In [58]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from collections import Counter
import re
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import nltk
import warnings
warnings.filterwarnings('ignore')
```

In [59]:
```python
#load and read dataset
df=pd.read_csv('Dataset.csv')
df
```

Out[59]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...
...	...	...	...	...	...	...	...	...	...	...	...
9546	5915730	Namlı Gurme	208	��stanbul	Kemanke�� Karamustafa Pa��a Mahallesi, Rıhtım ...	Karak�_y	Karak�_y, ��stanbul	28.977392	41.022793	Turkish	...
9547	5908749	Ceviz A��acı	208	��stanbul	Ko��uyolu Mahallesi, Muhittin ��st�_nda�� Cadd...	Ko��uyolu	Ko��uyolu, ��stanbul	29.041297	41.009847	World Cuisine, Patisserie, Cafe	...
9548	5915807	Huqqa	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.034640	41.055817	Italian, World Cuisine	...
9549	5916112	A���k Kahve	208	��stanbul	Kuru�_e��me Mahallesi, Muallim Naci Caddesi, N...	Kuru�_e��me	Kuru�_e��me, ��stanbul	29.036019	41.057979	Restaurant Cafe	...
9550	5927402	Walter's Coffee Roastery	208	��stanbul	Cafea��a Mahallesi, Bademaltı Sokak, No 21/B, ...	Moda	Moda, ��stanbul	29.026016	40.984776	Cafe	...

9551 rows × 21 columns

In [ ]:

Level 3

Task 1

Cognifyz
Where Data Meets Intelligence

Task: Restaurant Reviews

Analyze the text reviews to identify the most common positive and negative keywords.

Calculate the average length of reviews and explore if there is a relationship between

review length and rating.

Task

1. Analyze the text reviews to identify the most common positive and negative keywords.

In [60]:
```
# Inspect rating distribution
rating_counts = df['Rating text'].value_counts()
print(rating_counts)
```

```
Rating text
Average      3737
Not rated    2148
Good         2100
Very Good    1079
Excellent     301
Poor          186
Name: count, dtype: int64
```

In [61]:
```
# Group by Aggregate rating, Rating color, and Rating text
grouped_ratings = df.groupby(['Aggregate rating', 'Rating color', 'Rating text']).size().reset_index(name='tota
print(grouped_ratings)
```

```
    Aggregate rating Rating color Rating text  total count
0                0.0        White   Not rated         2148
1                1.8          Red        Poor            1
2                1.9          Red        Poor            2
3                2.0          Red        Poor            7
4                2.1          Red        Poor           15
5                2.2          Red        Poor           27
6                2.3          Red        Poor           47
7                2.4          Red        Poor           87
8                2.5       Orange     Average          110
9                2.6       Orange     Average          191
10               2.7       Orange     Average          250
11               2.8       Orange     Average          315
12               2.9       Orange     Average          381
13               3.0       Orange     Average          468
14               3.1       Orange     Average          519
15               3.2       Orange     Average          522
16               3.3       Orange     Average          483
17               3.4       Orange     Average          498
18               3.5       Yellow        Good          480
19               3.6       Yellow        Good          458
20               3.7       Yellow        Good          427
21               3.8       Yellow        Good          400
22               3.9       Yellow        Good          335
23               4.0        Green   Very Good          266
24               4.1        Green   Very Good          274
25               4.2        Green   Very Good          221
26               4.3        Green   Very Good          174
27               4.4        Green   Very Good          144
28               4.5   Dark Green   Excellent           95
29               4.6   Dark Green   Excellent           78
30               4.7   Dark Green   Excellent           42
31               4.8   Dark Green   Excellent           25
32               4.9   Dark Green   Excellent           61
```

In [62]:
```
# Download stopwords
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\MANISH\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\MANISH\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[62]:
```
True
```

In [63]:
```
# Function to extract words from text
def extract_words(text):
    # Tokenize text
    tokens = word_tokenize(text.lower())
    # Remove punctuation and stopwords
    tokens = [word for word in tokens if word.isalpha() and word not in stopwords.words('english')]
    return tokens
```

In [64]:
```
# Analyze text reviews and extract keywords
positive_words = Counter()
negative_words = Counter()
```

In [65]:
```
positive_words
```

```
In [65]: positive_words
```

Out[65]:
```
Counter()
```

In [66]:
```
negative_words
```

Out[66]:
```
Counter()
```

Task

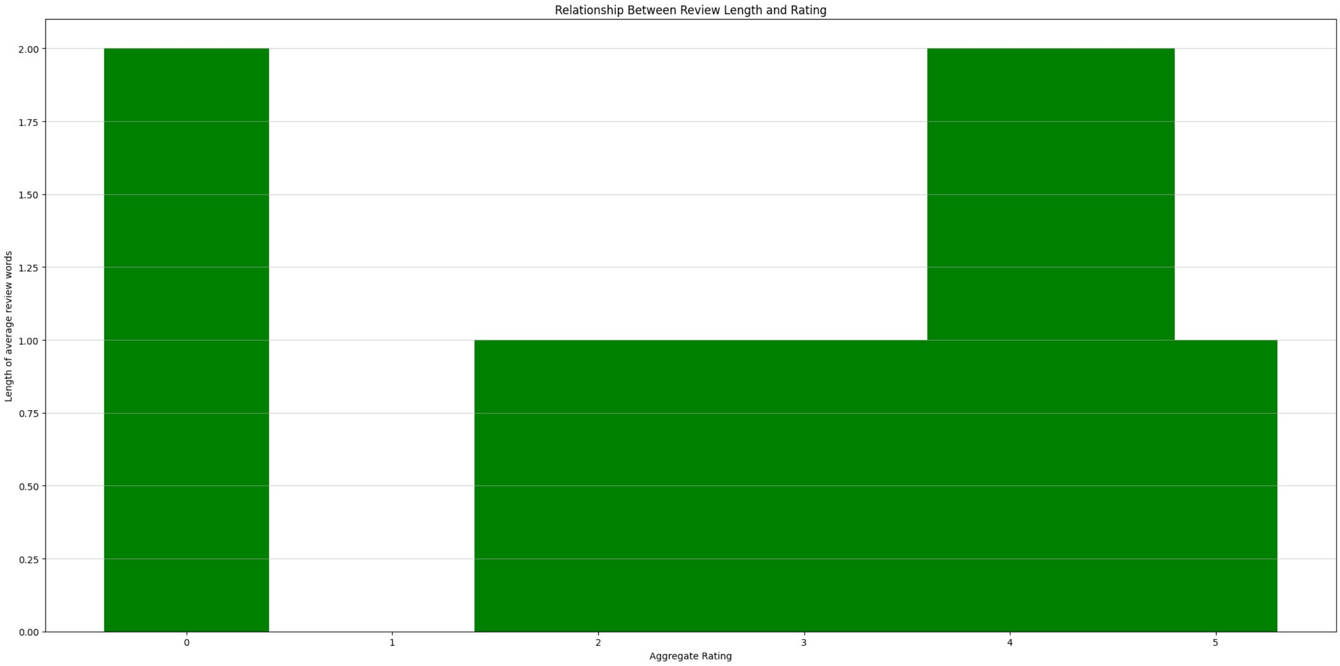
1. Calculate the average length of reviews and explore if there is a relationship between review length and rating.

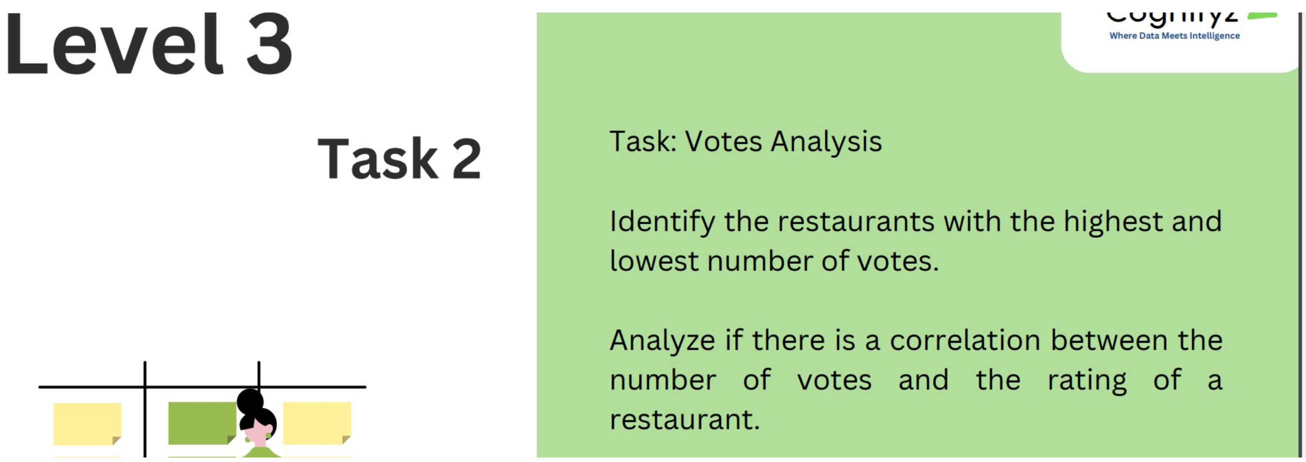
In [67]:
```
df['Review length']=df['Rating text'].apply(lambda x:len(x.split()))
```

In [68]:
```
avg_Rating_len = df.groupby('Aggregate rating')['Review length'].mean()
avg_Rating_len
```

Out[68]:
```
Aggregate rating
0.0    2.0
1.8    1.0
1.9    1.0
2.0    1.0
2.1    1.0
2.2    1.0
2.3    1.0
2.4    1.0
2.5    1.0
2.6    1.0
2.7    1.0
2.8    1.0
2.9    1.0
3.0    1.0
3.1    1.0
3.2    1.0
3.3    1.0
3.4    1.0
3.5    1.0
3.6    1.0
3.7    1.0
3.8    1.0
3.9    1.0
4.0    2.0
4.1    2.0
4.2    2.0
4.3    2.0
4.4    2.0
4.5    1.0
4.6    1.0
4.7    1.0
4.8    1.0
4.9    1.0
Name: Review length, dtype: float64
```

In [69]:
```
plt.figure(figsize=(20,10))
plt.bar(avg_Rating_len.index,avg_Rating_len.values,color='green')
plt.xlabel('Aggregate Rating')
plt.ylabel('Length of average review words')
plt.title("Relationship Between Review Length and Rating")
plt.grid(axis='y',linestyle='-',alpha=0.6)
plt.tight_layout()
plt.show()
```





Task

1. Identify the restaurants with the highest and lowest number of votes

In [70]:
```
hi_vote=df.groupby('Restaurant Name')["Votes"].sum().reset_index(name="vote count")
```

In [71]:
```
sort_votes=hi_vote.sort_values(by='vote count',ascending=False)
top_10 = sort_votes.head(10)
top_10
```

Out[71]:

	Restaurant Name	vote count
663	Barbeque Nation	28142
101	AB's - Absolute Barbecues	13400
6943	Toit	10934
785	Big Chill	10853
2297	Farzi Cafe	10098
6988	Truffles	9682
1510	Chili's	8156
2879	Hauz Khas Social	7931
3261	Joey's Pizza	7807
4902	Peter Cat	7574

Task

1. Analyze if there is a correlation between the number of votes and the rating of a restaurant.

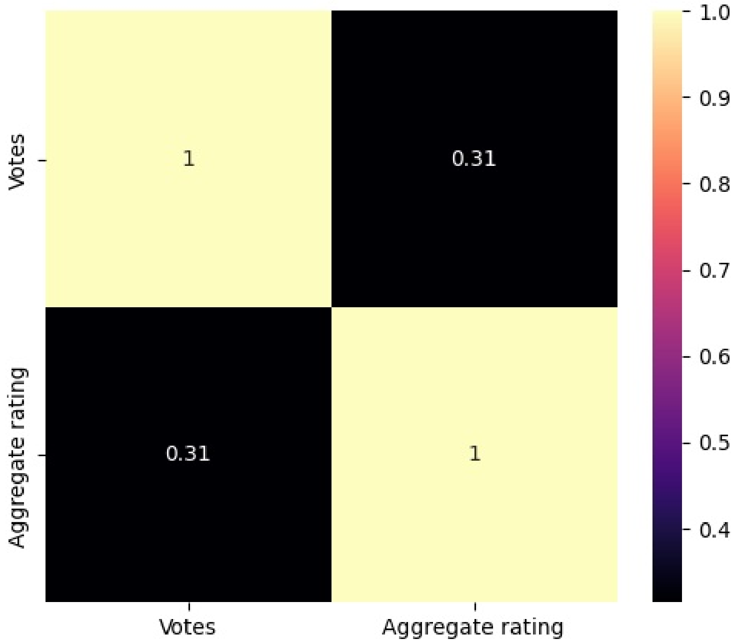
In [72]:
```
# Convert Votes to numeric (in case it is not)
```

```
df['Votes'] = pd.to_numeric(df['Votes'], errors='coerce')
```

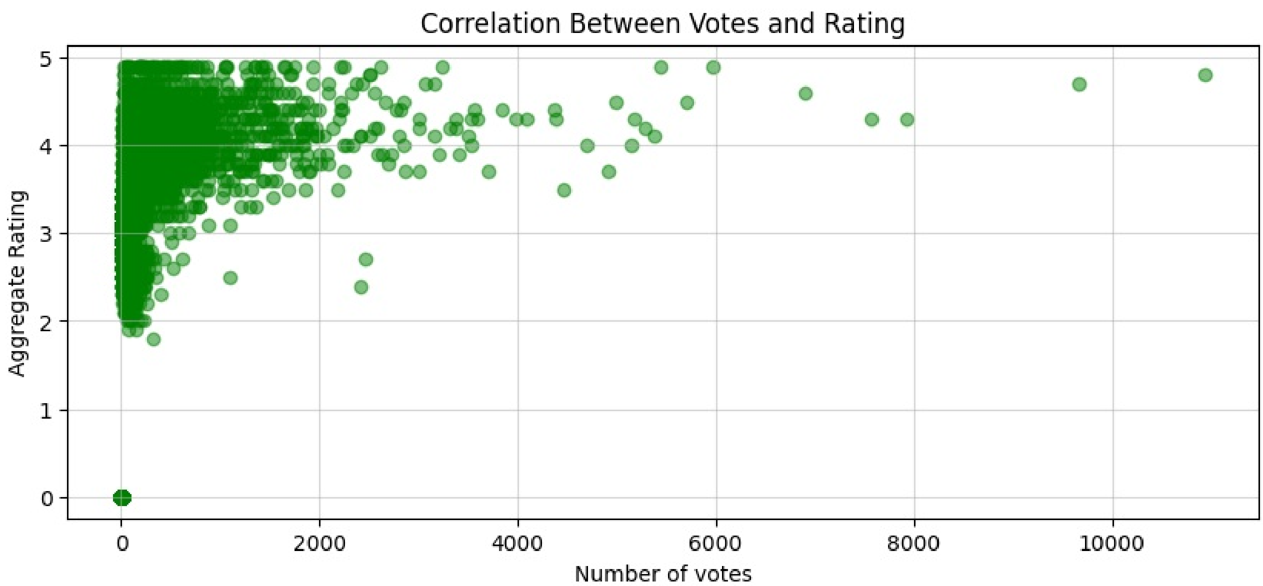
In [73]:
```
#calculate the correlation between number of vote and rating
correlation = df[['Votes','Aggregate rating']].corr()
print("Correlation between Number of votes and Rating:", correlation)
```

```
Correlation between Number of votes and Rating:                   Votes  Aggregate rating
Votes             1.000000          0.313691
Aggregate rating  0.313691          1.000000
```

In [74]:
```
plt.figure(figsize=(6,5))
sns.heatmap(correlation, cmap='magma', annot=True)
plt.show()
```



In [75]:
```
# top_10.plot(kind='bar')
plt.figure(figsize=(10,4))
plt.scatter(df['Votes'],df['Aggregate rating'],alpha=0.5,color='green')
plt.xlabel("Number of votes")
plt.ylabel("Aggregate Rating")
plt.title("Correlation Between Votes and Rating")
plt.grid(axis='both',linestyle='-',alpha=0.5)
```

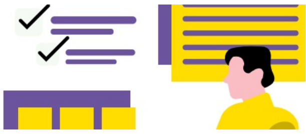


Level 3

Task 3

Task: Price Range vs. Online Delivery and Table Booking

Analyze if there is a relationship between the price range and the availability of online delivery and table booking.



Determine if higher-priced restaurants are more likely to offer these services.

Task

1. Analyze if there is a relationship between the price range and the availability of online delivery and table booking

In [76]:

```
# Convert relevant columns to categorical types for easier analysis
df['Price range'] = df['Price range'].astype('category')
df['Has Table booking'] = df['Has Table booking'].astype('category')
df['Has Online delivery'] = df['Has Online delivery'].astype('category')
# Cross-tabulation of Price range with Table booking and Online delivery
table_booking_vs_price = pd.crosstab(df['Price range'], df['Has Table booking'], normalize='index') * 100
online_delivery_vs_price = pd.crosstab(df['Price range'], df['Has Online delivery'], normalize='index') * 100
print(table_booking_vs_price)
print(online_delivery_vs_price)
```

```
Has Table booking          No        Yes
Price range
1                   99.977498   0.022502
2                   92.322518   7.677482
3                   54.261364  45.738636
4                   53.242321  46.757679
Has Online delivery          No        Yes
Price range
1                     84.225923  15.774077
2                     58.689367  41.310633
3                     70.809659  29.190341
4                     90.955631   9.044369
```

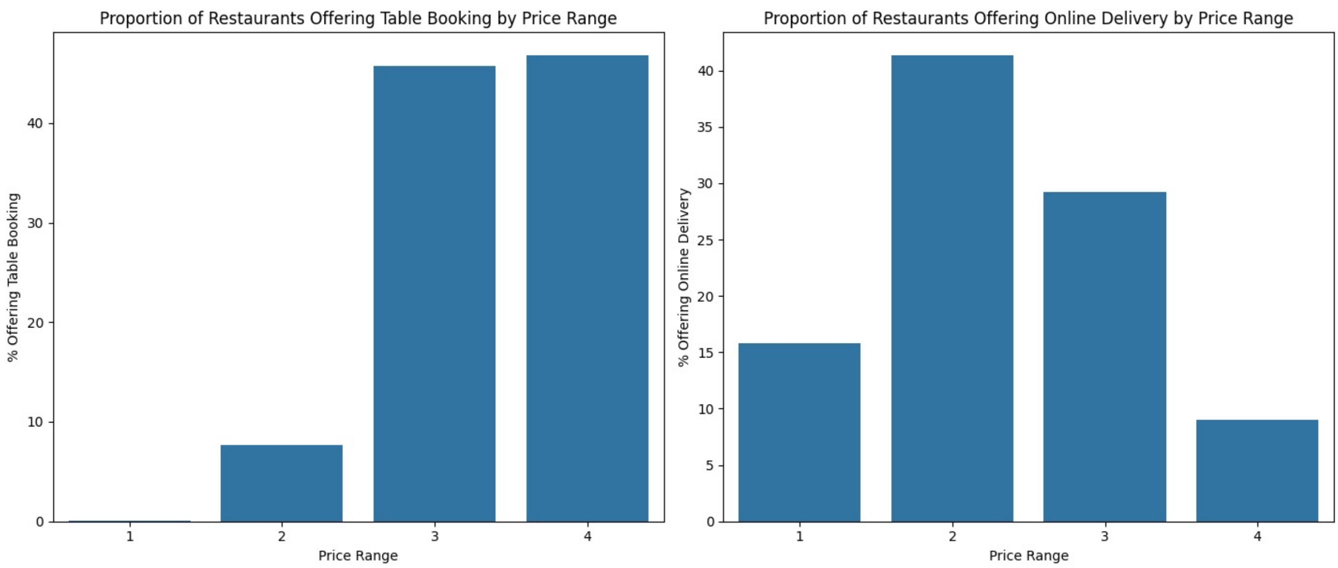
Task

1. Determine if higher-priced restaurants are more likely to offer these services.

In [77]:

```
# Calculate the correlation between Price range and service availability
table_booking_correlation = df['Price range'].cat.codes.corr(df['Has Table booking'].cat.codes)
online_delivery_correlation = df['Price range'].cat.codes.corr(df['Has Online delivery'].cat.codes)
print(table_booking_correlation,online_delivery_correlation)
# Plotting the trends
fig, axes = plt.subplots(1, 2, figsize=(14, 6))
# Table booking vs Price range
sns.barplot(x=table_booking_vs_price.index, y=table_booking_vs_price['Yes'], ax=axes[0])
axes[0].set_title('Proportion of Restaurants Offering Table Booking by Price Range')
axes[0].set_ylabel('% Offering Table Booking')
axes[0].set_xlabel('Price Range')
# Online delivery vs Price range
sns.barplot(x=online_delivery_vs_price.index, y=online_delivery_vs_price['Yes'], ax=axes[1])
axes[1].set_title('Proportion of Restaurants Offering Online Delivery by Price Range')
axes[1].set_ylabel('% Offering Online Delivery')
axes[1].set_xlabel('Price Range')
plt.tight_layout()
plt.show()
```

```
0.5019247250371413 0.07791776880448596
```



In [ ]:

In [ ]:

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js