In [1]:
```
import requests
from bs4 import BeautifulSoup
import pandas as pd

# URL of the Google search results review page
url = "https://www.google.com/search?q=stranger+things+google+review&rlz=1C1PRFI_enIN977IN977&oq=stranger&aqs
response = requests.get(url)
soup = BeautifulSoup(response.content, "html.parser")
review_snippets = soup.find_all("div", class_="tF2Cxc")
reviews = [snippet.get_text().strip() for snippet in review_snippets]
df = pd.DataFrame({"decription": reviews})
df.to_csv("stranger thing.csv", index=False, encoding="utf-8")
print("Scraping and saving using DataFrame completed.")
```

```
Scraping and saving using DataFrame completed.
```

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive",
force_remount=True).
```

In [4]:
```
path="/content/drive/MyDrive/Classroom/stranger thing.csv"
data=pd.read_csv(path)
```

In [5]:
```
data
```

Out[5]:

	User name	review posted date	description
0	Laila Byles	3 years ago	Wow, I'm shook! 😳 This series is absolutely mi...
1	Rida Haque	3 years ago	Unexpectedly awesome! Goosebumps all over. Gre...
2	Kelly Nichols	4 years ago	Way better than I thought. Goosebumps every ti...
3	chelsea mercer	4 years ago	Chills every episode.
4	ellie boulton	a year ago	Can't even! So darn good. Cheers to the team
...	...	...	...
5094	Natalia Marcos	5 years ago	Stranger Things is a love letter for the 80s. ...
5095	Jamie Broadnax	5 years ago	Winona Ryder delivers one of her best performa...
5096	Morgan Jeffery	5 years ago	We all fell for it, and fast - with even the d...
5097	Nick Venable	5 years ago	Created by Matt and Ross Duffer... Stranger Th...
5098	Jim Vorel	5 years ago	With a stellar cast of child actors and severa...

5099 rows × 3 columns

EDA

In [6]:
```
data.describe()
```

Out[6]:

	User name	review posted date	description
count	5090	5099	5027
unique	5061	31	4707
top	plargreg	a year ago	[Too long description]
freq	4	1465	318

data has almost 5099 data points and 3 columns

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5099 entries, 0 to 5098
Data columns (total 3 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   User name           5090 non-null   object
 1   review posted date  5099 non-null   object
 2   description         5027 non-null   object
dtypes: object(3)
memory usage: 119.6+ KB
```

In [8]:
```
data.isnull().sum()
```

Out[8]:
```
User name              9
review posted date     0
description           72
dtype: int64
```

Data Cleaning

In [9]:
```
data.drop_duplicates(inplace=True)
data.dropna(inplace=True)
```

In [10]:
```
data
```

Out[10]:

	User name	review posted date	description
0	Laila Byles	3 years ago	Wow, I'm shook! 😲 This series is absolutely mi...
1	Rida Haque	3 years ago	Unexpectedly awesome! Goosebumps all over. Gre...
2	Kelly Nichols	4 years ago	Way better than I thought. Goosebumps every ti...
3	chelsea mercer	4 years ago	Chills every episode.
4	ellie boulton	a year ago	Can't even! So darn good. Cheers to the team
...	...	...	...
5094	Natalia Marcos	5 years ago	Stranger Things is a love letter for the 80s. ...
5095	Jamie Broadnax	5 years ago	Winona Ryder delivers one of her best performa...
5096	Morgan Jeffery	5 years ago	We all fell for it, and fast - with even the d...
5097	Nick Venable	5 years ago	Created by Matt and Ross Duffer... Stranger Th...
5098	Jim Vorel	5 years ago	With a stellar cast of child actors and severa...

5017 rows × 3 columns

After cleaning we have around 5017 rows.

In [11]:
```
#removing irrelevant data like emojis
import re
import pandas as pd

def remove_emojis(text):
    emoji_pattern = re.compile("["
                               u"\U0001F600-\U0001F64F"
                               u"\U0001F300-\U0001F5FF"
                               u"\U0001F680-\U0001F6FF"
                               u"\U0001F700-\U0001F77F"
                               u"\U0001F780-\U0001F7FF"
                               u"\U0001F800-\U0001F8FF"
                               u"\U0001F900-\U0001F9FF"
                               u"\U0001FA00-\U0001FA6F"
                               u"\U0001FA70-\U0001FAFF"
                               u"\U00002702-\U000027B0"
                               u"\U000024C2-\U0001F251"
                               "]+", flags=re.UNICODE)
    return emoji_pattern.sub(r'', text)

data['description'] = data['description'].apply(remove_emojis)

data
```

Out[11]:

	User name	review posted date	description
0	Laila Byles	3 years ago	Wow, I'm shook! This series is absolutely min...
1	Rida Haque	3 years ago	Unexpectedly awesome! Goosebumps all over. Gre...
2	Kelly Nichols	4 years ago	Way better than I thought. Goosebumps every ti...
3	chelsea mercer	4 years ago	Chills every episode.
4	ellie boulton	a year ago	Can't even! So darn good. Cheers to the team
...	...	...	...
5094	Natalia Marcos	5 years ago	Stranger Things is a love letter for the 80s. ...
5095	Jamie Broadnax	5 years ago	Winona Ryder delivers one of her best performa...
5096	Morgan Jeffery	5 years ago	We all fell for it, and fast - with even the d...
5097	Nick Venable	5 years ago	Created by Matt and Ross Duffer... Stranger Th...
5098	Jim Vorel	5 years ago	With a stellar cast of child actors and severa...

5017 rows × 3 columns

Instead of using phrases like '3 years ago,' I am extracting the exact year. In this context, we have extracted the year from the 'Review Posted Date' column.

In [12]:
```
import re
import pandas as pd

def extract_year_month(text):
    years_ago = re.findall(r'(\d+) years? ago', text)
    months_ago = re.findall(r'(\d+) months ago', text)

    if years_ago:
        years_ago = int(years_ago[0])
        current_year = pd.Timestamp.now().year
        year = current_year - years_ago
        return year
    elif months_ago:
        months_ago = int(months_ago[0])
        current_year = pd.Timestamp.now().year
        current_month = pd.Timestamp.now().month
        year = 2022 if months_ago >= current_month else current_year
        return year
    elif "a year ago" in text:
        return pd.Timestamp.now().year - 1
    else:
        return None

data['year'] = data['review posted date'].apply(extract_year_month)
```

In [13]:
```
data
```

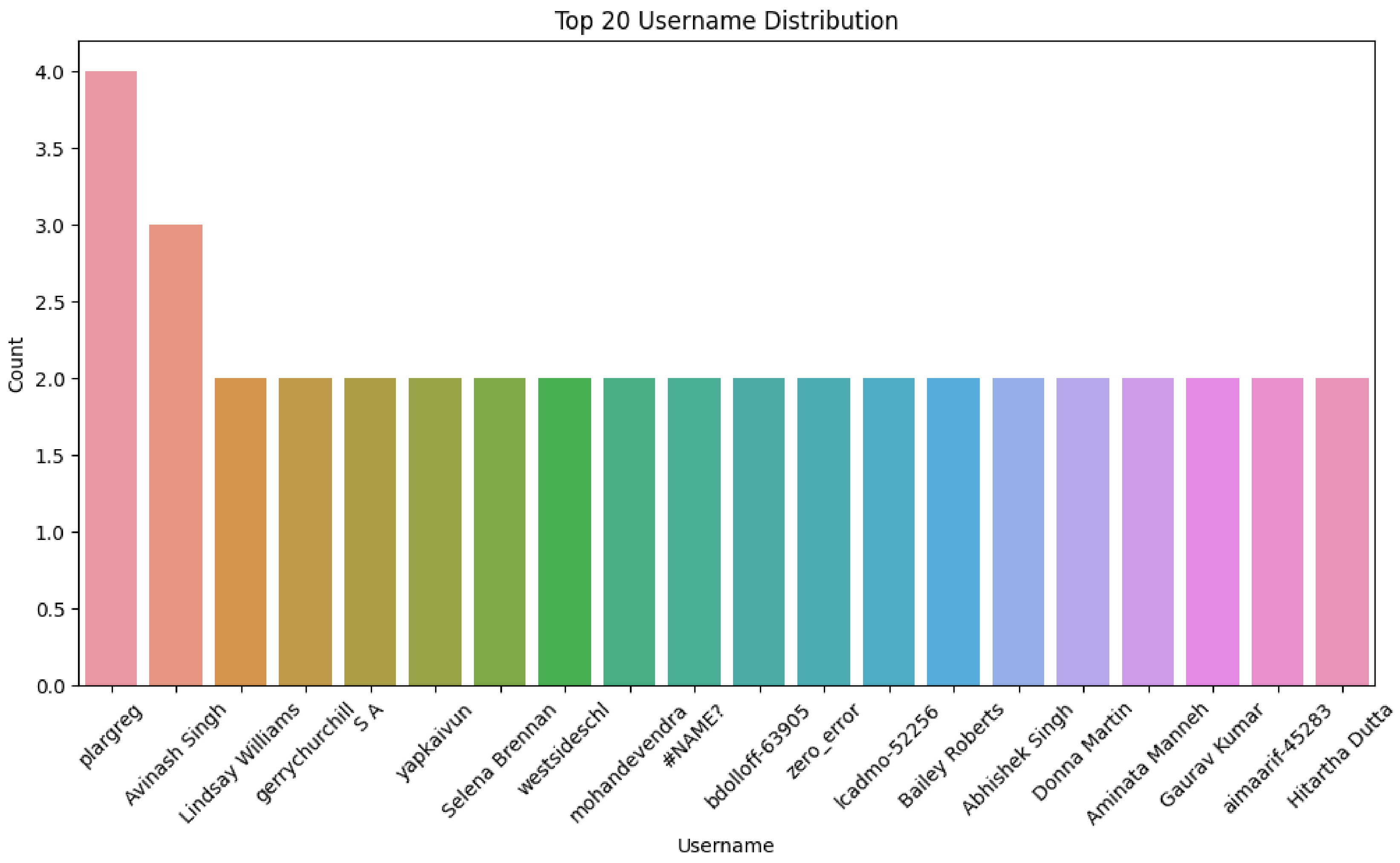
Out[13]:

	User name	review posted date	description	year
0	Laila Byles	3 years ago	Wow, I'm shook! This series is absolutely min...	2020.0
1	Rida Haque	3 years ago	Unexpectedly awesome! Goosebumps all over. Gre...	2020.0
2	Kelly Nichols	4 years ago	Way better than I thought. Goosebumps every ti...	2019.0
3	chelsea mercer	4 years ago	Chills every episode.	2019.0
4	ellie boulton	a year ago	Can't even! So darn good. Cheers to the team	2022.0
...	...	...	...	...
5094	Natalia Marcos	5 years ago	Stranger Things is a love letter for the 80s. ...	2018.0
5095	Jamie Broadnax	5 years ago	Winona Ryder delivers one of her best performa...	2018.0
5096	Morgan Jeffery	5 years ago	We all fell for it, and fast - with even the d...	2018.0
5097	Nick Venable	5 years ago	Created by Matt and Ross Duffer... Stranger Th...	2018.0
5098	Jim Vorel	5 years ago	With a stellar cast of child actors and severa...	2018.0

5017 rows × 4 columns

In [14]:
```
#renaming the columns
data.rename(columns={'User name': 'username'}, inplace=True)
```

In [15]:
```python
# Univariate Analysis - Improved Username Distribution
plt.figure(figsize=(12, 6))
top_users = data['username'].value_counts().head(20)  # Choose top 20 usernames
sns.barplot(x=top_users.index, y=top_users.values)
plt.xticks(rotation=45)
plt.title('Top 20 Username Distribution')
plt.xlabel('Username')
plt.ylabel('Count')
plt.show()
```

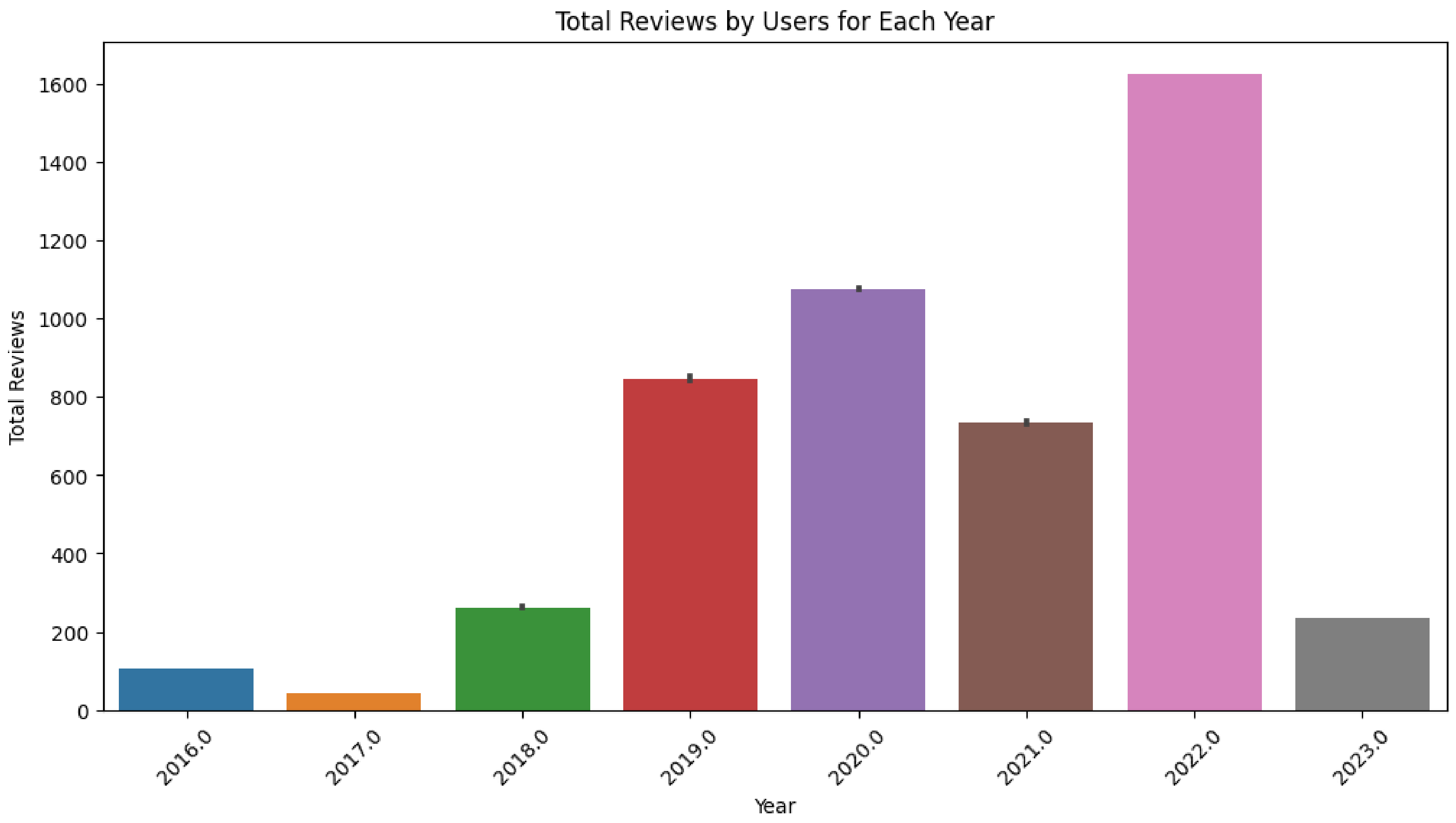


In [16]:
```
target_username = "plargreg"
user_data = data[data['username'] == target_username]
comment_count = len(user_data)
print(f"{target_username} has commented on reviews {comment_count} times.")
```

```
plargreg has commented on reviews 4 times.
```

We can see that the username 'plargreg' has commented on Google reviews for 'Stranger Things' TV show 4 times.

In [17]:
```python
# Group data by 'username' and 'year' and count the number of reviews per year
user_year_counts = data.groupby(['username', 'year'])['description'].count().reset_index()
plt.figure(figsize=(12, 6))
sns.barplot(data=user_year_counts, x='year', y='description', estimator=sum)
plt.title('Total Reviews by Users for Each Year')
plt.xlabel('Year')
plt.ylabel('Total Reviews')
plt.xticks(rotation=45)
plt.show()
```



Around the year 2022, a significant number of reviews were given for the TV show 'Stranger Things'

To check whether the movie/TV series has received a positive/negative or neutral response from the online community.

In [18]:
```
!pip install textblob pandas
```

```
Requirement already satisfied: textblob in /usr/local/lib/python3.10/dist-packages (0.17.1)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (1.5.3)
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.10/dist-packages (from textblob) (3.8.1)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/python3.10/dist-packages (from panda
s) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.
3)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas) (1.23.
5)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(8.1.6)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->t
extblob) (2023.6.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob)
(4.66.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.
8.1->pandas) (1.16.0)
```

In [19]:
```
from textblob import TextBlob
import pandas as pd


# Perform sentiment analysis using TextBlob
data['sentiment'] = data['description'].apply(lambda x: TextBlob(x).sentiment.polarity)

#stats calculation
positive_count = len(data[data['sentiment'] > 0])
negative_count = len(data[data['sentiment'] < 0])
neutral_count = len(data[data['sentiment'] == 0])
total_comments = len(data)

#percentage calculation
positive_percentage = (positive_count / total_comments) * 100
negative_percentage = (negative_count / total_comments) * 100
neutral_percentage = (neutral_count / total_comments) * 100

# Highlight insights
print(f"Positive Comments: {positive_percentage:.2f}%")
print(f"Negative Comments: {negative_percentage:.2f}%")
print(f"Neutral Comments: {neutral_percentage:.2f}%")
```

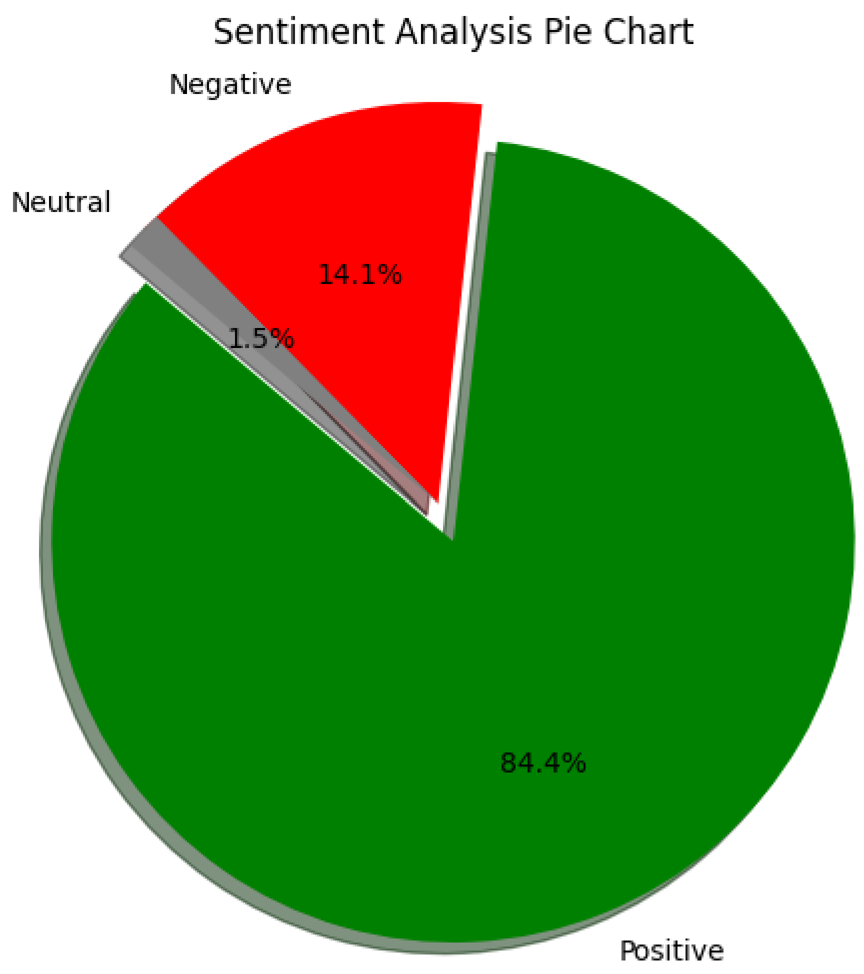
```
Positive Comments: 84.37%
Negative Comments: 14.11%
Neutral Comments: 1.51%
```

In [20]:
```
if positive_percentage > negative_percentage:
    print("The movie/TV series has received a positive response from the online community.")
elif negative_percentage > positive_percentage:
    print("The movie/TV series has received a negative response from the online community.")
else:
    print("The movie/TV series has received a neutral response from the online community.")
```

```
The movie/TV series has received a positive response from the online community.
```

The online community has responded positively to the movie/TV series.

In [21]:
```
# Create a pie chart
labels = ['Positive', 'Negative', 'Neutral']
sizes = [positive_percentage, negative_percentage, neutral_percentage]
colors = ['green', 'red', 'gray']
explode = (0.1, 0, 0)  # Explode the 1st slice (Positive)

plt.figure(figsize=(8, 6))
plt.pie(sizes, explode=explode, labels=labels, colors=colors,
        autopct='%1.1f%%', shadow=True, startangle=140)
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plt.title('Sentiment Analysis Pie Chart')
plt.show()
```



Most common words

```
In [22]: !pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.6)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2023.
6.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.1)
```

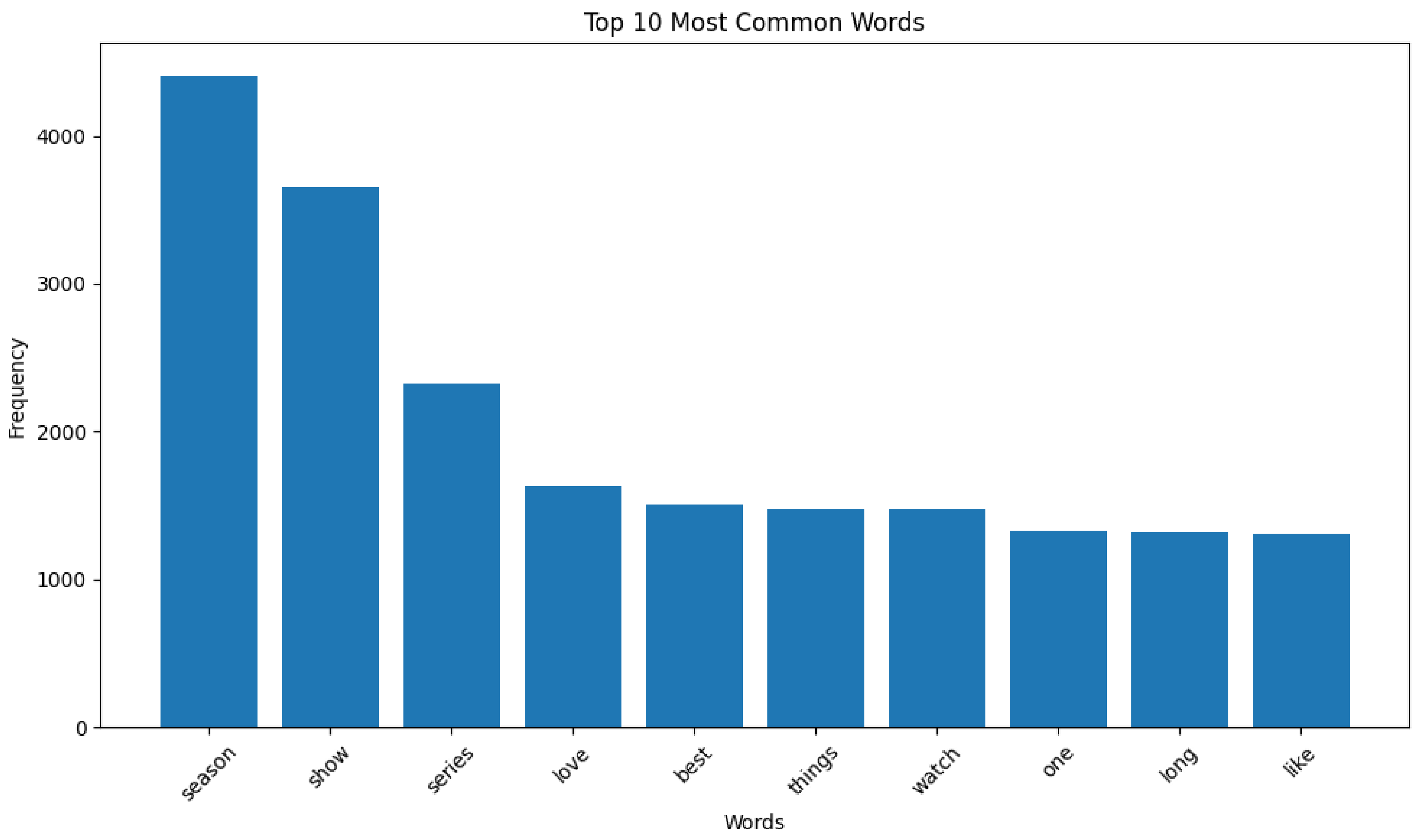
In [23]:
```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.tokenize import RegexpTokenizer
from collections import Counter
```

In [24]:
```
import nltk

# Download stopwords and punkt data
nltk.download('stopwords')
nltk.download('punkt')
```

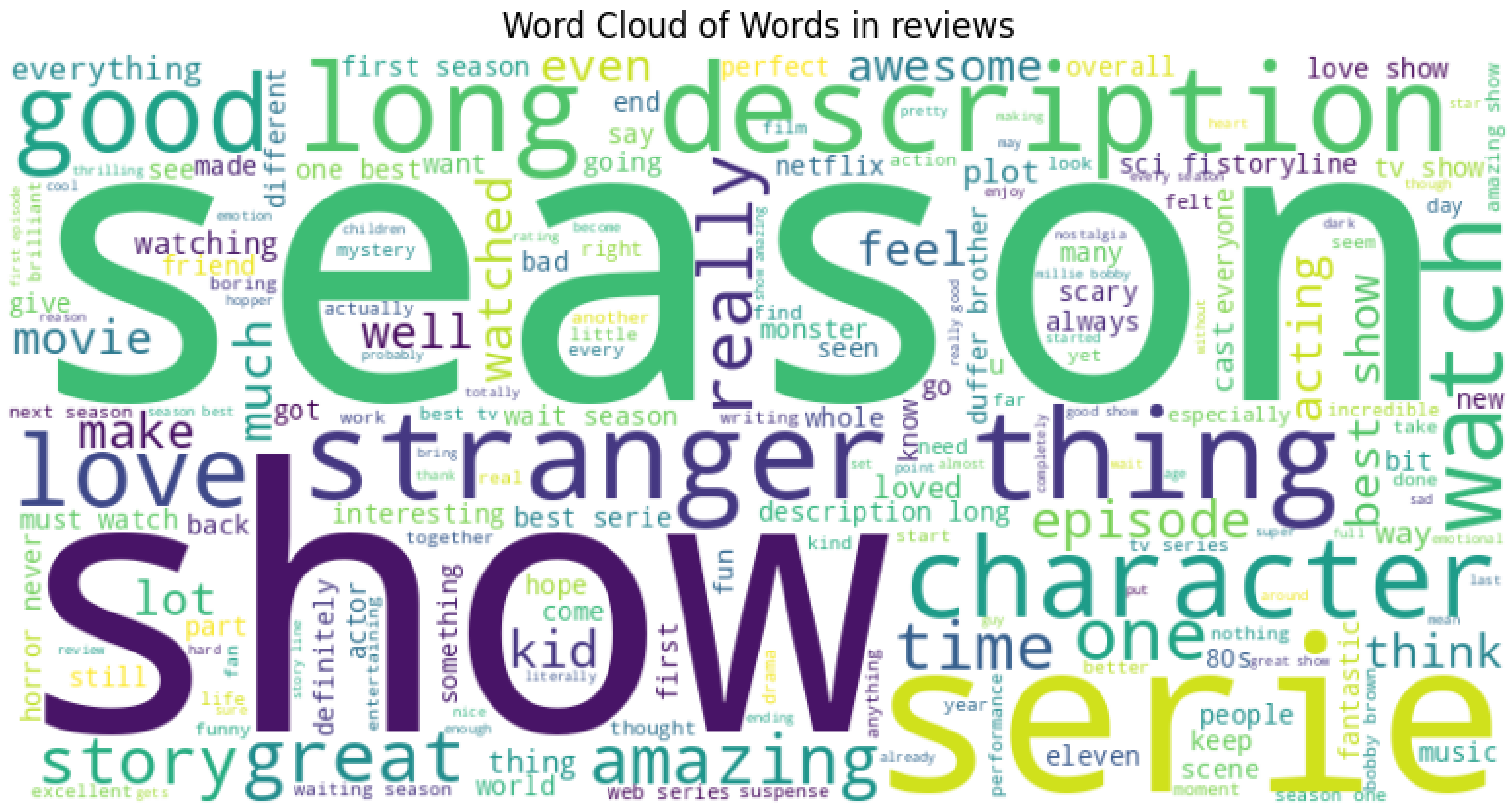
```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[24]:
```
True
```

In [25]:
```
# Get NLTK English stopwords
nltk_stopwords = set(stopwords.words('english'))

# Initialize tokenizer to remove punctuation
tokenizer = RegexpTokenizer(r'\w+')

filtered_words = []
for text in data['description']:
    words = tokenizer.tokenize(text.lower())
    filtered_words.extend([word for word in words if word not in nltk_stopwords])

word_counter = Counter(filtered_words) # Count word occurrences

top_common_words = word_counter.most_common(10)
top_words, top_counts = zip(*top_common_words)

plt.figure(figsize=(10, 6))
plt.bar(top_words, top_counts)
plt.xlabel('Words')
plt.ylabel('Frequency')
plt.title('Top 10 Most Common Words')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```


Top 10 Most Common Words
Frequency
0
1000
2000
3000
4000
season
show
series
love
best
things
watch
one
long
like
Words

In [26]:
```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

filtered_text = ' '.join(filtered_words)

wordcloud = WordCloud(width=800, height=400, background_color='white').generate(filtered_text)

plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Words in reviews')
plt.show()
```


Word Cloud of Words in reviews
season
show
serie
stranger
thing
long
description
good
love
watch
character
story
great
amazing
really
time
one
episode
watched
much
acting
make
movie
everything
first season
even
perfect
awesome
overall
love show
feel
well
watching
friend
kid
lot
hope
come
think
people
keep
eleven
scene
music
netflix
plot
storyline
sci fi
tv show
bad
right
scary
always
many
monster
seen
duffer brother
cast
everyone
fun
first
definitely
actor
something
still
part
horror
never
must watch
back
interesting
best serie
wait season
whole
loved
description long
thought
world
web series suspense
waiting season
excellent
next season
bit
way
new
incredible
especially
fantastic
bobby brown
performance
year
80s
nothing
better
tv series
kind
start
together
need
know
go
far
u
yet
another
little
every
find
day
felt
give
made
different
see
one best
want
going
say
end
film
action
look
plot
day
mystery
actually
boring
life
funny