web scraping magicbrick.com

In [ ]:
```
 from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from selenium.common.exceptions import TimeoutException
from bs4 import BeautifulSoup
import pandas as pd
import time, random, os

# --- Chrome Setup ---
options = Options()
options.add_argument("--start-maximized")
options.add_argument("--disable-blink-features=AutomationControlled")
# options.add_argument("--headless")  # optional

driver = webdriver.Chrome(options=options)
driver.set_script_timeout(60)

urls = [

 "https://www.magicbricks.com/property-for-sale/residential-real-estate?bedroom=

]

# store all cities data here
data_all=[]
for url in urls:
    try:
        city = url.split("cityName=")[-1]
        print(f"\n🌆 Scraping city: {city}")
        driver.get(url)
        time.sleep(random.uniform(4, 6))

        # --- scrolling ---
        print("    ⏬ Scrolling listings...")
        same_height_count = 0
        for scroll_num in range(40):  # up to 60 scrolls max
            last_height = driver.execute_script("return document.body.scrollHeig
            driver.execute_script("window.scrollTo(0, document.body.scrollHeight
            time.sleep(random.uniform(4, 6))  # wait for lazy-loaded listings
            new_height = driver.execute_script("return document.body.scrollHeigh
            print(f"    🔽 Scroll #{scroll_num + 1} | Height: {new_height}")
            if new_height == last_height:
                same_height_count += 1
                if same_height_count >= 3:
                    print("    ✅ End of page reached.")
                    break
            else:
                same_height_count = 0

        # --- Give time for last listings to load ---
        time.sleep(5)
        soup = BeautifulSoup(driver.page_source, "html.parser")
        listings = soup.find_all("div", class_="mb-srp__card")
        print(f"    📦 Total listings found: {len(listings)}")
```

```
        city_data = []
        for i in listings:
            title = i.find("h2").text.strip() if i.find("h2") else None
            carpet = i.find("div", class_="mb-srp__card__summary--value").text.s

            def get_summary_value(name):
                div = i.find("div", {"data-summary": name})
                return div.find("div", class_="mb-srp__card__summary--value").te

            status = get_summary_value("status")
            furnishing = get_summary_value("furnishing")
            floor = get_summary_value("floor")
            facing = get_summary_value("facing")
            bathroom = get_summary_value("bathroom")
            overlooking = get_summary_value("overlooking")

            pricing = i.find("div", class_="mb-srp__card__price--amount").text.s
            pricing_sqcm = i.find("div", class_="mb-srp__card__price--size").tex

            city_data.append({
                "city": city,
                "title": title,
                "carpet": carpet,
                "status": status,
                "furnishing": furnishing,
                "floor": floor,
                "facing": facing,
                "bathroom": bathroom,
                "overlooking": overlooking,
                "pricing": pricing,
                "pricing_sqcm": pricing_sqcm
            })

        df = pd.DataFrame(city_data)
        output_file = f"magicbricks_{city}.csv"
        df.to_csv(output_file, index=False)
        print(f"💾 Saved {len(df)} records for {city} → {output_file}")

        # store in master list too
        data_all.extend(city_data)

        time.sleep(random.uniform(3, 6))

    except TimeoutException:
        print(f"⏰ Timeout at {city}, skipping...")
        continue
    except Exception as e:
        print(f"❌ Error in {city}: {e}")
        continue

driver.quit()

# --- Save combined data ---
if data_all:
    final_df = pd.DataFrame(data_all)
    final_df.to_csv("magicbricks_all_cities.csv", index=False)
    print(f"\n✅ Scraping complete! Total records: {len(final_df)}")
else:
    print("\n⚠️ No data collected.")
```

In [104…
```
import pandas as pd
import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [105…
```
csv_file=[file for file in os.listdir() if file.endswith('.csv')]
print(csv_file)
```

```
['magicbricksMumbai.csv', 'magicbricks_all_cities.csv', 'magicbricks_Bangalore.csv', 'magicbricks_Bhopal.csv', 'magicbricks_Chandigarh.csv', 'magicbricks_Chennai.csv', 'magicbricks_Delhi-NCR.csv', 'magicbricks_Guwahati.csv', 'magicbricks_Kolkata.csv', 'magicbricks_Lucknow.csv', 'magicbricks_Pune.csv', 'magicbricks_Raipur.csv', 'magicbricks_Ranchi.csv', 'magicbricks_Shimla.csv']
```

In [106…
```
df =[]
for file in csv_file:
    dfs=pd.read_csv(file)
    df.append(dfs)

mdf=pd.concat(df,ignore_index=False)
```

In [107…
```
mdf.head()
```

Out[107…

	city	title	carpet	status	furnishing	floor	facing	bathroom	overlookin
0	Mumbai	3 BHK Flat for Sale in Sunteck City 4th Avenu...	1036 sqft	Poss. by Dec '25	Unfurnished	NaN	NaN	3	Nal
1	Mumbai	3 BHK Flat for Sale in Dosti Ambrosia, Wadala...	1156 sqft	Ready to Move	Furnished	NaN	North - West	3	Garden/Parl Main Roa
2	Mumbai	4 BHK Flat for Sale in Juhu, Mumbai	1775 sqft	Ready to Move	Semi-Furnished	4 out of 15	East	4	Main Roa
3	Mumbai	4 BHK Flat for Sale in Omkar Alta Monte, Mala...	3311 sqft	Ready to Move	Unfurnished	11 out of 54	East	4	Garden/Parl Poo
4	Mumbai	5 BHK Flat for Sale in Lodha Fiorenza, Gorega...	3500 sqft	Ready to Move	Unfurnished	NaN	North - West	7	Garden/Parl Poo

In [108…
```
mdf.shape
```

Out[108… `(11282, 11)`

cleaning data and making new columns

In [110…
```
#extracting a new columns from existing columnes

mdf['room_type'] = mdf['title'].str.extract(r'(\d+\s*BHK)', expand=False)
```

In [111…
```
import re
mdf['location'] = mdf['title'].str.extract(r'for\s+sale\s+in\s*(.*)', flags=re.I
```

In [112…
```
mdf['floor_no']= mdf['floor'].str.extract(r'([A-Za-z0-9])\s*out',expand=False)
```

In [113…
```
mdf['pricing_crore']=(mdf['pricing'].str.replace("₹","").str.replace(",","").str
```

In [114…
```
#cleaning the data and make a new columns

def convert_to_crore(value):

    if pd.isna(value):
        return np.nan

    # Clean up the string
    value = str(value).replace('₹', '').replace(',', '').strip()

    # Handle Crores
    if 'Cr' in value:
        try:
            return float(value.replace('Cr', '').strip())
        except:
            return value


    elif 'Lac' in value:
        try:
            return float(value.replace('Lac', '').strip()) / 100
        except:
            return value


    else:
        return value
```

In [115…
```
mdf['full_pricing']=mdf['pricing_crore'].apply(convert_to_crore)
```

In [116…
```
mdf['full_pricing'].str.contains('ac', case=False, na=False).sum()
```

Out[116… 0

In [117…
```
# Convert only numeric values (ignore non-numeric text)
mdf['price_rupees'] = pd.to_numeric(mdf['full_pricing'], errors='coerce') * 1e7
```

In [118…
```
#see uniquie values in this column
mdf['furnishing'].unique()
```

Out[118… array(['Unfurnished', 'Furnished', 'Semi-Furnished', nan], dtype=object)

In [119…
```
#this will fill most common data in nulls
most_common=mdf['furnishing'].mode()[0]
mdf['furnishing']=mdf['furnishing'].fillna(most_common)
```

In [120…
```
#this will fill most common data in nulls
most_common=mdf['status'].mode()[0]
mdf['status']=mdf['status'].fillna(most_common)
```

In [121…
```
#filling null
mdf['floor_no']=mdf['floor_no'].fillna('ground')
```

In [122…
```
#extract sqft area from carpet
mdf['carpet_num'] = mdf['carpet'].str.extract(r'(\d+\.?\d*)')
```

In [123…
```
 mdf['price_rupees']= pd.to_numeric( mdf['price_rupees'],errors='coerce')
 mdf['carpet_num']= pd.to_numeric(  mdf['carpet_num'],errors='coerce')
mdf['price_per_sqft_calc'] = mdf['price_rupees'] / mdf['carpet_num']
mdf['pricing_sqcm'] = mdf['pricing_sqcm'].fillna(
    '₹' + mdf['price_per_sqft_calc'].round(0).astype('Int64').astype(str) + ' pe
)
```

In [124…
```
mdf['price_per_sqft_calc'] = mdf['price_per_sqft_calc'].round(0).astype('Int64')
```

In [125…
```
#droping null in the columns
mdf = mdf.dropna(subset=['price_per_sqft_calc'])
```

In [126…
```
#filling null
mdf['facing']=mdf['facing'].fillna('Not-mention')
```

In [127…
```
mdf.head(80)
```

Out[127...

	city	title	carpet	status	furnishing	floor	facing	bathroom	overloo
0	Mumbai	3 BHK Flat for Sale in Sunteck City 4th Avenu...	1036 sqft	Poss. by Dec '25	Unfurnished	NaN	Not-mention	3	
1	Mumbai	3 BHK Flat for Sale in Dosti Ambrosia, Wadala...	1156 sqft	Ready to Move	Furnished	NaN	North - West	3	Garden/ Main
2	Mumbai	4 BHK Flat for Sale in Juhu, Mumbai	1775 sqft	Ready to Move	Semi-Furnished	4 out of 15	East	4	Main
3	Mumbai	4 BHK Flat for Sale in Omkar Alta Monte, Mala...	3311 sqft	Ready to Move	Unfurnished	11 out of 54	East	4	Garden/
4	Mumbai	5 BHK Flat for Sale in Lodha Fiorenza, Gorega...	3500 sqft	Ready to Move	Unfurnished	NaN	North - West	7	Garden/
...	...	...	...	...	...	...	...	...	
75	Mumbai	4 BHK Apartment for Sale in Godrej Avenue Elev...	2350 sqft	Poss. by Oct '28	Unfurnished	10 out of 54	West	6	Garden/ Pool,
76	Mumbai	4 BHK Apartment for Sale in Piramal Revanta, M...	1520 sqft	Ready to Move	Semi-Furnished	22 out of 40	West	4	Garden/
77	Mumbai	3 BHK Apartment for Sale in Mohan Sadan, Ghatk...	889 sqft	Ready to Move	Semi-Furnished	3 out of 16	East	3	
78	Mumbai	3 BHK Apartment for Sale in Godrej Prime, Chem...	1150 sqft	Ready to Move	Unfurnished	8 out of 14	East	3	Main

	city	title	carpet	status	furnishing	floor	facing	bathroom	overloo
79	Mumbai	2 BHK Apartment for Sale in Meraki Habitats On...	772 sqft	Ready to Move	Unfurnished	NaN	East	2	Garden/ Pool,

80 rows × 19 columns

In [129…
```
mdf.isnull().sum()
```

Out[129…
```
city                   0
title                  0
carpet                 0
status                 0
furnishing             0
floor               1251
facing                 0
bathroom              52
overlooking         4668
pricing                0
pricing_sqcm           0
room_type              0
location               0
floor_no               0
pricing_crore          0
full_pricing           0
price_rupees           0
carpet_num             0
price_per_sqft_calc    0
dtype: int64
```

1) What is the average property price in each city?

In [131…
```
mdf.columns
```

Out[131…
```
Index(['city', 'title', 'carpet', 'status', 'furnishing', 'floor', 'facing',
       'bathroom', 'overlooking', 'pricing', 'pricing_sqcm', 'room_type',
       'location', 'floor_no', 'pricing_crore', 'full_pricing', 'price_rupees',
       'carpet_num', 'price_per_sqft_calc'],
      dtype='object')
```

In [132…
```
avg_city=mdf.groupby('city')['price_rupees'].mean().sort_values(ascending=False)
)
```

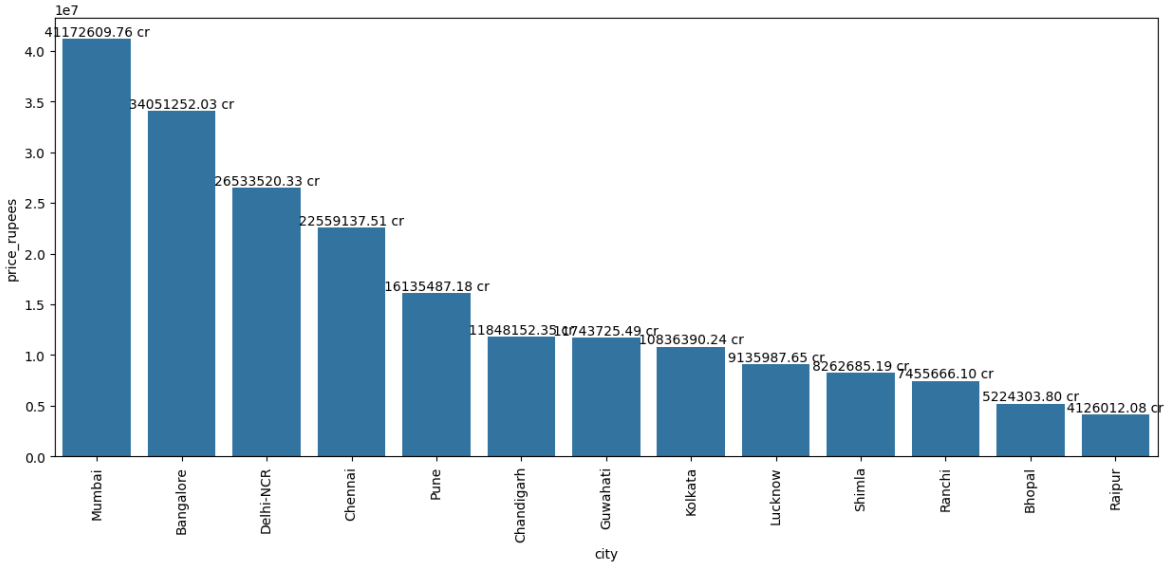
In [133…
```
pd.options.display.float_format = '{:,.0f}'.format

print(avg_city)
```

	city	price_rupees
0	Mumbai	41,172,610
1	Bangalore	34,051,252
2	Delhi-NCR	26,533,520
3	Chennai	22,559,138
4	Pune	16,135,487
5	Chandigarh	11,848,152
6	Guwahati	11,743,725
7	Kolkata	10,836,390
8	Lucknow	9,135,988
9	Shimla	8,262,685
10	Ranchi	7,455,666
11	Bhopal	5,224,304
12	Raipur	4,126,012

In [134…
```
plt.figure(figsize=(15,6))
ax=sns.barplot(data=avg_city,x="city",y="price_rupees")
plt.xticks(rotation=90)

for container in ax.containers:
    ax.bar_label(container,fmt = "%.2f cr",label_type="edge")
```



mumbai have the most hightest price rate with the avarge of 4 cr and raipur have the lowest price in this state

2) Which are the most common room types (1BHK, 2BHK, etc.)?

In [136…
```
room_counts = mdf["room_type"].value_counts().reset_index()
room_counts.columns = ["Room_Type", "Count"]
print(room_counts)
```

	Room_Type	Count
0	3 BHK	5090
1	2 BHK	4065
2	4 BHK	1015
3	1 BHK	782
4	5 BHK	166
5	6 BHK	29
6	10 BHK	12
7	8 BHK	4
8	7 BHK	4

plt.figure(figsize=(8,6)) sns.barplot(data=room_counts,x='Room_Type',y='Count')

most common room_type is 3 bhk

3)How does price per sqft vary by room type (BHK) in each city?

In [139…
```
price_bhk=mdf.groupby(["city","room_type"])['price_per_sqft_calc'].mean().sort_v
print(price_bhk)
```

```
city        room_type
Chennai     5 BHK       464,459
Shimla      10 BHK      103,448
Chandigarh  1 BHK        65,783
Mumbai      4 BHK        57,648
            5 BHK        54,810
                           ...
Bhopal      3 BHK         4,237
            2 BHK         3,791
Raipur      2 BHK         3,770
            1 BHK         2,982
Bhopal      6 BHK         2,609
Name: price_per_sqft_calc, Length: 82, dtype: Float64
```

In [140…
```
mdf.head()
```

Out[140…

	city	title	carpet	status	furnishing	floor	facing	bathroom	overlooki
0	Mumbai	3 BHK Flat for Sale in Sunteck City 4th Avenu...	1036 sqft	Poss. by Dec '25	Unfurnished	NaN	Not-mention	3	N
1	Mumbai	3 BHK Flat for Sale in Dosti Ambrosia, Wadala...	1156 sqft	Ready to Move	Furnished	NaN	North - West	3	Garden/Pa Main Rc
2	Mumbai	4 BHK Flat for Sale in Juhu, Mumbai	1775 sqft	Ready to Move	Semi-Furnished	4 out of 15	East	4	Main Rc
3	Mumbai	4 BHK Flat for Sale in Omkar Alta Monte, Mala...	3311 sqft	Ready to Move	Unfurnished	11 out of 54	East	4	Garden/Pa P
4	Mumbai	5 BHK Flat for Sale in Lodha Fiorenza, Gorega...	3500 sqft	Ready to Move	Unfurnished	NaN	North - West	7	Garden/Pa P

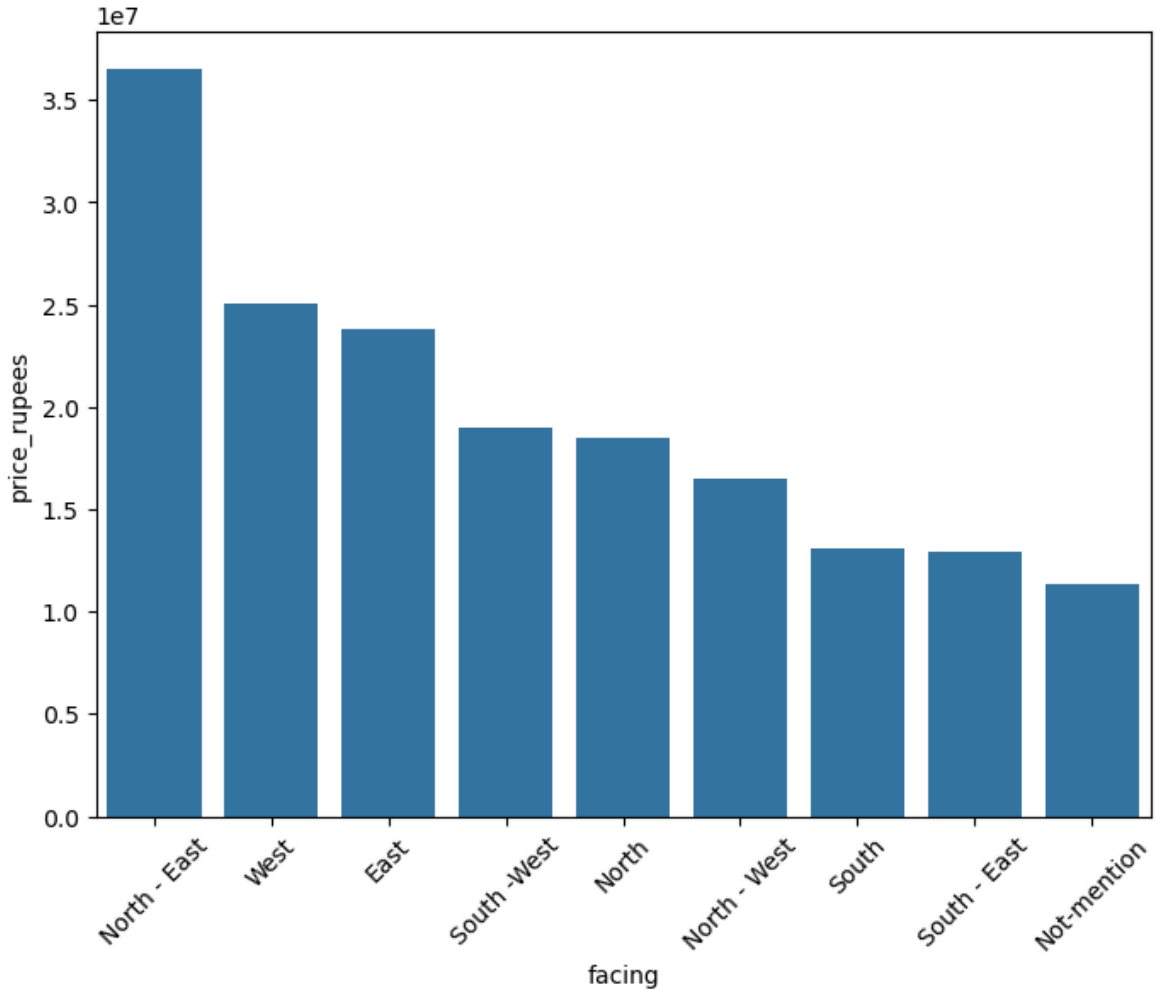
4)How does facing direction (East, North, etc.) affect pricing?

In [142…

```
direction =mdf.groupby('facing')['price_rupees'].mean().sort_values(ascending =
print(direction)
```

```
facing
North - East   36,557,200
West           25,025,171
East           23,845,263
South -West    19,009,197
North          18,518,253
North - West   16,497,444
South          13,051,591
South - East   12,914,766
Not-mention    11,319,914
Name: price_rupees, dtype: float64
```

In [143…
```
plt.figure(figsize=(8,6))
sns.barplot(data=direction)
plt.xticks(rotation = 45)
plt.show()
```



most common facing is north-east side and west

In [144…
```
mdf.columns
```

Out[144…
```
Index(['city', 'title', 'carpet', 'status', 'furnishing', 'floor', 'facing',
       'bathroom', 'overlooking', 'pricing', 'pricing_sqcm', 'room_type',
       'location', 'floor_no', 'pricing_crore', 'full_pricing', 'price_rupees',
       'carpet_num', 'price_per_sqft_calc'],
      dtype='object')
```

5) Which cities show the best price per sqft value (cheapest vs. costliest)?

In [192…
```
carpet_per_price=mdf.groupby("city")['price_per_sqft_calc'].mean().reset_index()

#make a bar chart for visualizations
plt.figure(figsize=(8,5))
sns.barplot(data=carpet_per_price,x='city',y='price_per_sqft_calc',palette="viri
plt.xticks(rotation = 90)
```

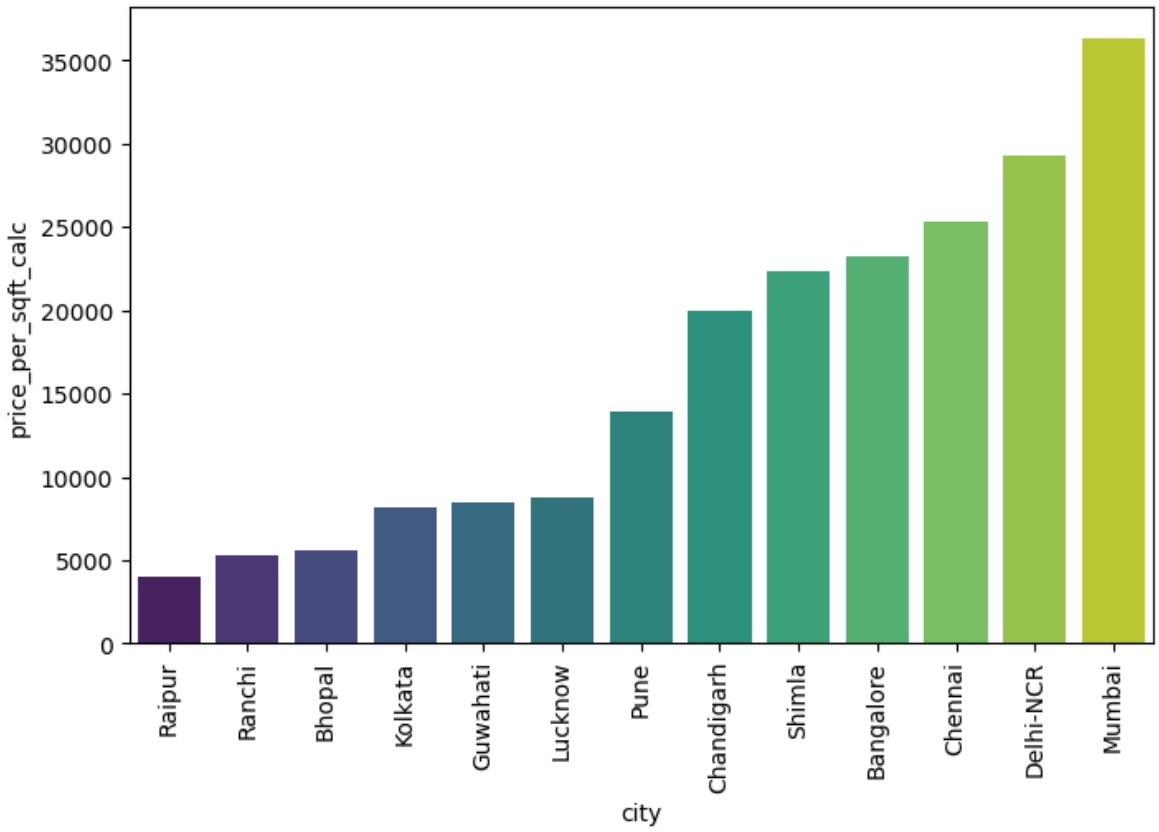
```
#cheapest vs costliest city
cheapest = carpet_per_price.iloc[-1]
costliest = carpet_per_price.iloc[0]
print(f"cheapest city per sqft {cheapest['city']}:{cheapest['price_per_sqft_calc
```

```
print(f"-------->costliest city per sqft {costliest['city']}:{costliest['price_p
```

```
C:\Users\46575\AppData\Local\Temp\ipykernel_14992\1097328906.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v
0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effe
ct.

  sns.barplot(data=carpet_per_price,x='city',y='price_per_sqft_calc',palette="vir
idis")
cheapest city per sqft Mumbai:36368/sqft
-------->costliest city per sqft Raipur:3997/sqft
```



In [194…

```
mdf.columns
```

Out[194…

```
Index(['city', 'title', 'carpet', 'status', 'furnishing', 'floor', 'facing',
       'bathroom', 'overlooking', 'pricing', 'pricing_sqcm', 'room_type',
       'location', 'floor_no', 'pricing_crore', 'full_pricing', 'price_rupees',
       'carpet_num', 'price_per_sqft_calc'],
      dtype='object')
```

6) Which city has the largest variation in property prices?

In [198…

```
variation=mdf.groupby('city')['price_rupees'].std().reset_index().sort_values(by
print(variation)
most_variations =variation.iloc[0]
print(f"######################    {most_variations['city']} have the largest var
```

```
          city  price_rupees
0    Bangalore   390,279,756
3      Chennai   379,720,193
4    Delhi-NCR    48,623,846
8       Mumbai    48,288,959
6      Kolkata    18,357,232
1       Bhopal    17,205,671
9         Pune    17,079,632
5     Guwahati    16,548,662
7      Lucknow     8,433,922
2   Chandigarh     8,141,774
12      Shimla     6,318,201
11      Ranchi     4,090,936
10      Raipur     2,883,735
######################    Bangalore have the largest variation    ###########
```

In [ ]: