FoodHub Data Analysis

Context

The number of restaurants in New York is increasing day by day. Lots of students and busy professionals rely on those restaurants due to their hectic lifestyles. Online food delivery service is a great option for them. It provides them with good food from their favorite restaurants. A food aggregator company FoodHub offers access to multiple restaurants through a single smartphone app.

The app allows the restaurants to receive a direct online order from a customer. The app assigns a delivery person from the company to pick up the order after it is confirmed by the restaurant. The delivery person then uses the map to reach the restaurant and waits for the food package. Once the food package is handed over to the delivery person, he/she confirms the pick-up in the app and travels to the customer's location to deliver the food. The delivery person confirms the drop-off in the app after delivering the food package to the customer. The customer can rate the order in the app. The food aggregator earns money by collecting a fixed margin of the delivery order from the restaurants.

Objective

The food aggregator company has stored the data of the different orders made by the registered customers in their online portal. They want to analyze the data to get a fair idea about the demand of different restaurants which will help them in enhancing their customer experience. Perform the data analysis to find answers to these questions that will help the company to improve the business.

Data Description

The data contains the different data related to a food order. The detailed data dictionary is given below.

Data Dictionary

- order_id: Unique ID of the order
- customer_id: ID of the customer who ordered the food
- restaurant_name: Name of the restaurant
- cuisine_type: Cuisine ordered by the customer
- cost: Cost of the order
- day_of_the_week: Indicates whether the order is placed on a weekday or weekend (The weekday is from Monday to Friday and the weekend is Saturday and Sunday)
- rating: Rating given by the customer out of 5
- food_preparation_time: Time (in minutes) taken by the restaurant to prepare the food. This is calculated by taking the difference between the timestamps of the restaurant's order confirmation and the delivery person's pick-up confirmation.
- delivery_time: Time (in minutes) taken by the delivery person to deliver the food package. This is calculated by taking the difference between the timestamps of the delivery person's pick-up confirmation and drop-off information

Let us start by importing the required libraries

In [1]:
```
# import libraries for data manipulation
import numpy as np
import pandas as pd

# import libraries for data visualization
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Command to tell Python to actually display the graphs
%matplotlib inline
```

Understanding the structure of the data

In [4]:
```
# uncomment and run the following lines for Google Colab
df = pd.read_csv('/Users/camilaconyers/Downloads/foodhub_order (1).csv')
```

In [6]:
```
# read the data
df = pd.read_csv('/Users/camilaconyers/Downloads/foodhub_order (1).csv')
# returns the first 5 rows
df.head()
```

Out[6]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time
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

Observations:

The DataFrame has 9 columns as mentioned in the Data Dictionary. Data in each row corresponds to the order placed by a customer.

In [7]:
```
# Check the shape of the dataset
df.shape
```

Out[7]: `(1898, 9)`

Observations: The data has 1898 rows and 9 columns.

In [8]:
```
# Use info() to print a concise summary of the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1898 entries, 0 to 1897
Data columns (total 9 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   order_id               1898 non-null   int64  
 1   customer_id            1898 non-null   int64  
 2   restaurant_name        1898 non-null   object 
 3   cuisine_type           1898 non-null   object 
 4   cost_of_the_order      1898 non-null   float64
 5   day_of_the_week        1898 non-null   object 
 6   rating                 1898 non-null   object 
 7   food_preparation_time  1898 non-null   int64  
 8   delivery_time          1898 non-null   int64  
dtypes: float64(1), int64(4), object(4)
```

```
memory usage: 133.6+ KB
```

Observations: The datatypes are float64= floating point (or number that has decimal componet), int64= Interger (a whole number without a fractional componet), and Object= (a string or character data type).

In [9]:
```
#Checking for missing values in the data. Print the sum of null values for each column
df.isnull().sum()
```

Out[9]:
```
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 were no missing values in all 9 columns.

In [7]:
```
#describe() Will get the summary statistics of the numerical data statitical summary of the data
df.describe().T
```

Out[7]:

	count	mean	std	min	25%	50%	75%	max
order_id	1898.0	1.477496e+06	548.049724	1476547.00	1477021.25	1477495.50	1.477970e+06	1478444.00
customer_id	1898.0	1.711685e+05	113698.139743	1311.00	77787.75	128600.00	2.705250e+05	405334.00
cost_of_the_order	1898.0	1.649885e+01	7.483812	4.47	12.08	14.14	2.229750e+01	35.41
food_preparation_time	1898.0	2.737197e+01	4.632481	20.00	23.00	27.00	3.100000e+01	35.00
delivery_time	1898.0	2.416175e+01	4.972637	15.00	20.00	25.00	2.800000e+01	33.00

Observations: The minimum time it takes for food to prepared once an order is placed is 20 minutes. The Average is 27 minutes. The Maximum time is 35 minutes.

In [10]:
```
#.value_counts: counts the occurrence of each unique value in the Series.
rating_counts = df["rating"].value_counts
rating_counts()
```

Out[10]:
```
Not given    736
5            588
4            386
3            188
Name: rating, dtype: int64
```

Observations: 736 order are not given a rating. The remaning 1162 orders were rating 3-5. 5 being the most rate given by customers and 3 being the lowest rate given by customers.

Exploratory Data Analysis (EDA)

Univariate Analysis

Order ID

In [11]:
```
#check the count of unique order IDs
df['order_id'].nunique()
```

Out[11]:
```
1898
```

Oberservation:

Repeat Customers: With **1898 unique orders IDs from 1200 unique customers IDs**, it appears that some customers have **placed more than one order**. This is a positive indication as it suggests customer retention and satisfaction to some extent.

Customer ID

In [12]:
```
# check the count of unique number of customer IDs
df['customer_id'].nunique()
```

Out[12]:
```
1200
```

Observation:

Potential for Increased Frequency: The fact that many customers have ordered more than once suggests there might be potential to encourage even more repeat ordering.

Loyalty Programs: If not already in place, this might be an indication to initiate some sort of **loyalty or rewards program to incentivize repeat purchases.** Customers who have already ordered more than once may be more likely to join such a program, increasing their lifetime value.

Restaurant Name

In [13]:
```
# check the count of unique number of restaurant names, value counts, and countplot.
df['restaurant_name'].nunique()
```

Out[13]:
```
178
```

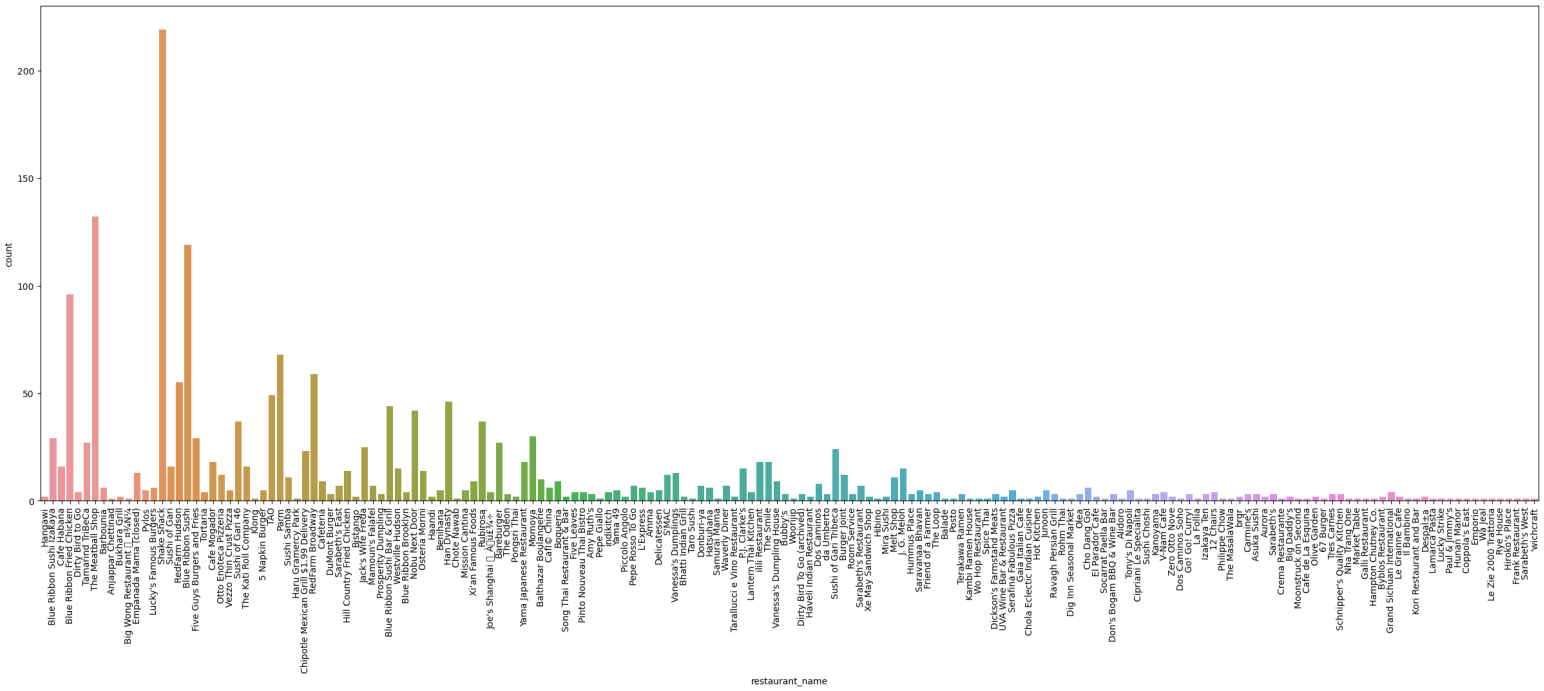
In [14]:
```
restaurant_name_counts = df["restaurant_name"].value_counts
restaurant_name_counts()
```

Out[14]:
```
Shake Shack                  219
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
                            ...
Sushi Choshi                   1
Dos Caminos Soho               1
La Follia                      1
Philippe Chow                  1
'wichcraft                     1
Name: restaurant_name, Length: 178, dtype: int64
```

In [15]:
```
plt.figure(figsize=(30,10))
sns.countplot(data=df, x='restaurant_name')
plt.xticks(rotation=90);
plt.show()
```

