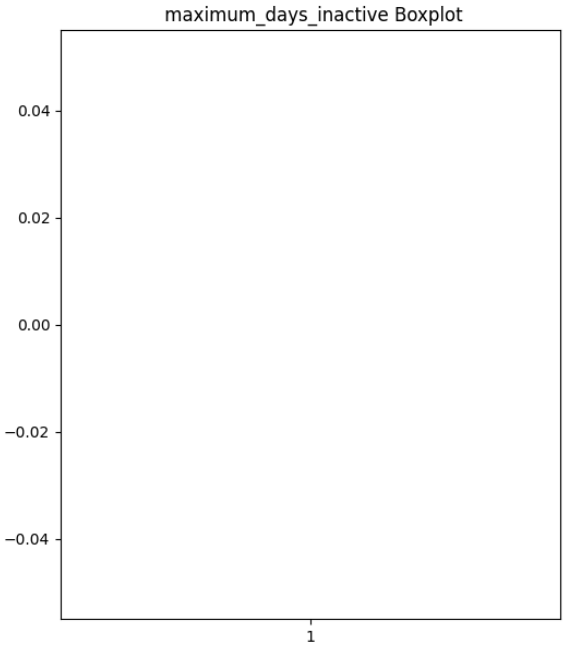

50
40
30
20
10
1

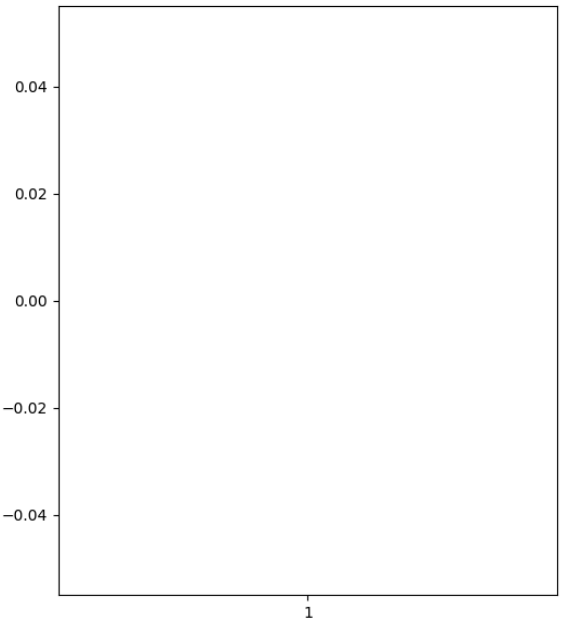

weekly_max_night_mins Boxplot
180
160
140
120
100
80
60
40
1

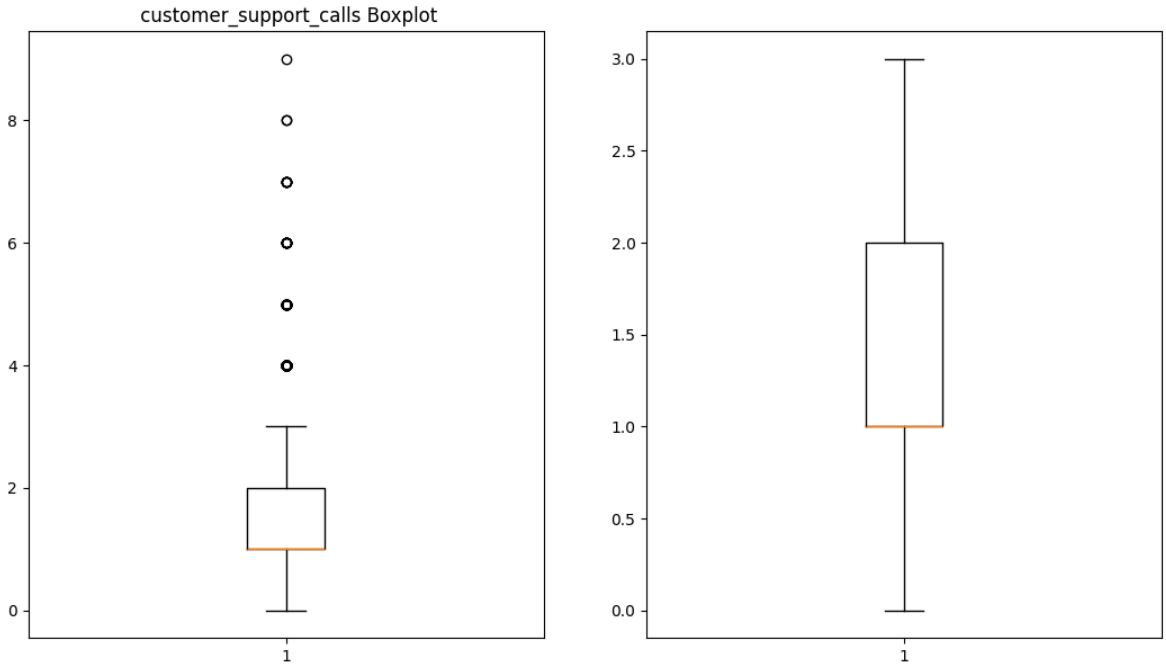

140
120
100
80
60
1


videos_watched Boxplot
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1


10
8
6
4
2
0
1


maximum_days_inactive Boxplot
0.04
0.02
0.00
−0.02
−0.04
1


0.04
0.02
0.00
−0.02
−0.04
1



In [ ]:

Biveriate and Multi Variate analysis

In [159…
```
cat_clm
```

Out[159…
```
Index(['phone_no', 'gender', 'multi_screen', 'mail_subscribed'], dtype='objec
t')
```

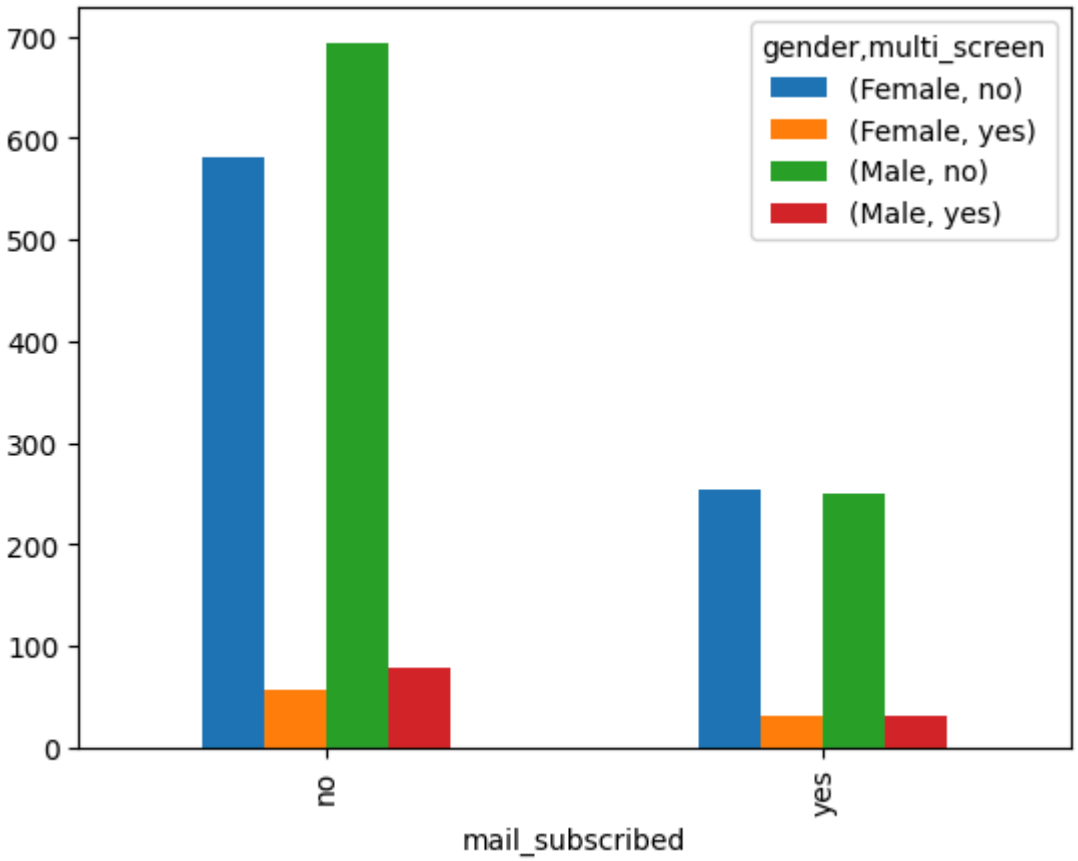
In [171…
```
index=tele_df['mail_subscribed']
con_1=tele_df['gender']
con_2=tele_df['multi_screen']
con_3=[con_1,con_2]
data=pd.crosstab(index,con_3)
data
```

Out[171…

gender	Female		Male	
multi_screen	no	yes	no	yes
mail_subscribed				
no	582	57	694	79
yes	253	31	249	31

In [175…
```
plt.figure(figsize=(10,5))
data.plot(kind="bar")
plt.show()
```

```
<Figure size 1000x500 with 0 Axes>
```



In [ ]:

Co relation between numerical column

In [183…
```
co_dataframe=tele_df.corr(numeric_only=True)
co_dataframe
```

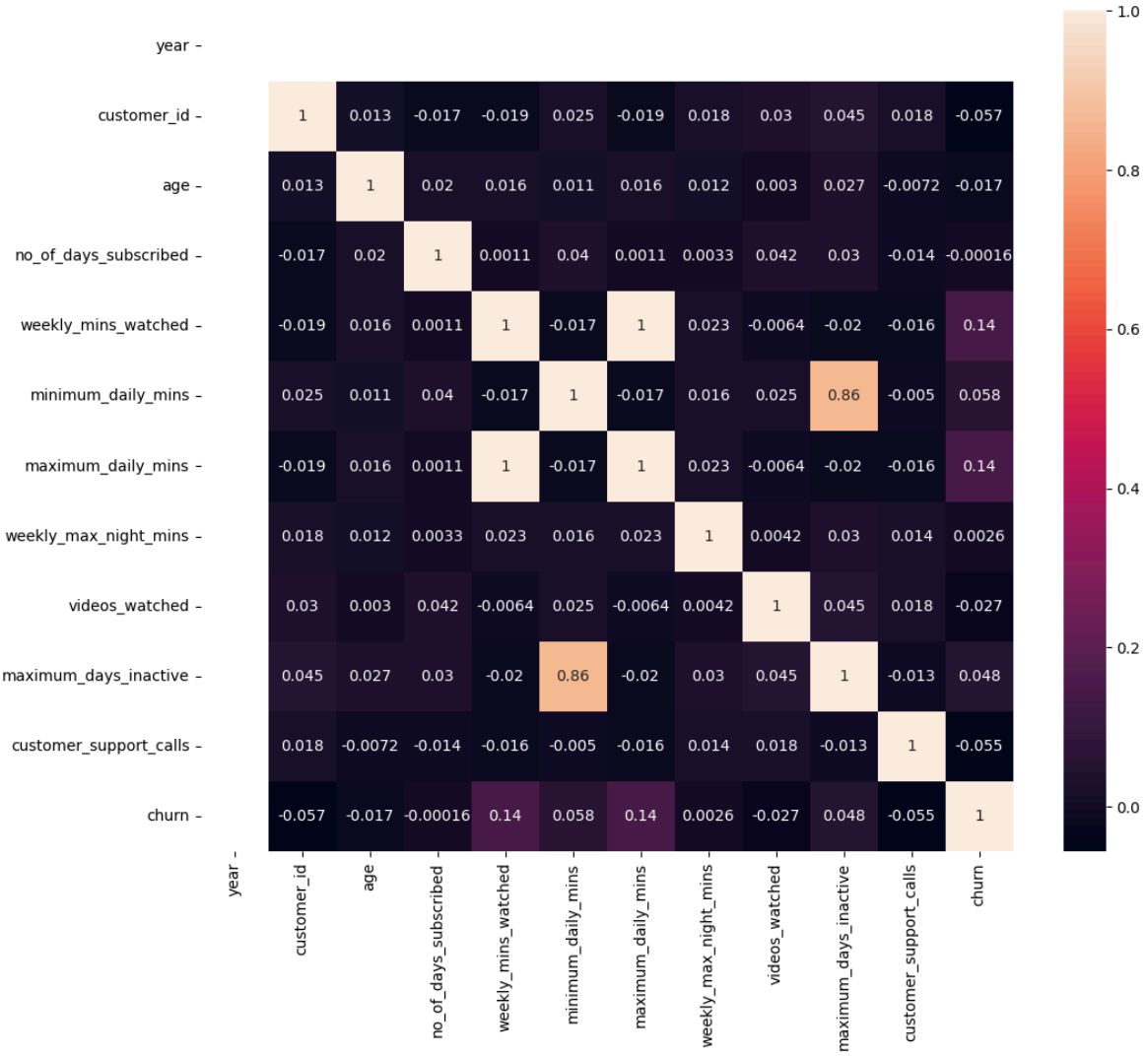
Out[183…

	year	customer_id	age	no_of_days_subscribed	weekly_mi
year	NaN	NaN	NaN	NaN	
customer_id	NaN	1.000000	0.012506	-0.017422	
age	NaN	0.012506	1.000000	0.020416	
no_of_days_subscribed	NaN	-0.017422	0.020416	1.000000	
weekly_mins_watched	NaN	-0.018533	0.016250	0.001131	
minimum_daily_mins	NaN	0.024945	0.011062	0.039527	
maximum_daily_mins	NaN	-0.018538	0.016265	0.001123	
weekly_max_night_mins	NaN	0.017802	0.012450	0.003276	
videos_watched	NaN	0.029513	0.002979	0.041836	
maximum_days_inactive	NaN	0.045145	0.026899	0.029951	
customer_support_calls	NaN	0.018197	-0.007193	-0.014171	
churn	NaN	-0.056777	-0.017213	-0.000157	

In [ ]:

HeatMaps

In [197…
```
plt.figure(figsize=(12,10))
sns.heatmap(co_dataframe,annot=True)
plt.show()
```



In [ ]:

Convert categorical to numerical data (Encoding)

LabelEncoder

In [209…
```
tele_df=pd.read_csv("telecom_churn_data.csv")
for i in cat_clm:
    from sklearn.preprocessing import LabelEncoder
    label=LabelEncoder()
    tele_df[i]=label.fit_transform(tele_df[i])
tele_df
```

Out[209…

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	1754	0	36	62	0
1	2015	100643	299	0	39	149	0
2	2015	100756	957	0	65	126	0
3	2015	101595	75	0	24	131	0
4	2015	101653	517	0	40	191	0
…	…	…	…	…	…	…	…
1995	2015	997132	1224	0	54	75	0
1996	2015	998086	1188	1	45	127	0
1997	2015	998474	553	2	53	94	0
1998	2015	998934	706	1	40	94	0
1999	2015	999961	1834	1	37	73	0

2000 rows × 16 columns

In [ ]:

Onehot Encoder

In [212…

```
pd.get_dummies(tele_df,dtype='int')
```

Out[212…

	year	customer_id	phone_no	gender	age	no_of_days_subscribed	multi_screen
0	2015	100198	1754	0	36	62	0
1	2015	100643	299	0	39	149	0
2	2015	100756	957	0	65	126	0
3	2015	101595	75	0	24	131	0
4	2015	101653	517	0	40	191	0
…	…	…	…	…	…	…	…
1995	2015	997132	1224	0	54	75	0
1996	2015	998086	1188	1	45	127	0
1997	2015	998474	553	2	53	94	0
1998	2015	998934	706	1	40	94	0
1999	2015	999961	1834	1	37	73	0

2000 rows × 16 columns

In [ ]: