Project: FoodHub Data Analysis

Project Context

As the number of restaurants in New York grows, a rising number of students and busy professionals rely on these establishments to accommodate their hectic lifestyles. Online food delivery apps cater to this demand by offering meals from multiple restaurants in a single smartphone application. One such food aggregator, **FoodHub**, coordinates the entire process—receiving orders, dispatching delivery personnel, and managing real-time updates—while collecting a margin on each delivery.

Objective

To gain a deeper understanding of FoodHub's operation, I analyzed a dataset capturing various aspects of customer orders—from restaurant names and cuisine types to preparation and delivery times. My primary aim was to identify trends in restaurant demand, examine efficiency metrics (e.g., preparation and delivery durations), and offer actionable insights that could enhance customer satisfaction and grow FoodHub's business.

Data Description

The dataset includes a wide range of features that collectively describe each food order placed through the platform:

1. **order_id**: A unique identifier for each order.
2. **customer_id**: The unique ID of the customer placing the order.
3. **restaurant_name**: The name of the restaurant fulfilling the order.
4. **cuisine_type**: The type of cuisine ordered (e.g., American, Italian, Japanese).
5. **cost_of_the_order**: The total monetary value of the order.
6. **day_of_the_week**: Classification of orders as placed on a weekday (Mon–Fri) or weekend (Sat–Sun).
7. **rating**: A 1–5 score provided by the customer (where "Not given" indicates no rating).
8. **food_preparation_time**: The time (in minutes) from when the restaurant confirms the order to when it's picked up by a delivery person.
9. **delivery_time**: The time (in minutes) from when the delivery person departs the restaurant to when the order is delivered to the customer.

Through this data, I set out to pinpoint restaurant popularity, analyze cost patterns, gauge operational efficiency, and explore factors that might influence customer satisfaction—ultimately offering strategic recommendations for FoodHub.

Why Specific Questions? Throughout the project, to simulate working with other stakeholders and teams, I posed a series of focused questions to guide the exploration and ensure alignment with broader business priorities. These questions—often prompted by internal stakeholder feedback—covered topics such as identifying the most popular restaurants, determining high- and low-performing delivery windows, analyzing order costs and revenue margins, and investigating how customer ratings relate to fulfillment speed. By keeping these focal points in mind, the analysis remained structured around actionable goals, driving deeper insights and fostering collaboration across different teams within the organization.

Let us start by importing the required libraries

In [33]:
```
# Installing the needed libraries
!pip install numpy pandas matplotlib seaborn -q --user
```

In [35]:
```
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns
```

Understanding the structure of the data

In [8]:
```
# Instead of Google Colab, for this project I'm using Jupyter Notebook on my laptop. Below, I'm importing the data
fho_data = pd.read_csv('/Users/estarconsulting/Downloads/PG - Data Science/foodhub_order.csv')
```

In [9]:
```
# Displaying the first 5 rows of the dataset
fho_data.head()
```

Out[9]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time	deli
0	1477147	337525	Hangawi	Korean	30.75	Weekend	Not given	25	
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend	Not given	25	
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday	5	23	
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend	3	25	
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday	4	25	

Question 1: How many rows and columns are present in the data?

In [11]:
```
# Getting the number of rows and columns in the dataset

num_rows, num_columns = fho_data.shape
print("The dataset contains", num_rows, "rows and", num_columns, "columns.")
```

```
The dataset contains 1898 rows and 9 columns.
```

Observations:

The dataset contains 1898 rows and 9 columns.

Question 2: What are the datatypes of the different columns in the dataset?

In [14]:
```
# Taking a look at the data types of each column
data_types = fho_data.dtypes
print(data_types)
```

```
order_id                   int64
customer_id                int64
restaurant_name           object
cuisine_type              object
cost_of_the_order        float64
day_of_the_week           object
rating                    object
food_preparation_time      int64
delivery_time              int64
dtype: object
```

Observations:

The data types for the columns are: **int64, int64, object, object, float64, object, object, int64, int64**

The types make sense for all columns except rating, which would be more intuitive as an integer or float. From the initial data inspection, rows without a rating are represented as strings, which explains the current data type. If further analysis is needed, the data type should be changed, and "Not given" may need to be replaced with a numerical value.

Question 3: Are there any missing values in the data?

In [17]:
```
# Checking for any missing values in the dataset
missing_values = fho_data.isnull().sum()
print(missing_values)
```

```
order_id                 0
customer_id              0
restaurant_name          0
cuisine_type             0
cost_of_the_order        0
day_of_the_week          0
rating                   0
food_preparation_time    0
delivery_time            0
dtype: int64
```

Observations:

There are no missing values in any of the columns in the dataset. Each column, there by each row has complete data, which simplifies things for future analysis meaning no imputation or data cleaning for missing values is needed.

Statistical summary of the data

In [20]:
```
# Statistical Summary of the data
stat_summary = fho_data.describe()
print(stat_summary)

# Getting the min, mean, and max for food preparation time
```

```
min_prep_time = stat_summary['food_preparation_time']['min']
avg_prep_time = stat_summary['food_preparation_time']['mean']
max_prep_time = stat_summary['food_preparation_time']['max']

# Printing the minimum, average, and maximum time it takes for food to be prepared once an order is placed
print("\n", "The minimum food preparation time is:", min_prep_time, "minutes")
print("The average food preparation time is:", round(avg_prep_time, 2), "minutes")
print("The maximum food preparation time is:", max_prep_time, "minutes")
```

```
           order_id    customer_id  cost_of_the_order  food_preparation_time  \
count  1.898000e+03    1898.000000        1898.000000            1898.000000
mean   1.477496e+06  171168.478398          16.498851              27.371970
std    5.480497e+02  113698.139743           7.483812               4.632481
min    1.476547e+06    1311.000000           4.470000              20.000000
25%    1.477021e+06   77787.750000          12.080000              23.000000
50%    1.477496e+06  128600.000000          14.140000              27.000000
75%    1.477970e+06  270525.000000          22.297500              31.000000
max    1.478444e+06  405334.000000          35.410000              35.000000

       delivery_time
count    1898.000000
mean       24.161749
std         4.972637
min        15.000000
25%        20.000000
50%        25.000000
75%        28.000000
max        33.000000

 The minimum food preparation time is: 20.0 minutes
The average food preparation time is: 27.37 minutes
The maximum food preparation time is: 35.0 minutes
```

Observations:

Statistical Summary of Food Preparation Time: The minimum food preparation time is: 20.0 minutes The average food preparation time is: 27.37 minutes The maximum food preparation time is: 35.0 minutes

General Data Quality: The statistical summary for other columns shows no extreme values, suggesting the absence of significant outliers. Overall, the data appears to be in good standing for analysis.

Question 5: How many orders are not rated?

In [23]:
```
# Checking to see all unque variables (To ensure missing ratings occur the same way)
unique_ratings = fho_data['rating'].unique()
print("Unique values in the 'rating' column:", unique_ratings)

# Counting the number of orders with 'Not given'
not_rated_count = fho_data[fho_data['rating'] == 'Not given'].shape[0]
print("Number of unrated orders:", not_rated_count)

# Calculating the percentage of unrated orders to understand gravity of missing data
total_orders = fho_data.shape[0]
unrated_percentage = (not_rated_count / total_orders) * 100
print(f"Percentage of unrated orders: {unrated_percentage:.2f}%")
```

```
Unique values in the 'rating' column: ['Not given' '5' '3' '4']
Number of unrated orders: 736
Percentage of unrated orders: 38.78%
```

Observations:

- The unique values in the 'rating' column are: 'Not given', '5', '3', '4'.
- The missing ratings are represented as 'Not given'. There are no other unexpected values in the column, which indicates that all missing ratings are consistently labeled.
- Number of Unrated Orders: There are 736 unrated orders.
- Percentage of Unrated Orders: 38.78% of the orders are unrated. This shows that a significant portion (nearly 39%) of the orders did not receive a customer rating, can affect analysis and or suggests a potential area for improvement in collecting feedback from customers.

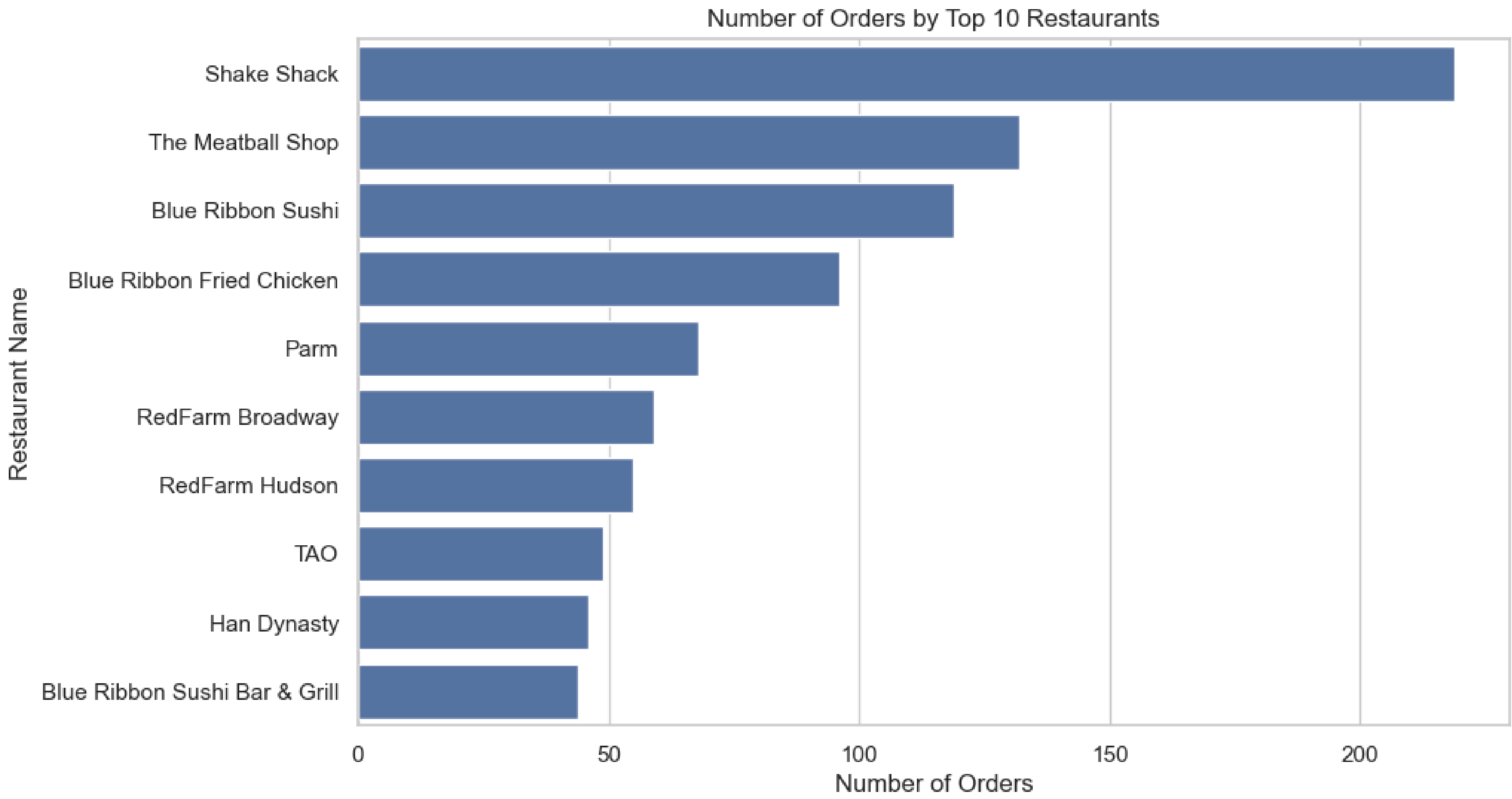
Exploratory Data Analysis (EDA)

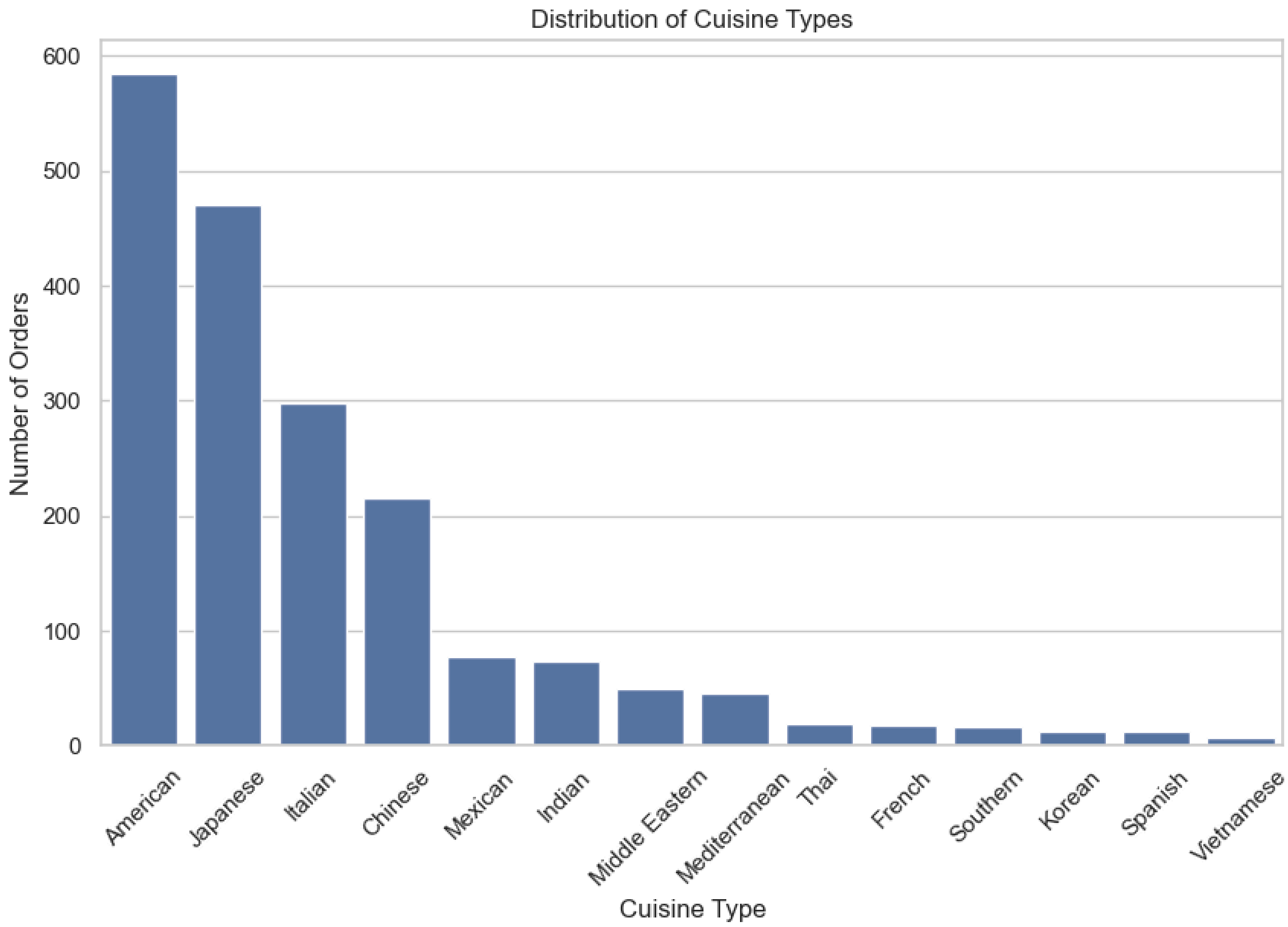
Univariate Analysis

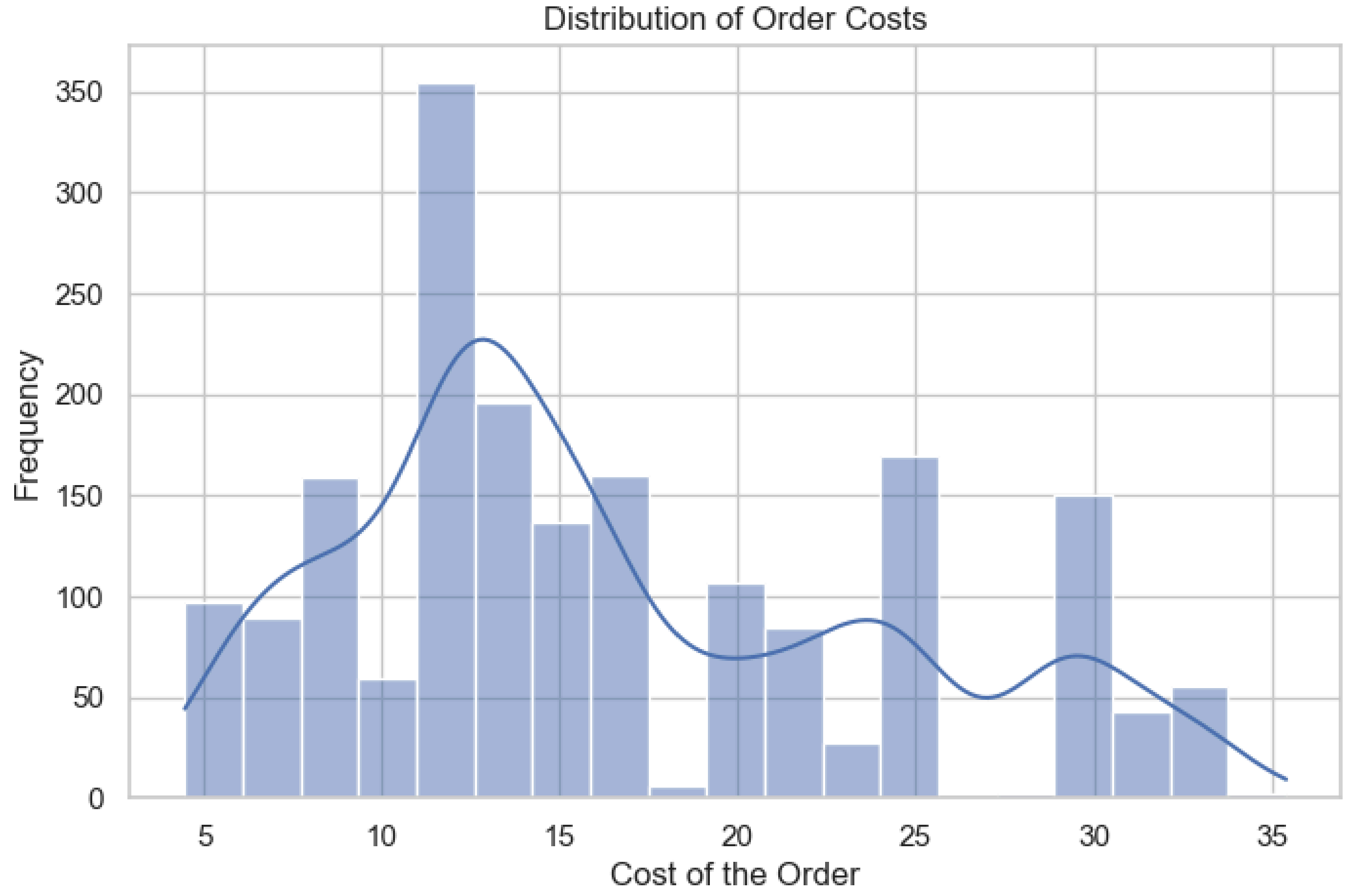
Question 6: Exploring all the variables and providing observations on their distributions.

In [28]:
```
# Setting the visual style for the plots to look nice
sns.set(style="whitegrid")
```

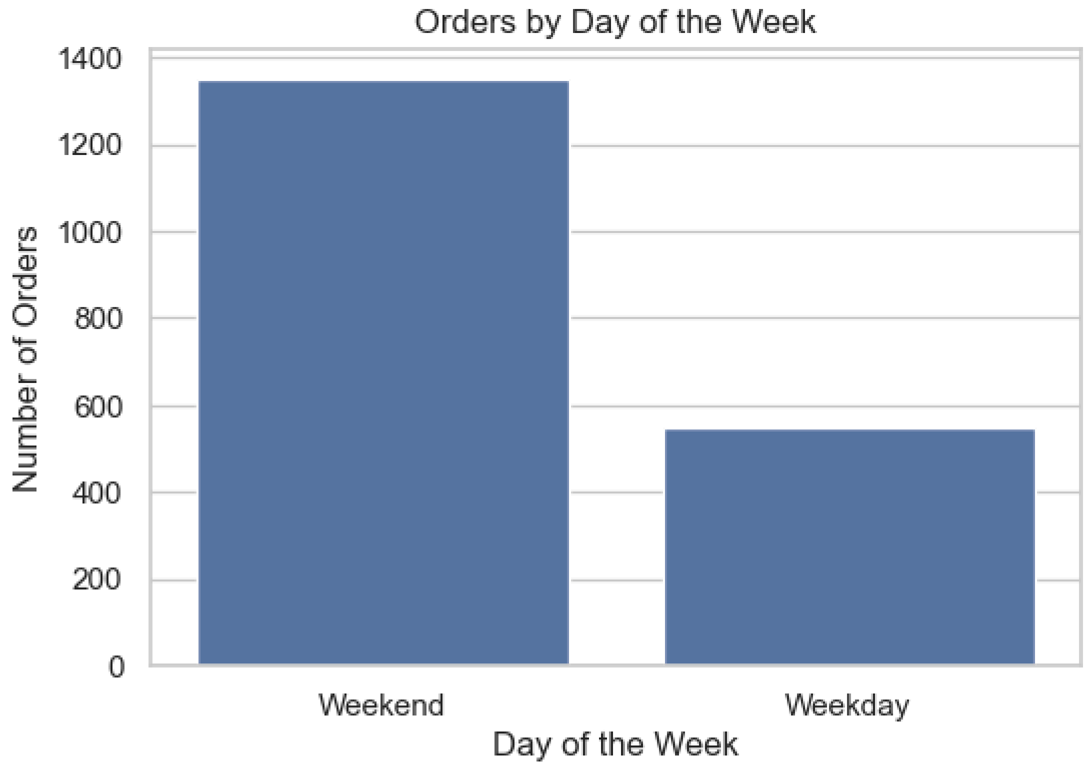
```python
# 1. Restaurant Name - (I Chose Top 10 Restaurants as viewing all unique resturant will not be visually appeasing)
top_10_restaurants = fho_data['restaurant_name'].value_counts().nlargest(10).index
plt.figure(figsize=(10, 6))
sns.countplot(data=fho_data[fho_data['restaurant_name'].isin(top_10_restaurants)], y='restaurant_name',
              order=top_10_restaurants)
plt.title('Number of Orders by Top 10 Restaurants')
plt.xlabel('Number of Orders')
plt.ylabel('Restaurant Name')
plt.show()

# 2. Cuisine Type
plt.figure(figsize=(10, 6))
sns.countplot(data=fho_data, x='cuisine_type', order=fho_data['cuisine_type'].value_counts().index)
plt.title('Distribution of Cuisine Types')
plt.xlabel('Cuisine Type')
plt.ylabel('Number of Orders')
plt.xticks(rotation=45)
plt.show()

# 3. Cost of the Order
plt.figure(figsize=(8, 5))
sns.histplot(fho_data['cost_of_the_order'], kde=True)
plt.title('Distribution of Order Costs')
plt.xlabel('Cost of the Order')
plt.ylabel('Frequency')
plt.show()

# 4. Day of the Week
plt.figure(figsize=(6, 4))
sns.countplot(data=fho_data, x='day_of_the_week', order=fho_data['day_of_the_week'].value_counts().index)
plt.title('Orders by Day of the Week')
plt.xlabel('Day of the Week')
plt.ylabel('Number of Orders')
plt.show()

# 5. Ratings
plt.figure(figsize=(8, 5))
sns.countplot(data=fho_data, x='rating', order=fho_data['rating'].value_counts().index)
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Number of Orders')
plt.show()

# 6. Food Preparation Time (Used both Boxplot & Histogram for different visual conceptualization)
plt.figure(figsize=(8, 5))
sns.boxplot(x=fho_data['food_preparation_time'])
plt.title('Boxplot of Food Preparation Time')
plt.xlabel('Food Preparation Time (minutes)')
plt.show()

plt.figure(figsize=(8, 5))
sns.histplot(fho_data['food_preparation_time'], kde=True)
plt.title('Distribution of Food Preparation Time')
plt.xlabel('Food Preparation Time (minutes)')
plt.ylabel('Frequency')
plt.show()

# 7. Delivery Time (Used both Boxplot & Histogram for different visual conceptualization)
plt.figure(figsize=(8, 5))
sns.boxplot(x=fho_data['delivery_time'])
plt.title('Boxplot of Delivery Time')
plt.xlabel('Delivery Time (minutes)')
plt.show()

plt.figure(figsize=(8, 5))
sns.histplot(fho_data['delivery_time'], kde=True)
plt.title('Distribution of Delivery Time')
plt.xlabel('Delivery Time (minutes)')
plt.ylabel('Frequency')
plt.show()
```


Number of Orders by Top 10 Restaurants
Shake Shack
The Meatball Shop
Blue Ribbon Sushi
Blue Ribbon Fried Chicken
Parm
RedFarm Broadway
RedFarm Hudson
TAO
Han Dynasty
Blue Ribbon Sushi Bar & Grill
Restaurant Name
0
50
100
150
200
Number of Orders