```
/Users/camilaconyers/opt/anaconda3/lib/python3.9/site-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 140 (\x8c) missing from current font.
```

```
  fig.canvas.print_figure(bytes_io, **kw)
/Users/camilaconyers/opt/anaconda3/lib/python3.9/site-packages/IPython/core/pylabtools.py:151: UserWarning: Glyph 142 (\x8e) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
```



Observation:

Popular Restaurants: Shake Shack seems to be the most popular restaurant with 219 orders, followed by The Meatball Shop and Blue Ribbon Sushi. These restaurants might be offering popular cuisines, have high customer satisfaction, and/or run effective promotions.

Potential Business Insights: FoodHub might want to investigate why some restaurants are receiving fewer orders. It could be due to **various reasons** like **cuisine type, price, customer service, delivery time, etc**. Addressing these issues could potentially increase their popularity and orders.

Promotion and Partnership Opportunities: FoodHub could be opportunity for FoodHub to **promote lesser-known or newer restaurants** to help increase their visibility and orders. Alternatively, FoodHub might want to form **strategic partnerships** with the most popular restaurants, given their apparent success on the platform.

Cuisine Type

In [16]:
```
# check the count of unique number of cuisine type, value counts, and countplot.
df['cuisine_type'].nunique()
```

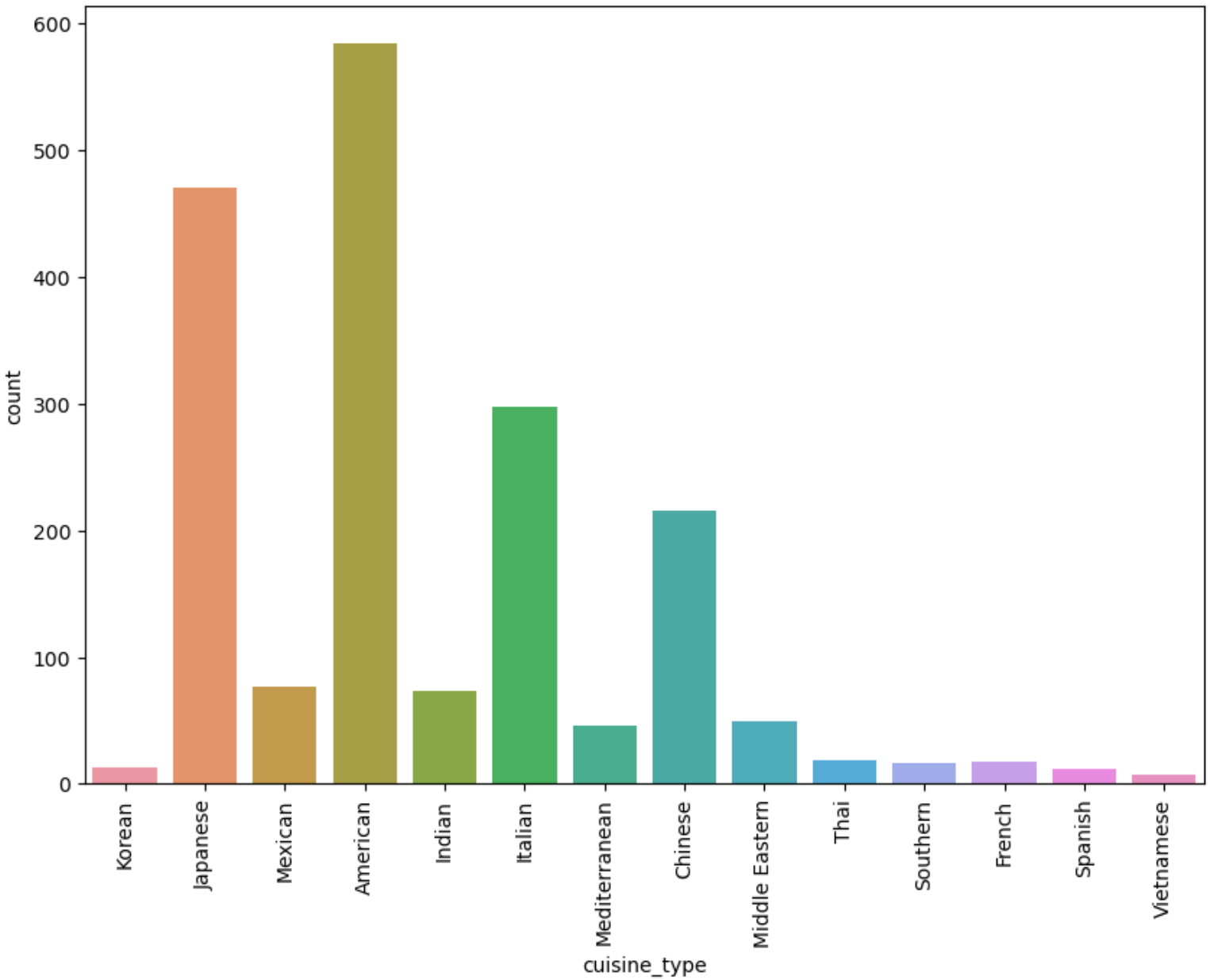
Out[16]: 14

In [17]:
```
cuisine_type_counts = df["cuisine_type"].value_counts
cuisine_type_counts()
```

Out[17]:
```
American          584
Japanese          470
Italian           298
Chinese           215
Mexican            77
Indian             73
Middle Eastern     49
Mediterranean      46
```

```
Thai                19
French              18
Southern            17
Korean              13
Spanish             12
Vietnamese           7
Name: cuisine_type, dtype: int64
```

In [18]:
```
plt.figure(figsize=(10,7))
sns.countplot(data=df, x='cuisine_type')
plt.xticks(rotation=90);
```



Observation :

The **Popularity** of **American cuisine** seems to be the most popular among customers, which could indicate a preference for American food among the customers of FoodHub.

Asian cuisine follows as the second and fourth most popular cuisine types.

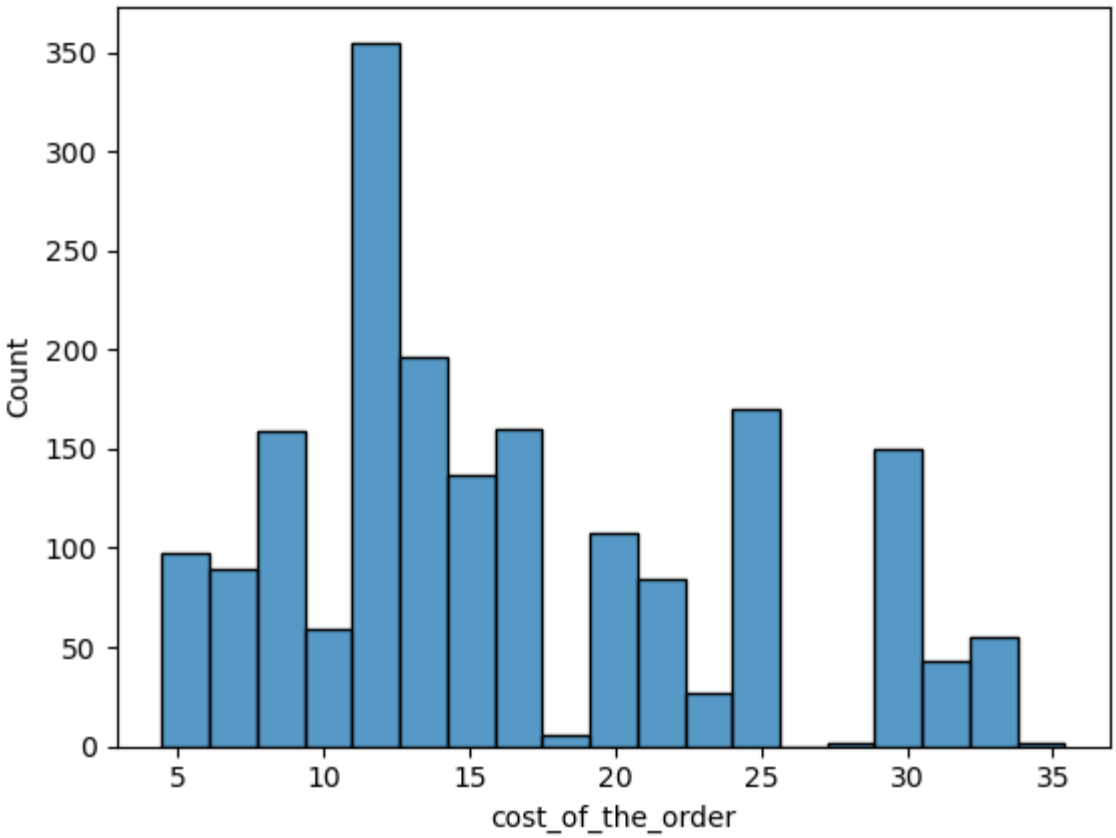
Euorpean cuisine follows as the third most popular type followed by French and Spanish cuisines.

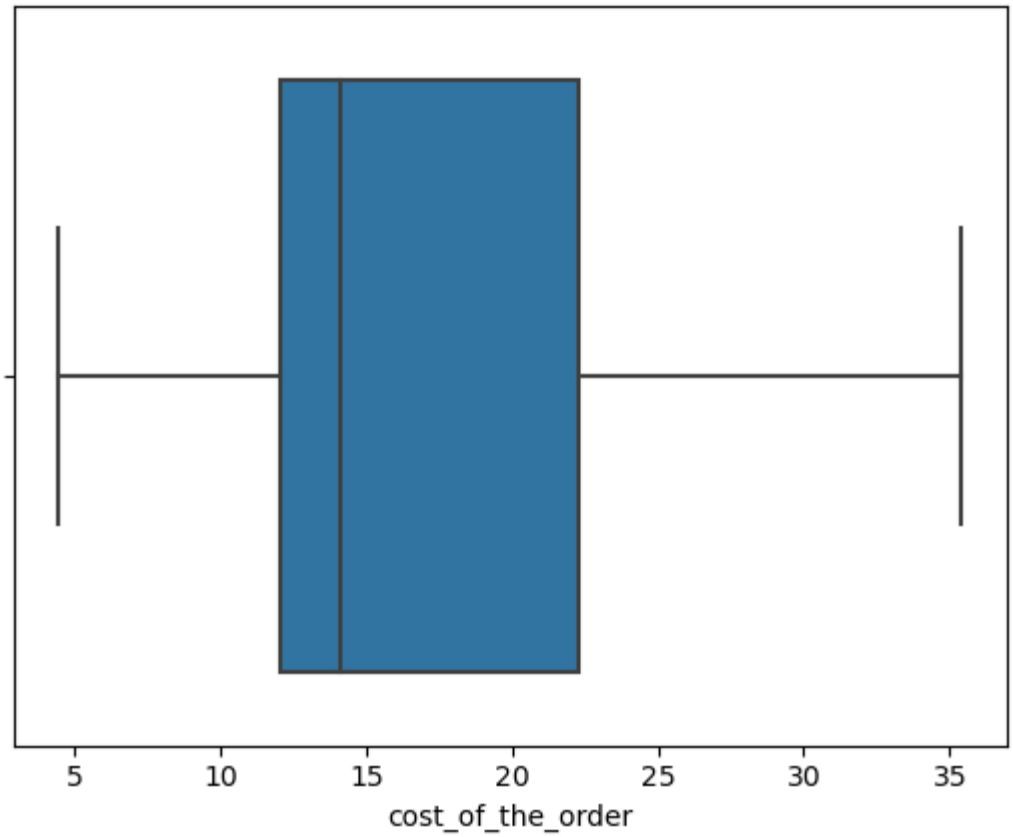
There is a **wide variety of demand** for the different cuisine. The gap of orders between the **American (highest) and Vietnamese (lowest) is quite large** and could suggest the platform might have a lot more restaurant offerings in certain cusines than others.

Potential Opporunities If the supply is low for cuisines like Vietnamese, Spanish, Korean, Southern, French, and Thai but the demand is potentially high, Food Hub could see a business opportunity. Might consider onboarding more restaurants offering these cusines

Cost of Order

In [19]:
```
# check the count of unique number of cost of orders, value counts, and countplot.
sns.histplot(data=df,x='cost_of_the_order')
plt.show()
sns.boxplot(data=df,x='cost_of_the_order')
plt.show()
```





Oberservation: The cost of the order data is positively (right) skewed.

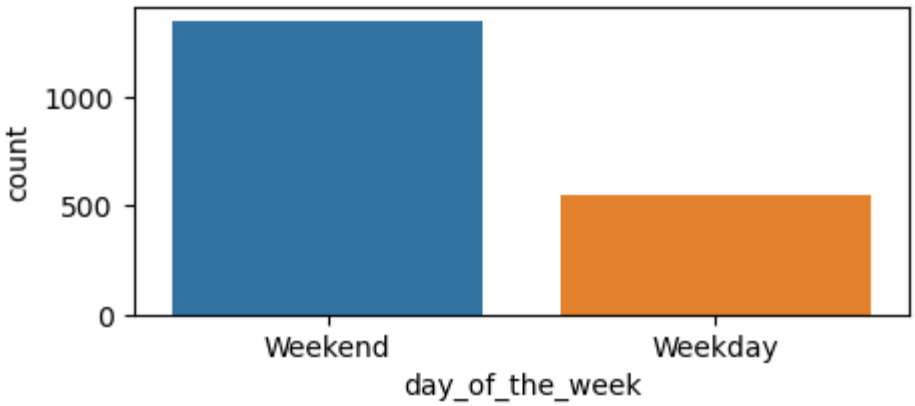
Day of the Week

In [20]:
```
# check the count of unique number of Days of the week, value counts, and countplot.
df['day_of_the_week'].nunique()
```

Out[20]:2

In [21]:
```
plt.figure(figsize = (5,2))
sns.countplot(data = df, x = 'day_of_the_week')
```

Out[21]:<AxesSubplot:xlabel='day_of_the_week', ylabel='count'>



Observation:

Higher Demand on Weekends: There's significantly more demand on **weekends** (1351 orders) compared to weekdays (547 orders). This could indicate that customers are more likely to order food using the FoodHub app during the weekend.

Potential Business Insights: The higher demand on weekends could be **driven by several factors**. People might have more leisure time, might be more likely to indulge, or might be more likely to spend more time with family or friends, **leading to larger orders**. If this trend is consistent, it can be used for **inventory management, staffing needs, and promotional activities.**

Resource Allocation: FoodHub could use this insight to better manage resources. For instance, ensuring enough delivery personnel on weekends to handle the increased number of orders.

Promotional Activities: FoodHub could consider **launching special weekday promotions** or offers to boost sales during the week. This could help **balance the demand** more evenly across the week and could lead to **increased overall sales.**

Data for Restaurants: This data could also be **shared with partnered restaurants**, allowing them to **better prepare for the demand increase over the weekend**. They could stock up on supplies or increase staff during those days to ensure they can meet the demand.

Rating

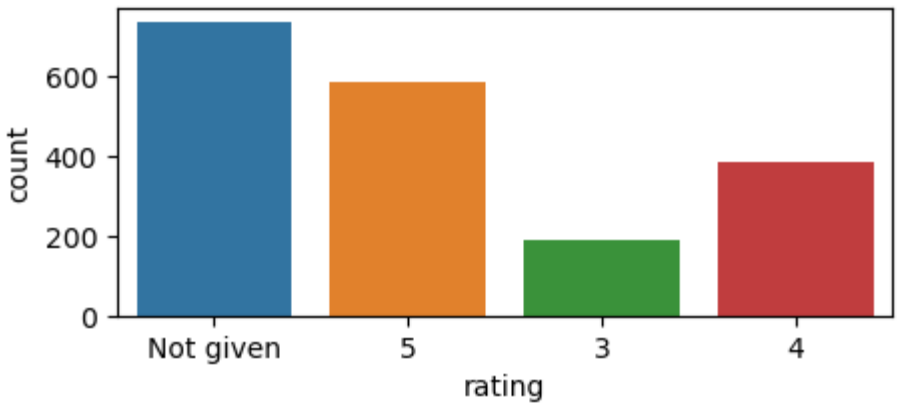
In [22]:
```
# check the count of unique number of ratings, value counts, and countplot.
df['rating'].nunique()
```

Out[22]:4

In [23]:
```
rating_counts = df["rating"].value_counts
rating_counts()
```

Out[23]:
```
Not given    736
5            588
4            386
3            188
Name: rating, dtype: int64
```

In [24]:
```
plt.figure(figsize=(5,2))
sns.countplot(data=df, x='rating');
```



Obersation:

Not Given Ratings: A substantial number of orders (736) were **not rated**. This might suggest that many customers are not providing feedback on their orders, which could be due to several reasons - **they might not have time, they might not find the process easy or rewarding, or they might not feel motivated to rate unless their experience was particularly good or bad.**

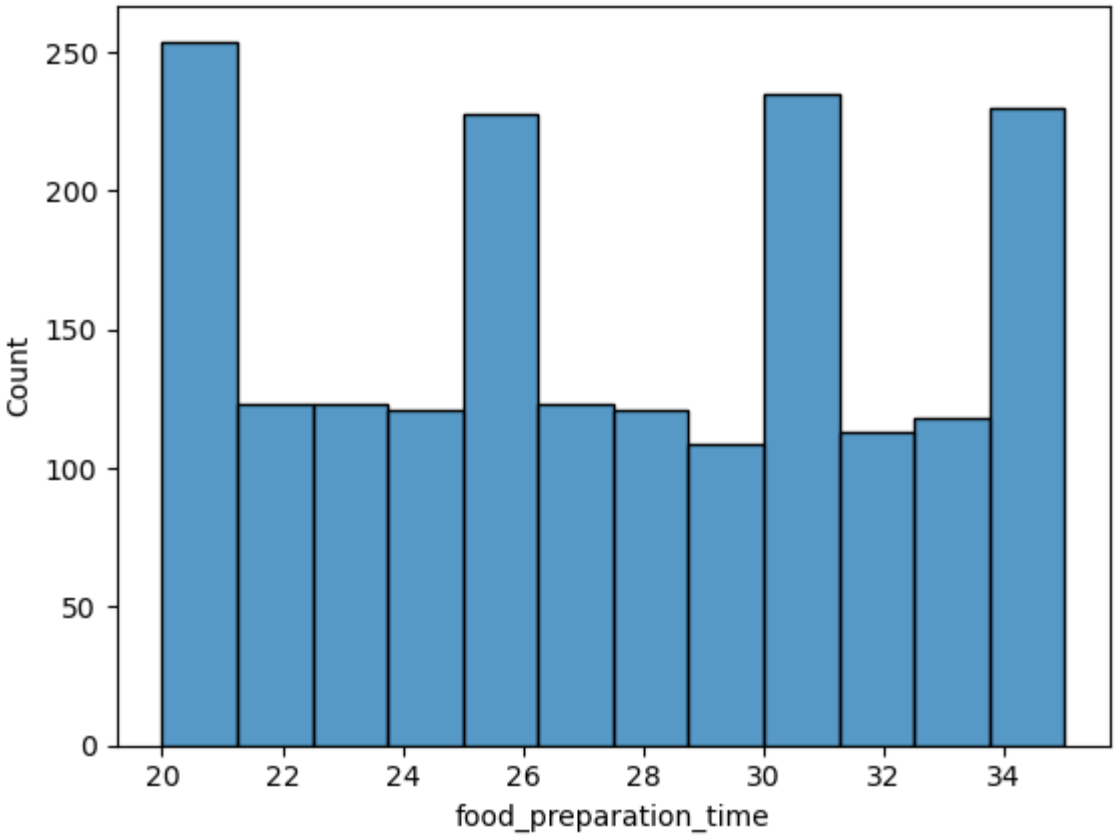
High Ratings: Among the orders that were rated, the **majority** of them have received high ratings (5 and 4). This is a **positive sign** and suggests that customers are generally satisfied with their orders.

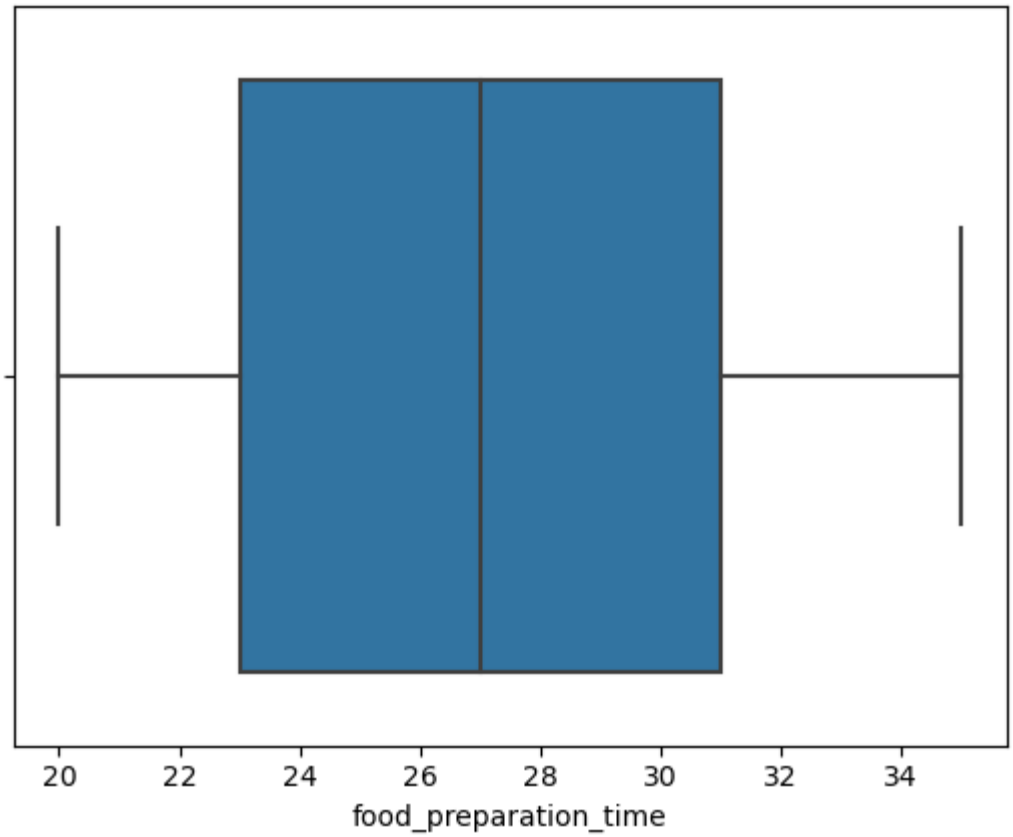
Low Ratings: A smaller portion of orders have received a rating of 3, which could be **considered average or below average**.

Food Preparation

In [25]:
```
# Visualize the Food preparation using a histplot and boxplot.

sns.histplot(data=df,x='food_preparation_time',stat='count')
plt.show()
sns.boxplot(data=df,x='food_preparation_time')
plt.show()
```

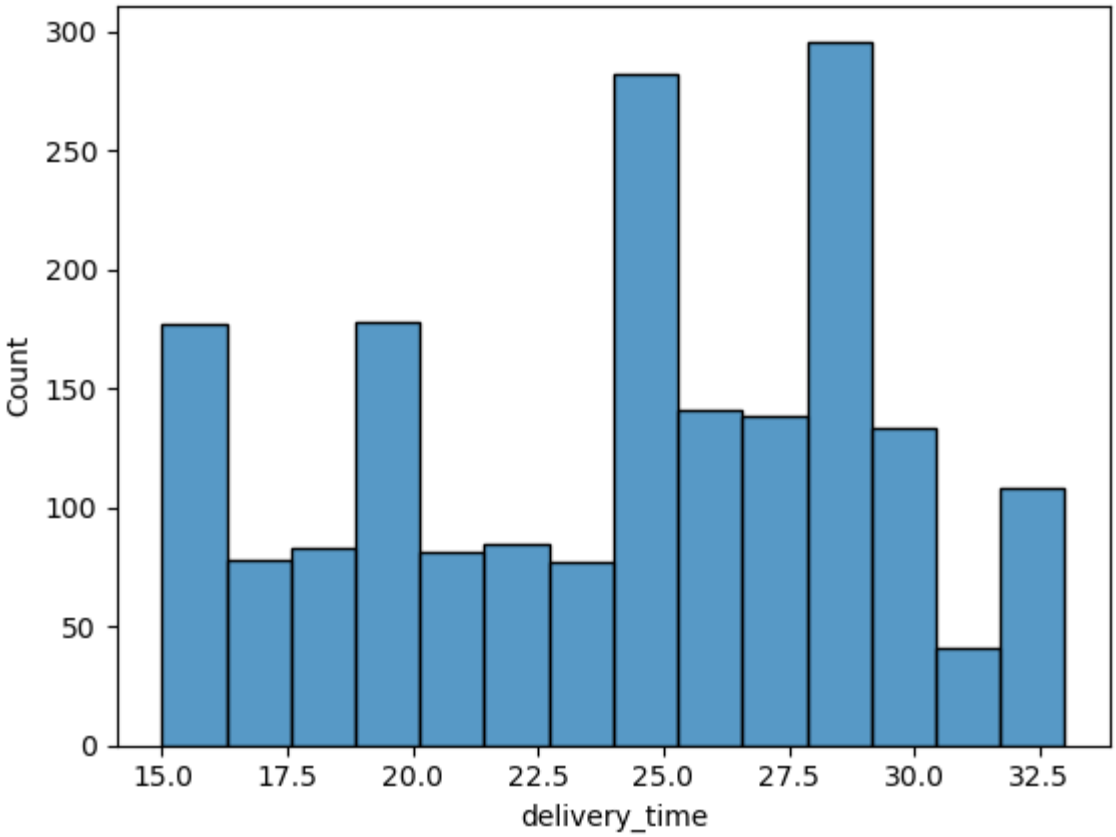


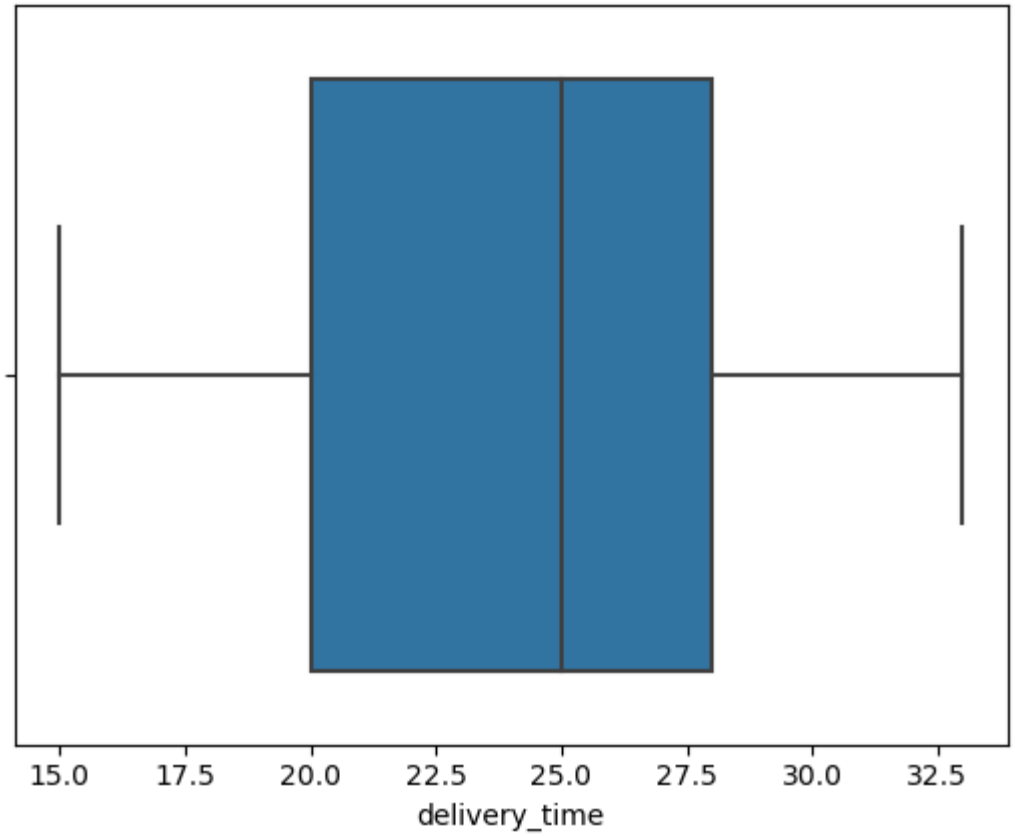


Observation : The food preparation time data has central tendency.

Delivery Time

In [26]:
```
# Visualize the Delivery time using a histplot and boxplot.
sns.histplot(data=df,x='delivery_time')
plt.show()
sns.boxplot(data=df,x='delivery_time')
plt.show()
```





:**Observation** The delivery time data is negatively (left) skewed.

In [27]:
```
# Get top 5 restaurants with highest number of orders
df['restaurant_name'].value_counts().head(5)
```

Out[27]:
```
Shake Shack                  219
```

```
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
Name: restaurant_name, dtype: int64
```

Observations:

Popular Restaurants: These restaurants might be offering popular cuisines, have high customer satisfaction, and/or run effective promotions.

Potiential Partneership: FoodHub might want to form strategic partnerships with the most popular restaurants, given their apparent success on the platform.

In [28]:
```
#Get most popular cuisine on weekends; code to check unique values for the cuisine type on weekend
df_weekend = df[df['day_of_the_week'] == 'Weekend']
cuisine_counts_weekend = df_weekend['cuisine_type'].value_counts()
print(cuisine_counts_weekend)
```

```
American          415
Japanese          335
Italian           207
Chinese           163
Mexican            53
Indian             49
Mediterranean      32
Middle Eastern     32
Thai               15
French             13
Korean             11
Southern           11
Spanish            11
Vietnamese          4
Name: cuisine_type, dtype: int64
```

Observations:

High Demand During Weekends: For each cuisine, a significant proportion of the weekly orders are made during the weekends. For instance, out of 584 total American cuisine orders, 415 occur during the weekend. This indicates a higher demand during weekends across all types of cuisines, which is expected as people generally have more leisure time.

Relative Popularity: The relative popularity of cuisines seems to stay the same during the weekends and the week overall. American, Japanese, and Italian cuisines are the most popular in both cases.

Opportunities for Promotion: If the demand for certain cuisines significantly increases over the weekend (like American and Japanese), there could be opportunities to run special weekend promotions for these cuisines to increase sales even further.

In [29]:
```
# Write the appropriate column name to get the orders having cost above $20; Calculate the number of total orders where the cost is above 20 dollars;Calculate percentage of such orders in the dataset
df_greater_than_20 = df[df['cost_of_the_order']>20]


print('The total number of orders that cost more than 20 dollars is:', df_greater_than_20.shape[0])
```

```
percentage = (df_greater_than_20.shape[0] / df.shape[0]) * 100

print("Percentage of orders more than 20 dollars:", round(percentage, 2), '%')
```

```
The total number of orders that cost more than 20 dollars is: 555
Percentage of orders more than 20 dollars: 29.24 %
```

Observations:

Nearly a third of all orders (29.24%) cost more than $20. This is a significant portion, and it indicates that many customers are making relatively high-value purchases.

Potential Upselling Opportunities: This suggests that there might be opportunities for upselling or cross-selling. If customers are already spending more than $20, they might be willing to add on additional items to their orders if they see value in them.

Revenue Contribution: While it's not clear without more data, it's likely that these high-value orders are contributing significantly to the company's revenue. This would be especially true if the $20 more orders are much larger than 20.00 on average.

Targeted Marketing: Since these customers are willing to make larger purchases, they might be a good target for marketing higher-priced items or specials. However, more analysis would be needed to validate this idea.

Pricing Strategy: Knowing that a good percentage of orders are above $20, it would be worth examining if a revised pricing strategy could increase profits. For example, could a small increase in prices (which might not significantly impact order volumes) result in a substantial increase in revenues?

Further Analysis: To fully understand the significance of these $20+ orders, it would be worth exploring these orders in more detail. Are certain cuisines or dishes more likely to lead to high-value orders? Are these orders concentrated among a small number of customers or widely distributed? Understanding these factors could lead to actionable business insights.

In [30]:
```
#The function will obtain the mean delivery time
mean_delivery_time = df['delivery_time'].mean()
print('The mean order dlivery time is', round(mean_delivery_time, 2), 'minutes')
```

```
The mean order dlivery time is 24.16 minutes
```

Observations:

Average Delivery Time: The mean delivery time of approximately 24 minutes can be used as a benchmark for your service. This can provide a general expectation for customers about how long they might wait for their orders, provided there are no extenuating circumstances.

Operational Efficiency: A delivery time of about 24 minutes suggests relatively efficient operations, as long as the food quality and order accuracy are maintained. It might indicate a good logistical setup, effective coordination between restaurants and delivery personnel, and an overall well-managed process.

Customer Satisfaction: Delivery time is a crucial aspect of customer satisfaction. If this average delivery time is acceptable or even impressive to your customers, it could be a strong selling point. If it's not, it might be an area to focus on for improvement.

Comparison with Industry Standards: This figure would be more meaningful if compared with industry standards or competitors' delivery times. If your delivery time is faster, it could be a competitive advantage; if it's slower, it might indicate a need for improvement.

Further Analysis: FoodHub may consider breaking down delivery times by factors like the day of the week, cuisine type, or restaurant. This might reveal patterns that could be used to further improve delivery times or manage customer expectations.

In [31]:
```
#Get the counts of  each customer_id; get the top 3 cmost frequent customers
top_customers = df['customer_id'].value_counts().head(3)
```

```
print(top_customers)
```

```
52832    13
47440    10
83287     9
Name: customer_id, dtype: int64
```

Observations:

The data suggests that these three customers (with IDs 52832, 47440, and 83287) are the most frequent, with 13, 10, and 9 orders placed respectively.

Insights: This may encourage higher spending, increase cusotmer retention, and inspire positive word of mouth markting.

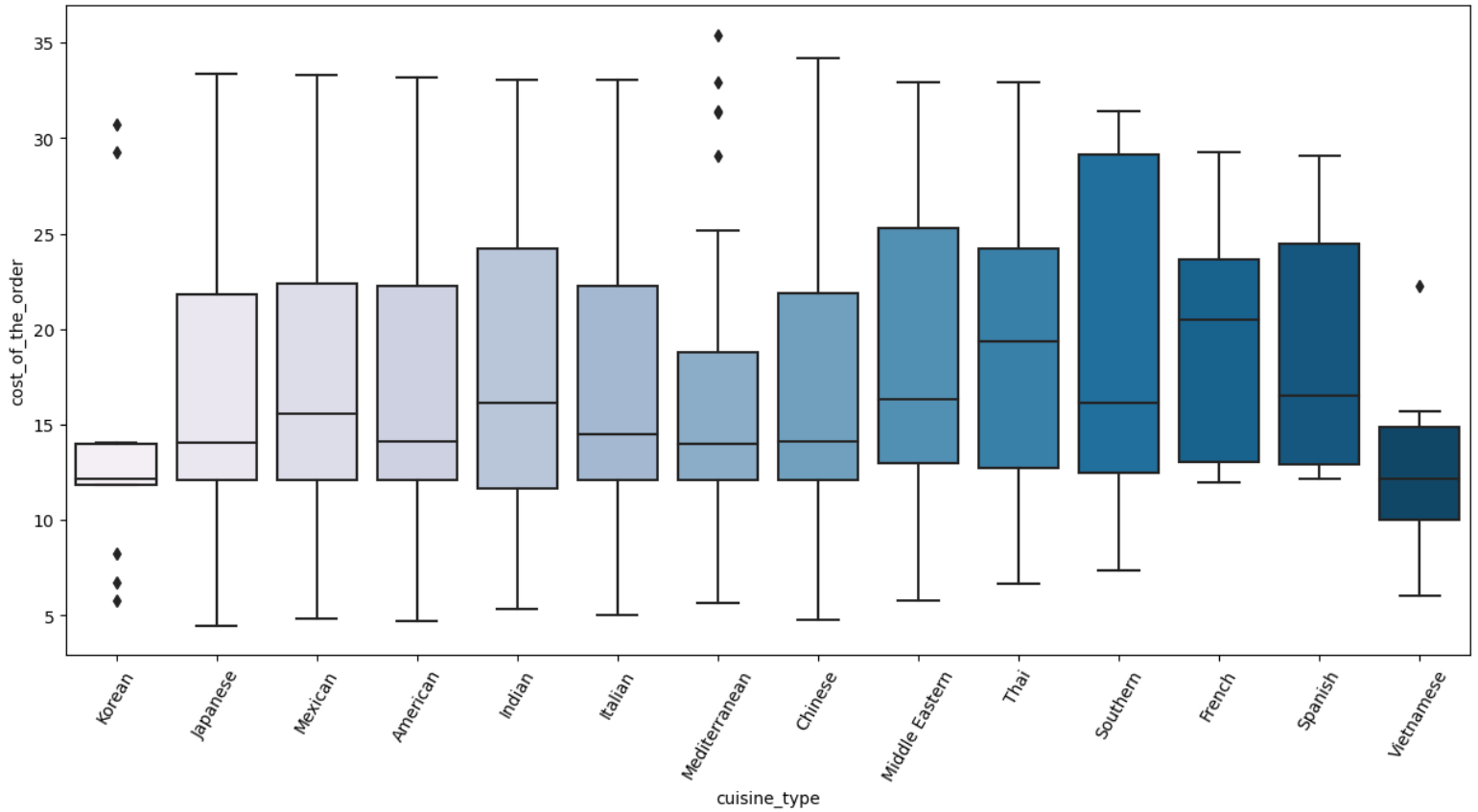
Data for Future Analysis: Tracking the behavior of these customers after they receive the discount could provide valuable data for future analysis. For example, do they order more often or spend more money after receiving the discount? This information could guide future decisions about discounts and promotions.

However, it's important to keep in mind that while these customers are valuable due to their frequency of orders, they may not necessarily be the ones generating the most revenue. **Their average order value could be lower than less frequent customers**. Therefore, it would be beneficial to consider both frequency and order value in developing customer loyalty rewards and incentives.

Multivariate Analysis

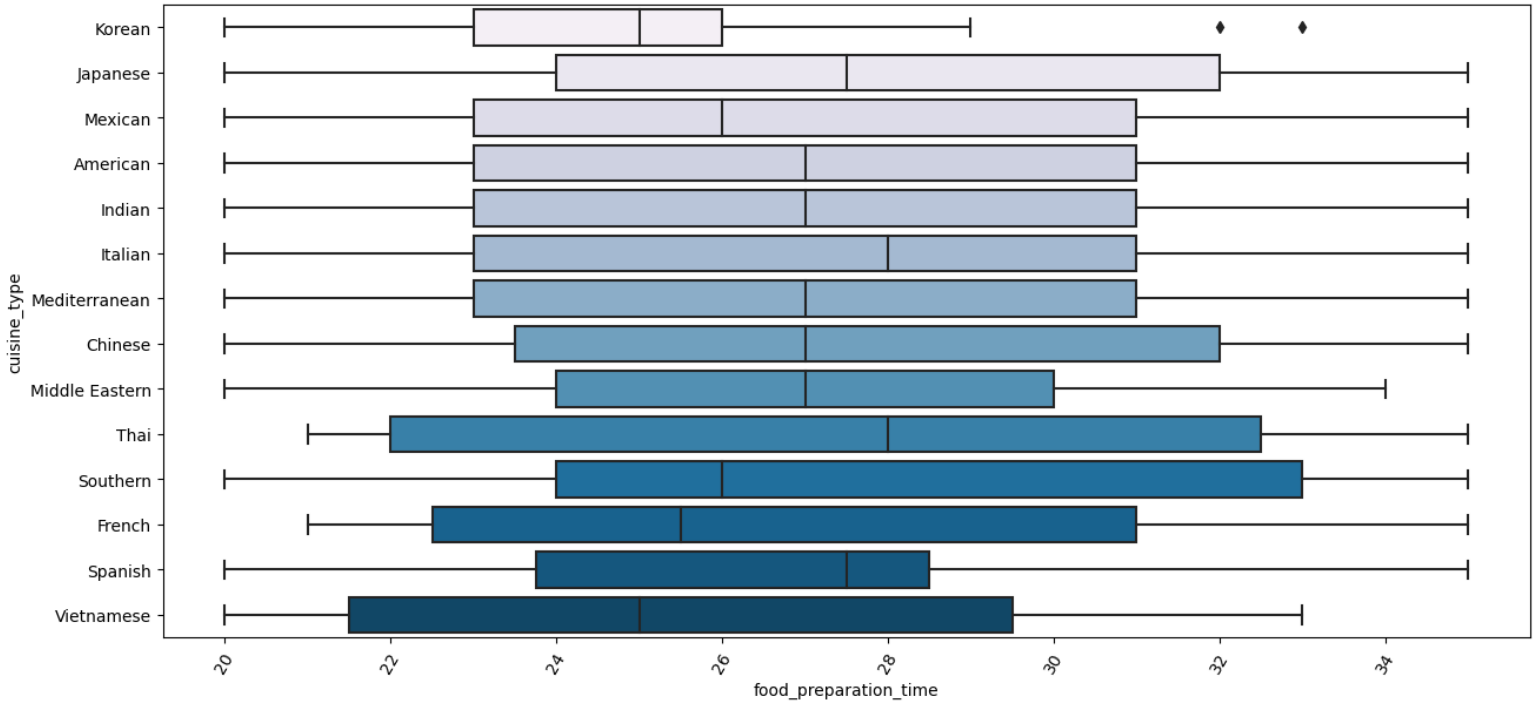
Cuisine vs Cost of the order

In [32]:
```
# Relationship between cost of the order and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(x = "cuisine_type", y = "cost_of_the_order", data = df, palette = 'PuBu')
plt.xticks(rotation = 60)
plt.show()
```



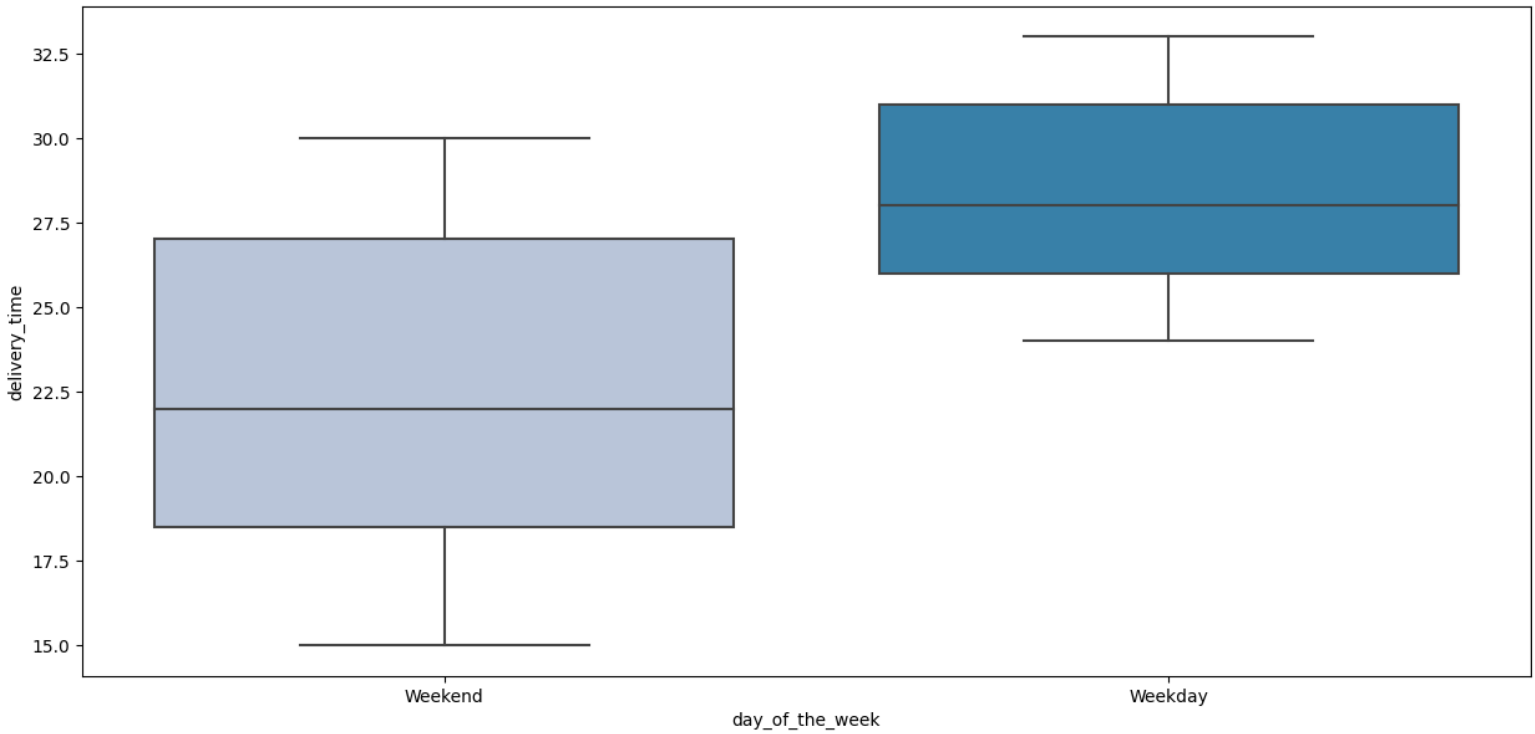
Food Preparation Time vs Cuisine Type

In [33]:
```
# Relationship between Food preparation and cuisine type
plt.figure(figsize=(15,7))
sns.boxplot(x = 'food_preparation_time' , y = 'cuisine_type' , data= df, palette =
'PuBu')
plt.xticks(rotation = 60)
plt.show()
```



Day of the Week vs Delivery Time

In [45]:
```
# Relationship between Day of the week and Delivery Time
plt.figure(figsize=(15,7))
sns.boxplot (x = 'day_of_the_week', y = 'delivery_time', data = df, palette= 'PuBu')
plt.show()
```



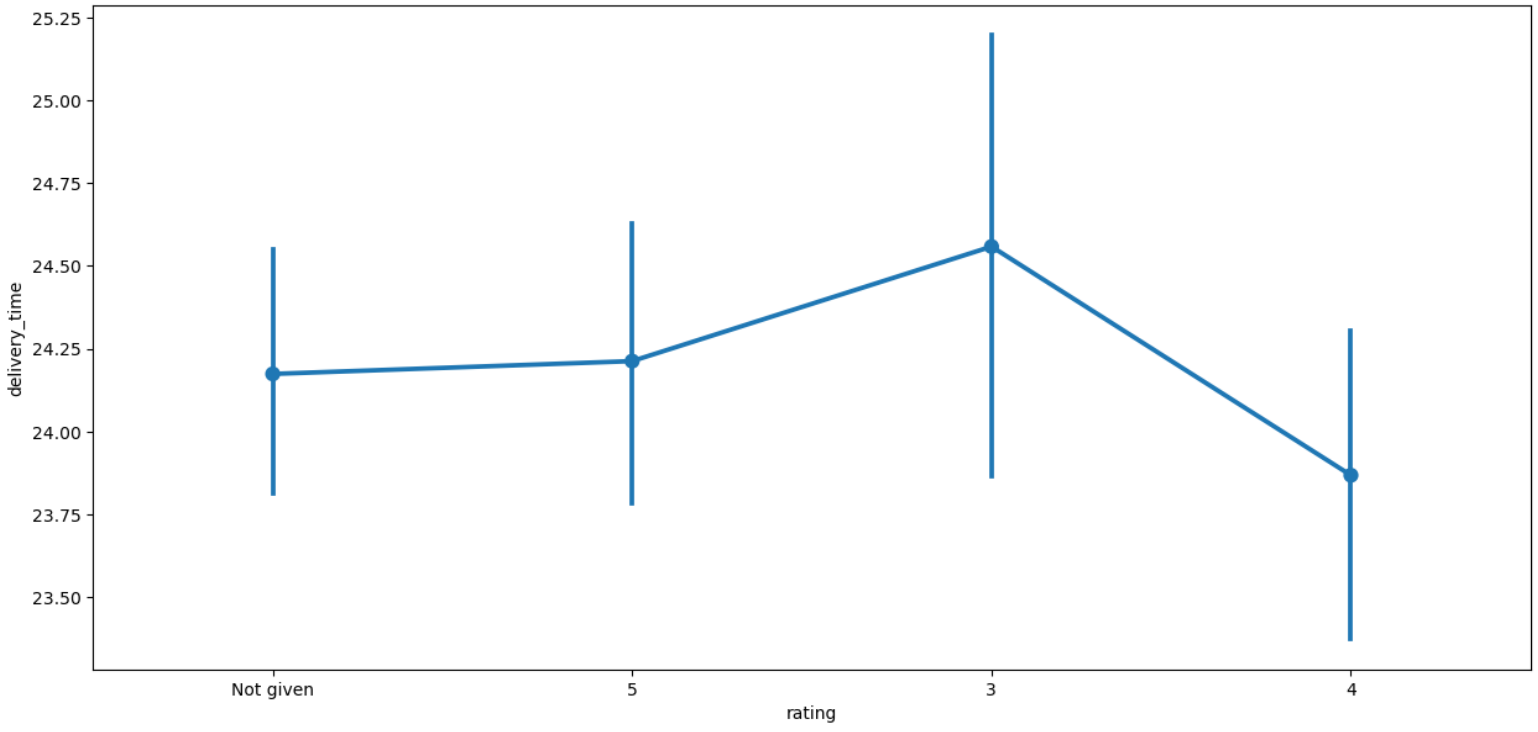
Revenue Generated per Restaruants

In [35]:
```
#relationship of the total order cost for each restaurant and then sorting these totals.
df.groupby(['restaurant_name'])['cost_of_the_order'].sum().sort_values(ascending = False).head(14)
```

Out[35]:
```
restaurant_name
Shake Shack                      3579.53
The Meatball Shop                2145.21
Blue Ribbon Sushi                1903.95
Blue Ribbon Fried Chicken        1662.29
Parm                             1112.76
RedFarm Broadway                  965.13
RedFarm Hudson                    921.21
TAO                               834.50
Han Dynasty                       755.29
Blue Ribbon Sushi Bar & Grill     666.62
Rubirosa                          660.45
Sushi of Gari 46                  640.87
Nobu Next Door                    623.67
Five Guys Burgers and Fries       506.47
Name: cost_of_the_order, dtype: float64
```

Rating vs Delivery Time

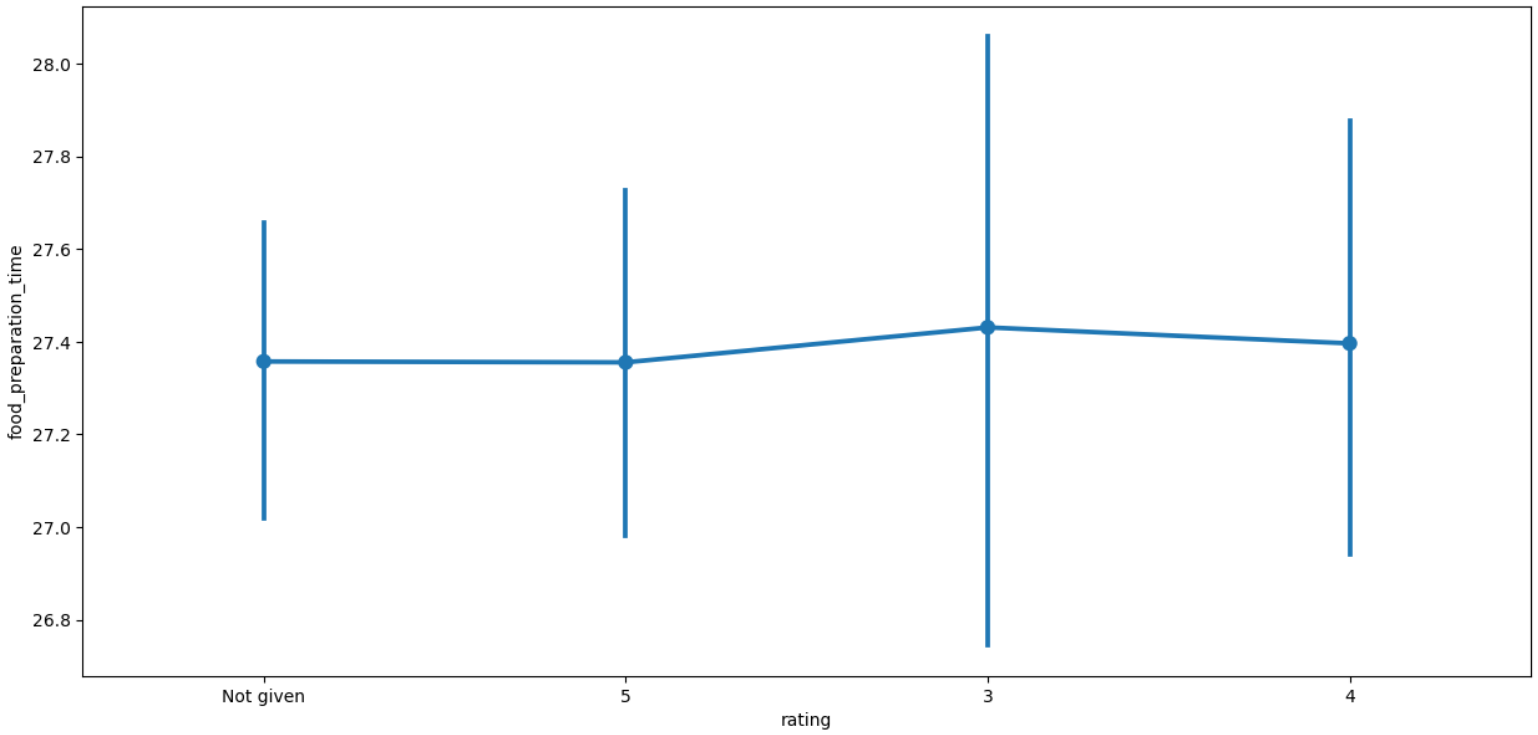
In [36]:
```
#Relationship between rating and Delivery Time
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating', y = 'delivery_time', data = df)
plt.show()
```



Rating vs Food preparation Time

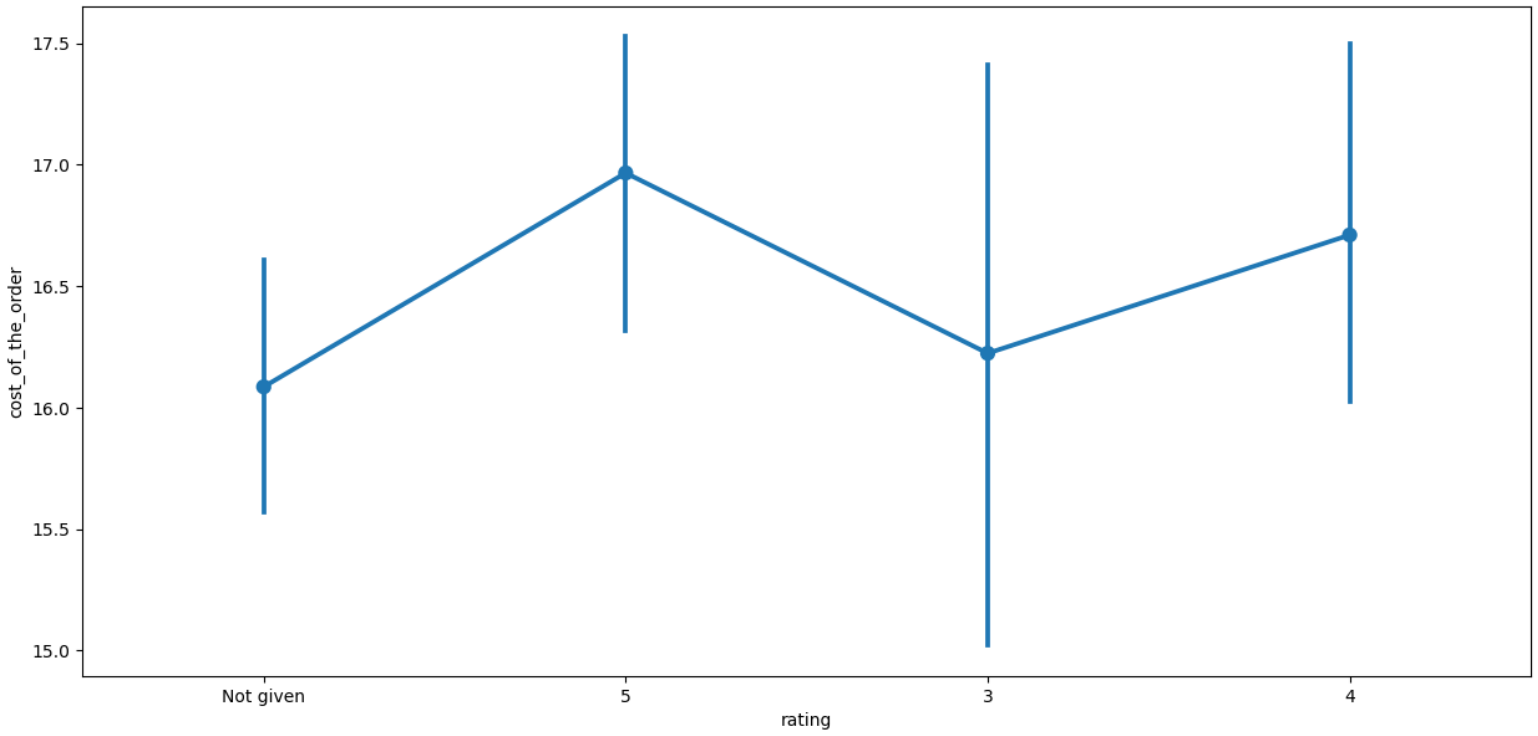
In [37]:
```
# Relationship between Rating and Food preperation
```

```
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating' , y = 'food_preparation_time' , data=df)
plt.show()
```



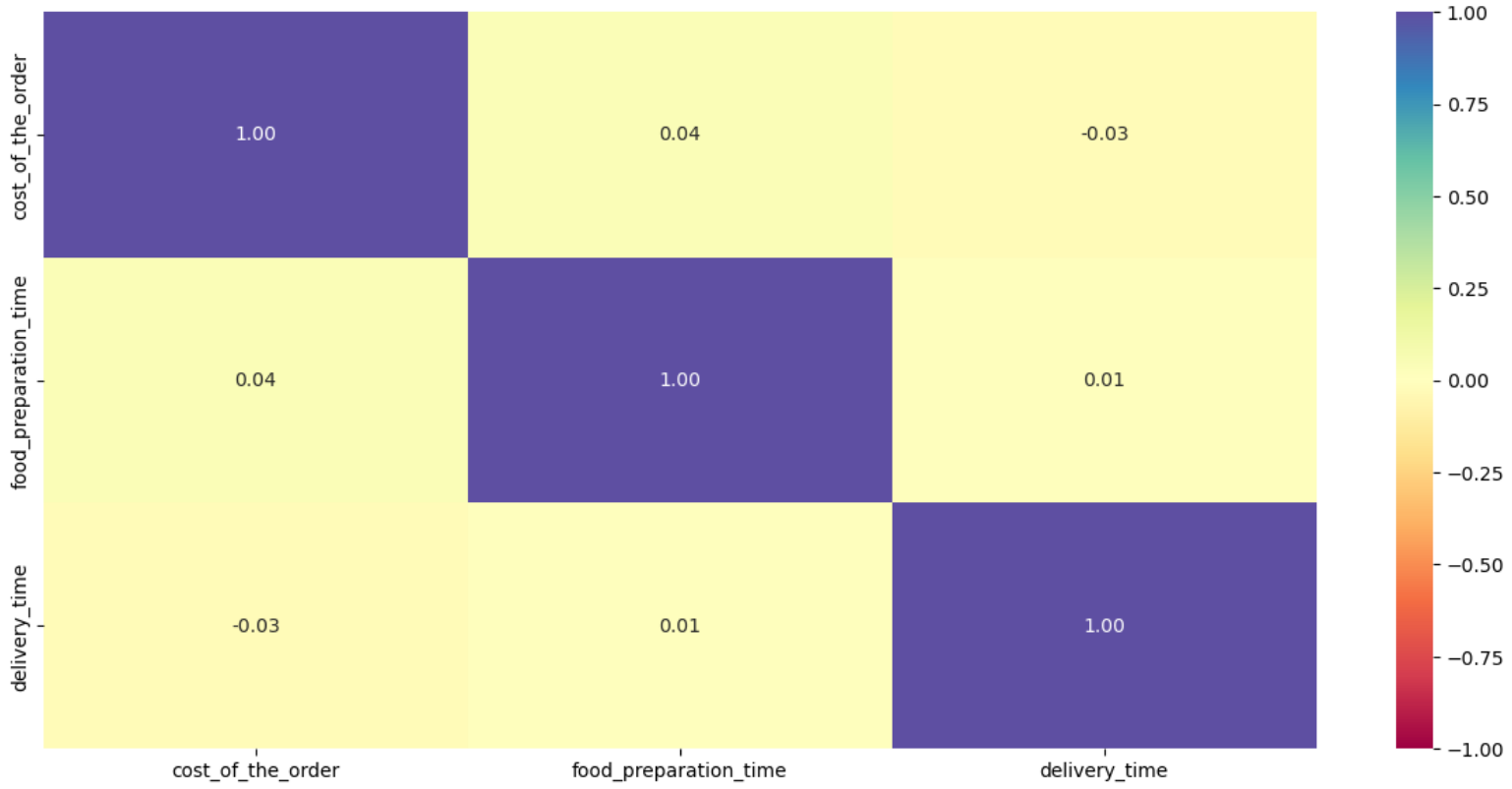
Rating vs Cost of the order

In [38]:
```
# Relationship between Rating and cost of the order
plt.figure(figsize=(15, 7))
sns.pointplot(x = 'rating' , y = 'cost_of_the_order' , data = df)
plt.show()
```



Correlation among variables

In [39]:
```
#Plot the heatmap
col_list = ['cost_of_the_order', 'food_preparation_time', 'delivery_time']
plt.figure(figsize=(15, 7))
sns.heatmap(df[col_list].corr(), annot=True, vmin=-1, vmax=1, fmt=".2f",
cmap="Spectral")
plt.show()
```



Question 13: The company wants to provide a promotional offer in the advertisement of the restaurants. 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. [3 marks]

In [40]:
```
# Filter the rated restaurants
df_rated = df[df['rating'] != 'Not given'].copy()

# Convert rating column from object to integer
df_rated['rating'] = df_rated['rating'].astype('int')

# Create a dataframe that contains the restaurant names with their rating counts
df_rating_count = df_rated.groupby(['restaurant_name'])
['rating'].count().sort_values(ascending = False).reset_index()
df_rating_count.head()
```

Out[40]:

	restaurant_name	rating
0	Shake Shack	133
1	The Meatball Shop	84

2	Blue Ribbon Sushi	73
3	Blue Ribbon Fried Chicken	64
4	RedFarm Broadway	41

In [41]:
```
# Get the restaurant names that have rating count more than 50
rest_names = df_rating_count[df_rating_count['rating'] > 50]['restaurant_name']

# Filter to get the data of restaurants that have rating count more than 50
df_mean_4 = df_rated[df_rated['restaurant_name'].isin(rest_names)].copy()

# Group the restaurant names with their ratings and find the mean rating of each restaurant
df_mean_4.groupby(['restaurant_name'])['rating'].mean().sort_values(ascending = False).reset_index().dropna()
```

Out[41]:

	restaurant_name	rating
0	The Meatball Shop	4.511905
1	Blue Ribbon Fried Chicken	4.328125
2	Shake Shack	4.278195
3	Blue Ribbon Sushi	4.219178

Observations:

Promotional Offer Candidates: Shake Shack, The Meatball Shop, Blue Ribbon Sushi, and Blue Ribbon Fried Chicken all have more than 50 ratings. **RedFarm Broadway** does not meet this criterion as it only has 41 ratings.

Looking at the average ratings, **The Meatball Shop**, Blue Ribbon Fried Chicken, Shake Shack, and Blue Ribbon Sushi all have an average rating greater than 4.

Combining both criteria, the restaurants that should receive the promotional offer are The Meatball Shop, Blue Ribbon Fried Chicken, Shake Shack, and Blue Ribbon Sushi, as they have both more than 50 ratings and an average rating greater than 4.

The restaurant with the highest average rating among these is The Meatball Shop, while the restaurant with the highest number of ratings is Shake Shack. **These two restaurants could be the most popular and might bring in more customers with the promotional offer.**

In [43]:
```
#function to determine the revenue and compute the revenue
def compute_rev(x):
    if x > 20:
        return x*0.25
    elif x > 5:
        return x*0.15
    else:
        return x*0


df['Revenue'] = df['cost_of_the_order'].apply(compute_rev)

#Function to get the total revenue
total_rev = df['Revenue'].sum()
print('The net revenue is around', round(total_rev, 2), 'dollars')
```

```
The net revenue is around 6166.3 dollars
```

Observations:

Profitability: This value represents the revenue that the company earns from restaurants. If we have data on the company's costs or expenses, we could further analyze profitability.

Revenue distribution: It would be beneficial to know how this revenue is distributed across different restaurants, cuisines, or days of the week. For instance, are there specific restaurants or cuisines that generate significantly more revenue? Or are there certain days where the company sees a surge in revenue?

Impact of pricing strategy: The company's strategy to charge 25% on orders over $20 and 15% on orders over $5 seems to be bringing in revenue. However, it's hard to judge the effectiveness of this strategy without comparing to alternative strategies. For instance, would a flat fee per order regardless of cost generate more revenue?

In [44]:
```
# Calculate total delivery time and add a new column to the dataframe df to store the
total delivery time
df['total_time'] = df['food_preparation_time'] + df['delivery_time']

#find the percentage of orders that have more than 60 minutes of total delivery time
df_greater_than_60 = df[df['total_time']>60]

print('The total number of orders that took more than 60 minutes to get delivered
is:', df_greater_than_60.shape[0])

percentage = (df_greater_than_60.shape[0] / df.shape[0]) * 100

print("Percentage of orders that took more than 60 minutes to get delivered:",
round(percentage, 2), '%')
```

```
The total number of orders that took more than 60 minutes to get delivered is: 200
Percentage of orders that took more than 60 minutes to get delivered: 10.54 %
```

Observations:

Approximately 10.54% of all orders take more than 60 minutes to get delivered from the time the order is placed.

This implies **close to one in ten** of the customers are waiting for more than an hour for their orders to arrive.

Depending upon the customer's perspective and the **nature of the food** being ordered, this could potentially be seen as an area of improvement. For instance, if the company is dealing with orders that are typically expected to be delivered quickly (like fast food), then a delivery time of more than an hour might be considered too long by some customers.

The company could **benefit from investigating** the causes of these **longer delivery times**. Are they due to longer food preparation times at certain restaurants, or are they caused by delays in the delivery process itself? Understanding this can help the company take targeted steps to reduce delivery times and improve customer satisfaction.

Also, this analysis may be used by the company to **improve their delivery time predictions.** This could help manage **customer expectations and increase overall satisfaction** even if the actual delivery times can't be reduced significantly.

However, if these longer delivery times are isolated to certain cuisines that typically take longer to prepare, or if they are a result of large orders that take longer to prepare and deliver, then these longer delivery times might be acceptable and expected by the customers.

In [45]:
```
#Code below will get the the mean delivery time on weekdays and print it

print('The mean delivery time on weekends is around',
      round(df[df['day of the week'] == 'Weekend']['delivery time'].mean()),
```

```
          'minutes')
```

```
The mean delivery time on weekends is around 22 minutes
```

In [46]:
```
#Code below will get the mean delivery time on weekends and print it
print('The mean delivery time on weekdays is around',
      round(df[df['day_of_the_week'] == 'Weekday']['delivery_time'].mean()),
      'minutes')
```

```
The mean delivery time on weekdays is around 28 minutes
```

Observations:

Comparing the mean delivery times provided, the data suggests that the **delivery time during the weekends is quicker compared to weekdays.** This could be due to a variety of factors such as lower traffic conditions, more staff available during the weekends, or even the nature of orders being different on weekends as compared to weekdays.

Also, the company might want to investigate this difference further. If it's possible to maintain the quicker delivery times from the weekend on weekdays, it could potentially lead to improved customer satisfaction. If the quicker times on the weekend are due to factors that cannot be replicated during the week, the company may want to communicate this to customers, to manage their expectations about delivery times.

Conclusion and Recommendations

Conclusions:

- From the thorough analysis of FoodHub's data, it's clear that the platform is providing a vital service in a bustling city like New York where the demand for online food delivery is high. The significant number of unique customers and orders suggest that the platform is extensively utilized, pointing towards the success of FoodHub's business model.

Key observations include:

- A majority of orders are placed during the weekends, indicating that FoodHub is a preferred choice for customers' leisure time or when they're likely away from their usual routine.
- A significant portion of orders are of cuisines like American, Japanese, and Italian, suggesting a diverse clientele with varied food preferences.
- Restaurants with higher ratings and more ratings tend to be more popular, reflecting the importance of quality and customer feedback in driving order volumes.
- The revenue generation from orders above $20 is substantial due to the higher margin on these orders.
- While the average delivery time is under the acceptable limit of 30 minutes, around 10% of orders take more than an hour, which suggests a potential area for improvement.

In conclusion, FoodHub's business is well-positioned in the online food delivery market, but there is room for further growth and efficiency. Focusing on promoting high-rated restaurants, encouraging customer feedback, diversifying restaurant and cuisine options, and improving delivery times could be key areas to consider for enhancing customer satisfaction and business expansion.

Recommendations:

- **Increase American and Japanese Cuisine Options**: Partner with more restaurants offering these cuisines and offer promotions to increase orders from these types of restaurants.
- **Promote High-rated Restaurants:** Advertise these restaurants more to boost customer satisfaction.
- **Investigate Delivery Time Differences:** Identify the reasons behind faster weekend delivery times and if possible, try to replicate these conditions on weekdays.

- **Encourage Order Ratings:** Provide incentives for customers to rate their orders, giving the company more data to identify improvement areas.
- **Loyalty or Rewards Program:** Consider a program to retain the top customers who place the most orders, encouraging them to order more frequently.
- **Re-evaluate Revenue Generation Strategy:** The company may want to explore if changing the fee structure could potentially increase revenue without negatively impacting customer orders.
- **Reduce Delivery Time:** Investigate why approximately 10% of orders take more than an hour to deliver and take targeted measures to reduce this percentage.
