In [28]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
import missingno as msno
import datetime as dt
```

Dataset 1 - Airbnb Dataset

In [9]:
```
airbnb =pd.read_csv("Airbnb Dataset 19.csv")
```

In [38]:
```
airbnb.head()
```

Out[38]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price
0	2539	Clean & quiet apt home by the park	2787	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149
1	2595	Skylit Midtown Castle	2845	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225
2	3647	THE VILLAGE OF HARLEM....NEW YORK !	4632	Elisabeth	Manhattan	Harlem	40.80902	-73.94190	Private room	150
3	3831	Cozy Entire Floor of Brownstone	4869	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89
4	5022	Entire Apt: Spacious Studio/Loft by central park	7192	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80

In [39]:
```
#column host_id is not needed, its dropped
airbnb.drop('host_id', axis = 1, inplace = True)
```

In [40]:
```
#data of Name column needs to be proper function
airbnb["name"] = airbnb["name"].str.upper().str.title()
```

In [41]:
```
#data remove which contain NAN in 'last-review'column
airbnb = airbnb.dropna(axis=0, subset = ['last_review'])
```

In [42]:
```
airbnb.head()
```

Out[42]:

	id	name	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_n
0	2539	Clean & Quiet Apt Home By The Park	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	
1	2595	Skylit Midtown Castle	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	
3	3831	Cozy Entire Floor Of Brownstone	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	
4	5022	Entire Apt: Spacious Studio/Loft By Central Park	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	
5	5099	Large Cozy 1 Br Apartment In Midtown East	Chris	Manhattan	Murray Hill	40.74767	-73.97500	Entire home/apt	200	

In [8]:
```
airbnb.dtypes
```

Out[8]:
```
id                                int64
name                             object
host_name                        object
neighbourhood_group              object
neighbourhood                    object
latitude                        float64
longitude                       float64
room_type                        object
price                             int64
minimum_nights                    int64
number_of_reviews                 int64
last_review                      object
reviews_per_month               float64
calculated_host_listings_count    int64
availability_365                  int64
dtype: object
```

In [9]:
```
airbnb.describe()
```

Out[9]:

	id	latitude	longitude	price	minimum_nights	number_of_reviews	reviews_per_month	calculate
count	242.000000	242.000000	242.000000	242.000000	242.000000	242.000000	242.000000	
mean	31667.024793	40.729170	-73.964527	144.272727	8.479339	106.438017	1.091653	
std	17953.882898	0.048392	0.029916	92.279028	20.365172	100.201324	1.000168	
min	2539.000000	40.631880	-74.080880	40.000000	1.000000	1.000000	0.010000	
25%	16430.250000	40.688108	-73.985222	85.000000	2.000000	24.250000	0.280000	
50%	28651.500000	40.720280	-73.965835	125.000000	3.000000	79.500000	0.795000	
75%	46864.000000	40.759568	-73.948373	175.000000	5.000000	167.000000	1.650000	
max	62430.000000	40.864820	-73.765970	800.000000	200.000000	467.000000	4.720000	

In [10]:
```
airbnb.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 242 entries, 0 to 248
Data columns (total 15 columns):
 #   Column                          Non-Null Count  Dtype  
---  ------                          --------------  -----  
 0   id                              242 non-null    int64  
 1   name                            242 non-null    object 
 2   host_name                       242 non-null    object 
 3   neighbourhood_group             242 non-null    object 
 4   neighbourhood                   242 non-null    object 
 5   latitude                        242 non-null    float64
 6   longitude                       242 non-null    float64
 7   room_type                       242 non-null    object 
 8   price                           242 non-null    int64  
 9   minimum_nights                  242 non-null    int64  
 10  number_of_reviews               242 non-null    int64  
 11  last_review                     242 non-null    object 
 12  reviews_per_month               242 non-null    float64
 13  calculated_host_listings_count  242 non-null    int64  
 14  availability_365                242 non-null    int64  
dtypes: float64(3), int64(6), object(6)
memory usage: 30.2+ KB
```

In [11]:
```
airbnb.isna().sum()
```

Out[11]:
```
id                                0
name                              0
host_name                         0
neighbourhood_group               0
neighbourhood                     0
latitude                          0
longitude                         0
room_type                         0
price                             0
minimum_nights                    0
number_of_reviews                 0
last_review                       0
reviews_per_month                 0
calculated_host_listings_count    0
availability_365                  0
dtype: int64
```

In [12]:
```
airbnb.shape[0]
```

Out[12]:
```
242
```

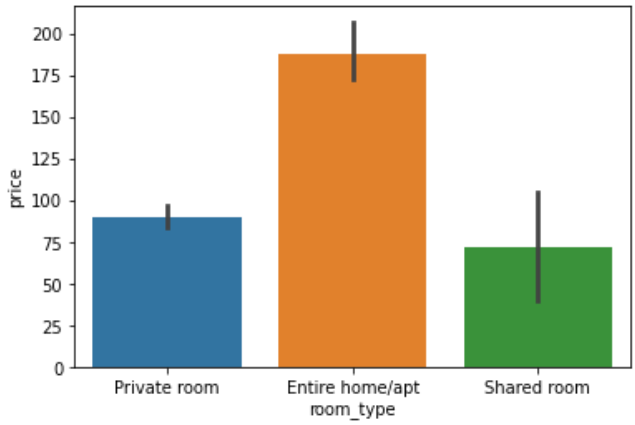
In [13]:
```
airbnb['price'].mean()
```

Out[13]:
```
144.27272727272728
```

```
# Bar plot is use to show the price and the room_type.  price and room_type column is used from the Airbnb Dataset
```

In [14]:
```
sns.barplot(y='price', x ='room_type', data =airbnb)
```

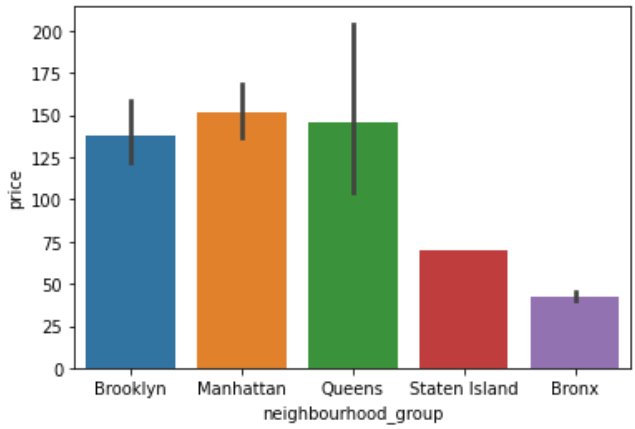
Out[14]:
```
<AxesSubplot:xlabel='room_type', ylabel='price'>
```



Bar plot is use to show the price and the locality. price and neighbourhood_group column is used from the Airbnb Dataset

In [15]:
```
sns.barplot(y='price', x ='neighbourhood_group', data =airbnb)
```

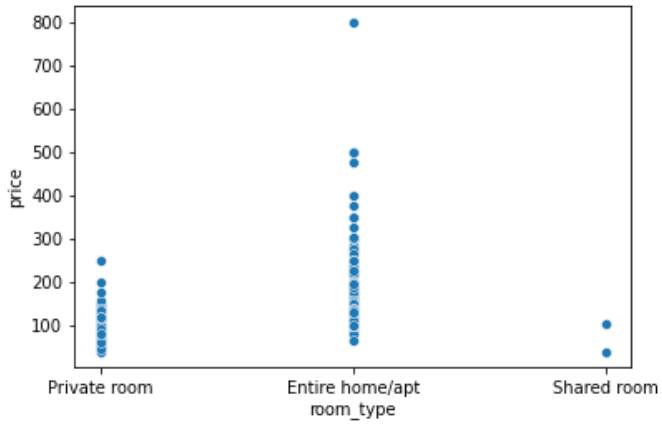
Out[15]:
```
<AxesSubplot:xlabel='neighbourhood_group', ylabel='price'>
```



Scatterplot is use to show the price and the roomtype. price and room_type column is used from the Airbnb Dataset

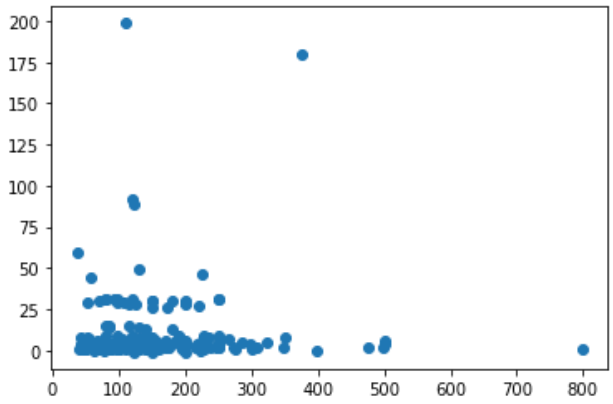
In [16]:
```
sns.scatterplot(x=airbnb.room_type, y =airbnb.price)
```

Out[16]:
```
<AxesSubplot:xlabel='room_type', ylabel='price'>
```



Scatterplot is use to show the price with minimum_nights. price and minimum_nights column is used from the Airbnb Dataset

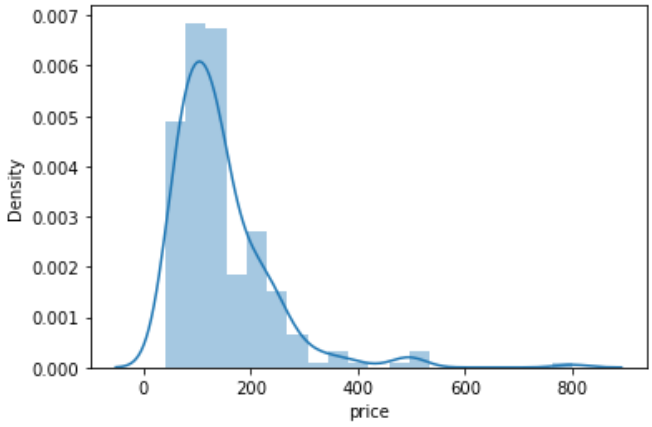
In [32]:
```
x = np.random.normal(airbnb.price)
y = np.random.normal(airbnb.minimum_nights)

plt.scatter(x, y)
plt.show()
```



Distributionplot is use to show the price. price column is used from the Airbnb Dataset

In [17]:
```
sns.distplot(airbnb['price'], bins = 20)
plt.show()
```

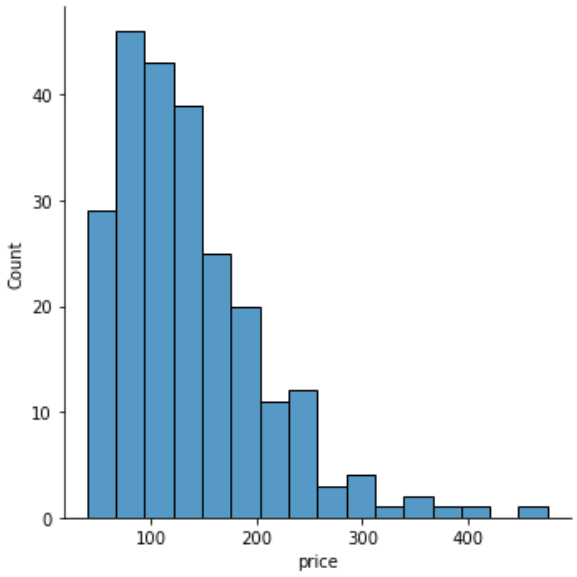
```
C:\Users\V3iT\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Displot is use with threshold the price to show the price. price column is used from the Airbnb Dataset

In [18]:
```
threshold = airbnb['price'].quantile(0.99) # get 99th quantile of Price
tmpData = airbnb[airbnb['price'] < threshold] # I'm slicing data where price < threshold
sns.displot(data = tmpData, x = 'price')
```

Out[18]:
```
<seaborn.axisgrid.FacetGrid at 0x21e0e550400>
```



In [20]:
```
airbnb.head()
```

Out[20]:

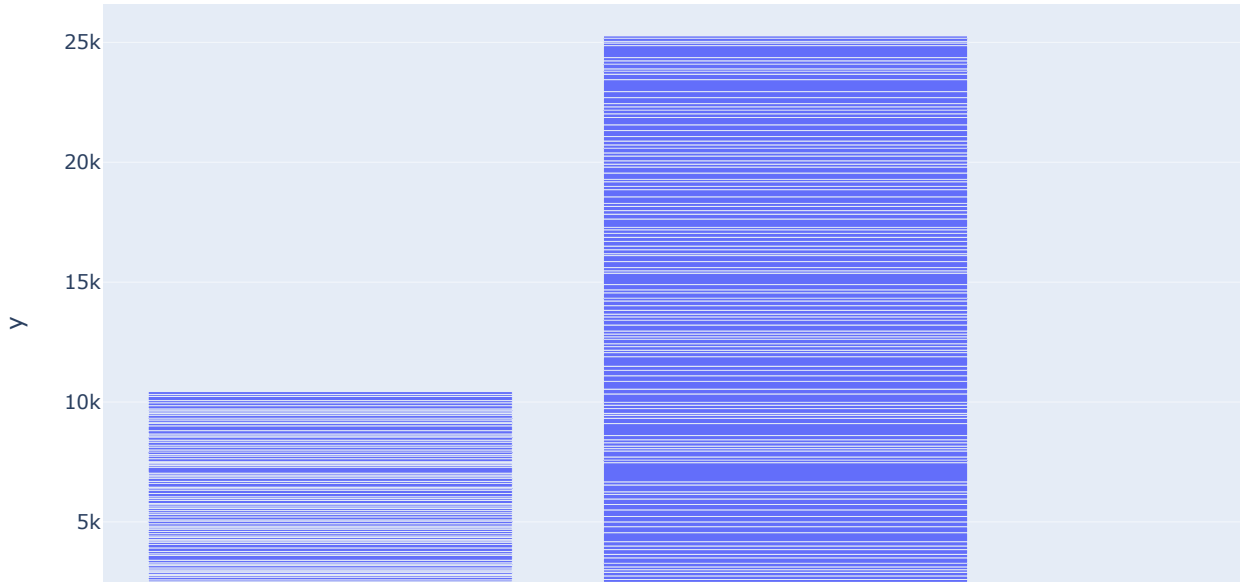
	id	name	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_n
0	2539	Clean & Quiet Apt Home By The Park	John	Brooklyn	Kensington	40.64749	-73.97237	Private room	149	
1	2595	Skylit Midtown Castle	Jennifer	Manhattan	Midtown	40.75362	-73.98377	Entire home/apt	225	
3	3831	Cozy Entire Floor Of Brownstone	LisaRoxanne	Brooklyn	Clinton Hill	40.68514	-73.95976	Entire home/apt	89	
4	5022	Entire Apt: Spacious Studio/Loft By Central Park	Laura	Manhattan	East Harlem	40.79851	-73.94399	Entire home/apt	80	
5	5099	Large Cozy 1 Br Apartment In Midtown East	Chris	Manhattan	Murray Hill	40.74767	-73.97500	Entire home/apt	200	

In [43]:
```
pip install plotly.express
```

```
Requirement already satisfied: plotly.express in c:\users\v3it\anaconda3\lib\site-packages (0.4.1)
Requirement already satisfied: patsy>=0.5 in c:\users\v3it\anaconda3\lib\site-packages (from plotl
y.express) (0.5.2)
Requirement already satisfied: plotly>=4.1.0 in c:\users\v3it\anaconda3\lib\site-packages (from plo
tly.express) (5.6.0)
Requirement already satisfied: numpy>=1.11 in c:\users\v3it\anaconda3\lib\site-packages (from plotl
y.express) (1.21.5)
Requirement already satisfied: pandas>=0.20.0 in c:\users\v3it\anaconda3\lib\site-packages (from pl
otly.express) (1.4.2)
Requirement already satisfied: statsmodels>=0.9.0 in c:\users\v3it\anaconda3\lib\site-packages (fro
m plotly.express) (0.13.2)
Requirement already satisfied: scipy>=0.18 in c:\users\v3it\anaconda3\lib\site-packages (from plotl
y.express) (1.7.3)
Requirement already satisfied: pytz>=2020.1 in c:\users\v3it\anaconda3\lib\site-packages (from pand
as>=0.20.0->plotly.express) (2021.3)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\v3it\anaconda3\lib\site-packages
(from pandas>=0.20.0->plotly.express) (2.8.2)
Requirement already satisfied: six in c:\users\v3it\anaconda3\lib\site-packages (from patsy>=0.5->p
lotly.express) (1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in c:\users\v3it\anaconda3\lib\site-packages (from p
lotly>=4.1.0->plotly.express) (8.0.1)
Requirement already satisfied: packaging>=21.3 in c:\users\v3it\anaconda3\lib\site-packages (from s
tatsmodels>=0.9.0->plotly.express) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\users\v3it\anaconda3\lib\site-package
s (from packaging>=21.3->statsmodels>=0.9.0->plotly.express) (3.0.4)
Note: you may need to restart the kernel to use updated packages.
```

PLotly- barplot is use to show the price with room_type. price and room_type column is used from the Airbnb Dataset

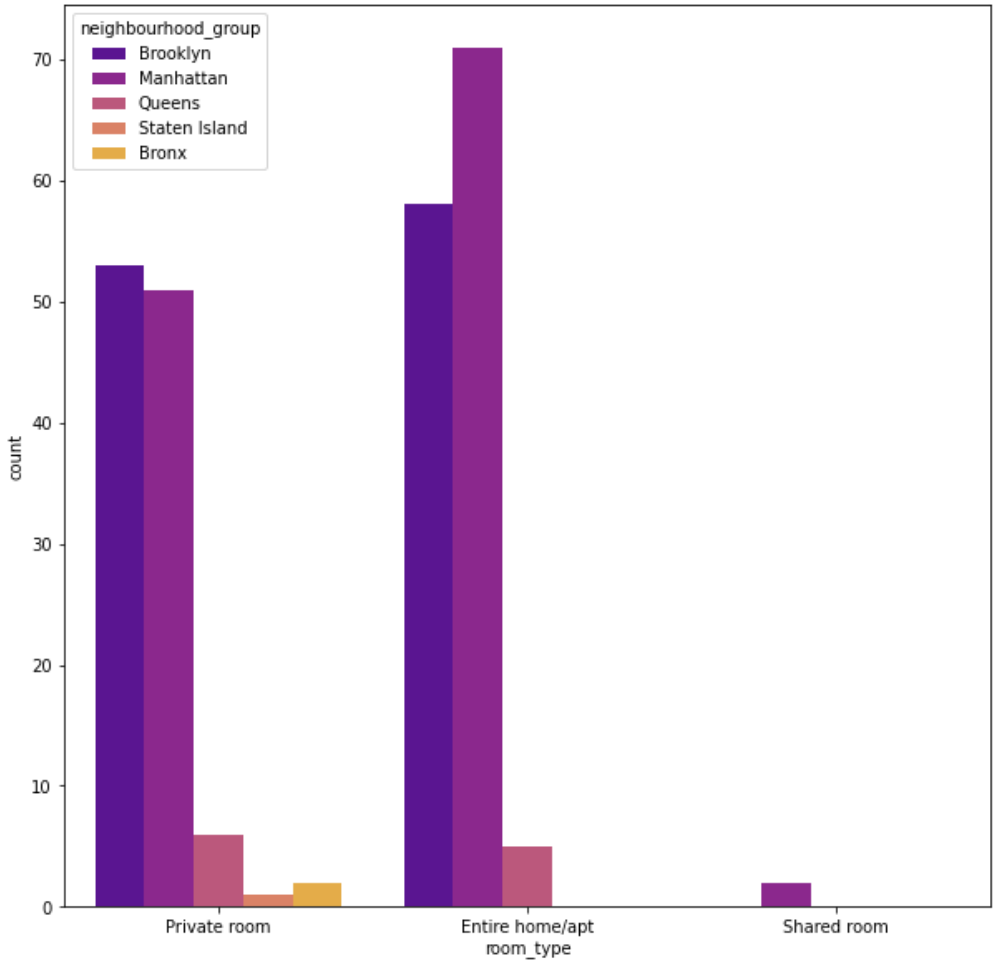
In [18]:
```
import plotly.express as px
fig = px.bar(x=airbnb["room_type"], y=airbnb["price"])
fig.show()
```



Countlot is use to show the location/place with room_type. room_type and neighbourhood_group column is used from the Airbnb Dataset

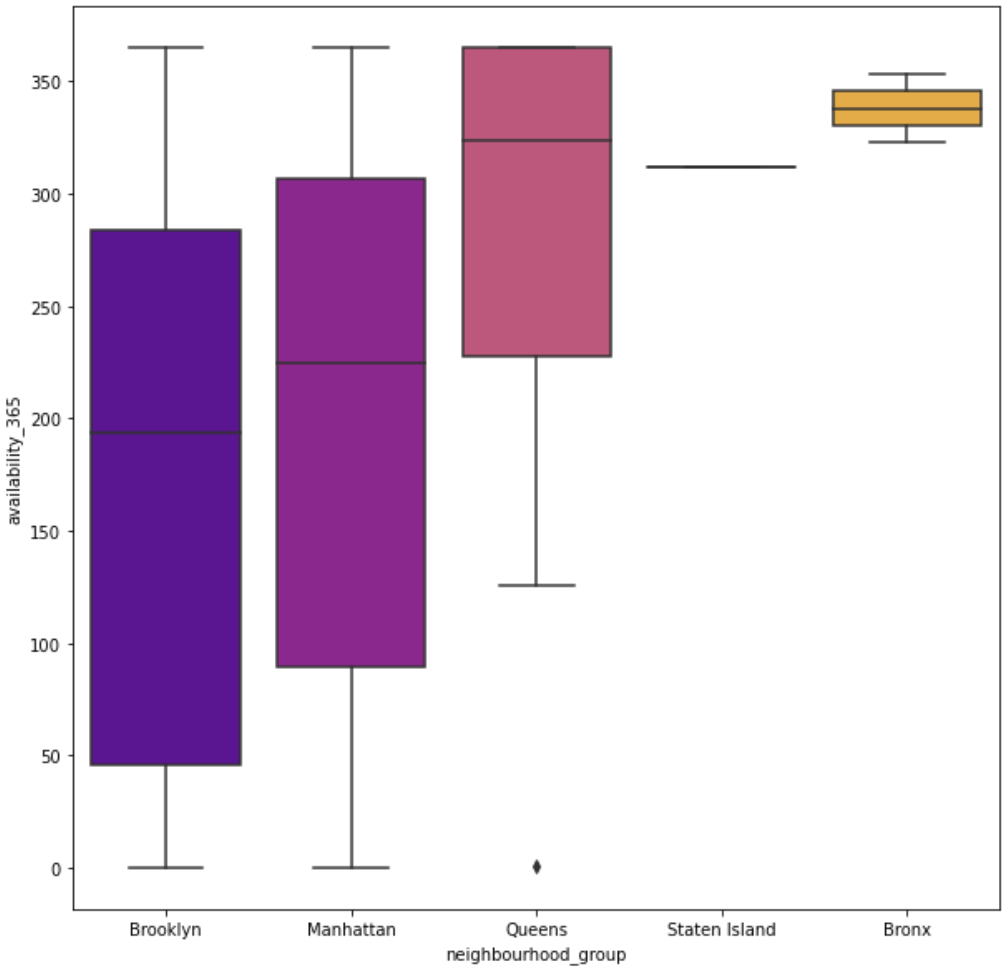
In [23]:
```
plt.figure(figsize=(10,10))
df1 = sns.countplot(airbnb['room_type'], hue=airbnb['neighbourhood_group'], palette='plasma')
```

```
C:\Users\V3iT\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning:

Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argum
ent will be `data`, and passing other arguments without an explicit keyword will result in an error
or misinterpretation.
```



 # Boxtlot is use to show the avaiability of property through year. availability_365 and neighbourhood_group column is used from the Airbnb Dataset

In [25]:
```
plt.figure(figsize=(10,10))
df1 = sns.boxplot(data=airbnb, x='neighbourhood_group',y='availability_365',palette='plasma')
```



DATASET - HRDatase

In [43]:
```
HRD =pd.read_csv("HRDataset_v14.csv")
```

In [44]:
```
HRD.head()
```

Out[44]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScoreID	FromDiversityJobFairID
0	Adinolfi, Wilson K	10026	0	0	1	1	5	4	0
1	Ait Sidi, Karthikeyan	10084	1	1	1	5	3	3	0
2	Akinkuolie, Sarah	10196	1	1	0	5	5	3	0
3	Alagbe,Trina	10088	1	1	0	1	5	3	0
4	Anderson, Carol	10069	0	2	0	5	5	3	0

5 rows × 36 columns

In [28]:
```
HRD["Employee_Name"] = HRD["Employee_Name"].str.replace(",", "")
```

In [29]:
```
HRD.head()
```

Out[29]:

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScoreID	FromDiversityJobFairID
0	Adinolfi Wilson K	10026	0	0	1	1	5	4	0
1	Ait Sidi Karthikeyan	10084	1	1	1	5	3	3	0
2	Akinkuolie Sarah	10196	1	1	0	5	5	3	0
3	AlagbeTrina	10088	1	1	0	1	5	3	0
4	Anderson Carol	10069	0	2	0	5	5	3	0

5 rows × 36 columns

In [30]:
```
HRD.dtypes
```

Out[30]:
```
Employee_Name                object
EmpID                         int64
MarriedID                     int64
MaritalStatusID               int64
GenderID                      int64
EmpStatusID                   int64
DeptID                        int64
PerfScoreID                   int64
FromDiversityJobFairID        int64
Salary                        int64
Termd                         int64
PositionID                    int64
Position                     object
State                        object
Zip                           int64
DOB                          object
Sex                          object
MaritalDesc                  object
CitizenDesc                  object
HispanicLatino               object
RaceDesc                     object
DateofHire                   object
DateofTermination            object
TermReason                   object
EmploymentStatus             object
Department                   object
ManagerName                  object
ManagerID                   float64
RecruitmentSource            object
PerformanceScore             object
EngagementSurvey            float64
EmpSatisfaction               int64
SpecialProjectsCount          int64
LastPerformanceReview_Date   object
DaysLateLast30                int64
Absences                      int64
dtype: object
```

In [31]:
```
HRD.describe()
```

Out[31]:

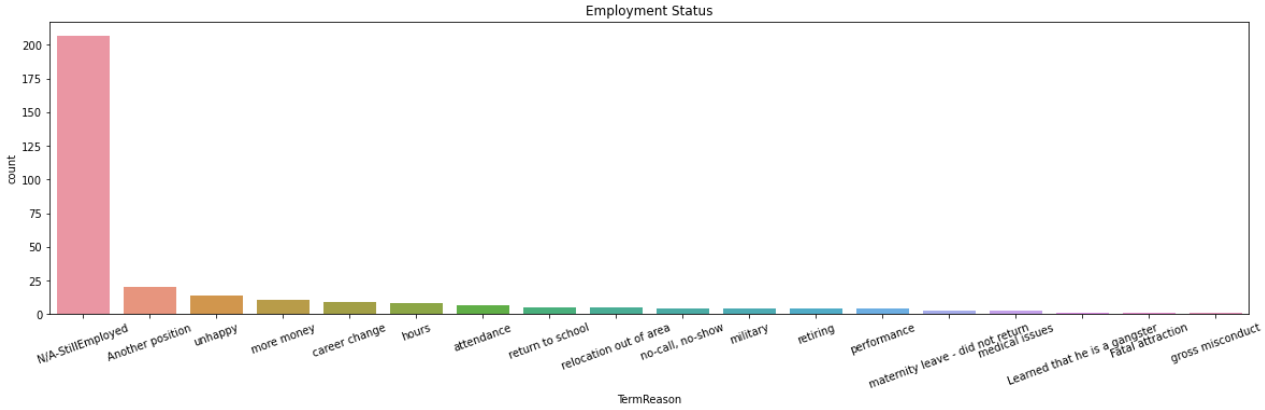
	EmpID	MarriedID	MaritalStatusID	GenderID	EmpStatusID	DeptID	PerfScoreID	FromDiversityJobFairID
count	311.000000	311.000000	311.000000	311.000000	311.000000	311.000000	311.000000	311.000000
mean	10156.000000	0.398714	0.810289	0.434084	2.392283	4.610932	2.977492	0.093248
std	89.922189	0.490423	0.943239	0.496435	1.794383	1.083487	0.587072	0.291248
min	10001.000000	0.000000	0.000000	0.000000	1.000000	1.000000	1.000000	0.000000
25%	10078.500000	0.000000	0.000000	0.000000	1.000000	5.000000	3.000000	0.000000
50%	10156.000000	0.000000	1.000000	0.000000	1.000000	5.000000	3.000000	0.000000
75%	10233.500000	1.000000	1.000000	1.000000	5.000000	5.000000	3.000000	0.000000
max	10311.000000	1.000000	4.000000	1.000000	5.000000	6.000000	4.000000	1.000000

In [32]:
```
HRD.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 311 entries, 0 to 310
Data columns (total 36 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Employee_Name               311 non-null    object
 1   EmpID                       311 non-null    int64
 2   MarriedID                   311 non-null    int64
 3   MaritalStatusID             311 non-null    int64
 4   GenderID                    311 non-null    int64
 5   EmpStatusID                 311 non-null    int64
 6   DeptID                      311 non-null    int64
 7   PerfScoreID                 311 non-null    int64
 8   FromDiversityJobFairID      311 non-null    int64
 9   Salary                      311 non-null    int64
 10  Termd                       311 non-null    int64
 11  PositionID                  311 non-null    int64
 12  Position                    311 non-null    object
 13  State                       311 non-null    object
 14  Zip                         311 non-null    int64
 15  DOB                         311 non-null    object
 16  Sex                         311 non-null    object
 17  MaritalDesc                 311 non-null    object
 18  CitizenDesc                 311 non-null    object
 19  HispanicLatino              311 non-null    object
 20  RaceDesc                    311 non-null    object
 21  DateofHire                  311 non-null    object
 22  DateofTermination           104 non-null    object
 23  TermReason                  311 non-null    object
 24  EmploymentStatus            311 non-null    object
 25  Department                  311 non-null    object
 26  ManagerName                 311 non-null    object
 27  ManagerID                   303 non-null    float64
 28  RecruitmentSource           311 non-null    object
 29  PerformanceScore            311 non-null    object
 30  EngagementSurvey            311 non-null    float64
 31  EmpSatisfaction             311 non-null    int64
 32  SpecialProjectsCount        311 non-null    int64
 33  LastPerformanceReview_Date  311 non-null    object
 34  DaysLateLast30              311 non-null    int64
 35  Absences                    311 non-null    int64
dtypes: float64(2), int64(16), object(18)
memory usage: 87.6+ KB
```

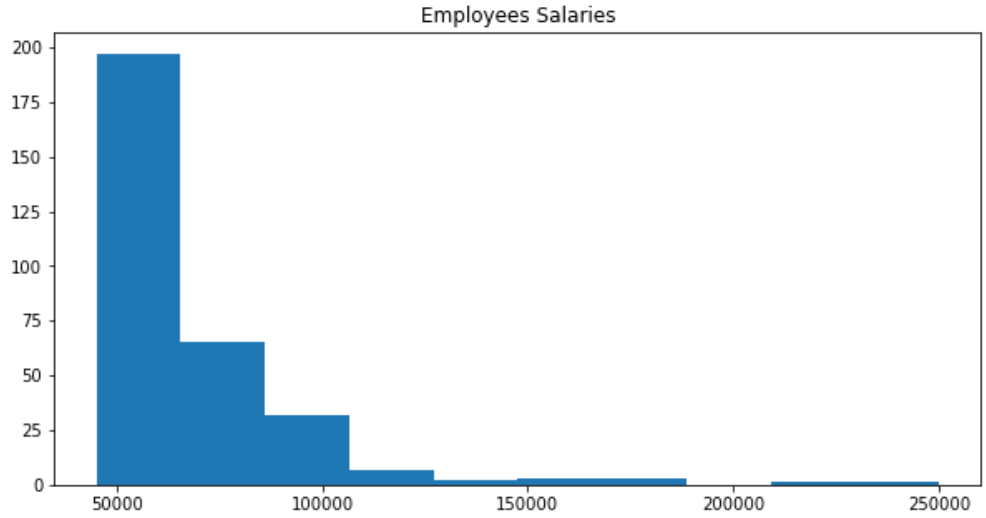
Next Count Plot is use to show the Reasons for Termination. TermReason column is used from the HRDataset

In [33]:
```
plt.figure(figsize = [20, 5])
ordered = HRD['TermReason'].value_counts().index
sns.countplot(data=HRD, x= 'TermReason', order = ordered);
plt.title('Employment Status');
plt.xticks(rotation=20);
```



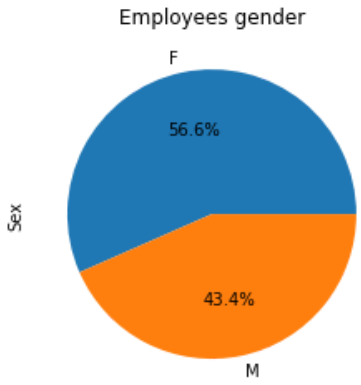
Next Histogram is use to show the salary scale of the company. Salary column is used from the HRDataset

In [34]:
```
# determine the salary scale of the company

plt.figure(figsize = [10, 5])
bas_color = sns.color_palette()[0]
plt.hist(data=HRD, x= 'Salary', color=bas_color);
plt.title('Employees Salaries');
```



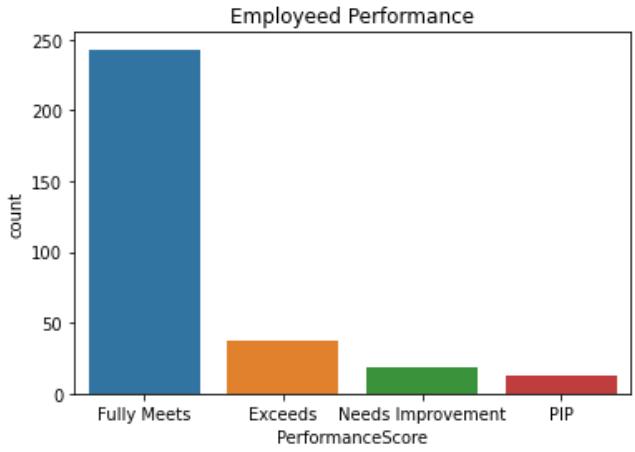
Next Piechart is use to show the Which gender is more in the employment. Salary column is used from the HRDataset

In [35]:
```
HRD['Sex'].value_counts().plot(kind='pie',autopct='%1.1f%%');
plt.title('Employees gender');
```



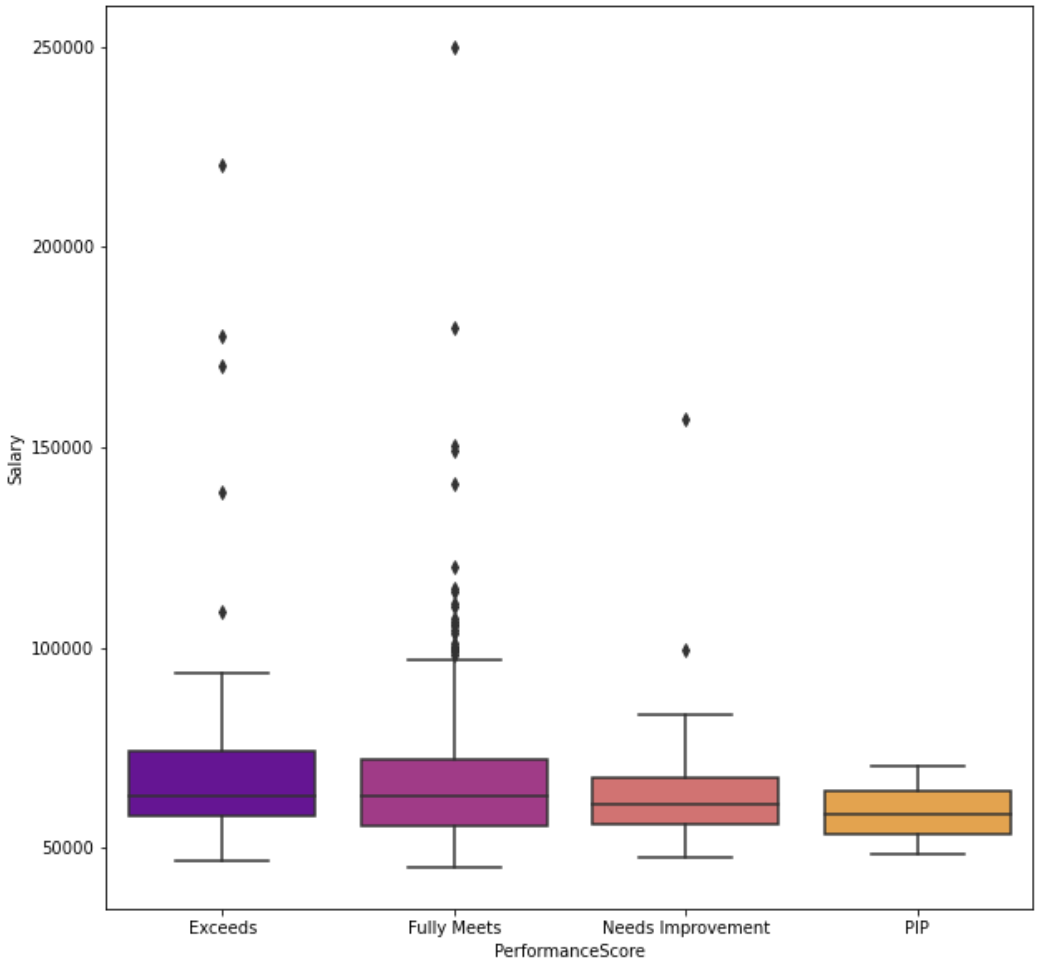
Next Count Plot is use to show the Performance Score of Employees. 'PerformanceScore' column is used from the HRDataset

In [36]:
```
ordered = HRD['PerformanceScore'].value_counts().index
sns.countplot(data=HRD, x= 'PerformanceScore', order = ordered);
plt.title('Employeed Performance');
```



Next BOX Plot is use to show the See the Performance of Employees and compare with Salary. 'PerformanceScore' and "Salary" olumn is used from the HRDataset

In [46]:
```
plt.figure(figsize=(10,10))
df1 = sns.boxplot(data=HRD, x='PerformanceScore',y='Salary',palette='plasma')
```



In [ ]: