# Importing libraries

Zomato (Bangalore Restaurant Explorer) The project aims to analyse and visualize restaurant data from Zomato in Bangalore, providing valuable insights into the city's culinary landscape.

In [1]:
```
# import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('dark_background')
```

In [2]:
```
df=pd.read_csv('zomato.csv') #reading csv file , loading a dataset.
```

In [12]:
```
# print Dataset
df
```

Out[12]:

| | url | address | name | online_order | book_ |
|---|---|---|---|---|---|
| **0** | https://www.zomato.com/bangalore/jalsa-banasha... | 942, 21st Main Road, 2nd Stage, Banashankari, ... | Jalsa | Yes | |
| **1** | https://www.zomato.com/bangalore/spice-elephan... | 2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ... | Spice Elephant | Yes | |
| **2** | https://www.zomato.com/SanchurroBangalore?cont... | 1112, Next to KIMS Medical College, 17th Cross... | San Churro Cafe | Yes | |
| **3** | https://www.zomato.com/bangalore/addhuri-udupi... | 1st Floor, Annakuteera, 3rd Stage, Banashankar... | Addhuri Udupi Bhojana | No | |
| **4** | https://www.zomato.com/bangalore/grand-village... | 10, 3rd Floor, Lakshmi Associates, Gandhi Baza... | Grand Village | No | |
| **...** | ... | ... | ... | ... | |
| **51712** | https://www.zomato.com/bangalore/best-brews-fo... | Four Points by Sheraton Bengaluru, 43/3, White... | Best Brews - Four Points by Sheraton Bengaluru... | No | |
| **51713** | https://www.zomato.com/bangalore/vinod-bar-and... | Number 10, Garudachar Palya, Mahadevapura, Whi... | Vinod Bar And Restaurant | No | |
| **51714** | https://www.zomato.com/bangalore/plunge-sherat... | Sheraton Grand Bengaluru Whitefield Hotel & Co... | Plunge - Sheraton Grand Bengaluru Whitefield H... | No | |
| **51715** | https://www.zomato.com/bangalore/chime-sherato... | Sheraton Grand Bengaluru Whitefield Hotel & Co... | Chime - Sheraton Grand Bengaluru Whitefield Ho... | No | |
| **51716** | https://www.zomato.com/bangalore/the-nest-the-... | ITPL Main Road, KIADB Export Promotion Industr... | The Nest - The Den Bengaluru | No | |

51717 rows × 17 columns

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

Out[4]:

| | url | address | name | online_order | book_table |
|---|---|---|---|---|---|
| **0** | https://www.zomato.com/bangalore/jalsa-banasha... | 942, 21st Main Road, 2nd Stage, Banashankari, ... | Jalsa | Yes | Yes |
| **1** | https://www.zomato.com/bangalore/spice-elephan... | 2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ... | Spice Elephant | Yes | No |
| **2** | https://www.zomato.com/SanchurroBangalore?cont... | 1112, Next to KIMS Medical College, 17th Cross... | San Churro Cafe | Yes | No |
| **3** | https://www.zomato.com/bangalore/addhuri-udupi... | 1st Floor, Annakuteera, 3rd Stage, Banashankar... | Addhuri Udupi Bhojana | No | No |
| **4** | https://www.zomato.com/bangalore/grand-village... | 10, 3rd Floor, Lakshmi Associates, Gandhi Baza... | Grand Village | No | No |

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

Out[4]:

| | url | address | name | online_order | book_tab |
|---|---|---|---|---|---|
| **51712** | https://www.zomato.com/bangalore/best-brews-fo... | Four Points by Sheraton Bengaluru, 43/3, White... | Best Brews - Four Points by Sheraton Bengaluru... | No | N |
| **51713** | https://www.zomato.com/bangalore/vinod-bar-and... | Number 10, Garudachar Palya, Mahadevapura, Whi... | Vinod Bar And Restaurant | No | N |
| **51714** | https://www.zomato.com/bangalore/plunge-sherat... | Sheraton Grand Bengaluru Whitefield Hotel & Co... | Plunge - Sheraton Grand Bengaluru Whitefield H... | No | N |
| **51715** | https://www.zomato.com/bangalore/chime-sherato... | Sheraton Grand Bengaluru Whitefield Hotel & Co... | Chime - Sheraton Grand Bengaluru Whitefield Ho... | No | Y |
| **51716** | https://www.zomato.com/bangalore/the-nest-the-... | ITPL Main Road, KIADB Export Promotion Industr... | The Nest - The Den Bengaluru | No | N |

In [3]:
```
df.shape
```

Out[3]:
```
(51717, 17)
```

## Dropping Duplicates

In [10]:
```
df.describe()
```

Out[10]:

|  | votes |
|---|---|
| **count** | 51717.000000 |
| **mean** | 283.697527 |
| **std** | 803.838853 |
| **min** | 0.000000 |
| **25%** | 7.000000 |
| **50%** | 41.000000 |
| **75%** | 198.000000 |
| **max** | 16832.000000 |

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51717 entries, 0 to 51716
Data columns (total 17 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   url                          51717 non-null  object
 1   address                      51717 non-null  object
 2   name                         51717 non-null  object
 3   online_order                 51717 non-null  object
 4   book_table                   51717 non-null  object
 5   rate                         43942 non-null  object
 6   votes                        51717 non-null  int64 
 7   phone                        50509 non-null  object
 8   location                     51696 non-null  object
 9   rest_type                    51490 non-null  object
 10  dish_liked                   23639 non-null  object
 11  cuisines                     51672 non-null  object
 12  approx_cost(for two people)  51371 non-null  object
 13  reviews_list                 51717 non-null  object
 14  menu_item                    51717 non-null  object
 15  listed_in(type)              51717 non-null  object
 16  listed_in(city)              51717 non-null  object
dtypes: int64(1), object(16)
memory usage: 6.7+ MB
```

In [35]:
```
df.columns
```

Out[35]:
```
Index(['url', 'address', 'name', 'online_order', 'book_table', 'rate', 'vote
s',
       'phone', 'location', 'rest_type', 'dish_liked', 'cuisines',
       'approx_cost(for two people)', 'reviews_list', 'menu_item',
       'listed_in(type)', 'listed_in(city)'],
      dtype='object')
```

## Drop columns

In [36]:
```
df=df.drop(['url' ,'address','phone','menu_item','dish_liked','reviews_list'],
df.head()
```

Out[36]:

| | name | online_order | book_table | rate | votes | location | rest_type | cuisines | approx_c two p |
|---|---|---|---|---|---|---|---|---|---|
| **0** | Jalsa | Yes | Yes | 4.1/5 | 775 | Banashankari | Casual Dining | North Indian, Mughlai, Chinese | |
| **1** | Spice Elephant | Yes | No | 4.1/5 | 787 | Banashankari | Casual Dining | Chinese, North Indian, Thai | |
| **2** | San Churro Cafe | Yes | No | 3.8/5 | 918 | Banashankari | Cafe, Casual Dining | Cafe, Mexican, Italian | |
| **3** | Addhuri Udupi Bhojana | No | No | 3.7/5 | 88 | Banashankari | Quick Bites | South Indian, North Indian | |
| **4** | Grand Village | No | No | 3.8/5 | 166 | Basavanagudi | Casual Dining | North Indian, Rajasthani | |

## Cleaning Rate Column

In [37]:
```
df['rate'].unique()
```

Out[37]:
```
array(['4.1/5', '3.8/5', '3.7/5', '3.6/5', '4.6/5', '4.0/5', '4.2/5',
       '3.9/5', '3.1/5', '3.0/5', '3.2/5', '3.3/5', '2.8/5', '4.4/5',
       '4.3/5', 'NEW', '2.9/5', '3.5/5', nan, '2.6/5', '3.8 /5', '3.4/5',
       '4.5/5', '2.5/5', '2.7/5', '4.7/5', '2.4/5', '2.2/5', '2.3/5',
       '3.4 /5', '-', '3.6 /5', '4.8/5', '3.9 /5', '4.2 /5', '4.0 /5',
       '4.1 /5', '3.7 /5', '3.1 /5', '2.9 /5', '3.3 /5', '2.8 /5',
       '3.5 /5', '2.7 /5', '2.5 /5', '3.2 /5', '2.6 /5', '4.5 /5',
       '4.3 /5', '4.4 /5', '4.9/5', '2.1/5', '2.0/5', '1.8/5', '4.6 /5',
       '4.9 /5', '3.0 /5', '4.8 /5', '2.3 /5', '4.7 /5', '2.4 /5',
       '2.1 /5', '2.2 /5', '2.0 /5', '1.8 /5'], dtype=object)
```

## Removing "NEW" , "-" and "/5" from Rate column

In [38]:
```
def handlerate(value):
    if value == 'NEW' or value == '-':
        return np.nan
    else:
        value = str(value).split('/')
        value = value[0]
        return float(value)

# Assuming you have a DataFrame named 'df'
df['rate'] = df['rate'].apply(handlerate)
df['rate'] = df['rate'].astype(float)  # Ensure the 'rate' column is of float
df['rate'].head()
```

Out[38]:
```
0    4.1
1    4.1
2    3.8
3    3.7
4    3.8
Name: rate, dtype: float64
```

In [23]:
```
df.rate.isnull().sum()
```

Out[23]:
```
10019
```

## Filling Null Values in Rate Column with Mean

In [39]:
```
df['rate'].fillna(df['rate'].mean(), inplace = True)
df['rate'].isnull().sum()
```

Out[39]:
```
0
```

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

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 51609 entries, 0 to 51716
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   name                         51609 non-null  object 
 1   online_order                 51609 non-null  object 
 2   book_table                   51609 non-null  object 
 3   rate                         51609 non-null  float64
 4   votes                        51609 non-null  int64  
 5   location                     51588 non-null  object 
 6   rest_type                    51382 non-null  object 
 7   cuisines                     51564 non-null  object 
 8   approx_cost(for two people)  51265 non-null  object 
 9   listed_in(type)              51609 non-null  object 
 10  listed_in(city)              51609 non-null  object 
dtypes: float64(1), int64(1), object(9)
memory usage: 4.7+ MB
```

## Dropping Null Values

In [40]:
```
df.dropna(inplace = True)
df.head()
```

Out[40]:

| | name | online_order | book_table | rate | votes | location | rest_type | cuisines | approx_cc two pe |
|---|---|---|---|---|---|---|---|---|---|
| **0** | Jalsa | Yes | Yes | 4.1 | 775 | Banashankari | Casual Dining | North Indian, Mughlai, Chinese | |
| **1** | Spice Elephant | Yes | No | 4.1 | 787 | Banashankari | Casual Dining | Chinese, North Indian, Thai | |
| **2** | San Churro Cafe | Yes | No | 3.8 | 918 | Banashankari | Cafe, Casual Dining | Cafe, Mexican, Italian | |
| **3** | Addhuri Udupi Bhojana | No | No | 3.7 | 88 | Banashankari | Quick Bites | South Indian, North Indian | |
| **4** | Grand Village | No | No | 3.8 | 166 | Basavanagudi | Casual Dining | North Indian, Rajasthani | |

## change column name

In [41]:
```
df.rename(columns = {'approx_cost(for two people)':'Cost2plates','listed_in(ty
df.head()
```

Out[41]:

| | name | online_order | book_table | rate | votes | location | rest_type | cuisines | Cost2plate |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Jalsa | Yes | Yes | 4.1 | 775 | Banashankari | Casual Dining | North Indian, Mughlai, Chinese | 8( |
| 1 | Spice Elephant | Yes | No | 4.1 | 787 | Banashankari | Casual Dining | Chinese, North Indian, Thai | 8( |
| 2 | San Churro Cafe | Yes | No | 3.8 | 918 | Banashankari | Cafe, Casual Dining | Cafe, Mexican, Italian | 8( |
| 3 | Addhuri Udupi Bhojana | No | No | 3.7 | 88 | Banashankari | Quick Bites | South Indian, North Indian | 3( |
| 4 | Grand Village | No | No | 3.8 | 166 | Basavanagudi | Casual Dining | North Indian, Rajasthani | 6( |

In [28]:
```
df['location'].unique()
```

Out[28]:
```
array(['Banashankari', 'Basavanagudi', 'Mysore Road', 'Jayanagar',
       'Kumaraswamy Layout', 'Rajarajeshwari Nagar', 'Vijay Nagar',
       'Uttarahalli', 'JP Nagar', 'South Bangalore', 'City Market',
       'Nagarbhavi', 'Bannerghatta Road', 'BTM', 'Kanakapura Road',
       'Bommanahalli', 'CV Raman Nagar', 'Electronic City', 'HSR',
       'Marathahalli', 'Wilson Garden', 'Shanti Nagar',
       'Koramangala 5th Block', 'Koramangala 8th Block', 'Richmond Road',
       'Koramangala 7th Block', 'Jalahalli', 'Koramangala 4th Block',
       'Bellandur', 'Sarjapur Road', 'Whitefield', 'East Bangalore',
       'Old Airport Road', 'Indiranagar', 'Koramangala 1st Block',
       'Frazer Town', 'RT Nagar', 'MG Road', 'Brigade Road',
       'Lavelle Road', 'Church Street', 'Ulsoor', 'Residency Road',
       'Shivajinagar', 'Infantry Road', 'St. Marks Road',
       'Cunningham Road', 'Race Course Road', 'Commercial Street',
       'Vasanth Nagar', 'HBR Layout', 'Domlur', 'Ejipura',
       'Jeevan Bhima Nagar', 'Old Madras Road', 'Malleshwaram',
       'Seshadripuram', 'Kammanahalli', 'Koramangala 6th Block',
       'Majestic', 'Langford Town', 'Central Bangalore', 'Sanjay Nagar',
       'Brookefield', 'ITPL Main Road, Whitefield',
       'Varthur Main Road, Whitefield', 'KR Puram',
       'Koramangala 2nd Block', 'Koramangala 3rd Block', 'Koramangala',
       'Hosur Road', 'Rajajinagar', 'Banaswadi', 'North Bangalore',
       'Nagawara', 'Hennur', 'Kalyan Nagar', 'New BEL Road', 'Jakkur',
       'Rammurthy Nagar', 'Thippasandra', 'Kaggadasapura', 'Hebbal',
       'Kengeri', 'Sankey Road', 'Sadashiv Nagar', 'Basaveshwara Nagar',
       'Yeshwantpur', 'West Bangalore', 'Magadi Road', 'Yelahanka',
       'Sahakara Nagar', 'Peenya'], dtype=object)
```

In [29]:
```
df['listed_in(city)'].unique()
```

Out[29]:
```
array(['Banashankari', 'Bannerghatta Road', 'Basavanagudi', 'Bellandur',
       'Brigade Road', 'Brookefield', 'BTM', 'Church Street',
       'Electronic City', 'Frazer Town', 'HSR', 'Indiranagar',
       'Jayanagar', 'JP Nagar', 'Kalyan Nagar', 'Kammanahalli',
       'Koramangala 4th Block', 'Koramangala 5th Block',
       'Koramangala 6th Block', 'Koramangala 7th Block', 'Lavelle Road',
       'Malleshwaram', 'Marathahalli', 'MG Road', 'New BEL Road',
       'Old Airport Road', 'Rajajinagar', 'Residency Road',
       'Sarjapur Road', 'Whitefield'], dtype=object)
```

# Listed in (city) and location, both are there , lets keep only one.

In [42]:
```
df=df.drop(['listed_in(city)'], axis =1)
df.head()
```

Out[42]:

| | name | online_order | book_table | rate | votes | location | rest_type | cuisines | Cost2plate |
|---|---|---|---|---|---|---|---|---|---|
| 0 | Jalsa | Yes | Yes | 4.1 | 775 | Banashankari | Casual Dining | North Indian, Mughlai, Chinese | 80 |
| 1 | Spice Elephant | Yes | No | 4.1 | 787 | Banashankari | Casual Dining | Chinese, North Indian, Thai | 80 |
| 2 | San Churro Cafe | Yes | No | 3.8 | 918 | Banashankari | Cafe, Casual Dining | Cafe, Mexican, Italian | 80 |
| 3 | Addhuri Udupi Bhojana | No | No | 3.7 | 88 | Banashankari | Quick Bites | South Indian, North Indian | 30 |
| 4 | Grand Village | No | No | 3.8 | 166 | Basavanagudi | Casual Dining | North Indian, Rajasthani | 60 |

In [43]:
```
df['Cost2plates'].unique()
```

Out[43]:
```
array(['800', '300', '600', '700', '550', '500', '450', '650', '400',
       '900', '200', '750', '150', '850', '100', '1,200', '350', '250',
       '950', '1,000', '1,500', '1,300', '199', '80', '1,100', '160',
       '1,600', '230', '130', '50', '190', '1,700', '1,400', '180',
       '1,350', '2,200', '2,000', '1,800', '1,900', '330', '2,500',
       '2,100', '3,000', '2,800', '3,400', '40', '1,250', '3,500',
       '4,000', '2,400', '2,600', '120', '1,450', '469', '70', '3,200',
       '60', '560', '240', '360', '6,000', '1,050', '2,300', '4,100',
       '5,000', '3,700', '1,650', '2,700', '4,500', '140'], dtype=object)
```

## Removing , from Cost2Plates Column

In [44]:
```
def handlecomma(value):
    value == str(value)
    if ',' in value:
        value = value.replace(',','')
        return float(value)
    else:
        return float(value)
df['Cost2plates'] = df['Cost2plates'].apply(handlecomma)
df['Cost2plates'].unique()
```

Out[44]:
```
array([ 800.,  300.,  600.,  700.,  550.,  500.,  450.,  650.,  400.,
        900.,  200.,  750.,  150.,  850.,  100., 1200.,  350.,  250.,
        950., 1000., 1500., 1300.,  199.,   80., 1100.,  160., 1600.,
        230.,  130.,   50.,  190., 1700., 1400.,  180., 1350., 2200.,
       2000., 1800., 1900.,  330., 2500., 2100., 3000., 2800., 3400.,
         40., 1250., 3500., 4000., 2400., 2600.,  120., 1450.,  469.,
         70., 3200.,   60.,  560.,  240.,  360., 6000., 1050., 2300.,
       4100., 5000., 3700., 1650., 2700., 4500.,  140.])
```

In [45]:
```
df.head()
```

Out[45]:

| | name | online_order | book_table | rate | votes | location | rest_type | cuisines | Cost2plate |
|---|---|---|---|---|---|---|---|---|---|
| **0** | Jalsa | Yes | Yes | 4.1 | 775 | Banashankari | Casual Dining | North Indian, Mughlai, Chinese | 800 |
| **1** | Spice Elephant | Yes | No | 4.1 | 787 | Banashankari | Casual Dining | Chinese, North Indian, Thai | 800 |
| **2** | San Churro Cafe | Yes | No | 3.8 | 918 | Banashankari | Cafe, Casual Dining | Cafe, Mexican, Italian | 800 |
| **3** | Addhuri Udupi Bhojana | No | No | 3.7 | 88 | Banashankari | Quick Bites | South Indian, North Indian | 300 |
| **4** | Grand Village | No | No | 3.8 | 166 | Basavanagudi | Casual Dining | North Indian, Rajasthani | 600 |

## Cleaning Rest Type Column

In [10]:
```
df['rest_type'].value_counts()
```

Out[10]:
```
Quick Bites                  19132
Casual Dining                10330
Cafe                          3732
Delivery                      2604
Dessert Parlor                2263
                             ...  
Dessert Parlor, Kiosk            2
Food Court, Beverage Shop        2
Dessert Parlor, Food Court       2
Sweet Shop, Dessert Parlor       1
Quick Bites, Kiosk               1
Name: rest_type, Length: 93, dtype: int64
```

In [11]:
```
rest_types =df['rest_type'].value_counts(ascending =False)
rest_types
```

Out[11]:
```
Quick Bites                  19132
Casual Dining                10330
Cafe                          3732
Delivery                      2604
Dessert Parlor                2263
                             ...  
Dessert Parlor, Kiosk            2
Food Court, Beverage Shop        2
Dessert Parlor, Food Court       2
Sweet Shop, Dessert Parlor       1
Quick Bites, Kiosk               1
Name: rest_type, Length: 93, dtype: int64
```

In [12]:
```
rest_type_lessthan1000 = rest_types[rest_types<1000]
rest_type_lessthan1000
```

Out[12]:
```
Beverage Shop                867
Bar                          697
Food Court                   624
Sweet Shop                   468
Bar, Casual Dining           425
                            ... 
Dessert Parlor, Kiosk          2
Food Court, Beverage Shop      2
Dessert Parlor, Food Court     2
Sweet Shop, Dessert Parlor     1
Quick Bites, Kiosk             1
Name: rest_type, Length: 85, dtype: int64
```

## Making Rest Type less than 1000 in Frequency as others

In [14]:
```
# Define the list of rest types that are less than 1000
rest_types_lessthan1000 = ['type1', 'type2', 'type3']  # Replace with your act

def handle_rest_type(value):
    if value in rest_types_lessthan1000:
        return 'others'
    else:
        return value

# Assuming 'df' is your DataFrame
df['rest_type'] = df['rest_type'].apply(handle_rest_type)
print(df['rest_type'].value_counts())
```

```
Quick Bites                  19132
Casual Dining                10330
Cafe                          3732
Delivery                      2604
Dessert Parlor                2263
                             ...
Dessert Parlor, Kiosk            2
Food Court, Beverage Shop        2
Dessert Parlor, Food Court       2
Sweet Shop, Dessert Parlor       1
Quick Bites, Kiosk               1
Name: rest_type, Length: 93, dtype: int64
```

## changing location Column

In [5]:
```
location = df['location'].value_counts(ascending = False)
location_lessthan300 = location[location<300]

def handle_location(value):
    if(value in location_lessthan300):
        return 'others'
    else:
        return value

df['location'] = df['location'].apply(handle_location)
df['location'].value_counts()
```

Out[5]:
```
BTM                      5124
others                   4707
HSR                      2523
Koramangala 5th Block    2504
JP Nagar                 2235
Whitefield               2144
Indiranagar              2083
Jayanagar                1926
Marathahalli             1846
Bannerghatta Road        1630
Bellandur                1286
Electronic City          1258
Koramangala 1st Block    1238
Brigade Road             1218
Koramangala 7th Block    1181
Koramangala 6th Block    1156
Sarjapur Road            1065
Ulsoor                   1023
Koramangala 4th Block    1017
MG Road                   918
Banashankari              906
Kalyan Nagar              853
Richmond Road             812
Frazer Town               727
Malleshwaram              725
Basavanagudi              684
Residency Road            675
Banaswadi                 664
Brookefield               658
New BEL Road              649
Kammanahalli              648
Rajajinagar               591
Church Street             569
Lavelle Road              529
Shanti Nagar              511
Shivajinagar              499
Domlur                    496
Cunningham Road           491
Old Airport Road          446
Ejipura                   439
Commercial Street         370
St. Marks Road            352
Koramangala 8th Block     320
Name: location, dtype: int64
```

## Cleaning Cuisines Column

In [5]:
```python
cuisines = df['cuisines'].value_counts(ascending = False)
cuisines_lessthan100 = cuisines[cuisines<100]

def handle_cuisines(value):
    if(value in cuisines_lessthan100):
        return 'others'
    else:
        return value

df['cuisines'] = df['cuisines'].apply(handle_cuisines)
df['cuisines'].value_counts()
```

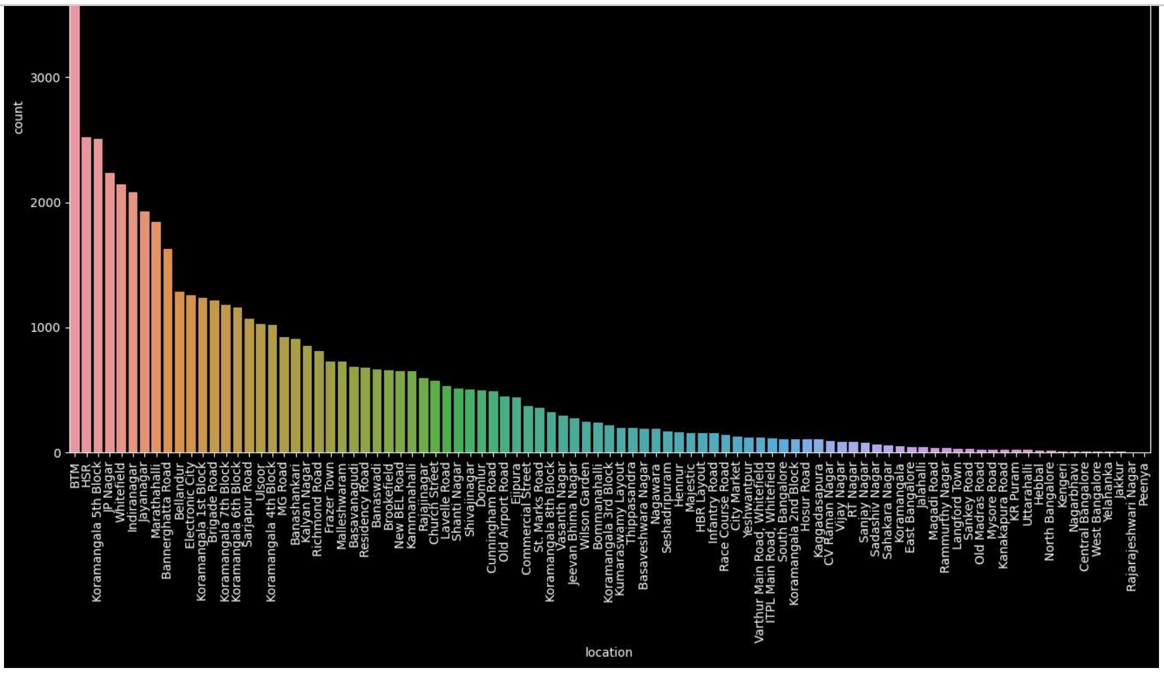
Out[5]:
```
others                                 26460
North Indian                            2913
North Indian, Chinese                   2385
South Indian                            1828
Biryani                                  918
                                       ...
South Indian, Chinese, North Indian      105
Italian, Pizza                           105
North Indian, Mughlai, Chinese           104
South Indian, Fast Food                  104
North Indian, Chinese, Seafood           102
Name: cuisines, Length: 70, dtype: int64
```

## Data is clean

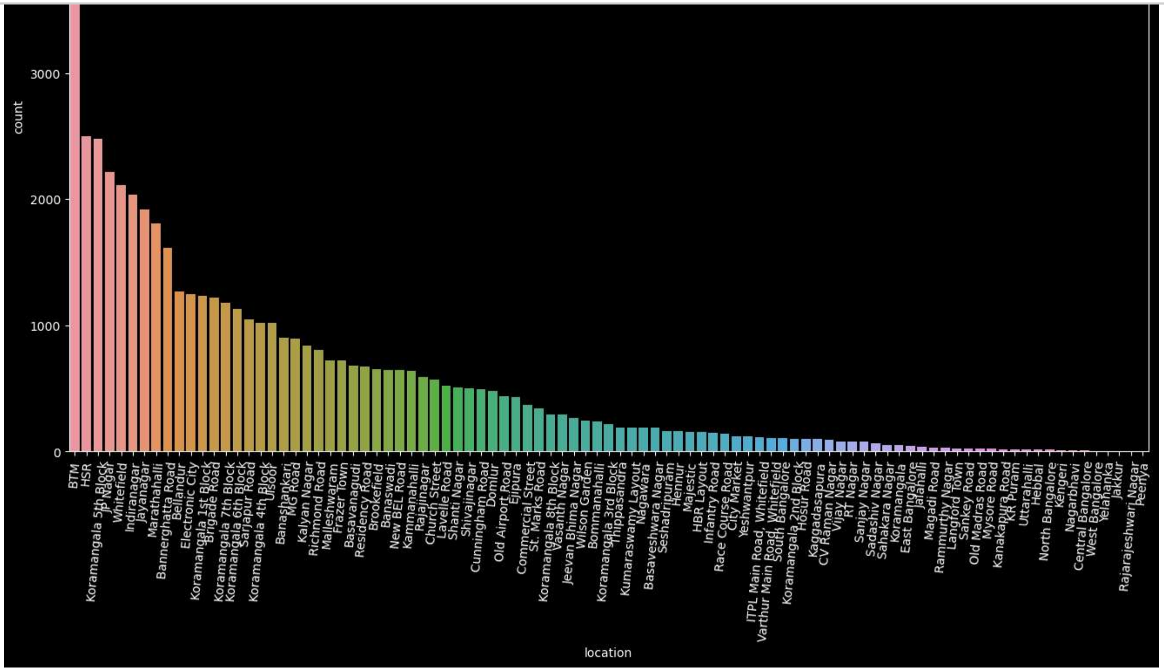
## Lets jumps to Visualization

## count plot of various location

In [23]:
```
plt.figure(figsize=(16, 10))
order = df['location'].value_counts().index  # Specify the order based on valu
x = sns.countplot(data=df, x='location', order=order)
plt.xticks(rotation=90)
```



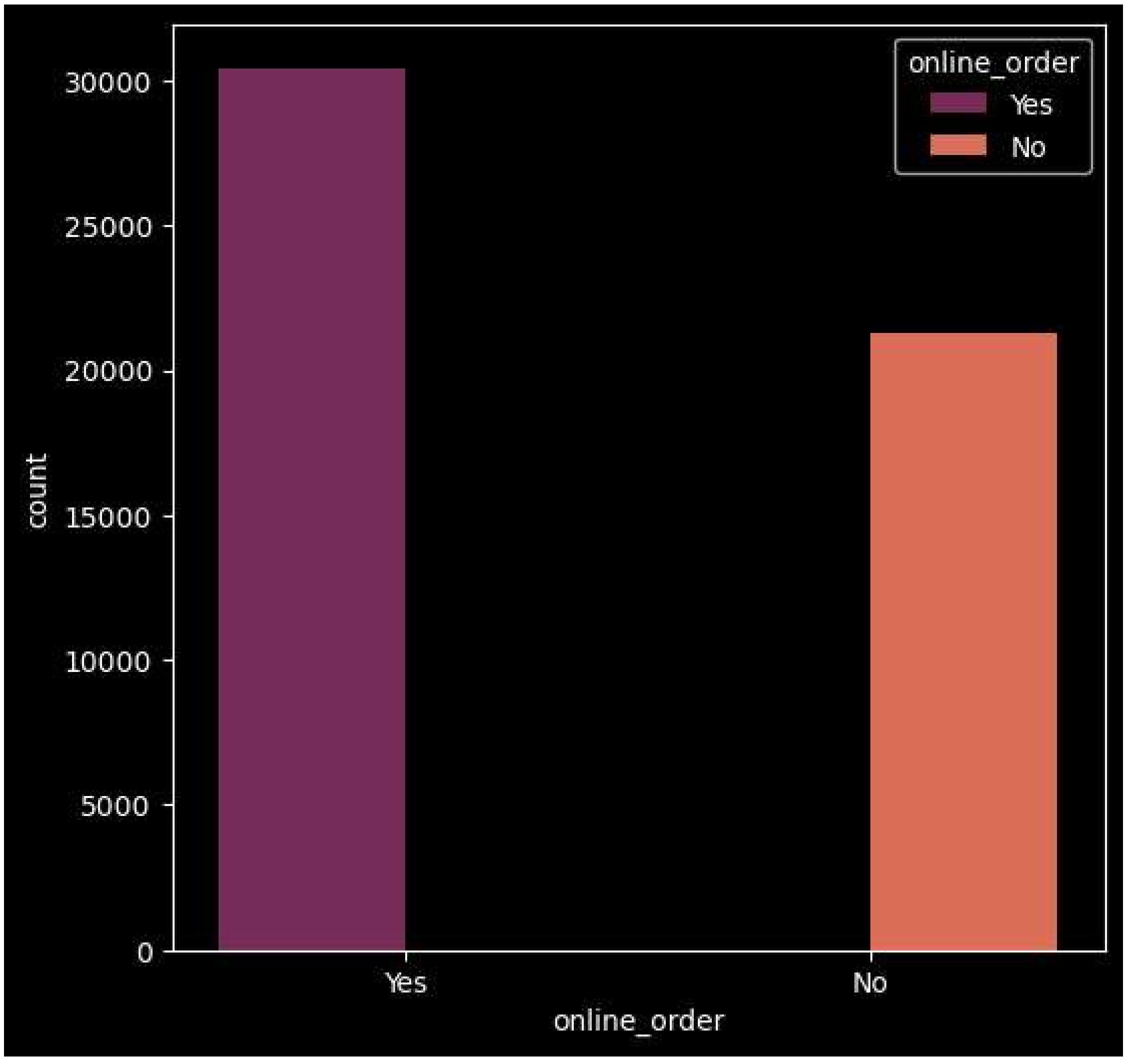
In [48]:
```
plt.figure(figsize=(16, 10))
order = df['location'].value_counts().index  # Specify the order based on valu
x = sns.countplot(data=df, x='location', order=order)
plt.xticks(rotation=85)
```



## Visualizing Online Order

In [27]:
```
# Visualizing Online Order
plt.figure(figsize=(6, 6))
sns.countplot(data=df, x='online_order', palette='rocket', hue='online_order')
```

Out[27]:
```
<Axes: xlabel='online_order', ylabel='count'>
```

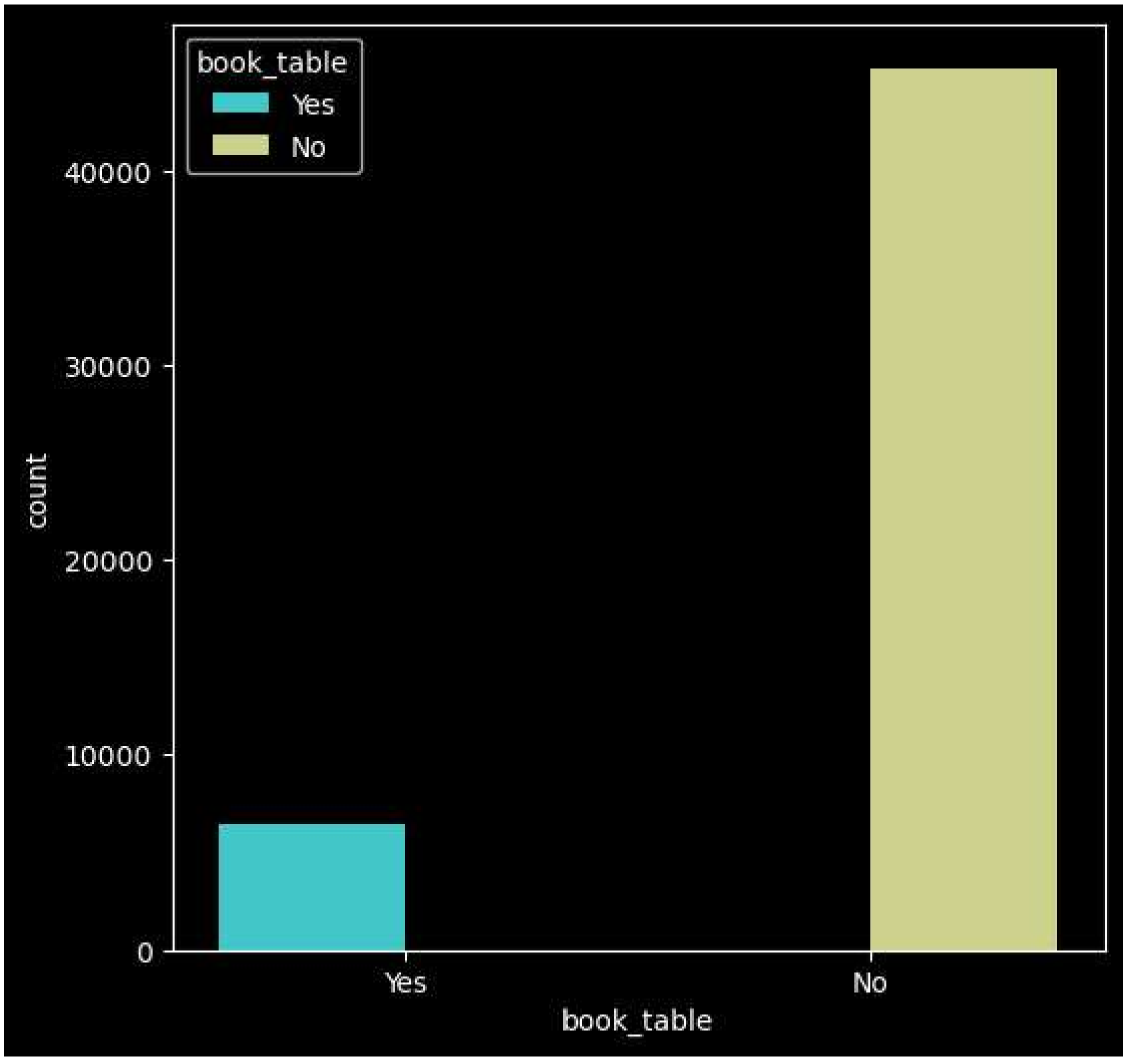


## Visualizing Book Table

In [28]:
```python
# Visualizing Book Table

plt.figure(figsize=(6, 6))
sns.countplot(data=df, x='book_table', palette='rainbow', hue='book_table')
```

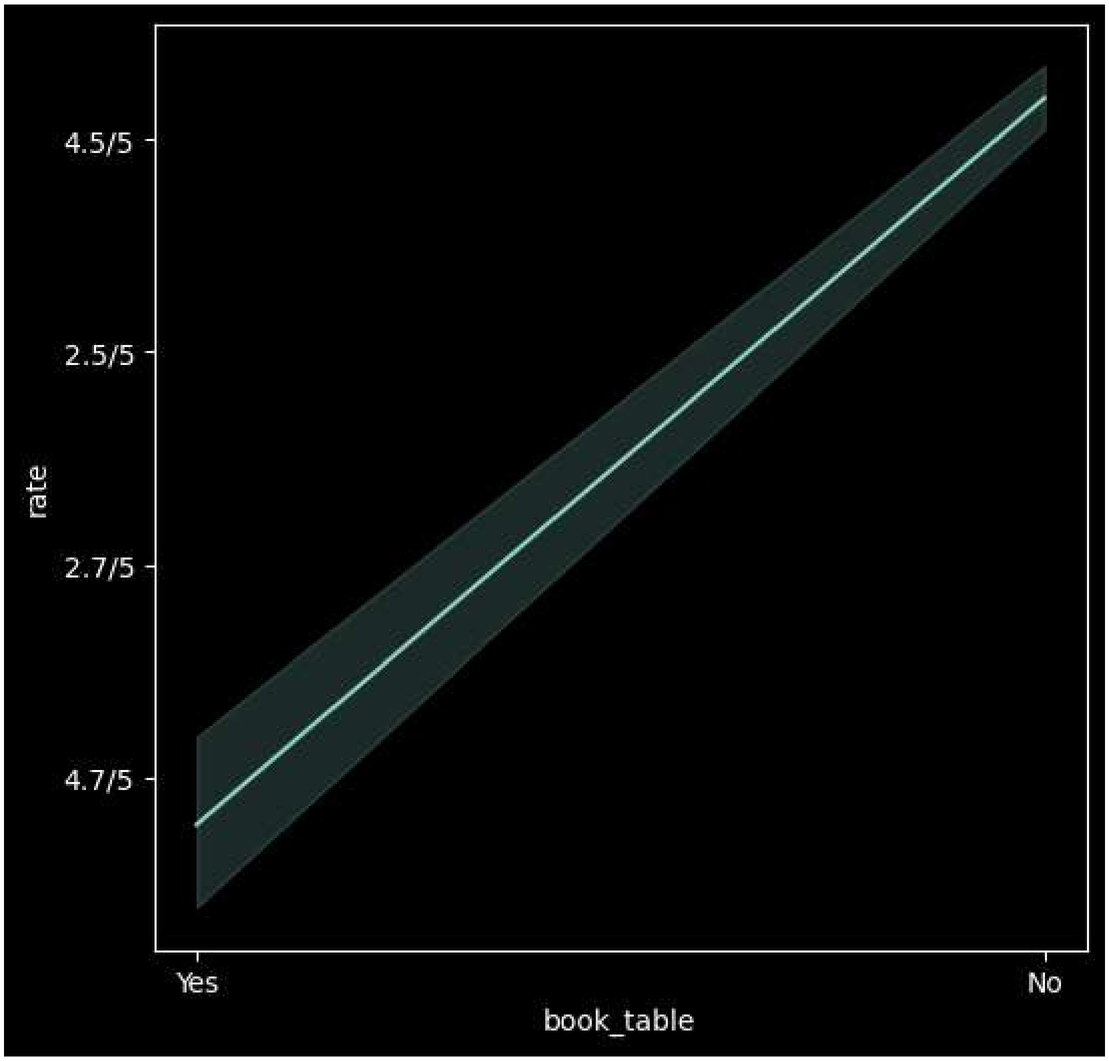
Out[28]:
```
<Axes: xlabel='book_table', ylabel='count'>
```



## Visualizing Online Vs Rate

In [17]:
```
plt.figure(figsize=(6, 6))
sns.lineplot(data=df, x='book_table', y='rate')
```

Out[17]:
```
<Axes: xlabel='book_table', ylabel='rate'>
```



In [49]:
```
df.columns
```

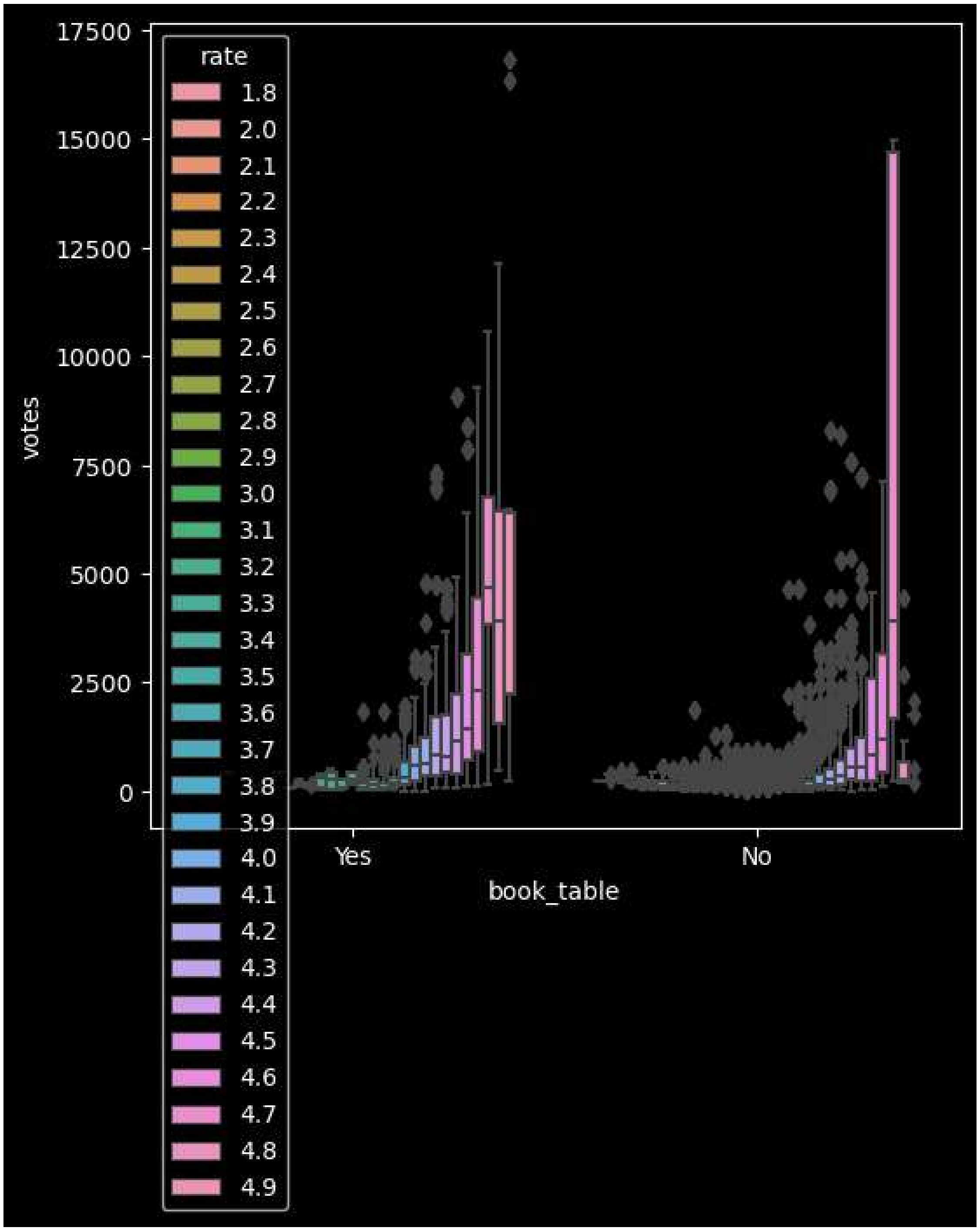
Out[49]:
```
Index(['name', 'online_order', 'book_table', 'rate', 'votes', 'location',
       'rest_type', 'cuisines', 'Cost2plates', 'Type'],
      dtype='object')
```

In [11]:
```
print(df.dtypes)
```

```
url                            object
address                        object
name                           object
online_order                   object
book_table                     object
rate                           object
votes                           int64
phone                          object
location                       object
rest_type                      object
dish_liked                     object
cuisines                       object
approx_cost(for two people)    object
reviews_list                   object
menu_item                      object
listed_in(type)                object
listed_in(city)                object
dtype: object
```

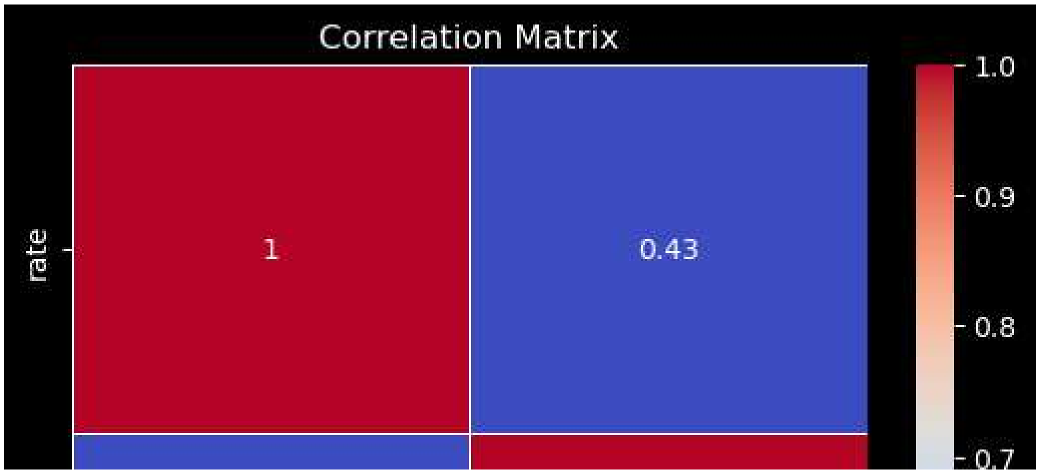
In [38]:
```
plt.figure(figsize=(6, 6))
sns.boxplot(data=df, x='book_table', y='votes', hue='rate')
```

Out[38]:
```
<Axes: xlabel='book_table', ylabel='votes'>
```



In [42]:
```
correlation_matrix = df[['book_table', 'rate', 'votes']].corr()
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=.5)
plt.title('Correlation Matrix')
plt.show()
```

```
C:\Users\kriti\AppData\Local\Temp\ipykernel_12772\706494751.py:1: FutureWa
rning: The default value of numeric_only in DataFrame.corr is deprecated.
In a future version, it will default to False. Select only valid columns o
r specify the value of numeric_only to silence this warning.
  correlation_matrix = df[['book_table', 'rate', 'votes']].corr()
```



## skew

In [6]:
```
df["votes"].skew()
```

Out[6]:
```
7.547819350060049
```

## Online order Facility, Location Wise

In [21]:
```
df1 = df.groupby(['location','online_order'])['name'].count()
df1.to_csv('location_online.csv')
df1 = pd.read_csv('location_online.csv')
df1 = pd.pivot_table(df1 , values=None, index=['location'], columns=['online_o
df1
```

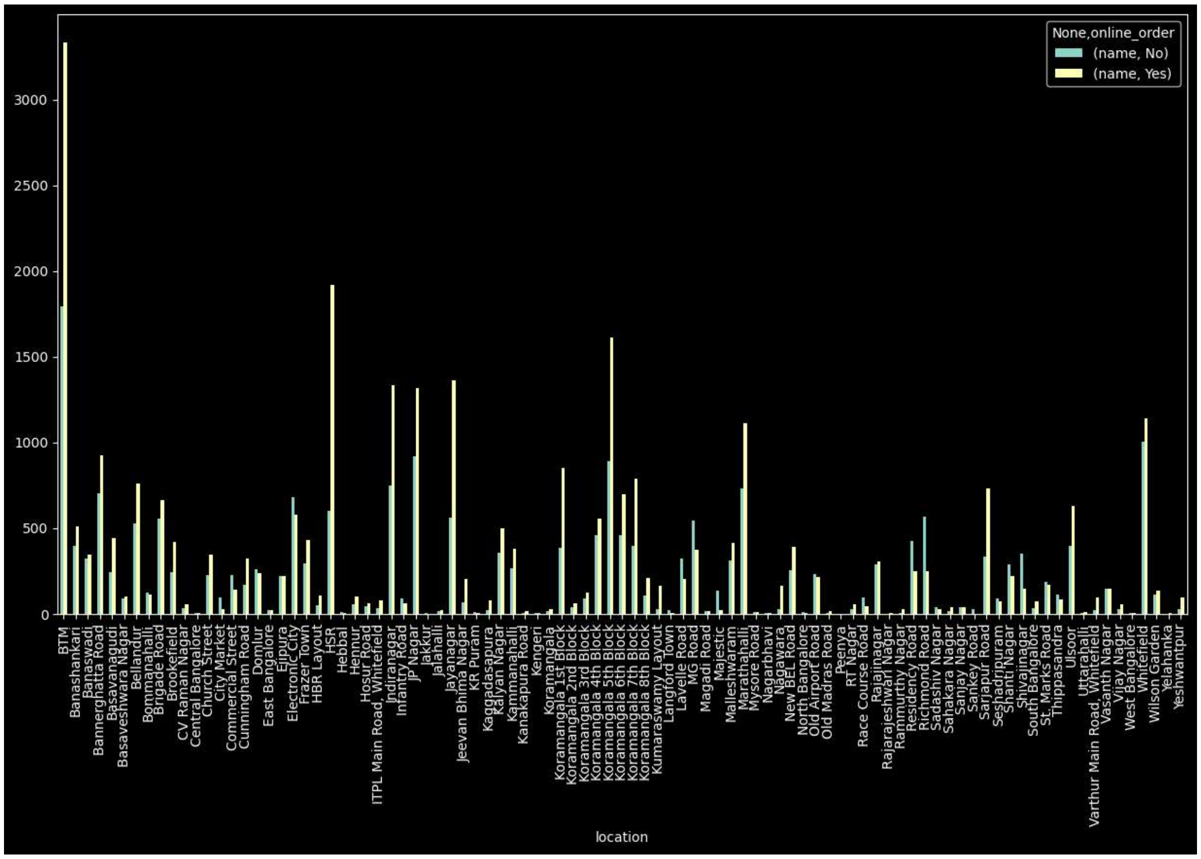
Out[21]:

| | name | |
|---|---|---|
| **online_order** | **No** | **Yes** |
| **location** | | |
| **BTM** | 1792 | 3332 |
| **Banashankari** | 397 | 509 |
| **Banaswadi** | 321 | 343 |
| **Bannerghatta Road** | 706 | 924 |
| **Basavanagudi** | 243 | 441 |
| **...** | ... | ... |
| **West Bangalore** | 4 | 2 |
| **Whitefield** | 1005 | 1139 |
| **Wilson Garden** | 112 | 134 |
| **Yelahanka** | 1 | 5 |
| **Yeshwantpur** | 26 | 93 |

93 rows × 2 columns

```
In [22]: df1.plot(kind ='bar' , figsize = (15,8))
```

```
Out[22]: <Axes: xlabel='location'>
```



## Book Table Facility , Location Wise

```
In [25]: # Book Table Facility , Location Wise
         df2 = df.groupby(['location','book_table'])['name'].count()
         df2.to_csv('location_booktable.csv')
         df2 = pd.read_csv('location_booktable.csv')
         df2 = pd.pivot_table(df2 , values=None, index=['location'], columns=['book_tab
         df2
```

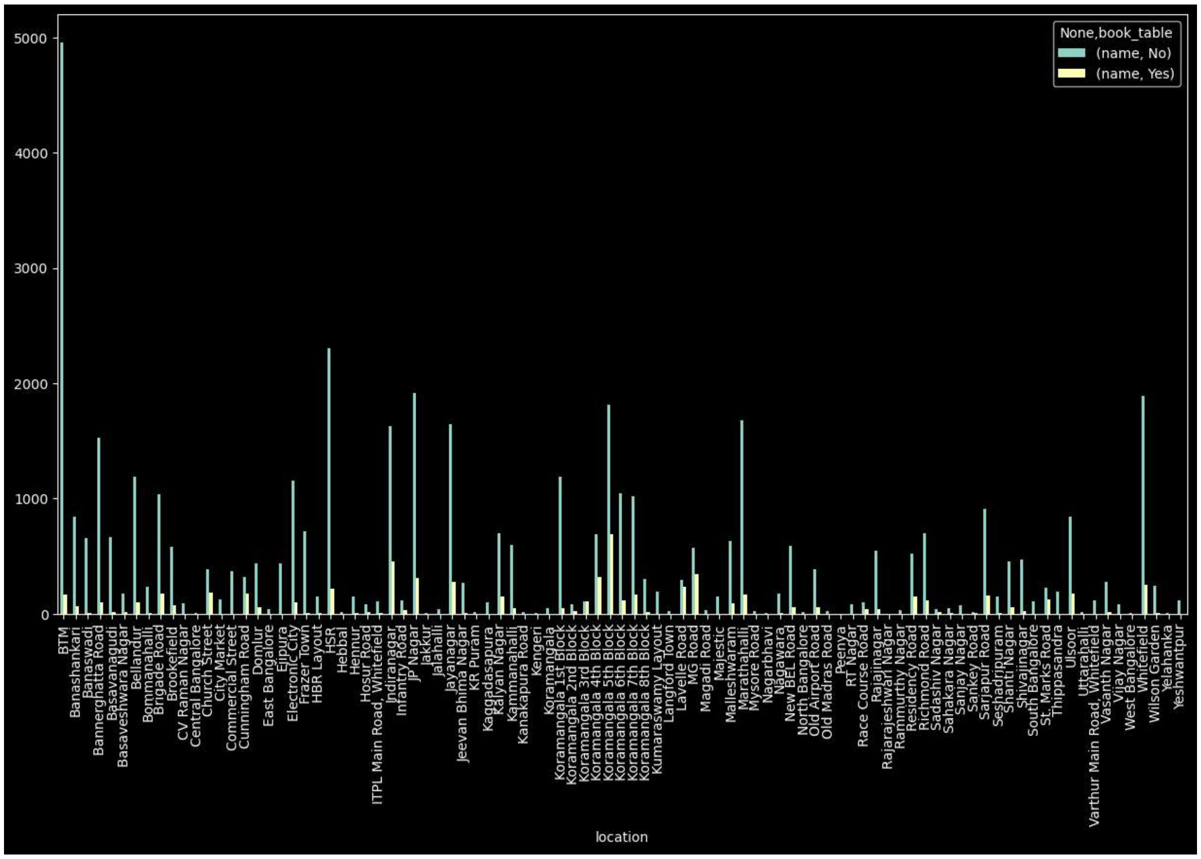
Out[25]:

| | name | |
|---|---|---|
| **book_table** | **No** | **Yes** |
| **location** | | |
| **BTM** | 4956 | 168 |
| **Banashankari** | 842 | 64 |
| **Banaswadi** | 656 | 8 |
| **Bannerghatta Road** | 1531 | 99 |
| **Basavanagudi** | 668 | 16 |
| **...** | ... | ... |
| **West Bangalore** | 6 | 0 |
| **Whitefield** | 1891 | 253 |
| **Wilson Garden** | 241 | 5 |
| **Yelahanka** | 6 | 0 |
| **Yeshwantpur** | 117 | 2 |

93 rows × 2 columns

In [34]:
```
df2.plot(kind ='bar' , figsize = (15,8))
```

Out[34]:
```
<Axes: xlabel='location'>
```



In [ ]: