Project Python Foundations: 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. Suppose you are hired as a Data Scientist in this company and the Data Science team has shared some of the key questions that need to be answered. 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_of_the_order: 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 [ ]:
```
# Installing the libraries with the specified version.
# Running in Google Colab so commenting out this.
#!pip install numpy==1.25.2 pandas==1.5.3 matplotlib==3.7.1 seaborn==0.13.1 -q --user
```

Note: *After running the above cell, kindly restart the notebook kernel and run all cells sequentially from the start again.*

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

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

Understanding the structure of the data

In [2]:
```
# Mounting Google Drive and setting path variable to the data location
from google.colab import drive
drive.mount('/content/drive')
path = '/content/drive/MyDrive/PGPAIML/Project-1/'
```

```
Mounted at /content/drive
```

In [3]:
```
# Loading the data in a DataFrame
```

```
data = pd.read_csv(path+'/foodhub_order.csv')
```

In [ ]:
```
# Displaying the first 5 rows
data.head()
```

Out[ ]:

	order_id	customer_id	restaurant_name	cuisine_type	cost_of_the_order	day_of_the_week	rating	food_preparation_time	delivery_time
0	1477147	337525	Hangawi	Korean	30.75	Weekend	Not given	25	20
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.08	Weekend	Not given	25	23
2	1477070	66393	Cafe Habana	Mexican	12.23	Weekday	5	23	28
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.20	Weekend	3	25	15
4	1478249	76942	Dirty Bird to Go	American	11.59	Weekday	4	25	24

Question 1: How many rows and columns are present in the data? [0.5 mark]

In [ ]:
```
# Displaying the number of rows and columns using the shape property of the DataFrame.
data.shape
```

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

Observations: This dataset have 1898 rows and 9 columns.

Question 2: What are the datatypes of the different columns in the dataset? (The info() function can be used) [0.5 mark]

In [ ]:
```
# Displaying all the columns and their data types.
data.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: In this dataset, we have 4 int columns, 1 float (cost_of_the_order), and 4 string data type columns. The entire dataset needs 133.6+ KB of memory at runtime. The column "rating" is expected to be a number data type, but it seems it has values that can be considered otherwise.

Question 3: Are there any missing values in the data? If yes, treat them using an appropriate method. [1 mark]

In [ ]:
```
# Using isnull() & sum() function.
data.isnull().sum()
```

Out[ ]:

	0
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: As observed above, no missing values exist in any of the columns. Hence, this dataset has all the values.

Question 4: Check the statistical summary of the data. What is the minimum, average, and maximum time it takes for food to be prepared once an order is placed? [2 marks]

In [ ]:
```
# Using the describe() function for statistical insight into the data.
data.describe().T.round(decimals=3)
```

Out[ ]:

	count	mean	std	min	25%	50%	75%	max
order_id	1898.0	1477495.500	548.050	1476547.00	1477021.25	1477495.50	1477969.750	1478444.00
customer_id	1898.0	171168.478	113698.140	1311.00	77787.75	128600.00	270525.000	405334.00
cost_of_the_order	1898.0	16.499	7.484	4.47	12.08	14.14	22.298	35.41
food_preparation_time	1898.0	27.372	4.632	20.00	23.00	27.00	31.000	35.00
delivery_time	1898.0	24.162	4.973	15.00	20.00	25.00	28.000	33.00

Observations:

- order_id & customer_id, as they are id columns, the stat data only provides a little insight except their min and max range.
- cost_of_the_order varies from $4.47 to $35.41 with mean=16.499, median=14.14, 25th percentile = 12.08 and 75th percentile = 22.29. Also, the standard deviation is 7.484.
- food_preparation_time varies from 20 min to 35.41 min with mean=27.37, median=27.00, 25th percentile = 23.00 and 75th percentile = 31.00. Also, the standard deviation is 4.623.
- delivery_time varies from 15.00 min to 33.00 min with mean=24.16, median=25.00, 25th percentile = 25.00 and 75th percentile = 28.00. Also, the standard deviation is 7.973.
- Also, as the food_preparation_time has a mean and median that are very close, it is expected that food preparation time is symmetrically distributed.
- Delivery time also can be said to have a symmetrical distribution tendency.
- It is also observed that the food preparation time is slightly higher than the delivery time in most cases.

Question 5: How many orders are not rated? [1 mark]

In [ ]:
```
# First, filter the data based on rating and count any column.
data[data['rating'] == 'Not given'].count()
```

Out[ ]:

	0
order_id	736
customer_id	736
restaurant_name	736
cuisine_type	736
cost_of_the_order	736
day_of_the_week	736
rating	736
food_preparation_time	736
delivery_time	736

dtype: int64

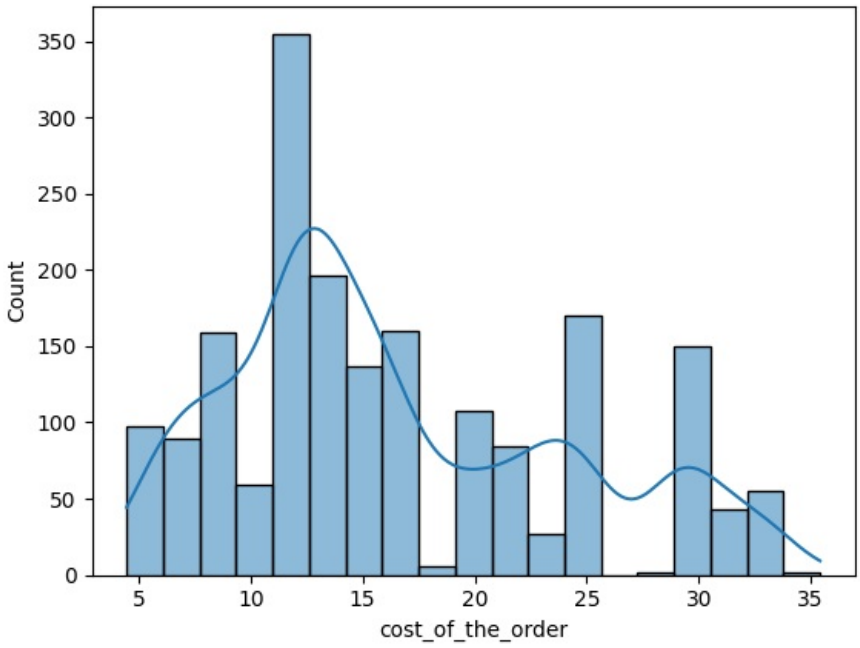
Observations: From the above data, we have 736 rows where the customer did not provide a rating.

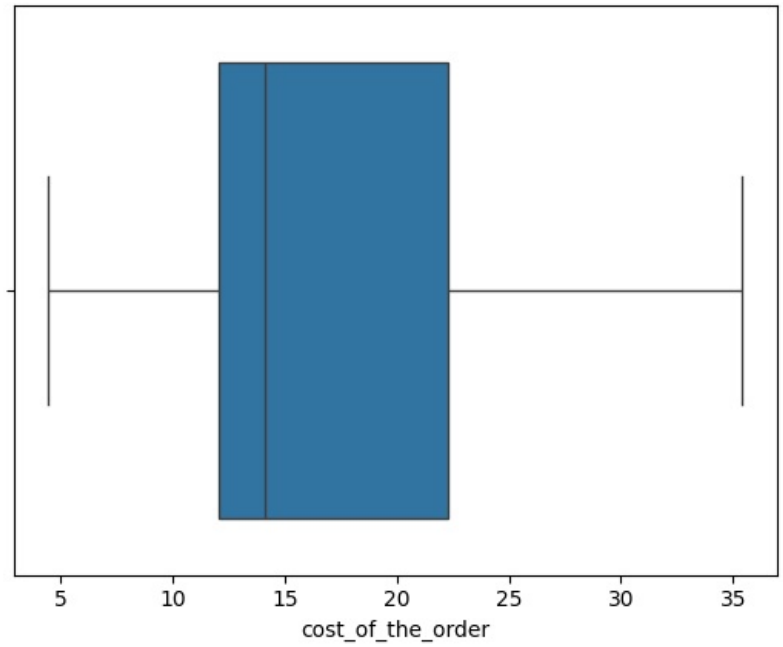
Exploratory Data Analysis (EDA)

Univariate Analysis

Question 6: Explore all the variables and provide observations on their distributions. (Generally, histograms, boxplots, countplots, etc. are used for univariate exploration.) [9 marks]

In [ ]:
```
# Visualize the cost_of_the_order data using a histogram and a boxplot.
sns.histplot(data=data,x='cost_of_the_order', kde=True)
plt.show()
sns.boxplot(data=data,x='cost_of_the_order')
plt.show()
```

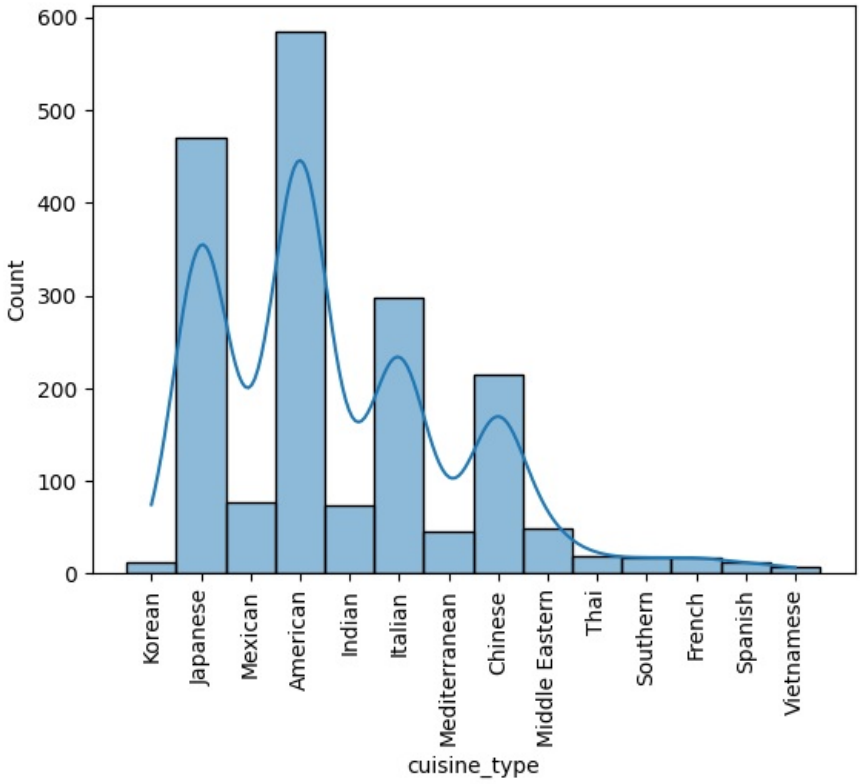


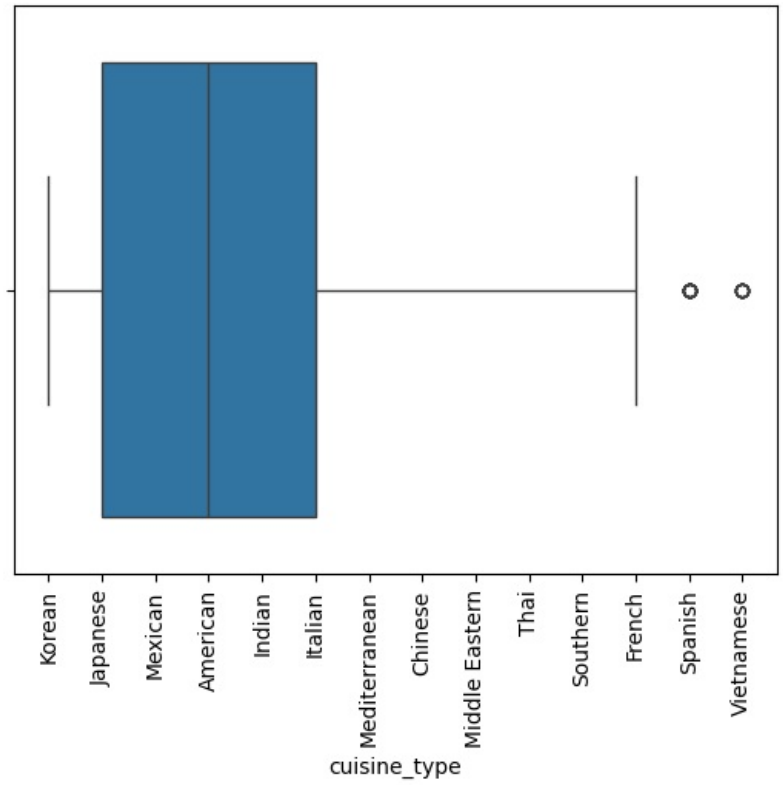


Observations: The cost of the order data is slightly right-skewed, with 14.14 as the median and 16.50 as the mean. There is no specific observation other than that the order cost can vary depending on many factors, such as cuisine type, amount of order, items ordered, etc.

In [ ]:
```
# Visualize order data based on cuisine type using histogram and box plot.
# Print the number of unique cuisine types
print("Number of unique cusine types:", data['cuisine_type'].nunique())
plt.xticks(rotation=90)
sns.histplot(data=data,x='cuisine_type', kde="True")
plt.show()
plt.xticks(rotation=90)
sns.boxplot(data=data,x='cuisine_type')
plt.show()
```

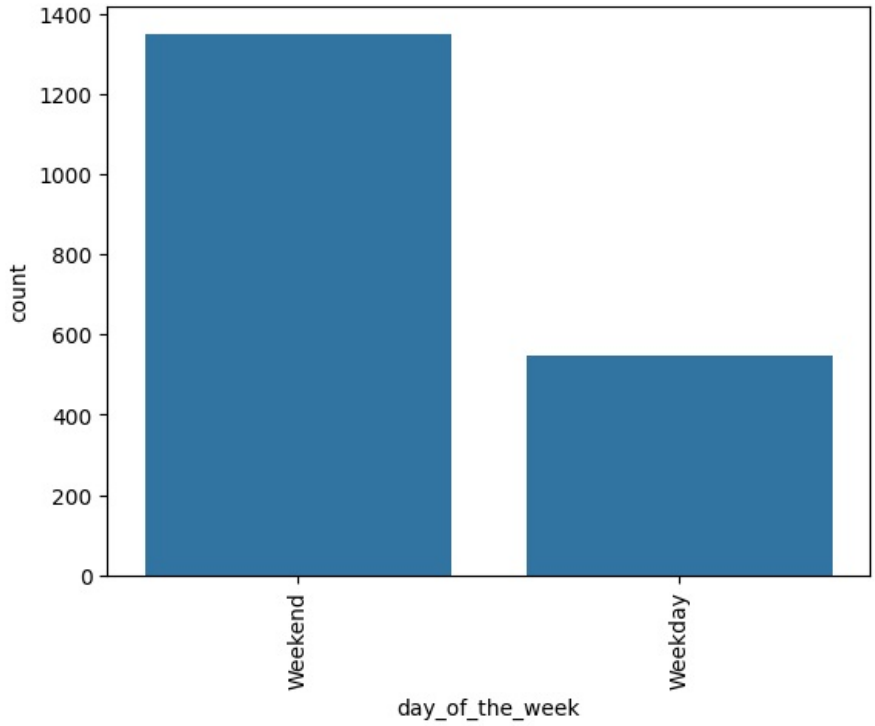
```
Number of unique cusine types: 14
```





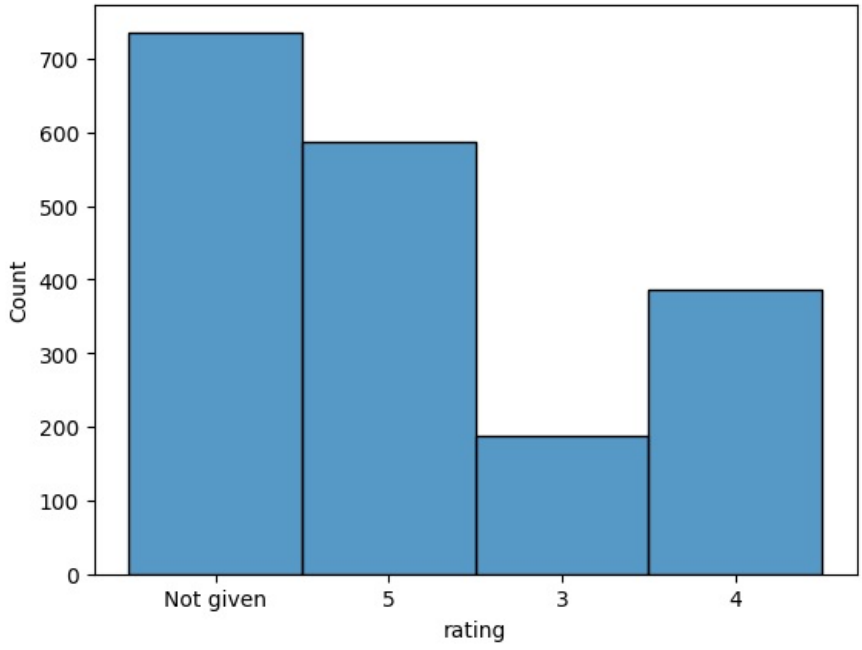
Observations: Order by cuisine type is a right-skewed distribution with few orders of Spanish and Vietnamese cuisines. The most ordered cuisine types are American and Japanese.

In [ ]:
```
# Visualize order data based on day of the week using count plot.
plt.xticks(rotation=90)
sns.countplot(data=data,x='day_of_the_week')
plt.show()
```



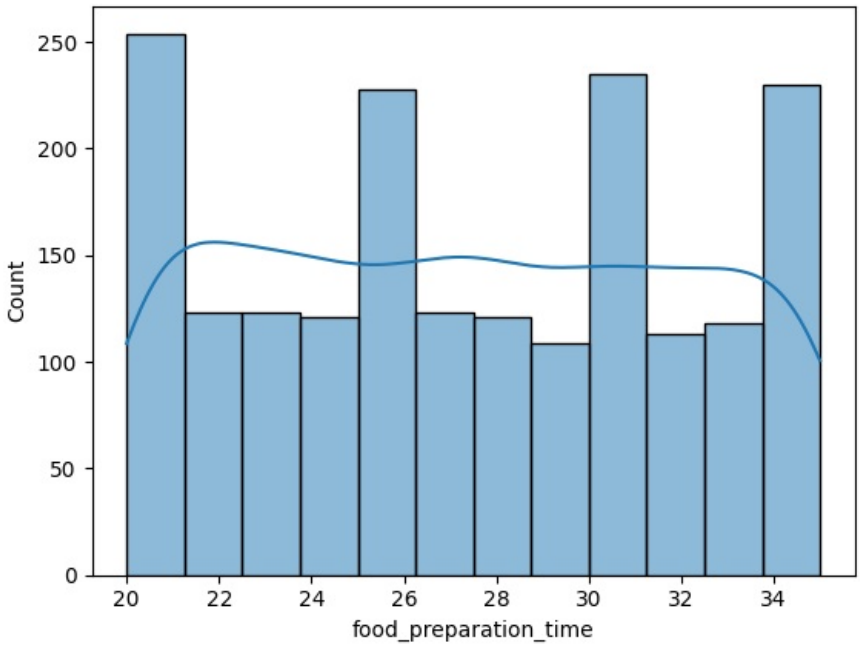
Observations: The above analysis shows that the number of orders on the weekends is much higher than on weekdays; in fact, it is slightly more than double.

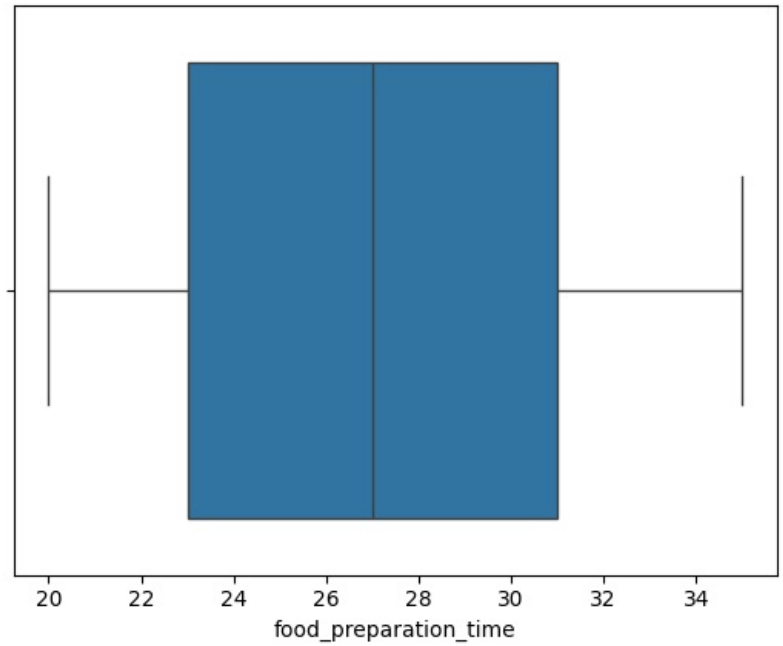
In [ ]:
```
# Visualize customer rating data.
sns.histplot(data=data,x='rating')
plt.show()
```



Observations: The above histogram shows that a significant number of orders are 'Not Rated'. Around 600 orders are rated 5, around 400 as 4, and around 175 as 3.

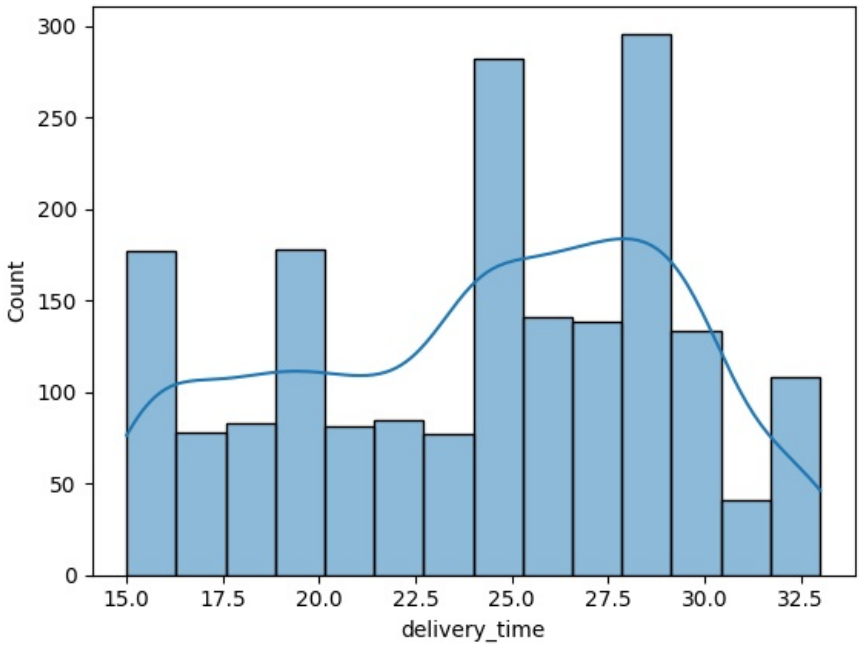
In [ ]:
```
# Visualize food preparation time using histogram and boxplot.
sns.histplot(data=data,x='food_preparation_time', kde=True)
plt.show()
sns.boxplot(data=data,x='food_preparation_time')
plt.show()
```

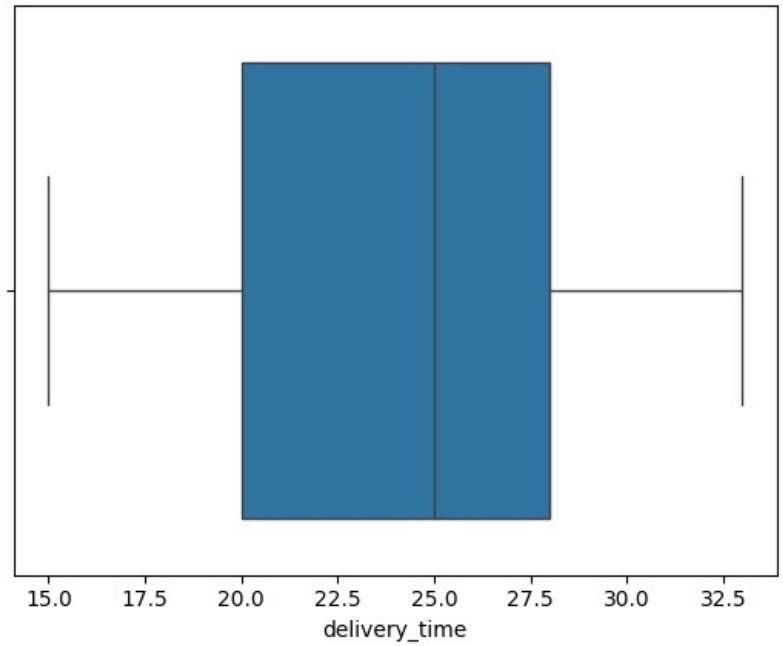




Observations: The food preparation time seems evenly distributed. Instead of a bell curve, we see a flat curve with few spikes. It also needs to be noted that food preparation time may also depend on many factors which are not detectable in single variant analysis.

In [ ]:
```
# Visualize delivery time using histogram and boxplot.
sns.histplot(data=data,x='delivery_time', kde=True)
plt.show()
sns.boxplot(data=data,x='delivery_time')
plt.show()
```

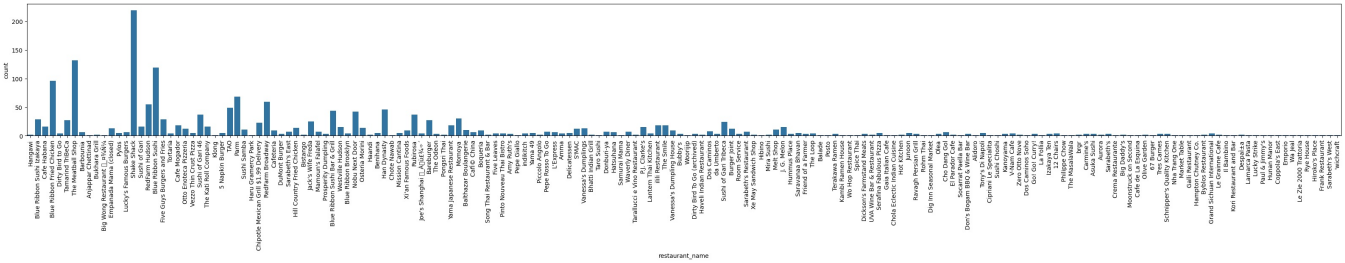




Observations: In the above plots, no specific trend is detectable. It appears that in most cases, the delivery time varies between 24 and 32 minutes. The mean and median are very close, which means a tendency toward a normal distribution. It also needs to be noted that delivery time may significantly vary between weekends and workdays depending on the time of day, rush hour, etc.

In [ ]:
```
# Finally Visualize number of orders restaurant name.

# Ignoring warnings, as some restaurant name contains special character's
import warnings
warnings.filterwarnings("ignore")

# Print number of unique restaurant names in the data set
print("Number of unique restarurant names:", data['restaurant_name'].nunique())
# Generate snd set specific plot size
plt.figure(figsize=(40, 4));
plt.xticks(rotation=90)
sns.countplot(data=data,x='restaurant_name');
plt.show()
```

```
Number of unique restarurant names: 178
```



Observations: From the above count plot, we can see that there are 178 total restaurants in the data set. Shake Shack is the most

popular restaurant, with more than 250 orders.

Excluded analysis of customer_id and order_id as they are purely artificial and id column

Question 7: Which are the top 5 restaurants in terms of the number of orders received? [1 mark]

In [ ]:
```
# Print some preliminary info - total number of orders and how many restaurants in the data set.
print("Total number of orders:", data['order_id'].count())
print("Total number of restaurents:", data['restaurant_name'].nunique())
# Using value_counts(), print the number of rows per unique restaurant name.
# value_count() function has sort=True by default hence it will sort the rows by count value in descending orde
data['restaurant_name'].value_counts()
```

```
Total number of orders: 1898
Total number of restaurents: 178
```

Out[ ]:

restaurant_name	count
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

178 rows × 1 columns

dtype: int64

Observations: From the above data, we can see that the top five restaurants that received the highest number of orders are Shake Shack-219, The Meatball Shop-132, Blue Ribbon Sushi-119, Blue Ribbon Fried Chicken-96, and Parm-68.

Question 8: Which is the most popular cuisine on weekends? [1 mark]

In [ ]:
```
# Print how many unique cousines exist in the data set.
print("Total number of cuisine type:", data['cuisine_type'].nunique())
# First filter the data set by column value 'day_of_the_week' == 'Weekend'.
# Then use value_counts(), to print the number of orders per unique 'cousine_type' using the filtered data for
# value_count() function has sort=True by default, hence it will sort the rows by count value in descending ord
data[data['day_of_the_week'] == 'Weekend']['cuisine_type'].value_counts()
```

```
Total number of cuisine type: 14
```

Out[ ]:

cuisine_type	count
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

dtype: int64

Observations: From the above analysis, we can see that American is the most popular cousine in the weekends.

Question 9: What percentage of the orders cost more than 20 dollars? [2 marks]

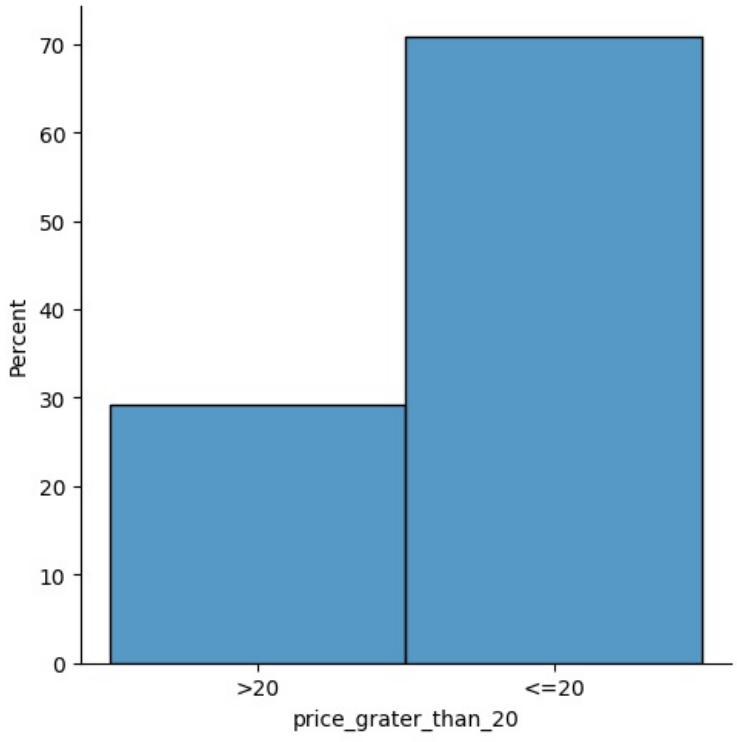
In [ ]:
```
# Frist count the number of orders when 'cost_of_the_order' > 20
# Second count total number of orders
# Devide first by second cont and multiply by 100 to get the percentage.
data[data['cost_of_the_order'] > 20]['order_id'].count() / data['order_id'].count() * 100
```

Out[ ]:
```
29.24130663856691
```

Observations: From the above calculation, only ***29.24%*** (rounded) of the orders cost is more than $20.00

In [ ]:
```
# Create a sub data set (DataFrame) for only order_id and cost_of_the_order
df = data[['order_id', 'cost_of_the_order']].copy()
# Add a new column price_grater_than_20 which will hold a boolean value of true or false
# Use the map method to assign True(>20) or False(<=20) in the new price_grater_than_20 column
df['price_grater_than_20'] = (data['cost_of_the_order'] > 20).map({False: '<=20', True: '>20'})
# Create a dist plot with stat='percent' to have 2 columns for each value.
sns.displot(data=df, x='price_grater_than_20', stat='percent');
plt.show()
```

```
29.24130663856691
```

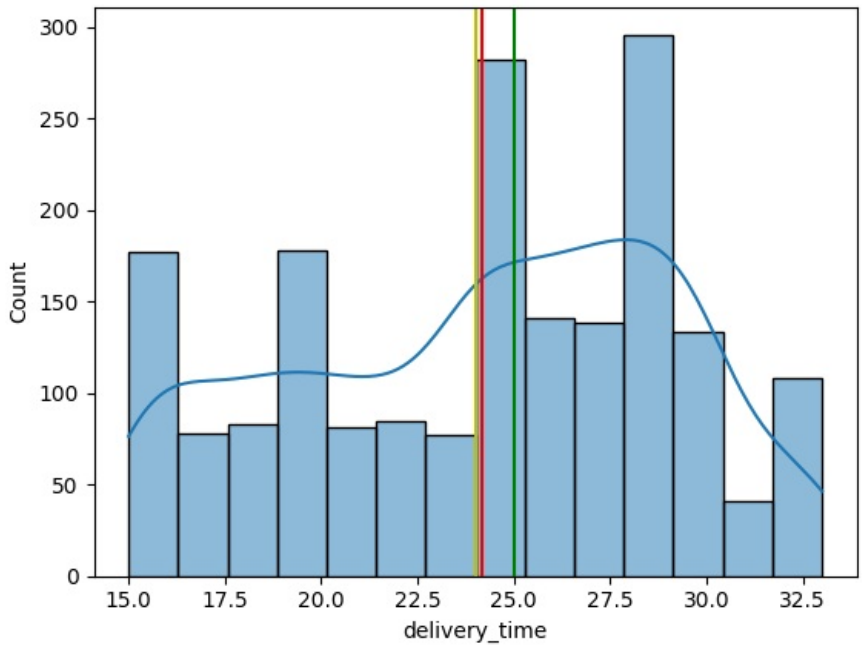


Observations: To confirm our calculation is correct, generated the above distribution plot using stat='percent'. From this plot, it confirms that it is likely that ***29.24%*** (rounded) of the orders cost is more than $20.00

Question 10: What is the mean order delivery time? [1 mark]

In [ ]:
```
# Display the mean value of the 'delivery_time' column using mean() function
print('Mean of column:delivery_time is',data['delivery_time'].mean())

# Added to visualize mean, median and mode - not part of original question

print ("\n\n\nAdded to visualize mean, median and mode - not part of original question \n\n")


# Define a figure with 1 X 1 subplot with an x-axis
fig, xaxis = plt.subplots(nrows=1, ncols=1,gridspec_kw=None)
# calculate mean, median & mode for 'delivery_time' column
mean=data['delivery_time'].mean()
median=data['delivery_time'].median()
mode=data['delivery_time'].mode().values[0]
# Create a histplot (histogram) for the 'delivery_time' column
sns.histplot(data=data, x="delivery_time", ax=xaxis, kde=True)
# using the pre defined x-axis object create 3 vertical line for mean(red), median (green) and mode (yellow) an
xaxis.axvline(mean, color='r', label="Mean")
xaxis.axvline(median, color='g', label="Median")
xaxis.axvline(mode, color='y', label="Mode")
# Display the plot
plt.show()
```

```
Mean of column:delivery_time is 24.161749209694417


Added to visualize mean, median and mode - not part of original question
```



Observations: From the above canculation we can see that mean for the delivery time is ***24.16***

Question 11: The company has decided 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. [1 mark]

In [ ]:
```
# We need their total orders in the data set to identify the most frequent customers.
# Doing a group by 'customer_id' and a count for 'order_id' in the dataset will provide the answer. Print the l
data.groupby(['customer_id'])['order_id'].count().sort_values(ascending=False)
```

Out[ ]:

customer_id	order_id
52832	13
47440	10
83287	9
250494	8
65009	7
...	...
105903	1
105992	1
106006	1
106324	1
405334	1

1200 rows × 1 columns

dtype: int64

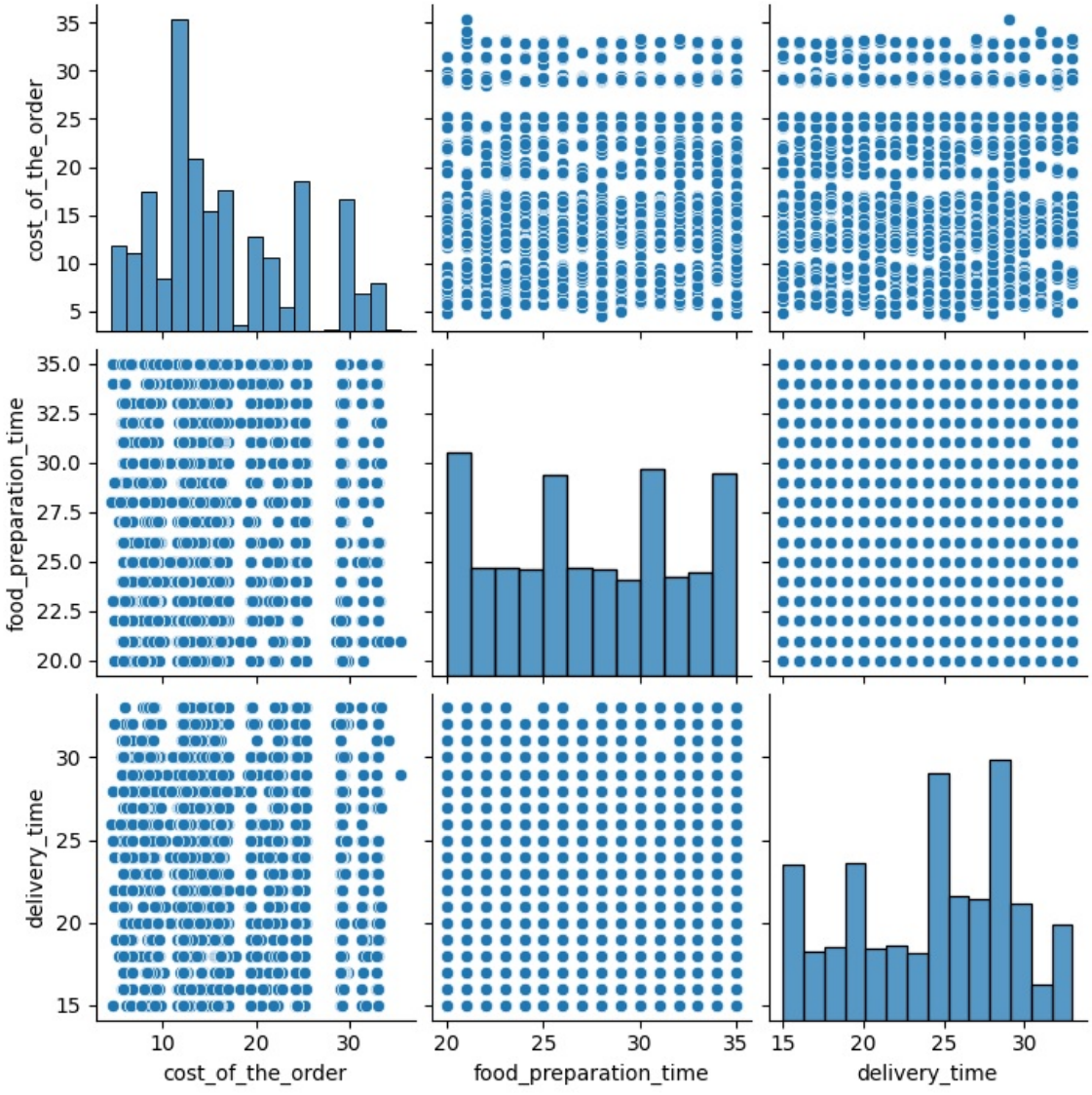
Observations: From the above table (sorted by descending order of order_count), we can find the top 3 customer_id and the total number of orders they placed.

1. Customer Id: 52832, number of orders: 13
2. Customer Id: 47440, number of orders: 10
3. Customer Id: 83287, number of orders: 9

Question 12: Perform a multivariate analysis to explore relationships between the important variables in the dataset. (It is a good idea to explore relations between numerical variables as well as relations between numerical and categorical variables) [10 marks]

In [ ]:
```
# First, we must do pairwise relationships between continuous/numerical variables within the data set
# cost_of_the_order Vs. food_preparation_time Vs. delivery_time
sns.pairplot(data=data[['cost_of_the_order', 'food_preparation_time', 'delivery_time']])
```

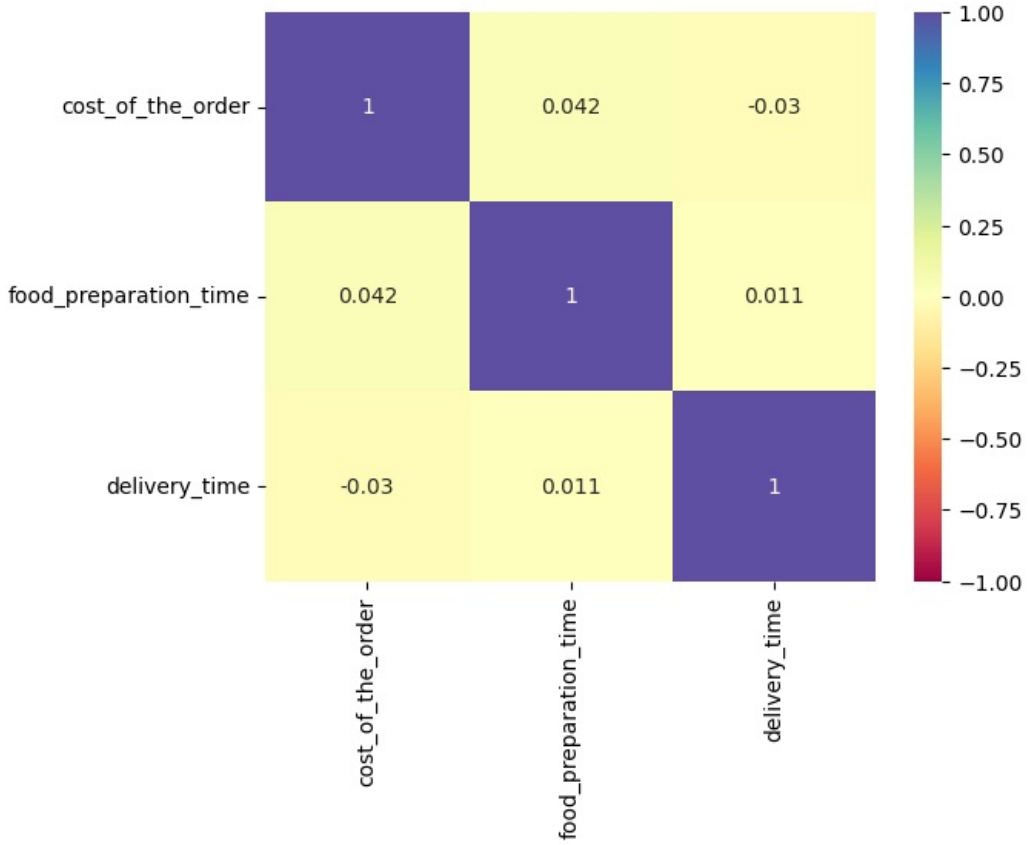
Out[ ]:
```
<seaborn.axisgrid.PairGrid at 0x7bc3e75d65f0>
```



Observations: From this plot, we do not see any strong relationships between the 3 numerical variables ['cost_of_the_order',

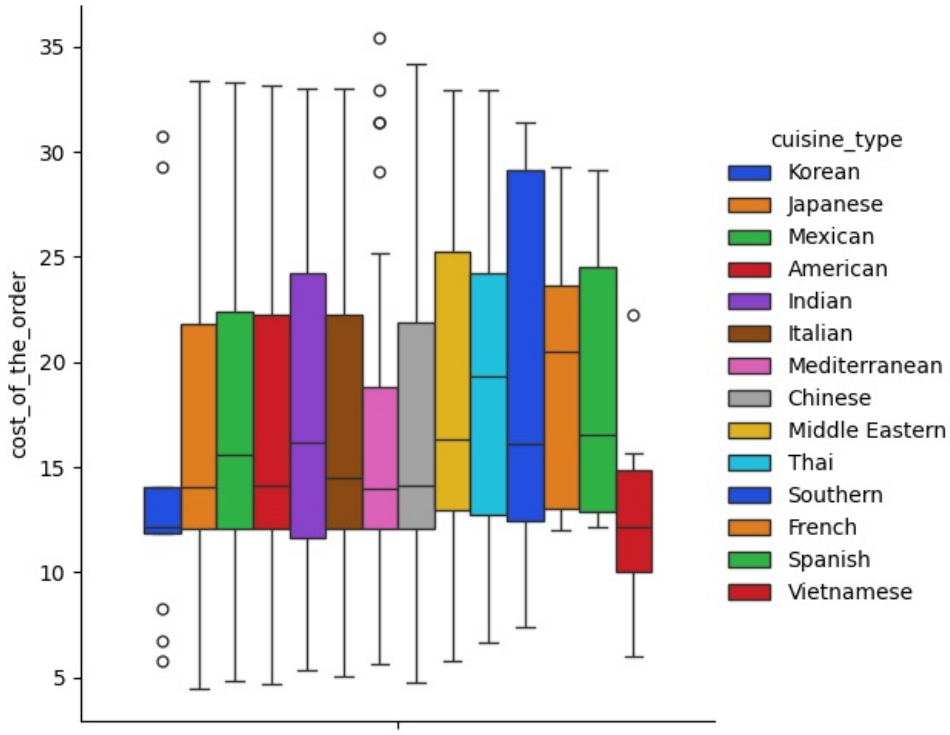
Observations: From this plot, we do not see any strong relationships between the 3 numerical variables ['cost_of_the_order', 'food_preparation_time,' 'delivery_time']. Thus, it appears that these variables may not be directly connected.

In [ ]:
```
# To confirm, we should create a heatmap for the same numerical variables and check their correlation
# cost_of_the_order Vs. food_preparation_time Vs. delivery_time
sns.heatmap(data=data[['cost_of_the_order', 'food_preparation_time', 'delivery_time']].corr(), annot=True,cmap=
plt.show()
```



Observations: The heat map confirms that there is no strong correlation between cost, preparation time, and delivery time. Even if I expected that the order cost may increase food preparation time, the correlation is negligible and has no impact.

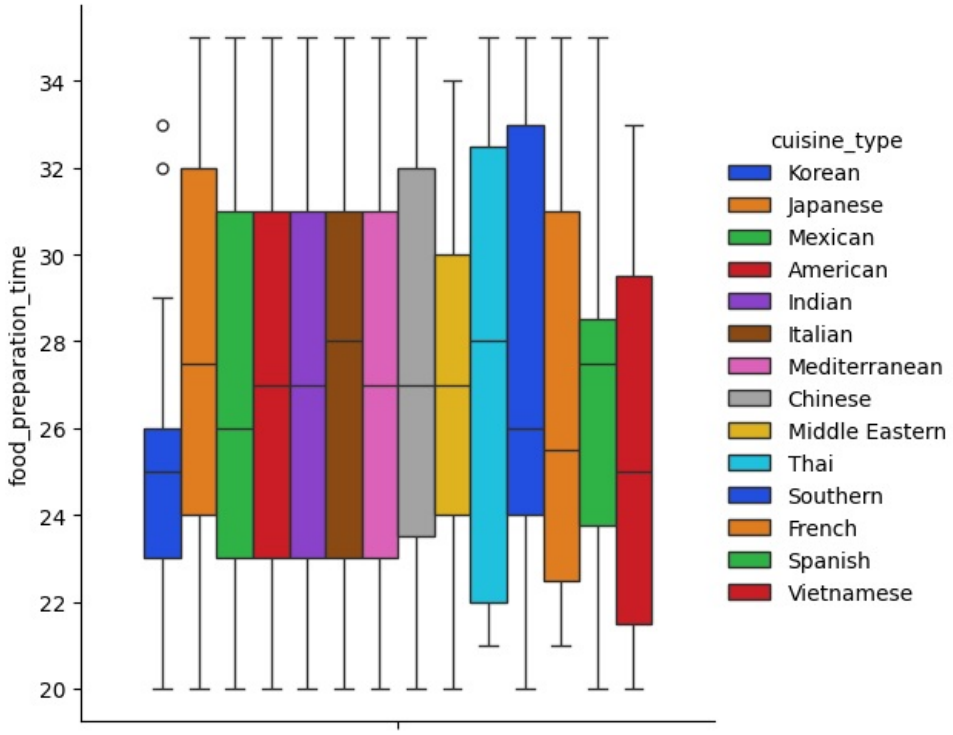
In [ ]:
```
# Next, we should check if the cost of the order has anything to do with the cuisine type.
# Using a catplot, we can generate a boxplot for each cuisine type for the price.
sns.catplot(data=data, y='cost_of_the_order', hue="cuisine_type", kind='box', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



Observations:

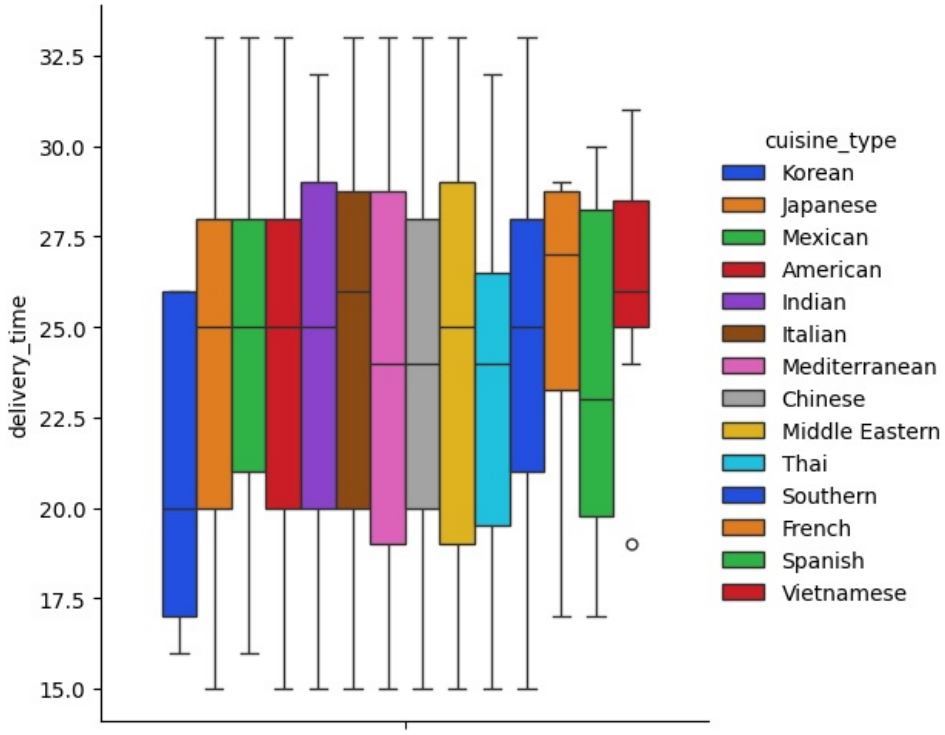
- Korean food has the slightest variation in cost except for some outliers on both sides.
- The Mediterranean cuisine type has some outliers on the higher side.
- Chinese cuisine has a price variation from very low to very high.
- Vietnamese cuisine never exceeded the price of $25.00
- Costs for all other cuisines vary almost equally.

In [ ]:
```
# Next, we should check if the food preparation time is related to the cuisine type.
# Using a catplot, we can generate a boxplot for each cuisine type.
sns.catplot(data=data, y='food_preparation_time', hue="cuisine_type", kind='box', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



Observations: This plot shows that food preparation time is equally distributed among cuisine types with slight variations, like it seems Korean food preparation time, on average, is lower with some outliers.

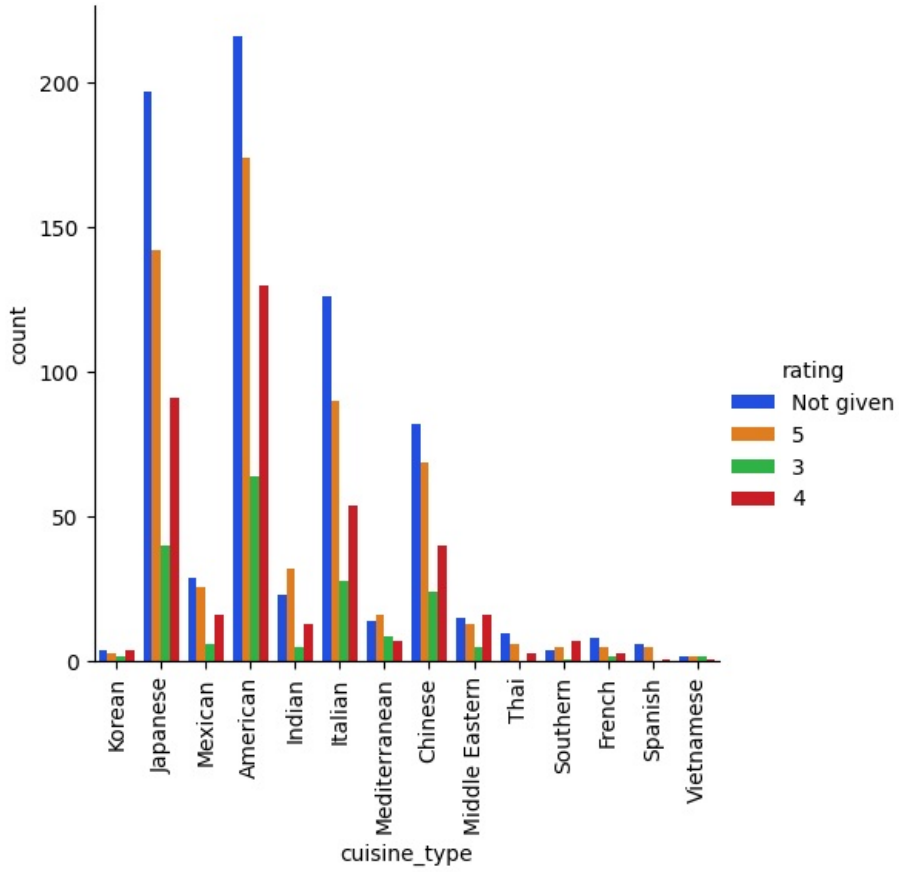
In [ ]:
```
# Is delivery time have any any impact based on cuisine type?
sns.catplot(data=data, y='delivery_time', hue="cuisine_type", kind='box', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



Observations: This represents that delivery time also has no significant impact by cuisine type either.

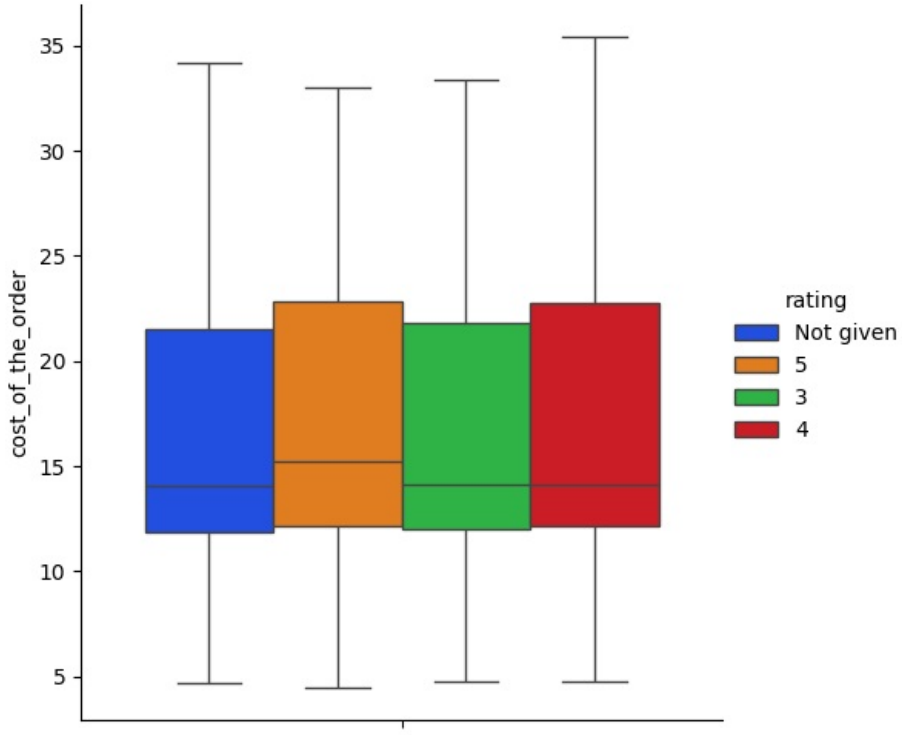
In [ ]:
```
# Checking rating by cuisine type using count plot
```

In [ ]:
```
# Checking rating by cuisine type using count plot
sns.catplot(data=data, x='cuisine_type', hue="rating", kind='count', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



Observations: No specific trend is visible here either. The ratings follow the same trend as the number of orders for that particular cuisine type. In most cases, customers have not left a rating for all types of cuisines.

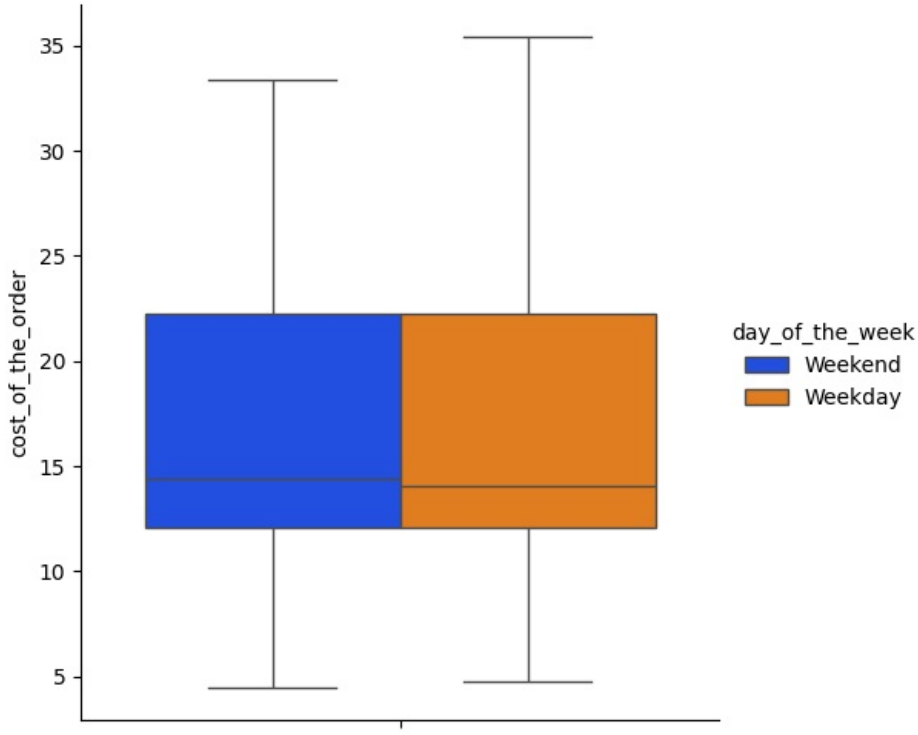
In [ ]:
```
# Checking if the cost of an order has anything to do with the customer's rating
sns.catplot(data=data, y='cost_of_the_order', hue="rating", kind='box', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



Observations: It confirms that customers' ratings about their orders have no relationship with the cost of the order. As all ratings and no-rating are almost equally distributed irrespective of the cost.

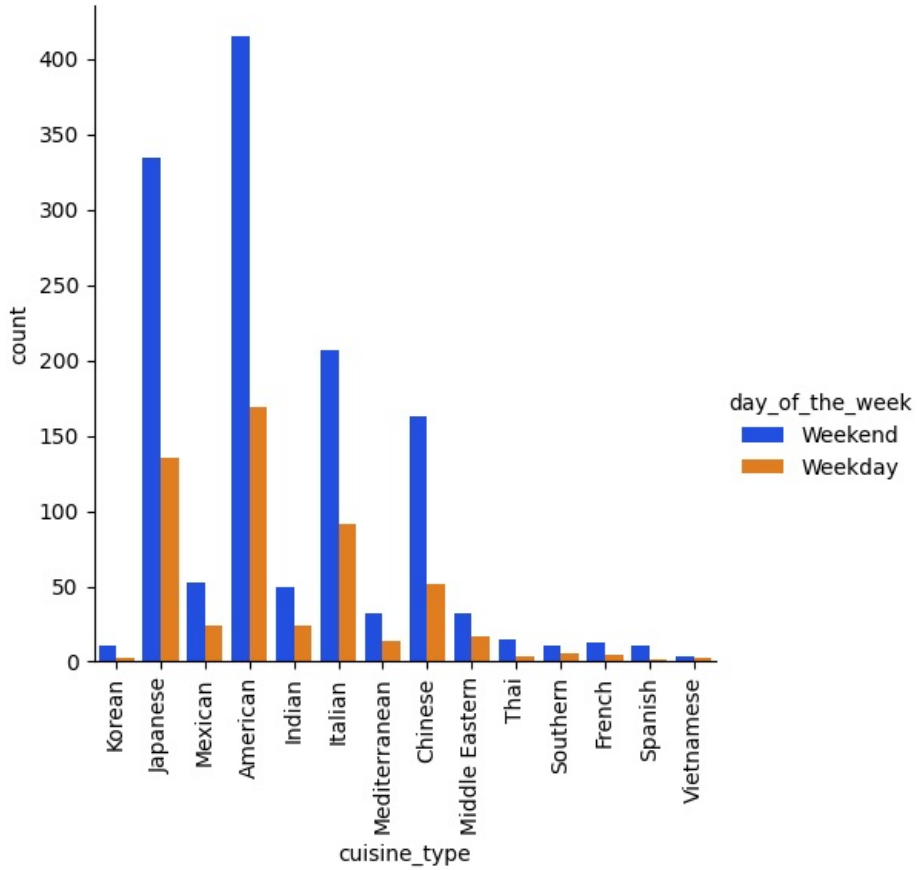
In [ ]:
```
# Does cost vary based on the day of the week
sns.catplot(data=data, y='cost_of_the_order', hue="day_of_the_week", kind='box', palette='bright');
plt.xticks(rotation=90)
```

```
plt.show()
```



Observations: Cost is not affected by the weekdays vs weekends except in some negligible cases

In [ ]:
```
# Is the number of orders vary based on cuisine type and day of the week
sns.catplot(data=data, x='cuisine_type', hue="day_of_the_week", kind='count', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



Observations: This graph definitely shows that the number of orders on weekends is higher than on weekdays for every cuisine type. However, the difference is not very significant based on cuisine type between weekends and weekdays.

In [ ]:
```
# Which restaurant(s) receives most of the order?
data.groupby(['restaurant_name'])['order_id'].count().sort_values(ascending=False)
```

Out[ ]:

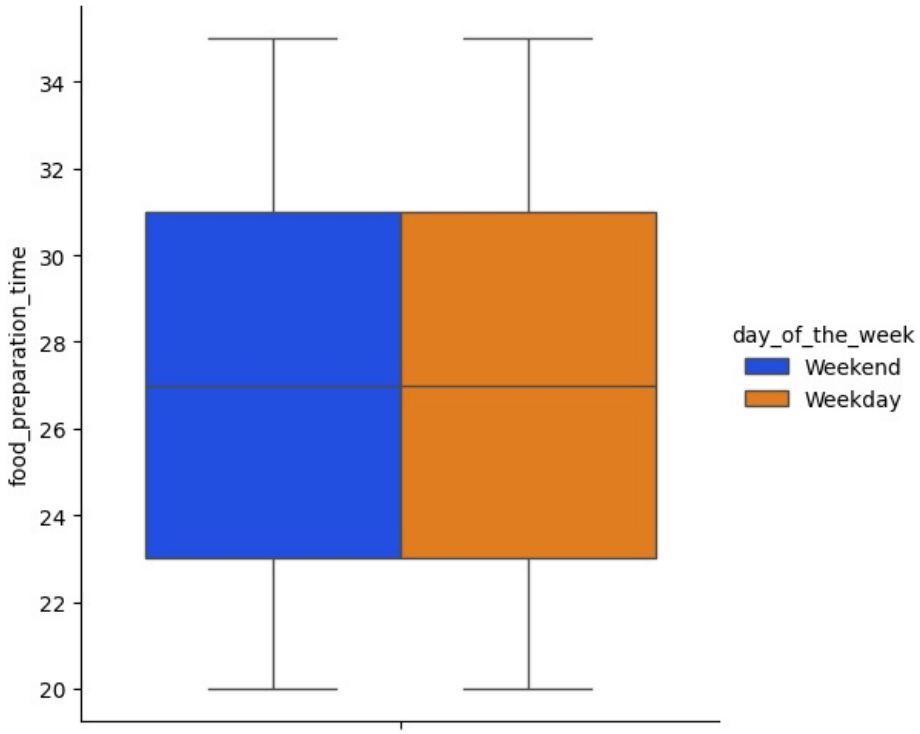
restaurant_name	order_id
Shake Shack	219
The Meatball Shop	132
Blue Ribbon Sushi	119
Blue Ribbon Fried Chicken	96
Parm	68
...	...
Klong	1
Kambi Ramen House	1
Il Bambino	1
Hunan Manor	1
Lamarca Pasta	1

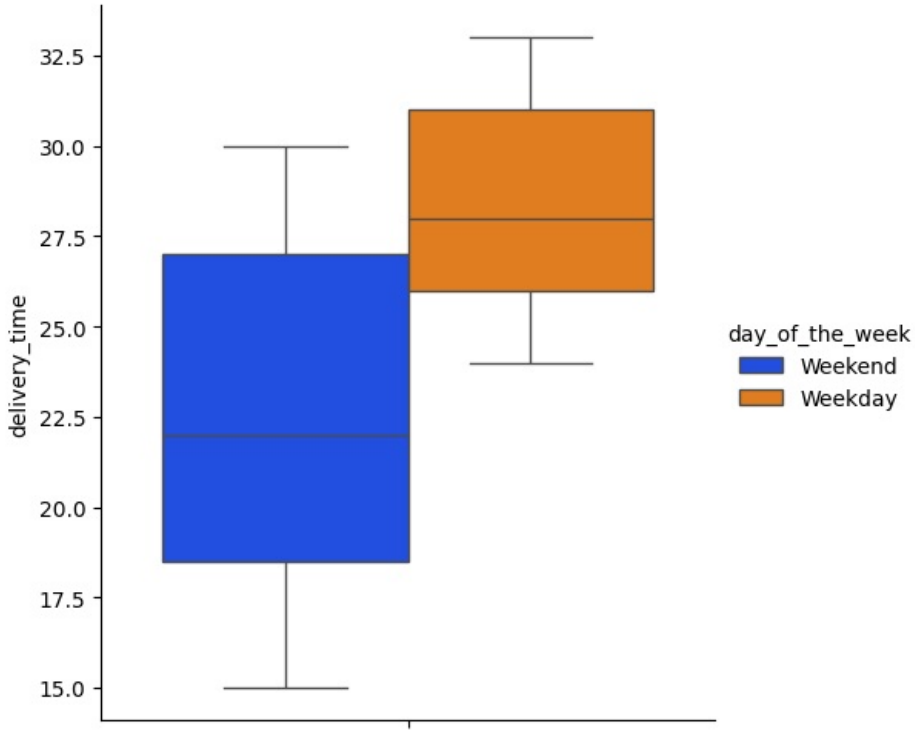
178 rows × 1 columns

dtype: int64

Observations: The above table shows the most popular restaurants by their number of orders.

In [ ]:
```
# Check the variations in food preparation and delivery time on weekdays vs. weekends.
sns.catplot(data=data, y='food_preparation_time', hue="day_of_the_week", kind='box', palette='bright');
plt.xticks(rotation=90)
plt.show()
sns.catplot(data=data, y='delivery_time', hue="day_of_the_week", kind='box', palette='bright');
plt.xticks(rotation=90)
plt.show()
```



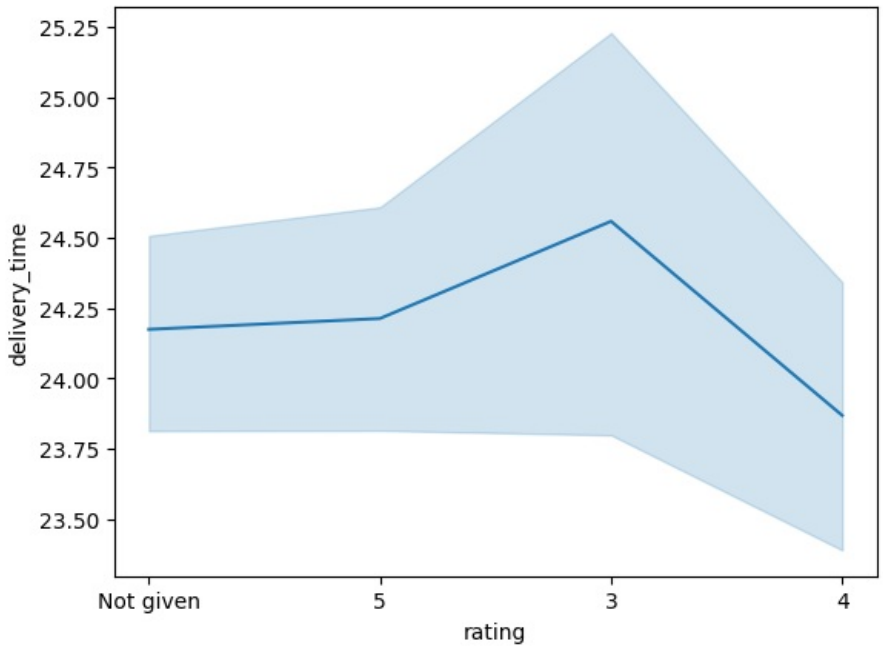


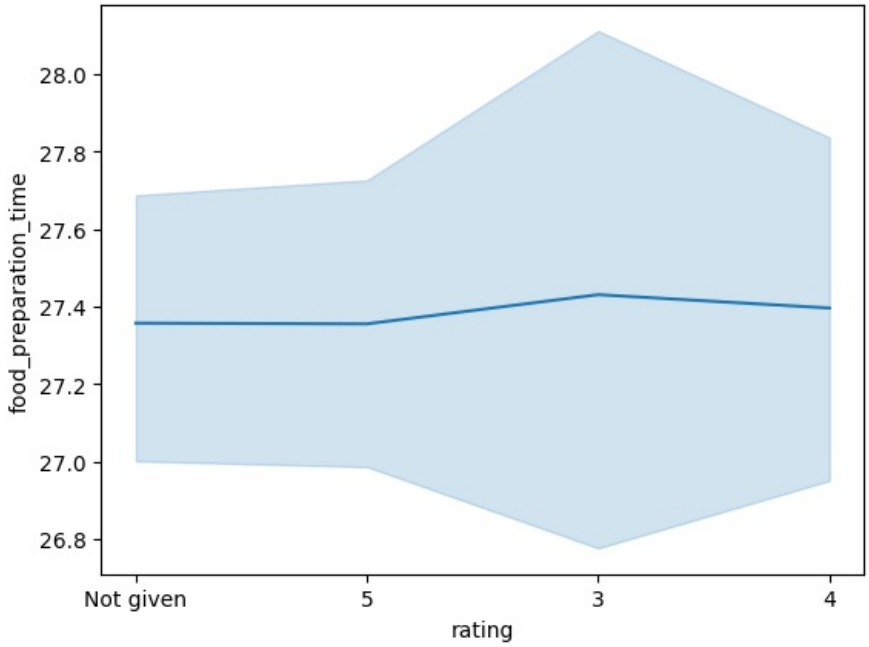
Observations:

- Definitely, food preparation time has no impact between weekdays and weekends
- Delivery time has a significant impact on weekdays compared to weekends. This might result from traffic, rush hour delays, and/or unavailability of delivery persons on weekdays.

To improve delivery time, the business should consider hiring more staff, providing the team with the necessary support to manage the workload effectively.

In [14]:
```
# Check if delivery time and food preparation time has any impact on rating using a line plot
sns.lineplot(data=data, y='delivery_time', x="rating");
plt.show()
sns.lineplot(data=data, y='food_preparation_time', x="rating");
plt.show()
```





Observations: It appears that both delivery time and food preparation time impact rating. The rating drops if delivery time is high, and the same is true for food preparation time.

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 [ ]:
```
# To identify the restaurants eligible for the promotional offer
# First, we need to remove missing / Not rated data from the dataset

# Create a secondary copy of the original dataset, removing the rows with rating as 'Not provided'.
data_with_no_rating = data[data['rating'] != 'Not given'].copy()

# Convert the 'rating' column from object to int64.
data_with_no_rating['rating'] = data_with_no_rating['rating'].astype(int)

# Filter out the data for restaurants with a rating count greater than 50 and put those data in a new data fram
data_filtered = data_with_no_rating.groupby(['restaurant_name']).filter(lambda x: x['rating'].count() > 50)

# Filter out the data for restaurants with average/mean rating count grater than 4.0. Put those data in a new d
data_filtered2 = data_filtered.groupby(['restaurant_name']).filter(lambda x: x['rating'].mean() > 4.0)

# Finally, print the data group by 'restaurant_name' along with the mean rating in descending order.
data_filtered2.groupby(['restaurant_name'])['rating'].mean().sort_values(ascending=False)
```

Out[ ]:

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

dtype: float64

Observations: From the above calculations and the result table, we can identify the restaurants that qualify the criteria to receive a promotional offer of the advertisement.

Question 14: 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. [3 marks]

In [ ]:
```
# Calculate total revenue in two steps for ease of understanding.
# First, calculate the revenue for all orders where the cost is greater than $20.00, and charge is 25%.
rev_above_20 = data[data['cost_of_the_order'] > 20.00]['cost_of_the_order'].sum() * 0.25

# Then calculate the revenue for all the orders between $5.00 and $20.00, and the charge is 15%.
rev_between_5_and_20 = data[(data['cost_of_the_order'] > 5.00) & (data['cost_of_the_order'] <= 20.00)]['cost_of

# Print the total revenue by adding both > 20.00 and between 5 & 20 revenue.
print('Total revenue=', (rev_above_20 + rev_between_5_and_20))
```

```
Total revenue= 6166.303
```

Observations: From the above calculation, the total revenue from the given data set will be ***$6166.30***

Question 15: 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.) [2 marks]

In [ ]:
```
# For this, we need to create a new column in the data set called 'total_time,' which is the sum of food_prepar
data['total_time'] = data['food_preparation_time'] + data['delivery_time']

#data.head()

# Now, we can calculate the number of rows where total_time > 60.
# We can divide the number of rows where total_time > 60 by the total number of rows to get the percentage of o
((data[data['total_time'] > 60]['total_time'].count() / data['total_time'].count())  * 100).round(decimals=2)
```

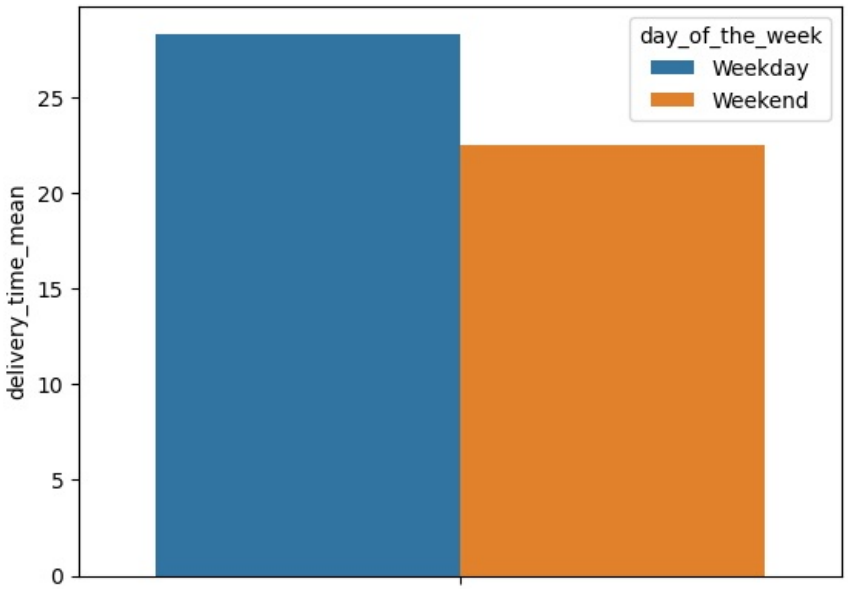
Out[ ]:
```
10.54
```

Observations: From the above calculation, we can say that at least ***10.54%*** of the orders took more than 60 minutes to complete from order to delivery.

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? [2 marks]

In [ ]:
```
# Calculate and print the mean delivery time on weekdays and weekends.
print('Mean of delivery_time in Weekday\'s',data[data['day_of_the_week'] == 'Weekday']['delivery_time'].mean().
print('Mean of delivery_time in Weekend\'s',data[data['day_of_the_week'] == 'Weekend']['delivery_time'].mean().

# To plot the variation in mean, we can create a secondary dataset - data_delivery_time_mean from the original
data_delivery_time_mean = data.groupby(['day_of_the_week'])['delivery_time'].mean(numeric_only=True).reset_inde
data_delivery_time_mean.rename(columns={'delivery_time': 'delivery_time_mean'}, inplace=True)

# We can visualize the difference in the mean by creating a barplot for two mean values for Weekdays and Weeken
sns.barplot(data=data_delivery_time_mean, y='delivery_time_mean', hue='day_of_the_week')
plt.show()
```

```
Mean of delivery_time in Weekday's 28.34
Mean of delivery_time in Weekend's 22.47
```



Observations: From the above calculations and the plot, we can clearly say that the average delivery time on weekdays is higher than on weekends. This is important information: the mean delivery time is around 6 minutes higher on weekdays than on weekends, in spite of the fact that the number of orders is higher on weekends.

Conclusion and Recommendations

Question 17: What are your conclusions from the analysis? What recommendations would you like to share to help improve the business? (You can use cuisine type and feedback ratings to drive your business recommendations.) [6 marks]

Conclusions: From this analysis, we can summarize below points

- Specific cuisine types are most popular both on weekdays and weekends. Example: American, Japanese, Italian & Chinese.
- Some restaurant receives the bulk of the orders. Examples: Shake Shack, The Meatball Shop, Blue Ribbon Sushi, Blue Ribbon Fried Chicken, Parm, etc.
- Most number of orders are received in the weekends.
- Delivery time is significantly higher on weekdays compared to weekends.
- The rating has an effect if food preparation and delivery time are high.

Recommendations:

- The business should consider the most popular cuisine types and restaurants and keep dedicated resources to serve the demand during peak hours.
- To improve delivery time, on weekdays, the business should consider hiring more staff, providing the team with the necessary support to manage the workload effectively (optimal and planned pickup and delivery).
- The business should look at the relationship between rating and total time (food preparation time + delivery time). For some restaurants, this time may impact the number of orders and ratings. Taking steps to improve the total time from order placement to delivery may increase the number of orders and improve ratings.