

Swiggy Sales Analysis

Import Libraries

```
In [9]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

Import Data

```
In [10]: df = pd.read_excel("D:/Python Data Science Projects/swiggy_data.xlsx")
```

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

</

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

197425	Sikkim	Gangtok	2025-01-25	Mama's Kitchen	Gangtok	Momos	Soya cheese chilli momo ...	112.0	4.4	0
197426	Sikkim	Gangtok	2025-07-02	Mama's Kitchen	Gangtok	Momos	Kurkure momo fried ...	140.0	4.4	0
197427	Sikkim	Gangtok	2025-03-25	Mama's Kitchen	Gangtok	Momos	Chilli cheese momo	128.0	4.4	0
197428	Sikkim	Gangtok	2025-03-26	Mama's Kitchen	Gangtok	Momos	Veg Momos (8 Pc)	85.0	4.4	0
197429	Sikkim	Gangtok	2025-03-27	Mama's Kitchen	Gangtok	Momos	Soya Momo	100.0	4.4	0

Metadata

```
In [15]: print('The number of rows:', df.shape[0])
The number of rows 197430

In [16]: print('The number of fields:', df.shape[1])
The number of fields 10

In [18]: df.info
```

Metadata

```
In [15]: print(f"The number of rows:", df.shape[0])
The number of rows 197430
```

```
In [16]: print(f"The number of fields:", df.shape[1])
The number of fields 10
```

```
In [18]: df.info
```

```
Out[18]: <bound method DataFrame.info of
0      Karnataka  Bengaluru  2025-06-29      Anand Sweets & Savouries
1      Karnataka  Bengaluru  2025-04-03      Srinidhi Sagar Deluxe
2      Karnataka  Bengaluru  2025-01-15      Srinidhi Sagar Deluxe
3      Karnataka  Bengaluru  2025-04-17      Srinidhi Sagar Deluxe
4      Karnataka  Bengaluru  2025-03-13      Srinidhi Sagar Deluxe
...
197425  Sikkim      Gangtok  2025-01-25      Mama's Kitchen
197426  Sikkim      Gangtok  2025-07-02      Mama's Kitchen
197427  Sikkim      Gangtok  2025-03-25      Mama's Kitchen
197428  Sikkim      Gangtok  2025-03-26      Mama's Kitchen
197429  Sikkim      Gangtok  2025-03-27      Mama's Kitchen
...
0      Rajarajeshwari Nagar      Snack
1      Kengeri      Recommended
2      Kengeri      Recommended
3      Kengeri      Recommended
4      Kengeri      Recommended
...
197425      Gangtok      Momos
197426      Gangtok      Momos
197427      Gangtok      Momos
197428      Gangtok      Momos
197429      Gangtok      Momos
...
0      Dish Name  Price (INR) \
1      Butter Murukku-200gm  133.9
2      Badam Milk  52.0
3      Chow Chow Bath  117.0
4      Kesar Bath  65.0
...
197425  Soya cheese chilli momo  ...  112.0
197426  Kurkure momo fried ...  140.0
197427  Chilli cheese momo  ...  128.0
197428  Veg Momos (8 Pc)  85.0
197429  Soya Momo  100.0
...
0      Rating  Rating Count
1      4.0  0
2      4.5  25
3      4.7  48
4      4.6  65
...
197425  4.4  0
197426  4.4  0
197427  4.4  0
197428  4.4  0
197429  4.4  0
[197430 rows x 10 columns]>
```

Data Types

```
In [19]: df.dtypes
```

```
Out[19]: State      object
City      object
Order Date  datetime64[ns]
Restaurant Name  object
Location    object
Category    object
Dish Name   object
Price (INR) float64
Rating      float64
Rating Count int64
dtype: object

In [20]: df.describe()
```

	Order Date	Price (INR)	Rating	Rating Count
count	197430	197430.000000	197430.000000	197430.000000
mean	2025-05-01 19:41:20.996808960	268.512820	4.341582	28.321805
min	2025-01-01 00:00:00	0.950000	1.500000	0.000000
25%	2025-03-01 00:00:00	139.000000	4.300000	0.000000
50%	2025-05-02 00:00:00	229.000000	4.400000	2.000000
75%	2025-07-01 00:00:00	329.000000	4.500000	15.000000
max	2025-08-31 00:00:00	8000.000000	5.000000	999.000000
std	NaN	219.338363	0.422585	87.542593

KPI's

Total Sales

```
In [22]: total_sales = df["Price (INR)"].sum()
print(f"Total Sales (INR):", round(total_sales,2))
```

Total Sales (INR): 53012505.77

Average Rating

```
In [23]: average_rating = df["Rating"].mean()
print(f"Average Rating:", round(average_rating,2))
```

Average Rating: 4.34

Average Order Value

```
In [26]: avg_order_value = df["Price (INR)"].mean()
print(f"Avg Order Value (INR):", round(avg_order_value,2))
```

Avg Order Value (INR): 269.51

Ratings Count

```
In [27]: ratings_count = df["Rating Count"].sum()
print(f"Ratings Count (INR):", round(ratings_count,2))
```

Ratings Count (INR): 5591574

Total Orders

```
In [29]: total_orders = len(df)
print(f"Total_Orders:", round(total_orders,2))
```

Total_Orders: 197430

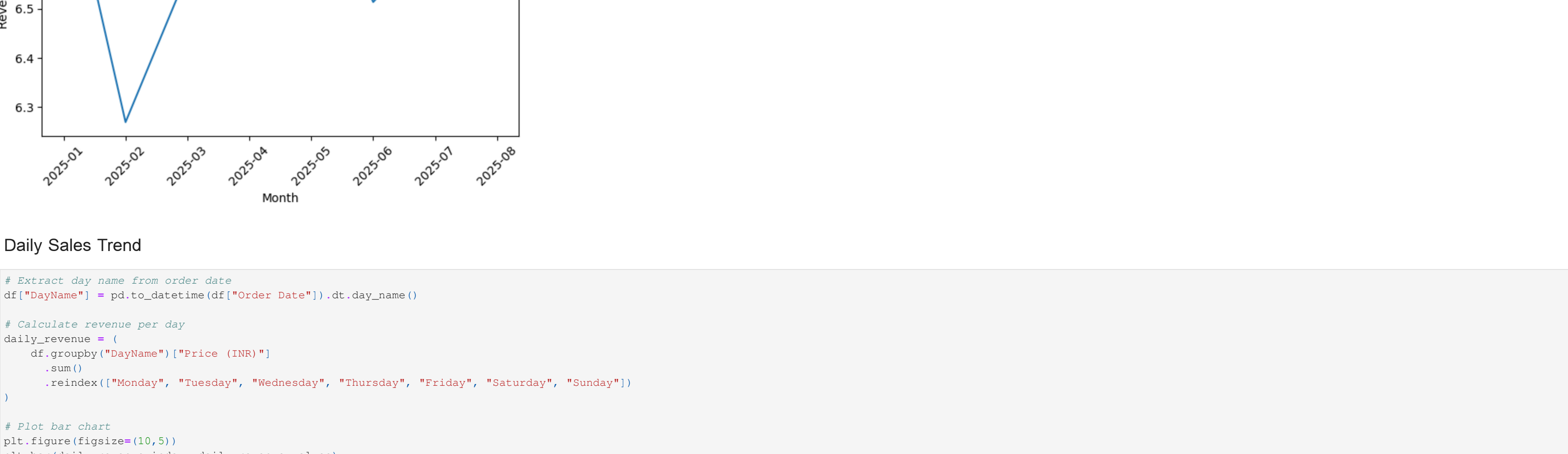
CHARTS DESIGN

```
In [30]: # Convert Order Date to datetime
df["Order Date"] = pd.to_datetime(df["Order Date"])

