Reading the File

In [1]:
```
import pandas as pd
file_path = "Restaurant_Reviews.tsv"
df = pd.read_csv(file_path, delimiter='\t', quoting=3)
df.head()
```

Out[1]:

	Review	Liked
0	Wow... Loved this place.	1
1	Crust is not good.	0
2	Not tasty and the texture was just nasty.	0
3	Stopped by during the late May bank holiday of...	1
4	The selection on the menu was great and so wer...	1

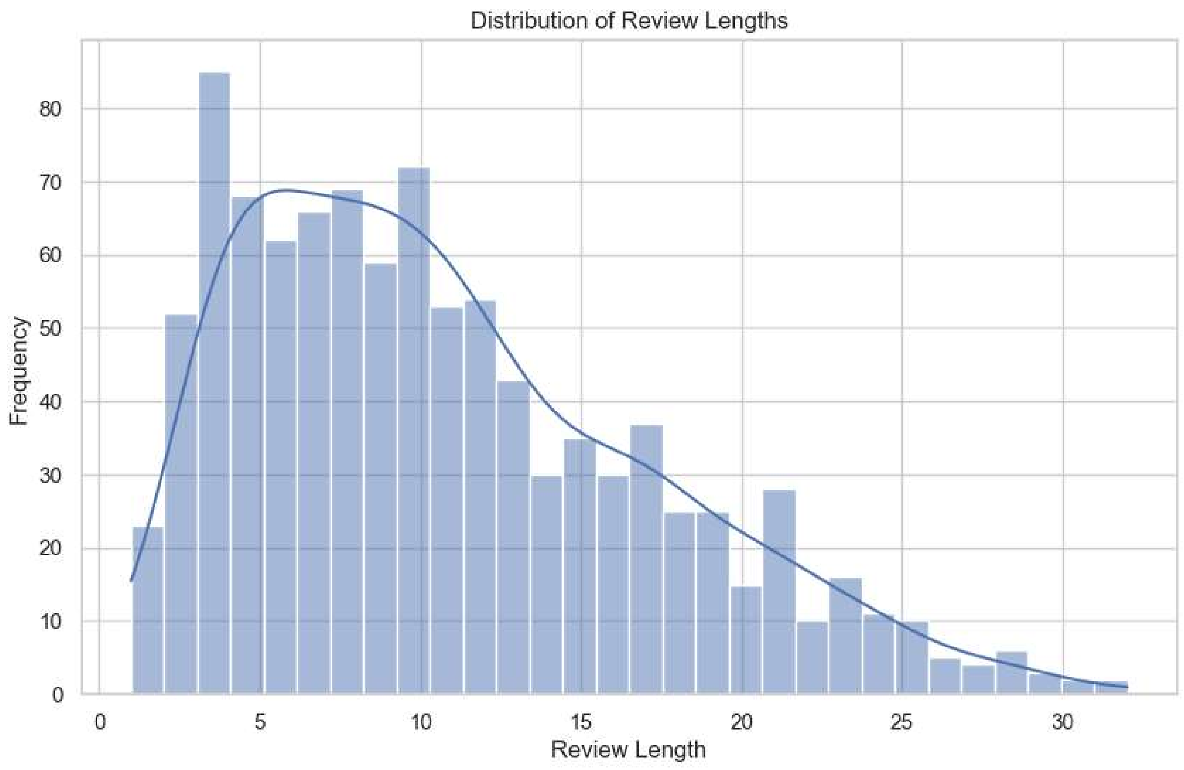
In [2]:
```
# Perform exploratory data analysis
# Calculate the number of reviews
total_reviews = len(df)

# Calculate the average length of reviews
df['Review_Length'] = df['Review'].apply(lambda x: len(x.split()))
average_review_length = df['Review_Length'].mean()

# Print EDA results
print("\nExploratory Data Analysis:")
print("Total number of reviews:", total_reviews)
print("Average review length:", average_review_length)

# Display descriptive statistics of review lengths
print("\nDescriptive statistics of review lengths:")
print(df['Review_Length'].describe())
```

```
Exploratory Data Analysis:
Total number of reviews: 1000
Average review length: 10.894

Descriptive statistics of review lengths:
count    1000.000000
mean       10.894000
std         6.257469
min         1.000000
25%         6.000000
50%        10.000000
75%        15.000000
max        32.000000
Name: Review_Length, dtype: float64
```

Distribution of Review Lengths

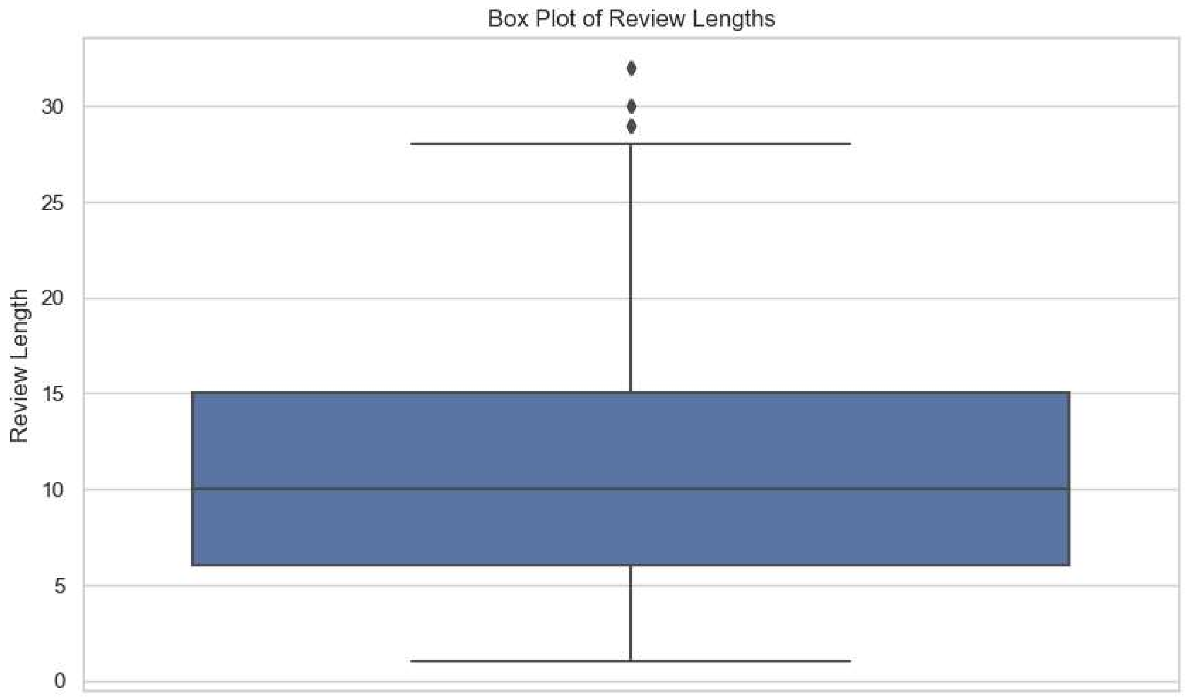
In [3]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Calculate review lengths
df['Review_Length'] = df['Review'].apply(lambda x: len(x.split()))

# Set up seaborn style
```

```
sns.set(style="whitegrid")

# Distribution of review lengths
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='Review_Length', bins=30, kde=True)
plt.title('Distribution of Review Lengths')
plt.xlabel('Review Length')
plt.ylabel('Frequency')
plt.show()
```



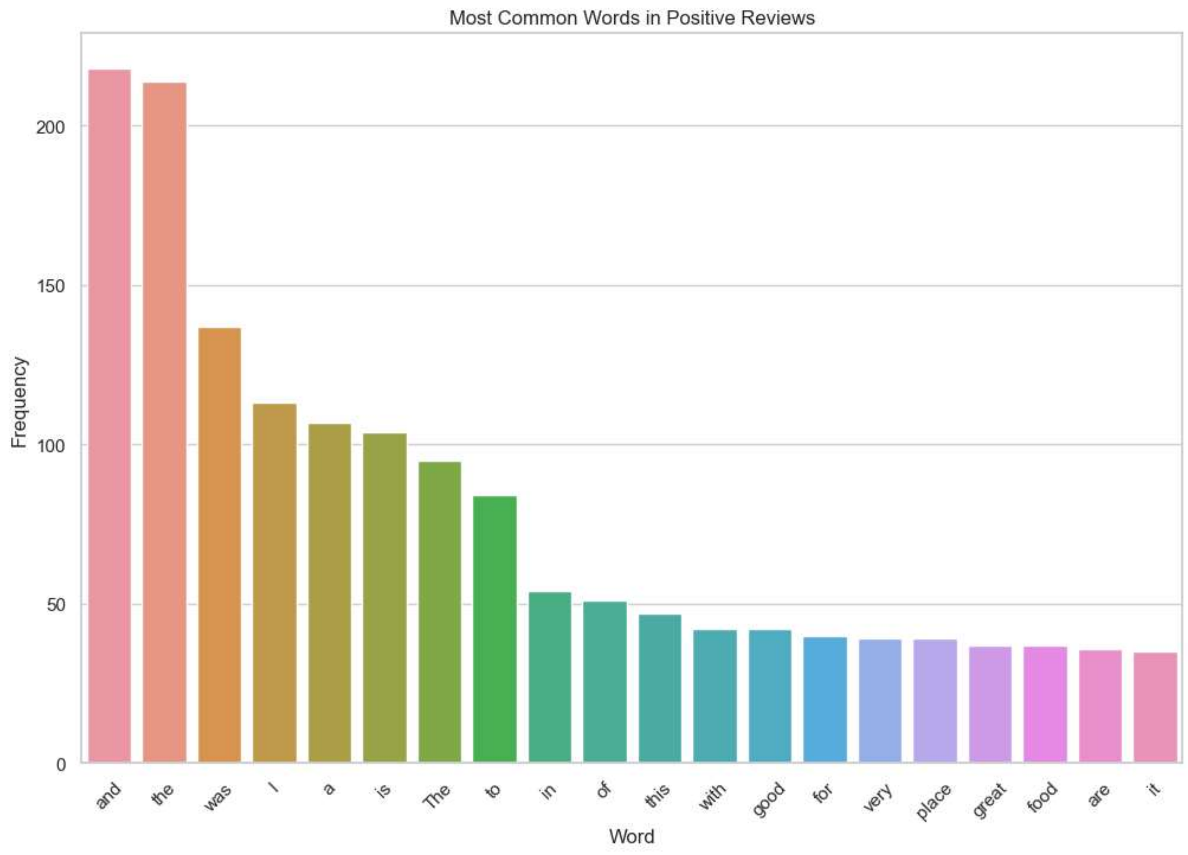
Box Plot of Review Lengths

In [4]:
```
