In [46]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

In [47]:
```
df=pd.read_excel("C:\\Users\\KAUSHIK\\OneDrive\\spotify.xlsx")
```

In [48]:
```
df.head(5)
```

Out[48]:

	track_id	track_name	track_artist	track_popularity	track_a
0	6f807x0ima9a1j3VPbc7VN	I Don't Care (with Justin Bieber) - Loud Luxur...	Ed Sheeran	66	2oCs0DGTsRO98G
1	0r7CVbZTWZgbTCYdfa2P31	Memories - Dillon Francis Remix	Maroon 5	67	63rPSO264uRjW1X
2	1z1Hg7Vb0AhHDiEmnDE79l	All the Time - Don Diablo Remix	Zara Larsson	70	1HoSmj2eLcsrR0
3	75FpbthrwQmzHlBJLuGdC7	Call You Mine - Keanu Silva Remix	The Chainsmokers	60	1nqYsOef1yKKuGC
4	1e8PAfcKUYoKkxPhrHqw4x	Someone You Loved - Future Humans Remix	Lewis Capaldi	69	7m7vv9wlQ4i0LF

5 rows × 23 columns

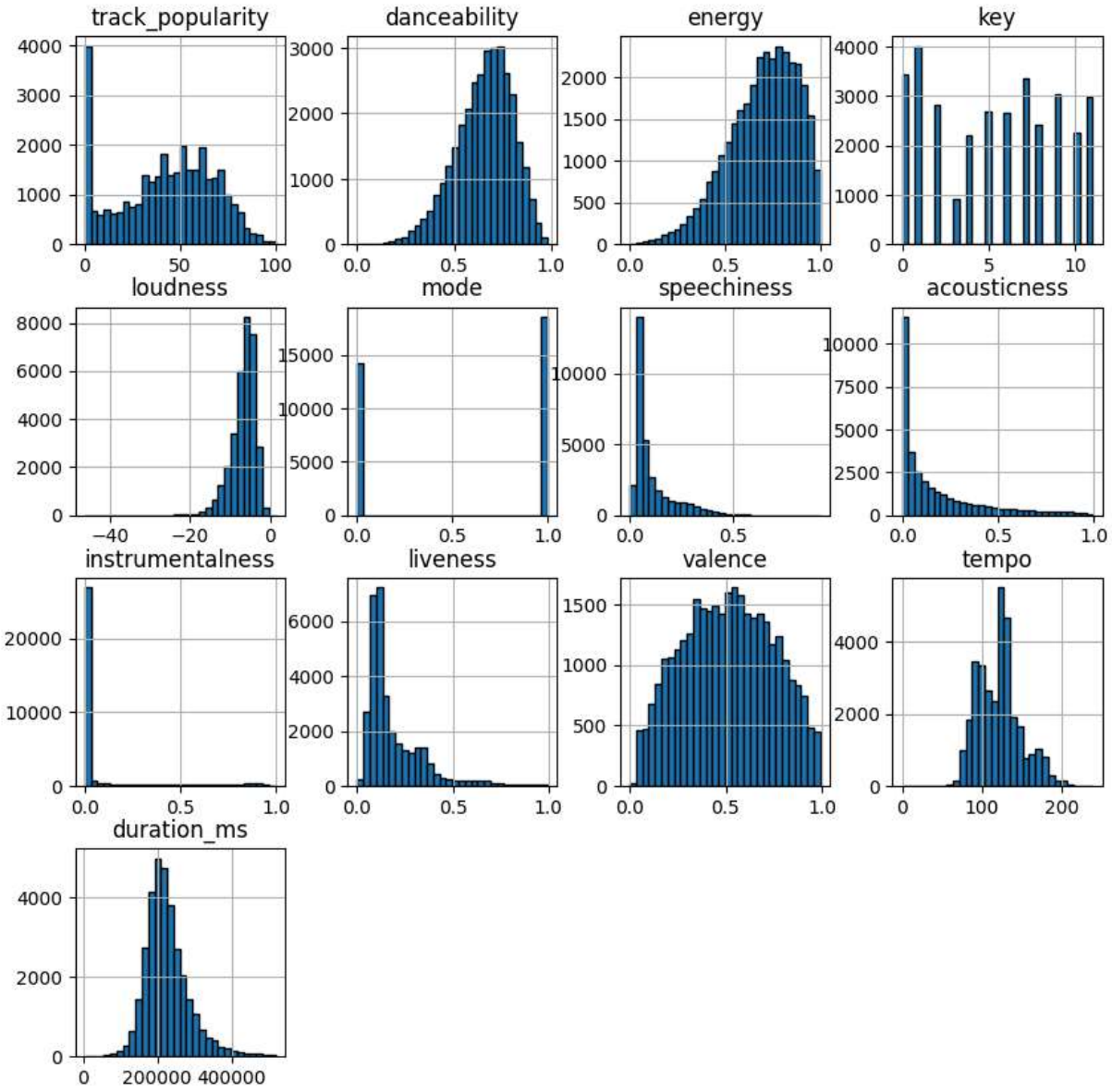
In [49]:
```
#preprocessing
df.isnull().sum()
df.dropna(inplace=True)
```

In [50]:
```
#CLeared the data by removing rows with null values
df.isnull().sum()
```

Out[50]:
```
track_id                    0
track_name                  0
track_artist                0
track_popularity            0
track_album_id              0
track_album_name            0
track_album_release_date    0
playlist_name               0
playlist_id                 0
playlist_genre              0
playlist_subgenre           0
danceability                0
energy                      0
key                         0
loudness                    0
mode                        0
speechiness                 0
acousticness                0
instrumentalness            0
liveness                    0
valence                     0
tempo                       0
duration_ms                 0
dtype: int64
```

In [51]:
```
df.hist(figsize=(10, 10), bins=30, edgecolor='black')
```

Out[51]:
```
array([[<Axes: title={'center': 'track_popularity'}>,
        <Axes: title={'center': 'danceability'}>,
        <Axes: title={'center': 'energy'}>,
        <Axes: title={'center': 'key'}>],
       [<Axes: title={'center': 'loudness'}>,
        <Axes: title={'center': 'mode'}>,
        <Axes: title={'center': 'speechiness'}>,
        <Axes: title={'center': 'acousticness'}>],
       [<Axes: title={'center': 'instrumentalness'}>,
        <Axes: title={'center': 'liveness'}>,
        <Axes: title={'center': 'valence'}>,
        <Axes: title={'center': 'tempo'}>],
       [<Axes: title={'center': 'duration_ms'}>, <Axes: >, <Axes: >,
        <Axes: >]], dtype=object)
```



In [52]:
```
df.drop_duplicates(inplace=True)
#Removed duplicate rows
df.duplicated().sum()
```

Out[52]: np.int64(0)

In [53]:
```
df['track_album_release_date'] = pd.to_datetime(df['track_album_release_date'], err
df['track_album_release_date'].isnull().sum()
```

Out[53]: np.int64(0)

In [54]:
```
from sklearn.preprocessing import LabelEncoder

df_encoded = df.copy()
label_encoders = {}

# Only encode columns of type 'object' (strings)
for col in df_encoded.select_dtypes(include=['object']).columns:
    # Check if all values are strings (skip columns with mixed types)
```

```
    if df_encoded[col].apply(lambda x: isinstance(x, str)).all():
        le = LabelEncoder()
        df_encoded[col] = le.fit_transform(df_encoded[col])
        label_encoders[col] = le

df_encoded.head()
```

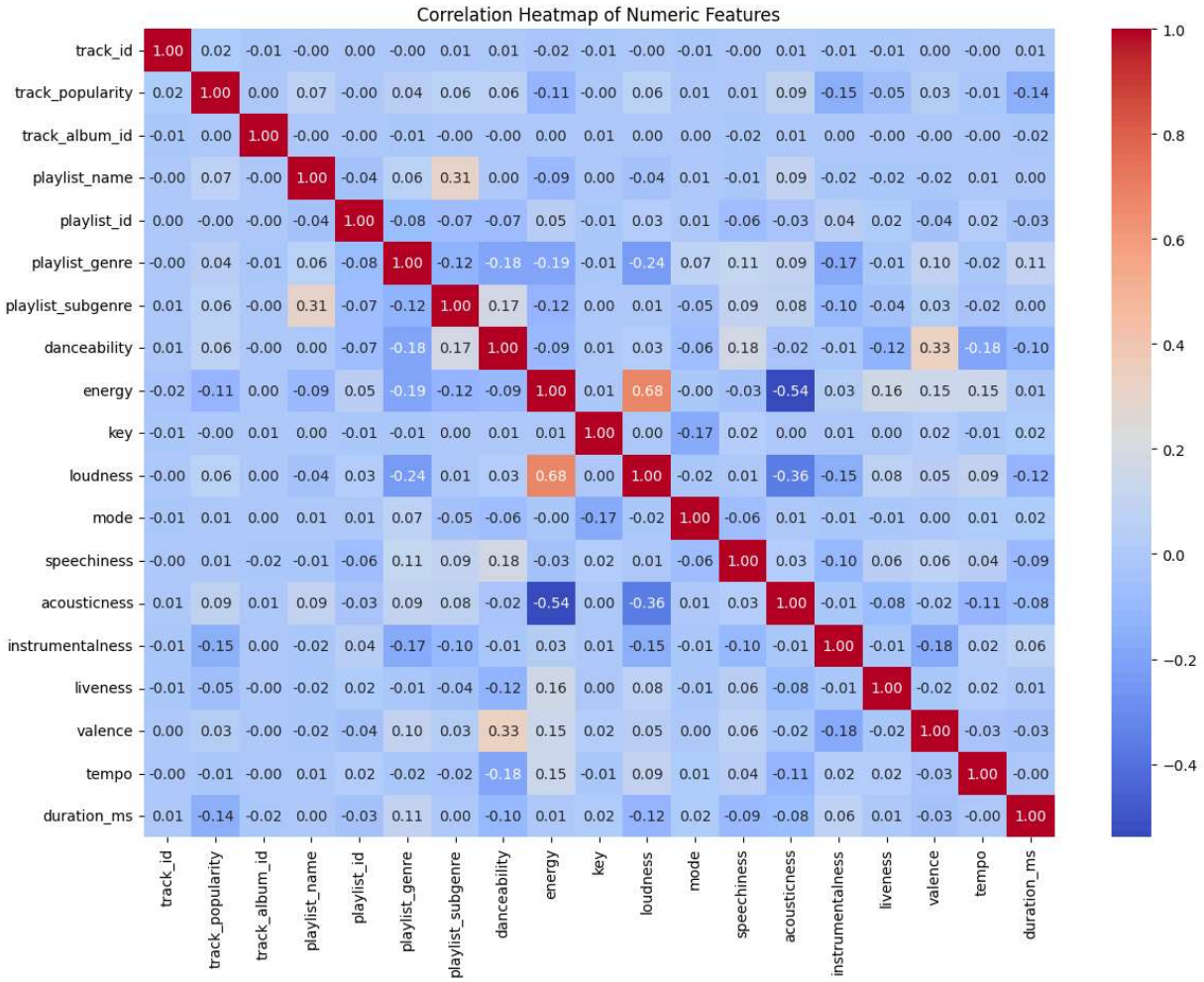
Out[54]:

	track_id	track_name	track_artist	track_popularity	track_album_id	track_album_name
0	24145	I Don't Care (with Justin Bieber) - Loud Luxur...	Ed Sheeran	66	8225	I Don't Care (with Justin Bieber) [Loud Luxury...
1	3061	Memories - Dillon Francis Remix	Maroon 5	67	17649	Memories (Dillon Francis Remix)
2	7219	All the Time - Don Diablo Remix	Zara Larsson	70	3798	All the Time (Don Diablo Remix)
3	25694	Call You Mine - Keanu Silva Remix	The Chainsmokers	60	5293	Call You Mine - The Remixes
4	5987	Someone You Loved - Future Humans Remix	Lewis Capaldi	69	21933	Someone You Loved (Future Humans Remix)

5 rows × 23 columns

In [55]:
```
# Select only numeric columns for correlation
numeric_df = df_encoded.select_dtypes(include=[np.number])
```

In [56]:
```
# Plot a heatmap of the correlation matrix for numeric_df
plt.figure(figsize=(14, 10))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap of Numeric Features')
plt.show()
```



In [57]:
```
numeric_df.corr()
```

Out[57]:

	track_id	track_popularity	track_album_id	playlist_name	playlist_id	pla
track_id	1.000000	0.018492	-0.010503	-0.001172	0.000599	
track_popularity	0.018492	1.000000	0.000799	0.069334	-0.001408	
track_album_id	-0.010503	0.000799	1.000000	-0.000002	-0.001897	
playlist_name	-0.001172	0.069334	-0.000002	1.000000	-0.036640	
playlist_id	0.000599	-0.001408	-0.001897	-0.036640	1.000000	
playlist_genre	-0.003941	0.042510	-0.012866	0.059686	-0.080788	
playlist_subgenre	0.006048	0.058148	-0.000367	0.308943	-0.068884	
danceability	0.007422	0.064758	-0.001633	0.002780	-0.069619	
energy	-0.023146	-0.108980	0.000865	-0.089547	0.045322	
key	-0.006866	-0.000395	0.006816	0.002966	-0.005746	
loudness	-0.003562	0.057715	0.003007	-0.035545	0.026522	
mode	-0.008030	0.010561	0.000923	0.011924	0.010604	
speechiness	-0.002628	0.007053	-0.021076	-0.014755	-0.062904	
acousticness	0.011683	0.085042	0.009578	0.090657	-0.026957	
instrumentalness	-0.006063	-0.150001	0.004659	-0.019119	0.043887	
liveness	-0.006750	-0.054599	-0.001036	-0.019488	0.016421	
valence	0.001291	0.033276	-0.000062	-0.021979	-0.042646	
tempo	-0.002779	-0.005557	-0.000641	0.009649	0.023292	
duration_ms	0.006342	-0.143638	-0.021032	0.002054	-0.031673	

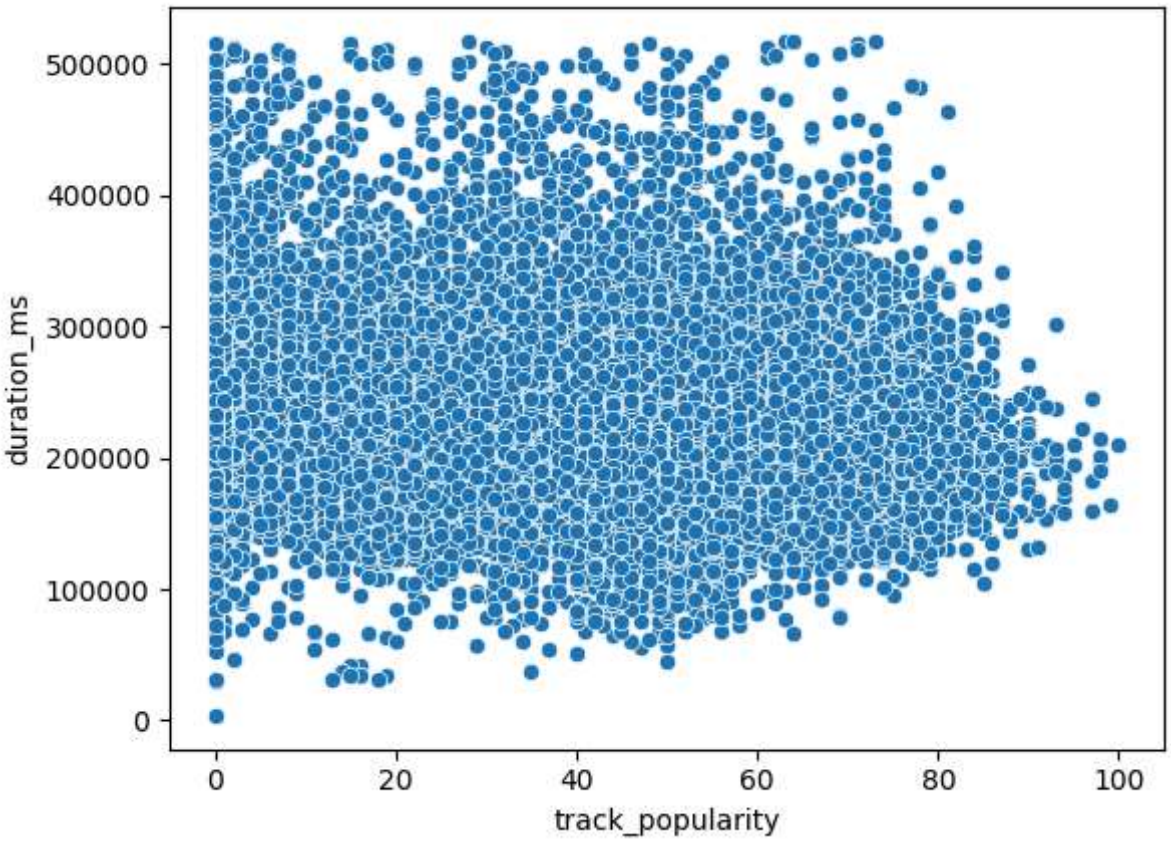
In [58]:
```
numeric_df.describe()
```

Out[58]:

	track_id	track_popularity	track_album_id	playlist_name	playlist_id	playlist
count	32827.000000	32827.000000	32827.000000	32827.000000	32827.000000	32827.
mean	14212.861943	42.483383	11244.872970	223.199866	233.690986	2.
std	8191.826044	24.980838	6503.101691	131.210157	135.794228	1.
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.
25%	7113.500000	24.000000	5607.500000	111.000000	112.000000	1.
50%	14210.000000	45.000000	11231.000000	219.000000	239.000000	2.
75%	21337.500000	62.000000	16826.500000	337.500000	349.000000	4.
max	28350.000000	100.000000	22541.000000	448.000000	470.000000	5.

In [59]:
```
sns.scatterplot(x='track_popularity', y='duration_ms', data=numeric_df)
```

Out[59]:
```
<Axes: xlabel='track_popularity', ylabel='duration_ms'>
```



In [60]:
```
numeric_df
```

Out[60]:

	track_id	track_popularity	track_album_id	playlist_name	playlist_id	playlist_genre
0	24145	66	8225	292	235	2
1	3061	67	17649	292	235	2
2	7219	70	3798	292	235	2
3	25694	60	5293	292	235	2
4	5987	69	21933	292	235	2
...	...	...	...	...	...	...
32828	26851	42	7586	443	420	0
32829	18772	20	19609	443	420	0
32830	26460	14	2263	443	420	0
32831	10083	15	4914	443	420	0
32832	7864	27	1558	443	420	0

32827 rows × 19 columns

In [61]:
```
#Implementing KMeans clustering
from sklearn.cluster import KMeans
```

In [62]:
```
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(numeric_df[['track_popularity', 'duration_ms']])
```

Out[62]:

KMeans

```
KMeans(n_clusters=3, random_state=42)
```

In [63]:
```
numeric_df['cluster'] = kmeans.labels_
```

In [64]:
```
numeric_df
```

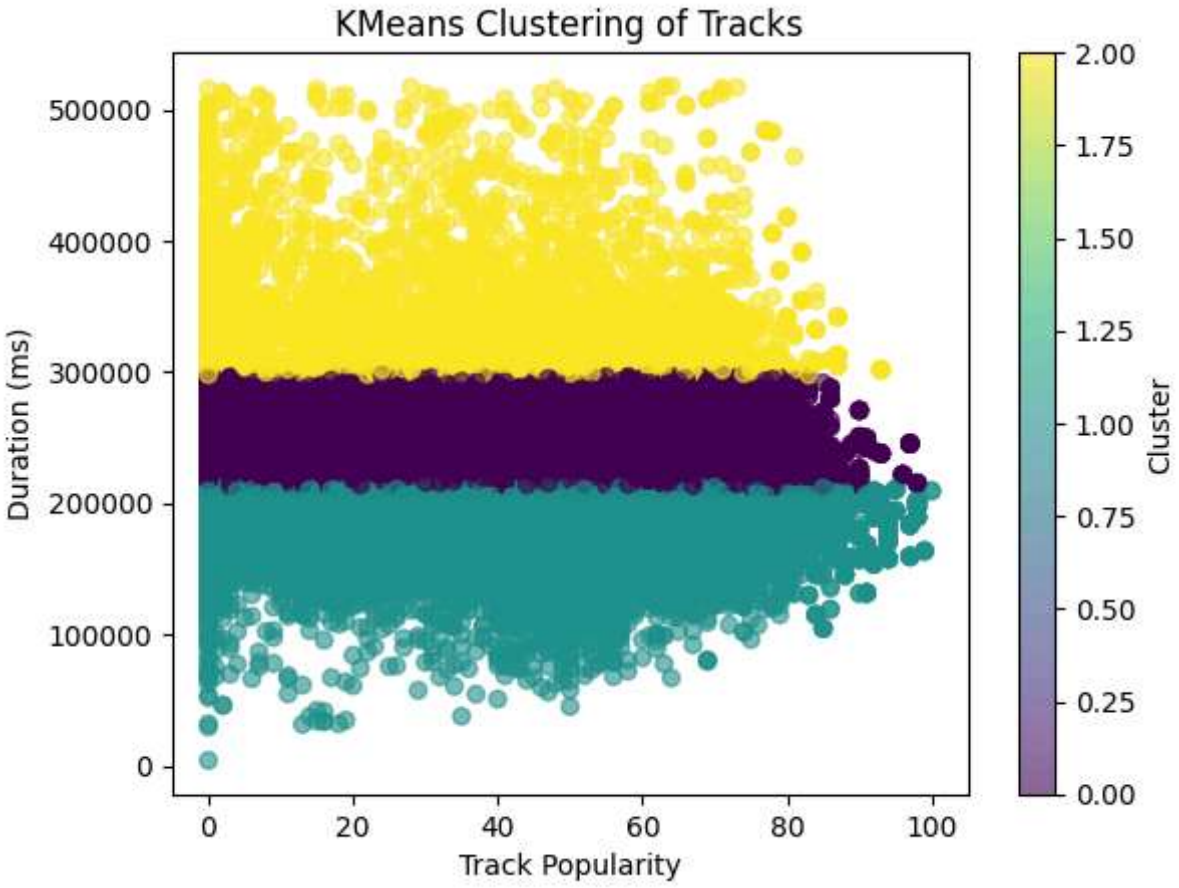
Out[64]:

	track_id	track_popularity	track_album_id	playlist_name	playlist_id	playlist_genre
0	24145	66	8225	292	235	2
1	3061	67	17649	292	235	2
2	7219	70	3798	292	235	2
3	25694	60	5293	292	235	2
4	5987	69	21933	292	235	2
...	...	...	...	...	...	...
32828	26851	42	7586	443	420	0
32829	18772	20	19609	443	420	0
32830	26460	14	2263	443	420	0
32831	10083	15	4914	443	420	0
32832	7864	27	1558	443	420	0

32827 rows × 20 columns

In [65]:

```
plt.scatter(numeric_df['track_popularity'], numeric_df['duration_ms'], c=numeric_df
plt.xlabel('Track Popularity')
plt.ylabel('Duration (ms)')
plt.title('KMeans Clustering of Tracks')
plt.colorbar(label='Cluster')
plt.show()
```



In [66]:
```python
# Recommending similar tracks using Nearest Neighbors
import warnings
from sklearn.neighbors import NearestNeighbors

# We'll use the numeric_df for recommendations (excluding the 'cluster' column)
features = numeric_df.drop(columns=['cluster'])

# Fit NearestNeighbors model
nn_model = NearestNeighbors(n_neighbors=6, metric='euclidean')
nn_model.fit(features)

def recommend_tracks(track_index, n_recommendations=5):
    distances, indices = nn_model.kneighbors([features.iloc[track_index]], n_neighb
    # Exclude the first result (itself)
    recommended_indices = indices[0][1:]
    return df.iloc[recommended_indices][['track_name', 'track_artist', 'track_album
warnings.filterwarnings("ignore")
# Example: Recommend tracks similar to the first track
recommend_tracks(0)
```

Out[66]:

	track_name	track_artist	track_album_name	track_popularity
29684	I Don't Care (with Justin Bieber) - Loud Luxur...	Ed Sheeran	I Don't Care (with Justin Bieber) [Loud Luxury...	66
8928	Tot	BTNG	Black Mamba	17
22179	Where This Flower Blooms (feat. Frank Ocean)	Tyler, The Creator	Flower Boy	69
915	Where Would I Be - Remix	Heart Youth	Where Would I Be (Remix)	33
24630	Old School	Aaliyah	Age Ain't Nothing But A Number	26

In [67]:
```
genre = 'pop'
n_recommendations = 5
filtered = df[df['playlist_genre'] == genre]
recommendations = filtered.sort_values('track_popularity', ascending=False).head(n_
recommendations[['track_name', 'track_artist', 'track_album_name', 'track_popularit
```

Out[67]:

	track_name	track_artist	track_album_name	track_popularity
1551	Dance Monkey	Tones and I	Dance Monkey (Stripped Back) / Dance Monkey	100
1605	Blinding Lights	The Weeknd	Blinding Lights	98
1301	Circles	Post Malone	Hollywood's Bleeding	98
716	Blinding Lights	The Weeknd	Blinding Lights	98
711	Memories	Maroon 5	Memories	98