# Create Year-Month column
df["YearMonth"] = df["Order Date"].dt.to_period("M").astype(str)

# Calculate monthly revenue
monthly_revenue = df.groupby("YearMonth")["Price (INR)"].sum().reset_index()

# Plot monthly revenue trend
plt.figure()
plt.plot(monthly_revenue["YearMonth"], monthly_revenue["Price (INR)"])
plt.xticks(rotation=45)
plt.xlabel("Month")
plt.ylabel("Revenue (INR)")
plt.title("Monthly Revenue Trend")
plt.tight_layout()
plt.show()
```

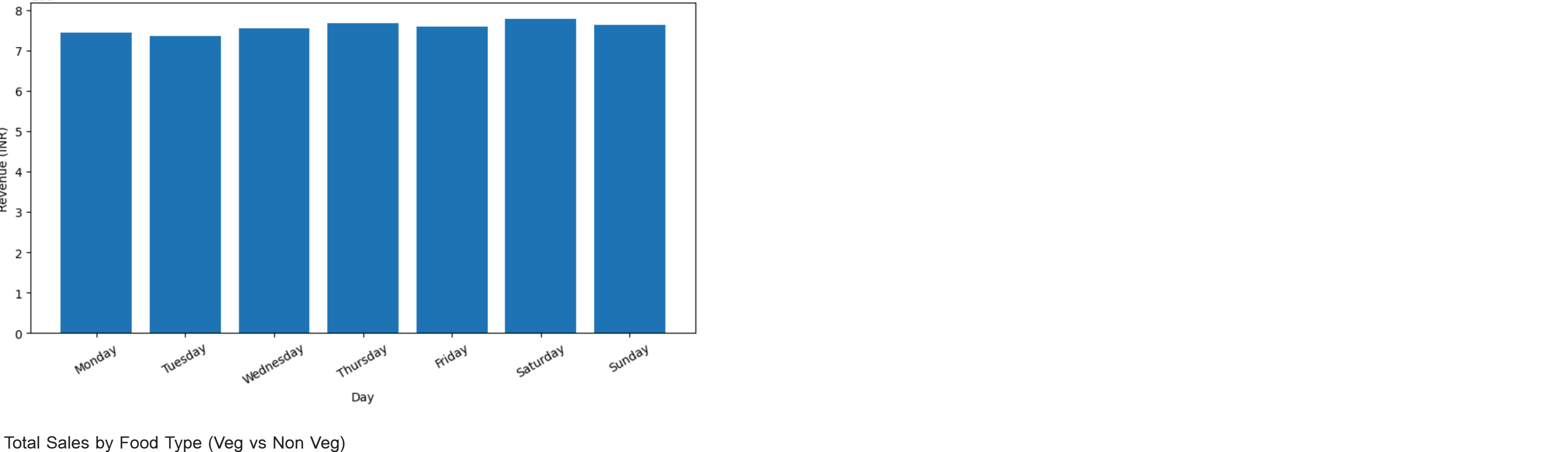


Daily Sales Trend

```
In [31]: # Extract day name from order date
df["DayName"] = pd.to_datetime(df["Order Date"]).dt.day_name()

# Calculate revenue per day
daily_revenue = {
    df.groupby("DayName")["Price (INR)"]
    .sum()
    .reindex(["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"])
}

# Plot bar chart
plt.figure(figsize=(10,5))
plt.bar(daily_revenue.index, daily_revenue.values)
plt.xlabel("Daily Revenue Trend (Mon-Sun)")
plt.ylabel("Revenue (INR)")
plt.xticks(rotation=30)
plt.show()
```



Total Sales by Food Type (Veg vs Non Veg)

```
In [32]: # List of keywords that indicate non-veg dishes
non_veg_keywords = [
    "chicken", "egg", "fish", "mutton",
    "prawn", "biryani", "kabab", "kebab",
    "non-veg", "non veg"
]

# Create Food Category column based on Dish Name
df["Food Category"] = np.where(
    df["Dish Name"].str.lower().str.contains("|".join(non_veg_keywords), na=False),
    "Non-Veg",
    "Veg"
)

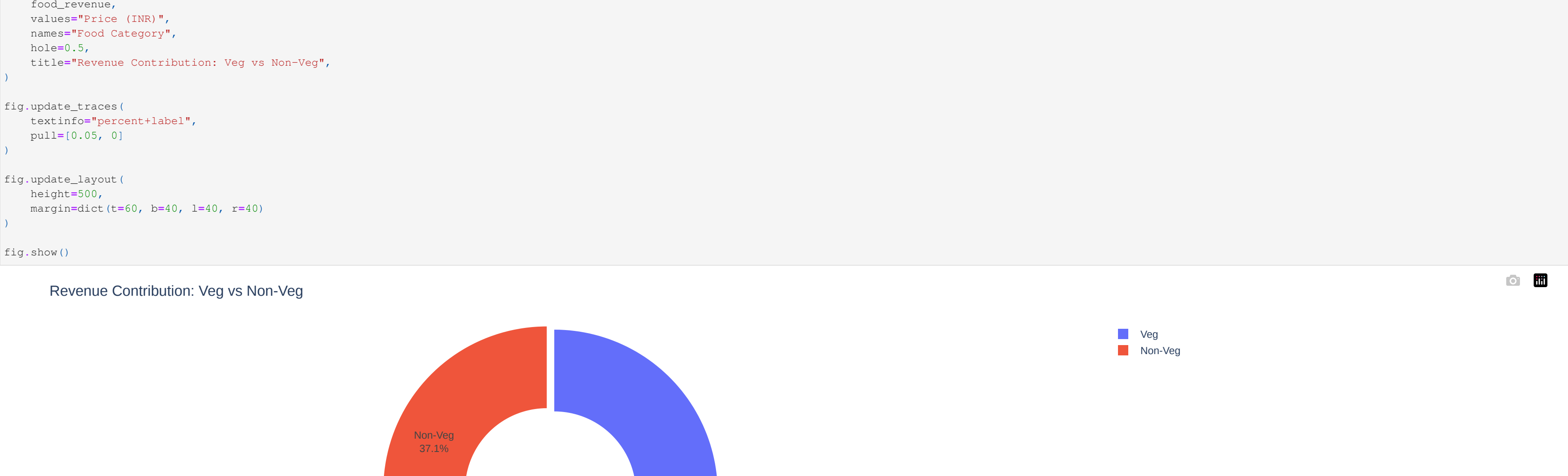
In [33]: food_revenue = (
    df.groupby("Food Category")["Price (INR)"]
    .sum()
    .reset_index()
)

In [34]: fig = px.pie(
    food_revenue,
    values=Price (INR)*,
    names=Food Category*,
    bcolor=,
    title=Revenue Contribution: Veg vs Non-Veg*,
)

fig.update_traces(
    textinfo="percentlabel",
    pull=[0.05, 0]
)

fig.update_layout(
    height=500,
    margin=dict(r=60, b=40, l=40, t=40)
)

fig.show()
```

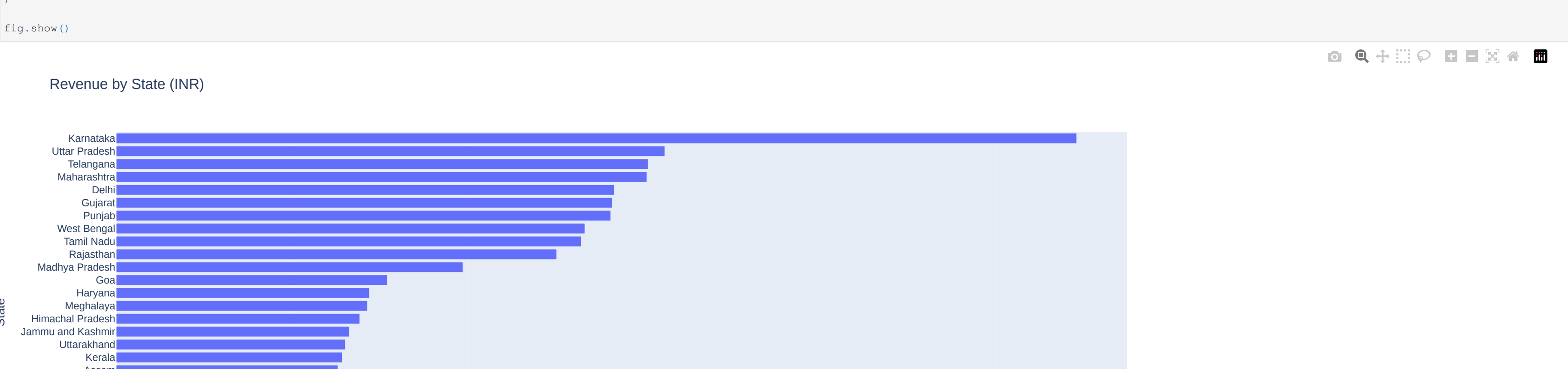


Total Sales by State

```
In [35]: fig = px.bar(
    df.groupby("State", as_index=False)["Price (INR)"].sum(),
    sort_values="Price (INR)", ascending=False,
    x="Price (INR)",
    y="State",
    orientation="h",
    title="Revenue by State (INR)"
)

fig.update_layout(
    height=400,
    yaxis=dict(autorange="reversed")
)

fig.show()
```



Quarterly Performance Summary

```
In [36]: # Convert Order Date to datetime
df["Order Date"] = pd.to_datetime(df["Order Date"])

# Create Quarter column
df["Quarter"] = df["Order Date"].dt.to_period("Q").astype(str)

# Quarterly summary
quarterly_summary = (
    df.groupby("Quarter", as_index=False)
    .agg(
        Total_Sales=("Price (INR)", "sum"),
        Avg_Rating=("Rating", "mean"),
        Total_Orders=("Order Date", "count")
    )
    .sort_values("Quarter")
)

# Round values
quarterly_summary["Total_Sales"] = quarterly_summary["Total_Sales"].round(0)
quarterly_summary["Avg_Rating"] = quarterly_summary["Avg_Rating"].round(2)

# Display result
quarterly_summary
```

Quarterly Performance Summary

Top 5 cities by Sale

```
In [37]: top_5_cities = (
    df.groupby("City")["Price (INR)"]
    .sum()
    .nlargest(5)
    .sort_values()
    .reset_index()
)

fig = px.bar(
    top_5_cities,
    x="Price (INR)",
    y="City",
    orientation="h",
    title="Top 5 Cities by Sales (INR)",
    color_discrete_sequence=["red"]
)

fig.show()
```

