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.
!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]:
```
# Mounting Google Drive to Colab
from google.colab import drive
drive.mount('/content/drive')
```

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

In [41]:
```
# 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

# import warnings to remove FutureWarnings messages
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# to restrict the float value to 3 decimal places
pd.set_option('display.float_format', lambda x: '%.3f' % x)
```

Understanding the structure of the data

In [42]:
```
# Write your code here to read the data
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/foodhub_order.csv')
```

In [43]:
```
# Write your code here to view the first 5 rows
df.head()
```

Out[43]:

	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	30.750	Weekend	Not given	25	20
1	1477685	358141	Blue Ribbon Sushi Izakaya	Japanese	12.080	Weekend	Not given	25	23
2	1477070	66393	Cafe Habana	Mexican	12.230	Weekday	5	23	28
3	1477334	106968	Blue Ribbon Fried Chicken	American	29.200	Weekend	3	25	15
4	1478249	76942	Dirty Bird to Go	American	11.590	Weekday	4	25	24

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

In [44]:
```
# Write your code here

# Code to display the number of (columns,rows) in the data.
df.shape
```

Out[44]:
```
(1898, 9)
```

Observations: There are 1898 rows and 9 columns present in the data.

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

In [45]:
```
# Write your code here

# Code to display the information of the data.
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 data types of different columns in the dataset are int64, object, and float64. It doesn't seem like there are any missing values in the data as well as there are 1898 rows and 1898 non-null counts in each of the columns listed.

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

In [46]:
```
# Write your code here

# Code to check if there are any missing values in the data.
df.isnull().sum()
```

Out[46]:
```
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: Using the code provided above, we can observe that there are no missing values in the data.

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 [47]:
```
# Write your code here

# Code to check the statistical summary of the data.
df.describe()
```

Out[47]:

	order_id	customer_id	cost_of_the_order	food_preparation_time	delivery_time
count	1898.000	1898.000	1898.000	1898.000	1898.000
mean	1477495.500	171168.478	16.499	27.372	24.162
std	548.050	113698.140	7.484	4.632	4.973
min	1476547.000	1311.000	4.470	20.000	15.000
25%	1477021.250	77787.750	12.080	23.000	20.000
50%	1477495.500	128600.000	14.140	27.000	25.000
75%	1477969.750	270525.000	22.297	31.000	28.000
max	1478444.000	405334.000	35.410	35.000	33.000

Observations: Using the describe function we can see that the minimum, average, and maximum time it takes for the food to prepared once it is ordered is:

- Minimum: 20
- Average (Mean): 1898
- Maximum: 35

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

In [48]:
```
# Write the code here

# Code to check the rows of the customers who did not provide a rating.
df.loc[df['rating'] == "Not given"].value_counts()
```

Out[48]:
```
order_id  customer_id  restaurant_name        cuisine_type   cost_of_the_order  day_of_the_week  rating     foo
d_preparation_time  delivery_time
1476551   49034        The Smile              American       12.180             Weekend          Not given  22
27                 1
1477772   91958        TAO                    Japanese       12.180             Weekday          Not given  26
33                 1
1477753   65306        Sushi of Gari Tribeca  Japanese       14.790             Weekend          Not given  32
24                 1
1477756   251607       Shake Shack            American       14.120             Weekday          Not given  31
28                 1
1477757   60688        Shake Shack            American       14.120             Weekend          Not given  29
30                 1

..
1477128   354016       Waverly Diner          American       14.940             Weekend          Not given  28
28                 1
1477129   52832        Han Dynasty            Chinese        19.300             Weekend          Not given  34
21                 1
1477133   175290       Shake Shack            American       12.180             Weekend          Not given  26
25                 1
1477135   62359        Pylos                  Mediterranean  19.400             Weekend          Not given  28
29                 1
1478441   228541       RedFarm Hudson         Chinese        29.100             Weekend          Not given  27
28                 1
Name: count, Length: 736, dtype: int64
```

In [49]:
```
df['rating'].describe()
```

Out[49]:
```
count          1898
unique            4
top       Not given
freq            736
Name: rating, dtype: object
```

Observations: There are 736 out of 1898 orders that have been rated.

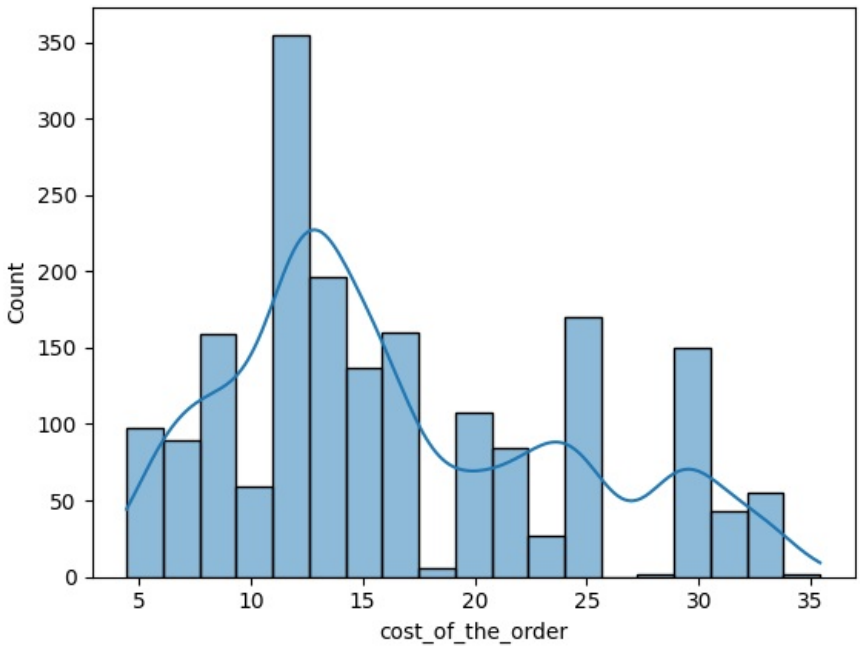
Exploratory Data Analysis (EDA)

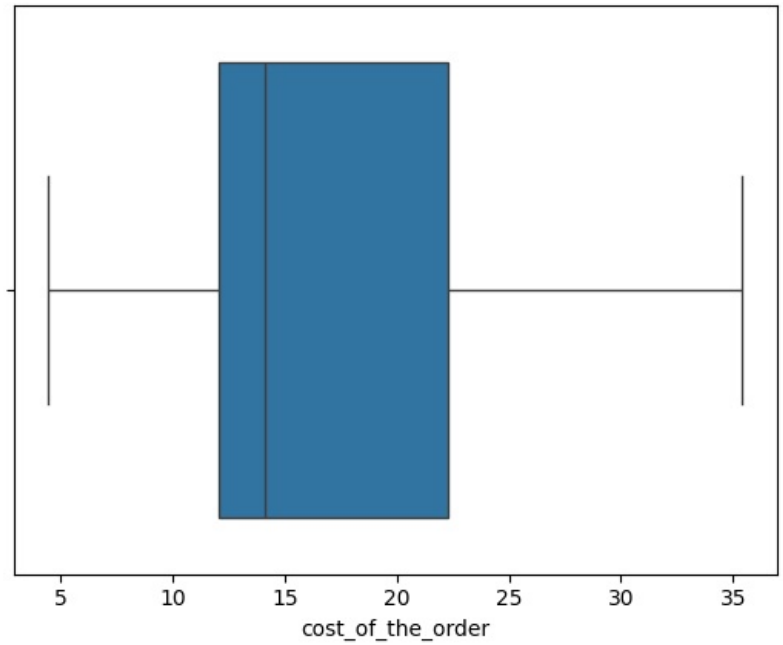
Univariate Analysis

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

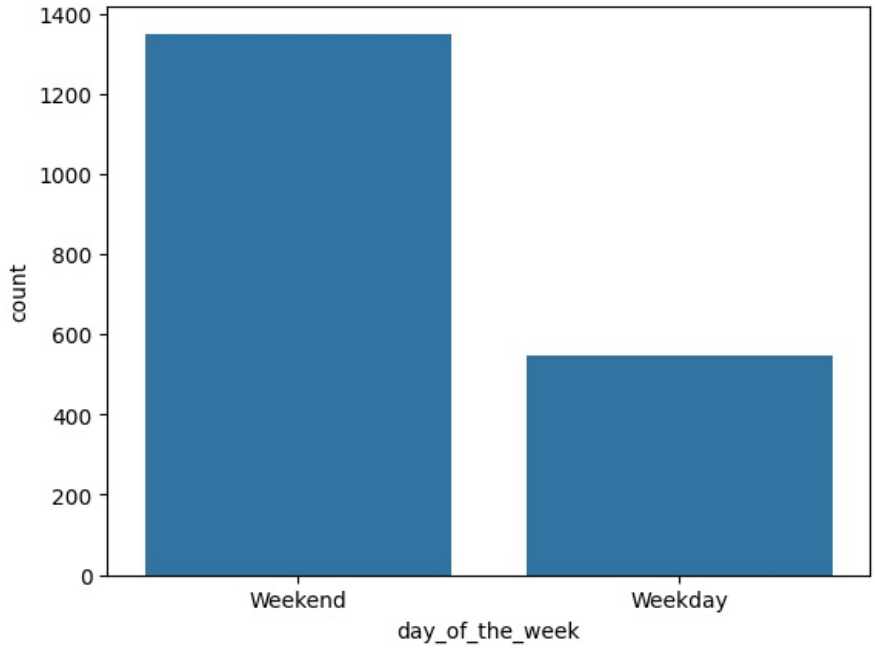
In [50]:
```
# Write the code here

# histogram and boxplot for 'cost_of_the_order'
sns.histplot(data=df, x='cost_of_the_order', kde=True)
plt.show()
sns.boxplot(data=df, x='cost_of_the_order')
plt.show()
```

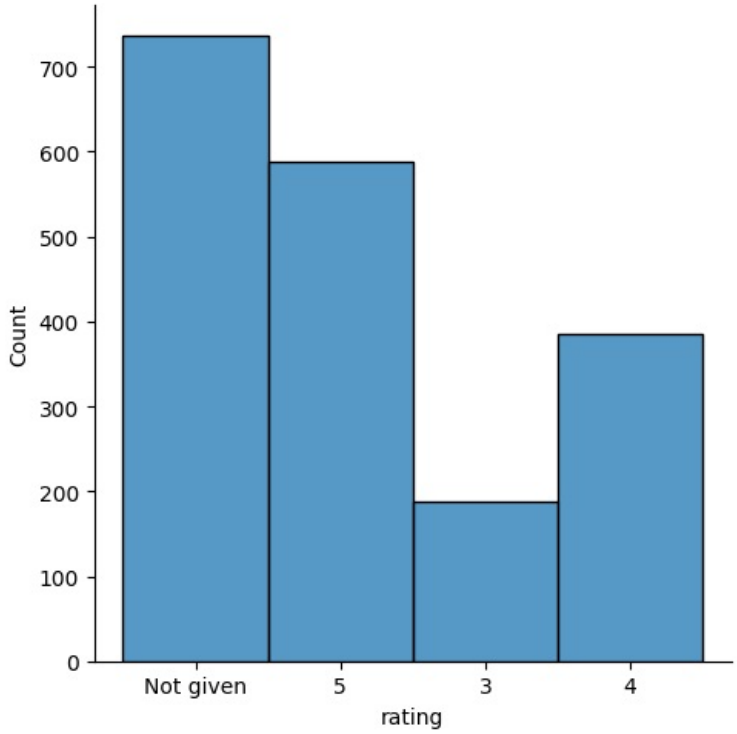


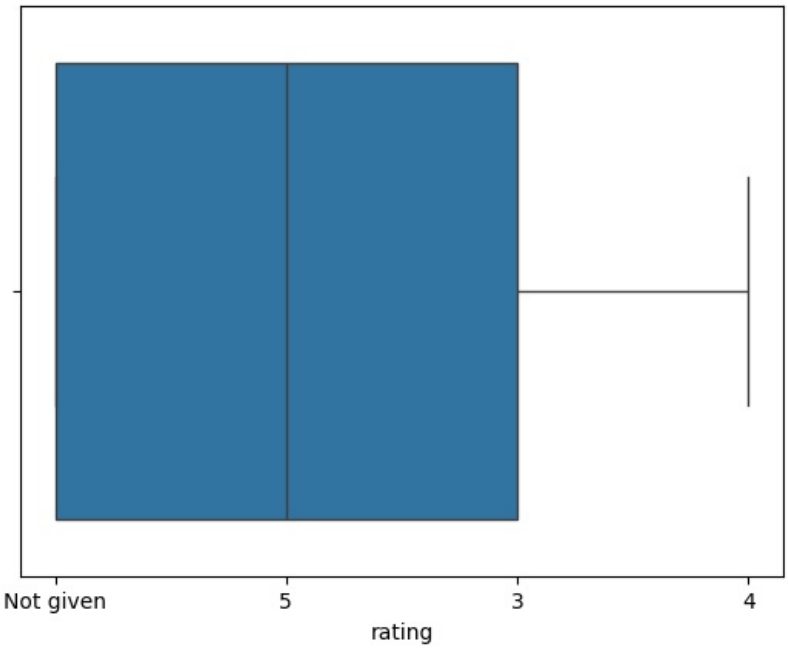


In [51]:
```
# count plot for 'day_of_the_week'
sns.countplot(data=df, x='day_of_the_week')
plt.show()
```

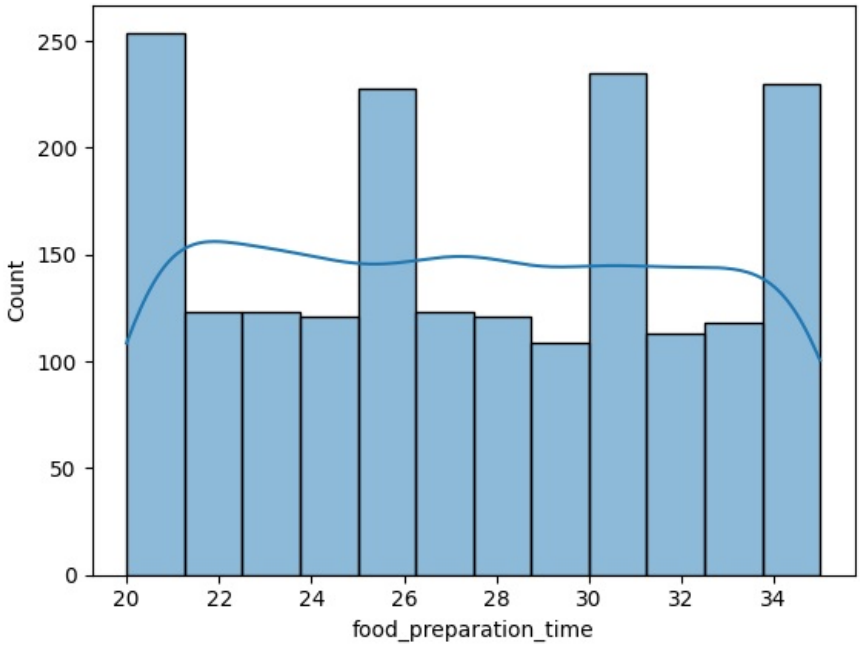


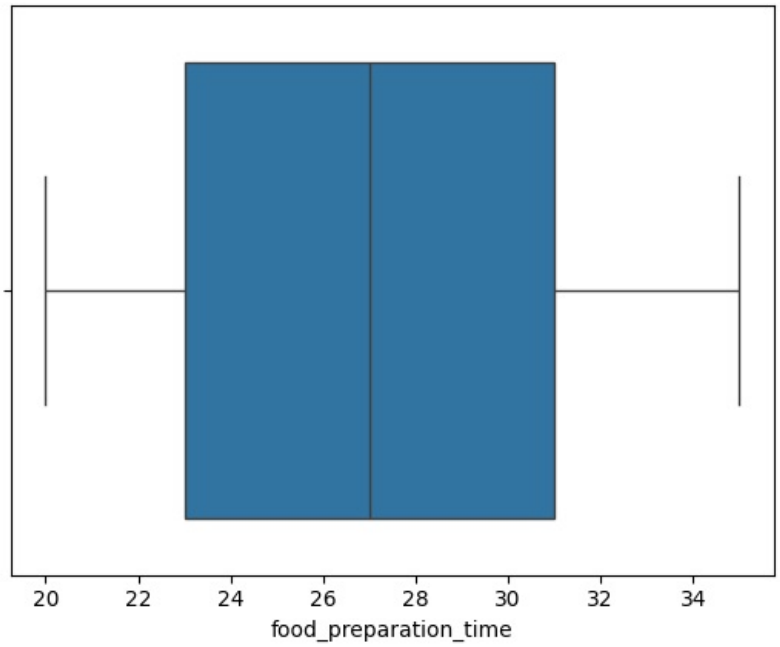
In [52]:
```
# displot and boxplot for 'rating'
sns.displot(data=df, x='rating')
plt.show()
sns.boxplot(data=df, x='rating')
plt.show()
```



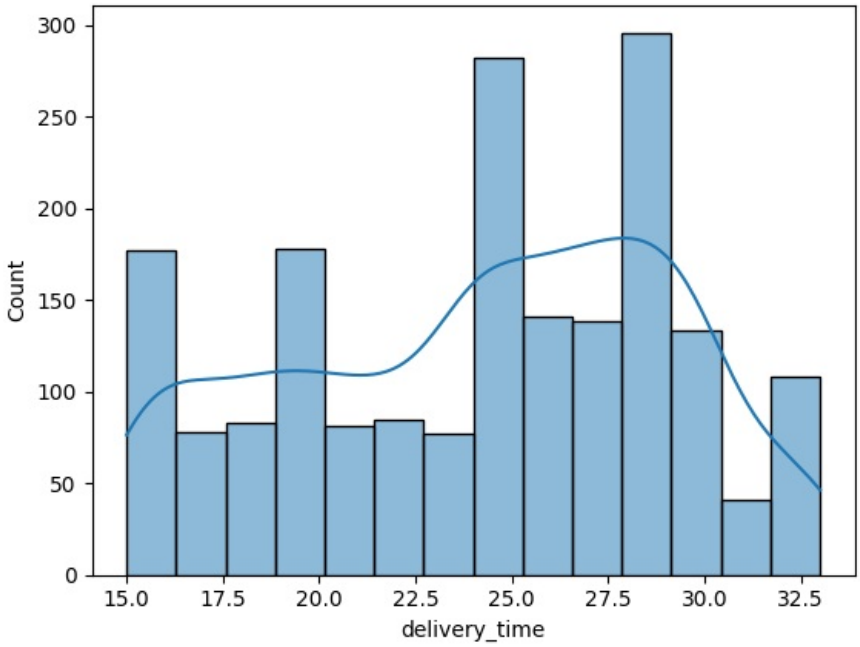


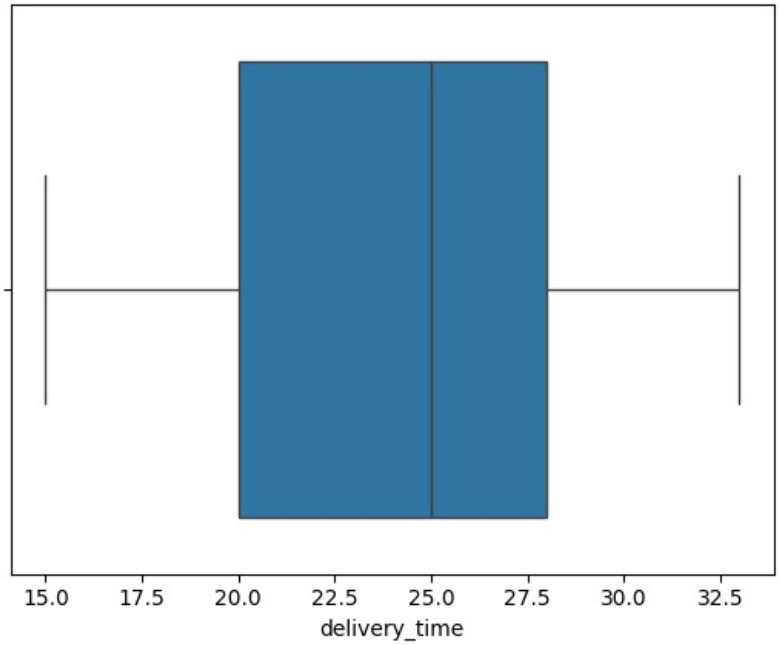
In [53]:
```
# histogram and boxplot for 'food_preperation_time'
sns.histplot(data=df, x='food_preparation_time', kde=True)
plt.show()
sns.boxplot(data=df, x='food_preparation_time')
plt.show()
```





In [54]:
```
# histogram and boxplot for 'delivery_time'
sns.histplot(data=df, x='delivery_time', kde=True)
plt.show()
sns.boxplot(data=df, x='delivery_time')
plt.show()
```





Observations: Using a Histogram & Boxplot to check on the observation for the distribution of the variables in the data.

Starting with the 'cost_of_the_order' variable.

- The distribution is skewed towards the right.
- There are no outliers present for the cost of the order.

Checking the distribution for the 'day_of_the_week' variable.

- There is more than twice the amount of orders placed on the weekend compared to the weekday.

Checking the distribution for the 'rating' variable.

- The distribution is right-skewed as seen in the box plot
- There seems to be more ratings that are 'Not given' than are rated by users.

Checking the distribution for the 'food_preperation_time' variable.

- The distribution for the time it takes for the restaurants to prepare their food is normally distributed.
- There is no skewness as shown in the box plot, where it is towards the center.

Checking the distribution for the 'delivery_time' variable.

- The distribution is skewed towards the left.
- There is a majority in the amount of time an order takes past 23 minutes.

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

In [55]:
```
# Write the code here

# Find the top 5 restaurants in terms of the num of orders received by using groupby of the size in a top-down
df.groupby('restaurant_name').size().sort_values(ascending=False).head()
```

Out[55]:
```
restaurant_name
Shake Shack                  219
The Meatball Shop            132
Blue Ribbon Sushi            119
Blue Ribbon Fried Chicken     96
Parm                          68
dtype: int64
```

Observations: We can see that Shake Shack comes out to be the number 1 most ordered restuarant by a good margin compared to the other top 4 restaurants.

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

In [56]:
```
# Write the code here

# To Find the most popular cusine in the weekend by specifying only the weekend values grouped by the cusine si
df[df['day_of_the_week'] == 'Weekend'].groupby('cuisine_type').size().sort_values(ascending=False).head()
```

Out[56]:
```
cuisine_type
American    415
Japanese    335
Italian     207
Chinese     163
Mexican      53
dtype: int64
```

Observations: As we can see, the most ordered cuisine type in the weekend is 'American'. Whereas the least is 'Mexican' out of the top 5.

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

In [57]:
```
# Write the code here

# To find the percentage of orders that cost more than 20 dollars, filter out the cost of orders that are great
df[df['cost_of_the_order'] > 20].shape[0] / df.shape[0] * 100
```

Out[57]:
```
29.24130663856691
```

Observations: As seen in the code above, the percentage of orders that cost more than 20 dollars is 29%.

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

In [58]:
```
# Write the code here

# To find the mean order of the delivery time, you use the .mean() function on the specified column ('delivery_
df['delivery_time'].mean()
```

Out[58]:
```
24.161749209694417
```

Observations: The mean order of the deliver time is 24 minutes, therefore the average time it takes for the delivery person to deliver the food package is 24 minutes.

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 [59]:
```
# Write the code here

# To find the unique ID's of the customers and the number of orders they had placed, groupby the size of custom
df.groupby('customer_id').size().sort_values(ascending=False).head(3)
```

Out[59]:
```
customer_id
52832    13
47440    10
83287     9
dtype: int64
```

Observations: Customer ID #52832 has the most amount of orders placed [13].

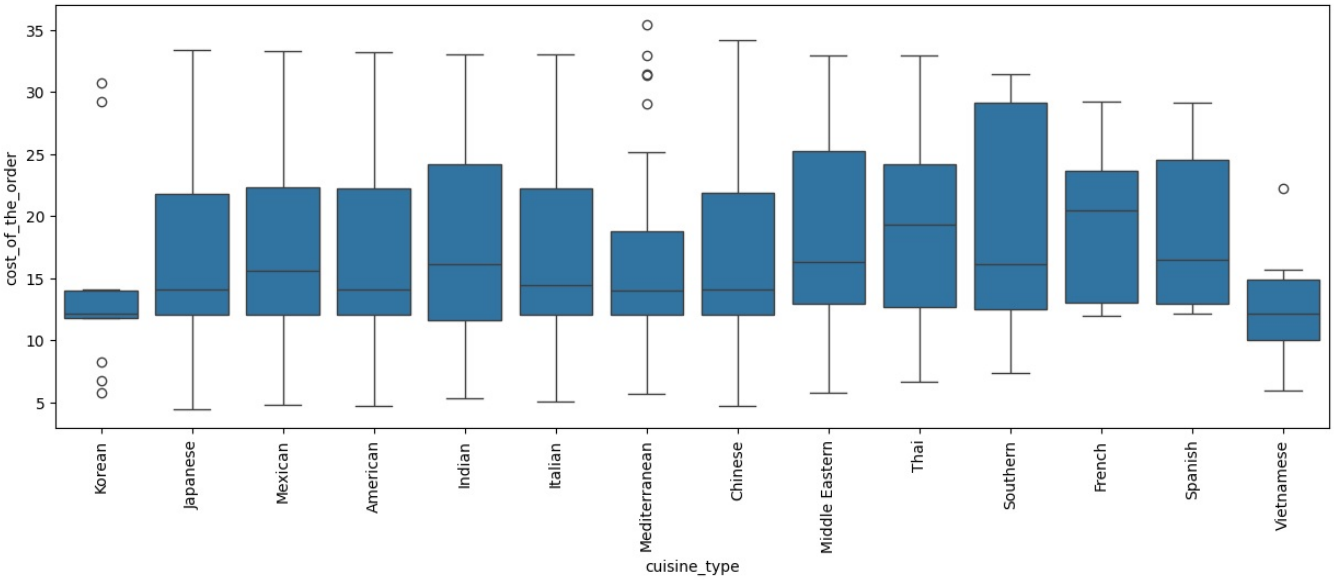
Multivariate Analysis

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]

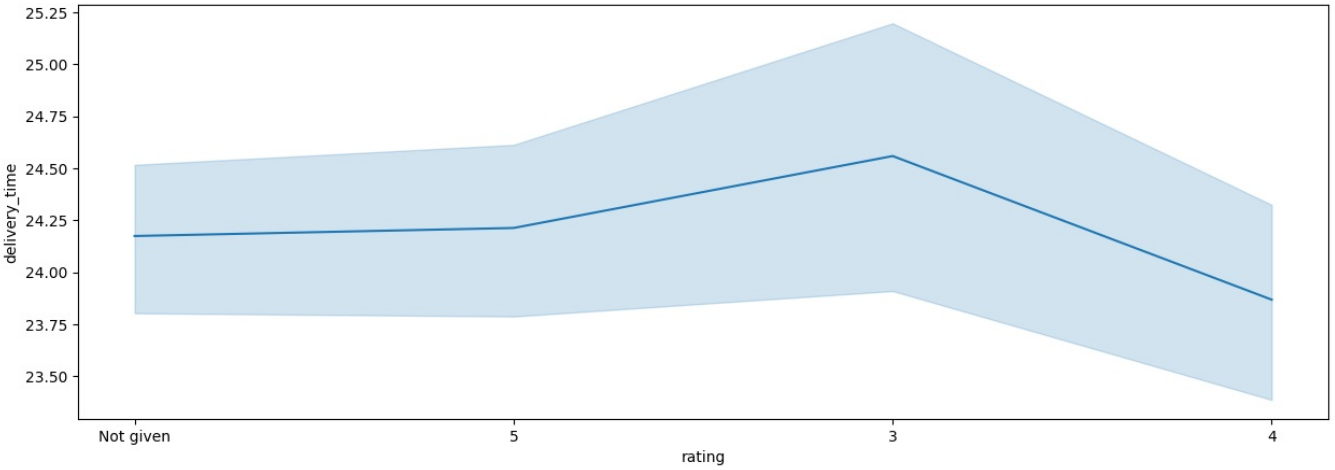
In [60]:
```
# Write the code here

# Exploring the average cost of the order per cuisine type, make a boxplot where x is the cusine type and y is
plt.figure(figsize=(15,5))
sns.boxplot(data=df, x='cuisine_type',  y='cost_of_the_order')
plt.xticks(rotation=90)
plt.show()
```



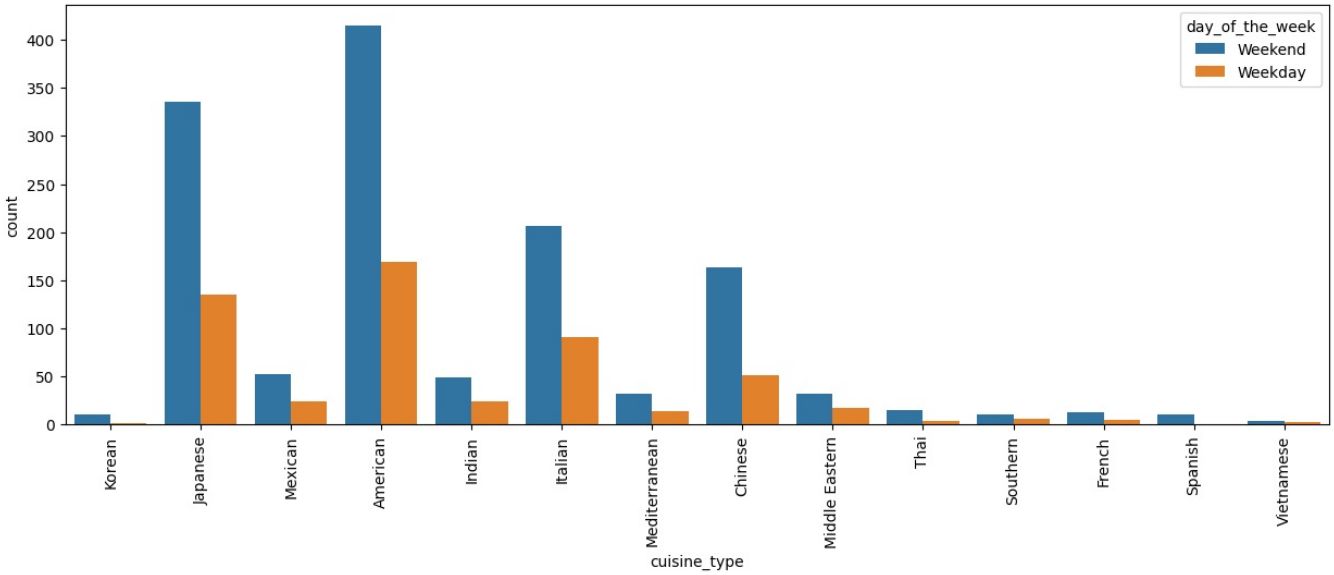
Observations: As seen in the boxplot above, the Vietnamese cuisine type averages the least to order, whilst average for French cusine's cost the most. There seems to be outliers most present in the Mediterranean cuisine type as well as the Korean cuisine type compared to the rest.

In [61]:
```
# Exploring the relationship between the ratings the customers give based on the
# quickness of the time it takes for the order to deliver, by using a line plot
# of x being the customers rating and the order delivery time.
plt.figure(figsize=(15,5))
sns.lineplot(data=df, x='rating', y='delivery_time')
plt.show()
```



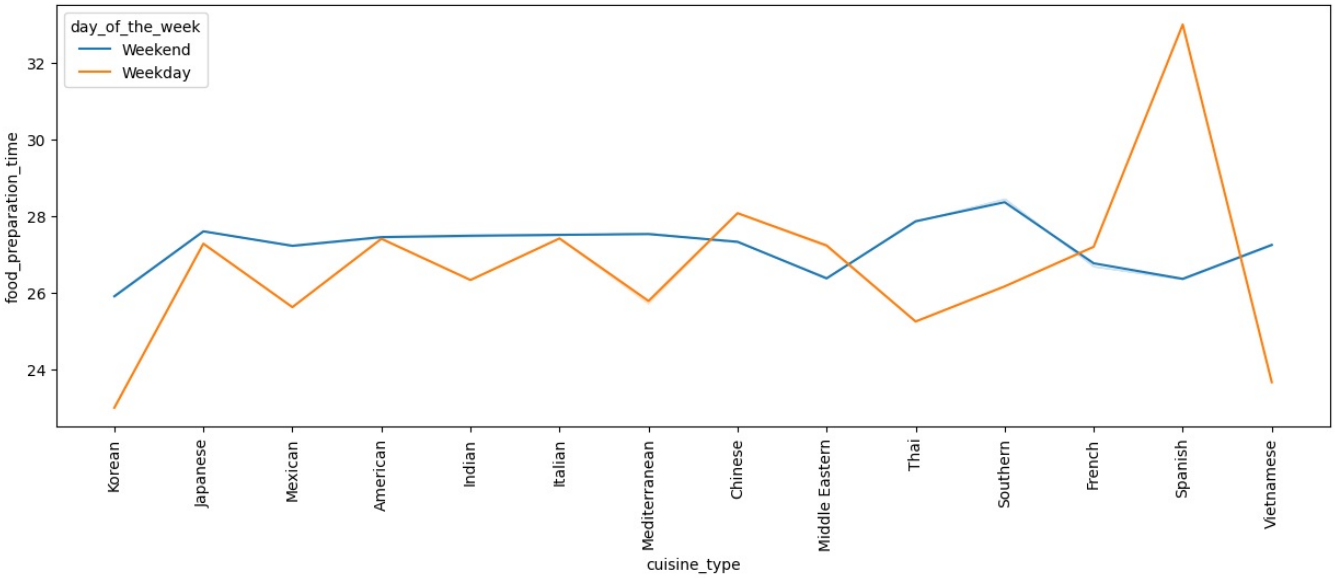
Observations: As seen in the above line plot, users that provided a rating of 3 were correlated to the most amount of time for orders to be delivered, where as ratings that were 5 and 4 were given in a quicker delivery time.

In [62]:
```
# Exploring the relationship between the type of cuisine the customers like to
# order from in either the weekdays or weekend. Using a count plot of cuisine type
# and filtering out between the weekend and weekdays.
plt.figure(figsize=(15,5))
sns.countplot(data=df, x='cuisine_type', hue='day_of_the_week')
plt.xticks(rotation=90)
plt.show()
```



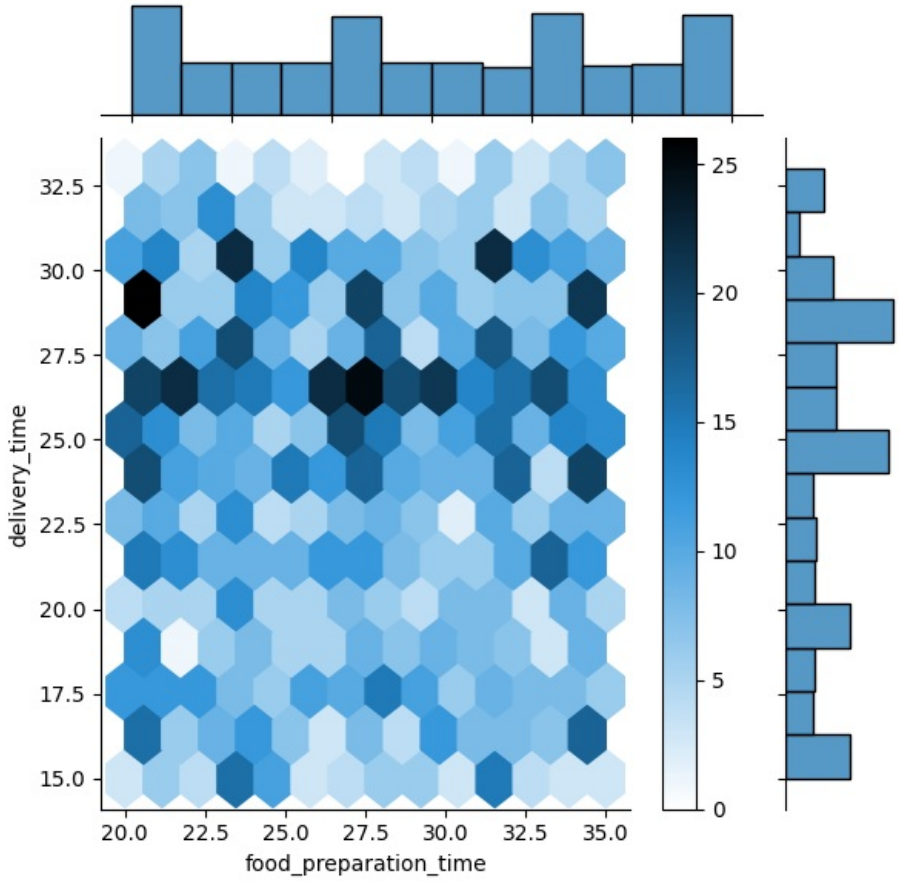
Observations: As seen in the above count plot, the American and Japanese cuisine types are favorited the most throughout the entire week by the customers. Whereas Korean and French are not so much ordered. This graph also presents a distribution that is skewed towards the right.

In [63]:
```
# Exploring which cuisine type contains the fastest food prepartion time depending
# on the day of the week. Using a line plot of cusine type and food prepartion time
# by filtering it to the days of the week.
plt.figure(figsize=(15,5))
sns.lineplot(data=df, x='cuisine_type', y='food_preparation_time', ci = False, hue='day_of_the_week')
plt.xticks(rotation=90)
plt.show()
```



Observations: As seen in the line plot, the cuisine types contain around the same amount of time to prepare the food on the weekends compared to the weekdays. The Spanish cuisine type has a very significant time to prepare food compared to all of the others. It seems that there is an outlier in the time it takes the spanish cuisine type to prepare their food compared to the rest which is not good for the customers satisfaction.

In [64]:
```
# Finding the correaltion between the food preperation time and the delivery time
# by using a hexagonal joint plot for the time it takes the food to prepare to the
# time it takes the order to be delivered to the customers house.
sns.jointplot(data=df, x='food_preparation_time', y='delivery_time', kind='hex');
plt.colorbar()
plt.show()
```



Observations: In the above joint plot, there is a high correlation between the time it takes for the food to prepare and the delivery time around 20 to 28 minutes in the x-axis and 25 to 31 minute in the y-axis. For both the x and y variables, there doesn't seem to be much skewness in occuring in the outer histograms, rather they appear to be normally distributed.

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 [65]:
```
# Write the code here

# First convert the 'Not given' values in the rating column to NaN values
df['rating'] = df['rating'].replace('Not given', np.nan)
df['rating'] = df['rating'].astype(float)

# Find the names of the restuarants that contain a rating count of more than 50 and the average rating should b
restaurants = df.groupby('restaurant_name').filter(lambda x: len(x) > 50 and x['rating'].mean() > 4)['restauran

# Printing the output of the above calculations
print("Restaurants with more than 50 reviews and an average rating above 4:\n")
for restaurant in restaurants:
    print(f"- {restaurant}")
```

```
Restaurants with more than 50 reviews and an average rating above 4:

- Blue Ribbon Fried Chicken
- The Meatball Shop
- Shake Shack
- RedFarm Hudson
- Blue Ribbon Sushi
- Parm
- RedFarm Broadway
```

In [66]:
```
# Finding the total amount of unique restaurants in the data for better observation.
df['restaurant_name'].describe()
```

Out[66]:
```
count            1898
unique            178
top       Shake Shack
freq              219
Name: restaurant_name, dtype: object
```

Observations: In the above code, we filtered out the restaurants that were rated over 50 times and had an average rating of over 4. With that calculation, only 7 out of 178 restaurants ended up being awarded the promotional offer.

Question 14: The company charges the restaurant 25% on the orders having cost greater than

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 [67]:
```
# Write the code here

# Applying the percentages based on order cost thresholds of the question.
df['net_revenue'] = np.where(df['cost_of_the_order'] > 20, df['cost_of_the_order'] * 0.25, np.where(df['cost_of

# Finding the net revenue generated by the company across all orders by using the sum function.
total_net_revenue = df['net_revenue'].sum()

print("The net revenue generated by the company across all orders is:", total_net_revenue)
```

```
The net revenue generated by the company across all orders is: 6166.303
```

In [68]:
```
# Finding the total sum of orders without any charge from the company.
total_orders = df['cost_of_the_order'].sum()

print("The total sum of orders without any discount is:", total_orders)
```

```
The total sum of orders without any discount is: 31314.82
```

In [69]:
```
# Finding the percent of charges the company had taken off from the restuarants.
total_orders = 31314
total_net_revenue = 6166

# Calculate the percentage by dividing the total net revenue from the total amount of orders times 100.
percentage = (total_net_revenue / total_orders) * 100

# Printing the result in two decimal format.
print(f"The percentage taken by the company from the total amount is: {percentage:.2f}%")
```

```
The percentage taken by the company from the total amount is: 19.69%
```

Observations: We observed that the company had made 6166 dollars when applying the chargers of the orders. The total sum of the orders without any charge from the company is 31314 dollars, therefore the total percent the company had taken from the restaurants is 19.70%.

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 [70]:
```
# Write the code here

# First adding up the food preperation time and the delivery time columns for better data filterization
df['total_time'] = df['food_preparation_time'] + df['delivery_time']

# Finding the percentage of orders that take more than 60 minutes to get delivered
calculation = df.loc[df['total_time'] > 60 ]['total_time'].count() / df['total_time'].count() * 100

print(f"The percentage of orders that take more than 60 minutes to get delivered from the time they are placed
```

```
The percentage of orders that take more than 60 minutes to get delivered from the time they are placed is: 10.5
4%
```

In [71]:
```
# Finding the total amount of orders there are in the data for better analysis on the observation.
total_orders = df['total_time'].count()
print("The total amount of orders is:" ,total_orders, "Orders")
```

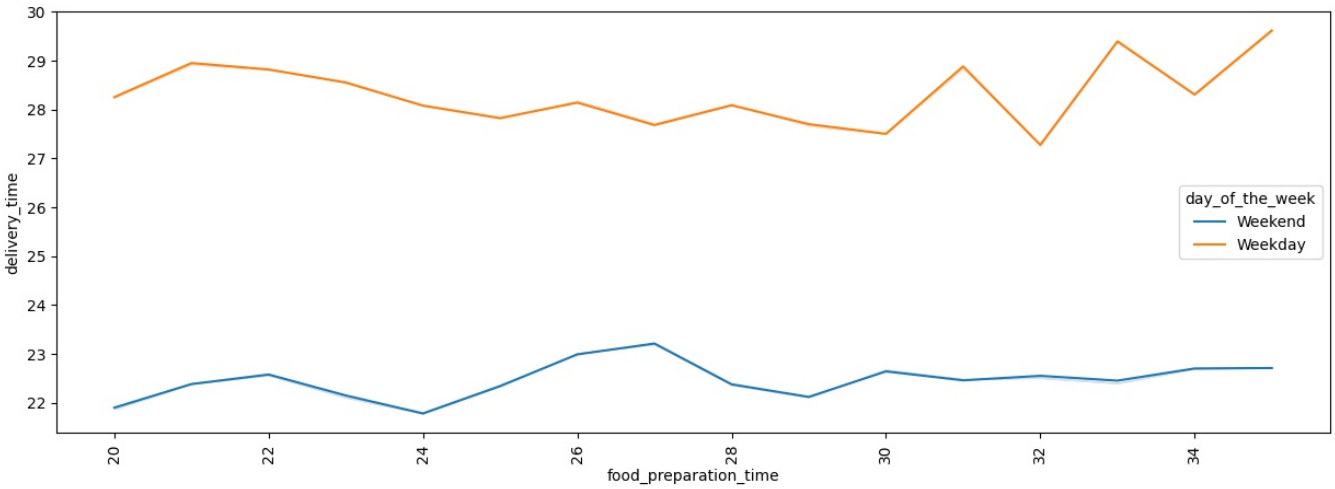
```
The total amount of orders is: 1898 Orders
```

Observations: As we can observed in the above calculation, there are only 10.5% of orders out off 1898 orders that take more than 60 minutes to get delivered once they are first placed by the customer. This percentage being a low number is quite good because it displays that orders do not take more than an hour to get to the customers reach.

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]

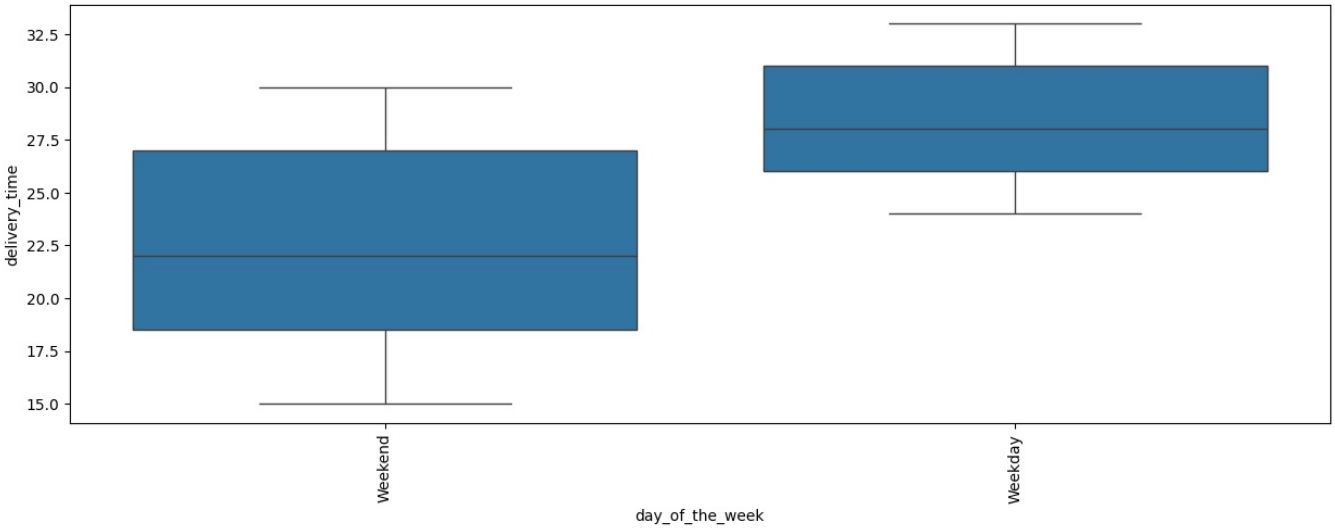
In [72]:
```
# displaying a line plot to analyze the average deliver time of the orders made on the weekend compared to the

plt.figure(figsize=(15,5))
sns.lineplot(data=df, x='food_preparation_time', y='delivery_time',  hue='day_of_the_week', ci=False)
plt.xticks(rotation=90)
plt.show()
```



In [73]:
```
# Utilizing a box plot for further analyzation.

plt.figure(figsize=(15,5))
sns.boxplot(data=df, x='day_of_the_week', y='delivery_time')
plt.xticks(rotation=90)
plt.show()
```

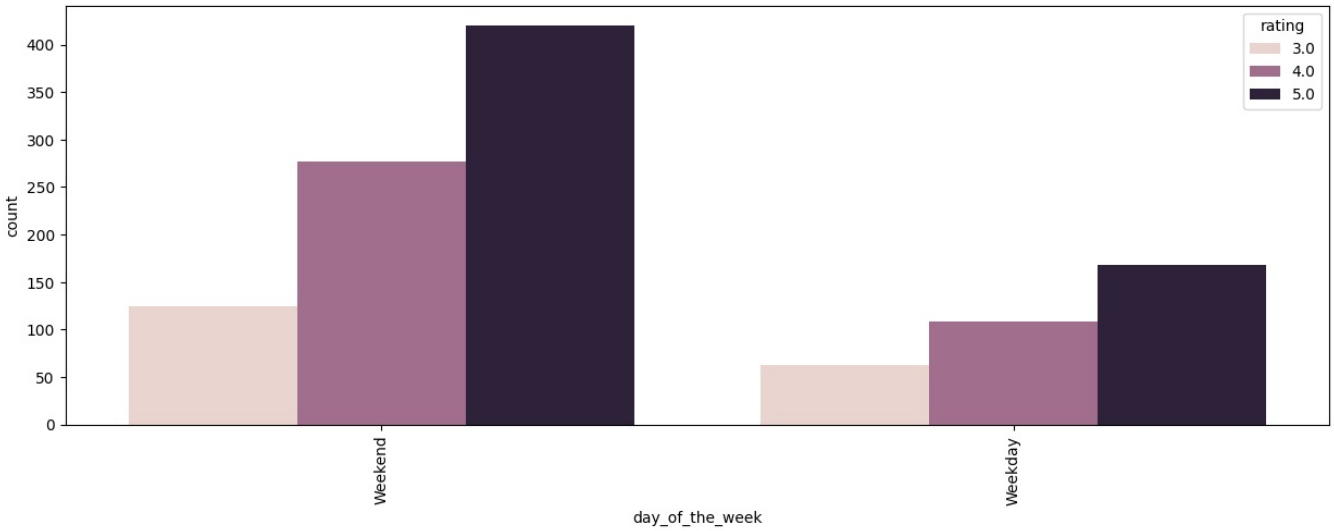


Observations: As seen in the above line plot and box plot, during the weekends, the average delivery time is unexpectedly much less compared in the weekdays. I would of assumed the value to be a little more compared to the weekdays. Although, having the location of this data being from New York and the majority of the customers are busy students and professionals, the average timings between weekends and weekdays makes sense.

Conclusion and Recommendations

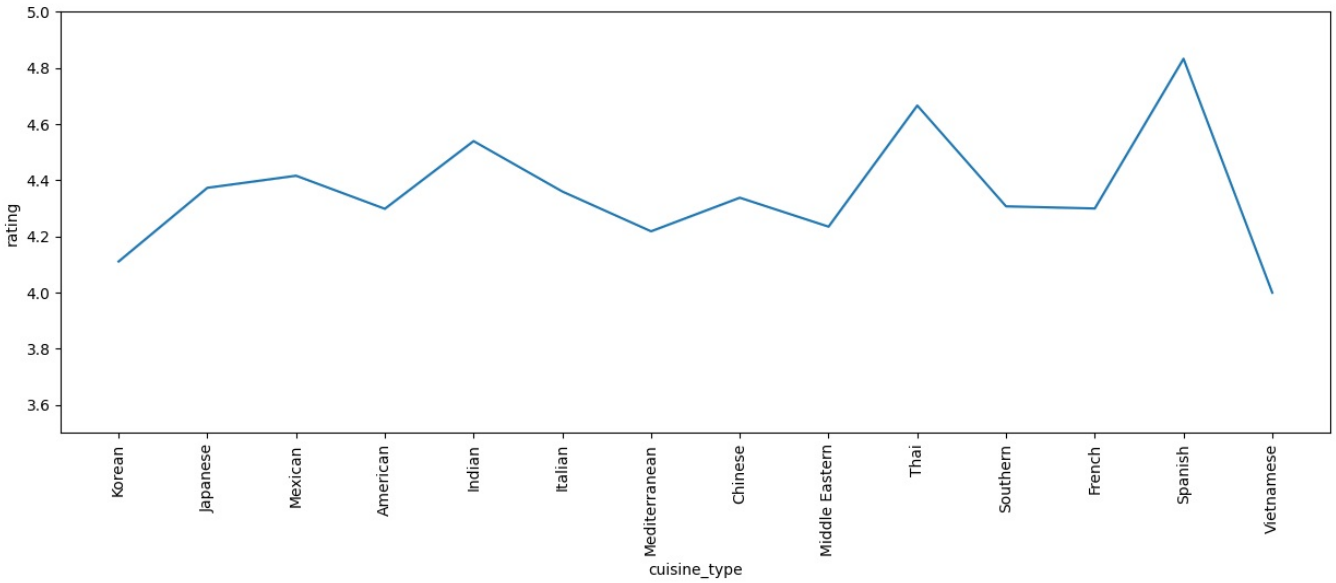
Question 17: What are your conclusions from the analysis? What recommendations would you like to share to help improve the business? (You can use cuisine type and feedback ratings to drive your business recommendations.) [6 marks]

In [74]:
```
# using a count plot to analyze the ratings throughout the days of the week
plt.figure(figsize=(15,5))
sns.countplot(data=df, x='day_of_the_week', hue='rating')
plt.xticks(rotation=90)
plt.show()
```



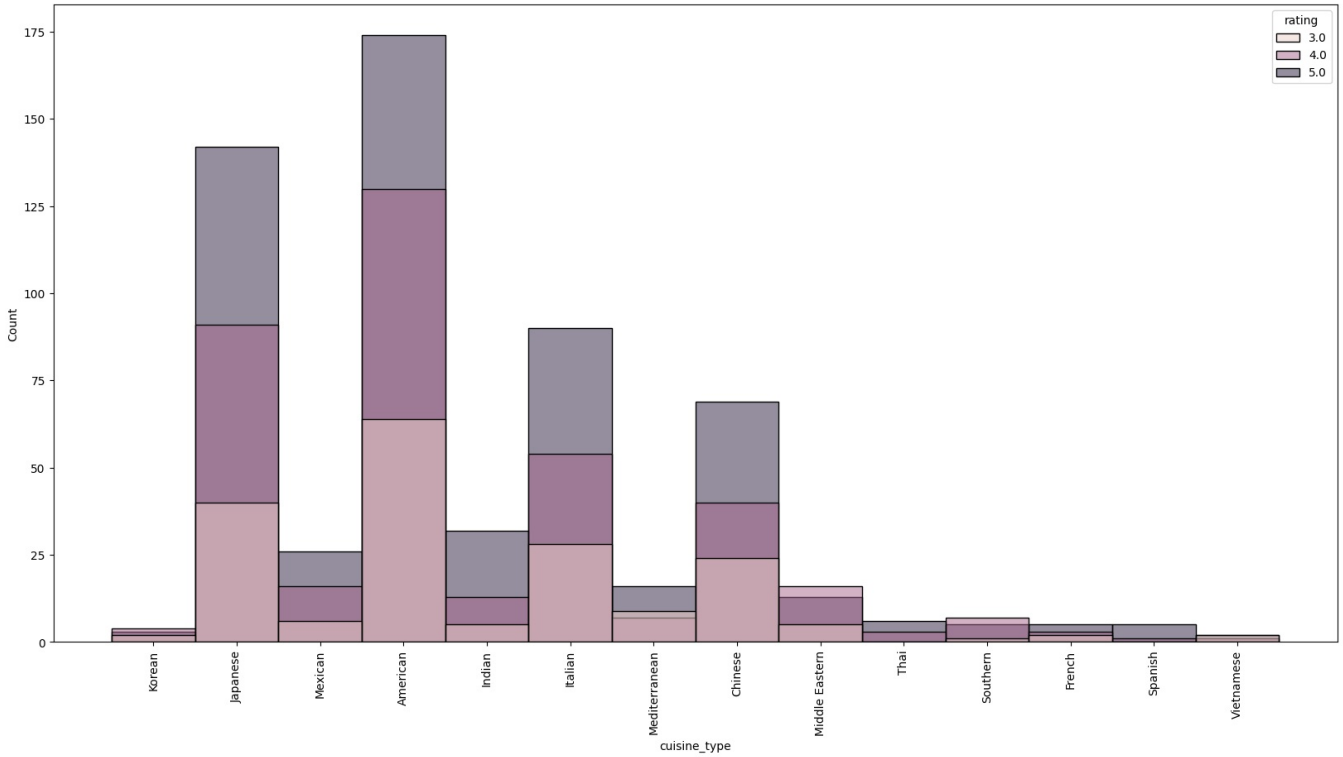
In [75]:
```
# Using a line plot to help formulate recommendations.

plt.figure(figsize=(15,5))
sns.lineplot(data=df, x='cuisine_type', y='rating', ci=False)
plt.xticks(rotation=90)
plt.ylim(3.5,5)
plt.show()
```



In [76]:
```
# Using a histogram to support recommendations.

plt.figure(figsize=(20,10))
sns.histplot(data=df, x='cuisine_type', hue='rating')
plt.xticks(rotation=90)
plt.show()
```



Conclusions:

- In conclusion, there seems to be a good amount of missing and uneven data throughout the FoodHub. For instance, there is a lot of customers who did not provide ratings which doesn't end up helping the restaurants representation.
- There are also some cuisine types that do not have enough popularity compared to the others by a majority. For example, Korean and Vietnamese are heavily low rated in terms of amount of customer ratings compared to American, Japanese, and Italian.
- The average delivery time in the weekdays is higher by a good margin compared to the weekends, this could be due to the type of customers that are in that area which are busy students and professionals.

Recommendations:

1. I would recommend that there should be more of a balance between the popularity among the cuisine types
2. Require customers to provide feedback on their ratings to better help the restuarants and their cuisine types more profitable
    - Although, for making this happen those types of restuarants should have a better overall service of food, the quality, and hospitality.
    - This could be one reason why there are much less reviews and customers buying from them amongst the rest.
3. There are more ratings in the weekend compared to the weekdays, encourage customers to review their orders throughout the week.
    - However, this obviously could mean that the customers have more time in the weekends compared to the weekdays to rate their order, when they are not working/studying.