In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from wordcloud import WordCloud
```

In [2]:
```
df = pd.read_csv("AB_NYC_2019.csv")
df.head()
```

Out[2]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimu
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 [3]:
```
# Task 1: Examine the data, there may be some anomalies in the data, and you will have to clean the data
# before you move forward to other tasks. Explain what you did to clean the data.
```

In [4]:
```
# Check for zeroes in host_id, latitude, longitude, price, minimum nights, # of reviews, availability_365 and
# evaluate Zeroes in host_id, there are none
((df['host_id'] == 0)).any()
```

Out[4]:
```
False
```

In [5]:
```
# Zeroes in latitude, there are none
((df['latitude'] == 0)).any()
```

Out[5]:
```
False
```

In [6]:
```
# Zeroes in longitude, there are none
((df['longitude'] == 0)).any()
```

Out[6]:
```
False
```

In [7]:
```
# Zeroes in minimum_nights, there are none
((df['minimum_nights'] == 0)).any()
```

Out[7]:
```
False
```

In [8]:
```
# Zeroes in price, there are some
((df['price'] == 0)).any()
```

Out[8]:
```
True
```

In [9]:
```
# (df['price'] == 0).sum()
len(df.loc[df['price'] == 0])
```

Out[9]:
```
11
```

In [10]:
```
# Check number of listings before
len(df)
```

Out[10]:
```
48895
```

In [11]:
```
# Remove zero rows because removing only 11 out of all 48895 values wouldn't change much. Values can be
# removed with below code
# df.drop(df[df['price'] <= 0].index, inplace = True)
```

In [12]:
```
# Check number of listings after if you choose to remove 0 rows due to price
len(df)
```

Out[12]:
```
48895
```

In [13]:
```
# Zeroes in number_of_reviews, there are some, which is fine because some listings may not have been occupied
((df['number_of_reviews'] == 0)).any()
```

Out[13]:
```
True
```

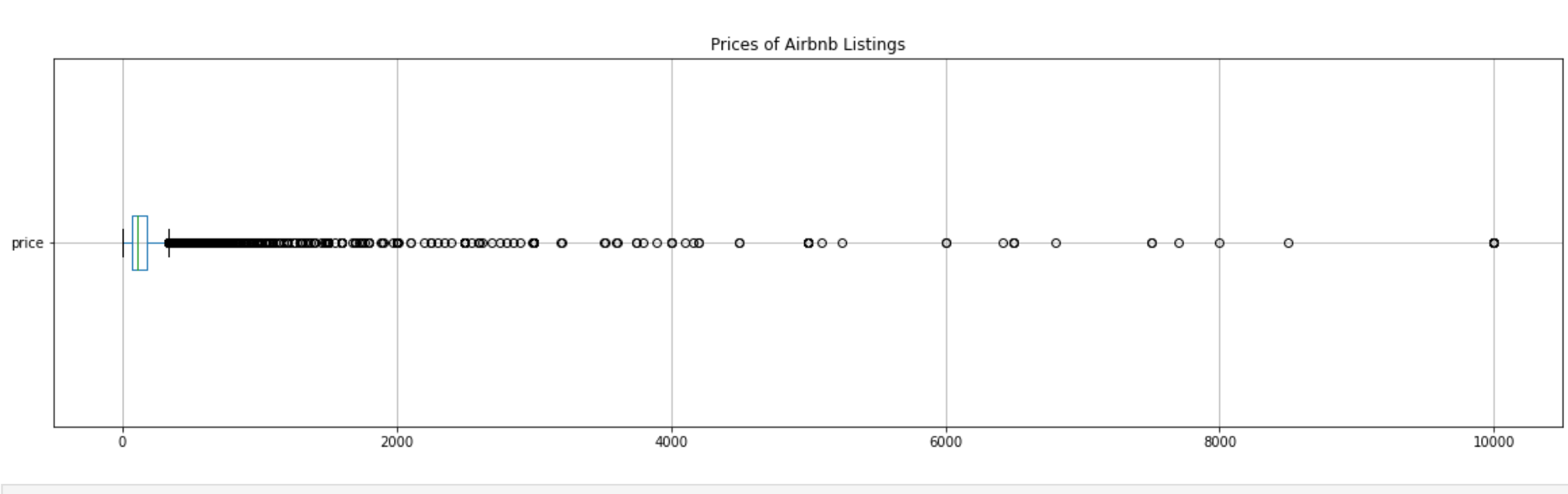
In [14]:
```
# Zeroes in availability_365, there are some, which is fine because some listings might be booked fully
((df['availability_365'] == 0)).any()
```

Out[14]:
```
True
```

In [15]:
```
# Next step of clean is to look for outliers, can do so by using box plots
```

In [16]:
```
# Boxplot for prices, there are some outliers
df.boxplot(column=['price'], vert=False, figsize=(20,5)).set_title("Prices of Airbnb Listings")
```

Out[16]:
```
Text(0.5, 1.0, 'Prices of Airbnb Listings')
```



In [17]:
```
# Checking listings with prices over 4000. They seem to be expensive because of the location, minimum nights,
# and because some are luxury apartments as indicated by the name. Wouldn't make sense to remove these
# listings.
df.loc[df['price'] > 4000].sort_values(by=['price'])
```

Out[17]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type
42736	33171891	30 days minimum Time square West Midtown apart...	177396569	Yanina	Manhattan	Hell's Kitchen	40.76043	-73.99132	Entire home/apt
28947	22296197	Chelsea Gallery for events, exhibitions, fashion	3750764	Kevin	Manhattan	Chelsea	40.74888	-74.00481	Entire home/apt
45867	34981637	bay ridge & sunset park furnished apartment	263564234	Nony	Brooklyn	Bay Ridge	40.63087	-74.02006	Entire home/apt
43670	33796251	Beautiful private Brooklyn room with kitchenette	8748976	Jeffrey	Brooklyn	Bedford-Stuyvesant	40.68807	-73.95426	Private room
4376	2952861	Photography Location	1177497	Jessica	Brooklyn	Clinton Hill	40.69127	-73.96563	Entire home/apt
46614	35345358	Northside Williamsburg Stunner	956324	Alex	Brooklyn	Williamsburg	40.71705	-73.96470	Entire home/apt
38000	30035166	4-Floor Unique Event Space 50P Cap. - #10299B	172611460	Rasmus	Manhattan	Harlem	40.82511	-73.94961	Entire home/apt
26739	21238053	Broadway 1	153497815	Sarah-B	Brooklyn	Bedford-Stuyvesant	40.68742	-73.91957	Entire home/apt
25825	20654227	Fulton 2	100069033	Sarah-2	Brooklyn	Cypress Hills	40.68185	-73.88128	Entire home/apt
22353	18051877	Victorian Film location	2675644	Alissa	Staten Island	Randall Manor	40.63952	-74.09730	Entire home/apt
2698	1448703	Beautiful 1 Bedroom in Nolita/Soho	213266	Jessica	Manhattan	Nolita	40.72193	-73.99379	Entire home/apt
4345	2919330	NearWilliamsburg bridge 11211 BK	14908606	Bianca	Brooklyn	Bedford-Stuyvesant	40.69572	-73.95731	Private room
43009	33397385	Midtown Manhattan great location (Gramacy park)	16105313	Debra	Manhattan	Midtown	40.74482	-73.98367	Entire home/apt
3720	2243699	SuperBowl Penthouse Loft 3,000 sqft	1483320	Omri	Manhattan	Little Italy	40.71895	-73.99793	Entire home/apt
3537	2110145	UWS 1BR w/backyard + block from CP	2151325	Jay And Liz	Manhattan	Upper West Side	40.77782	-73.97848	Entire home/apt
15560	12520066	Luxury townhouse Greenwich Village	66240032	Linda	Manhattan	Greenwich Village	40.73046	-73.99562	Entire home/apt
29664	22780103	Park Avenue Mansion by (Hidden by Airbnb)	156158778	Sally	Manhattan	Upper East Side	40.78517	-73.95270	Entire home/apt
3774	2271504	SUPER BOWL Brooklyn Duplex Apt!!	11598359	Jonathan	Brooklyn	Clinton Hill	40.68766	-73.96439	Entire home/apt
37194	29547314	Apartment New York \nHell's Kitchens	35303743	Patricia	Manhattan	Upper West Side	40.76835	-73.98367	Private room
48043	36056808	Luxury TriBeCa Apartment at an amazing price	271248669	Jenny	Manhattan	Tribeca	40.71206	-74.00999	Entire home/apt
44034	33998396	3000 sq ft daylight photo studio	3750764	Kevin	Manhattan	Chelsea	40.75060	-74.00388	Entire home/apt
42523	33007610	70' Luxury MotorYacht on the Hudson	7407743	Jack	Manhattan	Battery Park City	40.71162	-74.01693	Entire home/apt
45666	34895693	Gem of east Flatbush	262534951	Sandra	Brooklyn	East Flatbush	40.65724	-73.92450	Private room
29662	22779726	East 72nd Townhouse by (Hidden by Airbnb)	156158778	Sally	Manhattan	Upper East Side	40.76824	-73.95989	Entire home/apt
4377	2953058	Film Location	1177497	Jessica	Brooklyn	Clinton Hill	40.69137	-73.96723	Entire home/apt
30268	23377410	Beautiful/Spacious 1 bed luxury flat-TriBeCa/Soho	18128455	Rum	Manhattan	Tribeca	40.72197	-74.00633	Entire home/apt
40433	31340283	2br - The Heart of NYC: Manhattans Lower East ...	4382127	Matt	Manhattan	Lower East Side	40.71980	-73.98566	Entire home/apt
6530	4737930	Spanish Harlem Apt	1235070	Olson	Manhattan	East Harlem	40.79264	-73.93898	Entire home/apt
12342	9528920	Quiet, Clean, Lit @ LES & Chinatown	3906464	Amy	Manhattan	Lower East Side	40.71355	-73.98507	Private room
29238	22436899	1-BR Lincoln Center	72390391	Jelena	Manhattan	Upper West Side	40.77213	-73.98665	Entire home/apt
9151	7003697	Furnished room in Astoria apartment	20582832	Kathrine	Queens	Astoria	40.76810	-73.91651	Private room
17692	13894339	Luxury 1 bedroom apt. -stunning Manhattan views	5143901	Erin	Brooklyn	Greenpoint	40.73260	-73.95739	Entire home/apt

In [18]:
```
# Task 2: Examine how the prices of the Airbnb changes with the change in the neighborhood.
# a. Find Top 5 and Bottom 5 neighborhood based on the price of the Airbnb in that neighborhood
# (select only neighborhoods with more than 5 listings). (10 Points)
# b. Analyze, the price variation between different neighborhood group, and plot these trends. (5 Points)
```

In [19]:
```
# Top 5 neighborhood based on prices with neighborhoods with more than 5 listings are:
# Williamsburg, Bedford-Stuyvesant, Harlem, Bushwick and Upper West Side
top5df = df['neighbourhood'].value_counts()
top5df = top5df[top5df > 5]
top5df
```

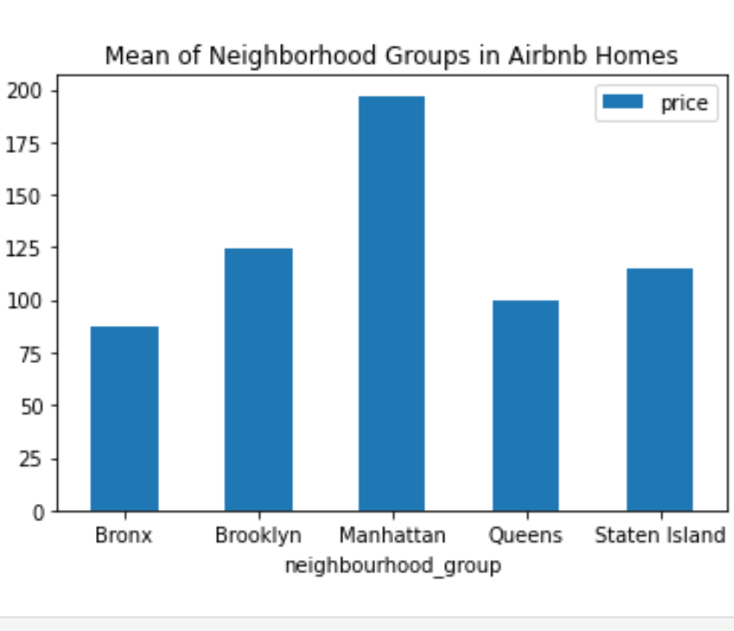
Out[19]:
```
Williamsburg          3920
Bedford-Stuyvesant    3714
Harlem                2658
Bushwick              2465
Upper West Side       1971
                      ...
Bull's Head              6
Midland Beach            6
Grant City               6
Mount Eden               6
Bay Terrace              6
Name: neighbourhood, Length: 190, dtype: int64
```

In [20]:
```
# Bottom 5 neighborhood based on prices with neighborhoods with more than 5 listings are:
# Bull's Head, Midland Beach, Grant City, Mount Eden, Bay Terrace.
# Filtered == 6 to make sure there were only 5 neighborhoods that were above 5, as seen by above cell.
bot5df = df['neighbourhood'].value_counts()
bot5df = bot5df[bot5df == 6]
bot5df
```

Out[20]:
```
Bull's Head      6
Midland Beach    6
Grant City       6
Mount Eden       6
Bay Terrace      6
Name: neighbourhood, dtype: int64
```

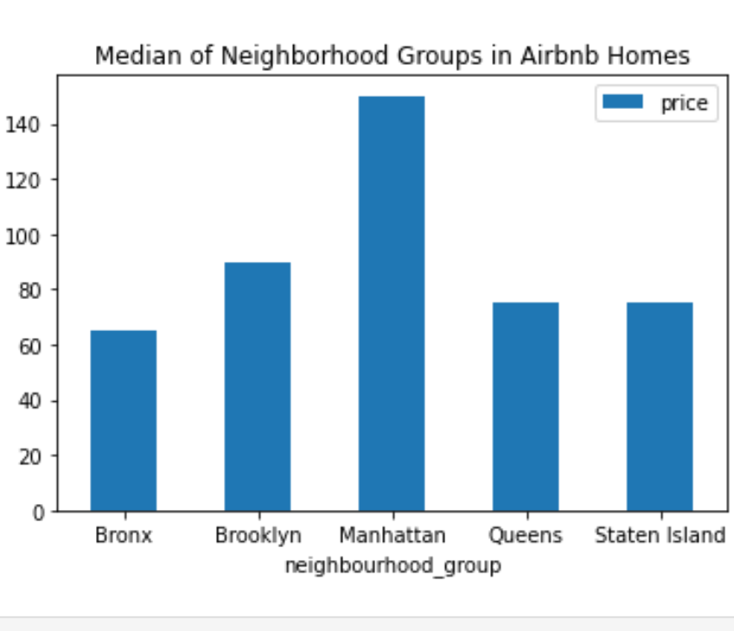
In [21]:
```
# Part B, check price variation between different groups
ngdf = df.groupby('neighbourhood_group')['price'].mean().to_frame()
ngdf.plot.bar(y='price', rot=0, title='Mean of Neighborhood Groups in Airbnb Homes')
```

Out[21]:
```
<AxesSubplot:title={'center':'Mean of Neighborhood Groups in Airbnb Homes'}, xlabel='neighbourhood_group'>
```



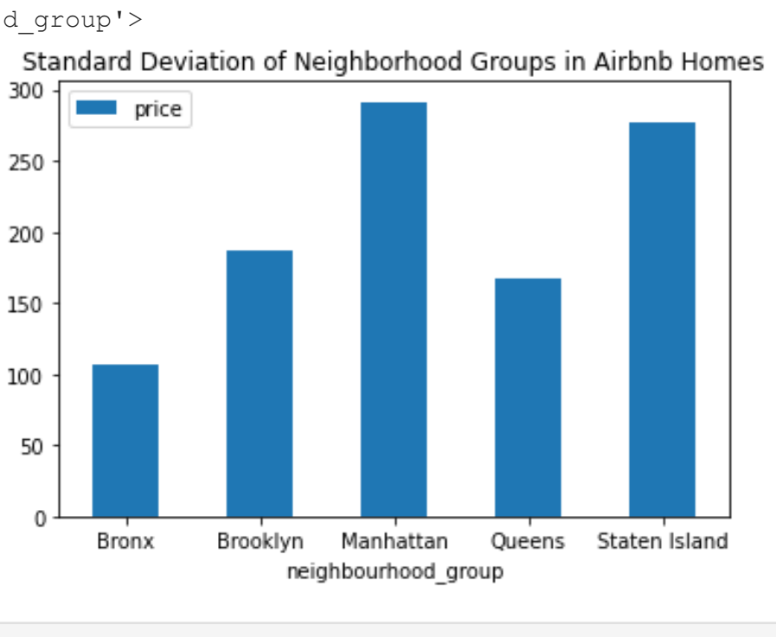
In [22]:
```
ngdf = df.groupby('neighbourhood_group')['price'].median().to_frame()
ngdf.plot.bar(y='price', rot=0, title='Median of Neighborhood Groups in Airbnb Homes')
```

Out[22]:
```
<AxesSubplot:title={'center':'Median of Neighborhood Groups in Airbnb Homes'}, xlabel='neighbourhood_group'>
```



In [23]:
```
ngdf = df.groupby('neighbourhood_group')['price'].std().to_frame()
ngdf.plot.bar(y='price', rot=0, title='Standard Deviation of Neighborhood Groups in Airbnb Homes')
```

Out[23]:
```
<AxesSubplot:title={'center':'Standard Deviation of Neighborhood Groups in Airbnb Homes'}, xlabel='neighbourhoo
d_group'>
```



In [24]:
```
# Task 3: Select a set of the most interesting features. Do a pairwise Pearson correlation analysis on all
# pairs of these variables. Show the result with a heat map and find out most positive and negative
# correlations. (5 points)
```

In [25]:
```
df.corr()
```

Out[25]:

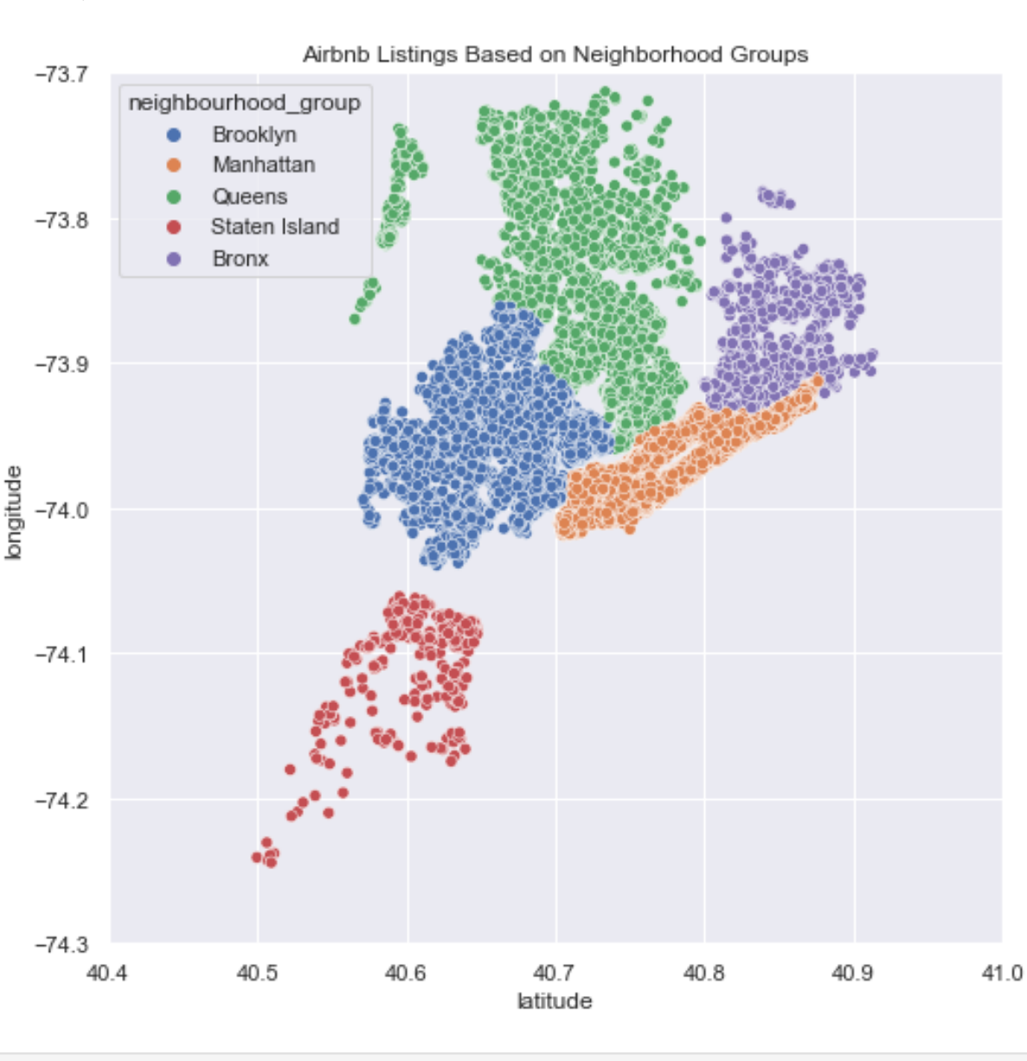
	id	host_id	latitude	longitude	price	minimum_nights	number_of_reviews	reviews_p
id	1.000000	0.588290	-0.003125	0.090908	0.010619	-0.013224	-0.319760	
host_id	0.588290	1.000000	0.020224	0.127055	0.015309	-0.017364	-0.140106	
latitude	-0.003125	0.020224	1.000000	0.084788	0.033939	0.024869	-0.015389	
longitude	0.090908	0.127055	0.084788	1.000000	-0.150019	-0.062747	0.059094	
price	0.010619	0.015309	0.033939	-0.150019	1.000000	0.042799	-0.047954	-
minimum_nights	-0.013224	-0.017364	0.024869	-0.062747	0.042799	1.000000	-0.080116	
number_of_reviews	-0.319760	-0.140106	-0.015389	0.059094	-0.047954	-0.080116	1.000000	
reviews_per_month	0.291828	0.296417	-0.010142	0.145948	-0.030608	-0.121702	0.549868	
calculated_host_listings_count	0.133272	0.154950	0.019517	-0.114713	0.057472	0.127960	-0.072376	
availability_365	0.085468	0.203492	-0.010983	0.082731	0.081829	0.144303	0.172028	

In [26]:
```
# Interesting Features:
# Most positive correlations:
# Reviews per month and number of reviews (0.55). This makes sense because the number of reviews per month
# is the total number of reviews divided by the number of months a property was listed for.
# Most negative correlations:
# Minimum nights and reviews per month
corrdf = df.copy()
corrdf = corrdf.drop(columns=['id', 'host_id'])
dataplot = sb.heatmap(corrdf.corr(), cmap="YlGnBu", annot=True).set_title("Correlations between Interesting Lis

sb.set(rc = {'figure.figsize':(8,8)})
plt.show()
```

In [27]:
```
# Task 4: The Latitude and Longitude of all the Airbnb listings are provided in the dataset.
# a. Plot a scatter plot based on these coordinates, where the points represent the location of an
# Airbnb, and the points are color coded based on the neighborhood group feature. (5 Points)
# b. Now again, plot a scatter plot based on these coordinates, where the points represent the
# location of an Airbnb, and the points are color coded based on the price of the particular Airbnb,
# where price of the listing is less than 1000. Looking at the graph can you tell which
# neighborhood group is the most expensive. (5 Points)
```

In [28]:
```
# Part A
sb.scatterplot(data=df, x="latitude", y="longitude", hue="neighbourhood_group").set_title("Airbnb Listings Base
plt.ylim(-74.3, -73.7)
plt.xlim(40.4, 41)
```

Out[28]:
```
(40.4, 41.0)
```



In [29]:
```
# Part B
# After looking at the graph, it is clear that the most expensive location seems to be Manhattan as the area
# where the most expensive listings are correspond with where Manhattan is, as seen in the previous
# scatter plot.
pricedf = df.copy()
pricedf = pricedf.loc[pricedf['price'] < 1000].sort_values(by="price")
sb.scatterplot(data=pricedf, x="latitude", y="longitude", hue="price").set_title("Airbnb Listings Based on Pric
plt.ylim(-74.3, -73.7)
plt.xlim(40.4, 41)
```

Out[29]:
```
(40.4, 41.0)
```

In [30]:
```
# Task 5:
# Word clouds are useful tool to explore the text data. Extract the words from the name of the Airbnb
# and generate a word cloud.
```

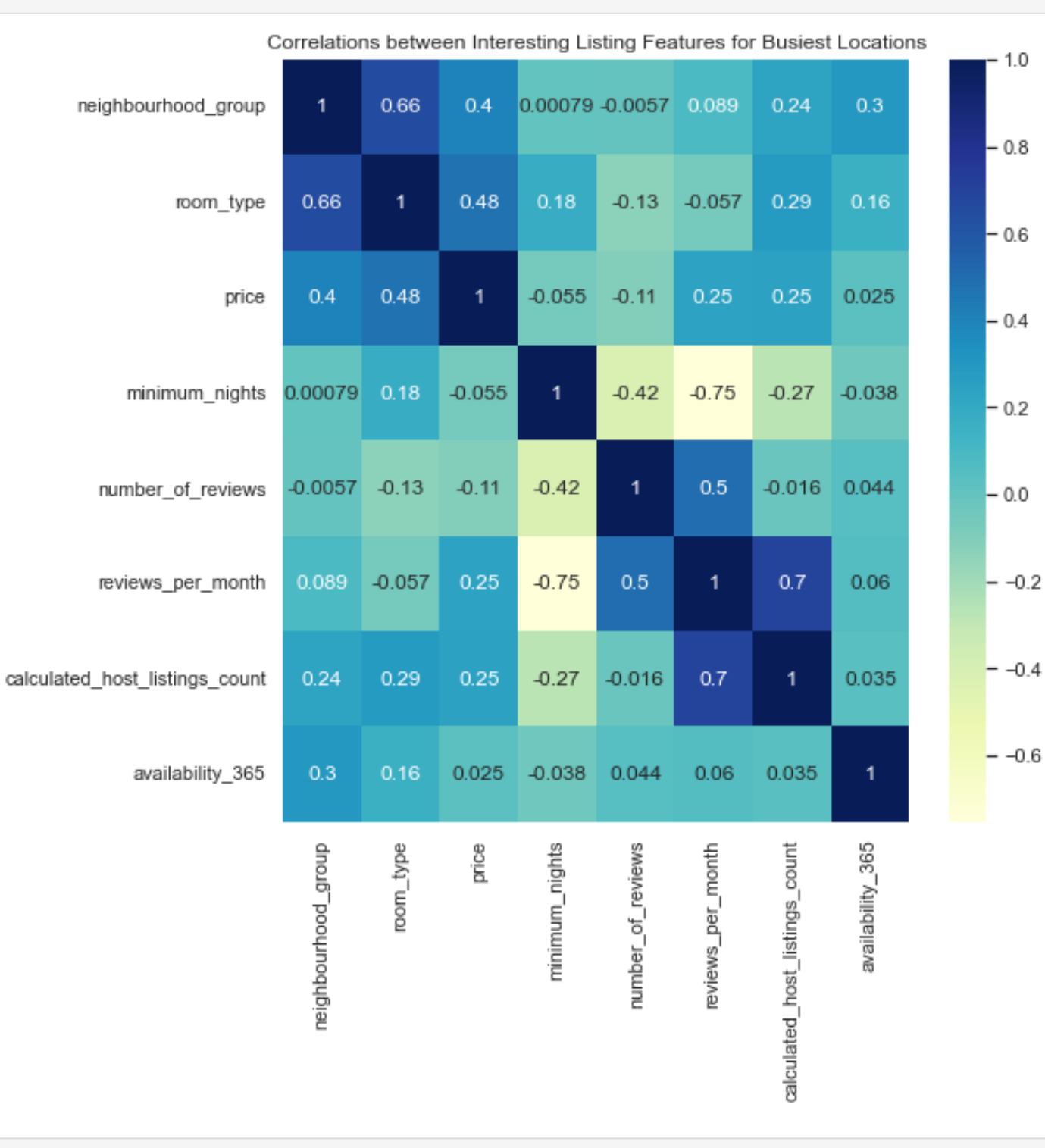
In [31]:
```
list = df['name'].to_frame().stack().str.split("[^\w+]").explode().tolist()
# list2 = []
# for i in list:
#     list2.append(i.replace('\'', ""))
# # print(list[0])
# wordcloud = WordCloud().generate(str(list2))
wordcloud = WordCloud(width = 1000, height = 1000).generate(str(list))

plt.figure()
plt.imshow(wordcloud)
plt.axis("off")
plt.show()
```

In [32]:
```
# Task 6: Find out which areas has the busiest (hosts with high number of listings) host? Are there any
# reasons, why these hosts are the busiest, considers factors such as availability, price, review, etc.?
# Bolster you reasoning with different plots and correlations. (10 Points)
```

In [33]:
```
# This scatter plot is made to see the locations where hosts have a lot of listings. The locations can be
# eyeballed with the graph scatter plots that were made in Part 4 in order to see the neighborhood groups
# based on the latitude and longitude of the locations.
busydf = df.copy()
busydf = busydf.loc[busydf['calculated_host_listings_count'] > 30].sort_values(by="calculated_host_listings_cou
sb.scatterplot(data=busydf, x="latitude", y="longitude", hue="calculated_host_listings_count").set_title("Airbn
plt.ylim(-74.3, -73.7)
plt.xlim(40.4, 41)
```

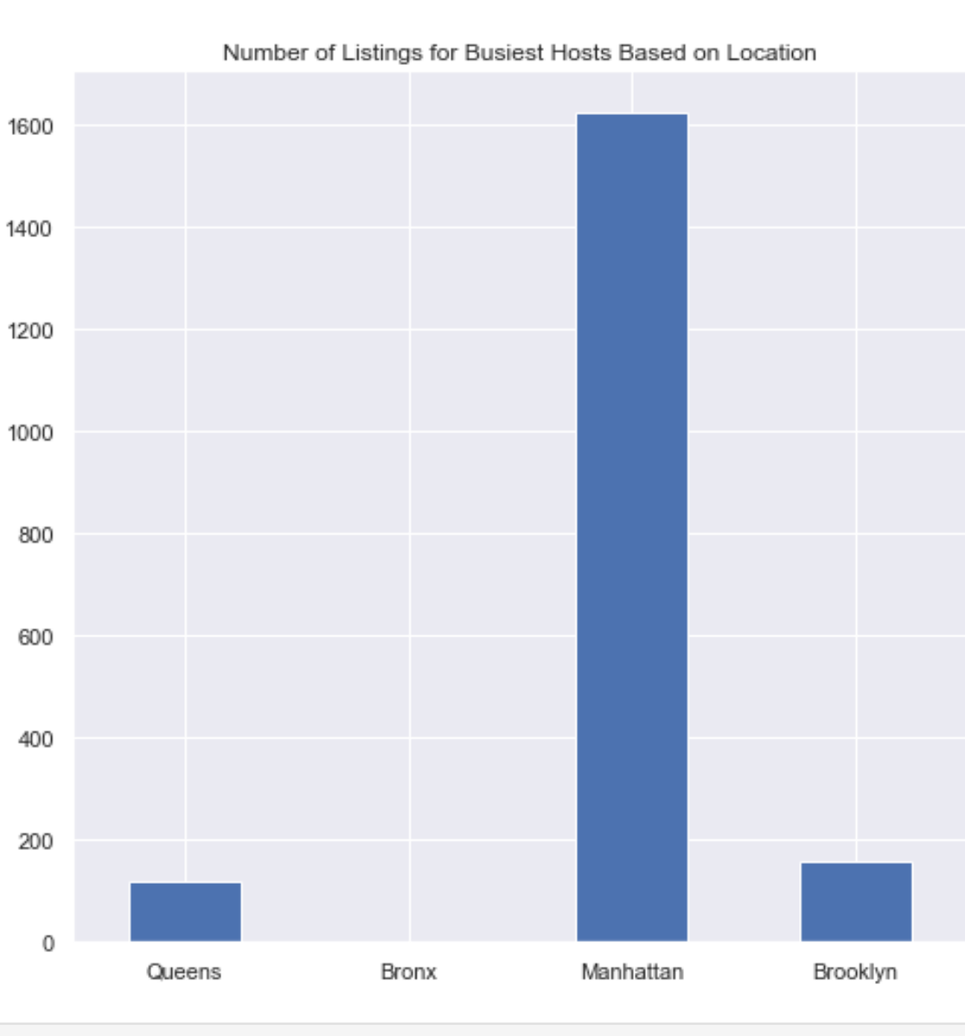
Out[33]:
```
(40.4, 41.0)
```

In [34]:
```
# In order to see the correlation between the busiest locations, the listings with owners that had few
# listings were removed as a lot of those listings wouldn't be super representative of the busiest areas.

corrdf = df.copy()
corrdf = corrdf.loc[corrdf['calculated_host_listings_count'] > 30].sort_values(by=['price'])
# corrdf = corrdf.drop_duplicates(subset=['host_id'], keep='first')
corrdf = corrdf.drop(columns=['id', 'host_id', 'latitude', 'longitude'])
corrdf['neighbourhood_group'] = corrdf['neighbourhood_group'].map({'Staten Island':0, 'Bronx':1, 'Queens':2, 'B
corrdf['room_type'] = corrdf['room_type'].map({'Shared room':0, 'Private room':1, 'Entire home/apt':2 })

dataplot = sb.heatmap(corrdf.corr(), cmap="YlGnBu", annot=True).set_title("Correlations between Interesting Lis
sb.set(rc = {'figure.figsize':(8,8)})
plt.show()
```



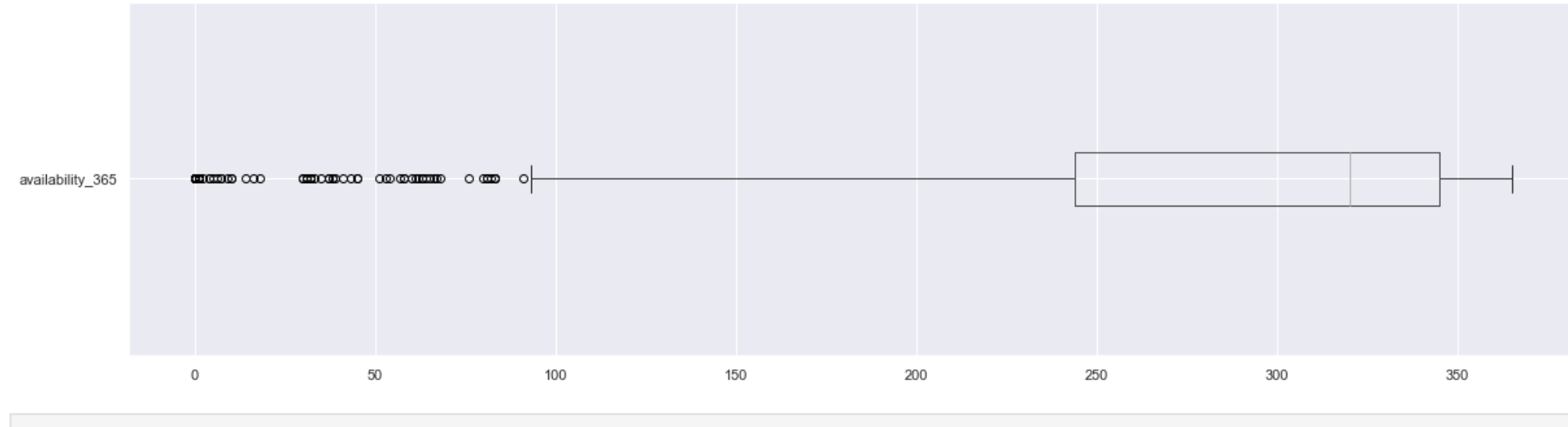
In [35]:
```
corrdf = df.copy()
corrdf = corrdf.loc[corrdf['calculated_host_listings_count'] > 30].sort_values(by=['price'])
corrdf = corrdf.drop(columns=['id', 'host_id', 'latitude', 'longitude'])
corrdf['neighbourhood_group'].value_counts(sort=False).plot.bar(y='neighbourhood_group', rot=0, title='Number o
```

Out[35]:
```
<AxesSubplot:title={'center':'Number of Listings for Busiest Hosts Based on Location'}>
```



In [36]:
```
corrdf.boxplot(column=['availability_365'], vert=False, figsize=(20,5)).set_title("Availability for Busiest Hos
```

Out[36]:
```
Text(0.5, 1.0, 'Availability for Busiest Hosts Based on Location')
```

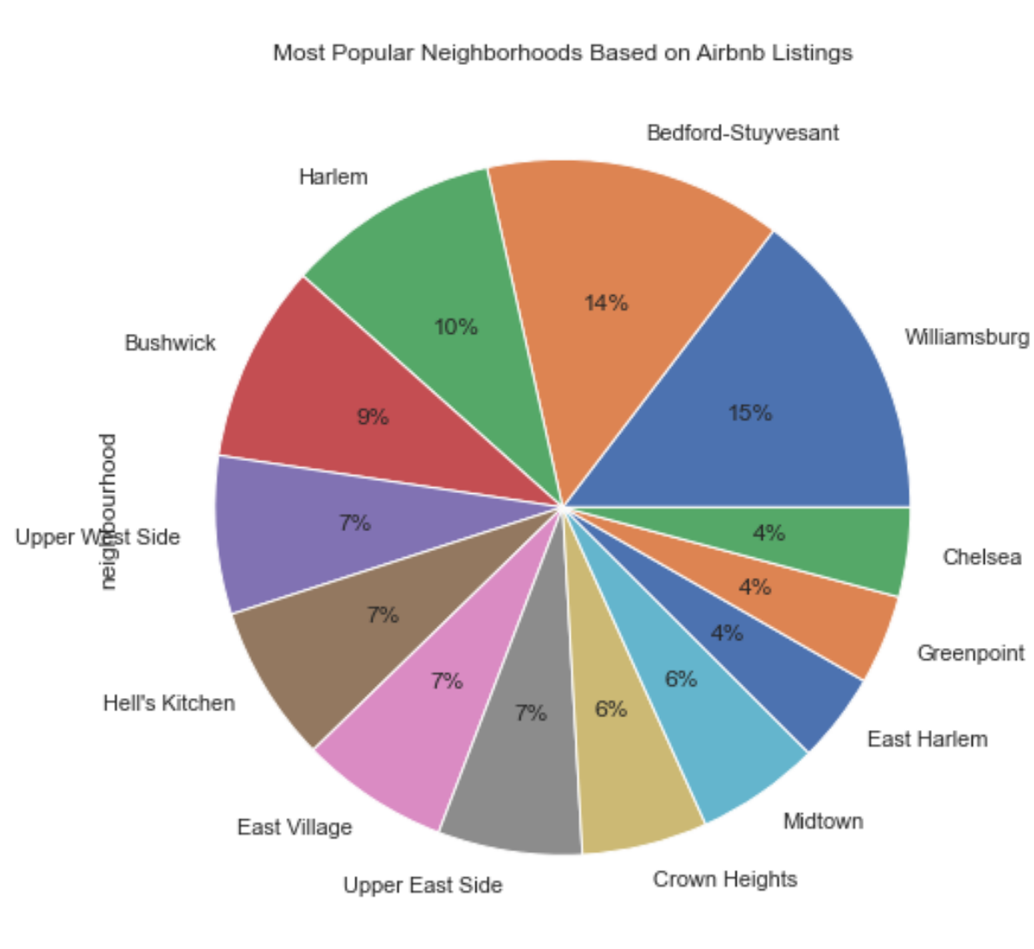


In [37]:
```
# Part 6 Analysis:
# After looking at the scatter plot and the previous scatter plot in Part 4, it seems that most of the busiest
# listings seem to be in Manhattan. This fact is later confirmed by the bar chart for the Number of Listings
# for Busiest Hosts Based on Location. When looking at the correlation analysis for the busiest hosts, there
# are some numbers that stand out:
# 1. Correlation Coefficient between Neighborhood Group and Room Type = 0.66
# 2. Calculated Host Listings Count and Reviews Per Month = 0.7
# 3. Neighborhood Group and Price = 0.4
# 4. Room Type and Price = 0.48
# The coefficient between Neighborhood Group and Room Type is interesting because when thinking about it,
# whether a listing is shared, private or the entire home/apartment does relate to the group. In the 5
# boroughs, Manhattan can have both entire apartments or private or shared rooms, while in places like
# Queens, you're probably getting the entire home/apartment.Another value that was very intriguing was the
# coefficient between Calculated Host Listings Count and Reviews per Month because it is telling us that
# with the busiest hosts, the more listings they have, the more reviews they are getting. With the last
# two values, it makes sense that the location and room type affects how costly or cheap a listing is.
```

In [38]:
```
# Task 7: Create two plots (at least one unique plot not used above) of your own using the dataset that you
# think reveals something very interesting. Explain what it is, and anything else you learned. (10 Points)
```

In [39]:
```
# The first unique plot that is created is a pie plot, that is used as a proportional representation of data
# in a column. Pie plots take in numeric data and the neighborhood column is of a different type, so the
# number of occurences were counted in order to quantify this data. Since the number of listings are well
# over over 40000 listings, I wanted to find the most popular neighborhoods and see which places have the
# most listings within those areas. I selected locations that have more than 1000 listings and saw that
# Williamsburg and Bedford-Stuyvesant, which are both in Brooklyn, were the most popular neighborhoods in
# the data set.
top5df = df['neighbourhood'].value_counts()
top5df = top5df[top5df > 1000]
top5df.plot.pie(autopct='%1.0f%%').set_title("Most Popular Neighborhoods Based on Airbnb Listings")
```

Out[39]:
```
Text(0.5, 1.0, 'Most Popular Neighborhoods Based on Airbnb Listings')
```



In [40]:
```
# Another element I wanted to reveal about the data set was the types of rooms that were in the Airbnb
# listings, based on the the location of the listing. I used a scatter plot to do so After creating the
# chart (and using a previous chart from Part 4 for the location based on the latitude and longitude),
# I was able to see that in Bronx and in Brooklyn there were more private rooms, in Queens, there was a
# mix between private rooms and entire homes/apartments, and in Manhattan there were mostly listings that
# included the entire home/apartment.
sb.scatterplot(data=df, x="latitude", y="longitude", hue="room_type").set_title("Airbnb Listings Based on Room
plt.ylim(-74.3, -73.7)
plt.xlim(40.4, 41)
```

Out[40]:
```
(40.4, 41.0)
```