# Box plot of review lengths
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, y='Review_Length')
plt.title('Box Plot of Review Lengths')
plt.ylabel('Review Length')
plt.show()
```



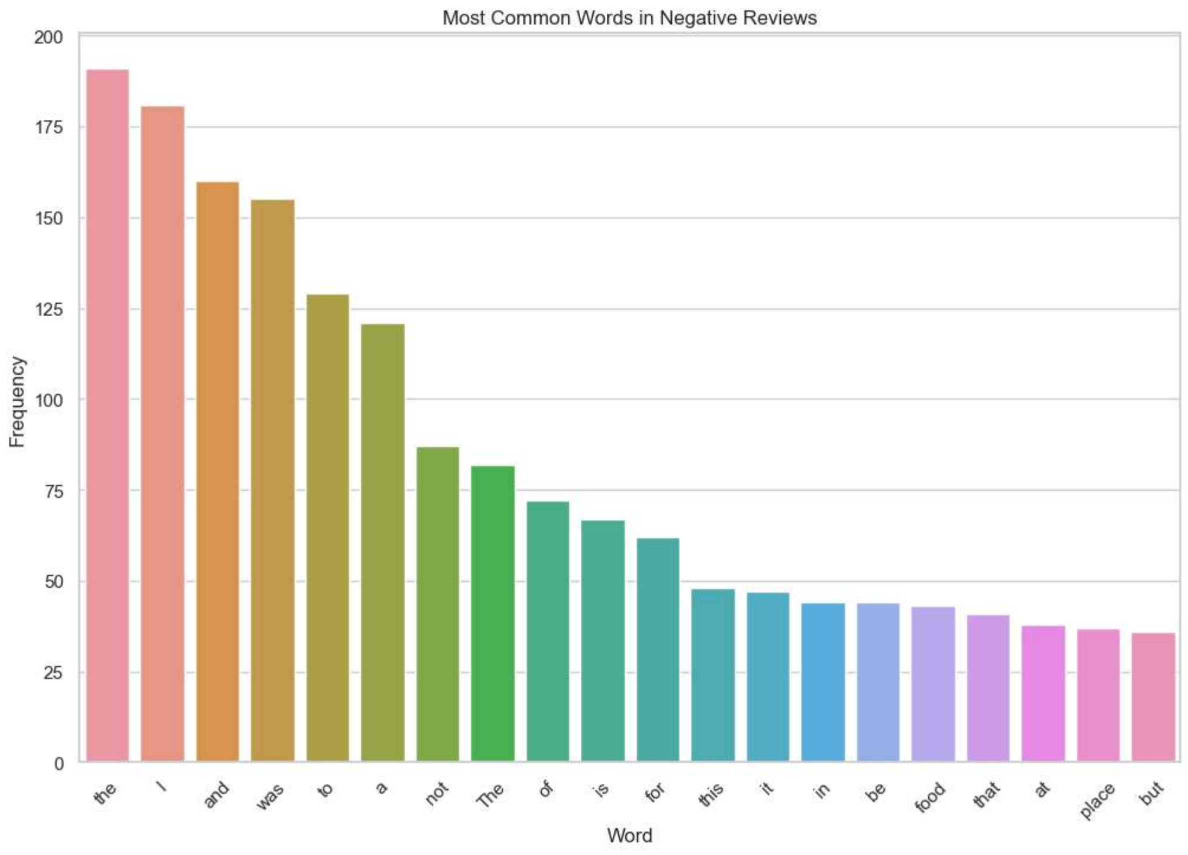
Most Common Words in Positive Reviews

In [5]:
```
from collections import Counter
from wordcloud import WordCloud

# Tokenize the reviews
df['Tokenized_Review'] = df['Review'].apply(lambda x: x.split())

# Separate positive and negative reviews
positive_reviews = df[df['Liked'] == 1]['Tokenized_Review']
negative_reviews = df[df['Liked'] == 0]['Tokenized_Review']

# Calculate word frequency in positive and negative reviews
positive_word_freq = Counter([word for review in positive_reviews for word in revi
negative_word_freq = Counter([word for review in negative_reviews for word in revi

# Set up seaborn style
sns.set(style="whitegrid")

# Plot most common words in positive reviews
plt.figure(figsize=(12, 8))
sns.barplot(x=[word[0] for word in positive_word_freq.most_common(20)],
            y=[word[1] for word in positive_word_freq.most_common(20)])
plt.title('Most Common Words in Positive Reviews')
plt.xlabel('Word')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```



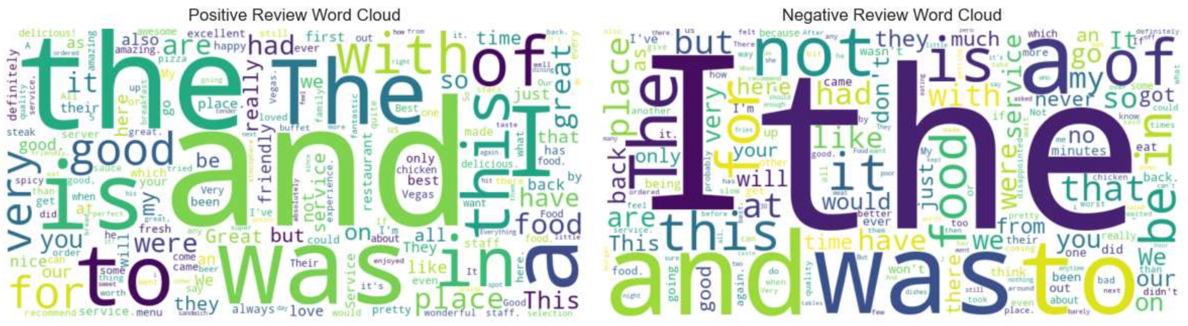
Most Common Words in Negative Reviews

In [6]:
```
# Plot most common words in negative reviews
plt.figure(figsize=(12, 8))
sns.barplot(x=[word[0] for word in negative_word_freq.most_common(20)],
            y=[word[1] for word in negative_word_freq.most_common(20)])
plt.title('Most Common Words in Negative Reviews')
plt.xlabel('Word')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```



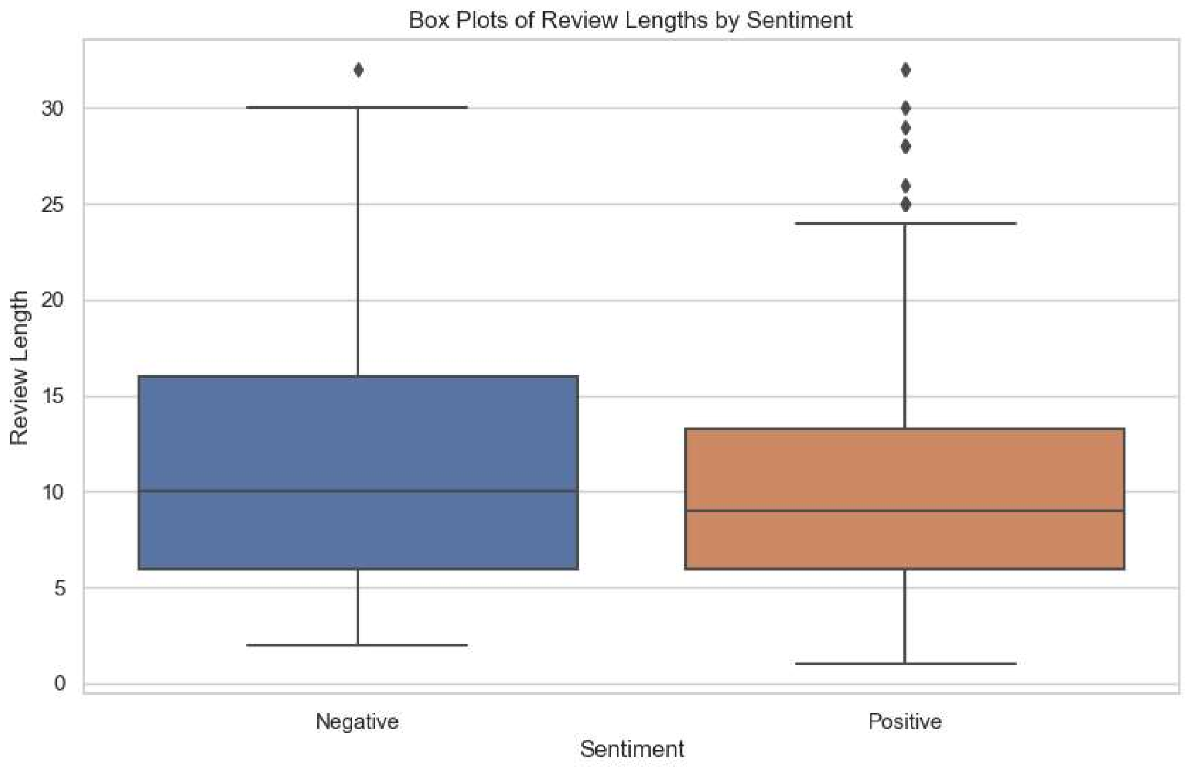
Positive and Negative Review Word Cloud

```
In [7]: # Create word clouds for positive and negative reviews
        positive_wordcloud = WordCloud(width=800, height=400, background_color='white').ge
        negative_wordcloud = WordCloud(width=800, height=400, background_color='white').ge

        # Display word clouds
        plt.figure(figsize=(12, 6))
        plt.subplot(1, 2, 1)
        plt.imshow(positive_wordcloud, interpolation='bilinear')
        plt.title('Positive Review Word Cloud')
        plt.axis('off')

        plt.subplot(1, 2, 2)
        plt.imshow(negative_wordcloud, interpolation='bilinear')
        plt.title('Negative Review Word Cloud')
        plt.axis('off')

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



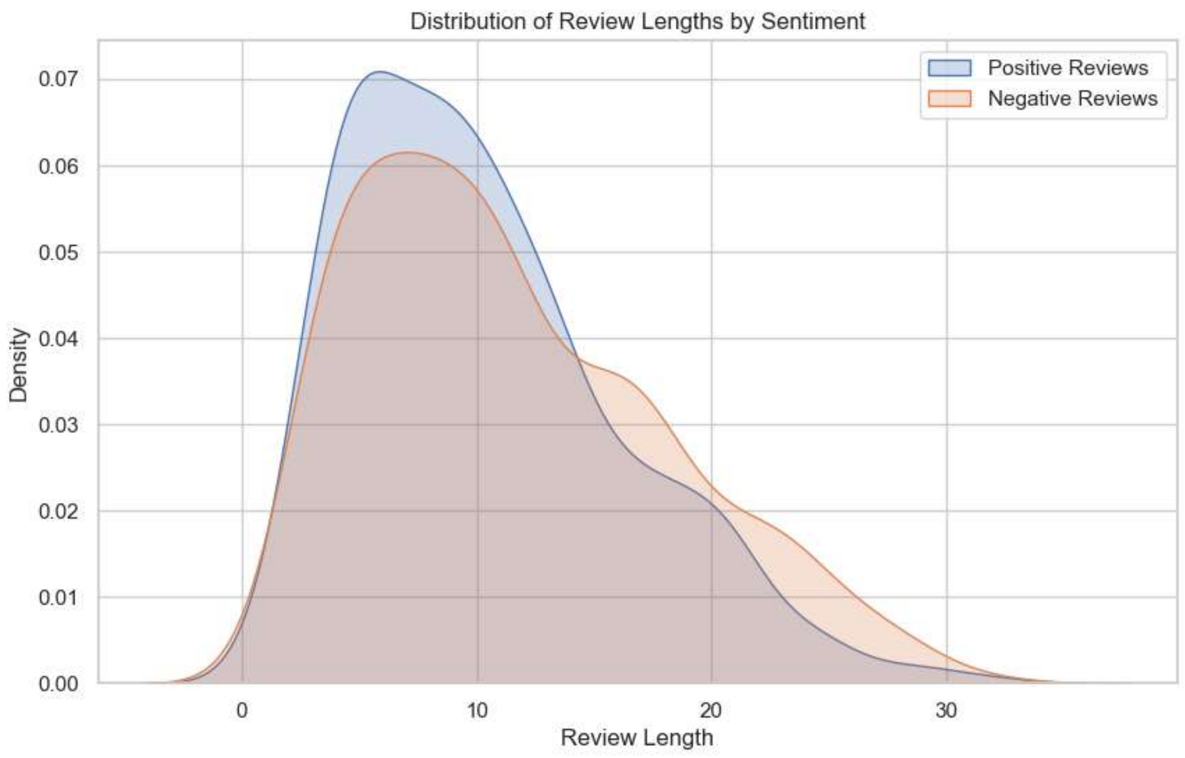
Positive and Negative Box Plots of Review Lengths by Sentiment

In [8]:
```python
# Box plots of review lengths by sentiment
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='Liked', y='Review_Length')
plt.title('Box Plots of Review Lengths by Sentiment')
plt.xlabel('Sentiment')
plt.ylabel('Review Length')
plt.xticks(ticks=[0, 1], labels=['Negative', 'Positive'])
plt.show()
```



Positive and Negative Reviews Lengths by Sentiment

In [9]:
```python
# Set up seaborn style
sns.set(style="whitegrid")

# Plot distribution of review lengths using kernel density estimates
plt.figure(figsize=(10, 6))
sns.kdeplot(data=positive_reviews.str.len(), label='Positive Reviews', shade=True)
sns.kdeplot(data=negative_reviews.str.len(), label='Negative Reviews', shade=True)
plt.title('Distribution of Review Lengths by Sentiment')
plt.xlabel('Review Length')
plt.ylabel('Density')
plt.legend()
plt.show()
```

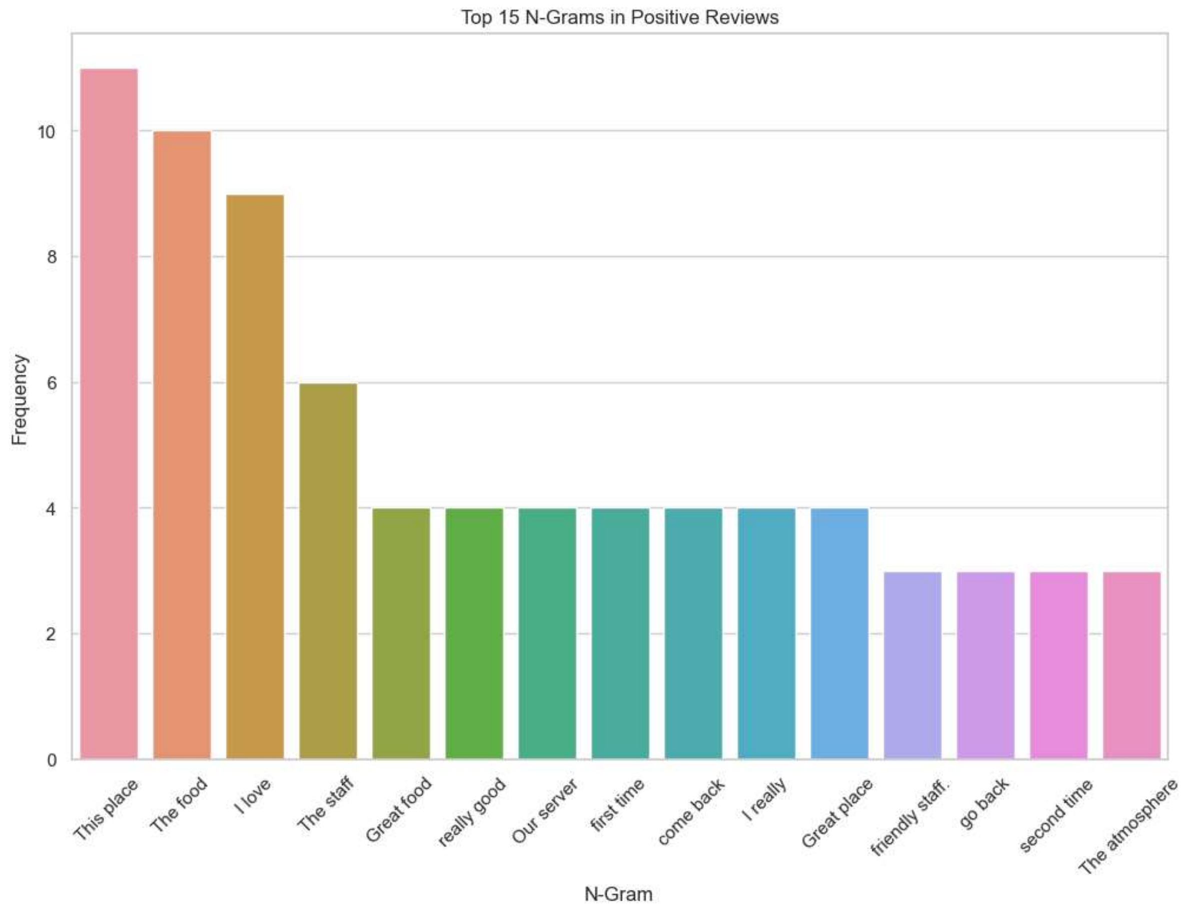


Top 15 N-Grams in Positive Reviews

In [15]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from collections import Counter
from nltk.util import ngrams
import nltk
from nltk.corpus import stopwords
from wordcloud import STOPWORDS

# Download the NLTK stopwords resource
nltk.download('stopwords')

# Read the TSV file
file_path = "Restaurant_Reviews.tsv"
df = pd.read_csv(file_path, delimiter='\t', quoting=3)

# Tokenize the reviews
df['Tokenized_Review'] = df['Review'].apply(lambda x: x.split())

# Separate positive and negative reviews
positive_reviews = df[df['Liked'] == 1]['Tokenized_Review']
negative_reviews = df[df['Liked'] == 0]['Tokenized_Review']

# Combine reviews for n-grams analysis
positive_words = [word for review in positive_reviews for word in review]
negative_words = [word for review in negative_reviews for word in review]

# Function to generate n-grams
def generate_ngrams(words, n):
    return list(ngrams(words, n))

# Generate n-grams for positive and negative reviews
n = 2  # Choose the n-gram size
positive_ngrams = generate_ngrams(positive_words, n)
negative_ngrams = generate_ngrams(negative_words, n)
```

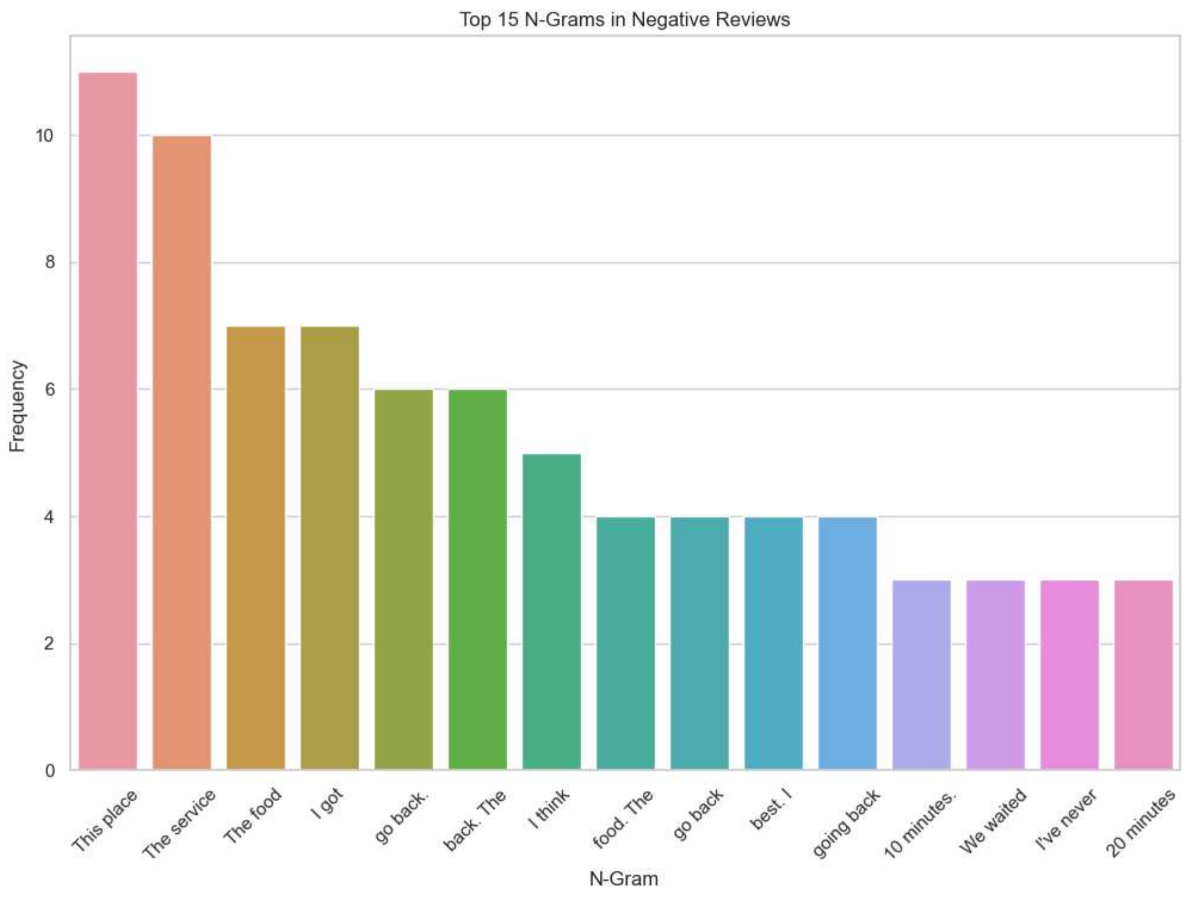
```
# Calculate n-gram frequency
positive_ngram_freq = Counter(positive_ngrams)
negative_ngram_freq = Counter(negative_ngrams)

# Remove common stopwords from n-gram frequency
stop_words = set(STOPWORDS).union(set(stopwords.words('english')))
positive_ngram_freq = {ngram: freq for ngram, freq in positive_ngram_freq.items()
negative_ngram_freq = {ngram: freq for ngram, freq in negative_ngram_freq.items()

# Get the top n-grams
top_n = 15
top_positive_ngrams = Counter(positive_ngram_freq).most_common(top_n)
top_negative_ngrams = Counter(negative_ngram_freq).most_common(top_n)

# Set up seaborn style
sns.set(style="whitegrid")

# Plot top n-grams in positive reviews
plt.figure(figsize=(12, 8))
sns.barplot(x=[" ".join(ngram) for ngram, _ in top_positive_ngrams],
            y=[count for _, count in top_positive_ngrams])
plt.title(f'Top {top_n} N-Grams in Positive Reviews')
plt.xlabel('N-Gram')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```

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



Top 15 N-Grams in Negative Reviews

In [16]:
```python
# Plot top n-grams in negative reviews
plt.figure(figsize=(12, 8))
sns.barplot(x=[" ".join(ngram) for ngram, _ in top_negative_ngrams],
            y=[count for _, count in top_negative_ngrams])
plt.title(f'Top {top_n} N-Grams in Negative Reviews')
plt.xlabel('N-Gram')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```


Top 15 N-Grams in Negative Reviews
Frequency
This place
The service
The food
I got
go back.
back. The
I think
food. The
go back
best. I
going back
10 minutes.
We waited
I've never
20 minutes
N-Gram