Zomato Food Market Analysis: A Complete End-to-End Data Analytics Project

In [31]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings as w
w.filterwarnings('ignore')
```

Data Loading

In [2]:
```
df = pd.read_csv(r"C:\Users\papus\OneDrive\Documents\Data set .csv\zomato_dataset.csv")
df.head()
```

Out[2]:

	Restaurant_Name	Dining_Rating	Delivery_Rating	Dining_Votes	Delivery_Votes	Cuisine	Place_Name	City	Item_Name	Best_Seller	...	Is_Bestseller	Restaurant_Popularity	A
0	Doner King	3.9	4.2	39	0	Fast Food	Malakpet	Hyderabad	Platter Kebab Combo	BESTSELLER	...	1	46	
1	Doner King	3.9	4.2	39	0	Fast Food	Malakpet	Hyderabad	Chicken Rumali Shawarma	BESTSELLER	...	1	46	
2	Doner King	3.9	4.2	39	0	Fast Food	Malakpet	Hyderabad	Chicken Tandoori Salad	NONE	...	1	46	
3	Doner King	3.9	4.2	39	0	Fast Food	Malakpet	Hyderabad	Chicken BBQ Salad	BESTSELLER	...	1	46	
4	Doner King	3.9	4.2	39	0	Fast Food	Malakpet	Hyderabad	Special Doner Wrap Combo	MUST TRY	...	1	46	

5 rows × 26 columns

In [4]:
```
df.shape
```

Out[4]:
```
(123657, 26)
```

In [5]:
```
df.columns
```

Out[5]:
```
Index(['Restaurant_Name', 'Dining_Rating', 'Delivery_Rating', 'Dining_Votes',
       'Delivery_Votes', 'Cuisine', 'Place_Name', 'City', 'Item_Name',
       'Best_Seller', 'Votes', 'Prices', 'Average_Rating', 'Total_Votes',
       'Price_per_Vote', 'Log_Price', 'Is_Bestseller', 'Restaurant_Popularity',
       'Avg_Rating_Restaurant', 'Avg_Price_Restaurant', 'Avg_Rating_Cuisine',
       'Avg_Price_Cuisine', 'Avg_Rating_City', 'Avg_Price_City',
       'Is_Highly_Rated', 'Is_Expensive'],
      dtype='object')
```

In [6]:
```
df.dtypes
```

Out[6]:
```
Restaurant_Name           object
Dining_Rating            float64
Delivery_Rating          float64
Dining_Votes               int64
Delivery_Votes             int64
Cuisine                   object
Place_Name                object
City                      object
Item_Name                 object
Best_Seller               object
Votes                      int64
Prices                   float64
Average_Rating           float64
Total_Votes                int64
Price_per_Vote           float64
Log_Price                float64
Is_Bestseller              int64
Restaurant_Popularity      int64
Avg_Rating_Restaurant    float64
Avg_Price_Restaurant     float64
Avg_Rating_Cuisine       float64
Avg_Price_Cuisine        float64
Avg_Rating_City          float64
Avg_Price_City           float64
Is_Highly_Rated            int64
Is_Expensive               int64
dtype: object
```

Data Cleaning

In [8]:
```
df.isnull()
```

Out[8]:

	Restaurant_Name	Dining_Rating	Delivery_Rating	Dining_Votes	Delivery_Votes	Cuisine	Place_Name	City	Item_Name	Best_Seller	...	Is_Bestseller	Restaurant_Popularity	A
0	False	False	False	False	False	False	False	False	False	False	...	False	False	
1	False	False	False	False	False	False	False	False	False	False	...	False	False	
2	False	False	False	False	False	False	False	False	False	False	...	False	False	
3	False	False	False	False	False	False	False	False	False	False	...	False	False	
4	False	False	False	False	False	False	False	False	False	False	...	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
123652	False	False	False	False	False	False	False	False	False	False	...	False	False	
123653	False	False	False	False	False	False	False	False	False	False	...	False	False	
123654	False	False	False	False	False	False	False	False	False	False	...	False	False	
123655	False	False	False	False	False	False	False	False	False	False	...	False	False	
123656	False	False	False	False	False	False	False	False	False	False	...	False	False	

123657 rows × 26 columns

In [11]:
```
print(df.duplicated().sum())
```

```
22127
```

In [12]:
```
df = df.drop_duplicates()
```

In [13]:
```
print("Number of duplicates : ",df.duplicated().sum())
print("Shape of Dataset : ",df.shape)
```

```
Number of duplicates :  0
Shape of Dataset :  (101530, 26)
```

In [14]:
```
df.nunique()
```

Out[14]:
```
Restaurant_Name            826
Dining_Rating               25
Delivery_Rating             19
Dining_Votes               294
Delivery_Votes             263
Cuisine                     48
Place_Name                 324
City                        17
Item_Name                55693
Best_Seller                 14
Votes                      760
Prices                    2710
Average_Rating              48
Total_Votes                450
Price_per_Vote           13483
Log_Price                 2710
Is_Bestseller                1
Restaurant_Popularity      308
Avg_Rating_Restaurant       91
Avg_Price_Restaurant       824
Avg_Rating_Cuisine          46
Avg_Price_Cuisine           48
Avg_Rating_City             17
Avg_Price_City              17
Is_Highly_Rated              2
Is_Expensive                 2
dtype: int64
```

In [15]:
```
df.describe()
```

Out[15]:

	Dining_Rating	Delivery_Rating	Dining_Votes	Delivery_Votes	Votes	Prices	Average_Rating	Total_Votes	Price_per_Vote	Log_Price	Is_Bestseller	Restau
count	101530.000000	101530.000000	101530.000000	101530.000000	101530.000000	101530.000000	101530.000000	101530.000000	101530.000000	101530.000000	101530.0	
mean	3.821385	3.959565	151.559726	116.704009	19.110371	243.690758	3.890475	268.263735	170.173815	5.271124	1.0	
std	0.350058	0.244990	230.662157	243.856446	108.264166	197.534461	0.238118	291.518139	212.517776	0.704622	0.0	
min	2.500000	2.500000	0.000000	0.000000	0.000000	0.950000	3.000000	0.000000	0.006586	0.667829	1.0	
25%	3.700000	3.800000	0.000000	0.000000	0.000000	130.000000	3.750000	11.000000	16.000000	4.875197	1.0	
50%	3.822264	4.000000	30.000000	0.000000	0.000000	209.000000	3.911132	162.000000	127.120000	5.347108	1.0	
75%	4.000000	4.100000	221.000000	37.000000	10.000000	299.000000	4.050000	445.000000	259.000000	5.703782	1.0	
max	4.800000	4.600000	997.000000	983.000000	9750.000000	12024.000000	4.650000	1393.000000	12024.000000	9.394743	1.0	

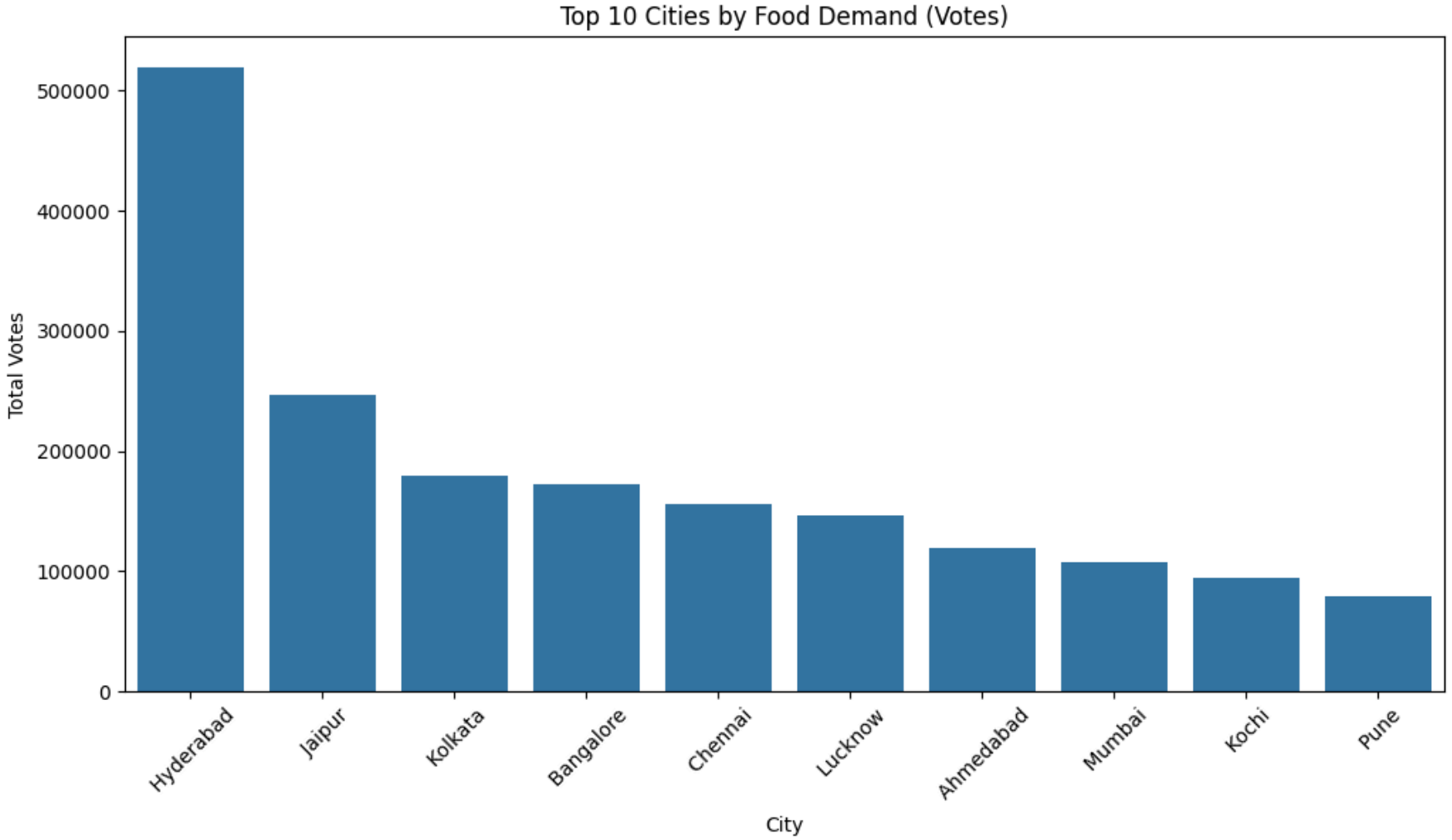
Solving Buisness Questions

Q.1::Which cities generate the highest food demand based on votes and reviews?

In [18]:
```
city_demand = df.groupby("City")["Votes"].sum().sort_values(ascending=False).head(10)
print (city_demand)
```

```
City
Hyderabad    518762
Jaipur       247204
Kolkata      179973
Bangalore    172303
Chennai      155828
Lucknow      145917
Ahmedabad    119334
Mumbai       107860
Kochi         94486
Pune          79020
Name: Votes, dtype: int64
```

In [19]:
```
plt.figure(figsize=(12,6))
sns.barplot(x=city_demand.index, y=city_demand.values)
plt.xticks(rotation=45)
plt.title("Top 10 Cities by Food Demand (Votes)")
plt.xlabel("City")
plt.ylabel("Total Votes")
plt.show()
```



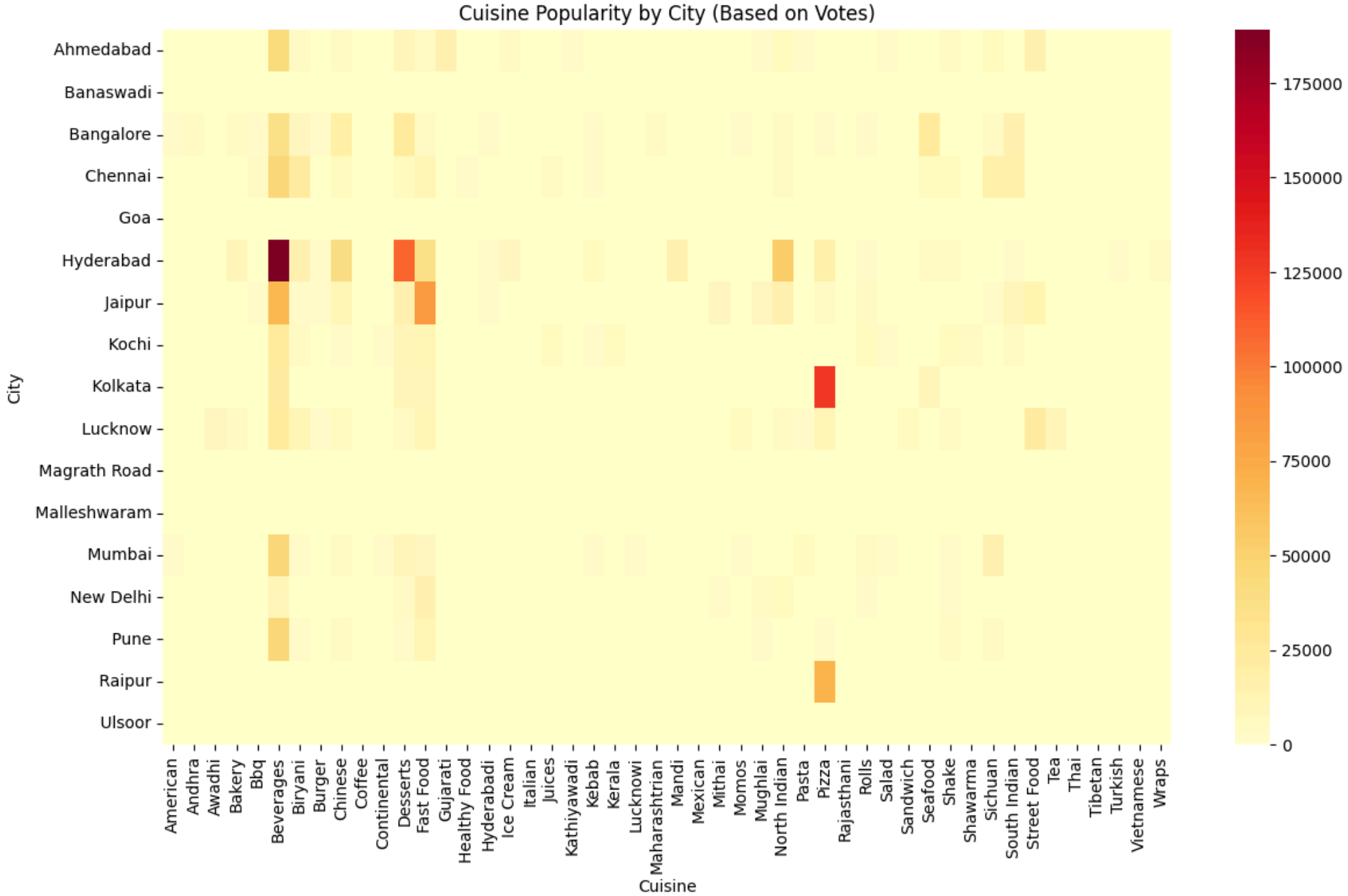
Q.2:: Which cuisines are most popular in each city?

In [23]:
```
city_cuisine_popularity = df.groupby(["City", "Cuisine"])["Votes"].sum().reset_index()
top_city_cuisine = city_cuisine_popularity.pivot(index="City", columns="Cuisine", values="Votes").fillna(0)
print (city_cuisine_popularity)
```

```
           City    Cuisine  Votes
0     Ahmedabad   American     28
1     Ahmedabad  Beverages  42628
2     Ahmedabad    Biryani   3391
3     Ahmedabad    Chinese   5024
4     Ahmedabad   Desserts  10146
..          ...        ...    ...
211        Pune      Pizza   1050
212        Pune      Shake   4878
213        Pune    Sichuan   5057
214      Raipur      Pizza  70009
215      Ulsoor   Desserts      0

[216 rows x 3 columns]
```

In [24]:

```
plt.figure(figsize=(14,8))
sns.heatmap(top_city_cuisine, cmap="YlOrRd")
plt.title("Cuisine Popularity by City (Based on Votes)")
plt.xlabel("Cuisine")
plt.ylabel("City")
plt.show()
```

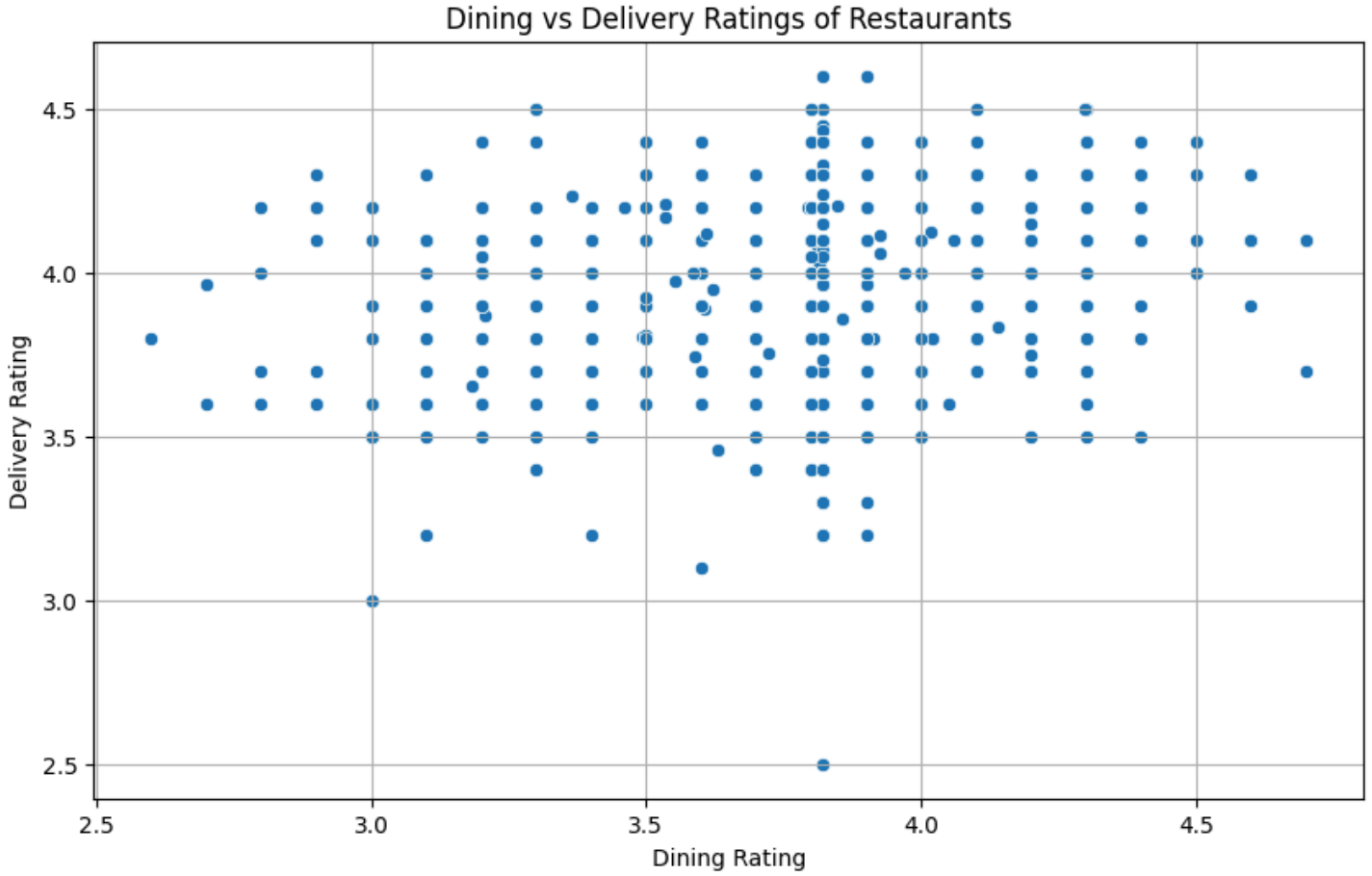


Q.3::Which restaurants have the highest dining vs delivery ratings?

In [25]:
```
top_resturants = df.groupby("Restaurant_Name")[["Dining_Rating", "Delivery_Rating"]].mean().reset_index()
print(top_resturants)
```

