In [2]:

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

1 DATA CLEANING_Dataset_1_Airbnb and Visualization

In [3]:

```
df = pd.read_csv('Airbnb_Dataset.csv')
```

In [4]:

```
dh = pd.read_csv('HR_Dataset.csv')
```

In [5]:

```
df.shape
```

Out[5]:

```
(249, 16)
```

In [8]:

```
df.head(2)
```

Out[8]:

ırhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews
ensington	40.64749	-73.97237	Private room	149	1	
Midtown	40.75362	-73.98377	Entire home/apt	225	1	

In [9]:

```
df.tail(2)
```

Out[9]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood
247	62427	Great East Village Apartment Rental	303882	Brie	Manhattan	East Villa
248	62430	BROWNSTONE SUNDRENCHED BEAUTY	197755	Sheila	Brooklyn	Bushw

In [14]:
```
df.columns
```

Out[14]:
```
Index(['id', 'name', 'host_id', 'host_name', 'neighbourhood_group',
       'neighbourhood', 'latitude', 'longitude', 'room_type', 'price',
       'minimum_nights', 'number_of_reviews', 'last_review',
       'reviews_per_month', 'calculated_host_listings_count',
       'availability_365'],
      dtype='object')
```

In [ ]:

In [ ]:

In [10]:
```
df.index
```

Out[10]:
```
RangeIndex(start=0, stop=249, step=1)
```

In [11]:
```
# check for dimension
df.ndim
```

Out[11]:
```
2
```

In [12]:
```
# check for basic  information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 249 entries, 0 to 248
Data columns (total 16 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              249 non-null    int64  
 1   name                            249 non-null    object 
 2   host_id                         249 non-null    int64  
 3   host_name                       249 non-null    object 
 4   neighbourhood_group             249 non-null    object 
 5   neighbourhood                   249 non-null    object 
 6   latitude                        249 non-null    float64
 7   longitude                       249 non-null    float64
 8   room_type                       249 non-null    object 
 9   price                           249 non-null    int64  
 10  minimum_nights                  249 non-null    int64  
 11  number_of_reviews               249 non-null    int64  
 12  last_review                     242 non-null    object 
 13  reviews_per_month               242 non-null    float64
 14  calculated_host_listings_count  249 non-null    int64  
 15  availability_365                249 non-null    int64  
dtypes: float64(3), int64(7), object(6)
memory usage: 31.3+ KB
```

In [15]:
```
# delete unnecessary coumns
df.drop(['reviews_per_month','calculated_host_listings_count'], axis=1, inp
```

In [16]:

```
# delete the null amount values
df.dropna(inplace =True)
```

In [17]:

```
df.shape
```

Out[17]:

```
(242, 14)
```

In [18]:

```
# check all columns name

for i in df.columns:
    print(i)
```

```
id
name
host_id
host_name
neighbourhood_group
neighbourhood
latitude
longitude
room_type
price
minimum_nights
number_of_reviews
last_review
availability_365
```

In [ ]:

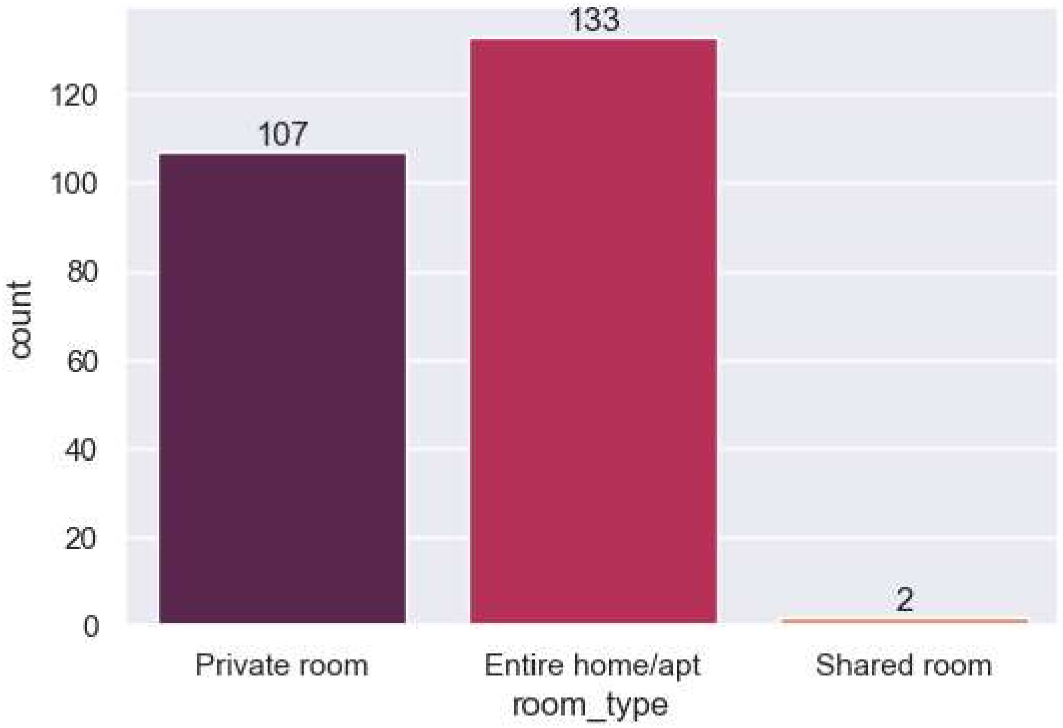
```
# change data  tyype if required
# change the amount dtype fro folat to int

df['price']=df['price'].astype(int)
```

In [ ]:

```
# to check statistics of data
df.describe()
```

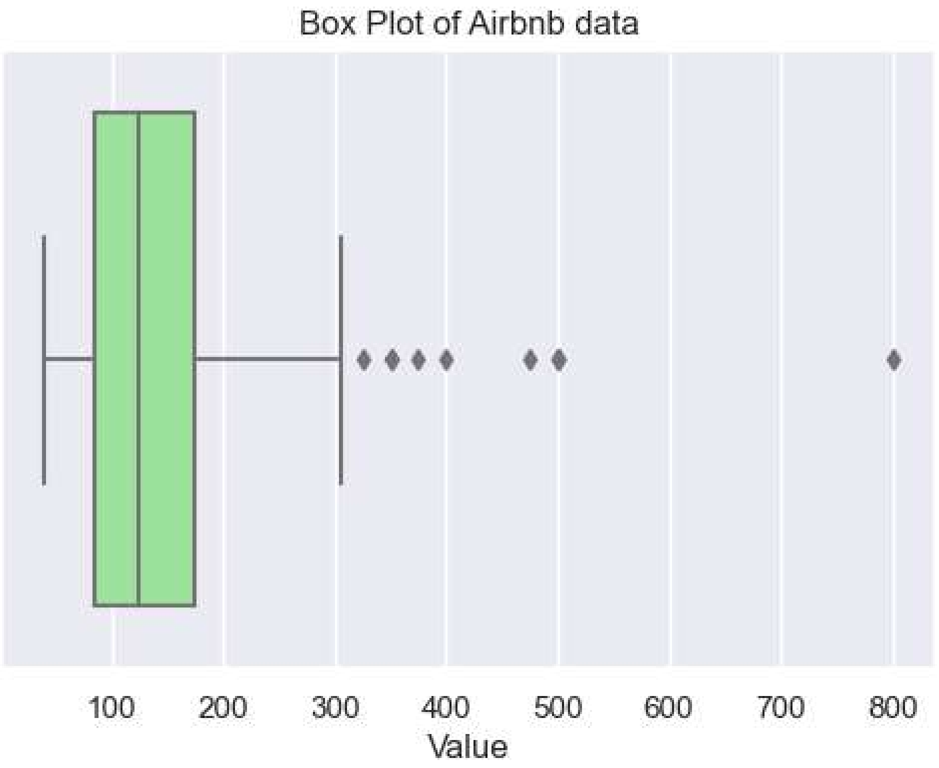
In [19]:

```
sns.set(rc={'figure.figsize':(6,4)})
plot = sns.countplot(x='room_type', data=df, palette='rocket')
for count in plot.containers:
    plot.bar_label(count)
plt.show()
```



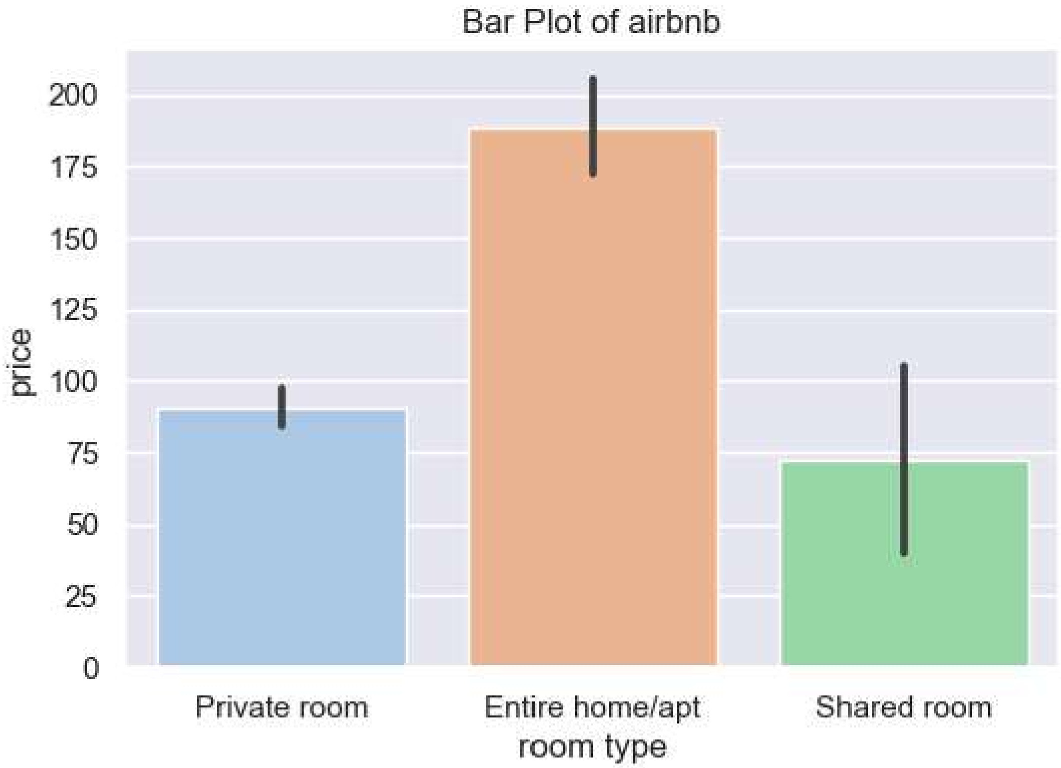
In [24]:

```
sns.boxplot(x=df['price'], color='lightgreen')
plt.title('Box Plot of Airbnb data')
plt.xlabel('Value')
plt.show()
```



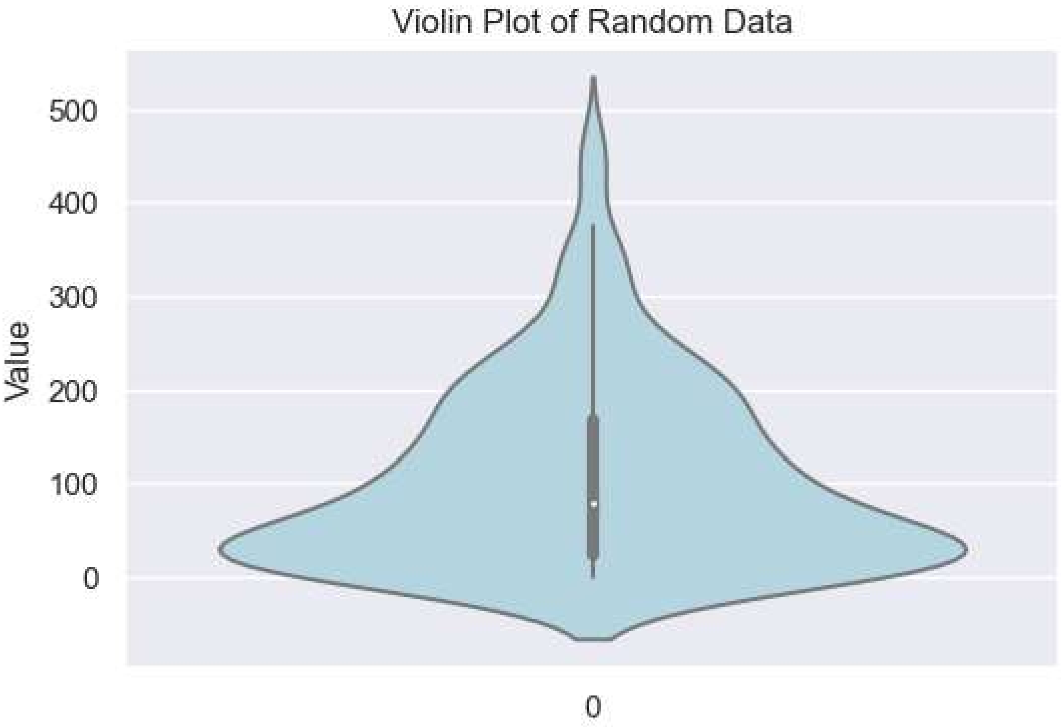
In [27]:

```
sns.barplot(x=df['room_type'], y=df['price'], palette='pastel')
plt.title('Bar Plot of airbnb')
plt.xlabel('room type')
plt.ylabel('price')
plt.show()
```



In [29]:

```
sns.violinplot(data=df['number_of_reviews'], color='lightblue')
plt.title('Violin Plot of Random Data')
plt.ylabel('Value')
plt.show()
```



2 Data cleaning Data Set 2 HR Dataset a nd Visualization

In [31]:

```
dh.head(2)
```

Out[31]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID
0	Adinolfi, Wilson K	10026	0	0	1	1
1	Ait Sidi, Karthikeyan	10084	1	1	1	5

2 rows × 36 columns

In [32]:

```
dh.tail()
```

Out[32]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID
306	Woodson, Jason	10135	0	0	1	1
307	Ybarra, Catherine	10301	0	0	0	5
308	Zamora, Jennifer	10010	0	0	0	1
309	Zhou, Julia	10043	0	0	0	1
310	Zima, Colleen	10271	0	4	0	1

5 rows × 36 columns

In [ ]:

```
dh.shape
dh.column
dh.index
```

In [34]:

```
df.ndim
```

Out[34]:

```
2
```

In [36]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 242 entries, 0 to 248
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   id                   242 non-null    int64  
 1   name                 242 non-null    object 
 2   host_id              242 non-null    int64  
 3   host_name            242 non-null    object 
 4   neighbourhood_group  242 non-null    object 
 5   neighbourhood        242 non-null    object 
 6   latitude             242 non-null    float64
 7   longitude            242 non-null    float64
 8   room_type            242 non-null    object 
 9   price                242 non-null    int64  
 10  minimum_nights       242 non-null    int64  
 11  number_of_reviews    242 non-null    int64  
 12  last_review          242 non-null    object 
 13  availability_365     242 non-null    int64  
dtypes: float64(2), int64(6), object(6)
memory usage: 36.5+ KB
```

In [37]:
```
df.describe()
```

Out[37]:

	id	host_id	latitude	longitude	price	minimum_nights	nu
count	242.000000	2.420000e+02	242.000000	242.000000	242.000000	242.000000	
mean	31667.024793	1.518822e+05	40.729170	-73.964527	144.272727	8.479339	
std	17953.882898	4.062905e+05	0.048392	0.029916	92.279028	20.365172	
min	2539.000000	2.787000e+03	40.631880	-74.080880	40.000000	1.000000	
25%	16430.250000	5.136225e+04	40.688108	-73.985222	85.000000	2.000000	
50%	28651.500000	1.023750e+05	40.720280	-73.965835	125.000000	3.000000	
75%	46864.000000	1.935678e+05	40.759568	-73.948373	175.000000	5.000000	
max	62430.000000	6.197784e+06	40.864820	-73.765970	800.000000	200.000000	

In [38]:
```
dh.dropna(inplace =True)
```

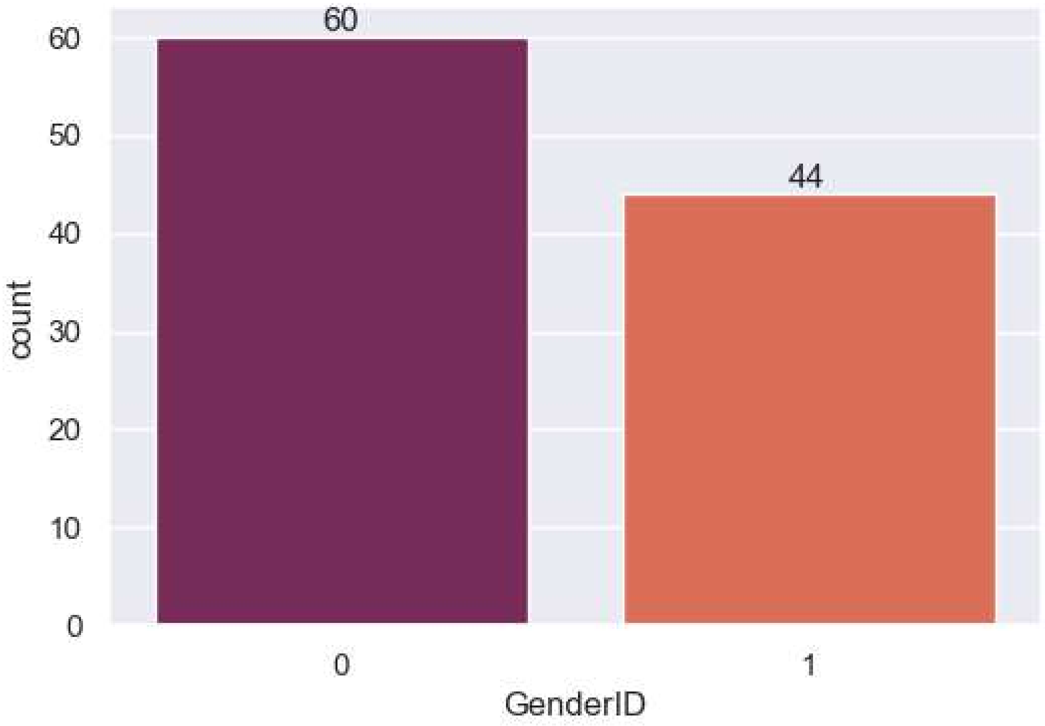
In [39]:

```
for i in dh.columns:
    print(i)
```

```
Employee_Name
EmpID
MarriedID
MaritalStatusID
GenderID
EmpStatusID
DeptID
PerfScoreID
FromDiversityJobFairID
Salary
Termd
PositionID
Position
State
Zip
DOB
Sex
MaritalDesc
CitizenDesc
HispanicLatino
RaceDesc
DateofHire
DateofTermination
TermReason
EmploymentStatus
Department
ManagerName
ManagerID
RecruitmentSource
PerformanceScore
EngagementSurvey
EmpSatisfaction
SpecialProjectsCount
LastPerformanceReview_Date
DaysLateLast30
Absences
```

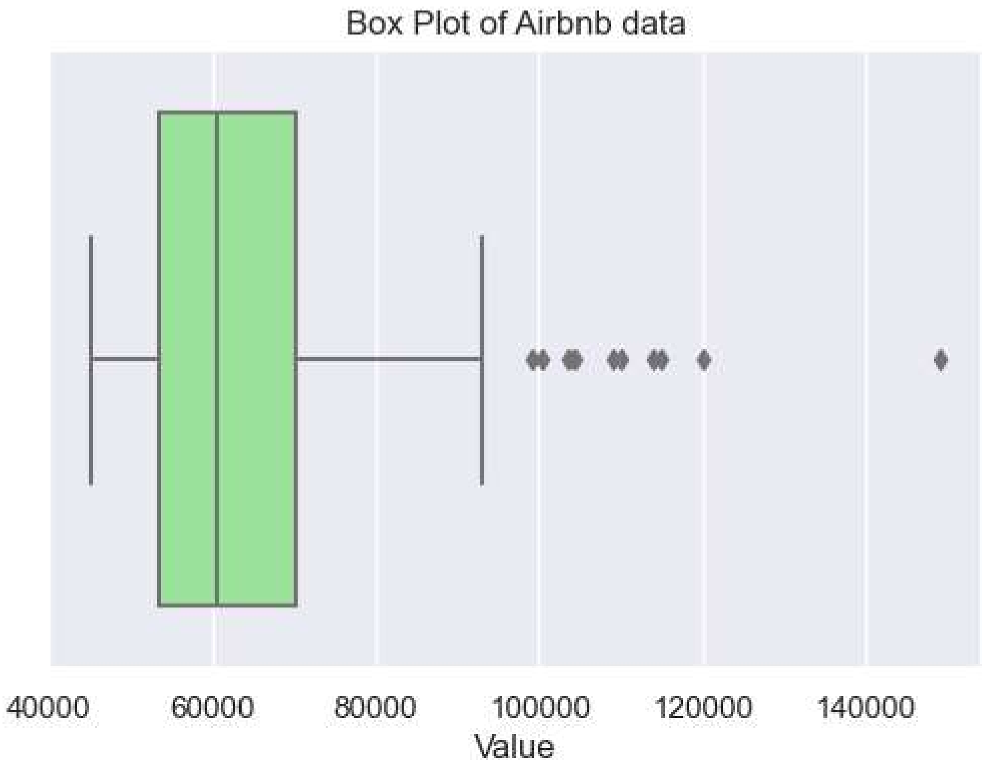
In [40]:

```
sns.set(rc={'figure.figsize':(6,4)})
plot = sns.countplot(x='GenderID', data=dh, palette='rocket')
for count in plot.containers:
    plot.bar_label(count)
plt.show()
```

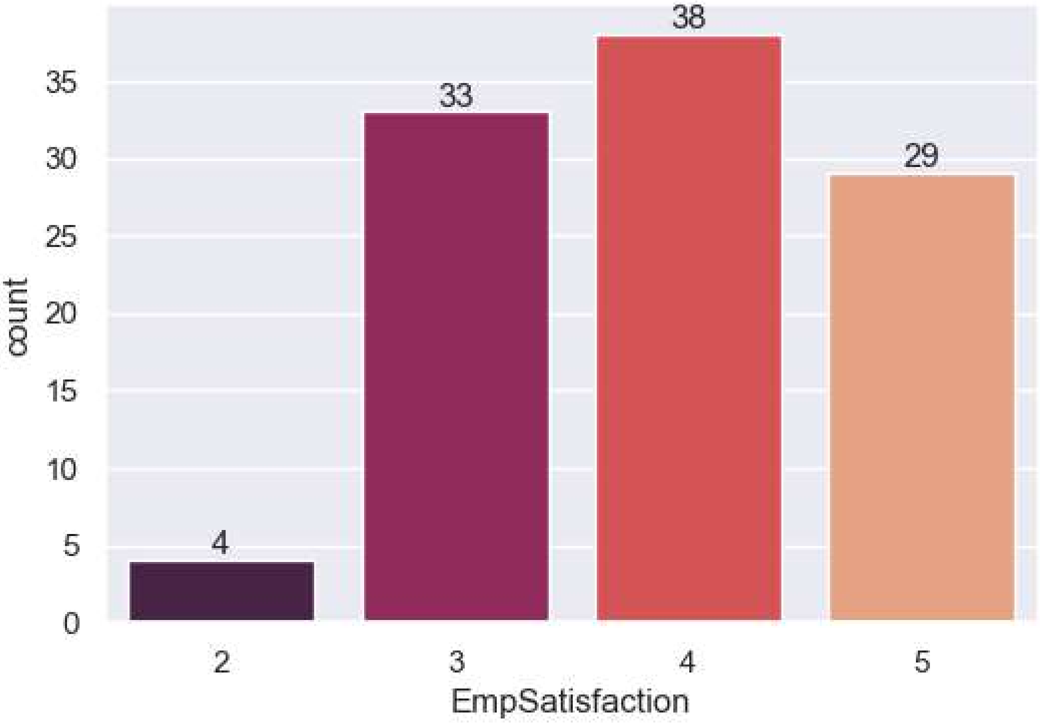


In [41]:

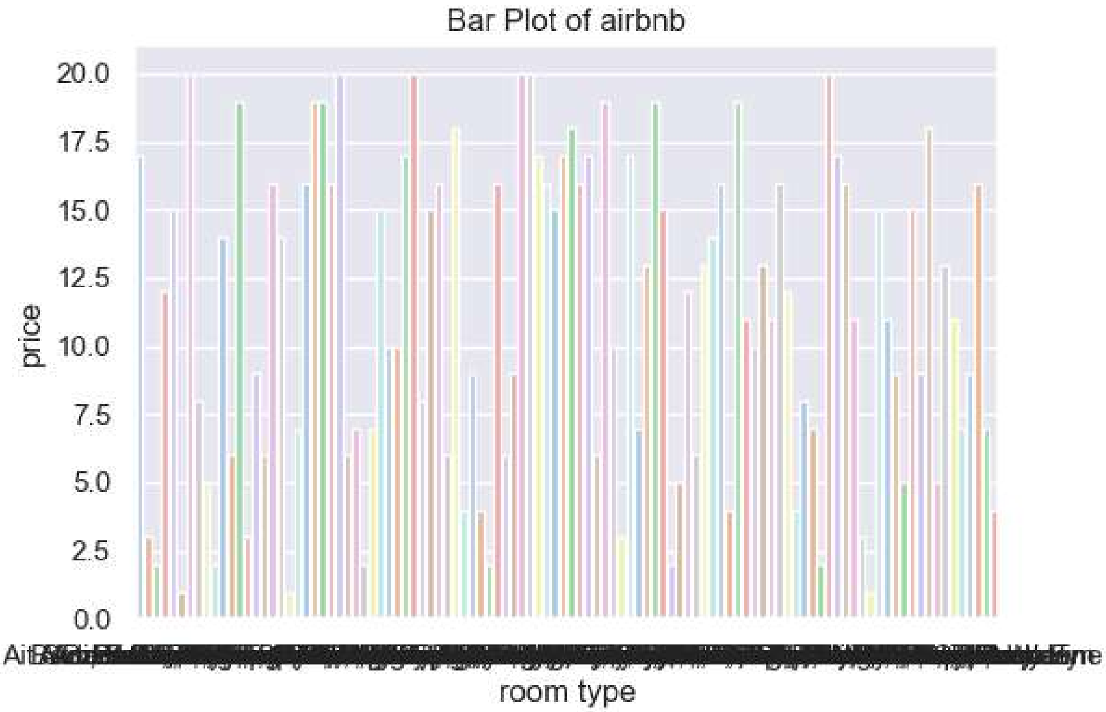
```
sns.boxplot(x=dh['Salary'], color='lightgreen')
plt.title('Box Plot of Airbnb data')
plt.xlabel('Value')
plt.show()
```



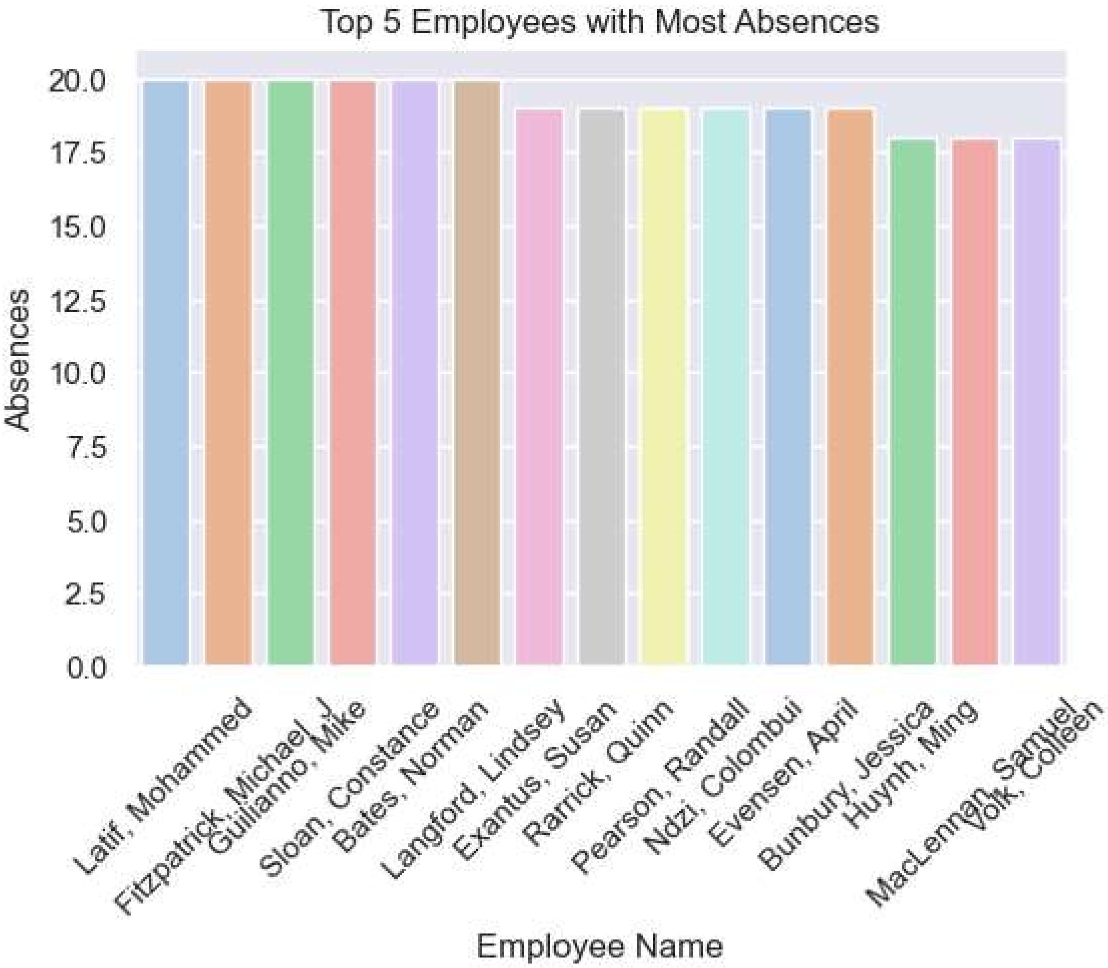
In [42]:
```
sns.set(rc={'figure.figsize':(6,4)})
plot = sns.countplot(x='EmpSatisfaction', data=dh, palette='rocket')
for count in plot.containers:
    plot.bar_label(count)
plt.show()
```



In [45]:
```
sns.barplot(x=dh['Employee_Name'], y=dh['Absences'], palette='pastel')
plt.title('Bar Plot of airbnb')
plt.xlabel('room type')
plt.ylabel('price')
plt.show()
```



In [49]:
```
# Sort the DataFrame by 'Absences' column in descending order
sorted_df = dh.sort_values(by='Absences', ascending=False)

# Select the top 5 rows
top_5 = sorted_df.head(15)

# Plot bar plot for the top 5 rows
sns.barplot(x=top_5['Employee_Name'], y=top_5['Absences'], palette='pastel'
plt.title('Top 5 Employees with Most Absences')
plt.xlabel('Employee Name')
plt.ylabel('Absences')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.show()
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: