In [3]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('Air pnb data.csv')
```

```
C:\Users\prita\AppData\Local\Temp\ipykernel_30304\3161680919.py:1: DtypeWarning: Columns (25) have mixed types. Specify dtype opti
on on import or set low_memory=False.
  df = pd.read_csv('Air pnb data.csv')
```

In [4]:
```
df.head()
```

Out[4]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	country	...
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	United States	...
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	United States	...
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	United States	...
3	1002755	NaN	85098326012	unconfirmed	Garry	Brooklyn	Clinton Hill	40.68514	-73.95976	United States	...
4	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	United States	...

5 rows × 26 columns

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 102599 entries, 0 to 102598
Data columns (total 26 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   id                              102599 non-null  int64
 1   NAME                            102349 non-null  object
 2   host id                         102599 non-null  int64
 3   host_identity_verified          102310 non-null  object
 4   host name                       102193 non-null  object
 5   neighbourhood group             102570 non-null  object
 6   neighbourhood                   102583 non-null  object
 7   lat                             102591 non-null  float64
 8   long                            102591 non-null  float64
 9   country                         102067 non-null  object
 10  country code                    102468 non-null  object
 11  instant_bookable                102494 non-null  object
 12  cancellation_policy             102523 non-null  object
 13  room type                       102599 non-null  object
 14  Construction year               102385 non-null  float64
 15  price                           102352 non-null  object
 16  service fee                     102326 non-null  object
 17  minimum nights                  102190 non-null  float64
 18  number of reviews               102416 non-null  float64
 19  last review                     86706 non-null   object
 20  reviews per month               86720 non-null   float64
 21  review rate number              102273 non-null  float64
 22  calculated host listings count  102280 non-null  float64
 23  availability 365                102151 non-null  float64
 24  house_rules                     50468 non-null   object
 25  license                         2 non-null       object
dtypes: float64(9), int64(2), object(15)
memory usage: 20.4+ MB
```

In [6]:
```
df.columns
```

Out[6]:
```
Index(['id', 'NAME', 'host id', 'host_identity_verified', 'host name',
       'neighbourhood group', 'neighbourhood', 'lat', 'long', 'country',
       'country code', 'instant_bookable', 'cancellation_policy', 'room type',
       'Construction year', 'price', 'service fee', 'minimum nights',
       'number of reviews', 'last review', 'reviews per month',
       'review rate number', 'calculated host listings count',
       'availability 365', 'house_rules', 'license'],
      dtype='object')
```

In [7]:
```
df.isnull()
```

Out[7]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	country	...	service fee	minimum nights	num revi
0	False	False	False	False	False	False	False	False	False	False	...	False	False	F
1	False	False	False	False	False	False	False	False	False	False	...	False	False	F
2	False	False	False	True	False	False	False	False	False	False	...	False	False	F
3	False	True	False	False	False	False	False	False	False	False	...	False	False	F
4	False	False	False	False	False	False	False	False	False	False	...	False	False	F
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
102594	False	False	False	False	False	False	False	False	False	False	...	False	False	F
102595	False	False	False	False	False	False	False	False	False	False	...	False	False	F
102596	False	False	False	False	False	False	False	False	False	False	...	False	False	F
102597	False	False	False	False	False	False	False	False	False	False	...	False	False	F
102598	False	False	False	False	False	False	False	False	False	False	...	False	False	F

102599 rows × 26 columns

In [8]:
```
df.isnull().sum()
```

Out[8]:
```
id                                0
NAME                            250
host id                           0
host_identity_verified          289
host name                       406
neighbourhood group              29
neighbourhood                    16
lat                               8
long                              8
country                         532
country code                    131
instant_bookable                105
cancellation_policy              76
room type                         0
Construction year               214
price                           247
service fee                     273
minimum nights                  409
number of reviews               183
last review                   15893
reviews per month             15879
review rate number              326
calculated host listings count  319
availability 365                448
house_rules                   52131
license                      102597
dtype: int64
```

Since null values present in our dataset we need to handle it first

In [13]:
```
# First handle 'last review' column
df['last review'] = pd.to_datetime(df['last review'], errors ='coerce')

# Then after that we need to fill missing values
df.fillna({'reviews per month' : 0 , 'last review' : df['last review'].min()}, inplace= True)

# Drop records with missing 'name' or 'host name'
df.dropna(subset=['NAME','host name'], inplace = True)

# Reseting index after dropping rows
df.reset_index(drop=True , inplace = True)
```

In [14]:
```
df.isnull().sum()
```

Out[14]:
```
id                                  0
NAME                                0
host id                             0
host_identity_verified            276
host name                           0
neighbourhood group                26
neighbourhood                      16
lat                                 8
long                                8
country                           526
country code                      122
instant_bookable                   96
cancellation_policy                70
room type                           0
Construction year                 200
price                             239
service fee                       268
minimum nights                    403
number of reviews                 182
last review                         0
reviews per month                   0
review rate number                314
calculated host listings count    318
availability 365                  420
house_rules                     51867
license                        101947
dtype: int64
```

In [ ]:
```
# Data type correction
```

In [17]:
```
df['price'] = df['price'].replace(r'[\$,]', '', regex=True).str.replace(' ', '').astype(float)
df['service fee'] = df['service fee'].replace(r'[\$,]', '',
                                              regex=True).str.replace(' ', '').astype(float)
```

In [18]:
```
numeric_cols = ['minimum nights', 'number of reviews' , 'reviews per month']

for col in numeric_cols:
    df[col] = pd.to_numeric(df[col], errors= 'coerce')
```

In [19]:
```
# Removing duplicates
df.drop_duplicates(inplace=True)
```

In [20]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 101410 entries, 0 to 101409
Data columns (total 26 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   id                              101410 non-null  int64
 1   NAME                            101410 non-null  object
 2   host id                         101410 non-null  int64
 3   host_identity_verified          101134 non-null  object
 4   host name                       101410 non-null  object
 5   neighbourhood group             101384 non-null  object
 6   neighbourhood                   101394 non-null  object
 7   lat                             101402 non-null  float64
 8   long                            101402 non-null  float64
 9   country                         100884 non-null  object
 10  country code                    101288 non-null  object
 11  instant_bookable                101314 non-null  object
 12  cancellation_policy             101340 non-null  object
 13  room type                       101410 non-null  object
 14  Construction year               101210 non-null  float64
 15  price                           101171 non-null  float64
 16  service fee                     101142 non-null  float64
 17  minimum nights                  101016 non-null  float64
 18  number of reviews               101228 non-null  float64
 19  last review                     101410 non-null  datetime64[ns]
 20  reviews per month               101410 non-null  float64
 21  review rate number              101103 non-null  float64
 22  calculated host listings count  101092 non-null  float64
 23  availability 365                100990 non-null  float64
 24  house_rules                     49831 non-null   object
 25  license                         2 non-null       object
dtypes: datetime64[ns](1), float64(11), int64(2), object(12)
memory usage: 20.9+ MB
```

In [21]:
```
df = df.drop(columns = ['license','house_rules'], errors='ignore')
```

In [22]:
```
df.head()
```

Out[22]:

	id	NAME	host id	host_identity_verified	host name	neighbourhood group	neighbourhood	lat	long	country	...
0	1001254	Clean & quiet apt home by the park	80014485718	unconfirmed	Madaline	Brooklyn	Kensington	40.64749	-73.97237	United States	...
1	1002102	Skylit Midtown Castle	52335172823	verified	Jenna	Manhattan	Midtown	40.75362	-73.98377	United States	...
2	1002403	THE VILLAGE OF HARLEM....NEW YORK !	78829239556	NaN	Elise	Manhattan	Harlem	40.80902	-73.94190	United States	...
3	1003689	Entire Apt: Spacious Studio/Loft by central park	92037596077	verified	Lyndon	Manhattan	East Harlem	40.79851	-73.94399	United States	...
4	1004098	Large Cozy 1 BR Apartment In Midtown East	45498551794	verified	Michelle	Manhattan	Murray Hill	40.74767	-73.97500	United States	...

5 rows × 24 columns

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

Out[23]:

	id	host id	lat	long	Construction year	price	service fee	minimum nights	number o reviews
count	1.014100e+05	1.014100e+05	101402.000000	101402.000000	101210.000000	101171.000000	101142.000000	101016.000000	101228.000000
mean	2.920959e+07	4.926155e+10	40.728082	-73.949663	2012.486908	625.381008	125.043998	8.113744	27.51185
min	1.001254e+06	1.236005e+08	40.499790	-74.249840	2003.000000	50.000000	10.000000	-1223.000000	0.000000
25%	1.507574e+07	2.459183e+10	40.688730	-73.982570	2007.000000	340.000000	68.000000	2.000000	1.000000
50%	2.922911e+07	4.912069e+10	40.722300	-73.954440	2012.000000	625.000000	125.000000	3.000000	7.000000
75%	4.328308e+07	7.399747e+10	40.762750	-73.932340	2017.000000	913.000000	183.000000	5.000000	31.000000
max	5.736742e+07	9.876313e+10	40.916970	-73.705220	2022.000000	1200.000000	240.000000	5645.000000	1024.000000
std	1.626820e+07	2.853703e+10	0.055850	0.049474	5.765130	331.609111	66.313374	30.378014	49.549258

Since everything looks good lets move to next

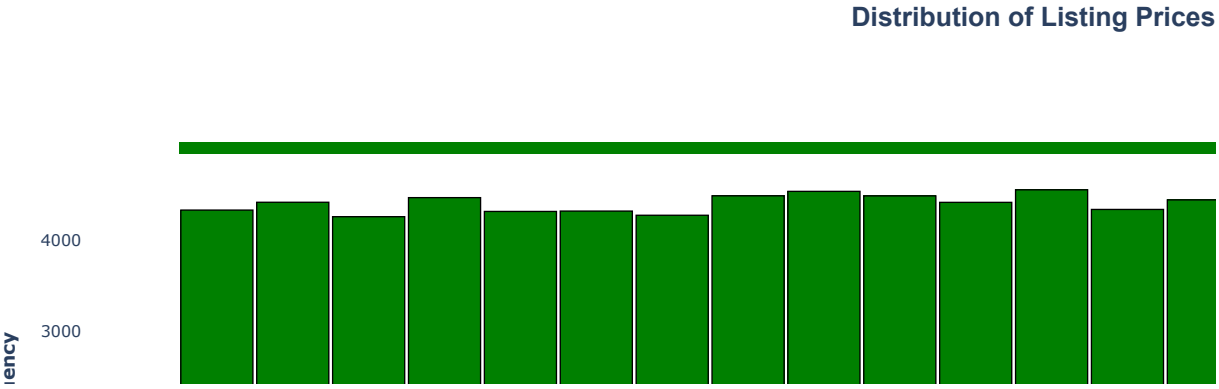
Visualization Phase

In [ ]:
```
# Distribution of Prices
```

In [24]:
```
plt.figure(figsize=(10,6))
sns.histplot(
    df['price'],
    bins = 50,
    kde = True,
    color = 'green'
)
plt.title('<b> Distribution of Listing Prices</b>')
plt.xlabel('Price ($)')
plt.ylabel('Frequency')
plt.show()
```

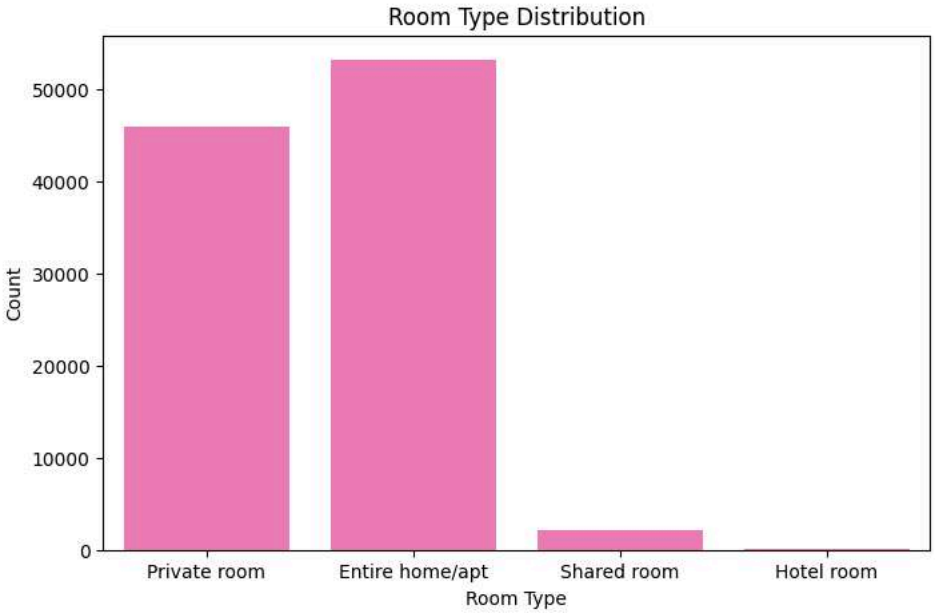
In [25]:
```
import plotly.express as px
```

In [33]:
```
fig = px.histogram(
    df,
    x ='price',
    nbins =50,
    title = '<b>Distribution of Listing Prices</b>',
    labels= {'price' : 'Price ($)'},
    color_discrete_sequence= ['green'],
    marginal= 'rug',
    hover_data={'price':'.2f'}
)
fig.update_traces(
    marker = dict(line = dict(width = 1 , color = 'black')),
    selector = dict(type = 'histogram')
)
fig.update_layout(
    title_font = dict(size =20, family = 'Arial'),
    xaxis_title = '<b>Price ($)</b>',
    yaxis_title = '<b>Frequency</b>',
    bargap = 0.05,
    plot_bgcolor = 'white',
    paper_bgcolor = 'white',
    title_x = 0.5,
    margin = dict(l =40 , r =40 , t =80 , b = 40),
    hovermode = "x unified"
)
fig.show()
```



Room Type Distribution

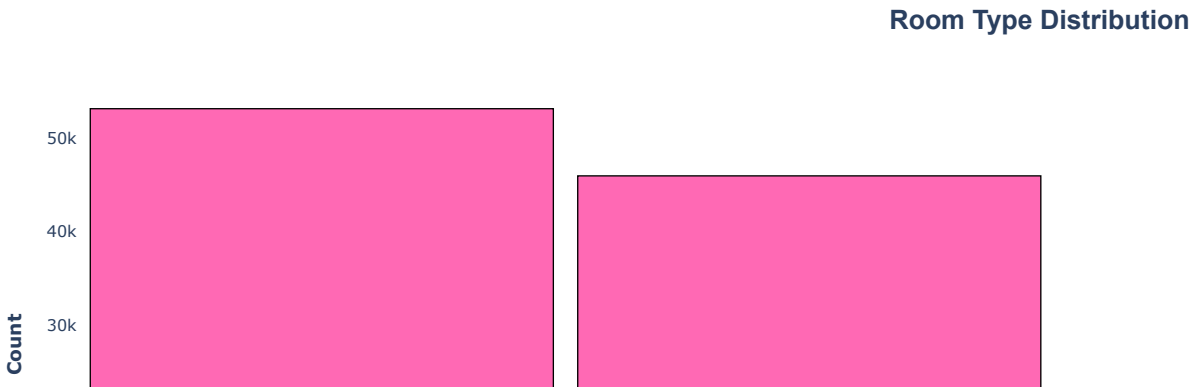
In [36]:
```
plt.figure(figsize=(8,5))
sns.countplot( x ='room type' , data = df , color = 'hotpink')
plt.title('Room Type Distribution')
plt.xlabel('Room Type')
plt.ylabel('Count')
plt.show()
```



In [37]:
```
room_counts = df['room type'].value_counts().reset_index()
room_counts.columns = ['room type' , 'count']
```

In [48]:
```
fig1 = px.bar(
    room_counts,
    x = 'room type',
    y = 'count',
    color_discrete_sequence=['hotpink'],
    title = '<b>Room Type Distribution</b>'
)
fig1.update_traces(
    marker = dict(line = dict(width = 1 , color = 'black')),
    selector = dict(type = 'bar')
)
fig1.update_layout(
    title_font = dict(size =20, family = 'Arial'),
    xaxis_title = '<b>Room Type</b>',
```

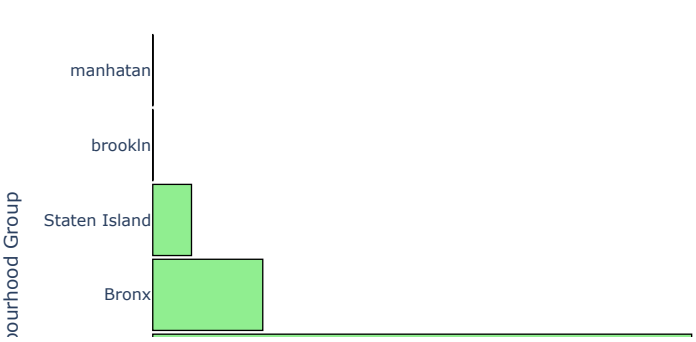
```
    yaxis_title = '<b>Count</b>',
    bargap = 0.05,
    plot_bgcolor = 'white',
    paper_bgcolor = 'white',
    title_x = 0.5,
    margin = dict(l =40 , r =40 , t =80 , b = 40),
    hovermode = "x unified"
)
fig1.show()
```



Neighborhood Analysis

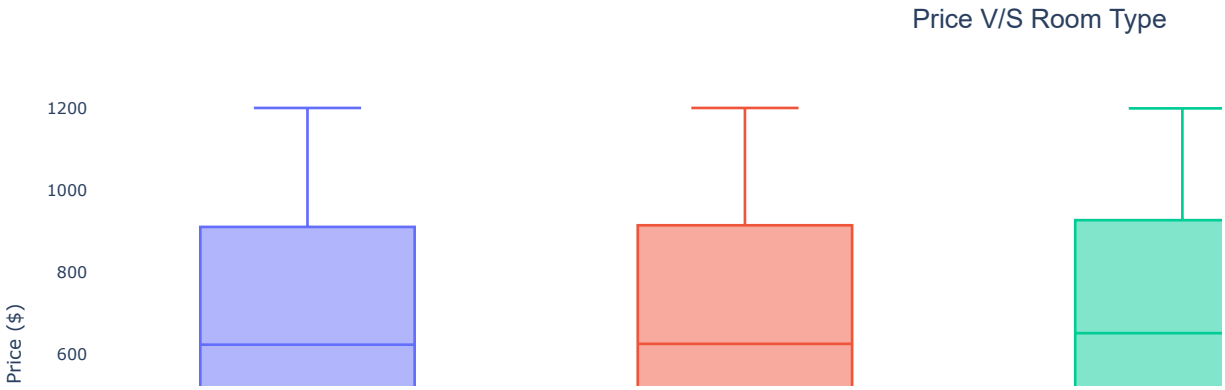
In [55]:
```
neighborhood_count = df['neighbourhood group'].value_counts().reset_index()
neighborhood_count.columns = ['neighbourhood group' , 'count']
```

In [58]:
```
fig2 = px.bar(
    neighborhood_count,
    x='count',
    y='neighbourhood group',
    title='Number of Listings by Neighbourhood Group',
    color_discrete_sequence=['lightgreen'],
)

fig2.update_traces(
    marker=dict(line=dict(width=1, color='black')),
    selector=dict(type='bar')
)

fig2.update_layout(
    title_font=dict(size=20, family='Arial'),
    xaxis_title='Number of Listings',
    yaxis_title='Neighbourhood Group',
    bargap=0.05,
    plot_bgcolor='white',
    paper_bgcolor='white',
    title_x=0.5,
    margin=dict(l=40, r=40, t=80, b=40),
    hovermode="y unified"
)

fig2.show()
```



Price V/S Room Type

In [63]:
```
fig3 = px.box(
    df,
    x = 'room type',
    y = 'price',
    # color_discrete_sequence = px.colors.qualitative.Set1,
    color='room type',
    title = 'Price V/S Room Type'
)

fig3.update_traces(
    marker=dict(line=dict(width=1, color='black')),
    selector=dict(type='box')
)

fig3.update_layout(
    title_font=dict(size=20, family='Arial'),
    xaxis_title='Room Type',
    yaxis_title='Price ($)',
    bargap=0.05,
    plot_bgcolor='white',
    paper_bgcolor='white',
    title_x=0.5,
    margin=dict(l=40, r=40, t=80, b=40),
    hovermode="x unified"
)

fig3.show()
```



In [65]:
```
df.info()
```

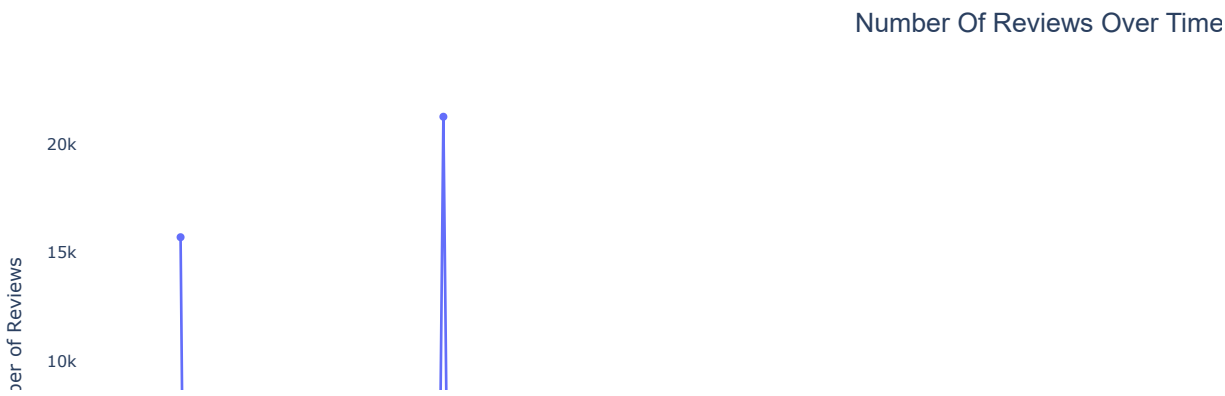
```
<class 'pandas.core.frame.DataFrame'>
Index: 101410 entries, 0 to 101409
Data columns (total 24 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   id                              101410 non-null  int64
 1   NAME                            101410 non-null  object
 2   host id                         101410 non-null  int64
 3   host_identity_verified          101134 non-null  object
 4   host name                       101410 non-null  object
 5   neighbourhood group             101384 non-null  object
 6   neighbourhood                   101394 non-null  object
 7   lat                             101402 non-null  float64
 8   long                            101402 non-null  float64
 9   country                         100884 non-null  object
 10  country code                    101288 non-null  object
 11  instant_bookable                101314 non-null  object
 12  cancellation_policy             101340 non-null  object
 13  room type                       101410 non-null  object
 14  Construction year               101210 non-null  float64
 15  price                           101171 non-null  float64
 16  service fee                     101142 non-null  float64
 17  minimum nights                  101016 non-null  float64
 18  number of reviews               101228 non-null  float64
 19  last review                     101410 non-null  datetime64[ns]
 20  reviews per month               101410 non-null  float64
 21  review rate number              101103 non-null  float64
 22  calculated host listings count  101092 non-null  float64
 23  availability 365                100990 non-null  float64
dtypes: datetime64[ns](1), float64(11), int64(2), object(10)
memory usage: 19.3+ MB
```

Reviews Over Time

In [67]:
```
df['last review'] = pd.to_datetime(df['last review'])
review_over_time = df.groupby(df['last review'].dt.to_period('M')).size()

review_over_time.index = review_over_time.index.to_timestamp()

review_df = review_over_time.reset_index()

review_df.columns = ['Month', 'Review Count']
```

In [69]:
```
fig4 = px.line(
    review_df,
    x = 'Month',
    y = 'Review Count',
    title = 'Number Of Reviews Over Time',
    markers = True,
)
fig4.update_traces(
     marker=dict(line=dict(width=1, color='black')),
    selector=dict(type='line')
)
```

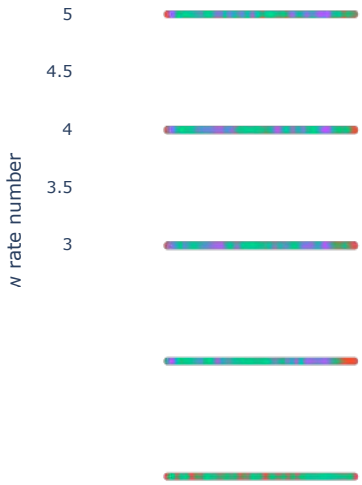
```
fig4.update_layout(
    title_font=dict(size=20, family='Arial'),
    xaxis_title='Month',
    yaxis_title='Number of Reviews',
    plot_bgcolor='white',
    paper_bgcolor='white',
    title_x=0.5,
    margin=dict(l=40, r=40, t=80, b=40),
    hovermode='x unified'
)

fig4.show()
```



In [72]:

```
fig5 = px.scatter(
    df,
    x = 'availability 365',
    y = 'review rate number',
    color = 'room type',
    title = 'Availability vs Review Rate',
    opacity = 0.6
)
fig5.update_traces(
     marker=dict(line=dict(width=1, color='black')),
    selector=dict(type='scatter')
)
fig5.update_layout(
    title_font=dict(size=20, family='Arial'),
    xaxis_title='availability 365',
    yaxis_title='review rate number',
    plot_bgcolor='white',
    paper_bgcolor='white',
    title_x=0.5,
    margin=dict(l=40, r=40, t=80, b=40),
    hovermode='y unified'
)

fig5.show()
```

Availability vs Review Rate

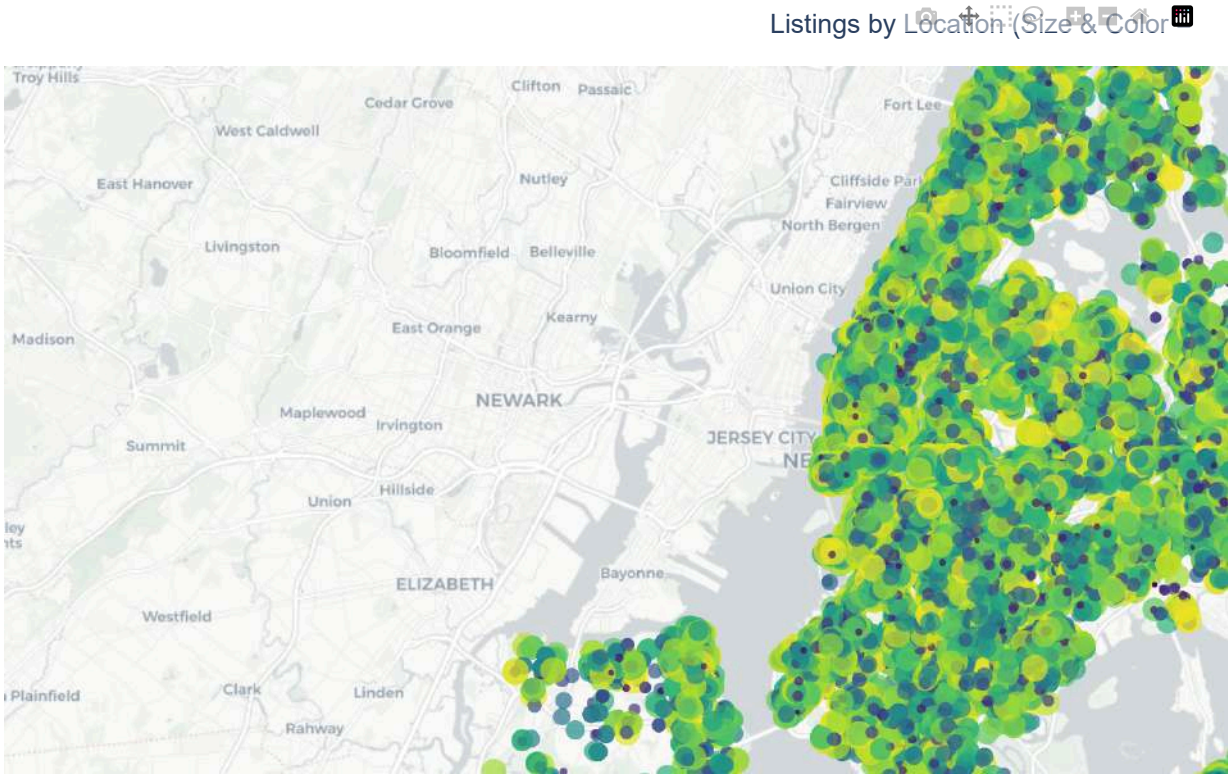


🌎 Geospatial Listing Distribution

In [74]:
```
map_data = df.dropna(subset=['lat', 'long', 'price', 'neighbourhood'])
```

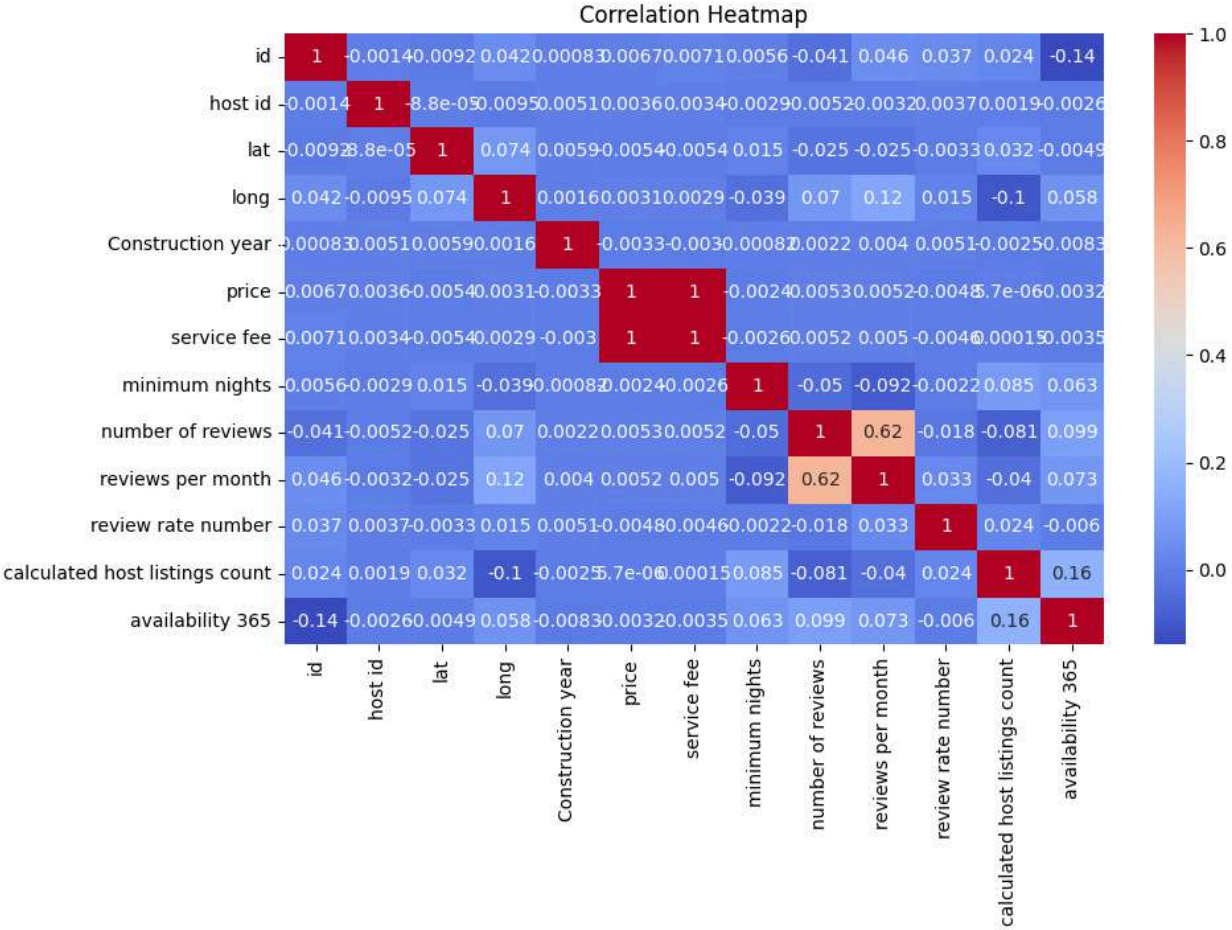
In [79]:
```
fig6 = px.scatter_mapbox(
    map_data,
    lat='lat',
    lon='long',
    color='price',
    size='price',
    hover_name='neighbourhood',
    hover_data={'lat': False, 'long': False, 'price': True},
    color_continuous_scale='Viridis',
    size_max=15,
    zoom=10,
    height=600,
    title='Listings by Location (Size & Color = Price)',
)

fig6.update_layout(
    mapbox_style='carto-positron',
    margin=dict(l=20, r=20, t=50, b=20),
    title_font=dict(size=20, family='Arial'),
    title_x=0.5
)

fig6.show()
```

```
C:\Users\prita\AppData\Local\Temp\ipykernel_30304\1690288773.py:1: DeprecationWarning:

*scatter_mapbox* is deprecated! Use *scatter_map* instead. Learn more at: https://plotly.com/python/mapbox-to-maplibre/
```



🧠 Correlation Heatmap

In [82]:
```
plt.figure(figsize=(10,6))
sns.heatmap(
    df.corr(numeric_only=True),
    annot= True,
    cmap = 'coolwarm'
)
plt.title('Correlation Heatmap')
plt.show()
```


Correlation Heatmap
id
host id
lat
long
Construction year
price
service fee
minimum nights
number of reviews
reviews per month
review rate number
calculated host listings count
availability 365
1.0
0.8
0.6
0.4
0.2
0.0