```
import pandas as pd

df = pd.read_csv("play (1).csv")
```

```
df.head(2)
```

	h3YV2d
0	Update: app support was very responsive, and t...
1	This application is perfect for what it's inte...

```
new_column_names = ['Review']

df.columns = new_column_names
```

```
df.head(2)
```

	Review
0	Update: app support was very responsive, and t...
1	This application is perfect for what it's inte...

```
df.columns
```

```
Index(['Review'], dtype='object')
```

```
print(df.head())
```

```
                                              Review
0  Update: app support was very responsive, and t...
1  This application is perfect for what it's inte...
2  Due to the recent updates, the ad section of t...
3  The app is simply brilliant. 5 stars. I would ...
4  When I first installed this app, about 3 years...
```

```
print(df.tail())
```

```
                                               Review
95  Please remove the Goal Distribution Table and ...
96  Please, put the starting time (of finished mat...
97  Nice app when given time , it adds value to yo...
98  Please why did you remove the scores on basket...
99  I love this app completely. Only wish I could ...
```

```
from textblob import TextBlob
from collections import Counter
import re
```

```
def clean_text(text):
    if isinstance(text, str):
        # Remove newlines and extra whitespaces
        text = re.sub(r'\s+', ' ', text)
        return text
    else:
        return ''

# Function to perform sentiment analysis
def get_sentiment(text):
    analysis = TextBlob(text)
    # Return polarity as sentiment
    return analysis.sentiment.polarity
```

```
# Clean text
df['Cleaned_Review'] = df['Review'].apply(clean_text)

# Perform sentiment analysis
df['Sentiment'] = df['Cleaned_Review'].apply(get_sentiment)

# Identify negative reviews
negative_reviews_df = df[df['Sentiment'] < 0]

# Issues identified based on reviews (could be extended)
issues = {
    'Theatre': ['screen','movie','sound','theatre'],
    'Food Court': ['food court', 'food'],
    'Cleanliness': ['maintained', 'cleanliness','pathetic'],
}
```

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Provided reviews dataset
reviews = negative_reviews_df.apply(str).tolist()

# Combine all reviews into a single string
text = ' '.join(reviews)

# Generate word cloud
wordcloud = WordCloud(width=800, height=400, background_color='white').generate(text)

# Display the word cloud
plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```


```
# Function to identify issues
def identify_issues(review, issues):
    identified_issues = []
    for issue, keywords in issues.items():
        for keyword in keywords:
            if keyword in review.lower():
                identified_issues.append(issue)
                break
    return identified_issues
```

```
# Apply issue identification
df['Identified_Issues'] = df['Cleaned_Review'].apply(lambda x: identify_issues(x, issues))

# Count frequency of each issue
issue_counter = Counter([issue for sublist in df['Identified_Issues'] for issue in sublist])

# Assign priority based on frequency
priority_list = {issue: index + 1 for index, (issue, _) in enumerate(issue_counter.most_common())}

print("Priority list of issues based on frequency:")
for issue, priority in priority_list.items():
    print(f"{issue}: Priority {priority}")
```

```
Priority list of issues based on frequency:
Theatre: Priority 1
Cleanliness: Priority 2
```

```
# Count number of negative reviews for each issue
print("\nNumber of negative reviews for each issue:")
for issue, count in issue_counter.items():
    print(f"{issue}: {count}")
```

```
Number of negative reviews for each issue:
Theatre: 11
Cleanliness: 1
```