```
             Restaurant_Name  Dining_Rating  Delivery_Rating
0               12 To 12 BBQ       4.000000              4.0
1              1441 Pizzeria       3.822264              4.0
2     1944 -The HOCCO Kitchen       4.300000              4.3
3            4M Biryani House       4.100000              4.2
4                    7 Plates       3.822264              4.3
..                        ...            ...              ...
821    Zaffran Mataam Alarabi       4.100000              4.1
822           Zam Zam Briyani       3.300000              3.4
823       Zam Zam Restaurant       4.200000              4.3
824    Zeeshan Biryani Corner       3.700000              4.0
825  Zomoz - The Momo Company       3.822264              4.3

[826 rows x 3 columns]
```

In [27]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(data=top_resturants, x="Dining_Rating", y="Delivery_Rating")
plt.title("Dining vs Delivery Ratings of Restaurants")
plt.xlabel("Dining Rating")
plt.ylabel("Delivery Rating")
plt.grid(True)
plt.show()
```



Q.4::How does price impact dining_ratings?

In [33]:
```
# Grouping prices into 4 segments to compare ratings
df["price_group"] = pd.qcut(df["Prices"], 4, labels=["Low", "Medium", "High", "Premium"])
price_dining_summary = df.groupby("price_group")["Dining_Rating"].mean()
print(price_dining_summary)

# Correlation between price and dining rating
price_dining_corr = df["Prices"].corr(df["Dining_Rating"])
print("Correlation between Price and Dining Rating:", price_dining_corr)
```

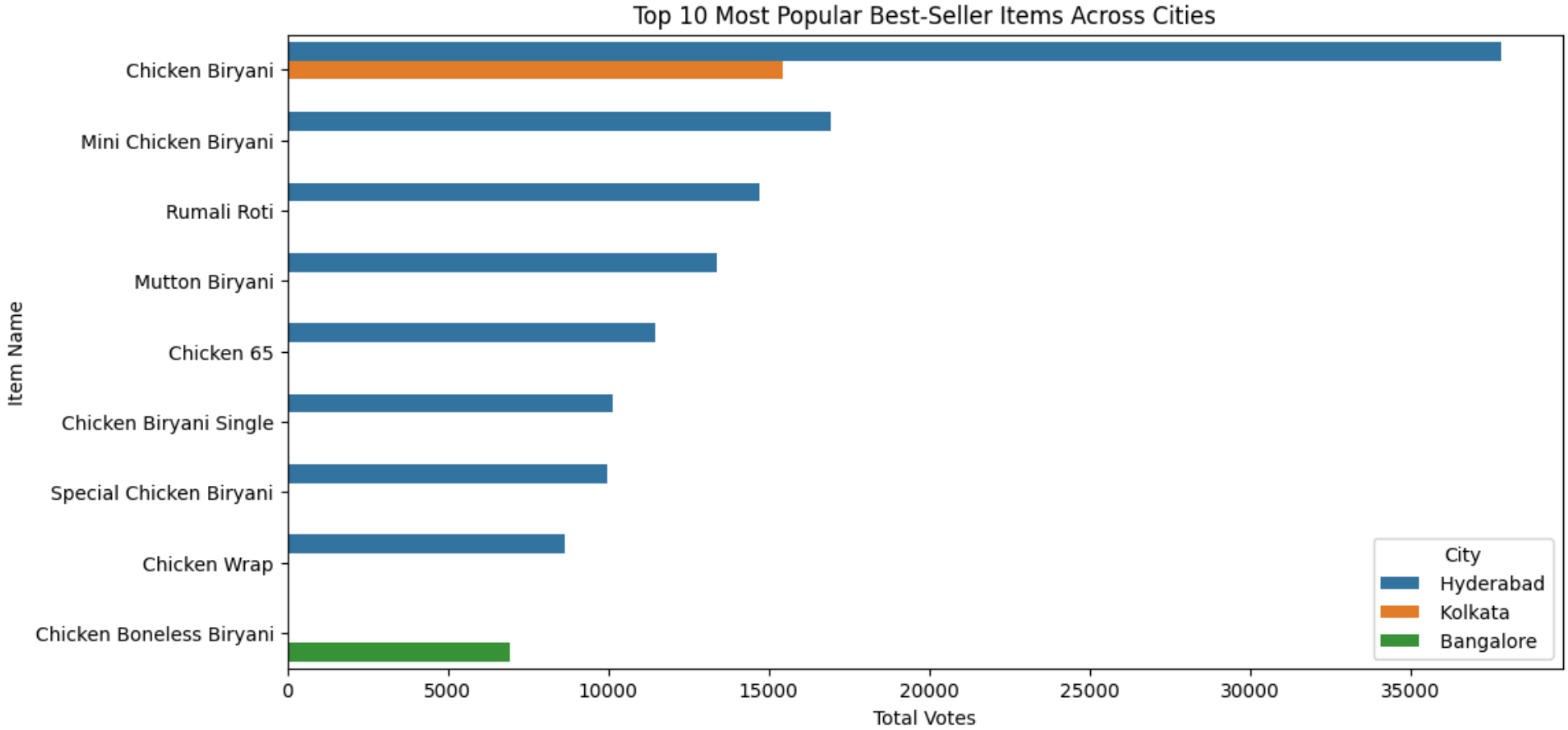
```
price_group
Low        3.776778
Medium     3.812564
High       3.837506
Premium    3.859165
Name: Dining_Rating, dtype: float64
Correlation between Price and Dining Rating: 0.06098472957398584
```

In [34]:
```
price_dining_summary.plot(kind="line", marker="o", figsize=(10,6))
plt.title("Dining Rating by Price Segments")
plt.xlabel("Price Segment")
plt.ylabel("Average Dining Rating")
plt.grid(True)
plt.show()
```

Q.5::What are the most popular best-seller items across cities?

In [42]:
```
df["Best_Seller"] = df["Best_Seller"].astype(str).str.lower()

best_sellers = df[df["Best_Seller"].isin(["yes", "y", "1", "true"])]

if best_sellers.empty:
    votes_cutoff = df["Votes"].quantile(0.80)
    best_sellers = df[df["Votes"] >= votes_cutoff]

best_item_popularity = best_sellers.groupby(["City", "Item_Name"])["Votes"].sum().reset_index()

# Top 10 best-sellers
top_best_sellers = best_item_popularity.sort_values(by="Votes", ascending=False).head(10)
print(top_best_sellers)

# Visualization – Horizontal bar chart
plt.figure(figsize=(12,6))
sns.barplot(data=top_best_sellers, y="Item_Name", x="Votes", hue="City")
plt.title("Top 10 Most Popular Best-Seller Items Across Cities")
plt.xlabel("Total Votes")
plt.ylabel("Item Name")
plt.legend(title="City")
plt.show()
```

```
           City                Item_Name  Votes
4404   Hyderabad          Chicken Biryani  37833
5332   Hyderabad      Mini Chicken Biryani  16914
8687     Kolkata          Chicken Biryani  15416
5706   Hyderabad              Rumali Roti  14713
5387   Hyderabad            Mutton Biryani  13398
4366   Hyderabad                Chicken 65  11460
4421   Hyderabad    Chicken Biryani Single  10134
5771   Hyderabad   Special Chicken Biryani   9983
4675   Hyderabad             Chicken Wrap   8657
1446   Bangalore  Chicken Boneless Biryani   6940
```

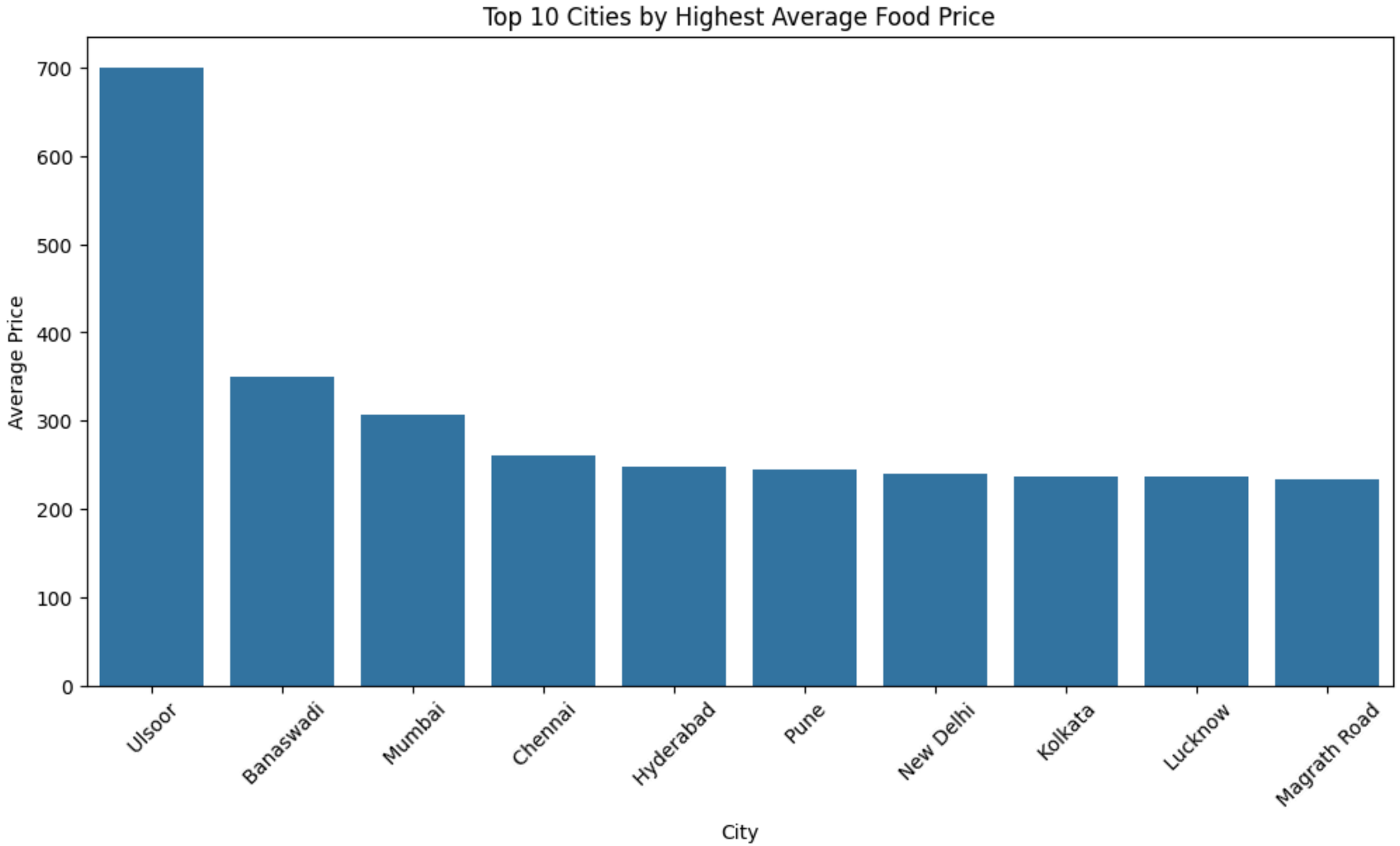


Q.6:: Which cities have the highest-priced food on average?

In [45]:
```
city_price = df.groupby("City")["Prices"].mean().sort_values(ascending=False)
print(city_price.head(10))
```

```
City
Ulsoor          700.389831
Banaswadi       349.466471
Mumbai          306.371528
Chennai         260.513679
Hyderabad       248.560141
Pune            245.113393
New Delhi       240.408962
Kolkata         237.214453
Lucknow         236.642409
Magrath Road    234.447111
Name: Prices, dtype: float64
```

In [48]:
```
plt.figure(figsize=(12,6))
sns.barplot(x=city_price.head(10).index, y=city_price.head(10).values)
plt.xticks(rotation=45)
plt.title("Top 10 Cities by Highest Average Food Price")
plt.xlabel("City")
plt.ylabel("Average Price")
plt.show()
```



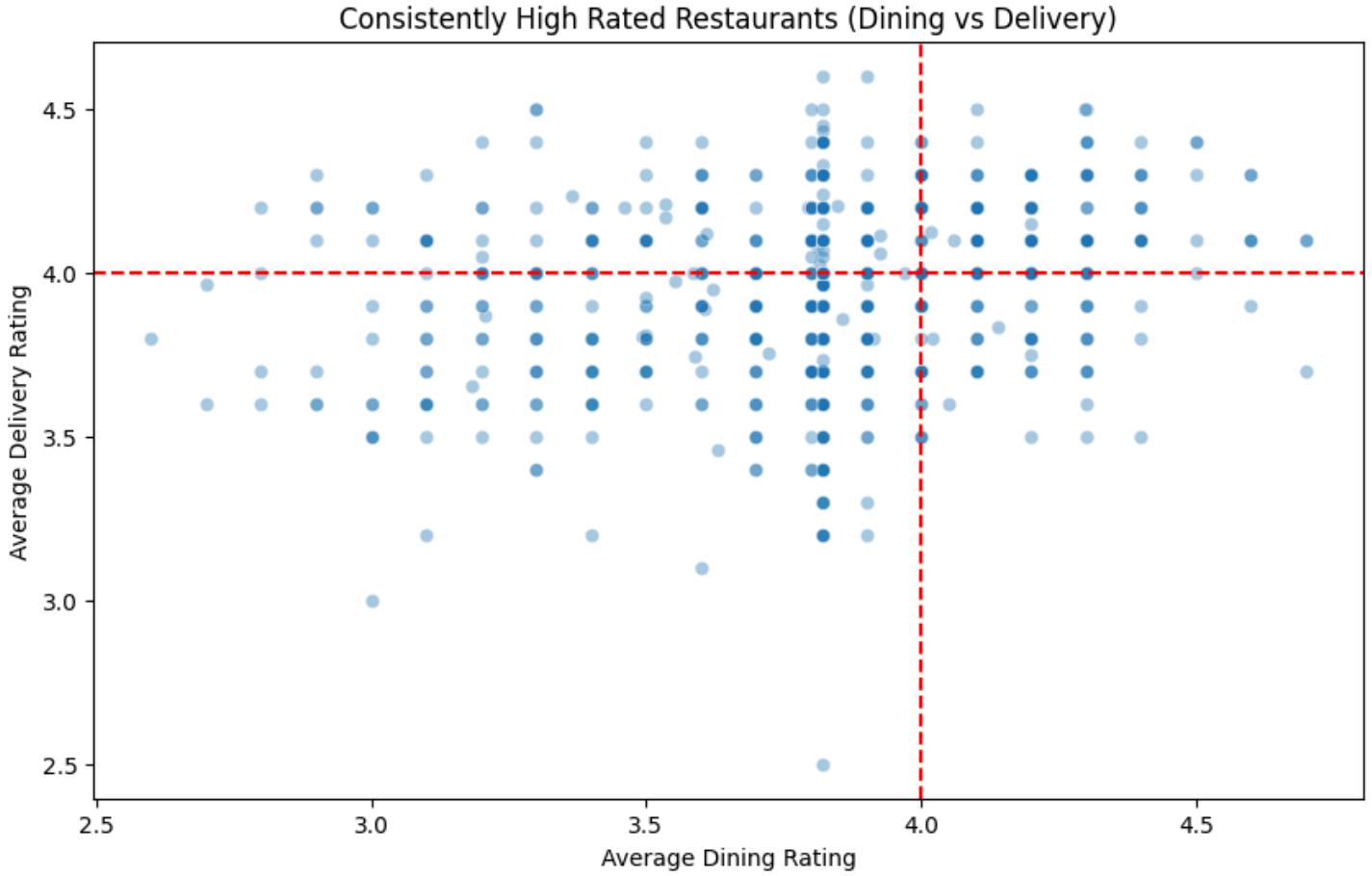
Q.7::Which restaurant is consistently rated high in both dining & delivery?

In [49]:
```
# Average dining & delivery rating per restaurant
rating_consistency = df.groupby("Restaurant_Name")[["Dining_Rating", "Delivery_Rating"]].mean().reset_index()
```

```
# Filter restaurants consistently above 4.0 in both
consistent_top = rating_consistency[
    (rating_consistency["Dining_Rating"] >= 4.0) &
    (rating_consistency["Delivery_Rating"] >= 4.0)
].sort_values(by=["Dining_Rating", "Delivery_Rating"], ascending=False)

print(consistent_top.head(10))
```

```
    Restaurant_Name  Dining_Rating  Delivery_Rating
729   Thali and More            4.7              4.1
782          Toscano            4.7              4.1
222          Exotica            4.6              4.3
785         Truffles            4.6              4.3
130          Cafe 17            4.6              4.1
218   Eating Circles            4.6              4.1
795    Urban Khichdi            4.6              4.1
155        Chaitanya            4.5              4.4
387      Kings Kulfi            4.5              4.4
107      Boojee Cafe            4.5              4.3
```

In [50]:
```
plt.figure(figsize=(10,6))
sns.scatterplot(data=rating_consistency, x="Dining_Rating", y="Delivery_Rating", alpha=0.4)

# Reference lines for consistency criteria
plt.axvline(4.0, color='red', linestyle='--')
plt.axhline(4.0, color='red', linestyle='--')

plt.title("Consistently High Rated Restaurants (Dining vs Delivery)")
plt.xlabel("Average Dining Rating")
plt.ylabel("Average Delivery Rating")
plt.show()
```

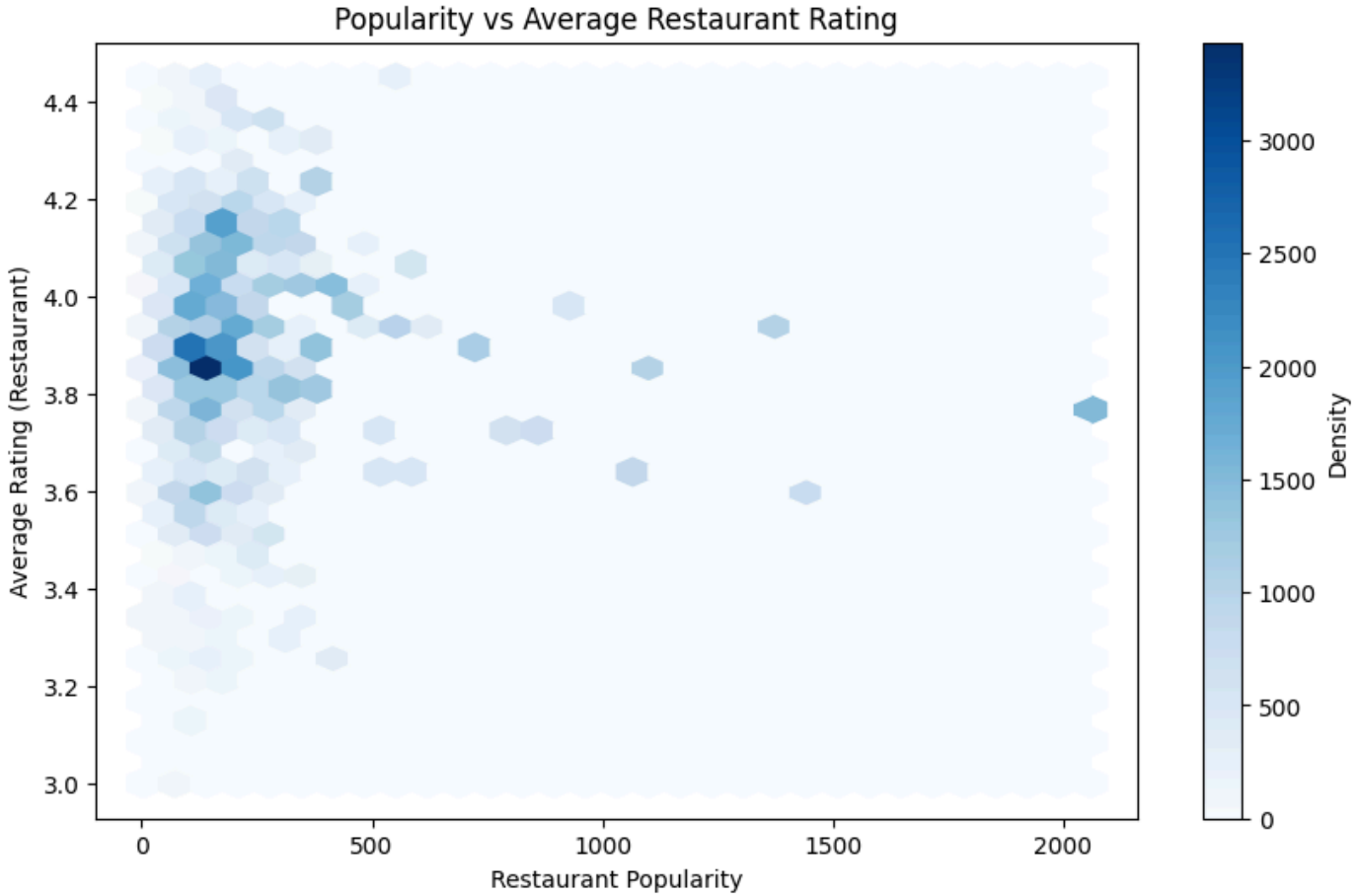


Q.8::Does higher restaurant popularity lead to better average ratings?

In [51]:
```
# Correlation check
popularity_corr = df["Restaurant_Popularity"].corr(df["Avg_Rating_Restaurant"])
print("Correlation between Restaurant Popularity and Avg Rating:", popularity_corr)
```

```
Correlation between Restaurant Popularity and Avg Rating: -0.08139211764357954
```

In [54]:
```
plt.figure(figsize=(10,6))
plt.hexbin(df["Restaurant_Popularity"], df["Avg_Rating_Restaurant"], gridsize=30, cmap="Blues")
plt.colorbar(label="Density")
plt.title("Popularity vs Average Restaurant Rating")
plt.xlabel("Restaurant Popularity")
plt.ylabel("Average Rating (Restaurant)")
plt.show()
```



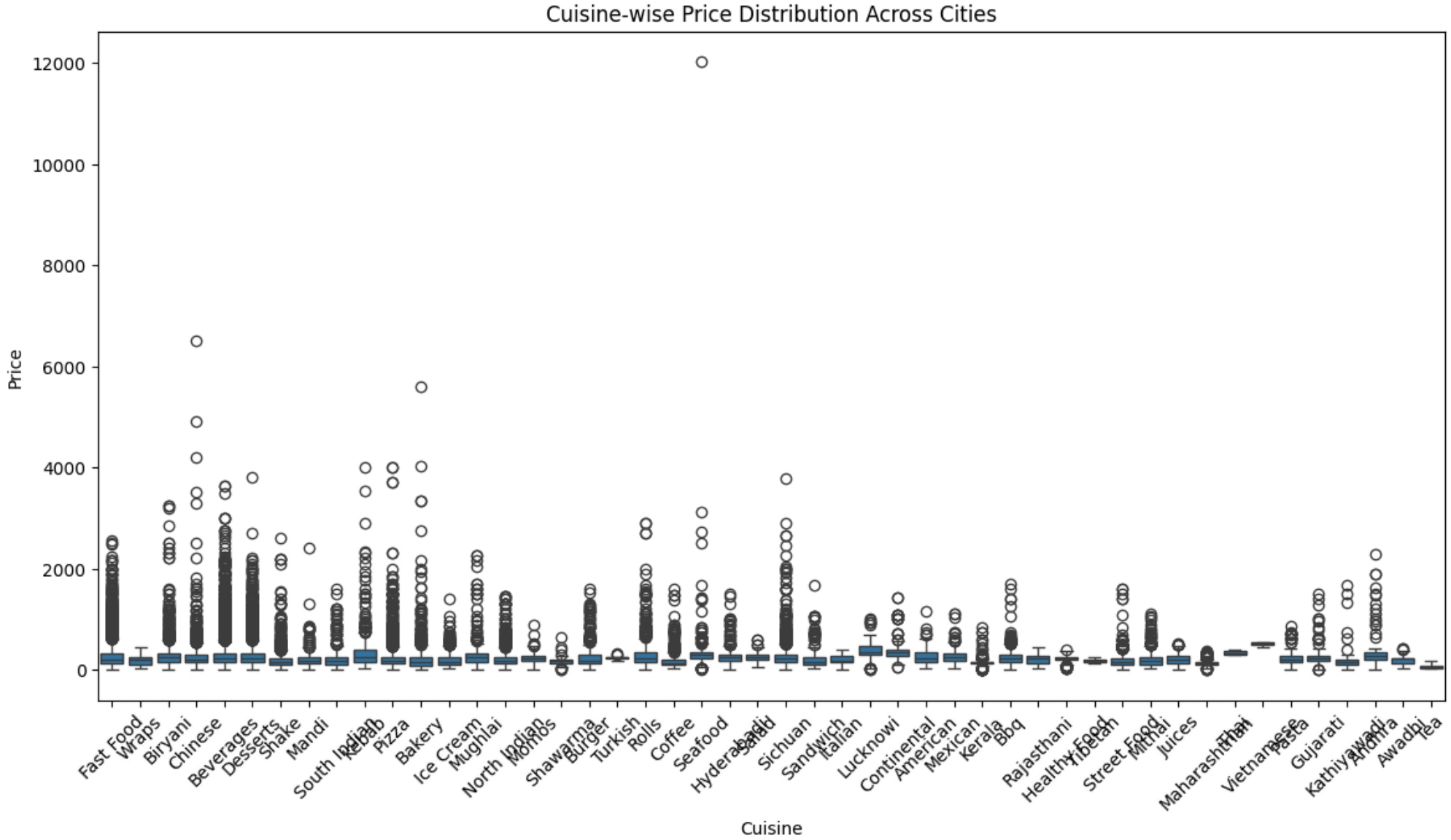
Q.9:: Which cuisines are considered expensive vs cheap across cities?

In [55]:
```
# Average price per cuisine across cities
cuisine_price = df.groupby("Cuisine")["Prices"].mean().sort_values(ascending=False)
print(cuisine_price.head(10))  # Most expensive cuisines
print(cuisine_price.tail(10))  # Cheapest cuisines
```

```
Cuisine
Vietnamese     512.500000
Andhra         399.505051
Lucknowi       376.963158
Continental    353.618216
Kebab          338.252466
Thai           336.500000
Seafood        299.275794
Rolls          277.747220
Mexican        272.376437
Biryani        271.045043
Name: Prices, dtype: float64
Cuisine
South Indian     189.309854
Shake            188.431239
Ice Cream        187.272308
Tibetan          173.705882
Street Food      172.497204
Shawarma         167.770701
Kathiyawadi      163.436975
Kerala           150.677215
Maharashtrian    131.048443
Tea               63.461538
Name: Prices, dtype: float64
```

In [56]:

```
plt.figure(figsize=(14,7))
sns.boxplot(data=df, x="Cuisine", y="Prices")
plt.xticks(rotation=45)
plt.title("Cuisine-wise Price Distribution Across Cities")
plt.xlabel("Cuisine")
plt.ylabel("Price")
plt.show()
```

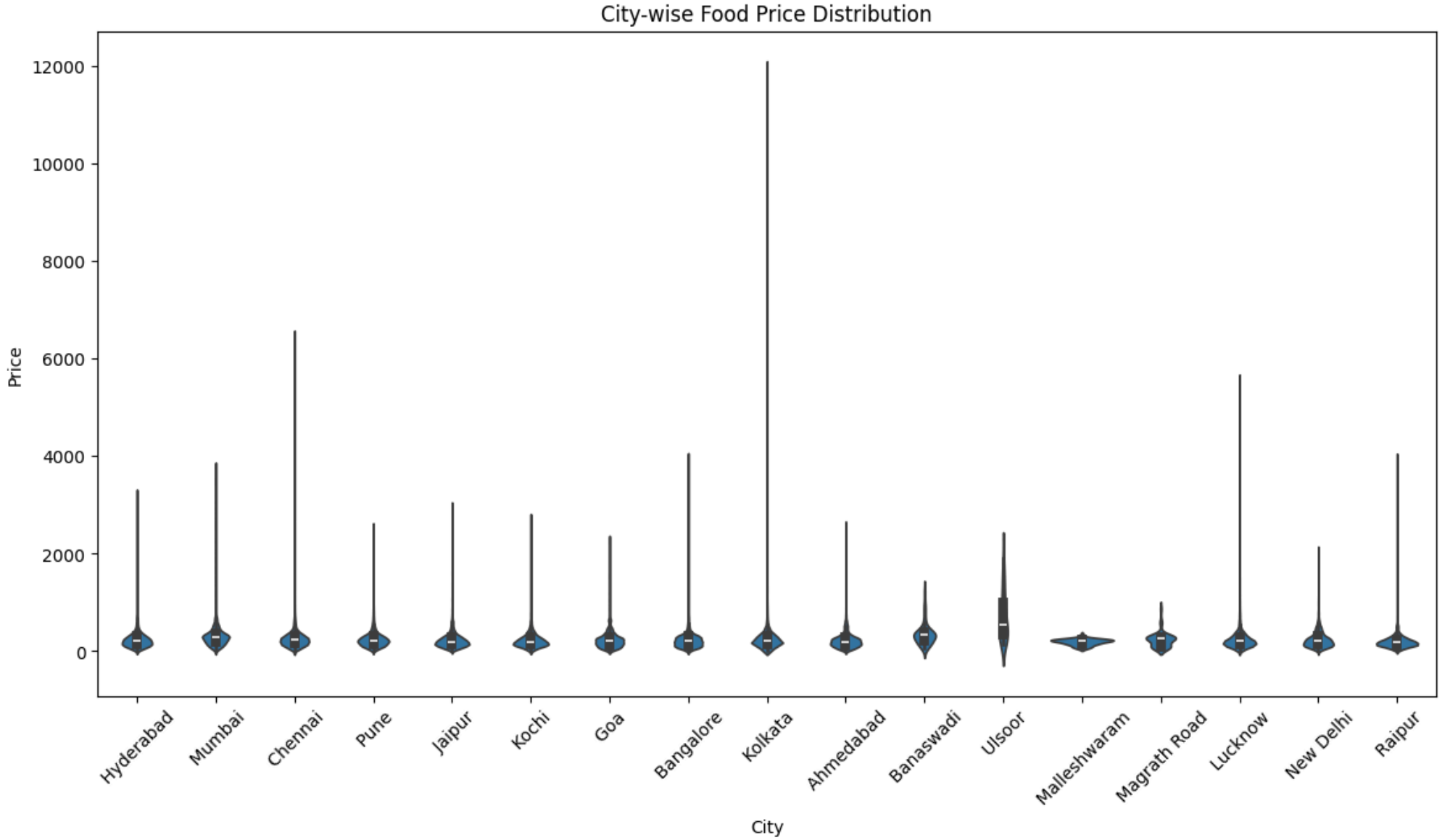


Q.10::Which cities have more expensive food markets overall?

In [57]:
```
# Average price per city
city_price_avg = df.groupby("City")["Prices"].mean().sort_values(ascending=False)
print(city_price_avg.head(10))  # Most expensive cities
```

```
City
Ulsoor          700.389831
Banaswadi       349.466471
Mumbai          306.371528
Chennai         260.513679
Hyderabad       248.560141
Pune            245.113393
New Delhi       240.408962
Kolkata         237.214453
Lucknow         236.642409
Magrath Road    234.447111
Name: Prices, dtype: float64
```

In [58]:
```
plt.figure(figsize=(14,7))
sns.violinplot(data=df, x="City", y="Prices")
plt.xticks(rotation=45)
plt.title("City-wise Food Price Distribution")
plt.xlabel("City")
plt.ylabel("Price")
plt.show()
```



In [65]:
```
from sqlalchemy import create_engine

username = "postgres"

password = "Papu1993"

host     = "localhost"

port     = "5432"
```

```
database = "Zomato_Data_Analysis"

engine = create_engine(
    f"postgresql+psycopg2://{username}:{password}@{host}:{port}/{database}"
)

table_name = "zomato_data"

df.to_sql(table_name, engine, if_exists="replace", index=False)

print(f"Data successfully loaded into table: {table_name}")
```

```
Data successfully loaded into table: zomato_data
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: