In [3]:
```
# importing the required libray.
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [4]:
```
# reading the provided data set.
df =pd.read_csv('test.csv')
```

In [5]:
```
# reading first 10 rows from data set.
df.head(10)
```

Out[5]:

	key	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	drop
0	2015-01-27 13:08:24.0000002	2015-01-27 13:08:24 UTC	-73.973320	40.763805	-73.981430	
1	2015-01-27 13:08:24.0000003	2015-01-27 13:08:24 UTC	-73.986862	40.719383	-73.998886	
2	2011-10-08 11:53:44.0000002	2011-10-08 11:53:44 UTC	-73.982524	40.751260	-73.979654	
3	2012-12-01 21:12:12.0000002	2012-12-01 21:12:12 UTC	-73.981160	40.767807	-73.990448	
4	2012-12-01 21:12:12.0000003	2012-12-01 21:12:12 UTC	-73.966046	40.789775	-73.988565	
5	2012-12-01 21:12:12.0000005	2012-12-01 21:12:12 UTC	-73.960983	40.765547	-73.979177	
6	2011-10-06 12:10:20.0000001	2011-10-06 12:10:20 UTC	-73.949013	40.773204	-73.959622	
7	2011-10-06 12:10:20.0000003	2011-10-06 12:10:20 UTC	-73.777282	40.646636	-73.985083	
8	2011-10-06 12:10:20.0000002	2011-10-06 12:10:20 UTC	-74.014099	40.709638	-73.995106	
9	2014-02-18 15:22:20.0000002	2014-02-18 15:22:20 UTC	-73.969582	40.765519	-73.980686	

In [6]:
```
# viewing the information about dataset.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9914 entries, 0 to 9913
Data columns (total 7 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   key                9914 non-null   object 
 1   pickup_datetime    9914 non-null   object 
 2   pickup_longitude   9914 non-null   float64
 3   pickup_latitude    9914 non-null   float64
 4   dropoff_longitude  9914 non-null   float64
 5   dropoff_latitude   9914 non-null   float64
 6   passenger_count    9914 non-null   int64  
dtypes: float64(4), int64(1), object(2)
memory usage: 542.3+ KB
```

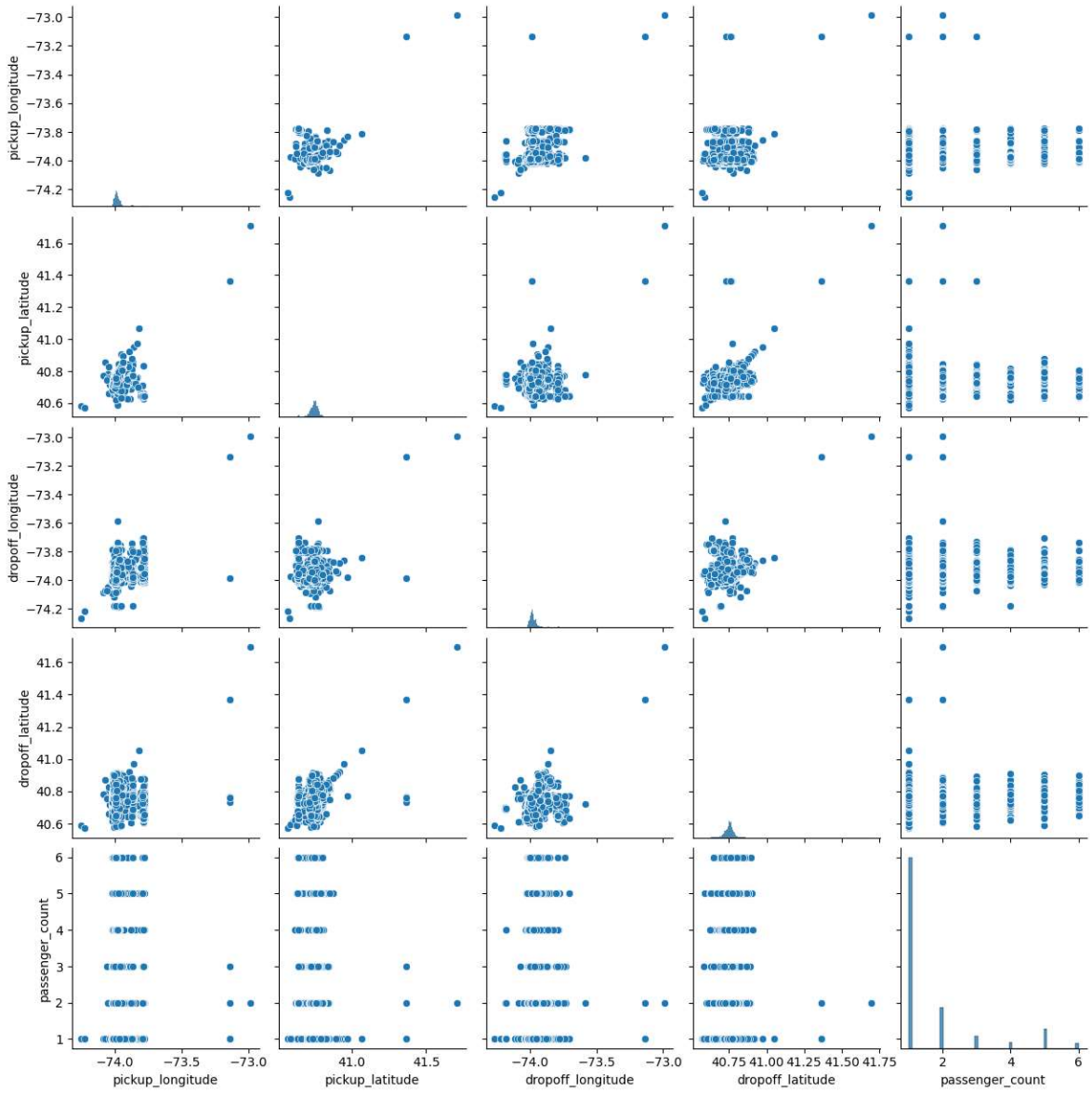
In [7]:
```
# describing the stats of the dataset.
df.describe()
```

Out[7]:

	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passenger_count
count	9914.000000	9914.000000	9914.000000	9914.000000	9914.000000
mean	-73.974722	40.751041	-73.973657	40.751743	1.671273
std	0.042774	0.033541	0.039072	0.035435	1.278747
min	-74.252193	40.573143	-74.263242	40.568973	1.000000
25%	-73.992501	40.736125	-73.991247	40.735254	1.000000
50%	-73.982326	40.753051	-73.980015	40.754065	1.000000
75%	-73.968013	40.767113	-73.964059	40.768757	2.000000
max	-72.986532	41.709555	-72.990963	41.696683	6.000000

In [8]:
```
sns.pairplot(df)
plt.figure(figsize=(18,5))
```

Out[8]: <Figure size 1800x500 with 0 Axes>



<Figure size 1800x500 with 0 Axes>

.

1. What is the data type of each column in the dataset, and how might the data types affect the analysis and visualization of the data?

In [7]:
```
df.dtypes
```

Out[7]:
```
key                  object
pickup_datetime      object
pickup_longitude    float64
pickup_latitude     float64
dropoff_longitude   float64
dropoff_latitude    float64
passenger_count       int64
dtype: object
```

Data type effect the analysis and visualization in various way like

- numerical data type used to perform statistical calculation.
- categorical data type is used to perform grouping of data . Example : Gender Male/Female.
- Date and Time is used to understand the time trends of data and understand the temporal analysis.
- in case of object data type we need to perform data cleaning and handle missing values so that data will become valid for analyis and visualization.

2. How many rows are in the dataset, and how might the size of the dataset affect the efficiency and accuracy of the analysis?

In [8]:
```
# it iwll give the total no. of rows available in the data set.
len(df)
```

Out[8]:
```
9914
```

In [9]:
```
# it will give the shape of the data set in format of  (rows, Columns)
df.shape
```

Out[9]:
```
(9914, 7)
```

In [10]:
```
# It will check is there any null vlaues in the dataset.
df.isnull()
```

Out[10]:

	key	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitu
0	False	False	False	False	False	Fa
1	False	False	False	False	False	Fa
2	False	False	False	False	False	Fa
3	False	False	False	False	False	Fa
4	False	False	False	False	False	Fa
...	...	...	...	...	...	
9909	False	False	False	False	False	Fa
9910	False	False	False	False	False	Fa
9911	False	False	False	False	False	Fa
9912	False	False	False	False	False	Fa
9913	False	False	False	False	False	Fa

9914 rows × 7 columns

In [11]:
```
df.isnull().value_counts()  # it will count the values.
```

Out[11]:
```
key    pickup_datetime  pickup_longitude  pickup_latitude  dropoff_longitude
dropoff_latitude  passenger_count
False  False            False             False            False
False             False              9914
dtype: int64
```

Size of the data set affects the analysis and visualization

- If the data set size if large then it will give the more accurate result and also having a drwaback like we have to perform data cleaning before processing that dataset.
- As the data size increases the computational cost is also increases and take more time to visualize and analysi.

3. How many unique values are there in the 'passenger_count' column, and what insights might this provide about the user behavior or demand patterns?

In [20]:
```
# it will find the unique value in the passanger count column.
uniq=df['passenger_count'].unique()
len(uniq)
```

Out[20]:
```
6
```

It will provides information about the behaviour of the passanger and the demand on basis of

- location
- Date and time

- it will helps in decisin making and undestand the passangers behaviour so the in future new services can be provided and proved the poor services.

In [13]:
```
a=df['passenger_count'].unique()
a.view()
```

Out[13]:
```
array([1, 2, 3, 4, 5, 6], dtype=int64)
```

4. What is the maximum longitude value in the dataset, and how might this value affect the spatial analysis or visualization of the data?

In [14]:
```
# It will give the maximumn pickup longtitude value.
df['pickup_longitude'].max()
```

Out[14]:
```
-72.986532
```

In [15]:
```
# It will give the maximum dropoff longitude vlaue.
df['dropoff_longitude'].max()
```

Out[15]:
```
-72.990963
```

In [16]:
```
# It will both pickup and dropoff longtitude value.
df[['pickup_longitude','dropoff_longitude']].max()
```

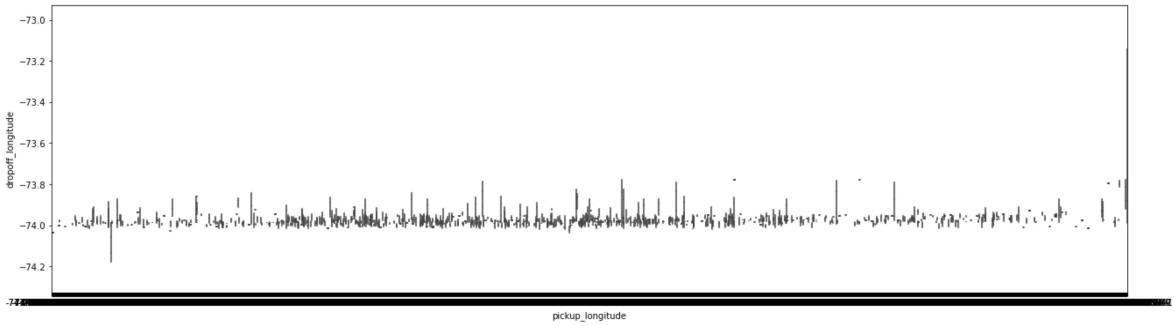
Out[16]:
```
pickup_longitude    -72.986532
dropoff_longitude   -72.990963
dtype: float64
```

spetial analysis helps to understand the human behaviour on the basis of geographical analyis / distance.

- It will help to undestand the service availabilty area how far able to provide the serivces. and find out the nearby area for serivces in future.
- It will helps to understand geographical coverage in better way and figure out the outliers area for services.

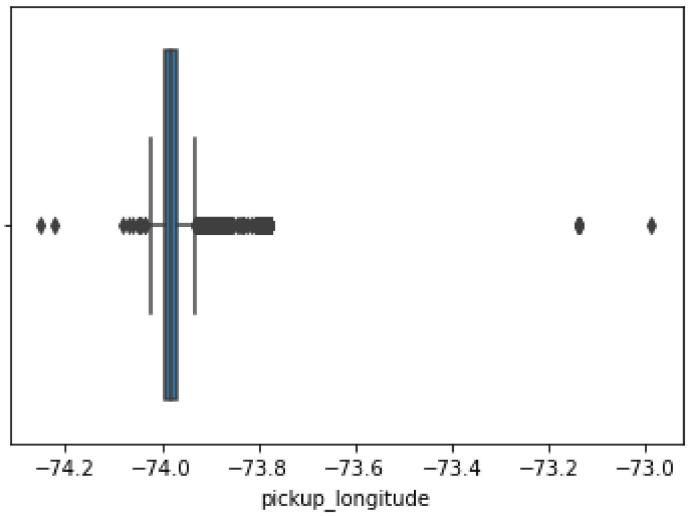
In [17]:
```
plt.figure(figsize=(22,6))
sns.boxplot(x=df['pickup_longitude'],y=df['dropoff_longitude'],fliersize=5,wi
```

Out[17]: <AxesSubplot:xlabel='pickup_longitude', ylabel='dropoff_longitude'>



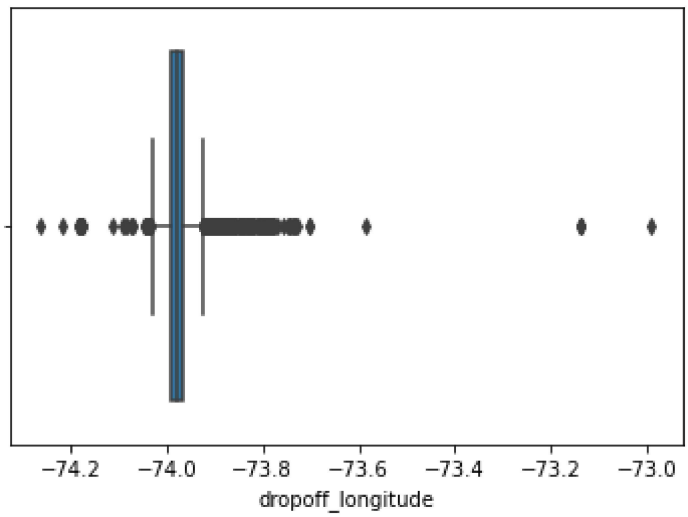
In [18]:
```
sns.boxplot(x=df['pickup_longitude'])
```

Out[18]: <AxesSubplot:xlabel='pickup_longitude'>



In [19]:
```
sns.boxplot(x=df['dropoff_longitude'])
```

Out[19]: <AxesSubplot:xlabel='dropoff_longitude'>



In [20]:
```
# It will give the maximum vlaue from each of the column.
df.max()
```

Out[20]:
```
key                  2015-06-30 20:03:50.0000005
pickup_datetime          2015-06-30 20:03:50 UTC
pickup_longitude                       -72.986532
pickup_latitude                         41.709555
dropoff_longitude                      -72.990963
dropoff_latitude                        41.696683
passenger_count                                 6
dtype: object
```

5. What is the average latitude value in the dataset, and how might this value provide insights into the spatial distribution or density of the pickups and dropoffs?

In [21]:
```
df['pickup_latitude'].mean()
```

Out[21]:
```
40.75104072349318
```

In [22]:
```
df['dropoff_latitude'].mean()
```

Out[22]:
```
40.75174278102696
```

In [23]:
```
# give the average pickup and dropoff latitude.
df[['pickup_latitude','dropoff_latitude']].mean()
```

Out[23]:
```
pickup_latitude     40.751041
dropoff_latitude    40.751743
dtype: float64
```

In [24]:
```
# it will give the average pickup and dropoff latitude upto 2 decimal value.
df[['pickup_latitude','dropoff_latitude']].mean().round(2)
```

Out[24]:
```
pickup_latitude     40.75
dropoff_latitude    40.75
dtype: float64
```

It helps in decison making and around areas to target for marketing and service expansion.

- It will helps to figure out the central loaction from where maximum no. of passingers can use the service.
- It will also helps to understand the comparison of different location spatial distribution or density of the pickups and dropoffs.

6. What is the pickup_datetime of the first row in the dataset, and how might the timestamp format affect the temporal analysis or visualization of the data?

In [25]:
```
df['pickup_datetime'].head(1)
```

Out[25]:
```
0    2015-01-27 13:08:24 UTC
Name: pickup_datetime, dtype: object
```

In [22]:
```
df['pickup_datetime'][0]
```

Out[22]:
```
'2015-01-27 13:08:24 UTC'
```

Temporal analysis is use to analyse multiple data point over time (ie. month , day, year, hour, minute , second etc)

- Timestamp formt is used to find the "time Range","time series" of the data.
- It also helps to understand real world problem in better way. Example stock market.

7. How many rows have a passenger_count greater than 1, and what insights might this provide about the user behavior or demand patterns?

In [26]:
```
# it will count the vlaues number.
df['passenger_count'].value_counts()
```

Out[26]:
```
1    6914
2    1474
5     696
3     447
4     206
6     177
Name: passenger_count, dtype: int64
```

In [27]:
```
# # rows having a passanger count greater than 1.
# df['passenger_count'][df['passenger_count']>1].count()
v=df['passenger_count']>1
#print(df[v].count())
print(df['passenger_count'][v].count())
```

```
3000
```

In [28]:
```
# rows having a passanger count equal to 1 so that we can verify the answer.
df['passenger_count'][df['passenger_count']==1].count()
```

Out[28]:
```
6914
```

It will helps to understand the passanger behaviour like grop size, pasanger behaviour at peak time.

- It will help to take better decison and imporove the efficiency of ride sharing service.

8. What is the range of values in the pickup_latitude column, and how might this value affect the spatial analysis or visualization of the data?

In [14]:
```
# find lower value of pickup lattitude.
lower_value =df['pickup_latitude'].min()
lower_value
```

Out[14]:
```
40.573143
```

In [15]:
```
# find maximum vlaue of pickup lattitude.
higher_value=df['pickup_latitude'].max()
higher_value
```

Out[15]:
```
41.709555
```

In [16]:
```
# range of values in the pickup_latitude column
print("range : ",lower_value-higher_value)
```

```
range :  -1.136412
```

The range of values in the pickup lattitude cloumn helps to understand the geographical area in beeter way also helps to understand better way on maps and improve services in that geographical area.

9. How many rows have missing values, and how might the missing data affect the accuracy and reliability of the analysis?

In [32]:
```
# it will find not a number value.
df.isna().sum()
```

Out[32]:
```
key                0
pickup_datetime    0
pickup_longitude   0
pickup_latitude    0
dropoff_longitude  0
dropoff_latitude   0
passenger_count    0
dtype: int64
```

In [33]:
```
# it will find null values.
df.isnull().sum()
```

Out[33]:
```
key                  0
pickup_datetime      0
pickup_longitude     0
pickup_latitude      0
dropoff_longitude    0
dropoff_latitude     0
passenger_count      0
dtype: int64
```

- There is no missing row.
- It impact the accuarcy of the result and also reduced the sample size.
- Can mislead the consclusion of data set which can impact in business decisions.

10. What is the correlation between pickup_longitude and dropoff_longitude, and how might this correlation provide insights into the spatial relationships or patterns of the pickups and dropoffs?

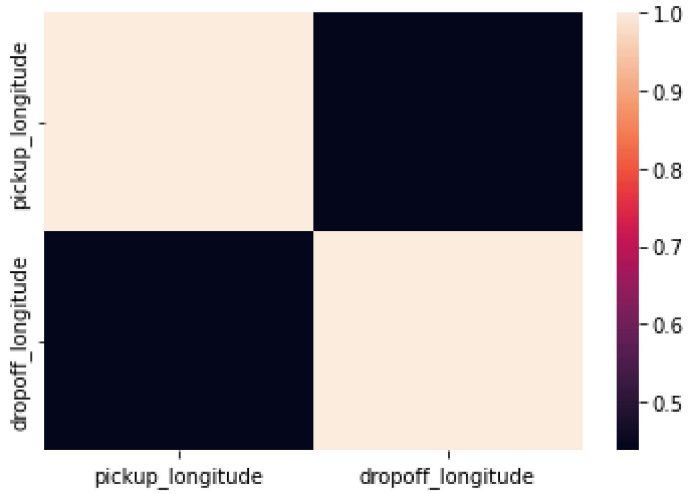
In [34]:
```
df[['pickup_longitude','dropoff_longitude']].corr()
```

Out[34]:

	pickup_longitude	dropoff_longitude
pickup_longitude	1.000000	0.437949
dropoff_longitude	0.437949	1.000000

In [35]:
```
# Heatmap of pickup longitude and dropoff longitude
sns.heatmap(df[['pickup_longitude','dropoff_longitude']].corr())
```

Out[35]:
```
<AxesSubplot:>
```



- Correlation defines the statistical linear realtionship between two variables.

- It will helps to understand the realtionship between picup and drop longitute and also helps to understand most busiest longitude and trends line so that help to take decison on

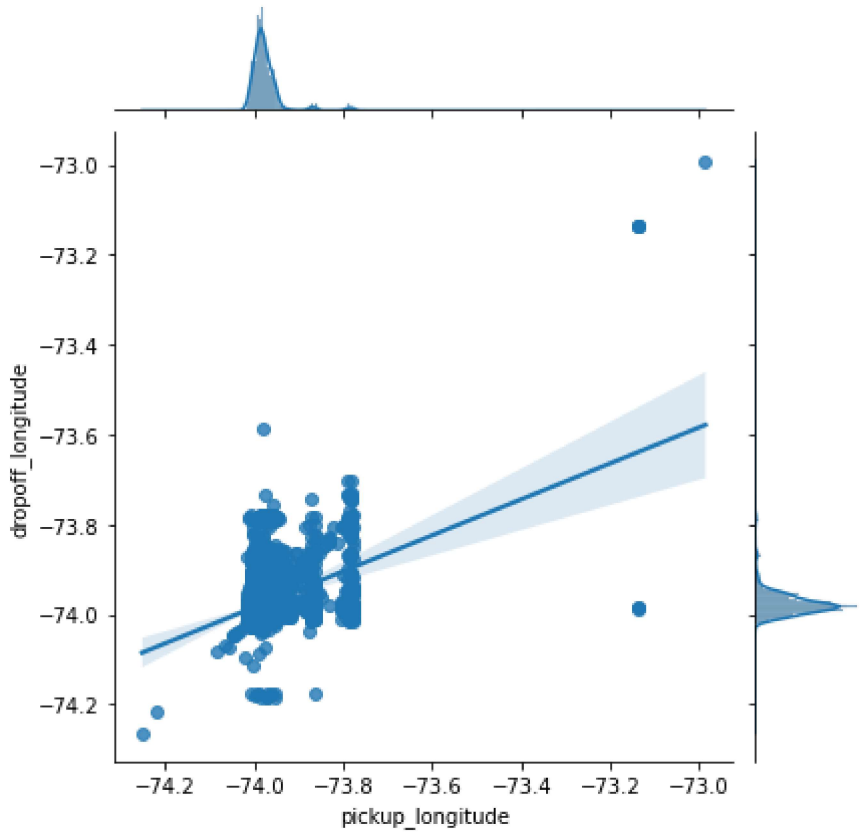
In [36]:
```
df.corr()
```

Out[36]:

	pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	passeng
pickup_longitude	1.000000	0.284509	0.437949	0.246595	
pickup_latitude	0.284509	1.000000	0.274527	0.520171	
dropoff_longitude	0.437949	0.274527	1.000000	0.344918	
dropoff_latitude	0.246595	0.520171	0.344918	1.000000	
passenger_count	0.006398	-0.025120	0.012389	0.005606	

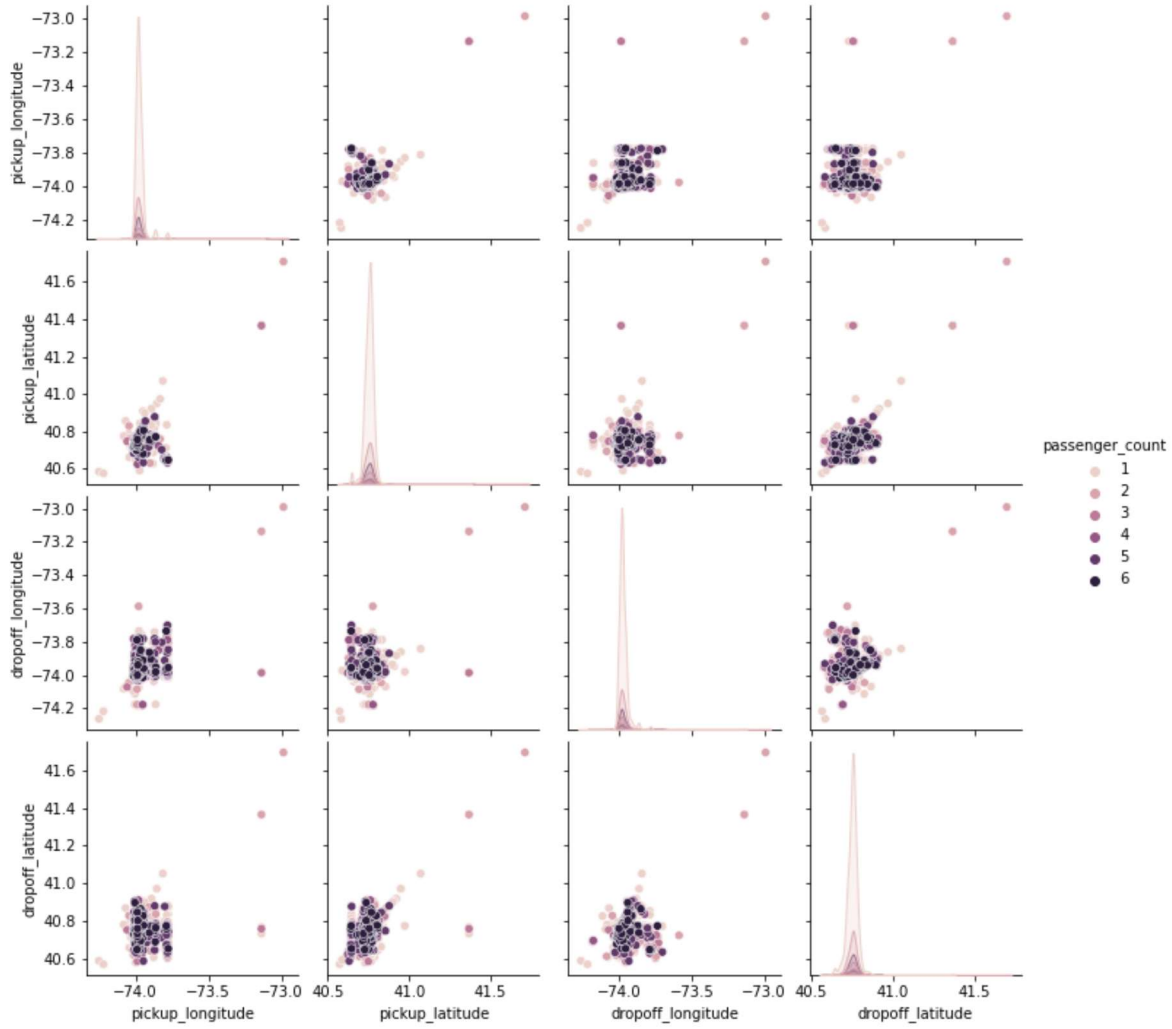
In [37]:
```
sns.jointplot(x=df['pickup_longitude'],y=df['dropoff_longitude'],kind='reg')
```

Out[37]:
```
<seaborn.axisgrid.JointGrid at 0x2dbdecc9550>
```



In [38]:
```
sns.pairplot(df,hue='passenger_count')
```

Out[38]:
```
<seaborn.axisgrid.PairGrid at 0x2dbdeccbeb0>
```



11. What is the median passenger_count in the dataset, and how might this value provide insights into the user behavior or demand patterns?

In [39]:
```
# median of passanger count in data set.
df['passenger_count'].median()
```

Out[39]:
```
1.0
```

- It helps to understand the user behaviour and demand patterns in following ways:
  - Group size of the passanger so that provides services on tagert passenger category.
  - User preference so that make availibility of products.

In [40]:
```
# it will find the of whole data sets which column has numberical data type.
df.median()
```

```
C:\Users\conec\AppData\Local\Temp/ipykernel_3064/2606440367.py:2: FutureWarn
ing: Dropping of nuisance columns in DataFrame reductions (with 'numeric_onl
y=None') is deprecated; in a future version this will raise TypeError.  Sele
ct only valid columns before calling the reduction.
  df.median()
```

Out[40]:
```
pickup_longitude    -73.982326
pickup_latitude      40.753051
dropoff_longitude   -73.980015
dropoff_latitude     40.754065
passenger_count       1.000000
dtype: float64
```

12. How many rows have pickup_latitude values greater than 41, and what insights might this provide about the spatial distribution or density of the pickups and dropoffs?

In [41]:
```
# it will give the vlaues of pickup lattitude greater than 41.
df['pickup_latitude'][df['pickup_latitude']>41].count()
```

Out[41]:
```
9
```

In [42]:
```
# it will count each values which is more than 41.
a=df['pickup_latitude'][df['pickup_latitude']>41].value_counts()
a
```

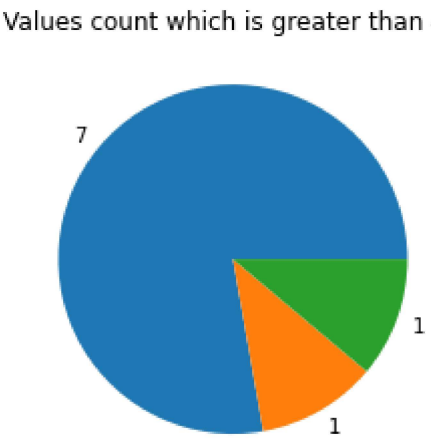
Out[42]:
```
41.366138    7
41.069660    1
41.709555    1
Name: pickup_latitude, dtype: int64
```

- it will help to understand pickup lattitude which value is grater than 41 so that take decison on that gerograpical area. hera we can see that only 9 rows are availbe out of total rows (i.e 9914) we can consider this as outliers point.

In [43]:
```
# pie shart to understad the
a=df['pickup_latitude'][df['pickup_latitude']>41].value_counts()
print(a)
plt.pie(a,labels=a)
plt.title("Values count which is greater than 41")
```

```
41.366138    7
41.069660    1
41.709555    1
Name: pickup_latitude, dtype: int64
```

Out[43]:
```
Text(0.5, 1.0, 'Values count which is greater than 41')
```



14. How many rows have pickup_longitude values between -74 and -73.5, and how might this spatial filter affect the analysis or visualization of the data?

In [44]:
```
# find the number of rows have pickup longitude values betweeb -74 and -73.5
a = df['pickup_longitude'][df['pickup_longitude'].between(-74, -73.5)]
```

In [45]:
```
# viwing data in sorted order.
a.sort_values()
```

Out[45]:
```
5694   -73.999979
8439   -73.999977
6349   -73.999974
8717   -73.999966
422    -73.999935
          ...
8944   -73.776672
3674   -73.776664
5940   -73.776654
7026   -73.776638
4270   -73.776585
Name: pickup_longitude, Length: 8522, dtype: float64
```

In [46]:
```
print("Rows have pickup_longitude values between -74 and -73.5 :",a.count())
```

```
Rows have pickup_longitude values between -74 and -73.5 : 8522
```

- It will help to filter the pickup longitud of geographicall area to understand in batter way.
- With the help of visualization we can view filter area very quickly.

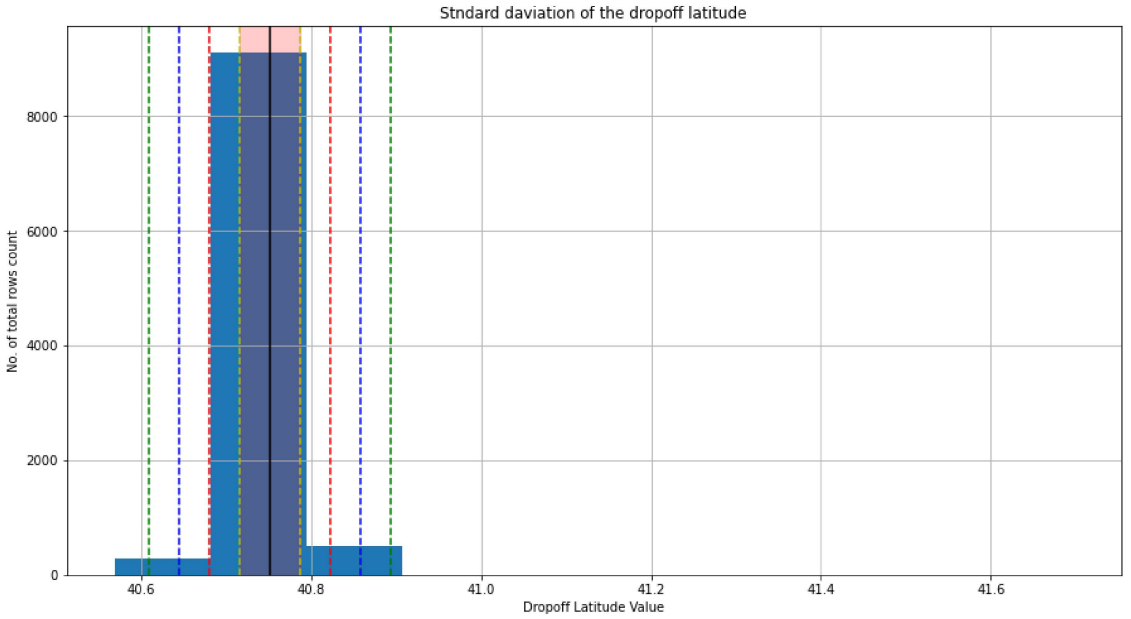
15. What is the standard deviation of the dropoff_latitude column, and how might this value provide insights into the spatial distribution or variation of the dropoff locations?

In [47]:
```
std=df['dropoff_latitude'].std()
std
```

Out[47]:
```
0.03543521142914974
```

In [62]:
```
plt.figure(figsize=(15,8))
df['dropoff_latitude'].hist()
plt.xlabel("Dropoff Latitude Value")
plt.ylabel("No. of total rows count")
plt.title("Stndard daviation of the dropoff latitude")

x=df['dropoff_latitude'].mean()

# 1std below the mean =  (mean-std)
# 1std above the mean =  (mean+std)
# 2std below the mean =  (mean-std*2)
# 2std above the mean =  (mean+std*2)

plt.axvline(x, color='k', linestyle='-')
plt.axvline(x - std, color='y', linestyle='--')
plt.axvline(x + std, color='y', linestyle='--')

plt.axvline(x - std*2, color='r', linestyle='--')
plt.axvline(x + std*2, color='r', linestyle='--')

plt.axvline(x - std*3, color='b', linestyle='--')
plt.axvline(x + std*3, color='b', linestyle='--')

plt.axvline(x - std*4, color='g', linestyle='--')
plt.axvline(x + std*4, color='g', linestyle='--')

plt.axvspan(x - std, x + std ,alpha=0.2, color='red')
```

Out[62]: <matplotlib.patches.Polygon at 0x2dbdea92f10>



- Standard deviation measure how data point is related to mean value of that data point.
- Standar deviation indicates how much the dropoff locations are spread out from the mean dropoff latitude value.
- here standard deviation value is very low ,it means that it will be very close to the mean value of the dropoff lattitude.
-

16. How many rows have passenger_count equal to 0, and what insights might this provide about the user behavior or data quality?

In [49]:
```
df['passenger_count'][df['passenger_count']==0]
```

Out[49]:
```
Series([], Name: passenger_count, dtype: int64)
```

- There is no row havivng passengar count equal to 0.
  - it provies insite user behavoir or data quality is any vlaue is misson or incorrect.
  - Here we can see that there is not any vlaue it means that there is no missing, error vlaues in the given data set.

17. What is the difference between the maximum and minimum pickup_longitude values, and how might this value affect the spatial analysis or visualization of the data?

In [50]:
```
# it will find the maximum pickup longitude vlaue
maximum=df['pickup_longitude'].max()
maximum
```

Out[50]:
```
-72.986532
```

In [51]:
```
# it will find the minimum pickup longitude value.
minimum=df['pickup_longitude'].min()
minimum
```

Out[51]:
```
-74.252193
```

In [52]:
```
# it will find difference between the maximum and minimum pickup longitude vl
difference=maximum-minimum
difference
```

Out[52]:
```
1.2656610000000086
```

- It will find the total span spread area of pickup longitude in the given data set.
- Large value denotes that it cover large geographical area.
  - But in given dataset it's vlaues is small so it will cover small pickup longitude area.

18. How many rows have pickup_longitude values less than -74.1 or greater than -73.9, and how might this spatial filter affect the analysis or visualization of the data?

In [53]:
```
# for verification purpose.
filt=(df['pickup_longitude']< (-74.1))
print("Less than -74.1 :",filt.count())

filt1=(df['pickup_longitude'] > (-73.9))
print("Greater than -73.9 :",filt1.count())

print("max pickup longtitude :",df['pickup_longitude'].max())
print("Min pickup longtitude :",df['pickup_longitude'].min())
```

```
Less than -74.1 : 9914
Greater than -73.9 : 9914
max pickup longtitude : -72.986532
Min pickup longtitude : -74.252193
```

In [54]:
```
#rows have pickup_longitude values less than -74.1 or greater than -73.9.
filt=(df['pickup_longitude']< -74.1) & (df['pickup_longitude'] > -73.9)
filt.count()
```

Out[54]:
```
9914
```

In [55]:
```
filtered_df = df[(df['pickup_longitude'] < -74.1) | (df['pickup_longitude'] >
num_rows = filtered_df.shape[0]
print(num_rows)
```

```
414
```

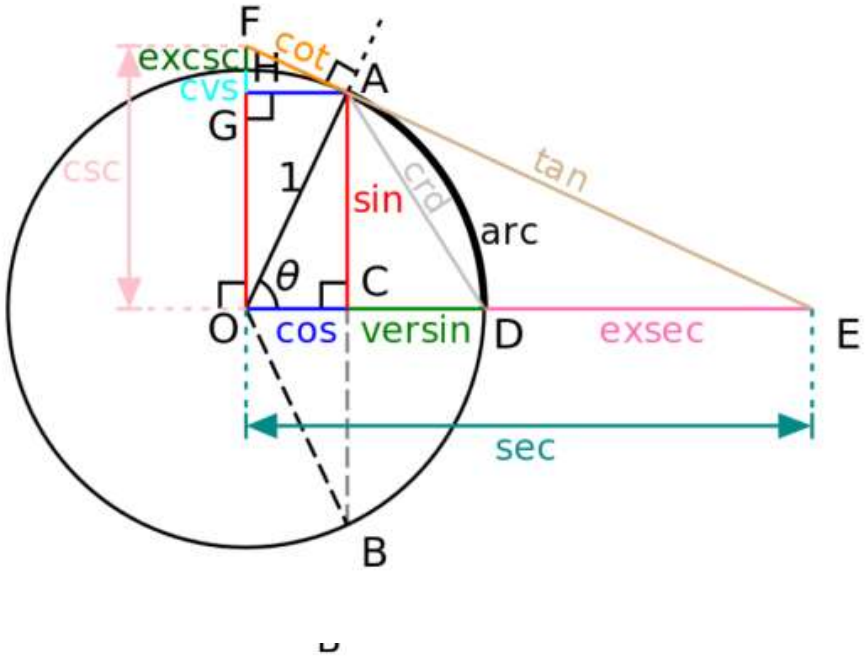
In [56]:
```
a = df['pickup_longitude'][df['pickup_longitude'].between(-74.1, -73.9)]
a.count()
```

Out[56]:
```
9500
```

specific range can provide insights into the spatial distribution of the pickups in the dataset.

19. What is the average distance (in miles) between pickup and dropoff locations, and how might this value provide insights into the spatial demand patterns or operational efficiency of the taxi service?

These function names have a simple naming pattern and in this example, the "Ha" in "Haversine" stands for "half versed sine" where haversin(θ) = versin(θ)/2.



Haversine Formula

The Haversine formula is perhaps the first equation to consider when understanding how to calculate distances on a sphere. The word "Haversine" comes from the function:

> $\text{haversine}(\theta) = \sin^2(\theta/2)$

The following equation where ϕ is latitude, λ is longitude, R is earth's radius (mean radius = 6,371km) is how we translate the above formula to include latitude and longitude coordinates. Note that angles need to be in radians to pass to trig functions:

> $a = \sin^2(\phi B - \phi A/2) + \cos \phi A * \cos \phi B * \sin^2(\lambda B - \lambda A/2)$
> $c = 2 * \text{atan2}(\sqrt{a}, \sqrt{(1-a)})$
> $d = R \cdot c$

In [9]:
```
import radians, sin, cos, sqrt, atan2
nce(lat1, lon1, lat2, lon2):
958.8 # earth radius in miles
 lon1, lat2, lon2 = map(radians, [lat1, lon1, lat2, lon2])
= lat2 - lat1
= lon2 - lon1
in(dlat / 2)**2 + cos(lat1) * cos(lat2) * sin(dlon / 2)**2
 * atan2(sqrt(a), sqrt(1 - a))
n (R * c)

nce_in_miles'] = df.apply(lambda row: distance(row['pickup_latitude'], row['pi
nce = df['distance_in_miles'].mean()

erage distance between pickup and dropoff locations: {:.2f} miles".format(avg_
```

```
Average distance between pickup and dropoff locations: 2.13 miles
```

- This value can provide valuable insights into the spatial demand patterns and operational efficiency of the service.
- lower average distance suggest that the service is more commonly used for short trips in geographical area.
- Used to monitor the changes in demand.

In [10]:
```
df['distance_in_miles'].max() # maximum ditance
```

Out[10]:
```
62.135351517555456
```

In [11]:
```
df['passenger_count'].max() # maximum passangers
```

Out[11]:
```
6
```

In [12]:
```
df['pickup_longitude'].max()
```

Out[12]:
```
-72.986532
```

20. How many rows have a pickup_datetime in January 2015, and how might this temporal filter affect the analysis or visualization of the data?

In [58]:
```
df.head(5)
```

Out[58]:

	key	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	drop
0	2015-01-27 13:08:24.0000002	2015-01-27 13:08:24 UTC	-73.973320	40.763805	-73.981430	
1	2015-01-27 13:08:24.0000003	2015-01-27 13:08:24 UTC	-73.986862	40.719383	-73.998886	
2	2011-10-08 11:53:44.0000002	2011-10-08 11:53:44 UTC	-73.982524	40.751260	-73.979654	
3	2012-12-01 21:12:12.0000002	2012-12-01 21:12:12 UTC	-73.981160	40.767807	-73.990448	
4	2012-12-01 21:12:12.0000003	2012-12-01 21:12:12 UTC	-73.966046	40.789775	-73.988565	

In [59]:
```
jan_row_15 = df[df['pickup_datetime'].str.startswith('2015-01')]

print(len(jan_row_15))
```

94

Another Method

In [60]:
```
# Converting pickup_datetime column into datetime format
df['date'] = pd.to_datetime(df['pickup_datetime'])
df.head(5)
```

Out[60]:

	key	pickup_datetime	pickup_longitude	pickup_latitude	dropoff_longitude	drop
0	2015-01-27 13:08:24.0000002	2015-01-27 13:08:24 UTC	-73.973320	40.763805	-73.981430	
1	2015-01-27 13:08:24.0000003	2015-01-27 13:08:24 UTC	-73.986862	40.719383	-73.998886	
2	2011-10-08 11:53:44.0000002	2011-10-08 11:53:44 UTC	-73.982524	40.751260	-73.979654	
3	2012-12-01 21:12:12.0000002	2012-12-01 21:12:12 UTC	-73.981160	40.767807	-73.990448	
4	2012-12-01 21:12:12.0000003	2012-12-01 21:12:12 UTC	-73.966046	40.789775	-73.988565	

In [61]:
```
# Filter the rows based on requirement.
num_rows = len(df[(df['date'].dt.year == 2015) & (df['date'].dt.month == 1)])

print(num_rows)
```

```
94
```

- there are 94 rows have a pickup datetime in january 2015.
- It will use to find the analytics and visualization of data in year 2015 in month of january.
- It will helps in decision making.

In [ ]: