Methodology Document:

Importing python libraries:

```
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import plotly.express as px

import numpy as np
```

Importing file:

```
airbnb = pd.read_csv("AB_NYC_2019.csv")

airbnb.head()
```

Analysing data and data checks:

```
airbnb.describe(percentiles=[.90,.95,.99])

airbnb.info()
```

The only two column that have significant number of nulls(~20%) are last_review and reviews_per_month. We will let those nulls remain. The rest of the dataset looks fine and does not need to be cleaned.

Airbnb Case study Analysis:

Analysis of no. of rooms for rent by minimum_nights

```
groupbyMinnights = airbnb.groupby(by=['minimum_nights']).count()

groupbyMinnights = groupbyMinnights.reset_index()

groupbyMinnights.head()
```

Code Snippet:

```
fig = px.line(groupbyMinnights,
        x='minimum_nights',
        y='id',
        width = 800, height=400)
fig.update_xaxes(range=[0, 50])
fig.update_yaxes(range=[0, 13000])
fig.update_layout(height=400, width=800,
                title_text='Number of rooms for rent vs minimum_nights',
                xaxis_title="minimum_nights",
                yaxis_title="Number of rooms for rent")
fig.show()
```

group_by minimum nights describe:

```
airbnb.groupby(by='neighbourhood_group').minimum_nights.describe()
```

Analysis of neighbourhood_group by minimum_nights:

Code Snippet:

```
groupbyNeighbourhoodMinnights =
airbnb.groupby(by=['neighbourhood_group','minimum_nights']).count()
groupbyNeighbourhoodMinnights =
groupbyNeighbourhoodMinnights.reset_index(level='minimum_nights')
groupbyNeighbourhoodMinnights.head(2)
```

Creating a list of all the neighbourhoods

```
neighbourhoodList = groupbyNeighbourhoodMinnights.index.unique().tolist()
```

Plotting the number of rooms for rent vs minimum_nights for each neighbourhood:

Code Snippet:

```
import plotly.graph_objects as go

from plotly.subplots import make_subplots

fig = make_subplots(rows=5, cols=1, start_cell="top-left",

                subplot_titles= neighbourhoodList,

#                 shared_xaxes=True,

                vertical_spacing=0.05)

for i in range (len(neighbourhoodList)):

   neighbourhood = neighbourhoodList[i]

fig.add_trace(go.Scatter(y=groupbyNeighbourhoodMinnights.loc[neighbourhood].id,

x=groupbyNeighbourhoodMinnights.loc[neighbourhood].minimum_nights,

                        name = neighbourhood),

               row=(i+1), col=1)

fig.update_layout(height=1000, width=1000,

               title_text="Different Neighbourhoods' number of rooms for rent at different minimum_nights")

fig.update_xaxes(range=[0, 35])

fig.update_yaxes(range=[0, 6000])

fig.show()
```

Price and no. of rooms variation per neighbourhood

Applying aggregation to columns:

Room prices and number of rooms for rent in each neighbourhood of Bronx

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(groupbyNeighbourhood.loc['Bronx'], lat="latitude",
lon="longitude", color="price", size="id",

                  text = groupbyNeighbourhood.loc['Bronx'].index,

              color_continuous_scale=px.colors.sequential.Plasma, size_max=20,
zoom=10)

fig.update_layout(title='Room prices and number of rooms for rent in each
neighbourhood of Bronx')

fig.update_geos(fitbounds="locations")

fig.show()
```

Room prices and number of rooms for rent in each neighbourhood of Brooklyn

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(groupbyNeighbourhood.loc['Brooklyn'], lat="latitude",
lon="longitude", color="price", size="id",
                    text = groupbyNeighbourhood.loc['Brooklyn'].index,
               color_continuous_scale=px.colors.sequential.Plasma, size_max=20,
zoom=10)
fig.update_layout(title='Room prices and number of rooms for rent in each
neighbourhood of Brooklyn')
fig.update_geos(fitbounds="locations")
fig.show()
```

Room prices and number of rooms for rent in each neighbourhood of Manhattan:

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'
px.set_mapbox_access_token(mapbox_access_token)


fig = px.scatter_mapbox(groupbyNeighbourhood.loc['Manhattan'],
lat="latitude", lon="longitude", color="price", size="id",
                    text = groupbyNeighbourhood.loc['Manhattan'].index,
               color_continuous_scale=px.colors.sequential.Plasma, size_max=20,
zoom=10)
fig.update_layout(title='Room prices and number of rooms for rent in each
neighbourhood of Manhattan')
fig.update_geos(fitbounds="locations")
fig.show()
```

Room prices and number of rooms rented in each neighbourhood of Queens

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ._nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(groupbyNeighbourhood.loc['Queens'], lat="latitude", lon="longitude", color="price", size="id",
                    text = groupbyNeighbourhood.loc['Queens'].index,
                color_continuous_scale=px.colors.sequential.Plasma, size_max=20, zoom=10)

fig.update_layout(title='Room prices and number of rooms for rent in each neighbourhood of Queens')

fig.update_geos(fitbounds="locations")

fig.show()
```

Room prices and number of rooms for rent in each neighbourhood of Staten Island:

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ._nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(groupbyNeighbourhood.loc['Staten Island'], lat="latitude", lon="longitude", color="price", size="id",
                    text = groupbyNeighbourhood.loc['Staten Island'].index,
              color_continuous_scale=px.colors.sequential.Plasma, size_max=20,
zoom=10)
fig.update_layout(title='Room prices and number of rooms for rent in each
neighbourhood of Staten Island')
fig.update_geos(fitbounds="locations")
fig.show()
```

Analysing mean price per neighbourhood in each neighbourhood_group for all neighbourhood_groups

Mean price per neighbourhood on map:

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'
px.set_mapbox_access_token(mapbox_access_token)
fig = px.scatter_mapbox(GroupbyNeighbourhoodmean,
                lat="latitude",
                lon="longitude",
                color="number_of_reviews",
                size="price",
#                 size='price',
#                 text = GroupbyNeighbourhoodmean.neighbourhood,
                hover_data = ['neighbourhood_group','neighbourhood'],
                color_continuous_scale=px.colors.sequential.Plasma,
                size_max=15,
                zoom=10)
fig.update_layout(height=600,
            title='Mean price per neighbourhood')
fig.update_geos(fitbounds="locations")
fig.show()
```

Correlation heatmap for checking correlation between quantitative variables.

Correlation of different variables

Code Snippet:

```
data =
airbnb[['price','minimum_nights','number_of_reviews','availability_365']]
plt.figure(figsize=(8,5))
fig = sns.heatmap(data.corr(), annot=True, linewidths=.5, cmap='YlGnBu')
plt.show()
```

Univariate analysis of availability_356

Code Snippet:

```
px.histogram(airbnb['availability_365'])
airbnb.loc[airbnb['availability_365'] == 0].shape
availability_365is0 = airbnb.loc[airbnb['availability_365'] == 0]
availability_365is0.head(2)
```

Plot of all listings with availability_365 = 0

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'
```

```
px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(availability_365is0,

                  lat="latitude",

                  lon="longitude",

                  color="neighbourhood_group",

                  hover_data = ['neighbourhood_group','neighbourhood'],

                  color_continuous_scale=px.colors.sequential.Plasma,

                  zoom=10)

fig.update_layout(height=650,

             mapbox_style = 'open-street-map',

             title = 'Map of all listings which are available 0 days out of 365')

fig.update_geos(fitbounds="locations")

fig.show()
```

Analysing properties price with respect to with availability_365, where availability_365 not equal to 0

```
fig = px.line(x =
airbnb.loc[airbnb['availability_365']!=0].groupby(by='availability_365').mean().i
ndex,

 y =
airbnb.loc[airbnb['availability_365']!=0].groupby(by='availability_365').mean().
price)

fig.update_layout(height=400, width=800,

             title_text='Price vs availability_365',

             xaxis_title="availability_365",

             yaxis_title="price")
```

Analysing properties number of rooms with respect to with availability_365, where availability_365 not equal to 0

Code Snippet:

```
figure = px.histogram(airbnb.loc[airbnb['availability_365']!= 0].availability_365)

figure.update_layout(height=400, width=800,

             title_text='Availability_365 vs number of rooms for rent',

             xaxis_title="Availability_365",

             yaxis_title="Number of rooms for rent")
```

Analysing number_of_reviews vs availability_365:

Code Snippet:

```
LimitedAvailability_365 = airbnb.loc[(airbnb['availability_365']<200) &
(airbnb['availability_365'] != 0)]

LimitedAvailability_365 =
LimitedAvailability_365.groupby(by=['availability_365']).mean()

LimitedAvailability_365 = LimitedAvailability_365.reset_index()

LimitedAvailability_365.head(2)
```

Creating bar plot for Availability 365 vs number of reviews

```
fig = px.bar(x=LimitedAvailability_365['availability_365'],

    y= LimitedAvailability_365['number_of_reviews'])

fig.update_layout(width = 900, height=500,

             title = 'Availability 365 vs number of reviews',

             xaxis_title="minimum_nights",

              yaxis_title='Number of reviews')

fig.show()
```

Analysing room_type

Breakup of room type by percentage

Creating a new column for percentage of each room type

Code Snippet:

```
GroupbyRoom = airbnb.groupby(by='room_type').count()

GroupbyRoom['%_of_room_type'] =
((GroupbyRoom['id']/airbnb.id.count())*100)

GroupbyRoom
```

Room type percentage breakup plot pie chart:

Code Snippet:

```
fig = px.pie(values=GroupbyRoom['%_of_room_type'],
        names=GroupbyRoom.index,
        color_discrete_sequence=px.colors.sequential.RdBu)
fig.update_layout(width = 500, height=500,
            title='Room type percentage breakup',
            margin=dict(l=2, r=2, t=40, b=2))
fig.show()
```

Room_type vs availability_365

Grouping by room type and ploting line plot for different room types by availability 365:

Code Snippet:

```
groupByRoomAvailability_365 = airbnb.groupby(by =['room_type',
'availability_365']).count().reset_index(level='availability_365')

groupByRoomAvailability_365 =
groupByRoomAvailability_365.loc[groupByRoomAvailability_365['availability_3
65'] !=0]

room_type = airbnb.room_type.unique()

fig = make_subplots(rows=3, cols=1, start_cell="top-left",

               subplot_titles= room_type,

#                 shared_xaxes=True,

               vertical_spacing=0.08)

for i in range (len(room_type)):

   room = room_type[i]

   fig.add_trace(go.Scatter(y=groupByRoomAvailability_365.loc[room].id,


x=groupByRoomAvailability_365.loc[room].availability_365,name = room),

              row=(i+1), col=1)

fig.update_layout(height=600, width=1000,

              title_text='Availability_365 vs number of rooms')

fig.update_yaxes(range=[0, 700])

fig.show()
```

Room_type vs neighbourhood_group

Code Snippet:

```
groupByRoomNeighbourhoodGroup = airbnb.groupby(by =
['neighbourhood_group', 'room_type']).agg({
    'id':'count',
    'host_id':sum,
    'price':'mean',
    'minimum_nights':sum,
    'number_of_reviews':sum,
    'reviews_per_month':sum,
    'calculated_host_listings_count':sum,
    'availability_365':sum,
    'latitude':'first',
    'longitude':'first'
})
groupByRoomNeighbourhoodGroup =
groupByRoomNeighbourhoodGroup.reset_index(level='neighbourhood_group'
)
groupByRoomNeighbourhoodGroup.head()
# Creating a column in dataset for percentage of room type in each
neighbourhood_group.head
mylist = []
for neighbourhood in neighbourhoodList:
    part2 =
(groupByRoomNeighbourhoodGroup.groupby(by='neighbourhood_group').su
m().loc[neighbourhood].id)
```

```
   part1 = 
(groupByRoomNeighbourhoodGroup.loc[groupByRoomNeighbourhoodGroup['
neighbourhood_group'] == neighbourhood].id)
   mylist.append((((part1/part2)*100)[0]))
   mylist.append((((part1/part2)*100)[1]))
   mylist.append((((part1/part2)*100)[2]))
myarray = np.asarray(mylist)
# print(myarray)
groupByRoomNeighbourhoodGroup['Perc_of_room_type_in_region'] = 
myarray
```

Changing the name of id column to Number of rooms for rent

```
groupByRoomNeighbourhoodGroup.rename({'id':'Number of rooms for rent'}, 
axis = 'columns', inplace=True)
groupByRoomNeighbourhoodGroup.head(2)
```

Number of rooms per room_type vs neighbourhood groups

Plotting clustered bar chart for number of rooms per room_type per neighbourhood_group

Code Snippet:

```
figure = px.bar(groupByRoomNeighbourhoodGroup, 
            x = 'neighbourhood_group', 
            y = 'Number of rooms for rent', 
            color = groupByRoomNeighbourhoodGroup.index, 
            barmode ='group', 
           hover_data = ['Perc_of_room_type_in_region','number_of_reviews'])
figure.update_layout(height=500, width=950,
            title_text='No of rooms per room_type per neighbourhood_group',
             xaxis_title = 'Neighbourhood Group',
             yaxis_title = 'No of rooms')
figure.show()
```

Plotting all room_types per neighbourhood on map

Spread of different room types in Brooklyn:

Code Snippet:

```
Brooklyn = airbnb.loc[airbnb['neighbourhood_group'] == 'Brooklyn']

Brooklyn.head(2)

mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

# color_discrete_map = {'Entire home/apt': 'rgb(215,48,39)', 'Private room':
'rgb(215,148,39)', 'Shared room':'rgb(215,248,39)'}

fig = px.scatter_mapbox(Brooklyn, lat="latitude", lon="longitude",
color='room_type',
#                  color_discrete_map = color_discrete_map,
                 size_max=20, zoom=10,
                opacity=0.7,
                hover_data=['neighbourhood'])
fig.update_geos(fitbounds="locations")
fig.update_layout(height=500, width=900,
title_text='Spread of different room types in Brooklyn')
fig.show()
```

Spread of different room types in Bronx:

Code Snippet:

```
Bronx = airbnb.loc[airbnb['neighbourhood_group'] == 'Bronx']

# Bronx.head(2)

mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(Bronx, lat="latitude", lon="longitude",
color='room_type',

                     size_max=20, zoom=10,

                    opacity=0.7,

                    hover_data=['neighbourhood'])

fig.update_geos(fitbounds="locations")

fig.update_layout(height=500, width=900,

               title_text='Spread of different room types in Bronx')

fig.show()
```

Spread of different room types in Manhattan:

Code Snippet:

```
Manhattan = airbnb.loc[airbnb['neighbourhood_group'] == 'Manhattan']

# Manhattan.head(2)

mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(Manhattan, lat="latitude", lon="longitude",
color='room_type',
                     size_max=20, zoom=10,
                    opacity=0.7,
                    hover_data=['neighbourhood'])
fig.update_geos(fitbounds="locations")
fig.update_layout(height=350, width=900,
               margin=dict(l=0, r=0, t=40, b=0),
               title='Spread of different room types in Manhattan')
fig.show()
```

Spread of different room types in Queens:

Code Snippet:

```
Queens = airbnb.loc[airbnb['neighbourhood_group'] == 'Queens']
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'
px.set_mapbox_access_token(mapbox_access_token)
fig = px.scatter_mapbox(Queens, lat="latitude", lon="longitude",
color='room_type',
                     size_max=20, zoom=10,
                    opacity=0.7,
                    hover_data=['neighbourhood'])
fig.update_geos(fitbounds="locations")
fig.update_layout(height=500, width=900,
               title_text='Spread of different room types in Queens')
fig.show()
```

Spread of different room types in Staten Island:

Code Snippet:

```
StatenIsland = airbnb.loc[airbnb['neighbourhood_group'] == 'Staten Island']

# Manhattan.head(2)

mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'

px.set_mapbox_access_token(mapbox_access_token)

fig = px.scatter_mapbox(StatenIsland, lat="latitude", lon="longitude",
color='room_type',

                     size_max=20, zoom=10,

                    opacity=0.7,

                    hover_data=['neighbourhood'])

fig.update_geos(fitbounds="locations")

fig.update_layout(height=500, width=900,

                title_text='Spread of different room types in Staten Island')

fig.show()
```

Price by room_type:

Grouping room type and price and aggregating different columns:

Grouping room type by price

Code Snippet:

```
groupByRoomPrice = airbnb.groupby(by =['room_type', 'price']).agg({
    'id':'count',
    'host_id':sum,
    'minimum_nights':sum,
    'number_of_reviews':sum,
    'reviews_per_month':sum,
    'calculated_host_listings_count':sum,
    'availability_365':sum,
    'latitude':'first',
    'longitude':'first'
})
groupByRoomPrice = groupByRoomPrice.reset_index(level='price')
groupByRoomPrice = groupByRoomPrice.loc[groupByRoomPrice['price'] !=0]
groupByRoomPrice.head(3)
```

Initialising figure

Plotting box plots price per room type:

Code Snippet:

```
fig = go.Figure()
df_room_type = airbnb[airbnb['price']<2000]
#Add Traces
fig.add_trace(
   go.Box(y=list(df_room_type[df_room_type['room_type'] == 'Entire home/apt']['price'].reindex()),
            name="Entire home/apt",
            line=dict(color="#33CFA5")))
fig.add_trace(
   go.Box(y=list(df_room_type[df_room_type['room_type'] == 'Private room']['price'].reindex()),
            name="Private room",
            line=dict(color="#F06A6A")))
fig.add_trace(
   go.Box(y=list(df_room_type[df_room_type['room_type'] == 'Shared room']['price'].reindex()),
            name="Shared room",
            line=dict(color="#AA0DFE")))
#Set title
fig.update_layout(title_text="Price Distribution by Room Type",xaxis_title="Room Type")
fig.show()
```

Price vs number of rooms for each room type

Plotting line chart for price vs different room types

```
# Defining three subplots starting with top left subplot
```

Code Snippet:

```
fig = make_subplots(rows=3, cols=1, start_cell="top-left",
                subplot_titles= room_type,
#                 shared_xaxes=True,
                vertical_spacing=0.08)


# For each room type in df, plot number of rooms for rent vs price for that type
of room
for i in range (len(room_type)):
   room = room_type[i]
   fig.add_trace(go.Scatter(y=groupByRoomPrice.loc[room].id,
                         x=groupByRoomPrice.loc[room].price,
                        name = room),
               row=(i+1), col=1)
fig.update_layout(height=900, width=1000,
               title_text='Price vs number of rooms')
# Limiting the x_axis range to avoid the main plot becoming small and
incoherent because of outliers in data
fig.update_yaxes(range=[0, 1500])
fig.update_xaxes(range=[0, 600])


fig.show()
```

Plotting violin chart for number of reviews vs roomtype:

Code Snippet:

```
fig = px.violin(x=groupByRoomPrice.index,
          y=groupByRoomPrice['number_of_reviews'])
fig.update_layout(title='Number of reviews vs room type')
fig.show()
```

Price vs number of reviews

Scatter plot for price vs number of reviews

Code Snippet:

```
fig = px.scatter(x=airbnb['price'], y=airbnb['number_of_reviews'], width = 900,
height=500)
fig.update_layout(
    title="Price vs Number of Review",
    xaxis_title="Price",
    yaxis_title="Number of Reviews",
    margin=dict(l=0, r=0, t=40, b=0)
)
fig.show()
```

Number of reviews by neighbourhood

Code Snippet:

```
GroupbyReviews =
airbnb.groupby(by=['neighbourhood_group','neighbourhood']).agg({
   'id':'count',
   'host_id':sum,
   'price':sum,
   'minimum_nights':sum,
   'number_of_reviews':sum,
   'reviews_per_month':sum,
   'calculated_host_listings_count':sum,
   'availability_365':sum,
   'latitude':'first',
   'longitude':'first'
})
GroupbyReviews = GroupbyReviews.reset_index()
GroupbyReviews.head()
```

Number of reviews per neighbourhood

Map for number of reviews per neighbourhood

Code Snippet:

```
mapbox_access_token =
'pk.eyJ1IjoiYWxha2FnIiwiYSI6ImNra29kOHJlaDAyMmYydW56YmsyeWhxbGIifQ.
_nGzLIEWkrvGKJVeRhizyA'
px.set_mapbox_access_token(mapbox_access_token)
fig = px.scatter_mapbox(GroupbyReviews,
                  lat="latitude",
                  lon="longitude",
                  color="number_of_reviews",
                  size="number_of_reviews",
                  text = GroupbyReviews.neighbourhood,
                  hover_data = ['neighbourhood_group','neighbourhood'],
                  color_continuous_scale=px.colors.sequential.Plasma,
                  size_max=20,
                  zoom=10)

fig.update_layout(margin=dict(l=10, r=10, t=40, b=10),
             title = 'Neighbourhoods by number of reviews')
fig.update_geos(fitbounds="locations")
fig.show()
```