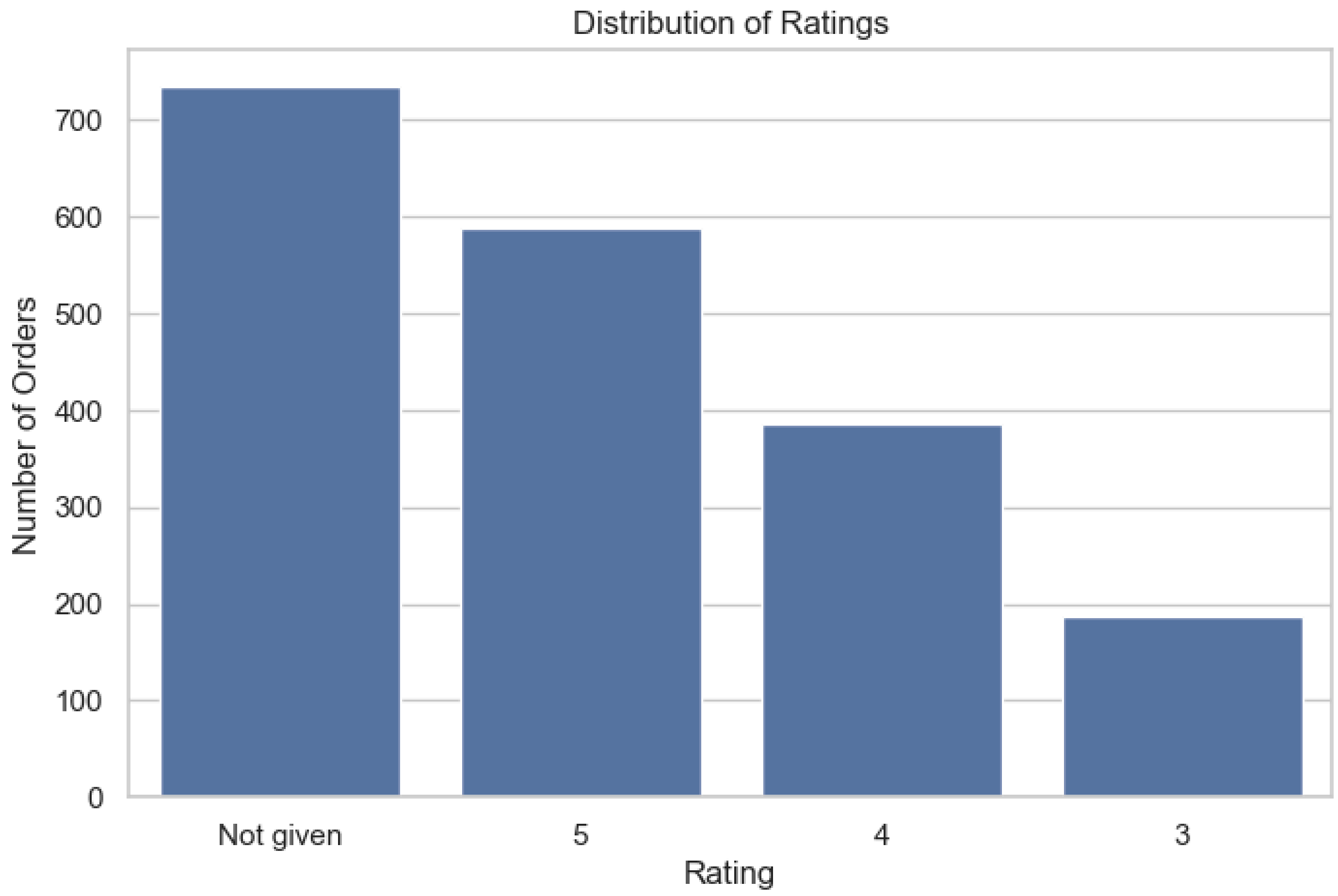

Distribution of Cuisine Types
600
500
400
300
200
100
0
Number of Orders
American
Japanese
Italian
Chinese
Mexican
Indian
Middle Eastern
Mediterranean
Thai
French
Southern
Korean
Spanish
Vietnamese
Cuisine Type



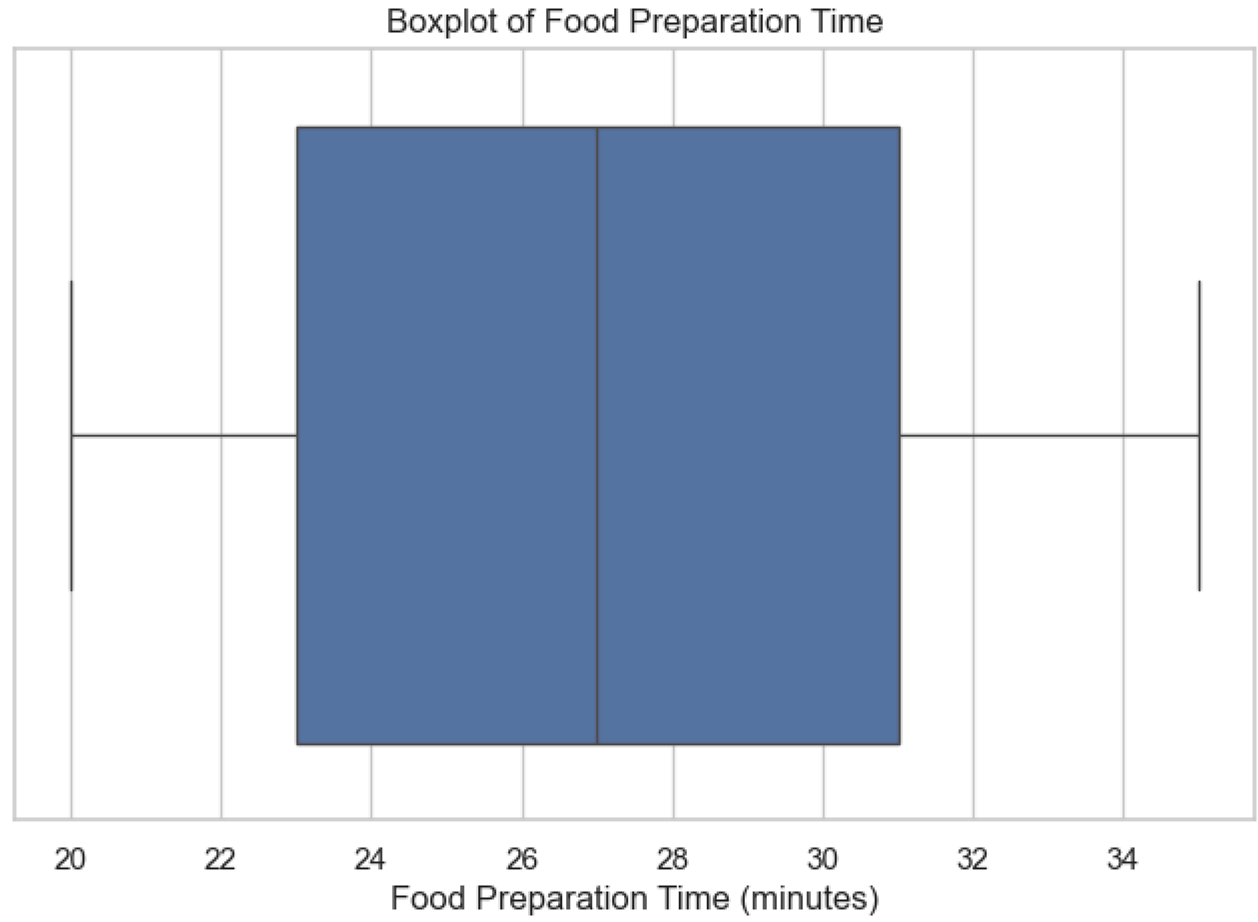
Distribution of Order Costs
Frequency
0
50
100
150
200
250
300
350
5
10
15
20
25
30
35
Cost of the Order

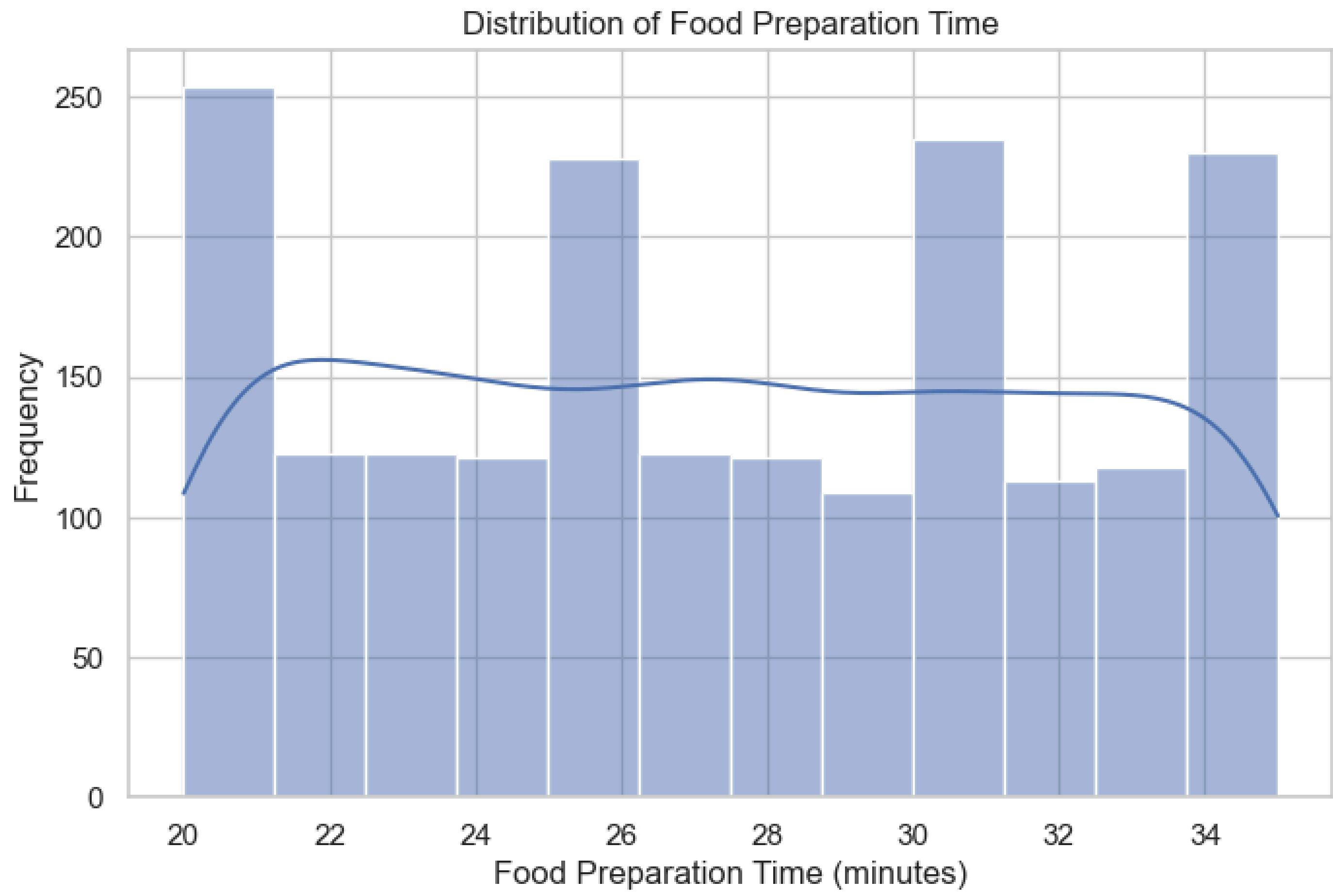


Orders by Day of the Week
Number of Orders
0
200
400
600
800
1000
1200
1400
Weekend
Weekday
Day of the Week

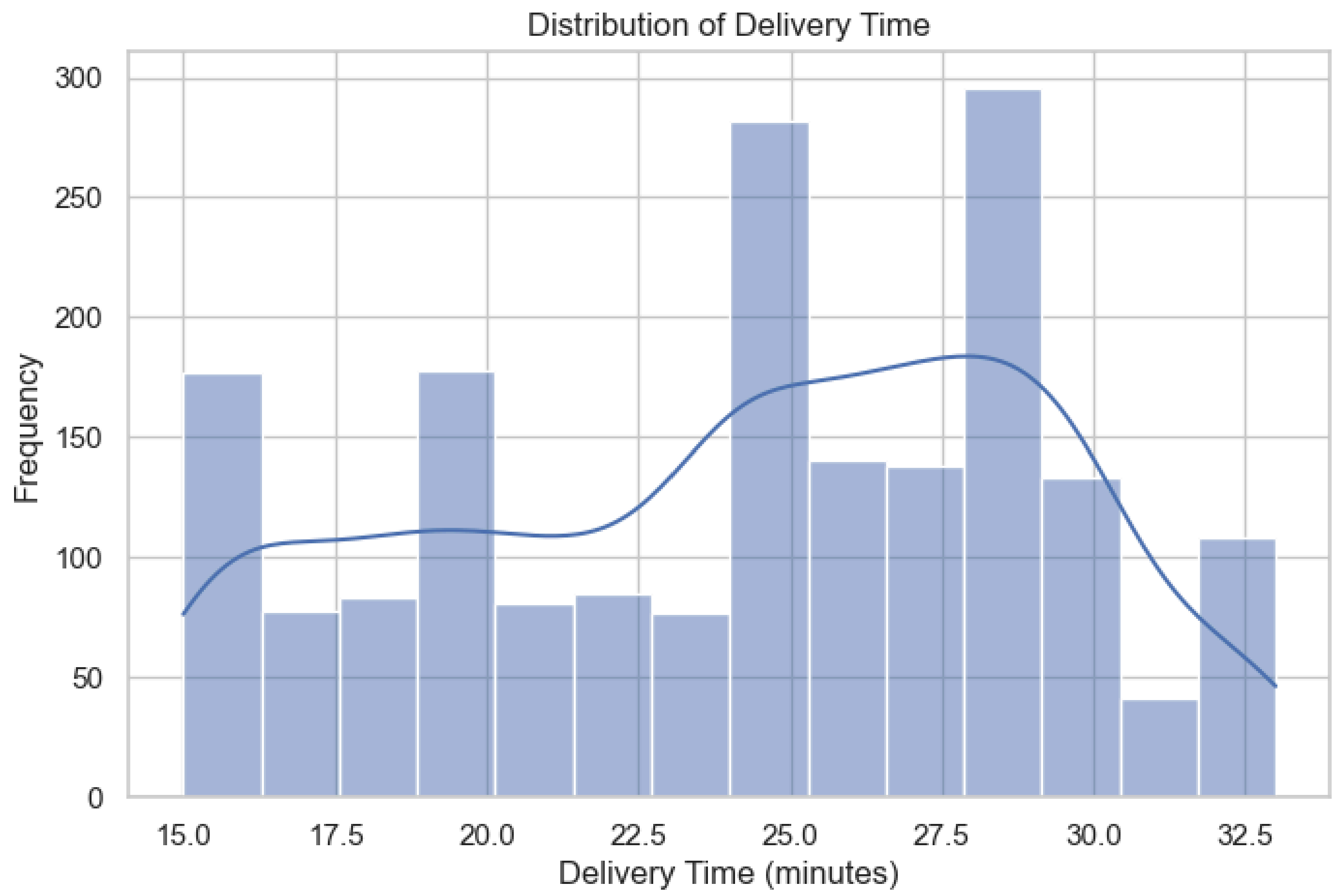


Distribution of Ratings
Number of Orders
0
100
200
300
400
500
600
700
Not given
5
4
3
Rating


Boxplot of Food Preparation Time
20
22
24
26
28
30
32
34
Food Preparation Time (minutes)


Distribution of Food Preparation Time
Frequency
0
50
100
150
200
250
20
22
24
26
28
30
32
34
Food Preparation Time (minutes)


Boxplot of Delivery Time
15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5
Delivery Time (minutes)



Observations:

1. **Restaurant Name**:

   - **Top 10 Restaurants** were selected for visualization because displaying all restaurants was chaotic and offer no meaningful visual information. By focusing on the **top 10** makes it easier to identify which restaurants have the highest demand.
   - From the visualization, we can see that a few restaurants receive significantly more orders compared to others. This insight can be useful in situation like deciding which restaurant promotions or exclusive deals are offered on.

2. **Cuisine Type**:

   - The **distribution of cuisines** shows which types are most popular among customers.
   - It's clear that a few cuisines are in higher demand than others. FoodHub could use this info to decide which cuisines to offer promotions, target new resturant to join the platform etc...

3. **Cost of the Order**:

   - The **histogram** shows that most orders fall within a moderate price range (7 & 17), with a peak around $12's.
   - There are fewer orders at very low or very high costs, indicating that customers tend to order within a common price range. This suggests that there might be an ideal price point that most customers are comfortable with.

4. **Day of the Week**:

   - The **countplot** reveals the distribution of orders between weekdays and weekends.
   - There is a significant difference in the number of orders based on the day. This info could help FoodHub allocate delivery resources more efficiently by understanding which days have higher or lower demand.

5. **Rating**:

   - The **countplot** of ratings, including "Not given," shows that a significant proportion of orders have missing ratings.
   - Most ratings provided are relatively high (3, 4, or 5). The fact that amolst 39% of orders are unrated identifies a feedback mechanism gap FoodHub can work to improve, maybe by incentivizing customers to leave ratings.

6. **Food Preparation Time**:

   - The boxplot and histogram reveal that most food preparation times fall between 23 and 31 minutes, which represents the interquartile range (IQR).
   - There are no significant outliers present, as shown by the absence of data points beyond the whiskers.
   - The average preparation time is around 27 minutes. This info can help in setting realistic expectations for customers.

7. **Delivery Time**:

   - The **boxplot** and **histogram** for delivery time show that most deliveries take between **15 and 33 minutes**.
   - The distribution is fairly consistent, and there are no extreme outliers, which indicates reliability in delivery times. This consistency can be a positive sign for customer satisfaction.

Columns Not Included in Univariate Analysis:

- `order_id` and `customer_id`:
  - These columns were not included in univariate analysis because they contain unique identifiers for each order and customer, respectively.
  - Visualizing these columns would not provide meaningful insights as each value is unique, and they do not contribute to understanding trends or patterns in the dataset.

Question 7: What are the top 5 restaurants in terms of the number of orders received?

In [31]:
```
# Finding the top 5 restaurants by number of orders

top_5_restaurants = fho_data['restaurant_name'].value_counts().nlargest(5)
print("Top 5 Restaurants by Number of Orders:",'\n',top_5_restaurants)
```

```
Top 5 Restaurants by Number of Orders:
 restaurant_name
Shake Shack                  219
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
Name: count, dtype: int64
```

Observations:

Top 5 Restaurants by Number of Orders:

- Shake Shack 219
- The Meatball Shop 132
- Blue Ribbon Sushi 119
- Blue Ribbon Fried Chicken 96
- Parm 68
- These restaurants have the highest demand among customers. This data can be useful in forming partnerships or promotions focused on these high-performing restaurants to further boost customer engagement.

Question 8: Which is the most popular cuisine on weekends?

In [34]:
```
# Filtering the data to only include orders from weekends
weekend_data = fho_data[fho_data['day_of_the_week'] == 'Weekend']

# Finding the most popular cuisine type on filtered weekends df
most_popular_cuisine_weekends = weekend_data['cuisine_type'].value_counts().idxmax()

print("The most popular cuisine on weekends is:", most_popular_cuisine_weekends)
```

```
The most popular cuisine on weekends is: American
```

Observations:

The most popular cuisine on weekends is: American The most popular cuisine on weekends is American. This could be a direct result of FoodHub's location. It also provides insights into customer preferences, the availability of options, or both.

Question 9: What percentage of the orders cost more than 20 dollars?

In [37]:
```
# Counting how many orders cost more than 20 dollars
orders_above_20 = fho_data[fho_data['cost_of_the_order'] > 20].shape[0]

# Calculating the % of orders that cost more than 20 dollars
percentage_above_20 = (orders_above_20 / total_orders) * 100

print("Percentage of orders costing more than 20 dollars:", percentage_above_20)
```

```
Percentage of orders costing more than 20 dollars: 29.24130663856691
```

Observations:

Percentage of Orders Costing More Than $20 is 29.24%.

This suggests that most customers prefer to pay $20 or below (about 70%), but a significant portion of customers are willing to spend more, possibly on higher-value items or larger orders. Further analysis involving order quantities or customer income brackets could provide deeper insights.

Question 10: What is the mean order delivery time?

In [40]:
```
# Calculating the avg delivery time for all orders

mean_delivery_time = fho_data['delivery_time'].mean()
print("The mean order delivery time is:", mean_delivery_time, "minutes")
```

```
The mean order delivery time is: 24.161749209694417 minutes
```

Observations:

The mean order delivery time is 24.16 minutes. This suggests that, on average, customers can expect their orders to be delivered in about 24 minutes. This info can be useful in setting customer expectations and a benchmark for delivery performance.

Question 11: If the company decides to give 20% discount vouchers to the top 3 most frequent customers. Find the IDs of these customers and the number of orders they placed.

In [43]:
```
# Filtering the top 3 customers based on the count of orders they've placed
top_3_customers = fho_data['customer_id'].value_counts().nlargest(3)
print("Top 3 Customers by Number of Orders:",'\n',top_3_customers)
```

```
Top 3 Customers by Number of Orders:
 customer_id
52832    13
47440    10
83287     9
Name: count, dtype: int64
```

Observations:

The top 3 customers by the number of orders are: Customer ID 52832: 13 orders Customer ID 47440: 10 orders Customer ID 83287: 9 orders

These customers have placed the most orders, making them eligible for the 20% discount vouchers. This kind of reward system for frequent customers can help increase loyalty and encourage future purchases.

Multivariate Analysis

Question 12: Perform a multivariate analysis to explore relationships between the important variables in the dataset.

For the multivariate analysis, I will use Question 13 to 16 to help inform the focus of the analysis.

In [48]:
```
# To include rating in corr analysis, I converted "Not given" to NaN
fho_data['rating'] = fho_data['rating'].replace("Not given", np.nan).astype(float)
```

In [49]:
```
print(fho_data['rating'].info())
```

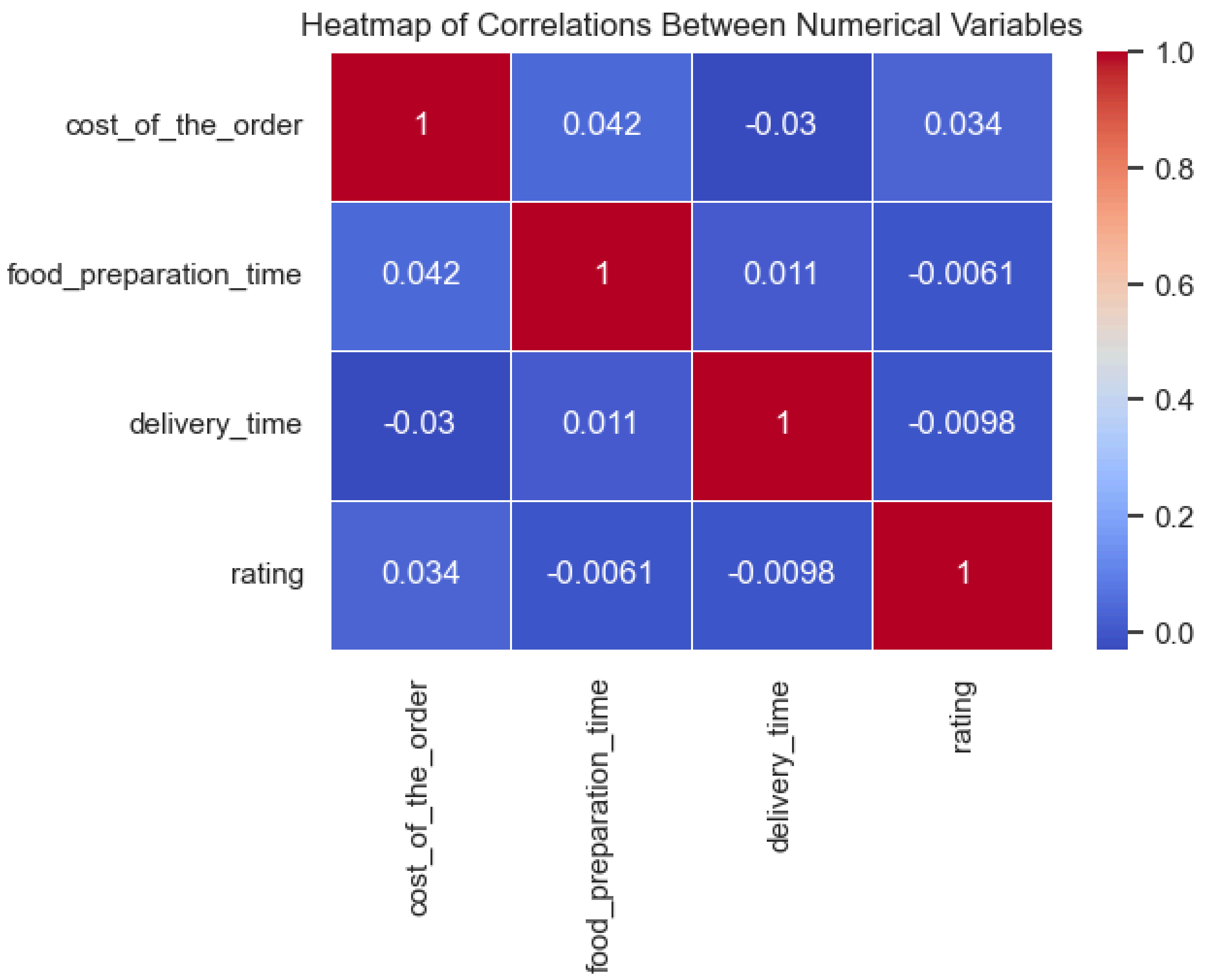
```
<class 'pandas.core.series.Series'>
RangeIndex: 1898 entries, 0 to 1897
Series name: rating
Non-Null Count  Dtype
--------------  -----
1162 non-null   float64
dtypes: float64(1)
memory usage: 15.0 KB
None
```

For Multivariate Analysis, I took the approach of

- Numerical column analysis
- Numerical and Categorical column analysis
- Categorical column analysis

Numerical column analysis

In [52]:
```
# Naming numerica columns only for Heatmap
numerical_columns = ['cost_of_the_order', 'food_preparation_time', 'delivery_time', 'rating']

# Using correlation matrix
correlation_matrix = fho_data[numerical_columns].corr()

# Plotting
plt.figure(figsize=(6, 4))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Heatmap of Correlations Between Numerical Variables')
plt.show()
```

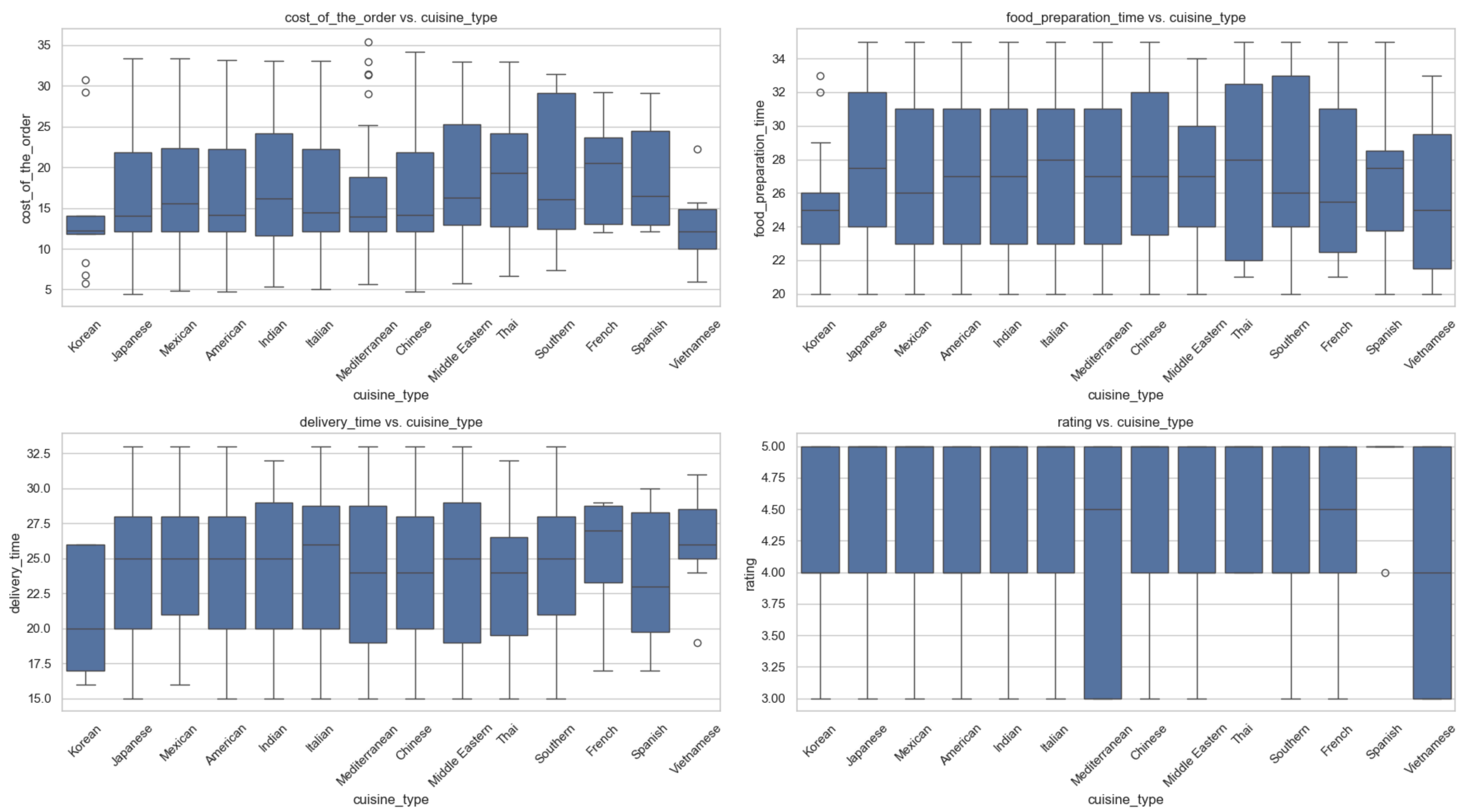


Observation

- The heatmap shows a weak correlation between most variables, with a slight negative correlation between "rating" and "delivery_time," suggesting that longer delivery times might be associated with lower ratings. This could imply that customers value prompt delivery, affecting their satisfaction ratings.

Numerical and Categorical column analysis

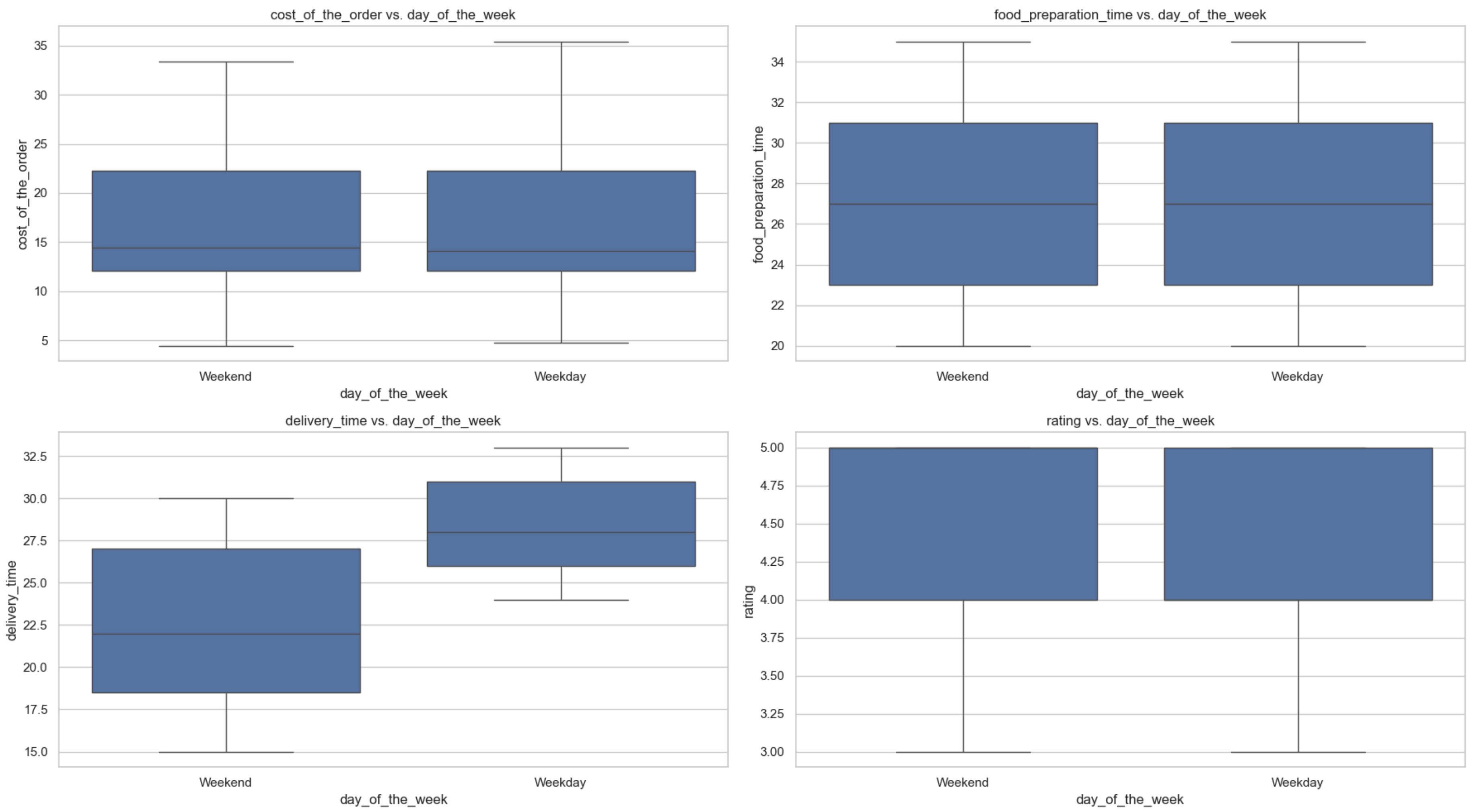
In [55]:
```
# Relationship btwn 'cuisine_type' and numerical variables using boxplots

# Setting up Plot
plt.figure(figsize=(18, 10))

# Creating boxplots for **each** numerical variable against 'cuisine_type'
for idx, numerical_var in enumerate(numerical_columns, 1):
    plt.subplot(2, 2, idx)
    sns.boxplot(data=fho_data, x='cuisine_type', y=numerical_var)
    plt.xticks(rotation=45)
    plt.title(f'{numerical_var} vs. cuisine_type')

plt.tight_layout()
plt.show()
```



Observation

- Thai and French cuisines have higher average costs per order (which may reflect pricing based on ingredient quality or perceived premium status) and Vietnamese cuisine shows the lowest average cost, which may reflect cheaper ingredient costs or simpler dishes.
- Most cuisines have similar mean preparation times, around 27 minutes, except for Korean and Vietnamese, which are slightly lower, possibly due to simpler or quicker dishes.
- There is a consistent delivery time across cuisines, but Korean cuisine has the shortest delievery time, maybe indicating ideal location. Another notable mention is Vietnamese cusisnes which on avgerge take longer to be delviered. Location to customer can also explain.
- Spanish cuisine has high ratings with very low variability, while Mediterranean and Vietnamese cuisines show lower ratings with higher variability, indicating mixed customer satisfaction for those cuisines.

In [57]:
```
# Relationship between 'day_of_the_week' and numerical variables

# Setting Up Plot
plt.figure(figsize=(18, 10))

# Creating boxplots for **each** numerical variable against 'day_of_the_week'
for idx, numerical_var in enumerate(numerical_columns, 1):
    plt.subplot(2, 2, idx)
    sns.boxplot(data=fho_data, x='day_of_the_week', y=numerical_var)
    plt.title(f'{numerical_var} vs. day_of_the_week')

plt.tight_layout()
plt.show()
```



Observation

- Only notable mention is delivery times and days of the week. Notably delivery take longer during the weekdays, which might be explained by increased traffic (work days etc..) or operational workload (prepardness for weekend order rush).

Categorical column analysis

- Analysis of the relationship between 'cuisine_type' and 'day_of_the_week' was conducted. However, no noteworthy differences were observed beyond a general reduction in order volume on weekdays compared to weekends. Therefore, this analysis has been removed for the sake of conciseness.

Question 13: If the company wants to provide a promotional offer in the advertisement of the restaurants and the condition to get the offer is that the restaurants must have a rating count of more than 50 and the average rating should be greater than 4, find the restaurants fulfilling the criteria to get the promotional offer.

In [61]:
```
# Dropping NaN values since it is literly not a rating and can be misunderstood if left in by machine
ratings_fho_data = fho_data.dropna(subset=['rating'])

restaurant_rating_summary = ratings_fho_data.groupby('restaurant_name')['rating'].agg(['count', 'mean'])

# Filtering restaurants with count greater than 50 and mean rating greater than 4
```

```
eligible_restaurants = restaurant_rating_summary[(restaurant_rating_summary['count'] > 50) & (restaurant_rating_sum
print(eligible_restaurants)
```

```
                          count      mean
restaurant_name
Blue Ribbon Fried Chicken    64  4.328125
Blue Ribbon Sushi            73  4.219178
Shake Shack                 133  4.278195
The Meatball Shop            84  4.511905
```

Observations:

Blue Ribbon Fried Chicken: 64 ratings, average rating of 4.33 Blue Ribbon Sushi: 73 ratings, average rating of 4.22 Shake Shack: 133 ratings, average rating of 4.28 The Meatball Shop: 84 ratings, average rating of 4.51 These restaurants qualify for the promotional offer based on their popularity and high customer satisfaction. Extra detail noticed, the restaurant list aligns with the observation that American cuisine are the most ordered/popular option.

Question 14: If the company charges the restaurant 25% on the orders having cost greater than 20 dollars and 15% on the orders having cost greater than 5 dollars, find the net revenue generated by the company across all orders.

In [64]:
```
# To make extra analysis on the percentage contribution of both fee levels
total_25_percent_commission = 0
total_15_percent_commission = 0

commissions = []

# Using for loop and if conditions to determine the commission for each order
for cost in fho_data['cost_of_the_order']:
    if cost > 20:
        commission = cost * 0.25
        total_25_percent_commission += commission  # Track 25% commission part
    elif cost > 5:
        commission = cost * 0.15
        total_15_percent_commission += commission  # Track 15% commission part
    else:
        commission = 0
    commissions.append(commission)

# Adding the calculated commissions above as a new column to the dataset
fho_data['commission'] = commissions

# Then Calculating the total revenue from all commissions
total_revenue = sum(fho_data['commission'])

print(total_revenue, total_25_percent_commission, total_15_percent_commission)
```

```
6166.303 3688.7275000000027 2477.5755000000036
```

Observations:

- The orders with a cost greater than 20 dollar contribute most to the company's revenue (25% resulting in ~$3688.73).
- Orders between 5 & 20 also added to the revenue but lower (15% resulting in ~$2477.58)

Question 15: If the company wants to analyze the total time required to deliver the food. What percentage of orders take more than 60 minutes to get delivered from the time the order is placed? (The food has to be prepared and then delivered.)

In [67]:
```
# To calculate the total time (food preparation time + delivery time)
fho_data['total_time'] = fho_data['food_preparation_time'] + fho_data['delivery_time']

# Proceeding with how many orders took more than 60 minutes in total

orders_above_60_minutes = fho_data[fho_data['total_time'] > 60].shape[0]

# Calculating the % of orders that took more than 60 minutes

total_orders = fho_data.shape[0]
percentage_above_60_minutes = (orders_above_60_minutes / total_orders) * 100

print(percentage_above_60_minutes)
```

```
10.537407797681771
```

Observations:

~10.54% of the orders take more than 60 minutes to get delivered from the time the order is placed (including both food preparation and delivery times).

This indicates that a relatively small portion of the orders exceed the 60-minute delivery time, which could be an area of focus for improvement to enhance overall customer satisfaction.

Question 16: The company wants to analyze the delivery time of the orders on weekdays and weekends. How does the mean delivery time vary during weekdays and weekends?

In [70]:
```
# Mean delivery time for orders on weekdays and weekends
mean_delivery_time_weekdays = fho_data[fho_data['day_of_the_week'] == 'Weekday']['delivery_time'].mean()
mean_delivery_time_weekends = fho_data[fho_data['day_of_the_week'] == 'Weekend']['delivery_time'].mean()

print(mean_delivery_time_weekdays, mean_delivery_time_weekends)
```

```
28.340036563071298 22.4700222057735
```

Observations:

The mean delivery time is higher on weekdays (28.34 minutes) compared to weekends (22.47 minutes), meaning deliveries tend to take longer during weekdays. Given that previous data shows more orders are made during weekends, this might indicate that restaurants are better prepared for the weekend rush. Alternatively, the longer delivery times on weekdays could be due to increased traffic related to work or commuting.

Conclusion and Recommendations

Question 17: What are your conclusions from the analysis? What recommendations would you like to share to help improve the business?

Conclusions:

- The analysis reveals that a small number of restaurants dominate the demand, suggesting that customers tend to prefer well-established or familiar brands. Shake Shack and The Meatball Shop, for instance, are leading in terms of order volume. This concentration of orders might indicate that customer loyalty and brand recognition play a significant role in decision-making.

- American cuisine is the most popular choice, particularly on weekends. This reflects a trend where customers prefer comfort food or familiar options during leisure days. On the other hand, there is notable demand for Asain(Japanese, Chinese) and Italian cuisine, which presents an opportunity for FoodHub to further diversify its offerings.

- Approx. 39% of orders are not rated showing a significant gap in collecting customer feedback. This not only limits FoodHub's ability to evaluate service quality but also restricts opportunities for making data-driven improvements based on customer experiences.

- Delivery times are longer on weekdays compared to weekends, which points to potential inefficiencies in handling weekday orders. The average delivery time of 24 minutes is reasonable, but the longer weekday times may be driven by factors such as traffic or insufficient delivery personnel. Additionally, about 10.5% of orders take more than 60 minutes, indicating that a subset of customers experiences considerable delays, which could negatively impact satisfaction.

- Most orders fall in the range of $7to$17, indicating that customers are sensitive to price. This price range appears to be the sweet spot for the majority of customers, suggesting that affordability is a key factor influencing ordering behavior.

Recommendations:

- Implement incentives such as discounts or loyalty points for customers to leave feedback, aiming to reduce the high percentage of unrated orders.
- Leverage the popularity of high-demand cuisines like American, Japanese, and Italian by collaborating with top-rated restaurants to offer promotions and double down on benefits of customer loyalty.
- If further analysis shows that the demand for other cuisines is due to a lack of available options on the platform, FoodHub can partner with new restaurants to onboard them. This would likely increase revenue and customer satisfaction.
- Analyze delivery routes and times to address the extended delivery durations on weekdays. Additionally, onboarding more restaurants offering high-demand cuisines that are closer to customers can promote cycling and walking deliveries, reducing reliance on driving through traffic.

In [ ]: