Exploratory Data Analysis

Firstly, let's install all tools and libraries to start our analysis. We would also need to mount the Google Drive and load the dataset that was prepared in our earlier stage.

In [ ]:
```
import pandas as pd
import seaborn as sns
import numpy as np
```

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

```
Mounted at /content/gdrive
```

While loading data, we will firstly load it into two separate datasets based on the song's popularity. We will also combine all songs in one combined dataset that we will use for some parts of our analysis.

In [ ]:
```
popular=pd.read_csv('/content/gdrive/MyDrive/Capstone/popular_with_yt.csv')
popular['Popular']=1
popular
```

Out[ ]:

	Title	Artist	Popular	danceability	energy	key	loudness	mode	speechiness	acousticness	...	
0	Bad Day	Daniel Powter	1	0.599	0.785	3.0	-4.013	1.0	0.0309	0.44800	...	https://api.spotify.com/v1/tra
1	Temperature	Sean Paul	1	0.951	0.600	0.0	-4.675	0.0	0.0685	0.10600	...	https://api.spotify.com/v1/tra
2	Promiscuous	Nelly Furtado Featuring Timbaland	1	0.808	0.970	10.0	-6.098	0.0	0.0506	0.05690	...	https://api.spotify.com/v1/tra
3	You're Beautiful	James Blunt	1	0.675	0.479	0.0	-9.870	0.0	0.0278	0.63300	...	https://api.spotify.com/v1/trac
4	Hips Don't Lie	Shakira Featuring Wyclef Jean	1	0.778	0.824	10.0	-5.892	0.0	0.0707	0.28400	...	https://api.spotify.com/v1/tr
...	...	...	...	...	...	...	...	...	...	...	...	
1408	Flower Shops	ERNEST Featuring Morgan Wallen	1	0.527	0.461	7.0	-5.908	1.0	0.0269	0.11800	...	https://api.spotify.com/v1/t
1409	To The Moon!	JNR CHOI & Sam Tompkins	1	0.745	0.650	2.0	-11.814	1.0	0.3460	0.04510	...	https://api.spotify.com/v1/t
1410	Unholy	Sam Smith & Kim Petras	1	0.714	0.472	2.0	-7.375	1.0	0.0864	0.01300	...	https://api.spotify.com/v1/trac
1411	One Mississippi	Kane Brown	1	0.471	0.846	0.0	-5.269	1.0	0.0389	0.00279	...	https://api.spotify.com/v1/tra
1412	Circles Around This Town	Maren Morris	1	0.591	0.814	4.0	-4.986	1.0	0.0468	0.01500	...	https://api.spotify.com/v1/tra

1413 rows × 27 columns

In [ ]:

Out[ ]:
```
Title                     0
Artist                    0
Popular                   0
danceability              1
energy                    1
key                       1
loudness                  1
mode                      1
speechiness               1
acousticness              1
instrumentalness          1
liveness                  1
valence                   1
tempo                     1
type                      1
id                        1
uri                       1
track_href                1
analysis_url              1
duration_ms               1
time_signature            1
track_id                  1
index                  1137
video_id                  0
youtube_view_count        0
youtube_like_count        0
youtube_comment_count     0
dtype: int64
```

In [ ]:
```
unpopular=pd.read_csv('/content/gdrive/MyDrive/Capstone/unpopular_with_yt.csv')
unpopular['Popular']=0
unpopular
```

Out[ ]:

	Title	Artist	track_id	Popular	danceability	energy	key	loudness	mode	speechiness	...	
0	This Is How We Do It	Montell Jordan	6uQKuonTU8VKBz5SHZuQXD	0	0.799	0.6230	0.0	-9.374	1.0	0.0812	...	6uQKuonTU8
1	Biking	Frank Ocean	2q0VexHJirnUPnEOhr2DxK	0	0.673	0.4630	2.0	-7.247	1.0	0.1910	...	2q0VexHJi
2	Nuyorican Soul	Carlos Henriquez	0ZDBZpIt3eUJ6LaApLJSiB	0	0.548	0.6050	5.0	-10.184	1.0	0.0428	...	0ZDBZpIt3
3	L'eau	Mindeliq	5Jk9uO3DbBAoa6JFXpPcYJ	0	0.838	0.3610	11.0	-11.513	0.0	0.2510	...	5Jk9uO3Db
4	Seduce (Feat. Capella Grey & Tamae)	Russ	5irzewJoHob42Nq3P9kOYh	0	0.825	0.3660	1.0	-6.973	0.0	0.0614	...	5irzewJoH
...	...	...	...	...	...	...	...	...	...	...	...	
1697	Sundress	A$AP Rocky	2aPTvyE09vUCRwVvj0I8WK	0	0.721	0.7070	6.0	-6.364	1.0	0.0595	...	2aPTvyE09
1698	Agora Hills	Doja Cat	7dJYggqjKo71KI9sLzqCs8	0	0.750	0.6740	8.0	-6.128	0.0	0.0970	...	7dJYggqj
1699	gfg	Miguel	7xK1qc3jSIIo0UHah9WHdn	0	0.673	0.8970	2.0	-4.421	1.0	0.1300	...	7xK1qc3jS
1700	UNA NOCHE EN MEDELLÍN - REMIX	KAROL G	6ejks4eS7DOoYW8hrpRcDV	0	0.843	0.7640	10.0	-2.494	0.0	0.0741	...	6ejks4eS7D
1701	Light White Noise	Evomin	3qBMFORCRUKzBPiftvwaJn	0	0.222	0.0507	1.0	-44.323	1.0	0.0681	...	3qBMFORC

1702 rows × 26 columns

In [ ]:
```
df=pd.concat([unpopular, popular])
```

In [ ]:
```
df=df[['Title', 'Artist', 'track_id', 'Popular', 'danceability', 'energy',
       'key', 'loudness', 'mode', 'speechiness', 'acousticness',
       'instrumentalness', 'liveness', 'valence', 'tempo', 'duration_ms', 'time_signature', 'youtube_view_count',
       'youtube_comment_count']]
df
```

Out[ ]:

	Title	Artist	track_id	Popular	danceability	energy	key	loudness	mode	speechiness	acousticness	in
0	This Is How We Do It	Montell Jordan	6uQKuonTU8VKBz5SHZuQXD	0	0.799	0.623	0.0	-9.374	1.0	0.0812	0.01410	
1	Biking	Frank Ocean	2q0VexHJirnUPnEOhr2DxK	0	0.673	0.463	2.0	-7.247	1.0	0.1910	0.68100	
2	Nuyorican Soul	Carlos Henriquez	0ZDBZpIt3eUJ6LaApLJSiB	0	0.548	0.605	5.0	-10.184	1.0	0.0428	0.70800	
3	L'eau	Mindeliq	5Jk9uO3DbBAoa6JFXpPcYJ	0	0.838	0.361	11.0	-11.513	0.0	0.2510	0.78800	
4	Seduce (Feat. Capella Grey & Tamae)	Russ	5irzewJoHob42Nq3P9kOYh	0	0.825	0.366	1.0	-6.973	0.0	0.0614	0.62400	
...	...	...	...	...	...	...	...	...	...	...	...	
1408	Flower Shops	ERNEST Featuring Morgan Wallen	0De9jFjJ4eRLI7Yww2eBw1	1	0.527	0.461	7.0	-5.908	1.0	0.0269	0.11800	
1409	To The Moon!	JNR CHOI & Sam Tompkins	5vUnjhBzRJJIAOJPde6zDx	1	0.745	0.650	2.0	-11.814	1.0	0.3460	0.04510	
1410	Unholy	Sam Smith & Kim Petras	3nqQXoyQOWXiESFLlDF1hG	1	0.714	0.472	2.0	-7.375	1.0	0.0864	0.01300	
1411	One Mississippi	Kane Brown	4FdPnT2cFrpWCmWZd7GXc3	1	0.471	0.846	0.0	-5.269	1.0	0.0389	0.00279	
1412	Circles Around This Town	Maren Morris	13G5xv1wUKvJYbK0wYmioN	1	0.591	0.814	4.0	-4.986	1.0	0.0468	0.01500	

3115 rows × 20 columns

In [ ]:
```
df=df.reset_index(drop=True)
```

In [ ]:
```
df.isnull().sum()
```

Out[ ]:
```
Title                   0
Artist                  0
track_id                1
Popular                 0
danceability            1
energy                  1
key                     1
loudness                1
mode                    1
speechiness             1
acousticness            1
instrumentalness        1
liveness                1
valence                 1
tempo                   1
duration_ms             1
time_signature          1
youtube_view_count      3
youtube_like_count      3
youtube_comment_count   3
dtype: int64
```

1. Descriptive statistics

First, we want to extract general insights from our data to identify some patterns

In [ ]:
```
# print(popular.describe())
df_=pd.DataFrame(popular.describe())
df_
```

Out[ ]:

	Popular	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness
count	1413.0	1412.000000	1412.000000	1412.000000	1412.000000	1412.000000	1412.000000	1412.000000	1412.000000	1412.000000
mean	1.0	0.659579	0.668345	5.259207	-5.963739	0.639518	0.101900	0.166936	0.013682	0.176221
std	0.0	0.137929	0.166497	3.630051	2.153023	0.480310	0.098693	0.211061	0.096987	0.129717
min	1.0	0.162000	0.040000	0.000000	-18.071000	0.000000	0.023100	0.000053	0.000000	0.021000
25%	1.0	0.566000	0.560000	2.000000	-7.045000	0.000000	0.038975	0.018475	0.000000	0.094600
50%	1.0	0.671000	0.683000	5.000000	-5.624500	1.000000	0.056950	0.079600	0.000000	0.125000
75%	1.0	0.755000	0.797000	8.000000	-4.500750	1.000000	0.123000	0.236000	0.000010	0.221000
max	1.0	0.975000	0.983000	11.000000	-1.190000	1.000000	0.730000	0.981000	0.945000	0.851000

In [ ]:
```
df1=pd.DataFrame(unpopular.describe())
df1
```

Out[ ]:

	Popular	danceability	energy	key	loudness	mode	speechiness	acousticness	instrumentalness	liveness
count	1702.0	1702.000000	1702.000000	1702.000000	1702.000000	1702.000000	1702.000000	1702.000000	1702.000000	1702.000000
mean	0.0	0.620658	0.598183	5.222092	-9.180958	0.586957	0.118866	0.300418	0.172322	0.179867
std	0.0	0.189127	0.238651	3.646816	6.565467	0.492525	0.119633	0.311328	0.326295	0.142915
min	0.0	0.000000	0.000288	0.000000	-45.434000	0.000000	0.000000	0.000000	0.000000	0.026700
25%	0.0	0.509000	0.454000	2.000000	-10.448500	0.000000	0.040600	0.033925	0.000000	0.096725
50%	0.0	0.651000	0.629000	5.000000	-7.311000	1.000000	0.063000	0.180000	0.000027	0.121000
75%	0.0	0.764000	0.777000	8.000000	-5.402500	1.000000	0.151750	0.498750	0.086725	0.213000
max	0.0	0.982000	1.000000	11.000000	0.326000	1.000000	0.936000	0.996000	0.999000	0.970000

Let us analyse this a little bit of the mean values feature-by-feature:

1. **Danceability:**
   - Popular songs (average danceability of 0.660) are often designed to be more dance-friendly and groovy, with catchier rhythms and beats. This can make them more appealing for listeners who want to move and dance to the music.
   - Unpopular songs (average danceability of 0.621) might not prioritize danceability as much, possibly focusing on other aspects of music, such as lyrics or emotional depth.
2. **Energy:**
   - Popular songs (average energy of 0.668) tend to be more energetic, with higher tempos, stronger beats, and more intense instrumentals. This higher energy level can grab the listener's attention and create excitement.
   - Unpopular songs (average energy of 0.598) may lean towards mellower or less intense musical styles, which might not appeal to a broader audience.
3. **Loudness:**
   - Popular songs being louder can contribute to their appeal in environments like clubs, parties, and radio where loud music stands out. This is because these venues have specific characteristics
   - Unpopular songs might prefer subtler dynamics, catering to a niche audience that values a different listening experience.
4. **Speechiness:**
   - Popular songs tend to have slightly less speechiness because they often focus on other aspects of the songs like rythm and instrumental. The lyrics focus more on quality than quantity. Lyrics that are catchy and easy to sing along with. This can make them more relatable and memorable for listeners.
   - Unpopular songs might prioritize lyrics a lot more and putting less effort in the instrumental elements, which can make them less immediately engaging for some listeners.
5. **Acousticness:**
   - Popular songs having lower acousticness values suggest that they are more likely to incorporate electronic instruments and production techniques, which can give them a modern and polished sound.
   - Unpopular songs might embrace acoustic instruments or a more organic sound, potentially appealing to listeners who appreciate a raw or traditional musical style.
6. **Key and Mode:**
   - Both popular and unpopular songs tend to be in similar keys, with an average key value around 5. This similarity in key might suggest that the choice of key doesn't strongly influence a song's popularity.
   - The absence of a significant difference in mode (major or minor) between popular and unpopular songs indicates that the emotional tone conveyed by the mode may not be a decisive factor in determining popularity.
7. **Instrumentalness:**

- Both popular and unpopular songs have low average instrumentalness values, indicating that most songs in both groups contain vocals or lyrics.
- This similarity suggests that vocals are a common and essential element in both popular and unpopular music, and instrumental tracks are less prevalent in both cases.

8. **Liveness:**

- There is not a significant difference in the average liveness between popular (0.176) and unpopular (0.172) songs.
- The similarity in liveness suggests that whether a song is performed live or in a studio setting does not strongly correlate with its popularity.

9. **Valence:**

- Popular songs have on average a higher valence. This can be explained by the positivity that comes in the song. The more valence in it, the better. This can also be influenced by the energy put in the songs. Energy could also influence the valence of a song.

10. **Tempo:**

- The average tempo is approximately the same for both popular and unpopular songs, with both groups having an average tempo around 121 BPM.
- This similarity suggests that tempo alone does not explain the difference in popularity, as both popular and unpopular songs encompass a wide range of tempos.

11. **Duration:**

- There is no significant difference in the average duration of popular and unpopular songs.
- The duration of a song may not be a decisive factor in determining its popularity, as both short and long songs can achieve popularity.

12. **Time Signature:**

- Most songs in both groups have a time signature of 4/4.
- This suggests that the choice of time signature, at least within the common 4/4 time signature, does not strongly influence a song's popularity.

1. **YouTube Metrics (View Count, Like Count, Comment Count):**
   - The significant differences in YouTube metrics are likely due to the self-reinforcing nature of popularity. Popular songs receive more visibility and engagement on platforms like YouTube, making them even more popular.
   - Unpopular songs might not have received the same level of exposure or marketing, resulting in lower YouTube engagement metrics. **Note:** The data collected from Youtube at the time of this analysis is the overall data until the day of the analysis. This means that we did not consider the reverse effect of the popular songs presence on the billboard on the Youtube(This will be considered in a later phase of this project). The presence of the song on the billboard could have influenced its visibility and pushed more people to go interact with it on Youtbe. Ideally, we would only consider the data(view count, like count, comment count) until the date the song made it to the billboard.

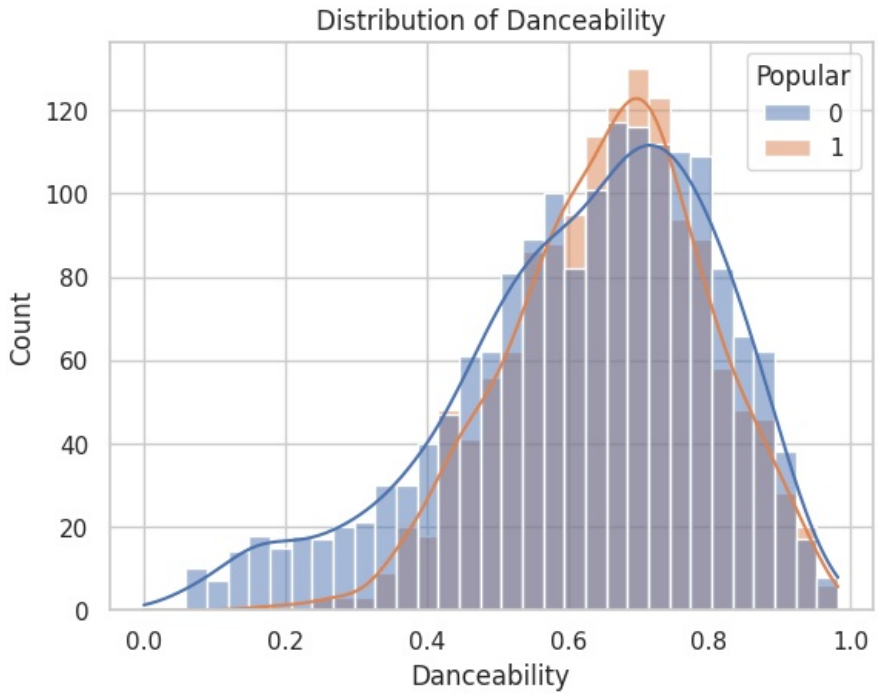
2. Distribution histograms for each feature

Now, let's look at the distribution of each feature to identify some trends and patterns within our data. We can analyse things, such as skewness, central tendency and modality.

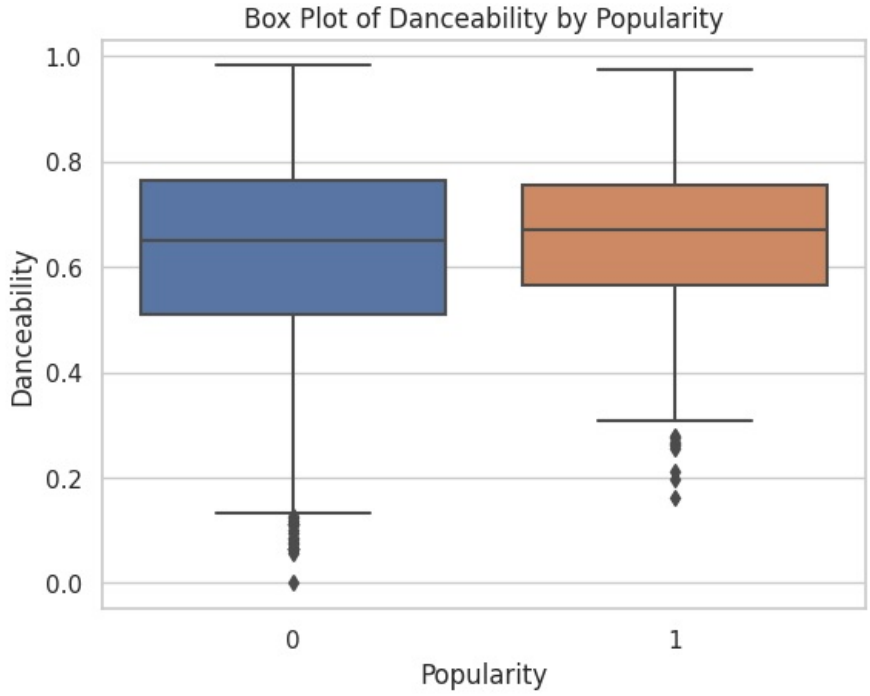
1. **Danceability**

- The following feature is negatively skewed (left-skewed) for both popular and unpopular songs. From this, we can refer that most of the songs have a high danceability score.
- Most of the songs (both popular and unpoular) have a danceability score between 0,6 and 0,8. Distribution is unimodal.
- Spread of data: IQR of 0,189 and 0,255.
- We can see that for both types of songs (popular and unpopular) we encounter low outliers (below 0,3 in popular, and below 0,17 in unpopular).

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

sns.set(style="whitegrid")

sns.histplot(data=df, x='danceability', hue='Popular', kde=True)
plt.title('Distribution of Danceability')
plt.xlabel('Danceability')
plt.ylabel('Count')
plt.show()
```



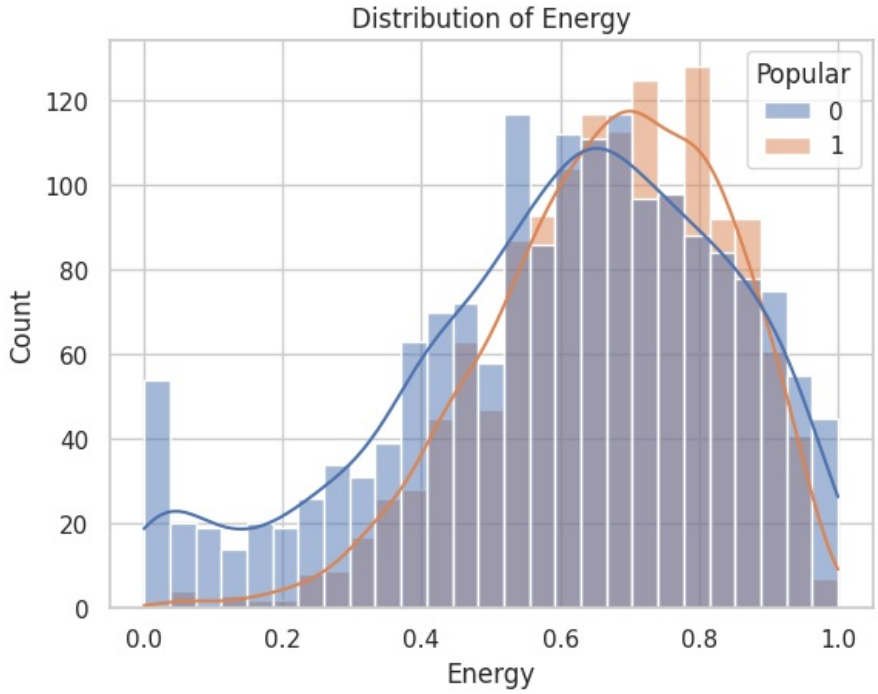
In [ ]:
```
# Box plot for a specific feature (e.g., 'danceability')
sns.boxplot(data=df, x='Popular', y='danceability')
plt.title('Box Plot of Danceability by Popularity')
plt.xlabel('Popularity')
plt.ylabel('Danceability')
plt.show()
```



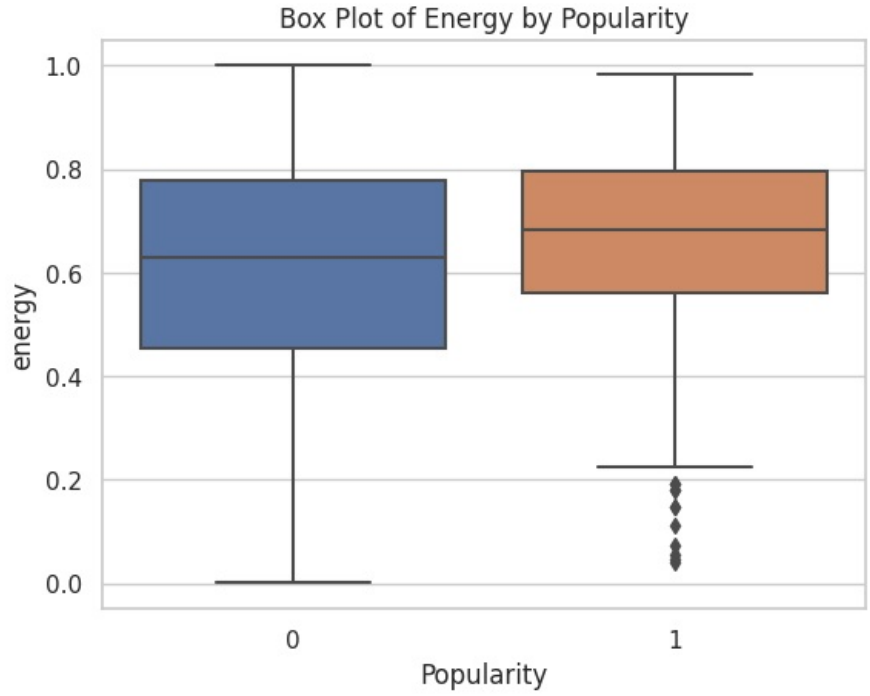
1. **Energy**

- The following feature is negatively skewed (left-skewed) for both popular and unpopular songs. From this, we can refer that most of the songs have a high energy score.
- Most of the songs (both popular and unpoular) have an energy score between 0,5 and 0,8. Distribution of data is unimodal.
- Spread of data: IQR of 0,237 and 0,323.
- We can see that for popular songs, we have low outliers (below 0,2). For unpopular songs, we do not observe any outliers.

In [ ]:
```
sns.histplot(data=df, x='energy', hue='Popular', kde=True)
plt.title('Distribution of Energy')
plt.xlabel('Energy')
plt.ylabel('Count')
plt.show()
```



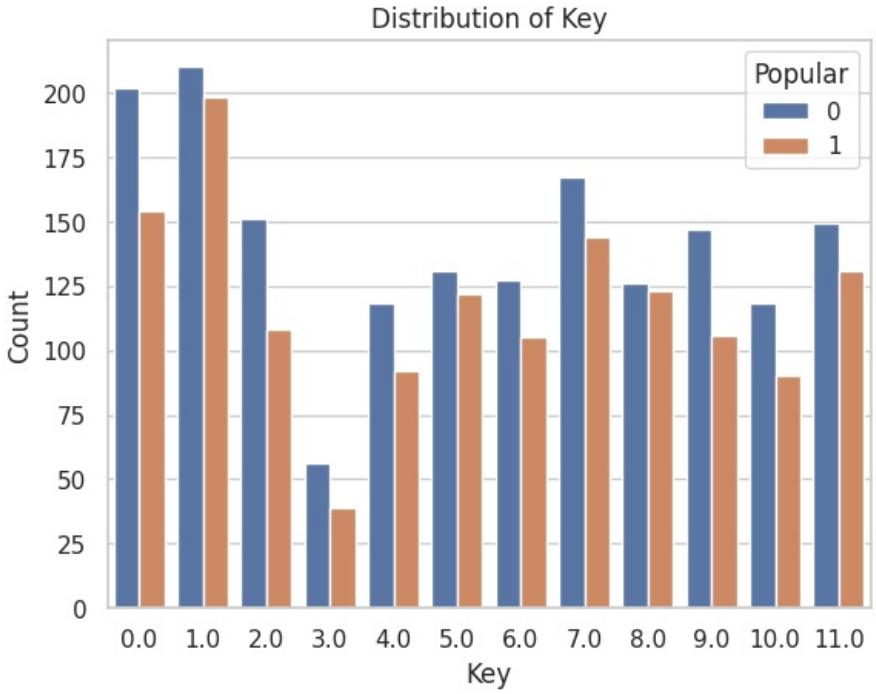
In [ ]:
```
# Box plot for a specific feature (e.g., 'energy')
sns.boxplot(data=df, x='Popular', y='energy')
plt.title('Box Plot of Energy by Popularity')
plt.xlabel('Popularity')
plt.ylabel('energy')
plt.show()
```



1. **Key**

- The distirbution of key among popular and unpopular songs seem to follow the similar trend. The frequency of each key seems to be slightly higher for popular songs.
- The most frequent keys are 0.0 (C), 1.0 (C#) and 7.0 (G) in both types of songs. Other keys have relatively similar frequency among songs. However, key 3.0 (D#) has the lowest frequency among all songs.

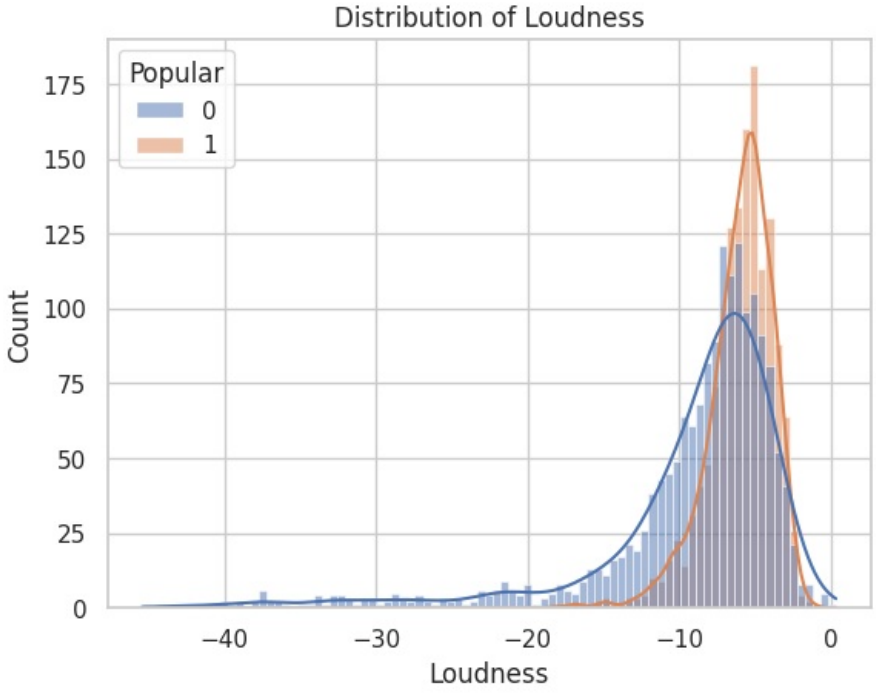
In [ ]:
```
sns.countplot(data=df, x='key', hue='Popular')
plt.title('Distribution of Key')
plt.xlabel('Key')
plt.ylabel('Count')

# Show the plot
plt.show()
```



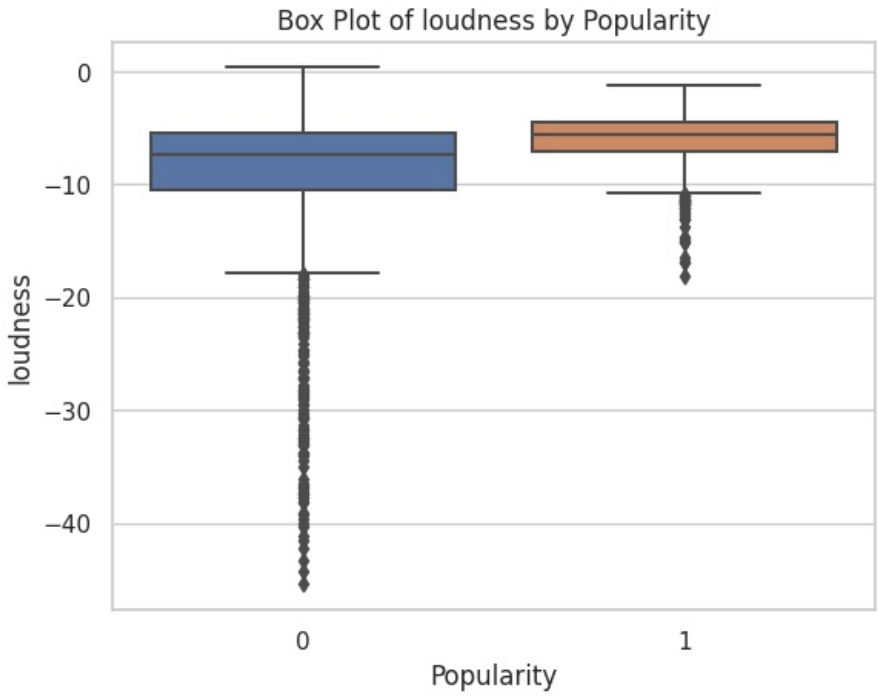
1. **Loudness**

- To provide some context, platforms like Youtube and Spotify has a standard for loudness of - 14 LUFS. Hence, music loudness of -14 LUFS and above is preferable in tracks.
- The following feature is negatively skewed (left-skewed) for both popular and unpopular songs. From this, we can refer that most of the songs have a high loudness score (higher than the Spotify standard).
- Most of the songs (both popular and unpoular) have a loudness score between -3 and -10. Distribution of data is unimodal.
- Spread of data: IQR of 2,545 and 5,046.
- We can see that for both types of songs, we have low outliers. For popular songs, outliers are below -11. For unpopular songs, outliers are below -17,5.

In [ ]:
```
sns.histplot(data=df, x='loudness', hue='Popular', kde=True)
plt.title('Distribution of Loudness')
plt.xlabel('Loudness')
plt.ylabel('Count')
plt.show()
```



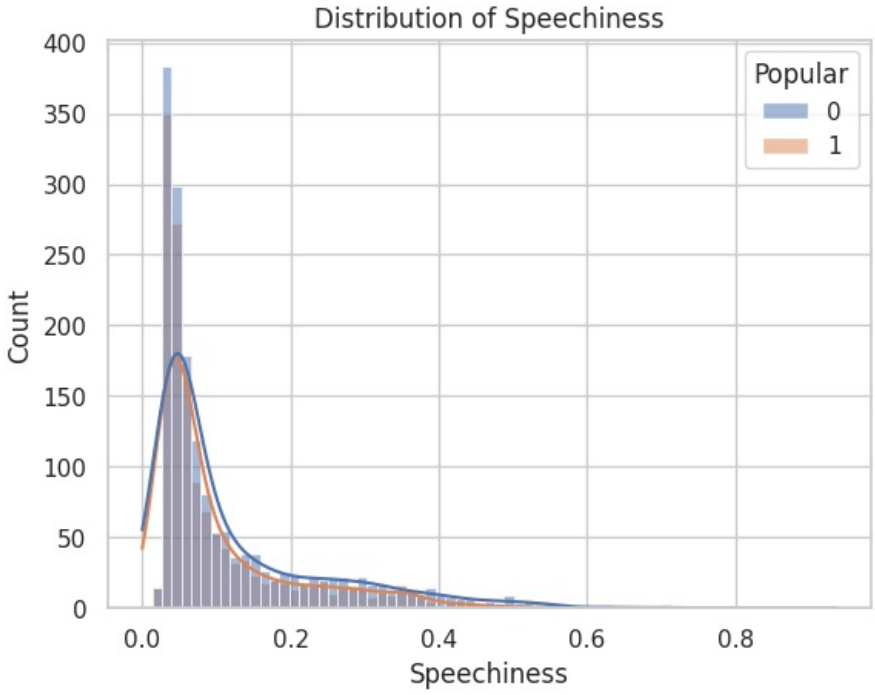
In [ ]:
```
# Box plot for a specific feature (e.g., 'loudness')
sns.boxplot(data=df, x='Popular', y='loudness')
plt.title('Box Plot of loudness by Popularity')
plt.xlabel('Popularity')
plt.ylabel('loudness')
plt.show()
```



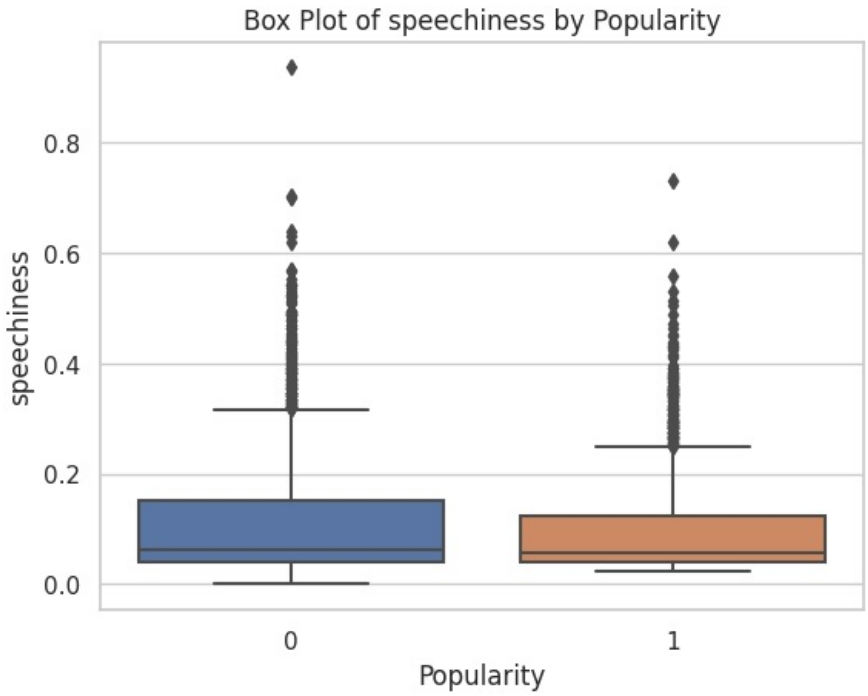
1. **Speechiness**

- Speechiness shows to which extent is song made out of spoken words. Closer it is to 1, more words the song contains.
- The following feature is positively skewed (right-skewed) for both popular and unpopular songs. From this, we can refer that most of the songs have a low speechiness score.
- Most of the songs (both popular and unpoular) have a speechiness score between 0,0 and 0,1. Distribution of data is unimodal.
- Spread of data: IQR of 0,084 and 0,254.
- We can see that for both types of songs, we have high outliers. For popular songs, outliers are above 0,24. For unpopular songs, outliers are above 0,26.

In [ ]:
```
sns.histplot(data=df, x='speechiness', hue='Popular', kde=True)
plt.title('Distribution of Speechiness')
plt.xlabel('Speechiness')
plt.ylabel('Count')
plt.show()
```



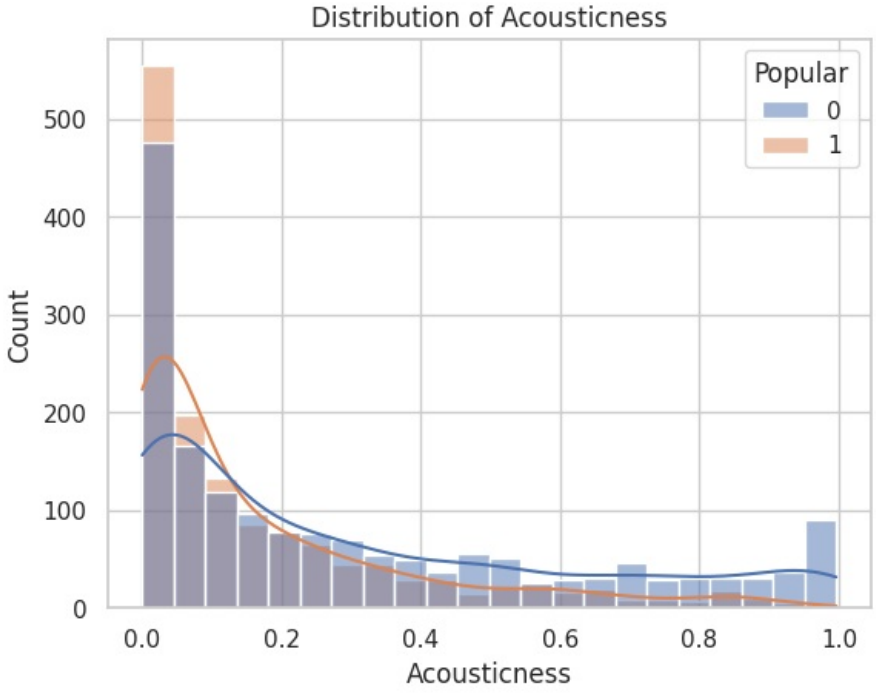
In [ ]:
```
# Box plot for a specific feature (e.g., 'speechiness')
sns.boxplot(data=df, x='Popular', y='speechiness')
plt.title('Box Plot of Speechiness by Popularity')
plt.xlabel('Popularity')
plt.ylabel('speechiness')
plt.show()
```



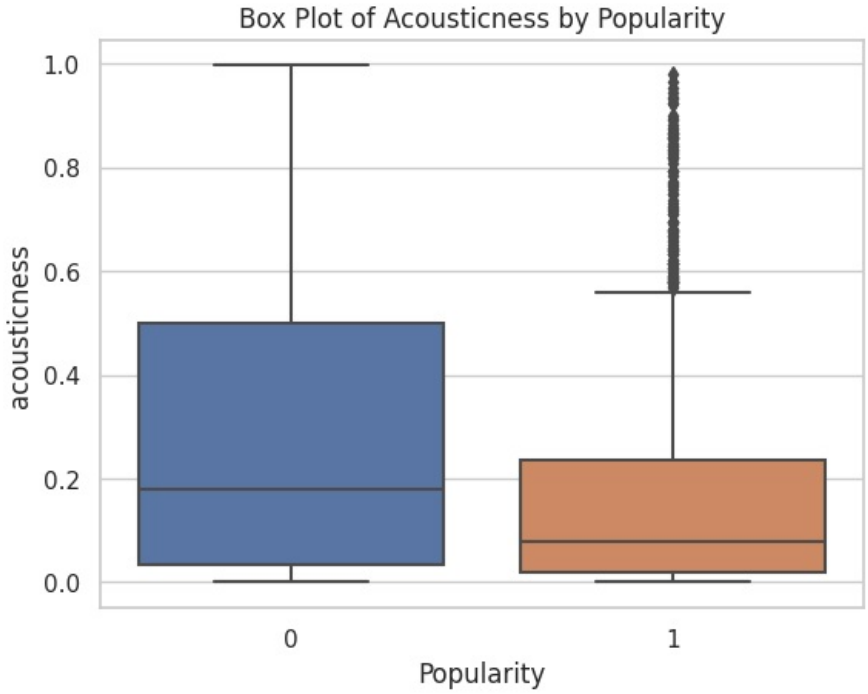
1. **Acousticness**

- Acousticness shows to which extent is song acoustic. Closer it is to 1, more acoustic it is.
- The following feature is positively skewed (right-skewed) for both popular and unpopular songs. From this, we can refer that most of the songs have a low acousticness score.
- Most of the songs (both popular and unpopular) have an acousticness score between 0,0 and 0,2. Distribution of data is unimodal.
- Spread of data: IQR of 0,218 and 0,465.
- We can see that for popular songs, we have high outliers (above 0,56). For unpopular songs, we do not have any outliers.

In [ ]:
```
sns.histplot(data=df, x='acousticness', hue='Popular', kde=True)
plt.title('Distribution of Acousticness')
plt.xlabel('Acousticness')
plt.ylabel('Count')
plt.show()
```



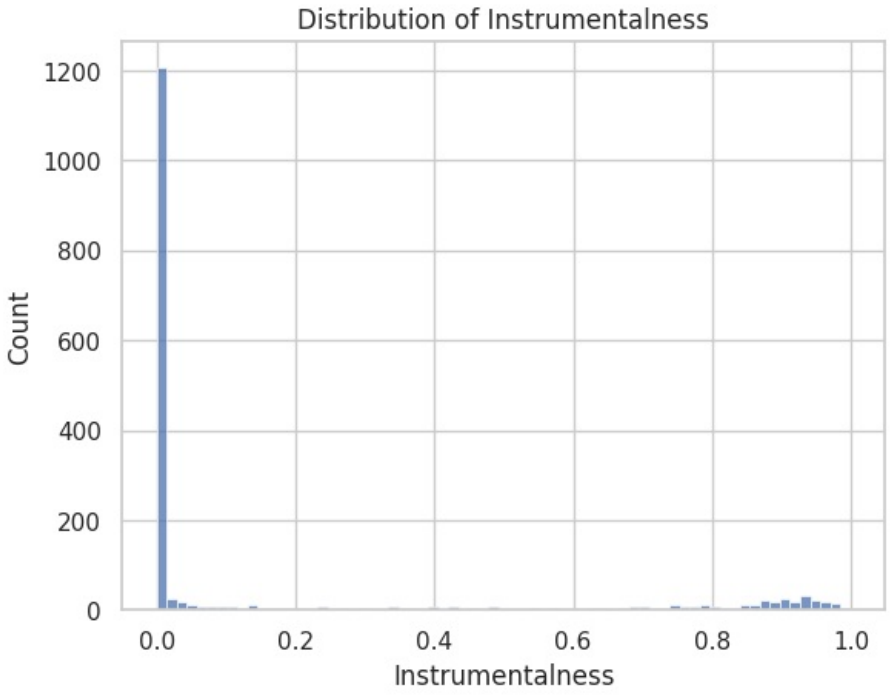
In [ ]:
```
# Box plot for a specific feature (e.g., 'acousticness')
sns.boxplot(data=df, x='Popular', y='acousticness')
plt.title('Box Plot of Acousticness by Popularity')
plt.xlabel('Popularity')
plt.ylabel('acousticness')
plt.show()
```



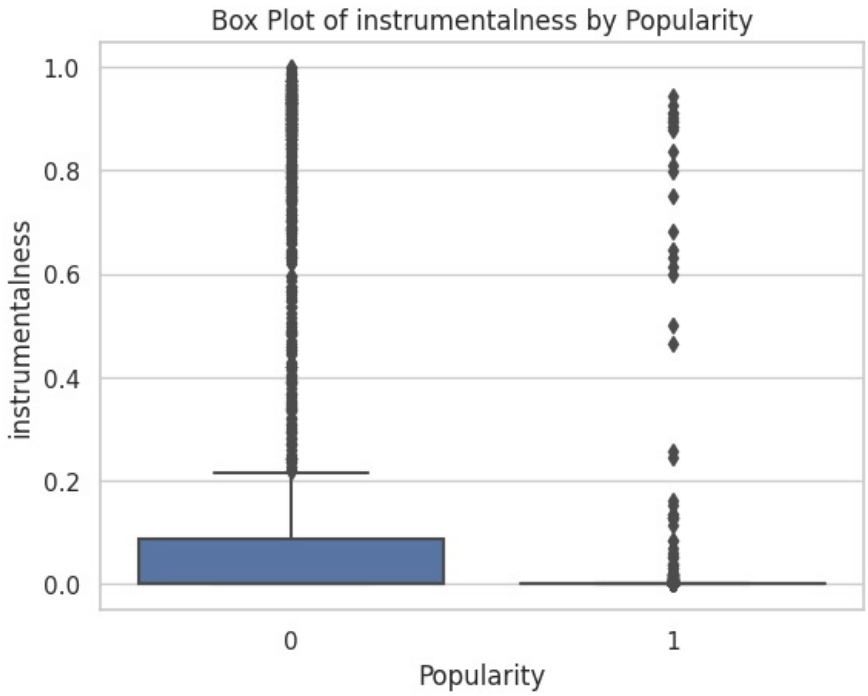
1. **Instrumentalness**

- We have an interesting observation here.
- Both types of songs are distributed quite randomly. We could not observe any pattern or trend within the data.
- Among unpopular songs, most of them have an instrumentalness score close to 0.

In [ ]:
```
sns.histplot(data=unpopular, x='instrumentalness')
plt.title('Distribution of Instrumentalness')
plt.xlabel('Instrumentalness')
plt.ylabel('Count')
plt.show()
```



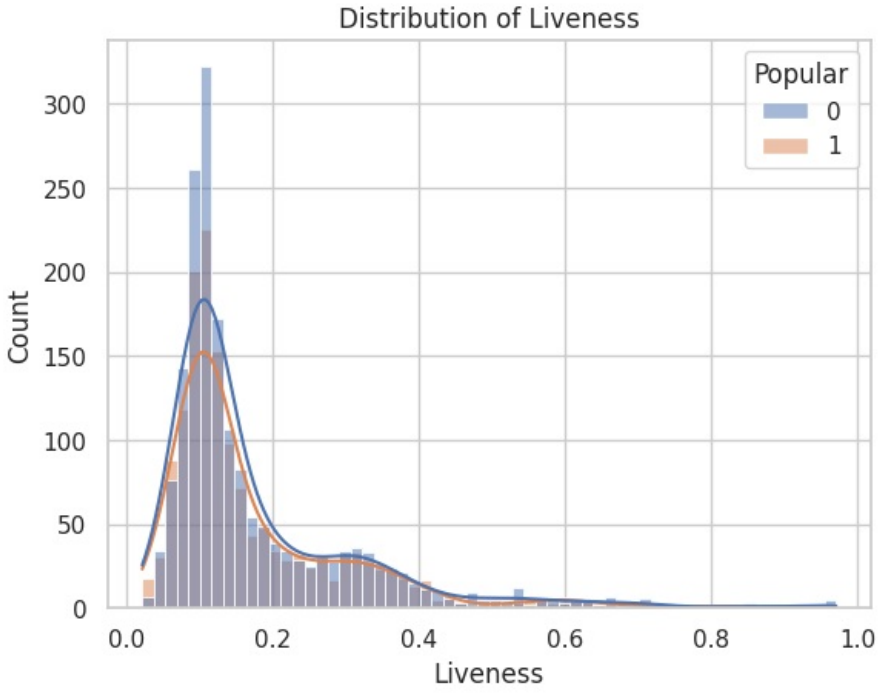
In [ ]:
```
sns.boxplot(data=df, x='Popular', y='instrumentalness')
plt.title('Box Plot of instrumentalness by Popularity')
plt.xlabel('Popularity')
plt.ylabel('instrumentalness')
plt.show()
```



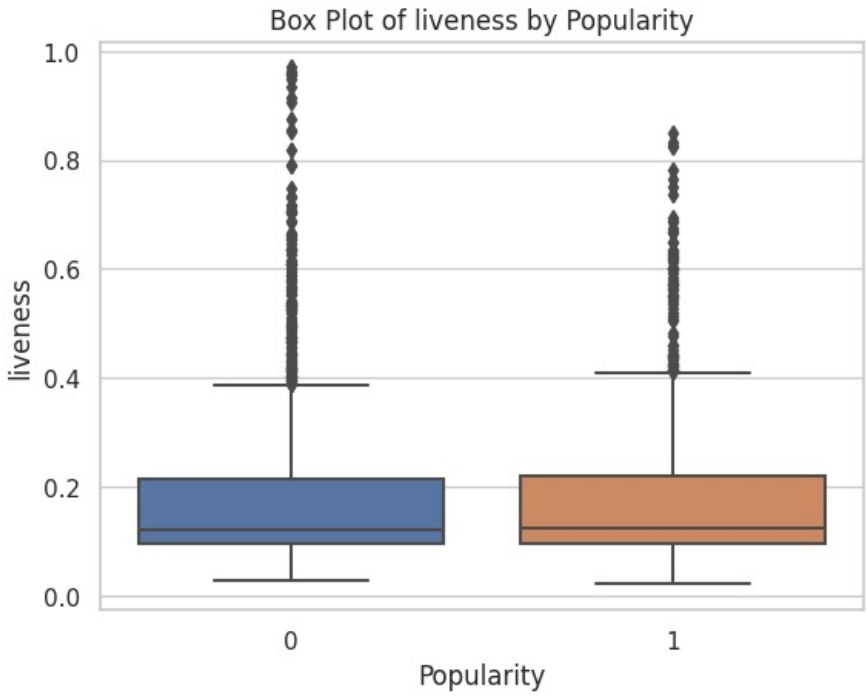
1. **Liveness**

- Liveness shows to which extent is song live. Closer it is to 1, more live it is.
- The following feature is positively skewed (right-skewed) for both popular and unpopular songs. From this, we can refer that most of the songs have a low liveness score.
- Most of the songs (both popular and unpopular) have an acousticness score between 0,0 and 0,2. Distribution of data is unimodal.
- Spread of data: IQR of 0,126 and 0,116.
- We can see that we have high outliers for both types of songs. For popular songs, it is above 0,41. For unpopular songs, it is above 0,39.

In [ ]:
```
sns.histplot(data=df, x='liveness', hue='Popular', kde=True)
plt.title('Distribution of Liveness')
plt.xlabel('Liveness')
plt.ylabel('Count')
plt.show()
```



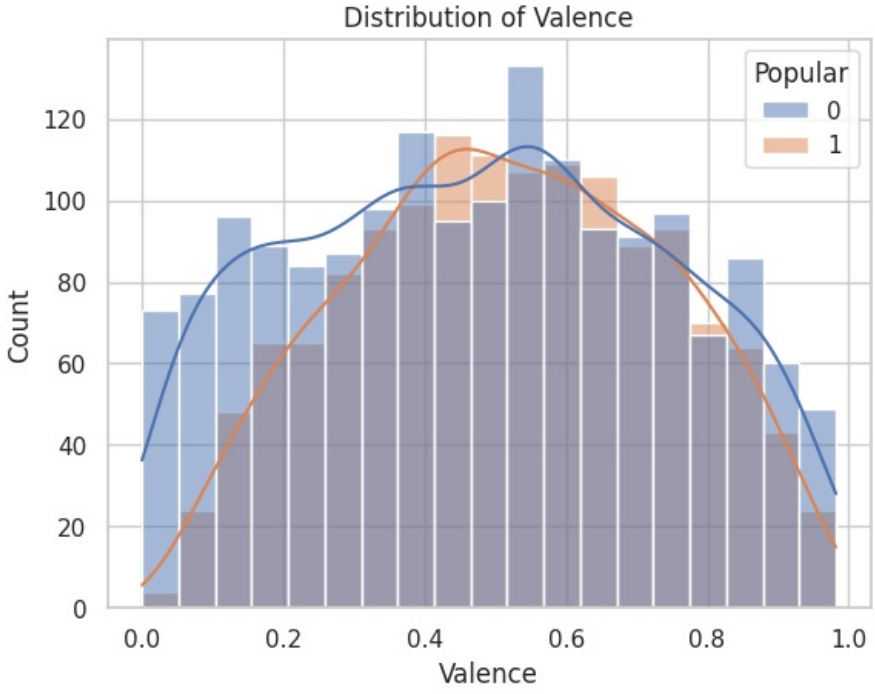
In [ ]:
```
# Box plot for a specific feature (e.g., 'liveness')
sns.boxplot(data=df, x='Popular', y='liveness')
plt.title('Box Plot of liveness by Popularity')
plt.xlabel('Popularity')
plt.ylabel('liveness')
plt.show()
```



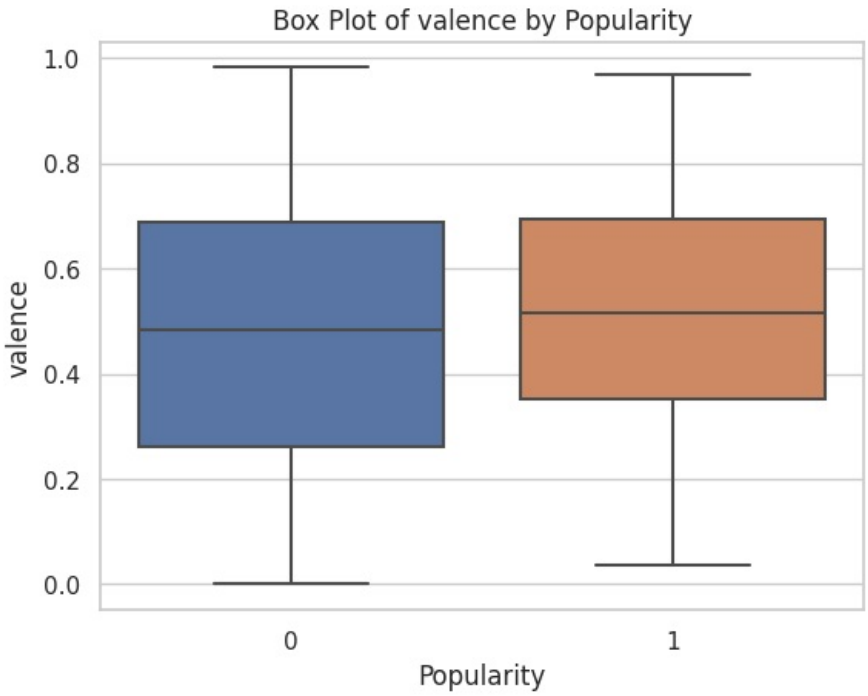
1. **Valence**

- Valence shows to which extent is song musically positive. Closer it is to 1, more positive it is.
- The following feature is normally distributed among both popular and unpopular songs. From this, we can refer that most of the songs have an average (normal) valence.
- Most of the songs (both popular and unpopular) have a valence score between 0,4 and 0,6. Distribution of data is unimodal.
- Spread of data: IQR of 0,342 and 0,425.
- We can see that we do not have any outliers for both types of songs.

In [ ]:
```
sns.histplot(data=df, x='valence', hue='Popular', kde=True)
plt.title('Distribution of Valence')
plt.xlabel('Valence')
plt.ylabel('Count')
plt.show()
```



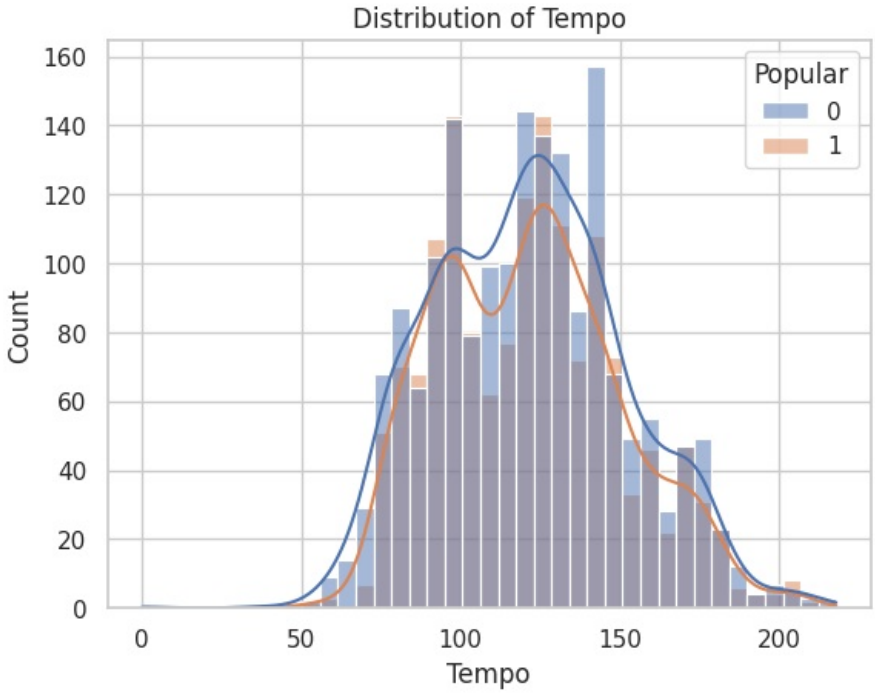
In [ ]:
```
# Box plot for a specific feature (e.g., 'valence')
sns.boxplot(data=df, x='Popular', y='valence')
plt.title('Box Plot of valence by Popularity')
plt.xlabel('Popularity')
plt.ylabel('valence')
plt.show()
```



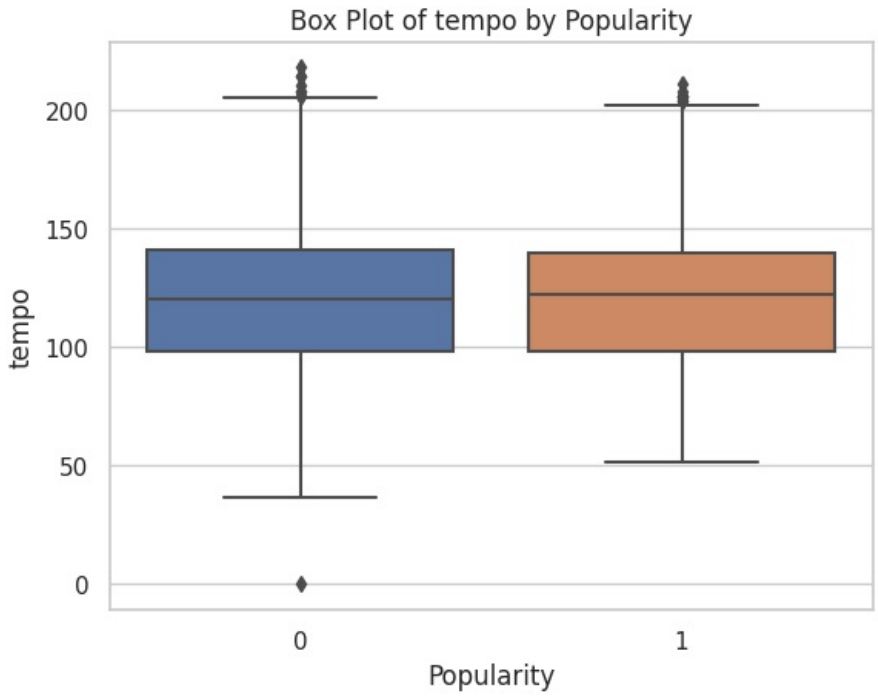
1. **Tempo**

- Tempo shows how many beats per minute does the song have.
- The following feature is normally distributed among both popular and unpopular songs. From this, we can refer that most of the songs have an average (normal) tempo.
- Most of the songs (both popular and unpopular) have a tempo score between 90 and 130. Distribution of data is bimodal.
- Spread of data: IQR of 0,342 and 0,425.
- We can see that we have some outliers. Popular songs have high outliers that are above 201. Unpopular songs have high outliers (above 210) and a low outlier (equals 0).

In [ ]:
```
sns.histplot(data=df, x='tempo', hue='Popular', kde=True)
plt.title('Distribution of Tempo')
plt.xlabel('Tempo')
plt.ylabel('Count')
plt.show()
```



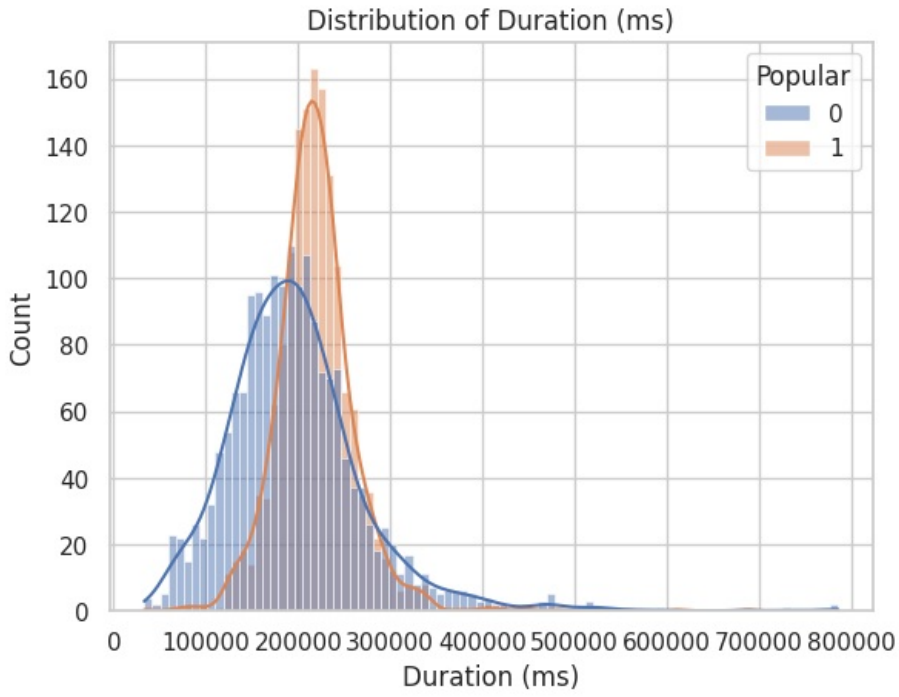
In [ ]:
```
# Box plot for a specific feature (e.g., 'tempo')
sns.boxplot(data=df, x='Popular', y='tempo')
plt.title('Box Plot of tempo by Popularity')
plt.xlabel('Popularity')
plt.ylabel('tempo')
plt.show()
```



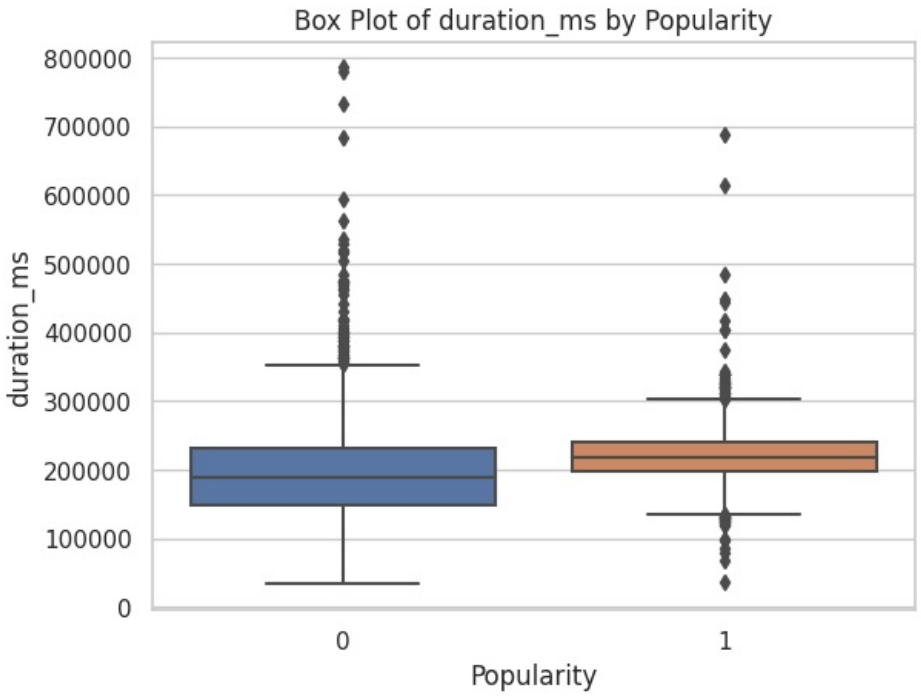
1. **Duration (ms)**

- Duration shows the length of a song.
- The following feature is normally distributed among popular and unpopular songs.
- Most of the songs (both popular and unpopular) have a duration of 100,000 - 300,000. The distribution of data is unimodal.
- Mean, mode and median for popular songs is slighlty higher (about 220000) compared to unpopular songs (about 185000).
- We can see that for popular songs, we have high outliers (above 350,000). For unpopular songs, we have low and high outliers (above 300,000 and below 140,000).

In [ ]:
```
sns.histplot(data=df, x='duration_ms', hue='Popular', kde=True)
plt.title('Distribution of Duration (ms)')
plt.xlabel('Duration (ms)')
plt.ylabel('Count')
plt.show()
```



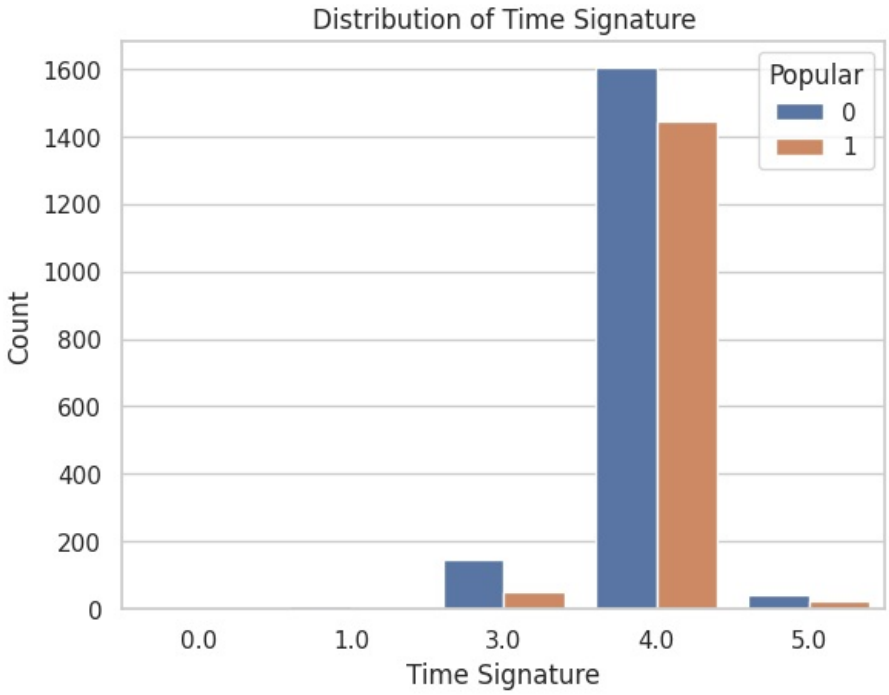
In [ ]:
```
# Box plot for a specific feature (e.g., 'duration_ms')
sns.boxplot(data=df, x='Popular', y='duration_ms')
plt.title('Box Plot of duration_ms by Popularity')
plt.xlabel('Popularity')
plt.ylabel('duration_ms')
plt.show()
```



1. **Time Signature**

- Identifies how many beats are in each bar. The value usually ranges from 3 to 7.
- The distribution of time signature is similar among popular and unpopular songs. The time signature with the highest frequency among both types is 4. However, there are some songs with time signature of 3 and 5.

In [ ]:
```
sns.countplot(data=df, x='time_signature', hue='Popular')
plt.title('Distribution of Time Signature')
plt.xlabel('Time Signature')
plt.ylabel('Count')

# Show the plot
plt.show()
```



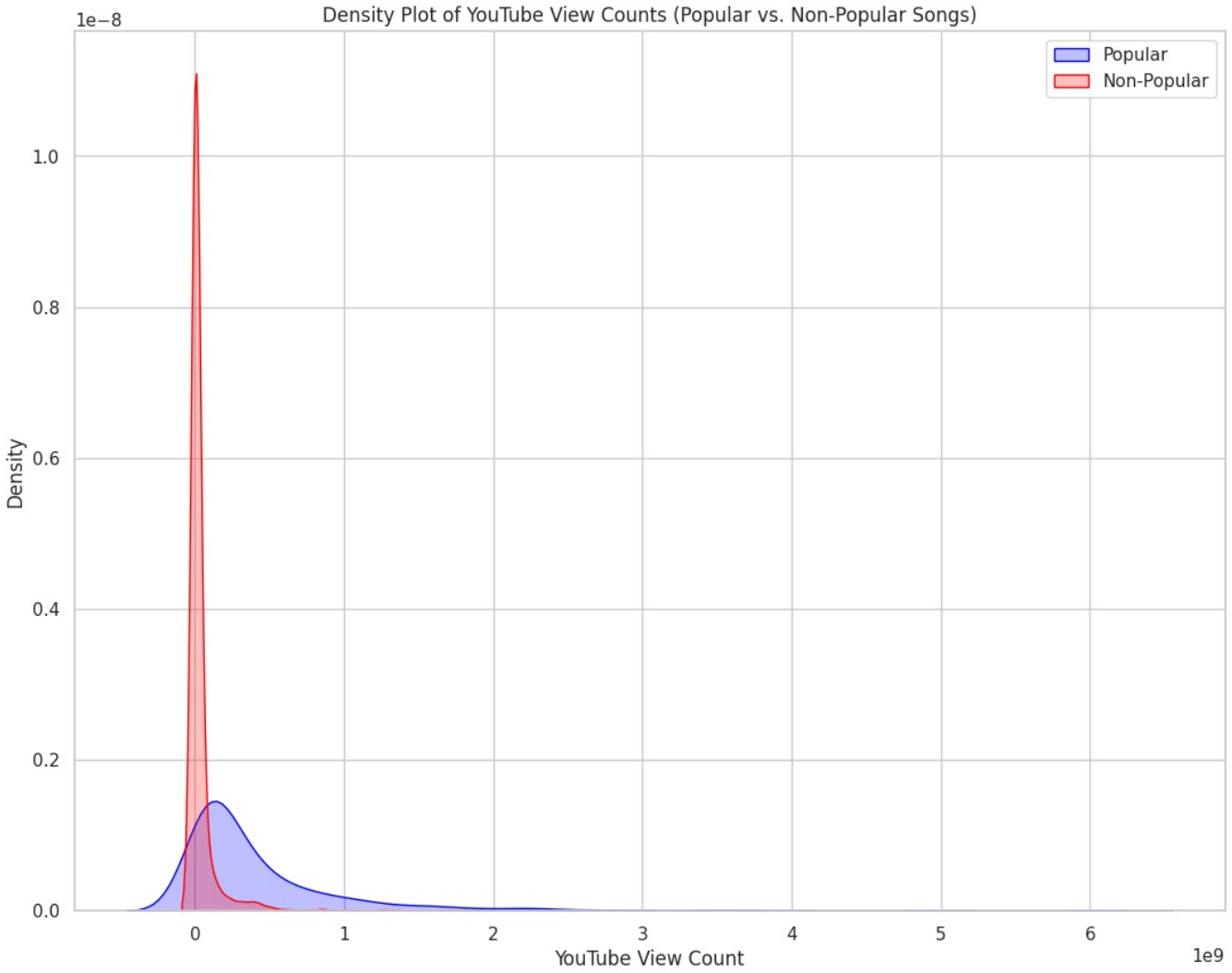
1. **Youtube View Count**

- The distribution for popular songs is positively skewed (right skewed). Most of the songs having a view count between 0 and 500,000,000. Popular songs have a lot of high outliers. Most of the outliers lie between 1,2 - 4 (*10^9), and some of the outliers go above that.
- Most unpopular songs have a view count close to 0, yet a lot of songs reach up to 1,3 (*10^9) views. There is a highest outlier of 1,8 10^9.

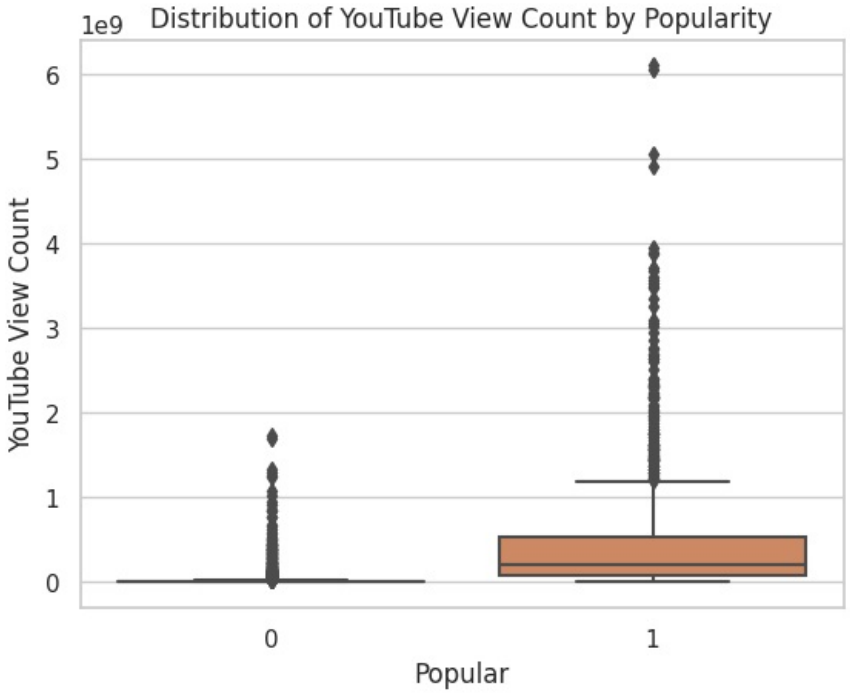
In [ ]:
```
plt.figure(figsize=(13, 10))
```

```
sns.kdeplot(data=popular, x='youtube_view_count', label='Popular', shade=True, color='blue')
sns.kdeplot(data=unpopular, x='youtube_view_count', label='Non-Popular', shade=True, color='red')
plt.title('Density Plot of YouTube View Counts (Popular vs. Non-Popular Songs)')
plt.xlabel('YouTube View Count')
plt.ylabel('Density')
plt.legend()
plt.show()
```

```
<ipython-input-59-2b868f523a3f>:3: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=popular, x='youtube_view_count', label='Popular', shade=True, color='blue')
<ipython-input-59-2b868f523a3f>:4: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=unpopular, x='youtube_view_count', label='Non-Popular', shade=True, color='red')
```



In [ ]:
```
sns.boxplot(data=df, x='Popular', y='youtube_view_count')
plt.title('Distribution of YouTube View Count by Popularity')
plt.xlabel('Popular')
plt.ylabel('YouTube View Count')
plt.show()
```

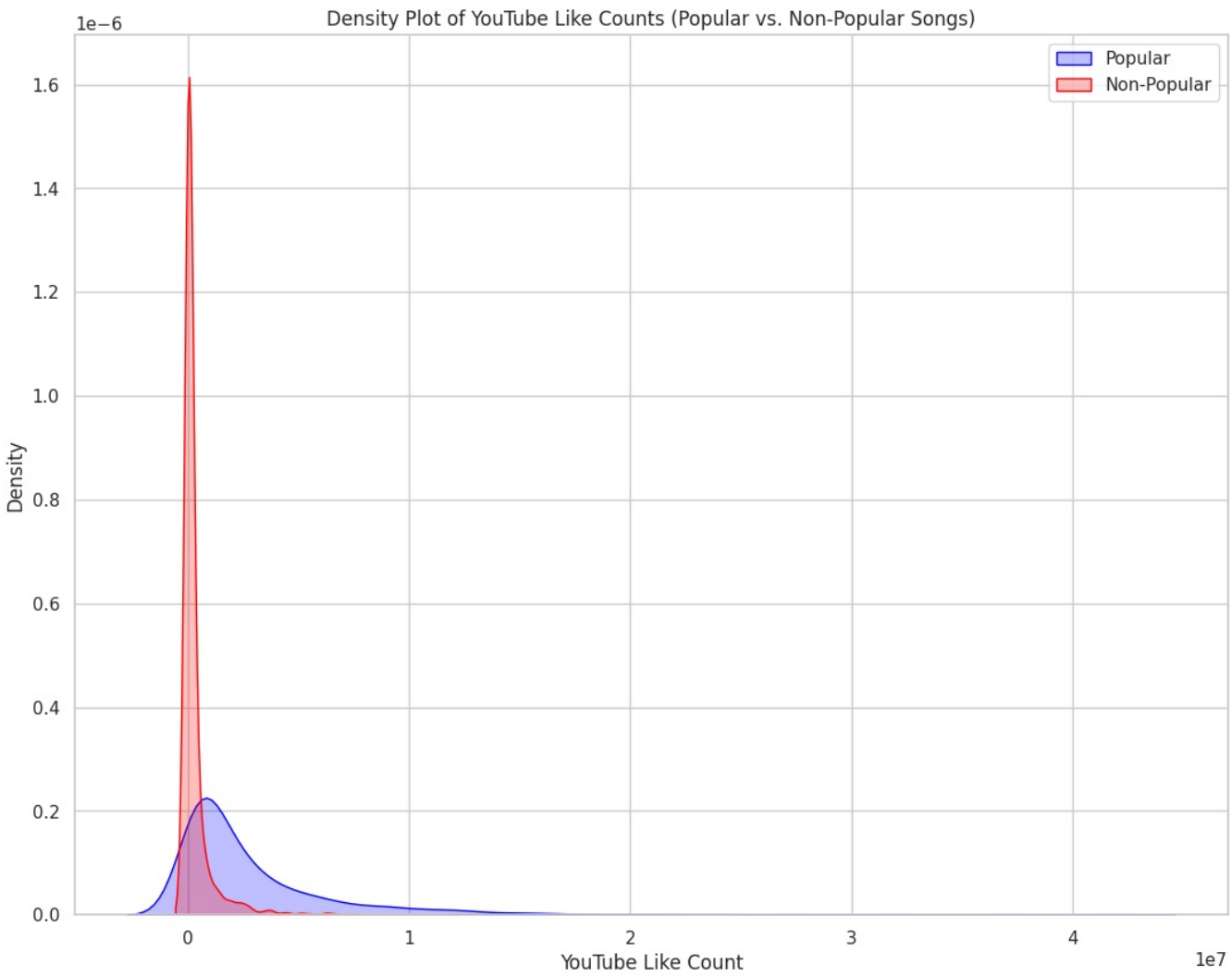


1. **Youtube like count**

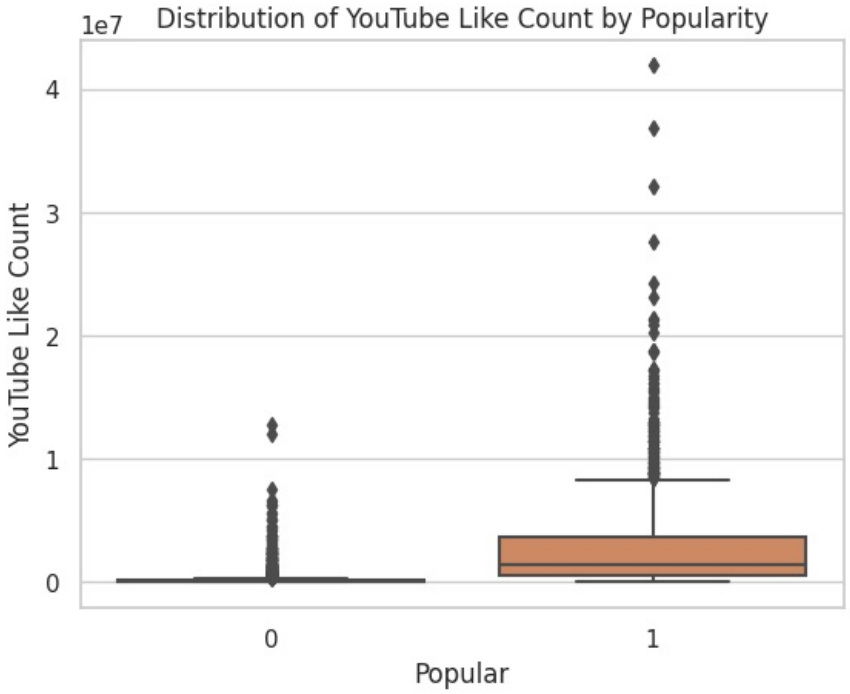
- The distribution for popular songs is positively skewed (right skewed). Most of the songs having a view count between 0 and 500,000,000. Popular songs have a lot of high outliers. Most of the outliers lie between 0,8 - 2,5 (*10^9), and some of the outliers go above that.
- Most unpopular songs have a view count close to 0, yet a lot of songs reach up to 0,8 (*10^9) views. Couple of the songs go beyond 1* 10^9.

In [ ]:
```
plt.figure(figsize=(13, 10))
sns.kdeplot(data=popular, x='youtube_like_count', label='Popular', shade=True, color='blue')
sns.kdeplot(data=unpopular, x='youtube_like_count', label='Non-Popular', shade=True, color='red')
plt.title('Density Plot of YouTube Like Counts (Popular vs. Non-Popular Songs)')
plt.xlabel('YouTube Like Count')
plt.ylabel('Density')
plt.legend()
plt.show()
```

```
<ipython-input-32-c95234ed5325>:2: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=popular, x='youtube_like_count', label='Popular', shade=True, color='blue')
<ipython-input-32-c95234ed5325>:3: FutureWarning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  sns.kdeplot(data=unpopular, x='youtube_like_count', label='Non-Popular', shade=True, color='red')
```



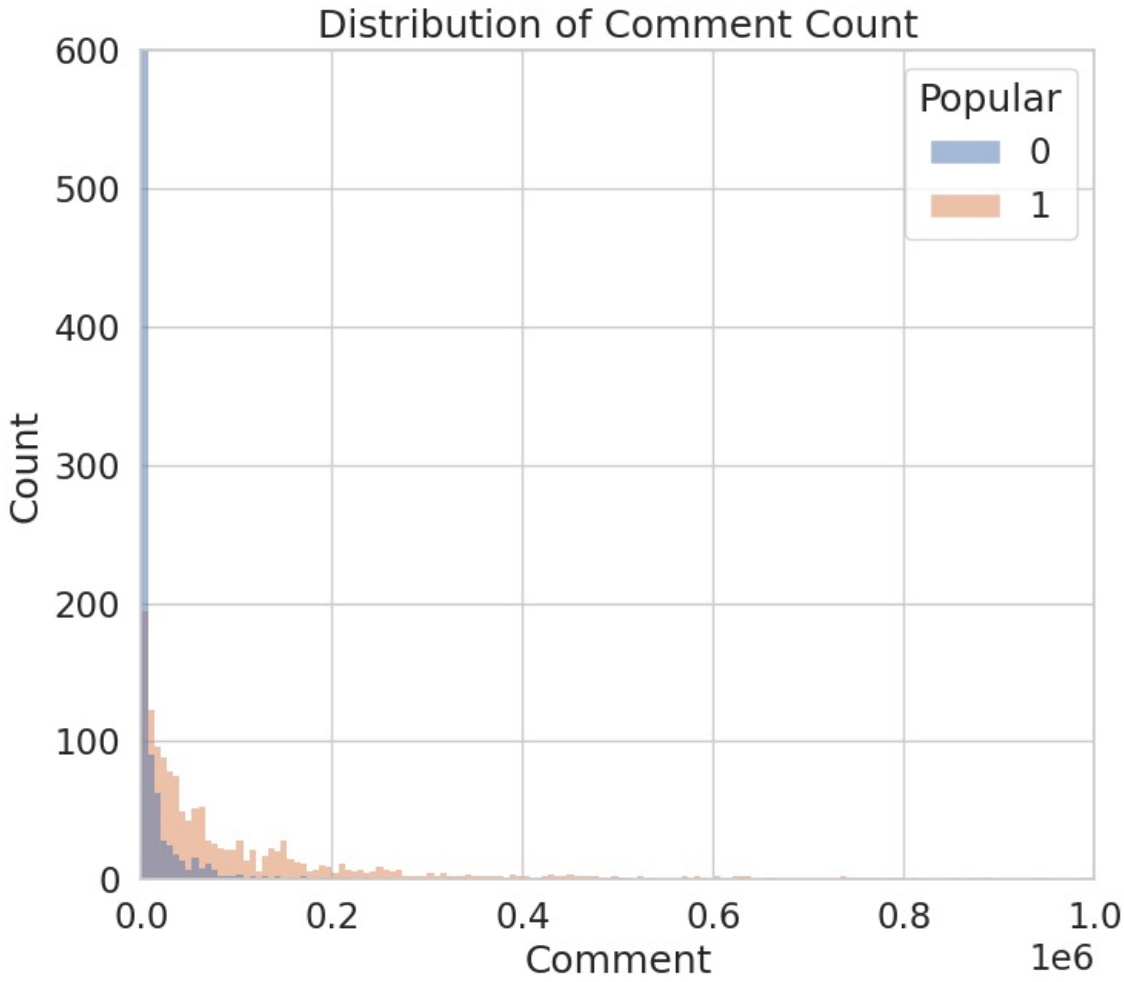
In [ ]:
```
sns.boxplot(data=df, x='Popular', y='youtube_like_count')
plt.title('Distribution of YouTube Like Count by Popularity')
plt.xlabel('Popular')
plt.ylabel('YouTube Like Count')
plt.show()
```



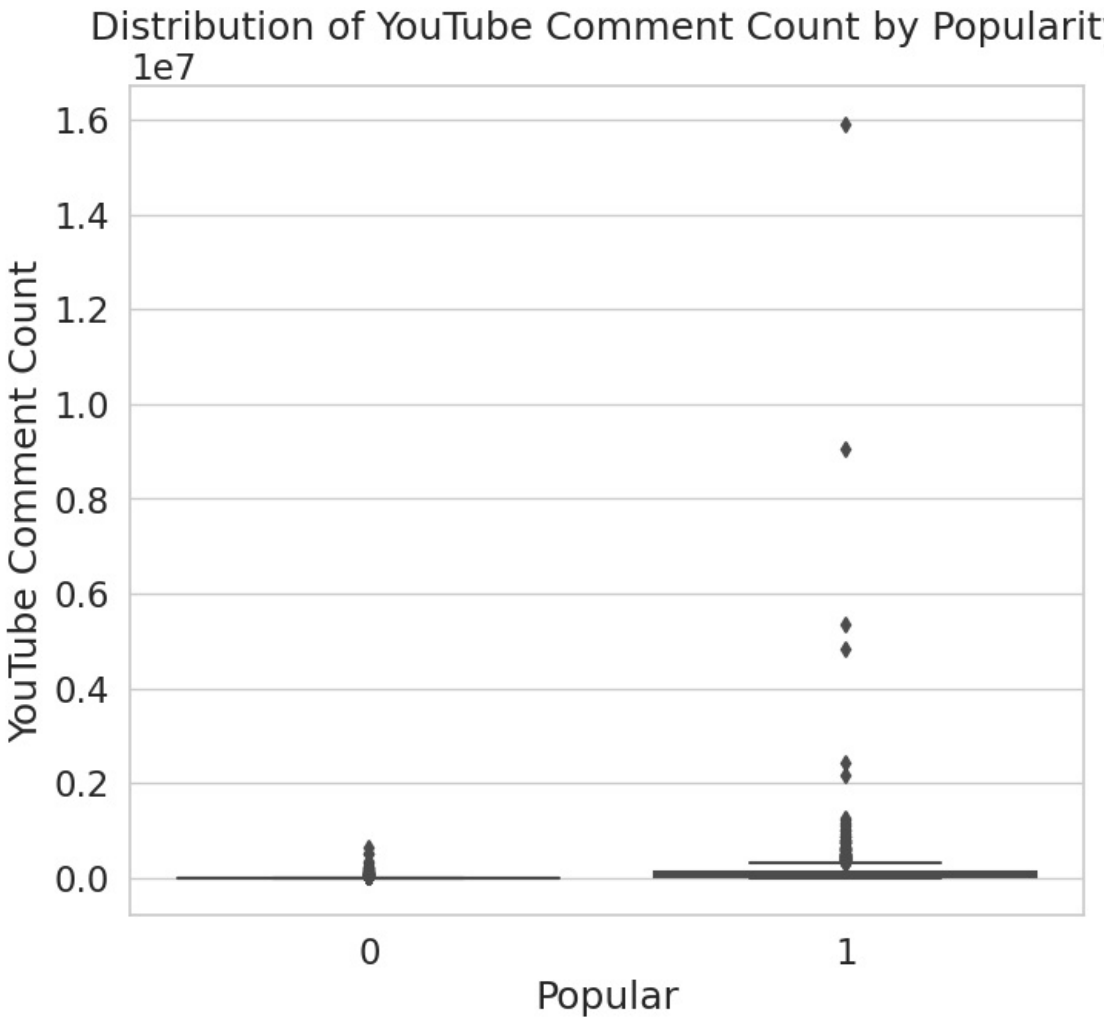
1. **Youtube Comment Count**

- Comment count for both types of songs (videos featuring the songs) follow similar trend. High number of unpopular songs have no comments at all. This is also true for popular songs, yet the number is dramatically lower.
- Number of songs decreases almost exponentially, as the number of comments increases.
- Among popular songs, many songs have higher number of comments (randomly distributed). Many more popular songs have comment count in between 0,001 - 0,2 (*10^6) compared to unpopular songs (highest number is zero comments).

In [ ]:
```
sns.histplot(data=df, x='youtube_comment_count', hue='Popular')
plt.title('Distribution of Comment Count')
plt.xlabel('Comment')
plt.ylabel('Count')
plt.xlim(0, 0.1 *10**7)
plt.ylim(0, 600)
plt.show()
```



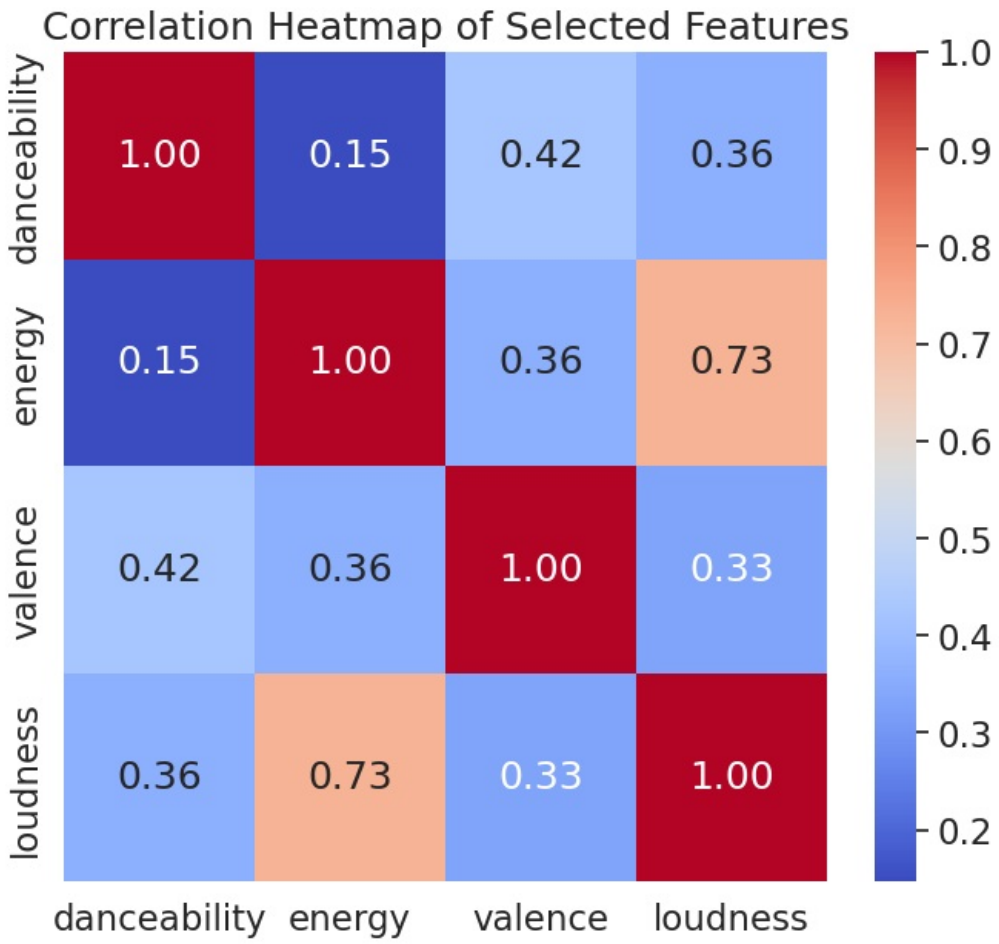
In [ ]:
```
sns.boxplot(data=df, x='Popular', y='youtube_comment_count')
plt.title('Distribution of YouTube Comment Count by Popularity')
plt.xlabel('Popular')
plt.ylabel('YouTube Comment Count')
plt.show()
```



3. Pairwise visualization

Let us try to evaluate some relations between the independent features:

In [ ]:
```
selected_features=['danceability', 'energy', 'valence', 'loudness']
correlation_matrix = df[selected_features].corr()
sns.heatmap(data=correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Selected Features')
plt.show()
```



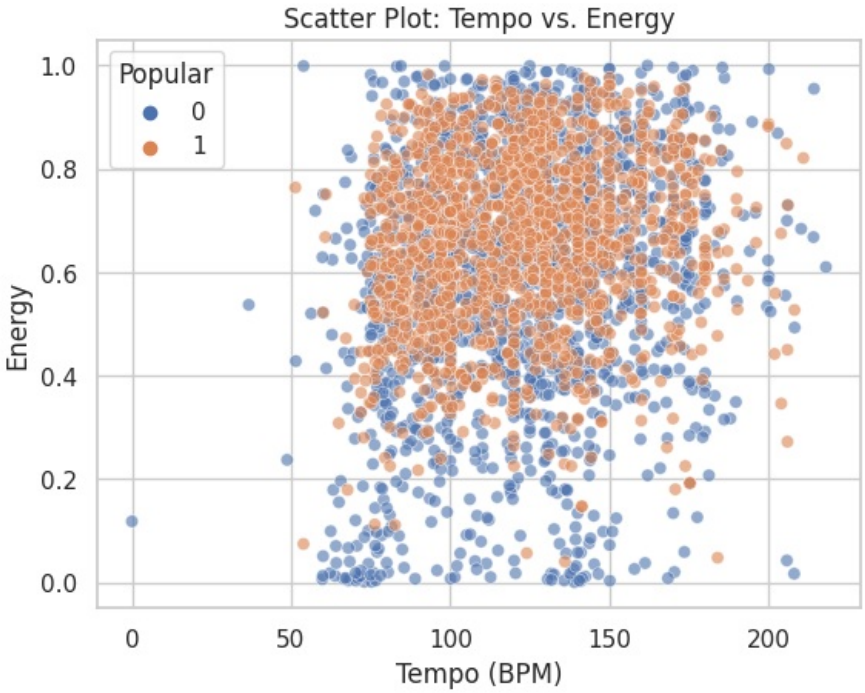
1. We can see in here that there is no real high correlation in between features. However, we see a 0.42 correlation effect between Valence and Danceability and 0.37 between Valence and Energy. Just as a reminder, valence refers to measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with higher valence sound more positive (e.g. happy, cheerful, euphoric), while tracks with lower valence sound more negative (e.g. sad, depressed, angry).

2. Hence, this could make sense since danceable songs tend to be more euphoric and have more rhythm. We rarely see sad and depressed songs that make people dance. So danceability and valence are definitely correlated. The same principle works for energy. Energy usually brings very heavy instruments and are highly influenced by drums. The higher the energy, the happier the song and melody will be.

3. However, it is also very interesting to see that the heatmap shows that there is not much of a correlation between danceability and energy. We can explain this with different hypotheses. First, for a song to be danceable, it is usually achieved in three ways. Either the rhythm is brought by the drums,instruments or brought by the melody, the drums, or a combination. While the energy is usually mainly brought by the drums.

4. Finally, we also notice that higher loudness is associated with higher energy for most of the songs and vice versa. This brings up an important aspect about how the music is perceived. Whenever a sound loudness is increased, we tend to hear more energetic instruments. Hence energy is correlated with loudness.

In [ ]:
```
sns.scatterplot(data=df, x='tempo', y='energy', hue='Popular', alpha=0.6)
plt.title('Scatter Plot: Tempo vs. Energy')
plt.xlabel('Tempo (BPM)')
plt.ylabel('Energy')
plt.show()
```



First of all, the tempo of songs are usually between 60BPM to 170BPM. Most of the high BPM songs have high energy. This makes sense, since higher BPM means faster pace and more rythm. This increases the excitment and hence the energy.

4. Correlation analysis

Next stage of EDA is correlation analysis between each of the independent features and dependent feature.

1. Danceability vs. Popularity

In [ ]:
```
import scipy.stats as stats
```

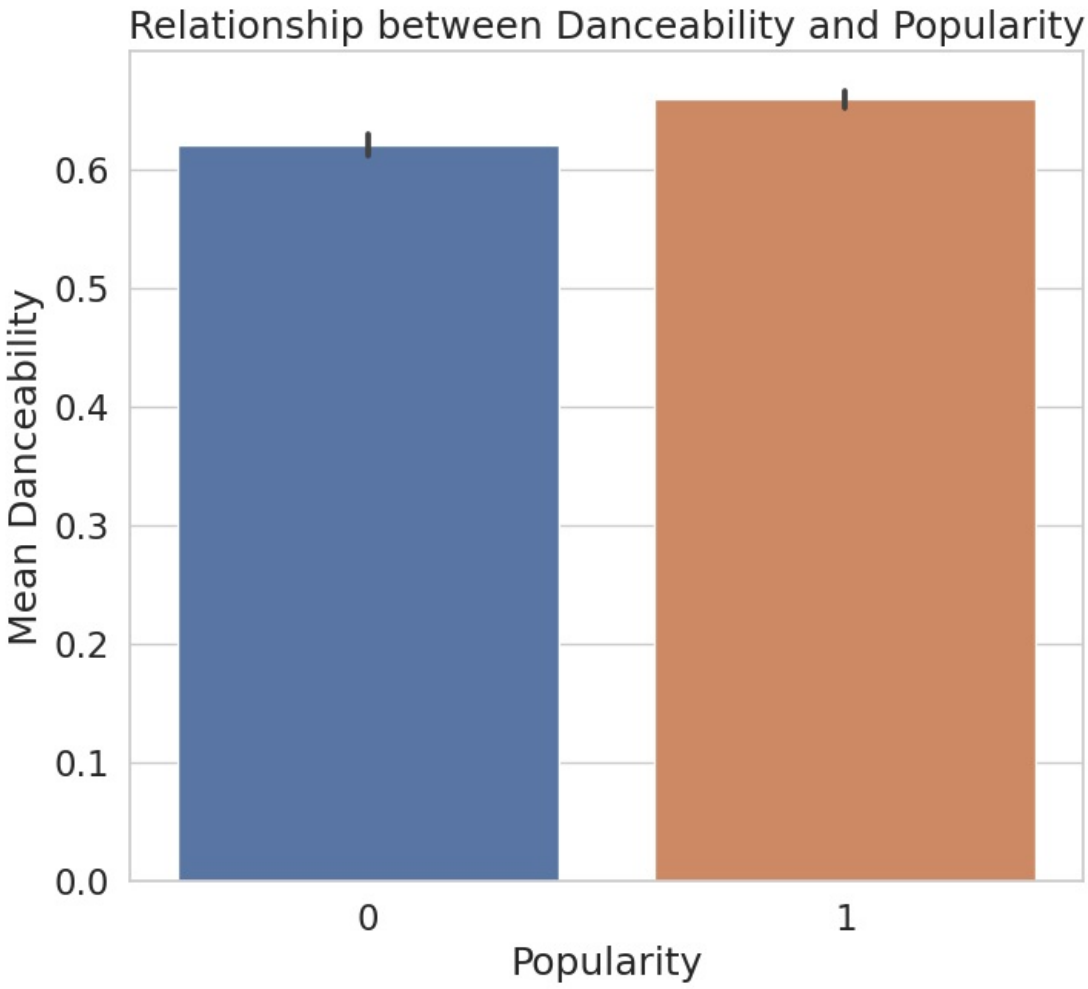
In [ ]:
```
np.isnan(df['danceability']).any()
```

Out[ ]:
```
True
```

In [ ]:
```
df['danceability'] = np.nan_to_num(df['danceability'])
```

In [ ]:
```
corr, p_value = stats.pearsonr(df['danceability'], df['Popular'])

print(f'Correlation between danceability and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='danceability')

plt.title('Relationship between Danceability and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Danceability')

plt.show()
```

```
Correlation between danceability and popularity: 0.11309124221912613
P-value: 2.45716247351586e-10
```



We can see a positive correlation between danceability and popularity of the song. Hence, higher danceability is usually associated with more popular songs. However, the correlation is quite weak.

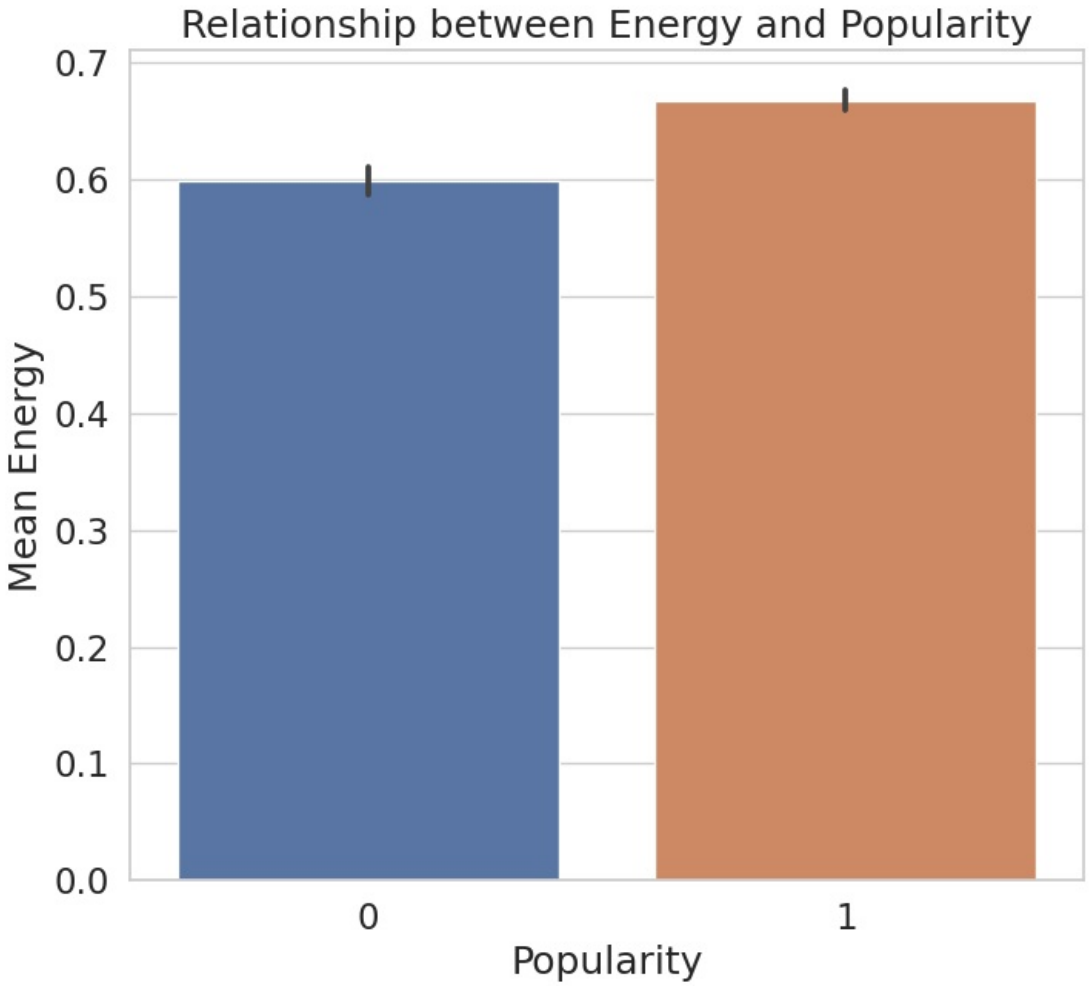
Small p-value indicates that this correlation is statistically significant.

2. Energy vs. Popularity

In [ ]:
```
df['energy'] = np.nan_to_num(df['energy'])
```

In [ ]:
```
corr, p_value = stats.pearsonr(df['energy'], df['Popular'])

print(f'Correlation between energy and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='energy')

plt.title('Relationship between Energy and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Energy')

plt.show()
```

```
Correlation between energy and popularity: 0.16354269960793452
P-value: 4.080171371691735e-20
```



The results show that there is a positive correlation between energy and popularity of a song. Correlation is relatively weak, as the value is close to 0.

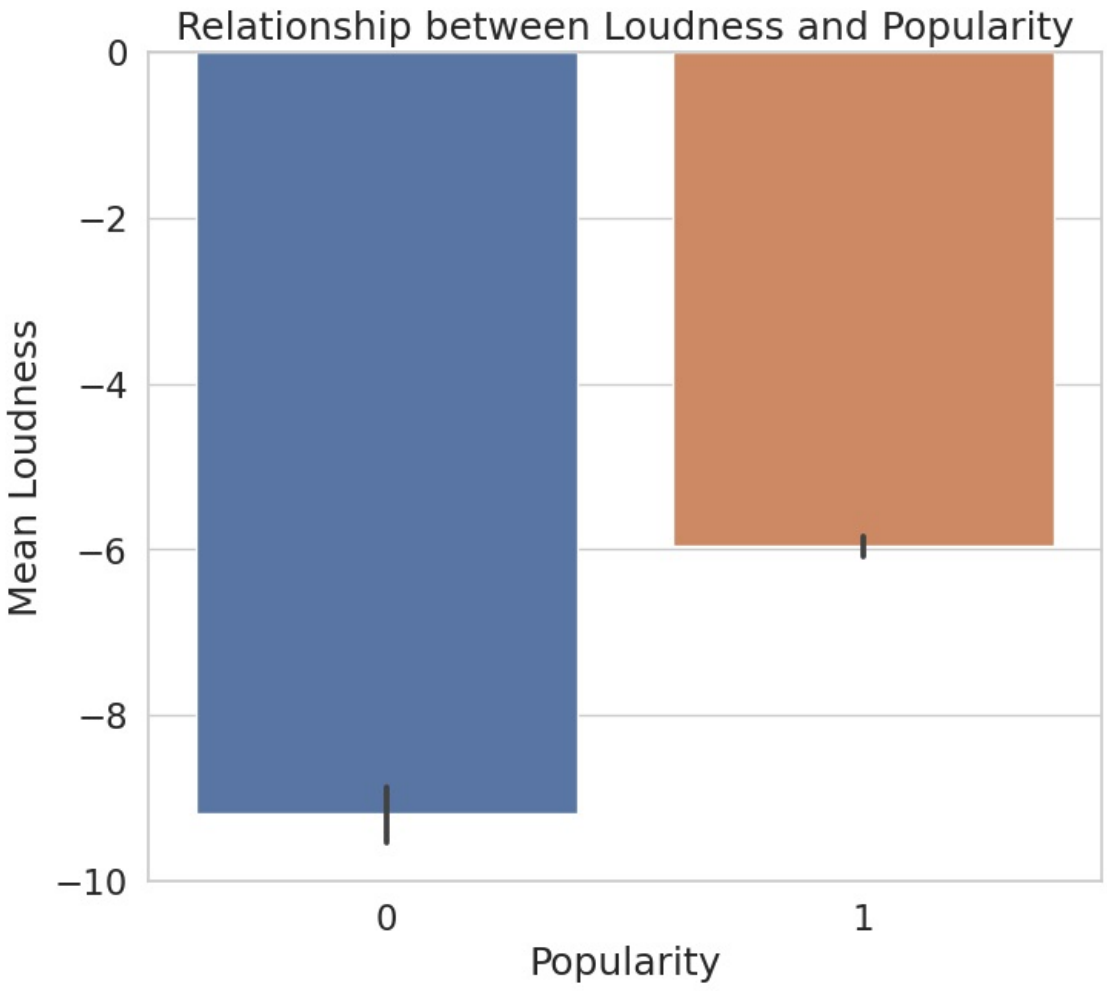
Small p-value shows that the correlation is statistically significant.

3. Loudness vs. Popularity

In [ ]:
```
df['loudness'] = np.nan_to_num(df['loudness'])
```

In [ ]:
```
corr, p_value = stats.pearsonr(df['loudness'], df['Popular'])

print(f'Correlation between loudness and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='loudness')

plt.title('Relationship between Loudness and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Loudness')

plt.show()
```

```
Correlation between loudness and popularity: 0.3018919369516627
P-value: 1.190106040088164e-66
```

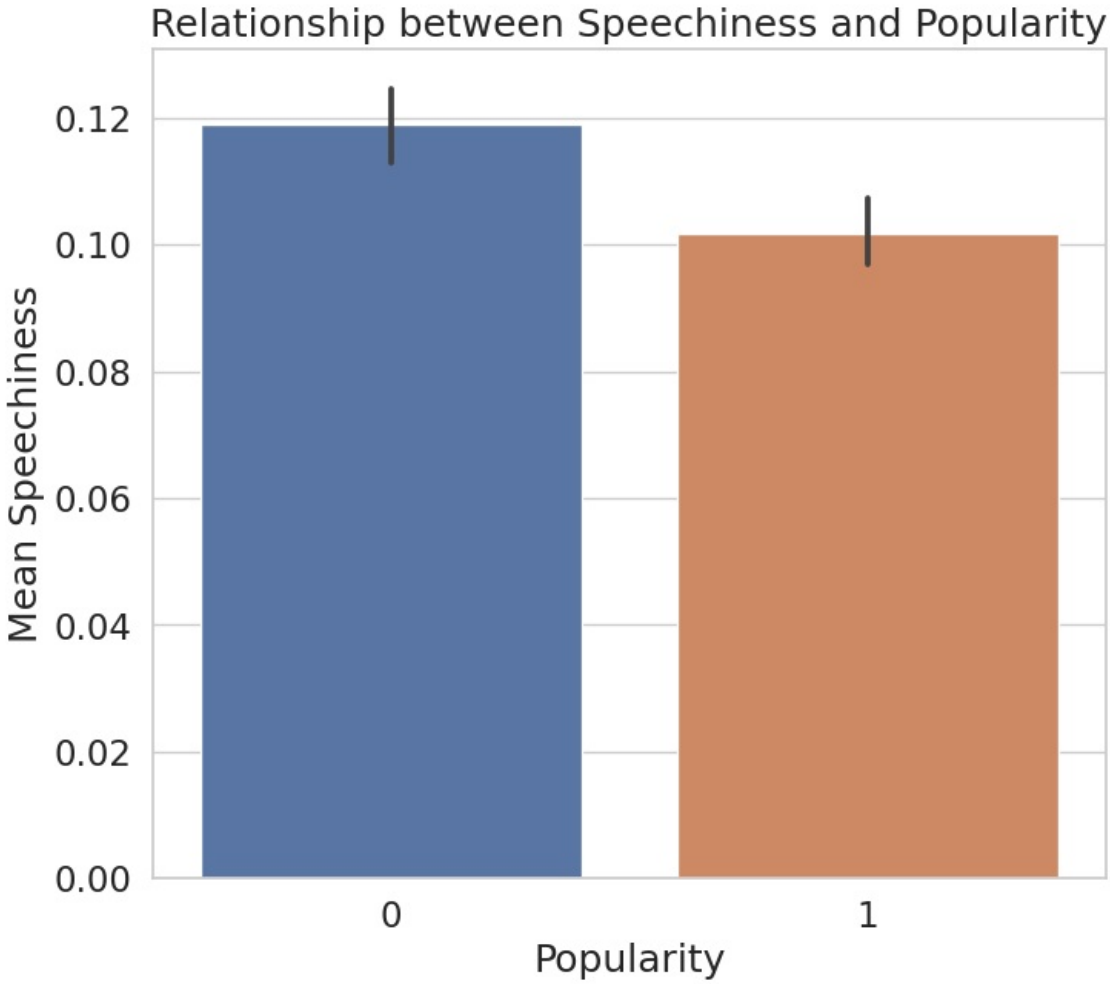


Results indicate that there is a positive correlation between loudness and popularity of a song. P-value value shows that the correlation is statistically significant.

4. Speechiness vs. Popularity

In [ ]:
```
df['speechiness'] = np.nan_to_num(df['speechiness'])
corr, p_value = stats.pearsonr(df['speechiness'], df['Popular'])

print(f'Correlation between speechiness and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='speechiness')

plt.title('Relationship between Speechiness and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Speechiness')

plt.show()
```

```
Correlation between speechiness and popularity: -0.07647490405270074
P-value: 1.930539345475327e-05
```

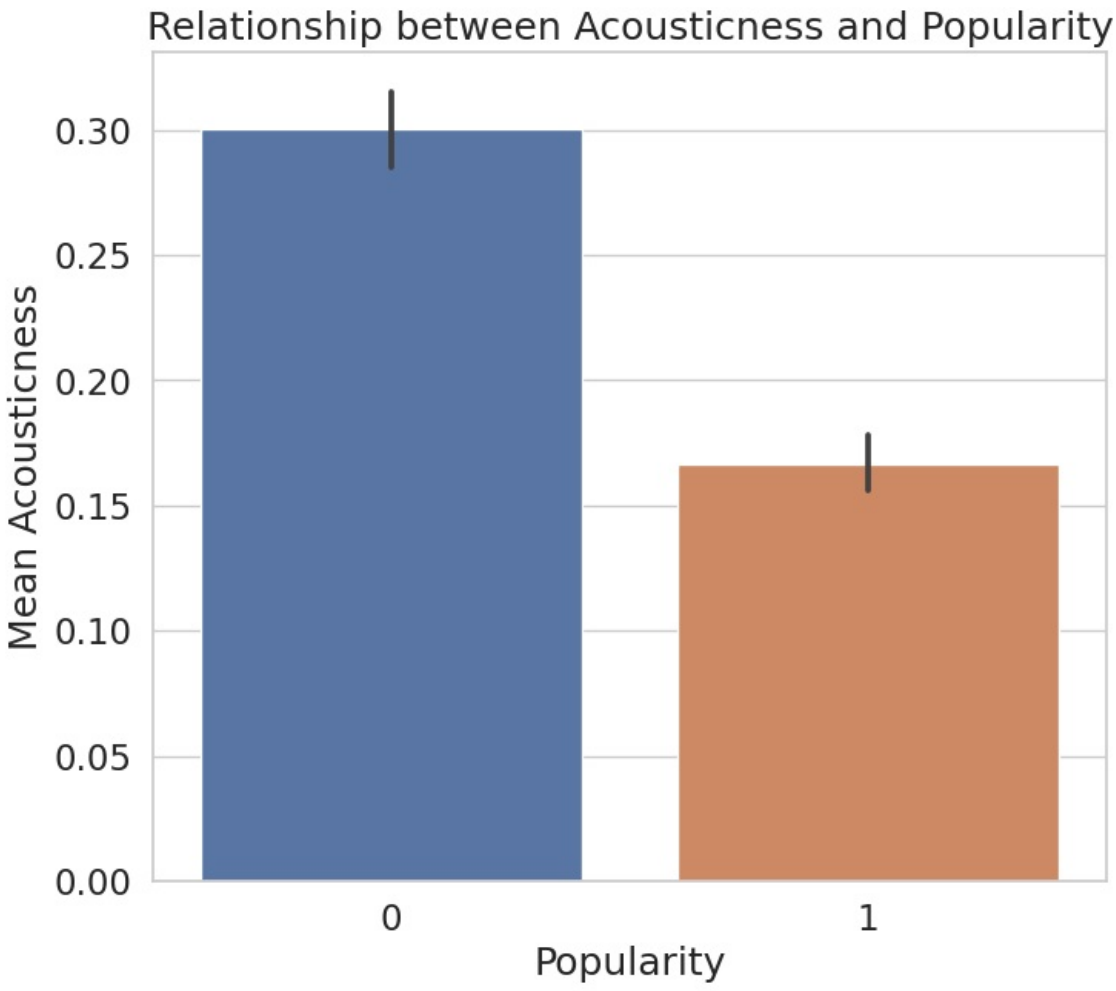


There is a weak negative correlation between speechiness and popularity. Hence, more popular songs tend to contain less spoken words in them. Small p-value indicates that the correlation is statistically significant.

5. Acousticness vs. Popularity

In [ ]:
```
df['acousticness'] = np.nan_to_num(df['acousticness'])
corr, p_value = stats.pearsonr(df['acousticness'], df['Popular'])

print(f'Correlation between acousticness and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='acousticness')

plt.title('Relationship between Acousticness and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Acousticness')

plt.show()
```

```
Correlation between acousticness and popularity: -0.23885912249551278
P-value: 1.158424539443365e-41
```

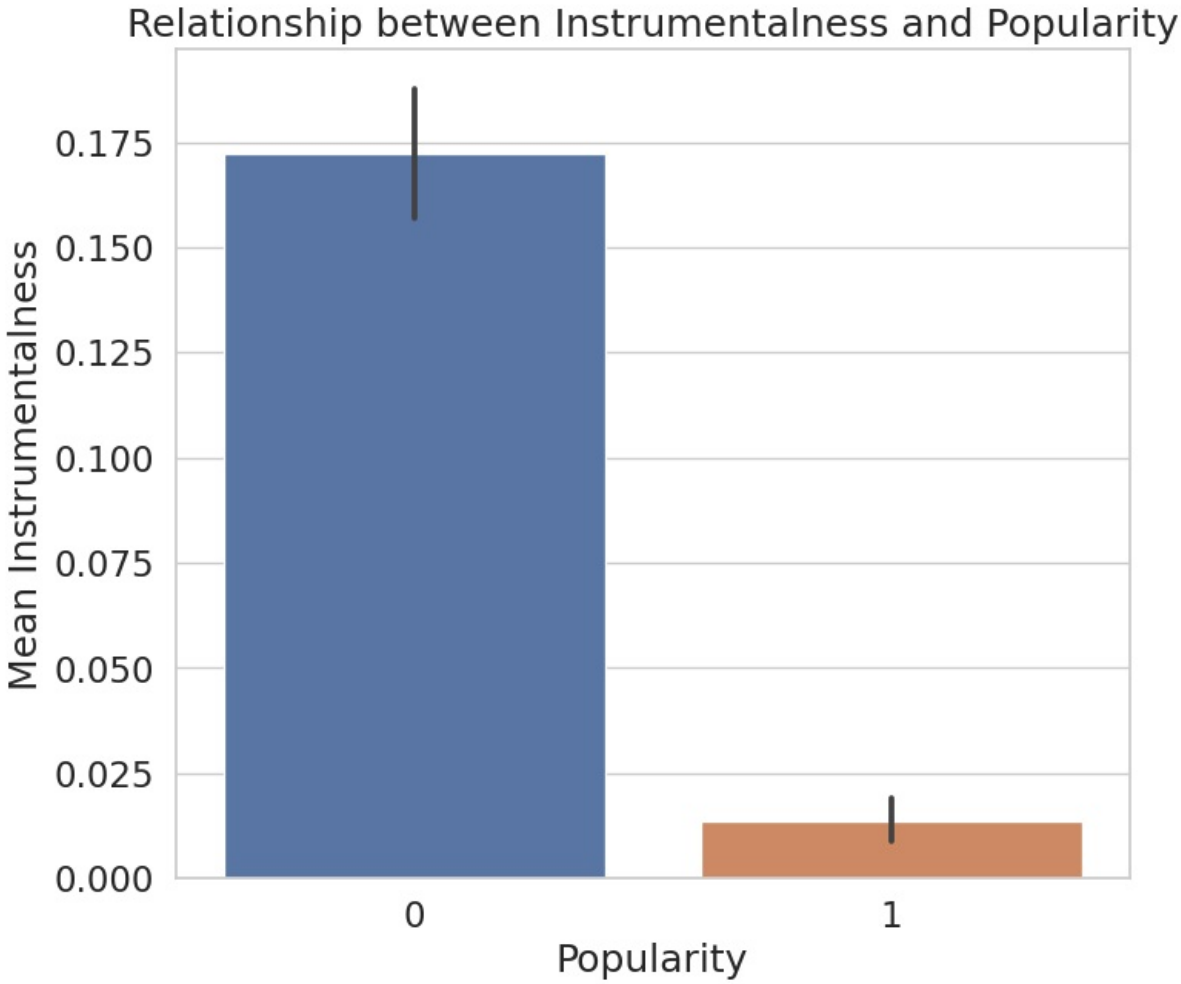


There is a negative correlation between acousticness and popularity. Therefore, popular songs tend to be less acoustic. Small p-value indicates that the correlations is statistically significant.

6. Instrumentalness vs. Popularity

In [ ]:
```
df['instrumentalness'] = np.nan_to_num(df['instrumentalness'])
corr, p_value = stats.pearsonr(df['instrumentalness'], df['Popular'])

print(f'Correlation between instrumentalness and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='instrumentalness')

plt.title('Relationship between Instrumentalness and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Instrumentalness')

plt.show()
```

```
Correlation between instrumentalness and popularity: -0.3014726024289189
P-value: 1.8379773204586386e-66
```



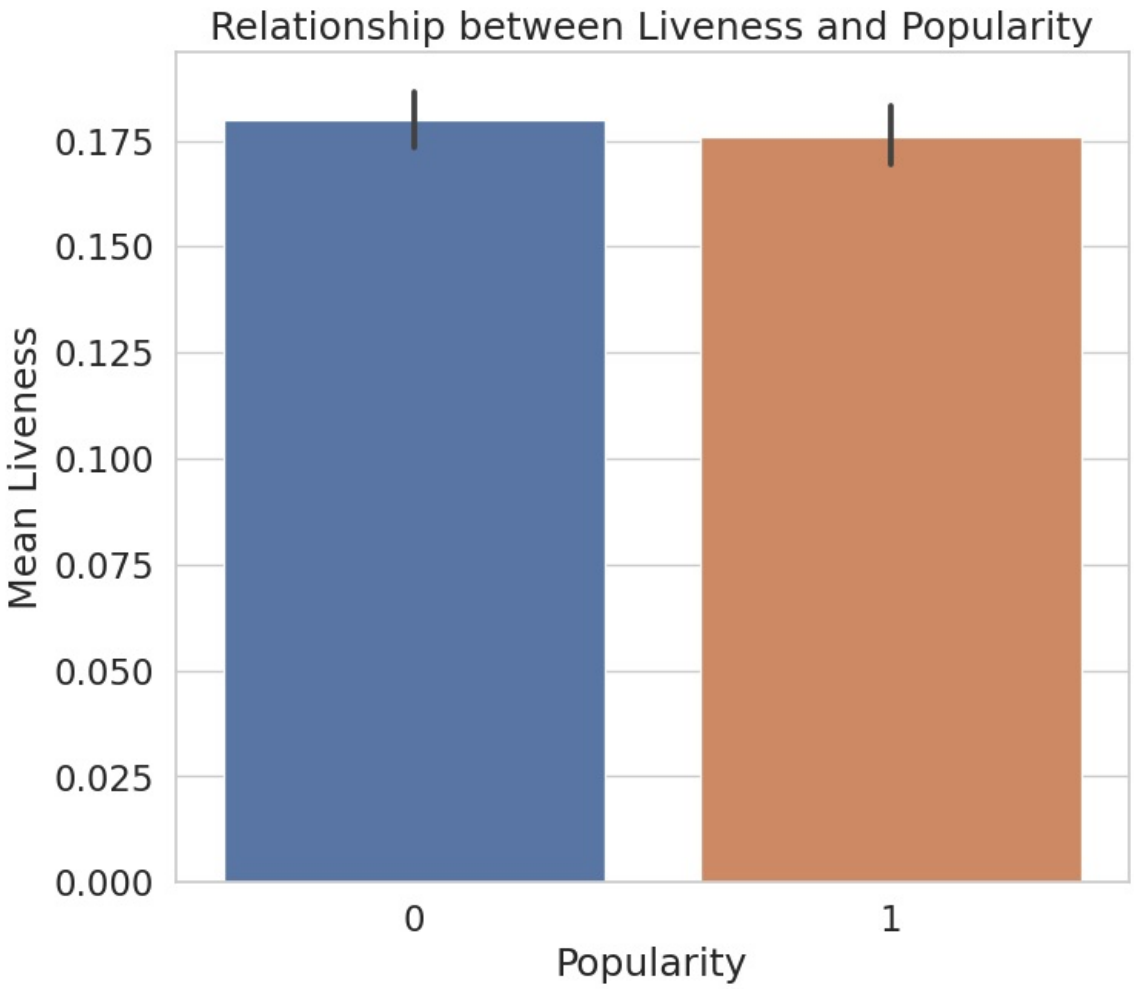
Results indicate that there is a negative correlation between instrumentalness and popularity. Therefore, popular songs tend to be less instrumental.

Small p-value indicates that the correlation is statistically significant.

7. Liveness vs. Popularity

In [ ]:
```
df['liveness'] = np.nan_to_num(df['liveness'])
corr, p_value = stats.pearsonr(df['liveness'], df['Popular'])

print(f'Correlation between liveness and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='liveness')

plt.title('Relationship between Liveness and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Liveness')

plt.show()
```

```
Correlation between liveness and popularity: -0.013694281462751104
P-value: 0.44484529291768904
```



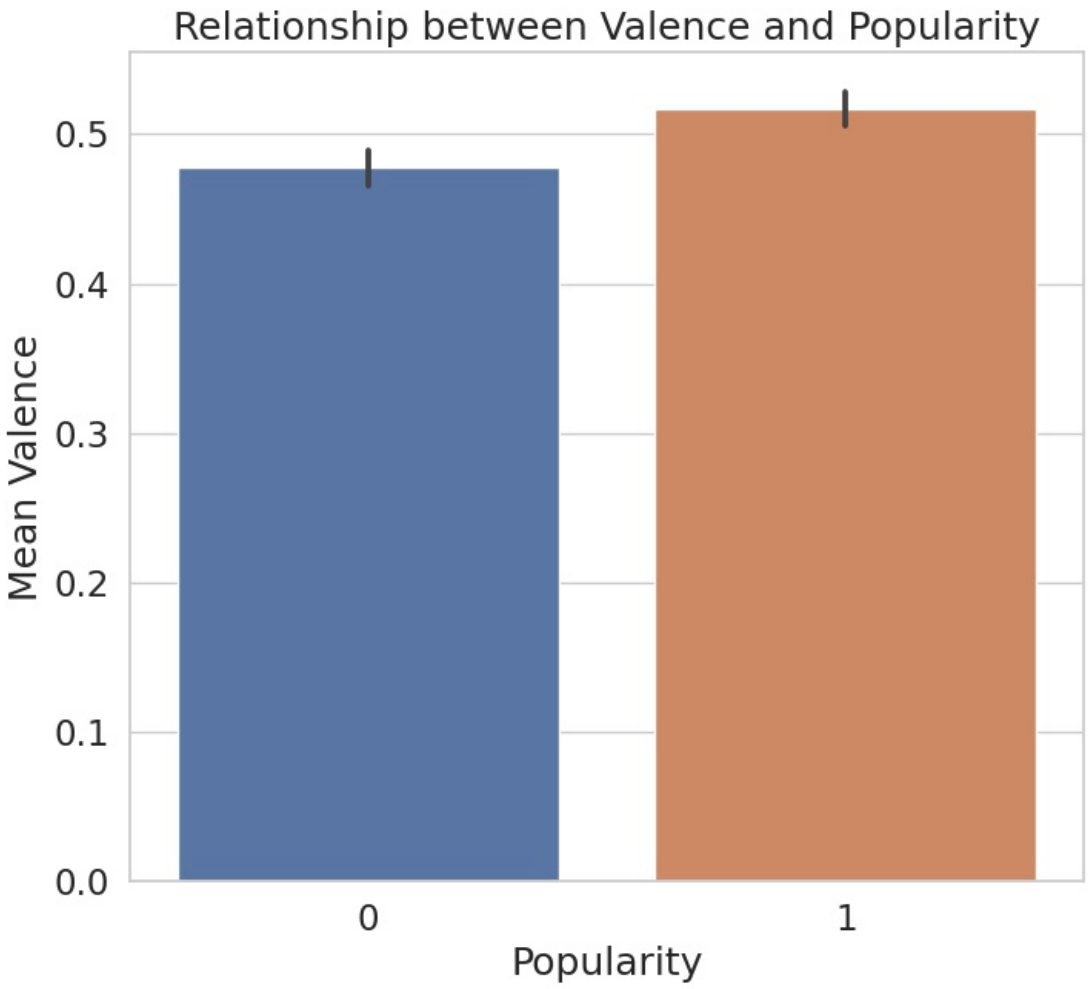
The results show that there is a weak negative correlation between song's liveness and popularity.

However, large p-value indicates that the correlation is not statistically significant.

8. Valence vs. Popularity

In [ ]:
```
df['valence'] = np.nan_to_num(df['valence'])
corr, p_value = stats.pearsonr(df['valence'], df['Popular'])

print(f'Correlation between valence and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='valence')

plt.title('Relationship between Valence and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Valence')

plt.show()
```

```
Correlation between valence and popularity: 0.08076852310803173
P-value: 6.3786411731203545e-06
```

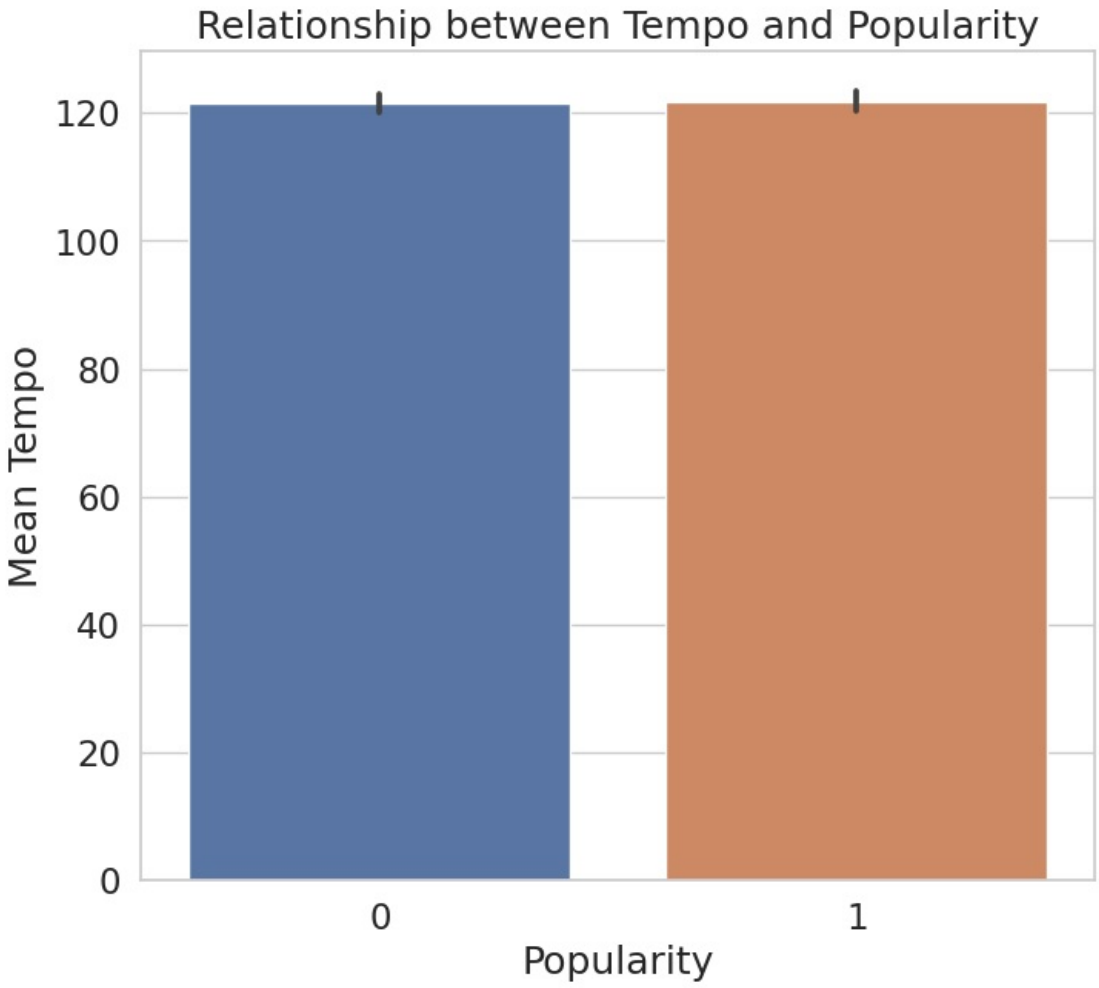


The results indicate that there is a weak positive correlation between valence and popularity. Small p-value shows that this correlation is statistically significant.

9. Tempo vs. Popularity

In [ ]:
```
df['tempo'] = np.nan_to_num(df['tempo'])
corr, p_value = stats.pearsonr(df['tempo'], df['Popular'])

print(f'Correlation between tempo and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='tempo')

plt.title('Relationship between Tempo and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Tempo')

plt.show()
```

```
Correlation between tempo and popularity: 0.005129814080056117
P-value: 0.7747302249796536
```

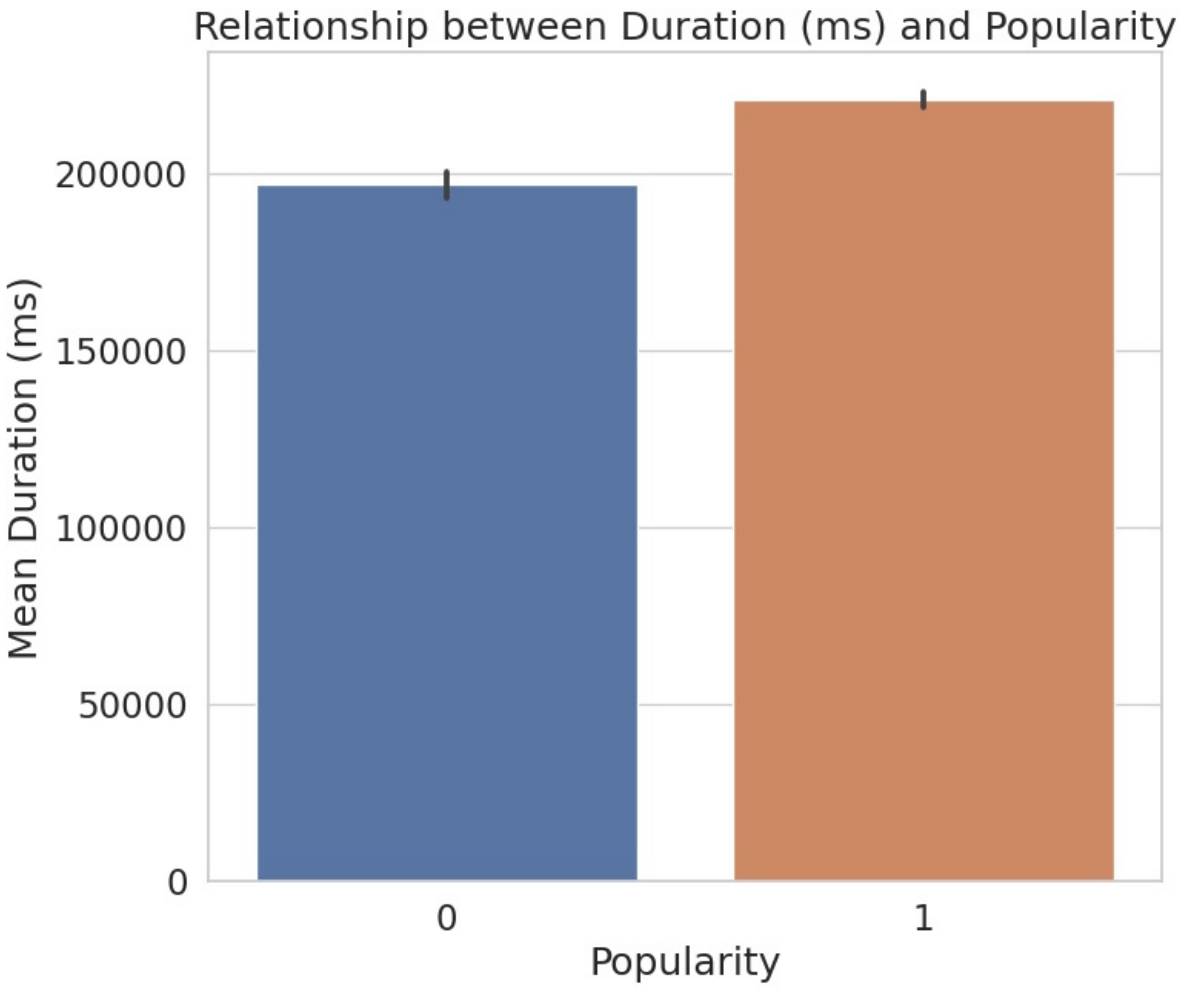


There is an extremely weak positive correlation, but it is not statistically significant.

10. Duration (ms) vs. Popularity

In [ ]:
```
df['duration_ms'] = np.nan_to_num(df['duration_ms'])
corr, p_value = stats.pearsonr(df['duration_ms'], df['Popular'])

print(f'Correlation between Duration (ms) and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='duration_ms')

plt.title('Relationship between Duration (ms) and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Mean Duration (ms)')

plt.show()
```

```
Correlation between Duration (ms) and popularity: 0.1798383613867128
P-value: 4.705393655838366e-24
```



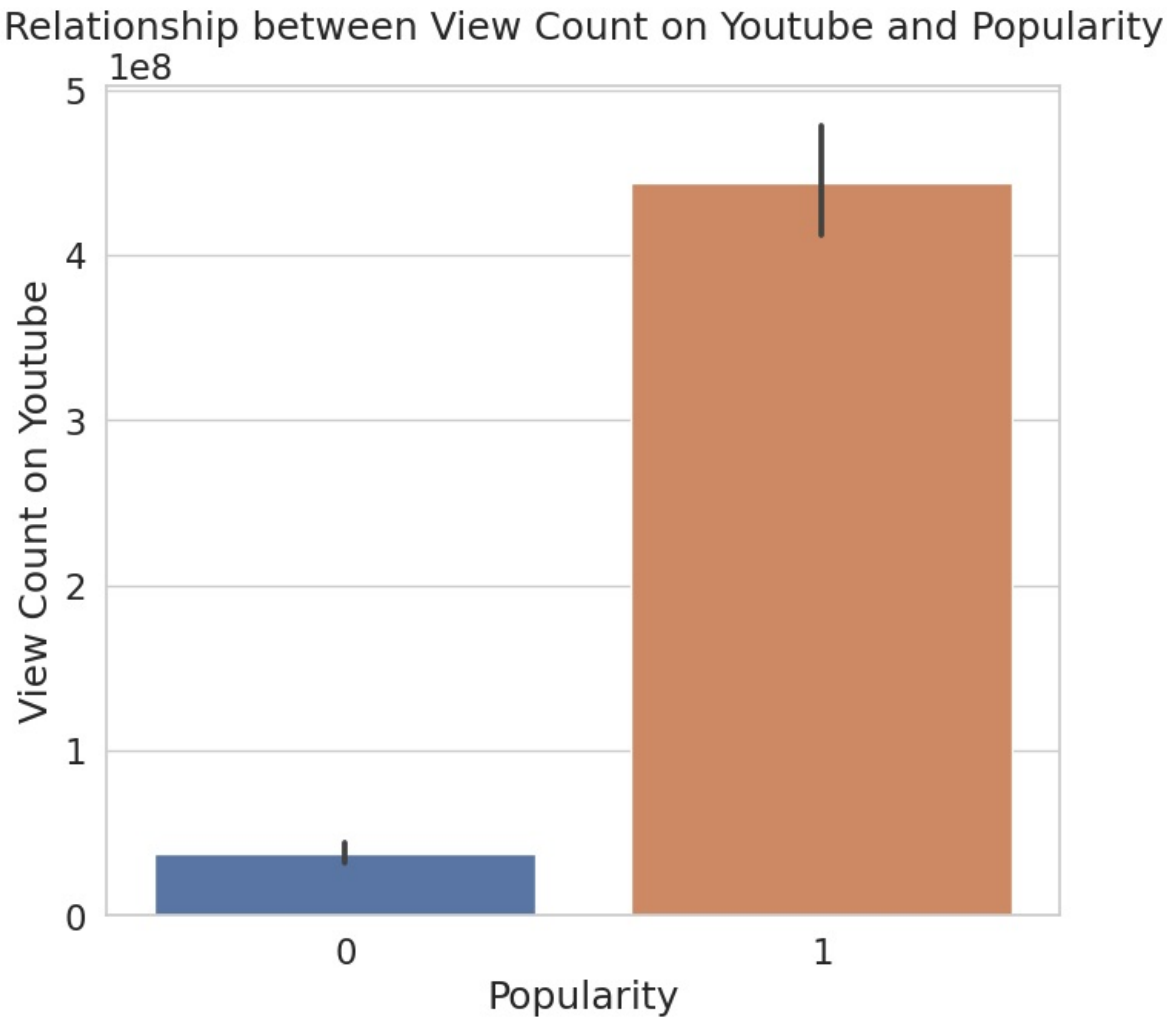
There is a positive correlation between duration in ms and song's popularity. Hence, popular songs tend to be longer compared to unpopular ones.

The correlation is statistically significant.

11. Youtube View Count

In [ ]:
```
df['youtube_view_count'] = np.nan_to_num(df['youtube_view_count'])
corr, p_value = stats.pearsonr(df['youtube_view_count'], df['Popular'])

print(f'Correlation between View Count on Youtube and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='youtube_view_count')

plt.title('Relationship between View Count on Youtube and Popularity')
plt.xlabel('Popularity')
plt.ylabel('View Count on Youtube')

plt.show()
```

```
Correlation between View Count on Youtube and popularity: 0.41204077461417626
P-value: 5.832473379190261e-128
```



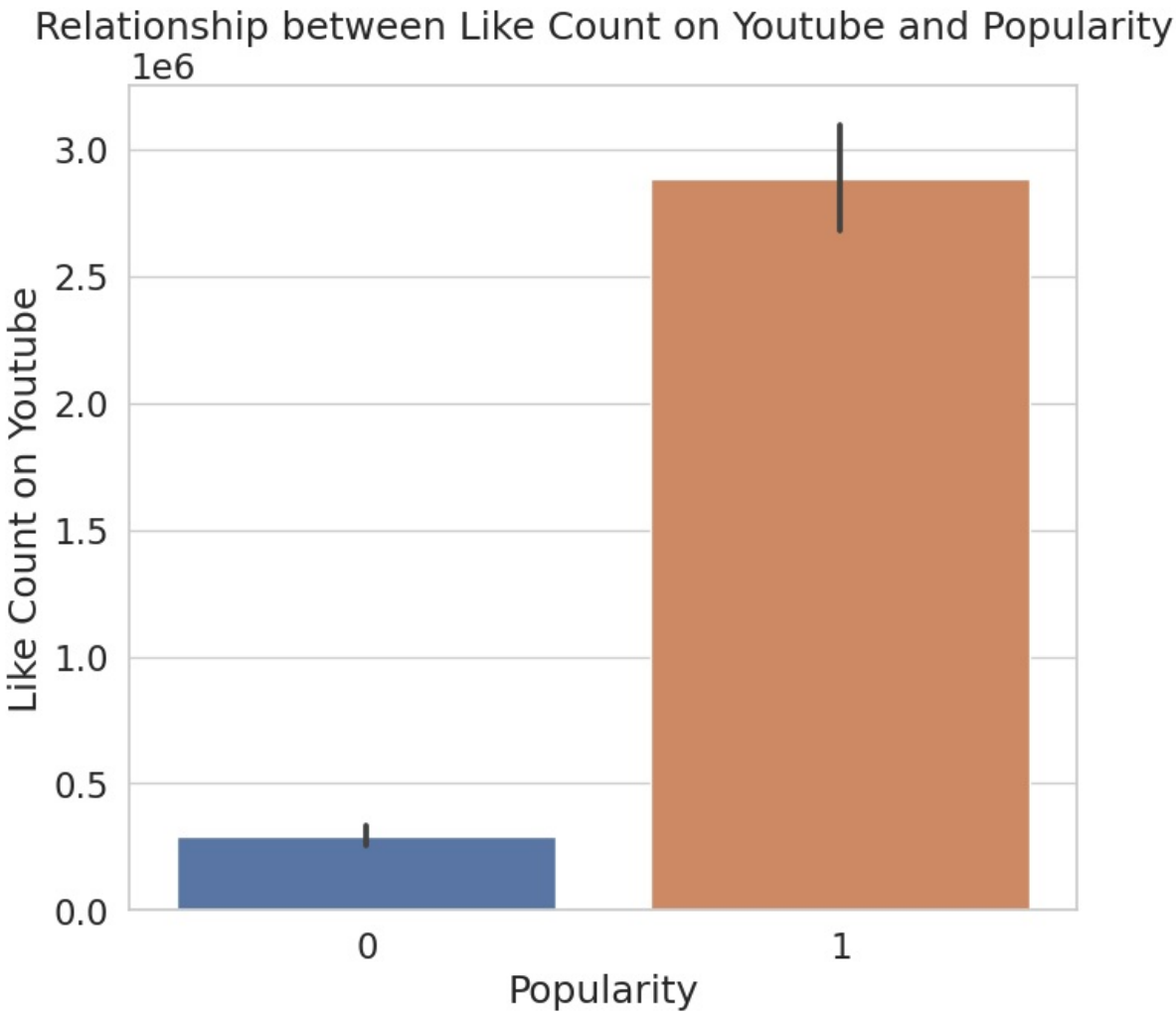
The results show a strong positive correlation between view count on Youtube and popularity of the song.

Extremely small p-value shows that the correlation is statistically significant.

12. Youtube Like Count

In [ ]:
```
df['youtube_like_count'] = np.nan_to_num(df['youtube_like_count'])
corr, p_value = stats.pearsonr(df['youtube_like_count'], df['Popular'])

print(f'Correlation between Like Count on Youtube and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='youtube_like_count')

plt.title('Relationship between Like Count on Youtube and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Like Count on Youtube')

plt.show()
```

```
Correlation between Like Count on Youtube and popularity: 0.4302807415121072
P-value: 1.319420003722745e-140
```



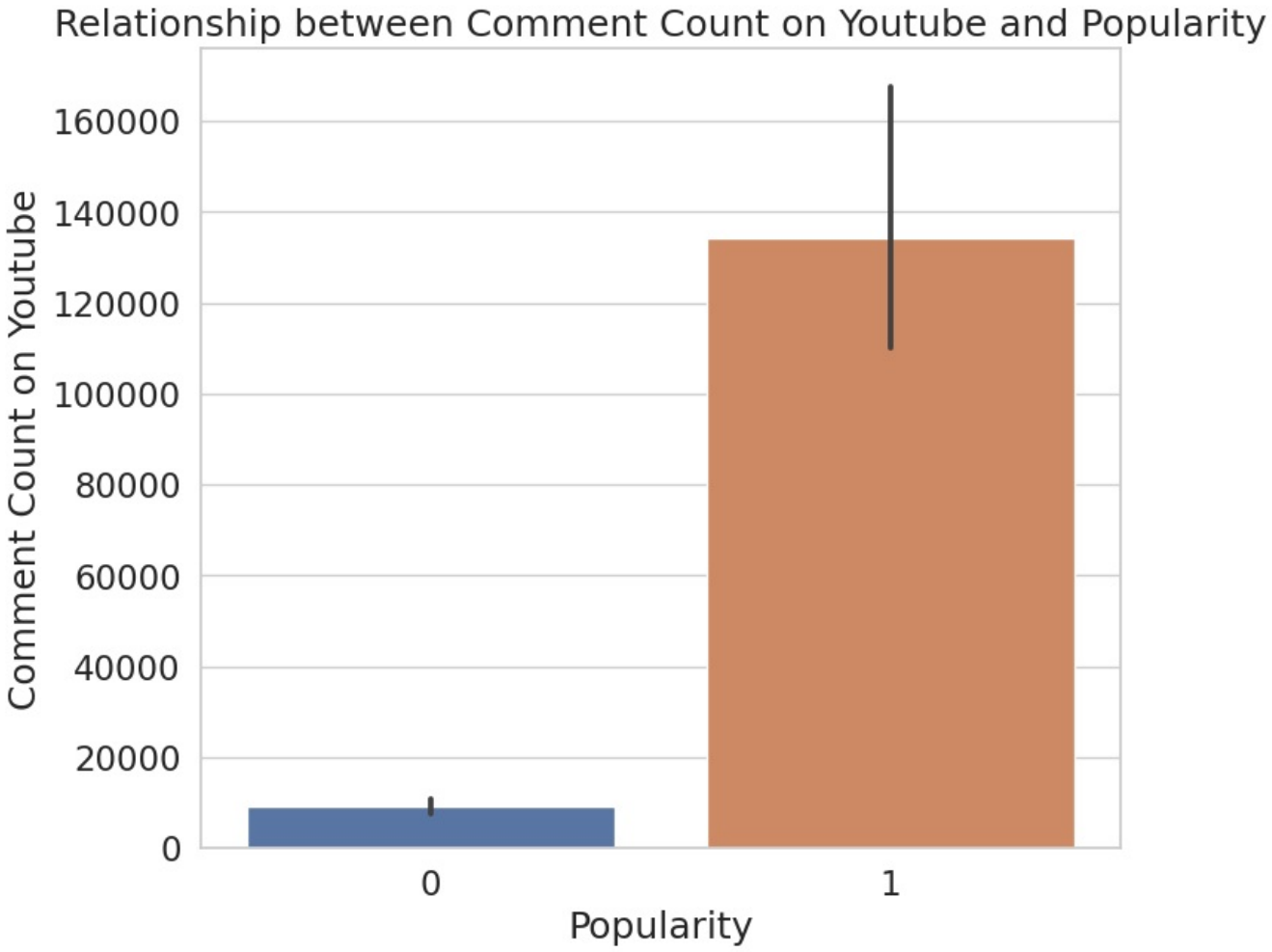
The results show a strong positive correlation between like count on Youtube and popularity of the song.

Extremely small p-value shows that the correlation is statistically significant.

13. Youtube Comment Count

In [ ]:
```
df['youtube_comment_count'] = np.nan_to_num(df['youtube_comment_count'])
corr, p_value = stats.pearsonr(df['youtube_comment_count'], df['Popular'])

print(f'Correlation between Comment Count on Youtube and popularity: {corr}')
print(f'P-value: {p_value}')

sns.barplot(data=df, x='Popular', y='youtube_comment_count')

plt.title('Relationship between Comment Count on Youtube and Popularity')
plt.xlabel('Popularity')
plt.ylabel('Comment Count on Youtube')

plt.show()
```

```
Correlation between Comment Count on Youtube and popularity: 0.16608032805482678
P-value: 1.0541983418674177e-20
```



The results show a strong positive correlation between comment count on Youtube and popularity of the song.

Extremely small p-value shows that the correlation is statistically significant.