In [102…

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import warnings
warnings.filterwarnings('ignore')
```

In [50]:

```
df = pd.read_csv('https://raw.githubusercontent.com/gaikwadshantanu12/adypsoe_
df
```

Out[50]:

	area_type	availability	location	size	society	total_sqft
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	NaN	1440
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	NaN	1200
...	...	...	...	...	...	...
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	NaN	3600
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	NaN	550

13320 rows × 9 columns

In [51]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13320 entries, 0 to 13319
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   area_type     13320 non-null  object 
 1   availability  13320 non-null  object 
 2   location      13319 non-null  object 
 3   size          13304 non-null  object 
 4   society       7818 non-null   object 
 5   total_sqft    13320 non-null  object 
 6   bath          13247 non-null  float64
 7   balcony       12711 non-null  float64
 8   price         13320 non-null  float64
dtypes: float64(3), object(6)
memory usage: 936.7+ KB
```

```
In [54]: df.isnull().sum()
```

```
Out[54]: area_type          0
         availability       0
         location           1
         size              16
         society         5502
         total_sqft         0
         bath              73
         balcony          609
         price              0
         dtype: int64
```

```
In [56]: # Clean column names: remove spaces, lowercase, replace special characters
         df.columns = df.columns.str.strip().str.lower().str.replace(" ", "_").str.repl
         print(df.columns)
```

```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')
```

```
In [58]: df['society'].fillna(value='Not Applicable', inplace = True)
```

```
In [60]: df['size'].value_counts()
```

```
Out[60]: size
         2 BHK         5199
         3 BHK         4310
         4 Bedroom      826
         4 BHK          591
         3 Bedroom      547
         1 BHK          538
         2 Bedroom      329
         5 Bedroom      297
         6 Bedroom      191
         1 Bedroom      105
         8 Bedroom       84
         7 Bedroom       83
         5 BHK           59
         9 Bedroom       46
         6 BHK           30
         7 BHK           17
         1 RK            13
         10 Bedroom      12
         9 BHK            8
         8 BHK            5
         11 BHK           2
         11 Bedroom       2
         10 BHK           2
         14 BHK           1
         13 BHK           1
         12 Bedroom       1
         27 BHK           1
         43 Bedroom       1
         16 BHK           1
         19 BHK           1
         18 Bedroom       1
         Name: count, dtype: int64
```

```
In [62]: df['size'].fillna(value = '2 BHK', inplace = True)
```

```
In [64]: df.dropna(subset=['location'], inplace=True)
```

```
In [66]: df['balcony'].value_counts()
```

```
Out[66]: balcony
         2.0    5112
         1.0    4897
         3.0    1672
         0.0    1029
         Name: count, dtype: int64
```

```
In [68]: df['balcony'].fillna(value='2.0', inplace=True)
```

```
In [70]: df['bath'].value_counts()
```

Out[70]:

```
bath
2.0     6908
3.0     3285
4.0     1226
1.0      788
5.0      524
6.0      273
7.0      102
8.0       64
9.0       43
10.0      13
12.0       7
13.0       3
11.0       3
16.0       2
27.0       1
40.0       1
15.0       1
14.0       1
18.0       1
Name: count, dtype: int64
```

In [72]:

```
df['bath'].fillna(value='2.0', inplace=True)
```

In [74]:

```
df.isnull().sum()
```

Out[74]:

```
area_type       0
availability    0
location        0
size            0
society         0
total_sqft      0
bath            0
balcony         0
price           0
dtype: int64
```

In [76]:

```
# Load additional dataset
#neighborhood = pd.read_csv("Neighborhood_Info.csv")

# Merge on common column (e.g., neighborhood)
#df = pd.merge(df, neighborhood, on="neighborhood", how="left")
```

In [84]:

```
def convert_sqft_to_num(x):
    tokens = x.split('-')
    if len(tokens) == 2:
        try:
            return (float(tokens[0])+float(tokens[1]))/2
        except ValueError:
            return None
    try:
        return float(x)
    except ValueError:
        return None
```

In [86]:

```
df.total_sqft = df.total_sqft.apply(convert_sqft_to_num)
df
```

Out[86]:

	area_type	availability	location	size	society	total_sqft
0	Super built-up Area	19-Dec	Electronic City Phase II	2 BHK	Coomee	1056.0
1	Plot Area	Ready To Move	Chikka Tirupathi	4 Bedroom	Theanmp	2600.0
2	Built-up Area	Ready To Move	Uttarahalli	3 BHK	Not Applicable	1440.0
3	Super built-up Area	Ready To Move	Lingadheeranahalli	3 BHK	Soiewre	1521.0
4	Super built-up Area	Ready To Move	Kothanur	2 BHK	Not Applicable	1200.0
...	...	...	...	...	...	...
13315	Built-up Area	Ready To Move	Whitefield	5 Bedroom	ArsiaEx	3453.0
13316	Super built-up Area	Ready To Move	Richards Town	4 BHK	Not Applicable	3600.0
13317	Built-up Area	Ready To Move	Raja Rajeshwari Nagar	2 BHK	Mahla T	1141.0
13318	Super built-up Area	18-Jun	Padmanabhanagar	4 BHK	SollyCl	4689.0
13319	Super built-up Area	Ready To Move	Doddathoguru	1 BHK	Not Applicable	550.0

13319 rows × 9 columns

In [94]:

```
df = df[df.total_sqft.notnull()]
```

In [88]:

```
df.columns
```

Out[88]:

```
Index(['area_type', 'availability', 'location', 'size', 'society',
       'total_sqft', 'bath', 'balcony', 'price'],
      dtype='object')
```

In [110…

```
df.location = df.location.apply(lambda x: x.strip())
location_stats = df['location'].value_counts(ascending=False)
```

```
location_stats
```

Out[110…

```
location
Whitefield                        447
Sarjapur  Road                    343
Electronic City                   303
Kanakpura Road                    266
Thanisandra                       227
                                 ...
Maruthi HBCS Layout                 1
t.c palya                           1
Manganahalli                        1
Housing Board Layout Vijay Nagar    1
Abshot Layout                       1
Name: count, Length: 1210, dtype: int64
```

In [116…

```
df['bhk'] = df['size'].apply(lambda x: int(x.split(' ')[0]))
```

In [120…

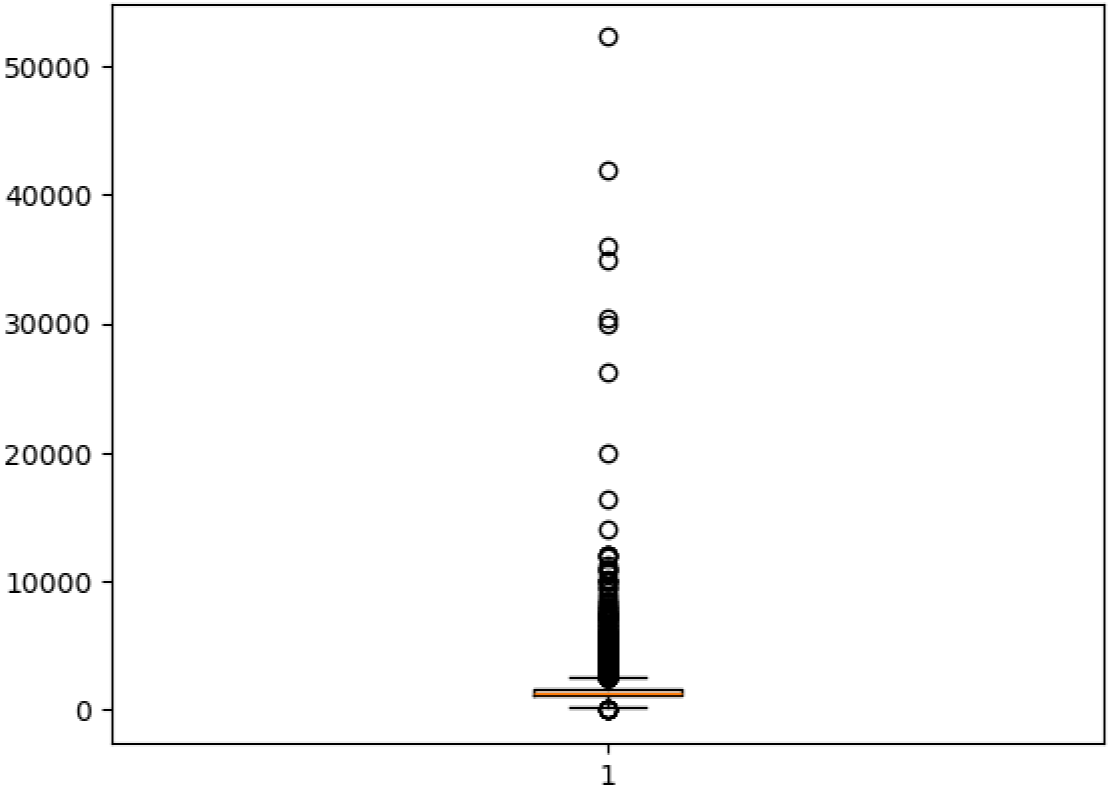
```
df = df[~(df.total_sqft/df.bhk<300)]
df.shape
```

Out[120…

```
(10833, 10)
```

In [96]:

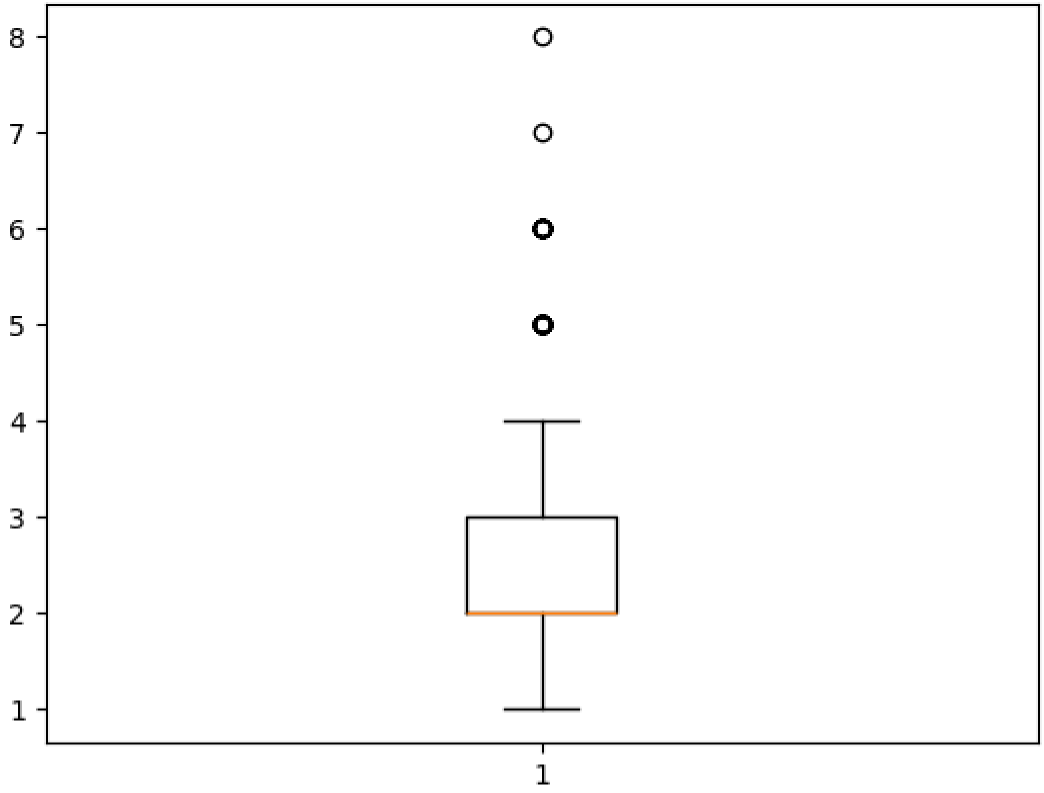
```
plt.boxplot(df['total_sqft'])
plt.show()
```



In [132…

```
Q1 = np.percentile(df['total_sqft'], 25.)
Q3 = np.percentile(df['total_sqft'], 75.)
```

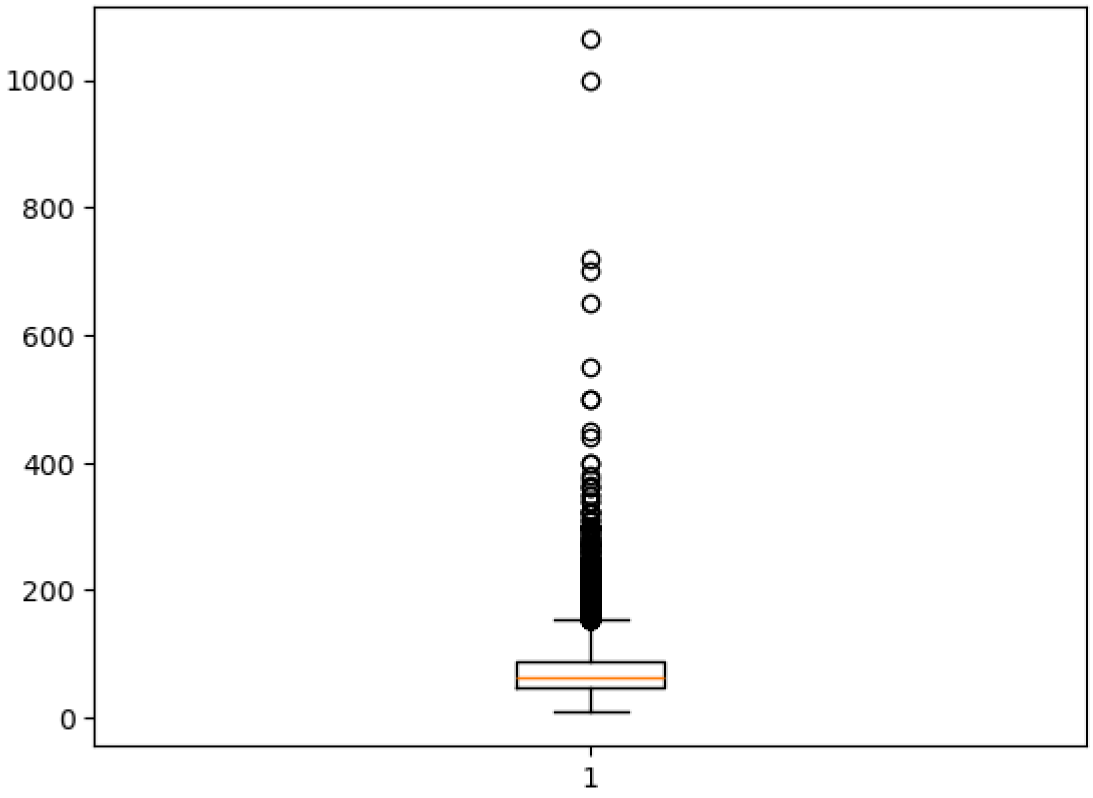
```
IQR = Q3-Q1
ll = Q1 - (1.5*IQR)
ul = Q3 + (1.5*IQR)
upper_outliers = df[df['total_sqft'] > ul].index.tolist()
lower_outliers = df[df['total_sqft'] < ll].index.tolist()
bad_indices = list(set(upper_outliers + lower_outliers))
drop = True
if drop:
    df.drop(bad_indices, inplace = True, errors = 'ignore')
df['bath'] = (
    df['bath']
    .astype(str)                                 # convert to string
    .str.replace('[^0-9.]', '', regex=True) # remove all characters except dig
    .replace('', np.nan)                         # replace empty strings with NaN
    .astype(float)                               # finally convert to float
)

plt.boxplot(df['bath'])
plt.show()
```



In [138…

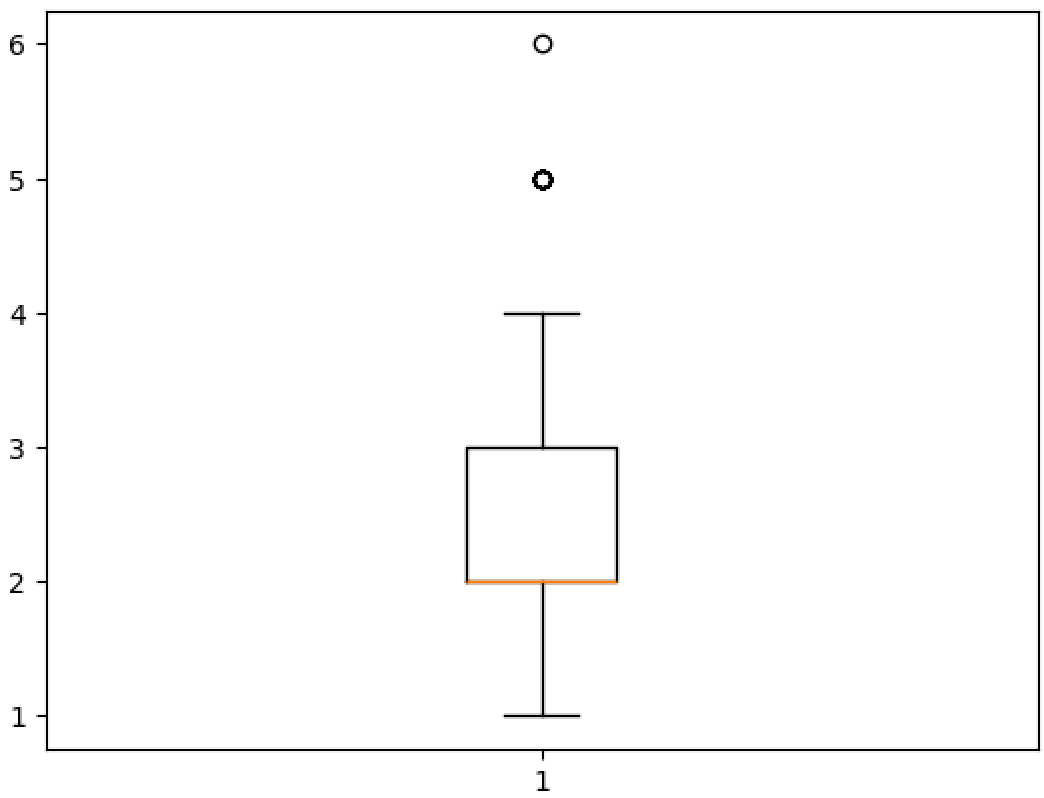
```
Q1 = np.percentile(df['bath'], 25.) # 25th percentile of the data of the given
Q3 = np.percentile(df['bath'], 75.) # 75th percentile of the data of the given
IQR = Q3-Q1 #Interquartile Range
ll = Q1 - (1.5*IQR)
ul = Q3 + (1.5*IQR)
upper_outliers = df[df['bath'] > ul].index.tolist()
lower_outliers = df[df['bath'] < ll].index.tolist()
bad_indices = list(set(upper_outliers + lower_outliers))
```

```
drop = True
if drop:
    df.drop(bad_indices, inplace = True, errors = 'ignore')
plt.boxplot(df['price'])
plt.show()
```

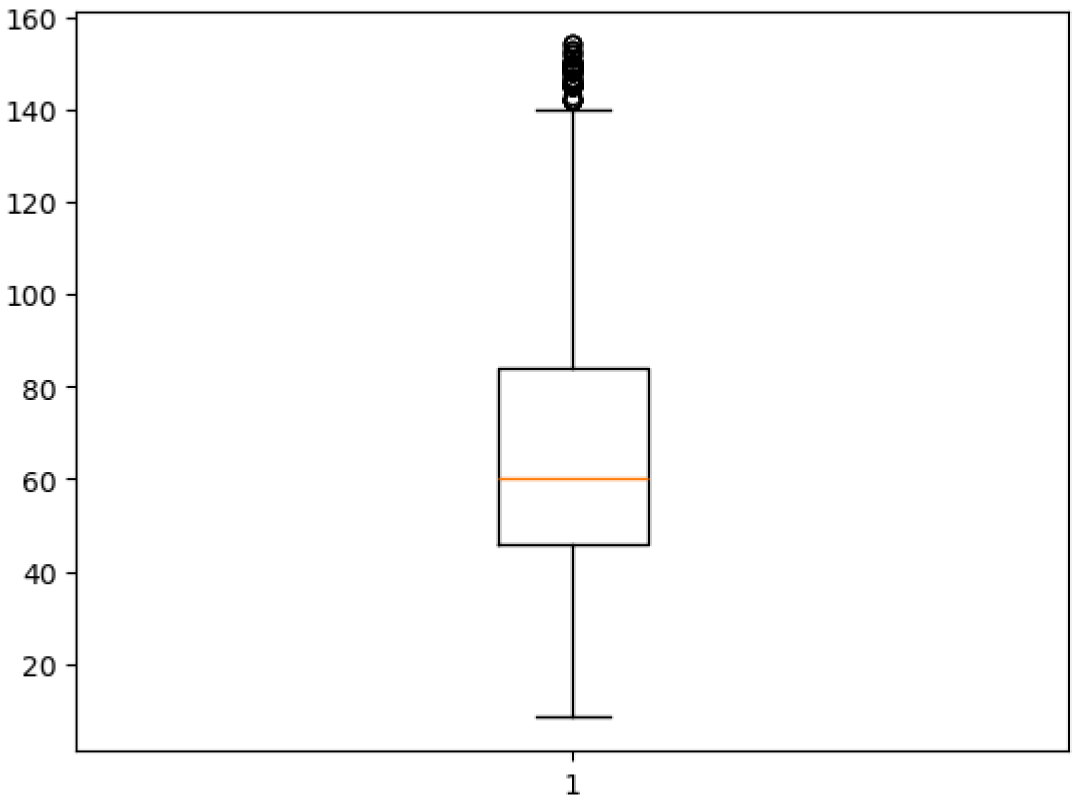


In [142…

```
Q1 = np.percentile(df['price'], 25.) # 25th percentile of the data of the give
Q3 = np.percentile(df['price'], 75.) # 75th percentile of the data of the give
IQR = Q3-Q1 #Interquartile Range
ll = Q1 - (1.5*IQR)
ul = Q3 + (1.5*IQR)

upper_outliers = df[df['price'] > ul].index.tolist()
lower_outliers = df[df['price'] < ll].index.tolist()
bad_indices = list(set(upper_outliers + lower_outliers))
drop = True
if drop:
    df.drop(bad_indices, inplace = True, errors = 'ignore')

plt.boxplot(df['bhk'])
plt.show()
```



```
In [148…  plt.boxplot(df['price'])
          plt.show()
```



In [ ]: