In [4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
features=pd.read_csv(r"C:\Users\SURYA K\OneDrive\Desktop\jupyter lab\SpotifyFeature
tracks=pd.read_csv(r"C:\Users\SURYA K\OneDrive\Desktop\jupyter lab\archive\tracks.c
```

In [6]:
```
tracks.head()
```

Out[6]:

	id	name	popularity	duration_ms	explicit	artists	
0	35iwgR4jXetI318WEWsa1Q	Carve	6	126903	0	['Uli']	
1	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['Fernando Pessoa']	['
2	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	0	['Ignacio Corsini']	['
3	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	0	['Ignacio Corsini']	['
4	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	0	['Dick Haymes']	

In [7]:
```
features.head()
```

Out[7]:

	genre	artist_name	track_name	track_id	popularity	acousticness	da
0	Movie	Henri Salvador	C'est beau de faire un Show	0BRjO6ga9RKCKjfDqeFgWV	0	0.611	
1	Movie	Martin & les fées	Perdu d'avance (par Gad Elmaleh)	0BjC1NfoEOOusryehmNudP	1	0.246	
2	Movie	Joseph Williams	Don't Let Me Be Lonely Tonight	0CoSDzoNIKCRs124s9uTVy	3	0.952	
3	Movie	Henri Salvador	Dis-moi Monsieur Gordon Cooper	0Gc6TVm52BwZD07Ki6tIvf	0	0.703	
4	Movie	Fabien Nataf	Ouverture	0IuslXpMROHdEPvSI1fTQK	4	0.950	

In [60]:

```
tracks.shape
```

Out[60]:

```
(586672, 21)
```

In [61]:

```
features.shape
```

Out[61]:

```
(232725, 19)
```

In [8]:

```
#checking null
pd.isnull(tracks).sum()
```

Out[8]:

```
id                   0
name                71
popularity           0
duration_ms          0
explicit             0
artists              0
id_artists           0
release_date         0
danceability         0
energy               0
key                  0
loudness             0
mode                 0
speechiness          0
acousticness         0
instrumentalness     0
liveness             0
valence              0
tempo                0
time_signature       0
dtype: int64
```

In [9]:

```
pd.isnull(features).sum()
```

Out[9]:

```
genre               0
artist_name         0
track_name          1
track_id            0
popularity          0
acousticness        0
danceability        0
duration_ms         0
energy              0
instrumentalness    0
key                 0
liveness            0
loudness            0
mode                0
speechiness         0
tempo               0
time_signature      0
valence             0
dtype: int64
```

In [10]:

```
#checking info
tracks.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 586672 entries, 0 to 586671
Data columns (total 20 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   id                586672 non-null  object 
 1   name              586601 non-null  object 
 2   popularity        586672 non-null  int64  
 3   duration_ms       586672 non-null  int64  
 4   explicit          586672 non-null  int64  
 5   artists           586672 non-null  object 
 6   id_artists        586672 non-null  object 
 7   release_date      586672 non-null  object 
 8   danceability      586672 non-null  float64
 9   energy            586672 non-null  float64
 10  key               586672 non-null  int64  
 11  loudness          586672 non-null  float64
 12  mode              586672 non-null  int64  
 13  speechiness       586672 non-null  float64
 14  acousticness      586672 non-null  float64
 15  instrumentalness  586672 non-null  float64
 16  liveness          586672 non-null  float64
 17  valence           586672 non-null  float64
 18  tempo             586672 non-null  float64
 19  time_signature    586672 non-null  int64  
dtypes: float64(9), int64(6), object(5)
memory usage: 89.5+ MB
```

In [11]:

```
#checking info
features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 232725 entries, 0 to 232724
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   genre             232725 non-null  object 
 1   artist_name       232725 non-null  object 
 2   track_name        232724 non-null  object 
 3   track_id          232725 non-null  object 
 4   popularity        232725 non-null  int64  
 5   acousticness      232725 non-null  float64
 6   danceability      232725 non-null  float64
 7   duration_ms       232725 non-null  int64  
 8   energy            232725 non-null  float64
 9   instrumentalness  232725 non-null  float64
 10  key               232725 non-null  object 
 11  liveness          232725 non-null  float64
 12  loudness          232725 non-null  float64
 13  mode              232725 non-null  object 
 14  speechiness       232725 non-null  float64
 15  tempo             232725 non-null  float64
 16  time_signature    232725 non-null  object 
 17  valence           232725 non-null  float64
dtypes: float64(9), int64(2), object(7)
memory usage: 32.0+ MB
```

In [12]:

```
#finding 10 least popular songs in the spotify dataset
least_songs=tracks.sort_values('popularity',ascending=True)[0:10]
least_songs[['name','popularity']]
```

Out[12]:

	name	popularity
32	The Dear Little Shamrock	0
78	Pobre Cotorro - Remasterizado	0
77	Entrerriana - Remasterizado	0
76	Capítulo 2.9 - Banquero Anarquista	0
75	Capítulo 1.9 - Banquero Anarquista	0
74	Capítulo 1.7 - Banquero Anarquista	0
73	The Girl That I Marry	0
72	Capítulo 2.14 - Banquero Anarquista	0
71	Capítulo 2.4 - Banquero Anarquista	0
70	Capítulo 1.21 - Banquero Anarquista	0

In [13]:

```
#descriptive statistics of tracks
tracks.describe().transpose()
```

Out[13]:

	count	mean	std	min	25%	50
popularity	586672.0	27.570053	18.370642	0.0	13.0000	27.0000
duration_ms	586672.0	230051.167286	126526.087418	3344.0	175093.0000	214893.0000
explicit	586672.0	0.044086	0.205286	0.0	0.0000	0.0000
danceability	586672.0	0.563594	0.166103	0.0	0.4530	0.5770
energy	586672.0	0.542036	0.251923	0.0	0.3430	0.5490
key	586672.0	5.221603	3.519423	0.0	2.0000	5.0000
loudness	586672.0	-10.206067	5.089328	-60.0	-12.8910	-9.2430
mode	586672.0	0.658797	0.474114	0.0	0.0000	1.0000
speechiness	586672.0	0.104864	0.179893	0.0	0.0340	0.0443
acousticness	586672.0	0.449863	0.348837	0.0	0.0969	0.4220
instrumentalness	586672.0	0.113451	0.266868	0.0	0.0000	0.0000
liveness	586672.0	0.213935	0.184326	0.0	0.0983	0.1390
valence	586672.0	0.552292	0.257671	0.0	0.3460	0.5640
tempo	586672.0	118.464857	29.764108	0.0	95.6000	117.3840
time_signature	586672.0	3.873382	0.473162	0.0	4.0000	4.0000

In [14]:

```
#descriptive of feature
features.describe().transpose()
```

Out[14]:

	count	mean	std	min	25%	
popularity	232725.0	41.127502	18.189948	0.00000	29.0000	43
acousticness	232725.0	0.368560	0.354768	0.00000	0.0376	0
danceability	232725.0	0.554364	0.185608	0.05690	0.4350	0
duration_ms	232725.0	235122.339306	118935.909299	15387.00000	182857.0000	220427
energy	232725.0	0.570958	0.263456	0.00002	0.3850	0
instrumentalness	232725.0	0.148301	0.302768	0.00000	0.0000	0
liveness	232725.0	0.215009	0.198273	0.00967	0.0974	0
loudness	232725.0	-9.569885	5.998204	-52.45700	-11.7710	-7
speechiness	232725.0	0.120765	0.185518	0.02220	0.0367	0
tempo	232725.0	117.666585	30.898907	30.37900	92.9590	115
valence	232725.0	0.454917	0.260065	0.00000	0.2370	0

In [15]:
```
#finding top 10 popular songs in the spotify dataset
least_songs=tracks
popular_songs=least_songs[least_songs['popularity']>90].sort_values('popularity',as
popular_songs[['name','popularity','artists']]
```

Out[15]:

	name	popularity	artists
93802	Peaches (feat. Daniel Caesar & Giveon)	100	['Justin Bieber', 'Daniel Caesar', 'Giveon']
93803	drivers license	99	['Olivia Rodrigo']
93804	Astronaut In The Ocean	98	['Masked Wolf']
92811	telepatía	97	['Kali Uchis']
92810	Save Your Tears	97	['The Weeknd']
92813	Blinding Lights	96	['The Weeknd']
93805	Leave The Door Open	96	['Bruno Mars', 'Anderson .Paak', 'Silk Sonic']
92814	The Business	95	['Tiësto']
91866	Streets	94	['Doja Cat']
93806	Fiel	94	['Los Legendarios', 'Wisin', 'Jhay Cortez']

In [16]:
```
#Make the Release Date Column as the Index Column.
tracks['release_date'] = pd.to_datetime(tracks['release_date'], dayfirst=True, erro
tracks.set_index('release_date',inplace=True)
tracks.index=pd.to_datetime(tracks.index)
tracks.head()
```

Out[16]:

	id	name	popularity	duration_ms	explicit	
release_date						
1922-02-22	35iwgR4jXetI318WEWsa1Q	Carve	6	126903	0	
1922-06-01	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	98200	0	['F
1922-03-21	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	0	
1922-03-21	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	0	
NaT	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	0	

In [17]:
```
#Find the Name of the Artist Present in the specific Row of the Dataset.
tracks[['artists']].iloc[24]
```

Out[17]:
```
artists    ['Fernando Pessoa']
Name: 1922-06-01 00:00:00, dtype: object
```

In [18]:
```
print("Unique Genre")
print(features['genre'].unique())
print("Unique Artists")
print(tracks['artists'].unique())
```

```
Unique Genre
['Movie' 'R&B' 'A Capella' 'Alternative' 'Country' 'Dance' 'Electronic'
 'Anime' 'Folk' 'Blues' 'Opera' 'Hip-Hop' "Children's Music"
 'Children’s Music' 'Rap' 'Indie' 'Classical' 'Pop' 'Reggae' 'Reggaeton'
 'Jazz' 'Rock' 'Ska' 'Comedy' 'Soul' 'Soundtrack' 'World']
Unique Artists
["['Uli']" "['Fernando Pessoa']" "['Ignacio Corsini']" ... "['阿YueYue']"
 "['ROLE MODEL']" "['Gentle Bones', 'Clara Benin']"]
```

In [19]:
```
#Converting the Duration of the Songs From Milliseconds to Seconds.
tracks['duration'] = tracks['duration_ms'].apply (lambda x : round(x/1000))
tracks.drop('duration_ms', inplace = True, axis=1)
tracks.duration.head()
```

Out[19]:
```
release_date
1922-02-22    127
1922-06-01     98
1922-03-21    182
1922-03-21    177
NaT           163
Name: duration, dtype: int64
```

In [20]:
```
#Most Common Artists in the Dataset
top_artists = tracks['artists'].value_counts().head(10)
print(top_artists)
```

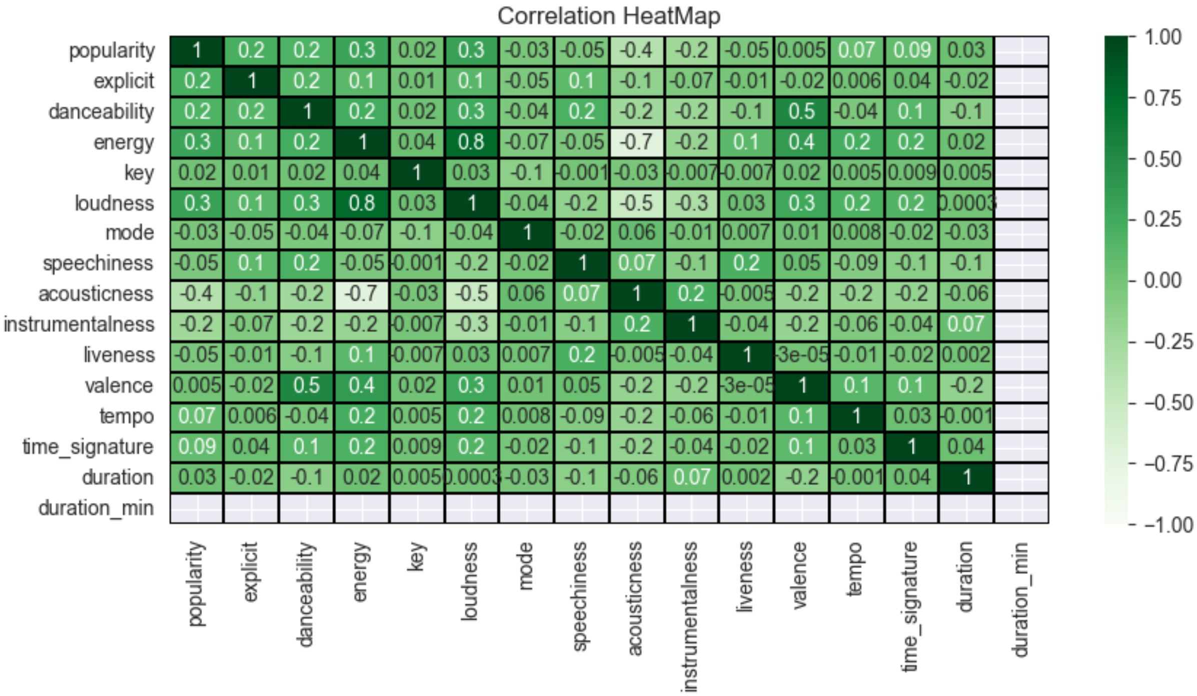
```
artists
['Die drei ???']                              3856
['TKKG Retro-Archiv']                         2006
['Benjamin Blümchen']                         1503
['Bibi Blocksberg']                           1472
['Lata Mangeshkar']                           1373
['Bibi und Tina']                              927
['Tintin', 'Tomas Bolme', 'Bert-Åke Varg']     905
['Francisco Canaro']                           891
['Ella Fitzgerald']                            870
['Tadeusz Dolega Mostowicz']                   838
Name: count, dtype: int64
```

In [58]:
```
# Keep only numeric columns for correlation
numeric_tracks = tracks.select_dtypes(include=['number'])

# Calculate correlation
correlation = numeric_tracks.corr(method='pearson')

# Plot heatmap
```

```
plt.figure(figsize=(9,5))
hmap = sns.heatmap(correlation, annot=True, fmt='.1g', vmin=-1, vmax=1, center=0, c
hmap.set_title('Correlation HeatMap')
hmap.set_xticklabels(hmap.get_xticklabels(), rotation=90)
plt.tight_layout()
plt.show()
```

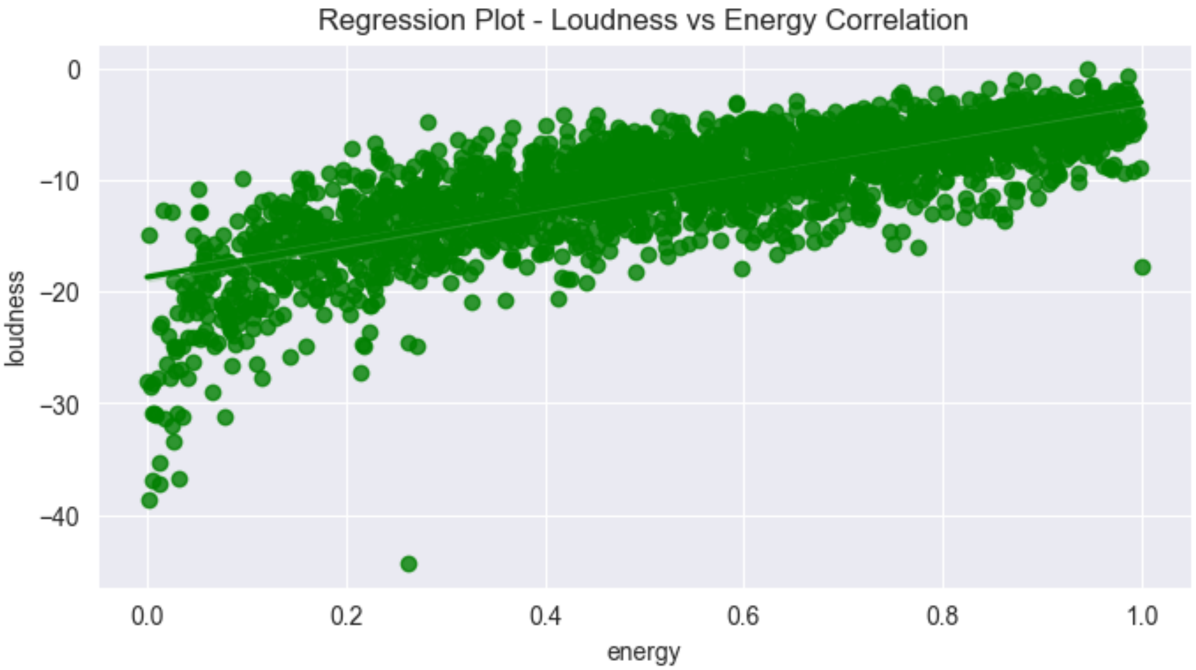


```
In [22]: #Sample Only 4 Percent of the Whole Dataset.
         sample=tracks.sample(int(0.004*len(tracks)))
         print(len(sample))

         2346
```

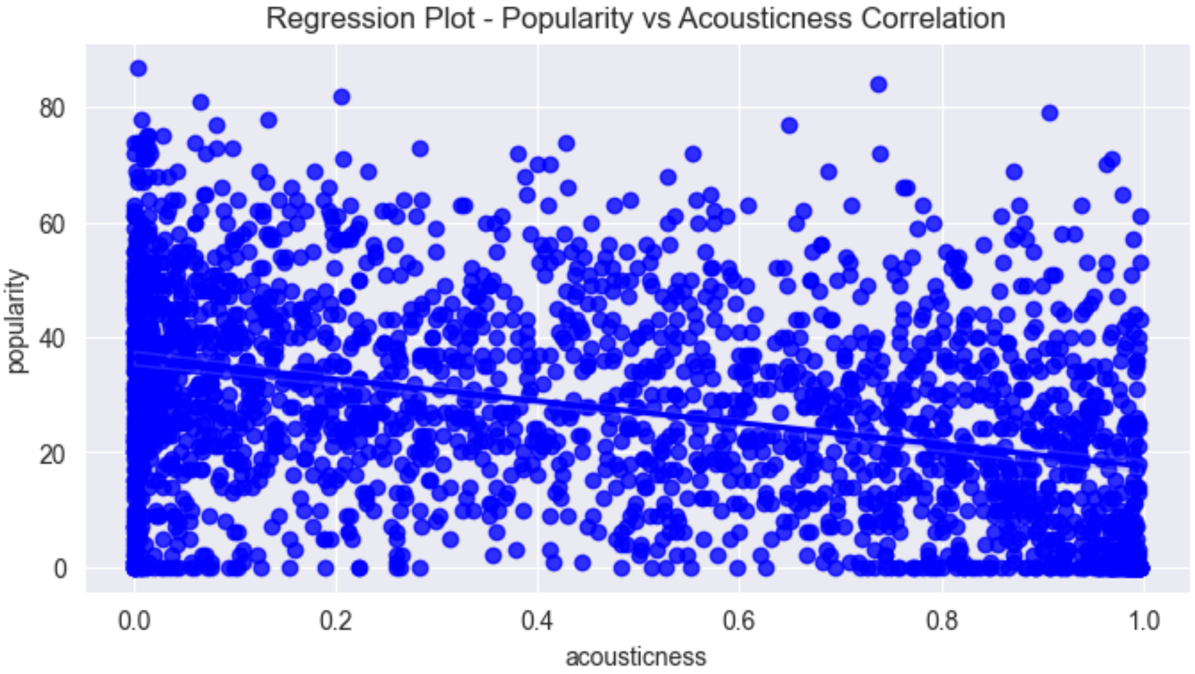
```
In [50]: #Create a Regression Plot Between Loudness and Energy. Let’s Plot It  in the Form o
         plt.figure(figsize=(8,4))
         sns.regplot(data=sample, y='loudness', x='energy', color='green').set(title='Regres

Out[50]: [Text(0.5, 1.0, 'Regression Plot - Loudness vs Energy Correlation')]
```



```
In [49]: #Create a Regression Plot Between Popularity and Acousticness in the Form of a Regr
         plt.figure(figsize=(8,4))
         sns.regplot(data=sample, y='popularity', x='acousticness', color='blue').set(title=
```

```
Out[49]: [Text(0.5, 1.0, 'Regression Plot - Popularity vs Acousticness Correlation')]
```



```
In [25]: #creating new column in tracks table
         tracks['dates']=tracks.index.get_level_values('release_date')
         tracks.dates=pd.to_datetime(tracks.dates)
         years=tracks.dates.dt.year
         tracks.head()
```

Out[25]:

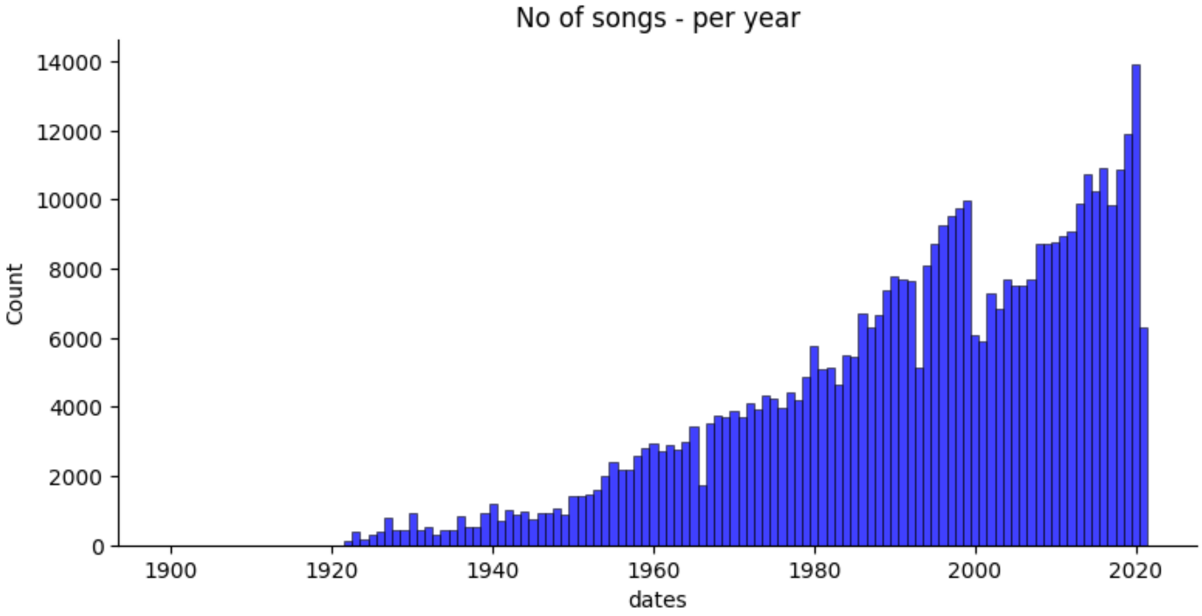
release_date	id	name	popularity	explicit	artists	
1922-02-22	35iwgR4jXetI318WEWsa1Q	Carve	6	0	['Uli']	
1922-06-01	021ht4sdgPcrDgSk7JTbKY	Capítulo 2.16 - Banquero Anarquista	0	0	['Fernando Pessoa']	['14jt
1922-03-21	07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	0	['Ignacio Corsini']	['5LiC
1922-03-21	08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	0	['Ignacio Corsini']	['5LiC
NaT	08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	0	['Dick Haymes']	['3

In [28]:

```
#Number of Songs per Year
sns.displot(years, discrete=True, aspect=2, height=4, kind='hist',color='blue').set
```

Out[28]:

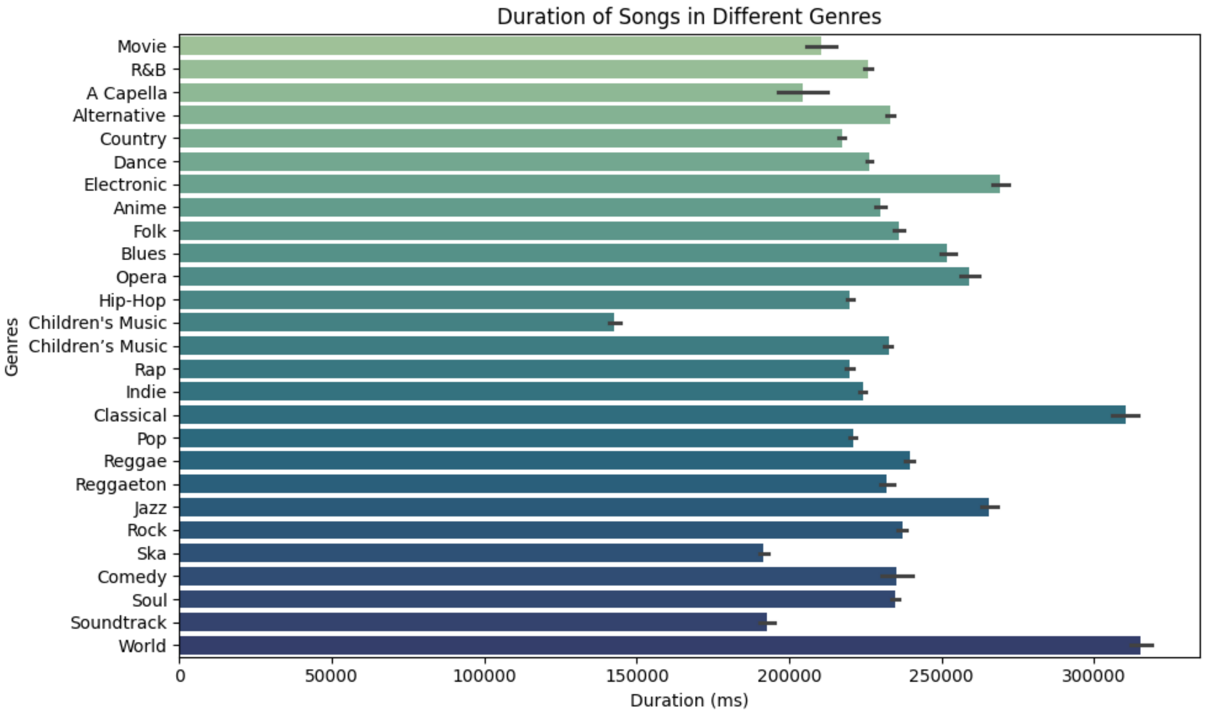
```
<seaborn.axisgrid.FacetGrid at 0x17291e3dd10>
```



In [ ]:

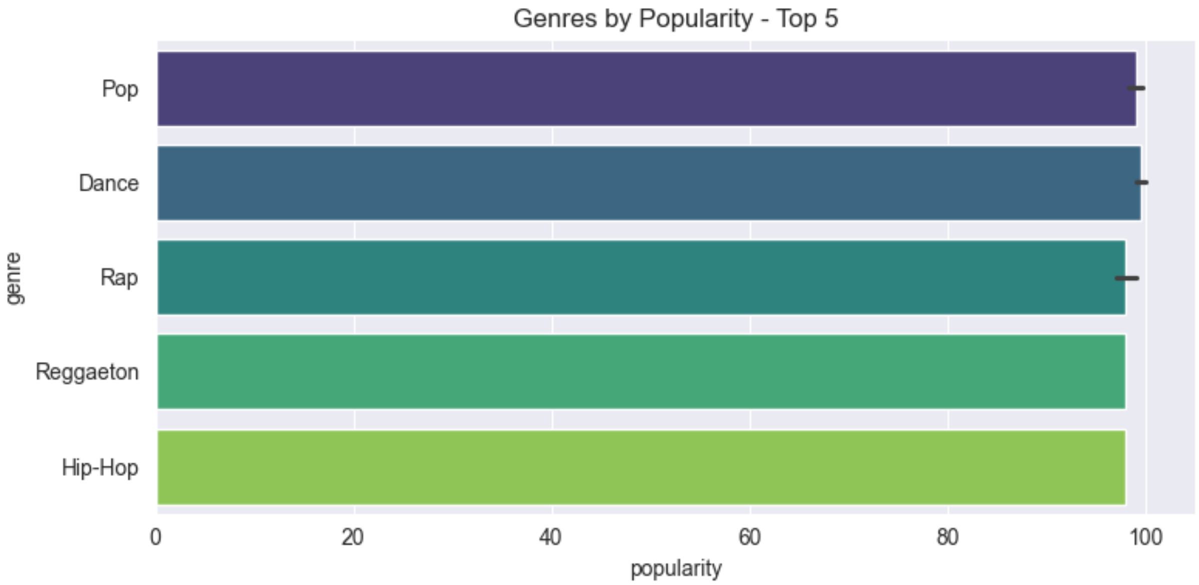
In [32]:

```
#spotify feature analysis
#Horizontal Bar Plot: Song Duration Across Different Genres
plt.figure(figsize=(10,6))
plt.title('Duration of Songs in Different Genres')
sns.barplot(y='genre', x='duration_ms', data=features, hue='genre', palette='crest'
plt.xlabel('Duration (ms)')
plt.ylabel('Genres')
plt.tight_layout()
plt.show()
```



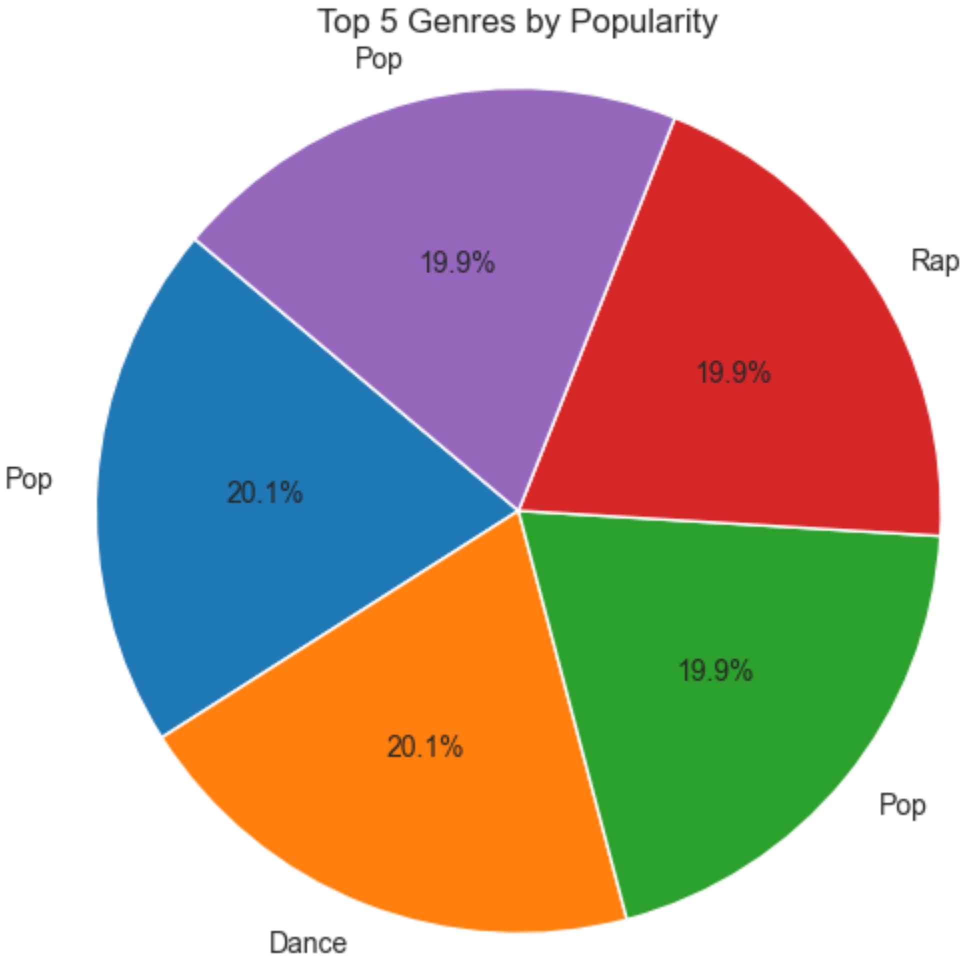
In [48]:

```
#Find top five genres by Popularity and polting a barplot for the same.
top_genres = features.sort_values('popularity', ascending=False).head(10)
sns.set_style('darkgrid')
plt.figure(figsize=(8,4))
sns.barplot(y='genre',x='popularity',data=top_genres,hue='genre',palette='viridis',
plt.tight_layout()
plt.show()
```



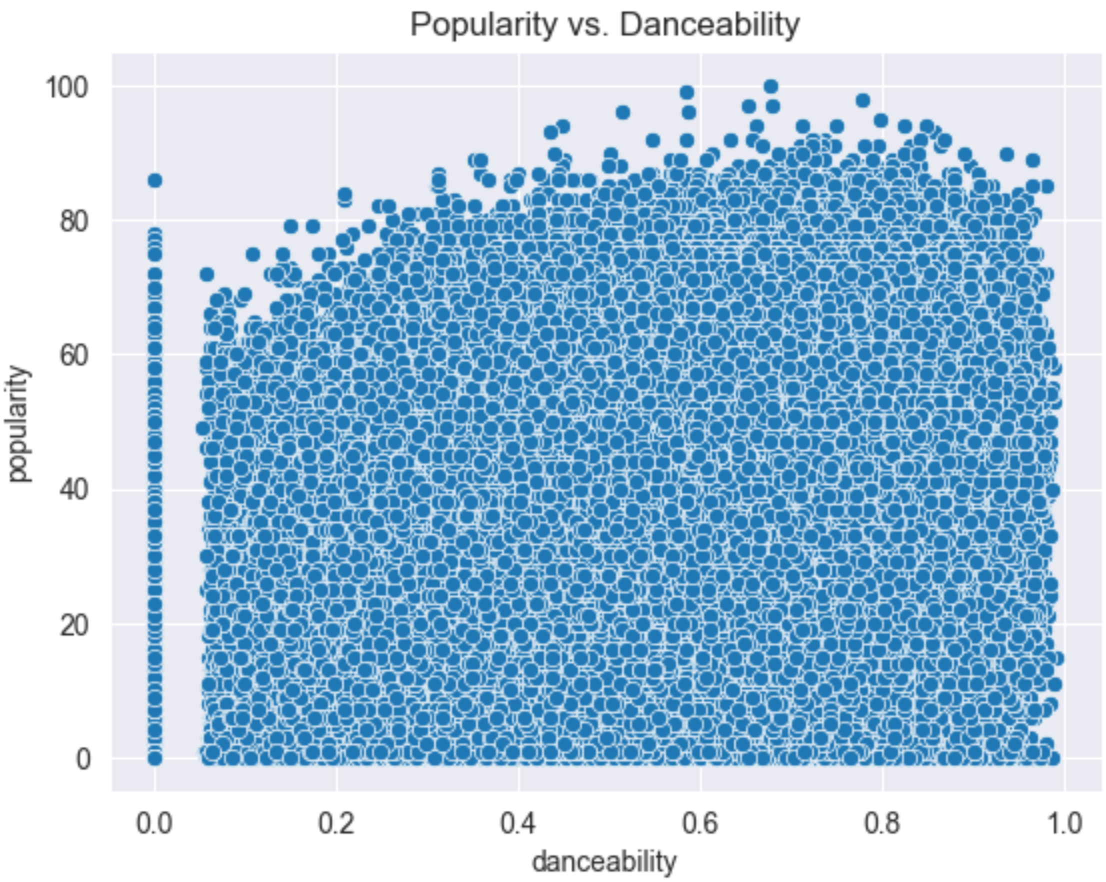
In [59]:

```
# Get top 5 genres by popularity
Top = features.sort_values('popularity', ascending=False).head(5)
# Create pie chart
plt.figure(figsize=(6,6))
plt.pie(Top['popularity'], labels=Top['genre'], autopct='%1.1f%%', startangle=140)
plt.title('Top 5 Genres by Popularity')
plt.axis('equal')  # Equal aspect ratio ensures the pie is a circle
plt.show()
```



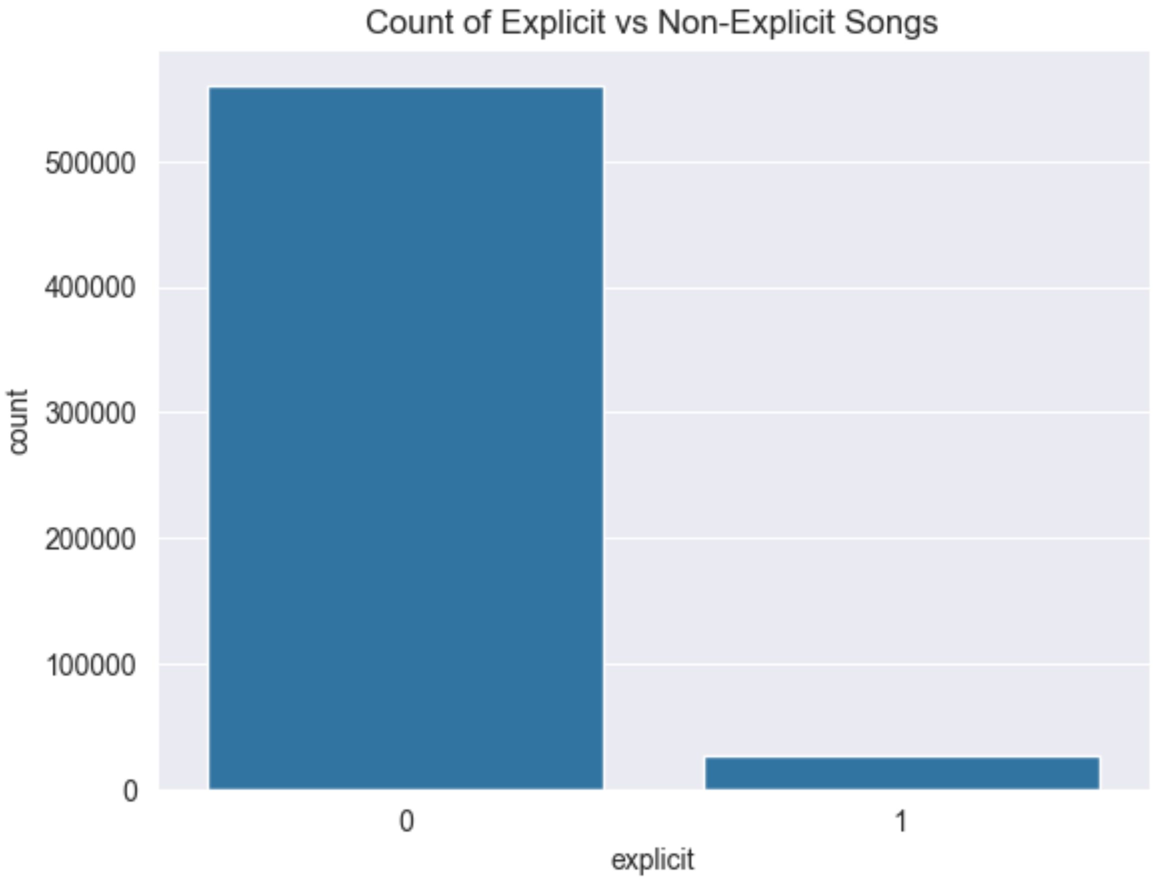
In [143…

```
sns.scatterplot(x='danceability', y='popularity', data=tracks)
plt.title('Popularity vs. Danceability')
plt.show()
```



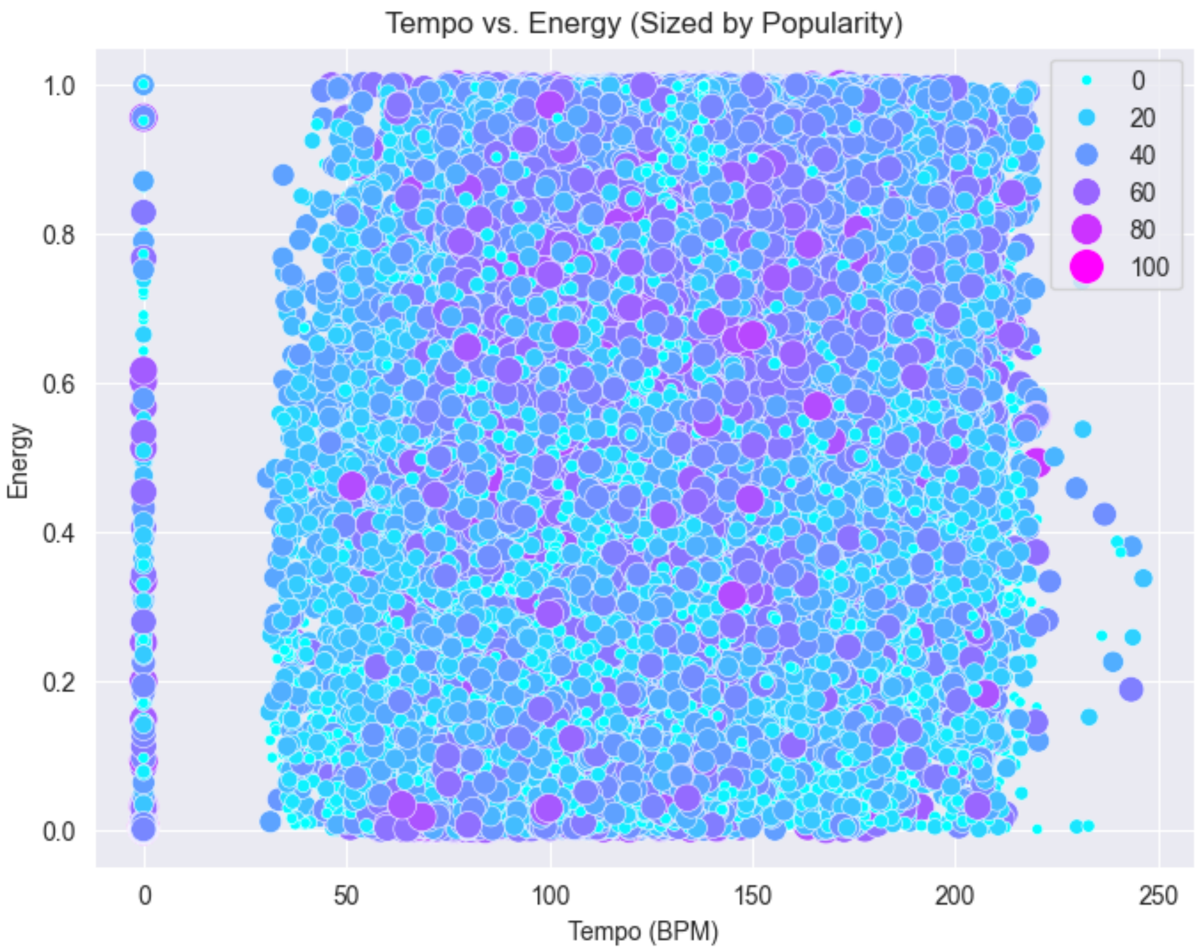
In [144…

```
sns.countplot(x='explicit', data=tracks)
plt.title('Count of Explicit vs Non-Explicit Songs')
plt.show()
```



In [151…

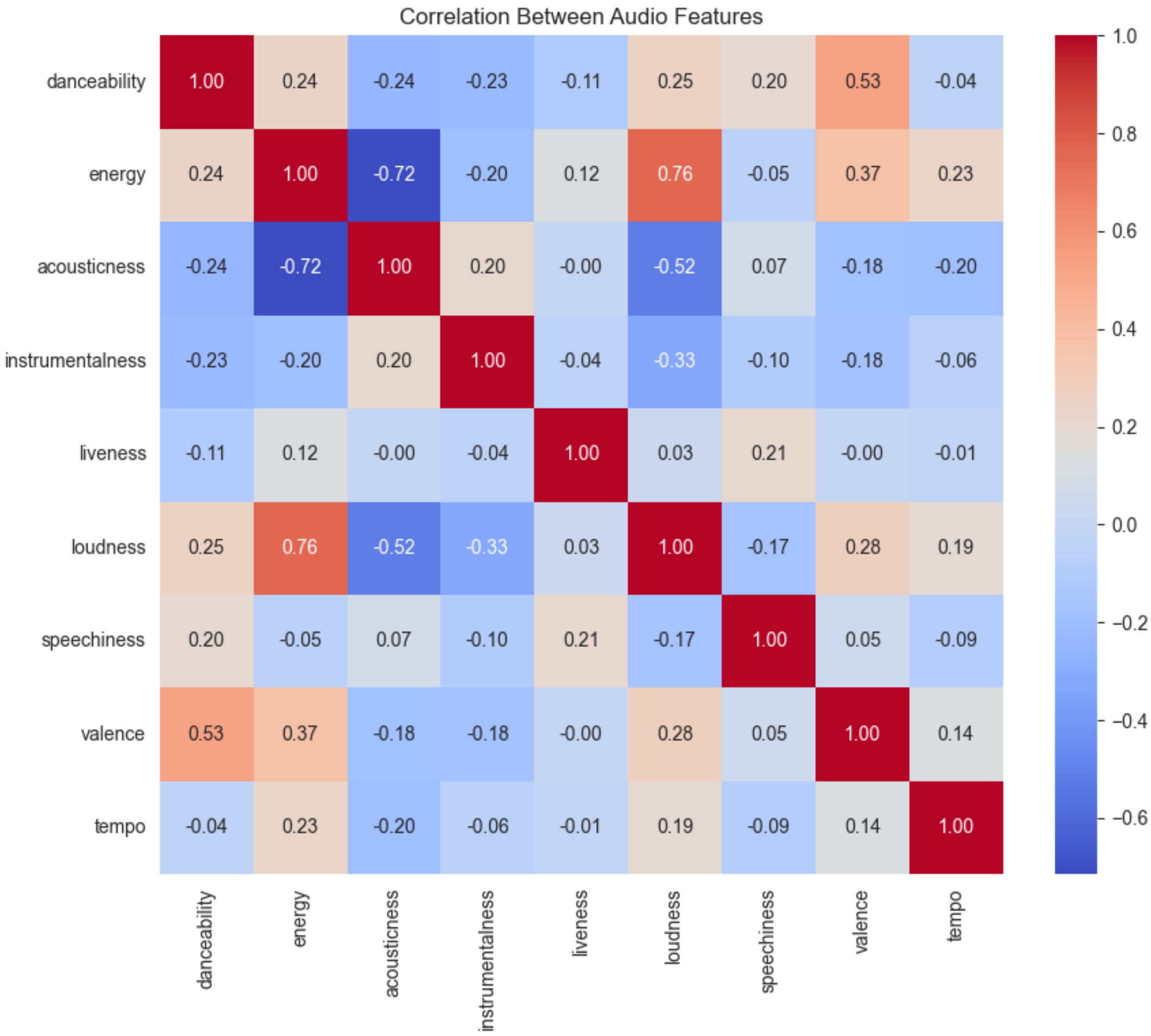
```
#Tempo vs Energy Scatter Plot
plt.figure(figsize=(8,6))
sns.scatterplot(x='tempo', y='energy', data=tracks, hue='popularity', palette='cool
plt.title('Tempo vs. Energy (Sized by Popularity)')
plt.xlabel('Tempo (BPM)')
plt.ylabel('Energy')
plt.legend()
plt.show()
```



In [152…

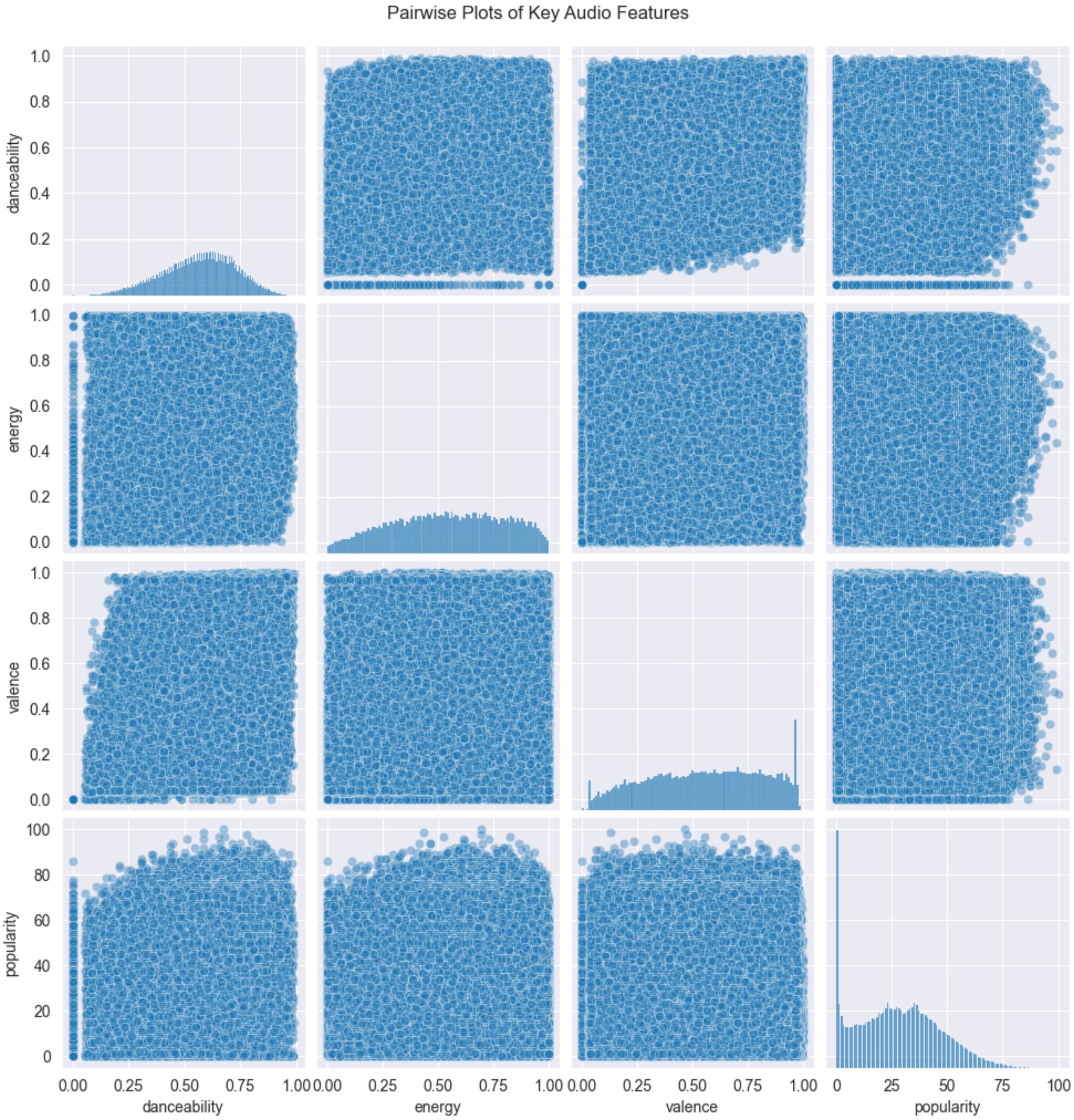
```
audio_features = ['danceability', 'energy', 'acousticness', 'instrumentalness',
                  'liveness', 'loudness', 'speechiness', 'valence', 'tempo']

plt.figure(figsize=(10,8))
sns.heatmap(tracks[audio_features].corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Between Audio Features')
plt.show()
```



In [153…

```
sns.pairplot(tracks[['danceability', 'energy', 'valence', 'popularity']], kind='sca
plt.suptitle('Pairwise Plots of Key Audio Features', y=1.02)
plt.show()
```



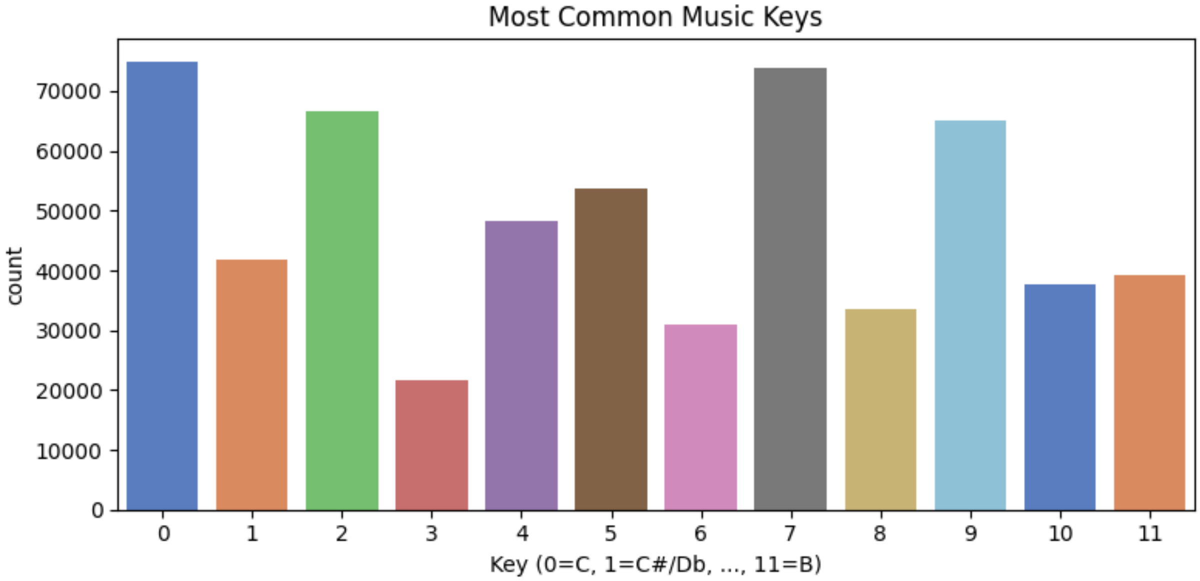
In [166…

```
#Number of Songs Released Per Year
tracks['year'] = pd.to_datetime(tracks['release_date']).dt.year
print(tracks['year'])
```

```
release_date
1922-02-22    1970.0
1922-06-01    1970.0
1922-03-21    1970.0
1922-03-21    1970.0
NaT              NaN
               ...
2020-09-26    1970.0
2020-10-21    1970.0
2020-09-02    1970.0
2021-03-05    1970.0
2015-07-01    1970.0
Name: year, Length: 586672, dtype: float64
```

In [39]:

```
plt.figure(figsize=(8,4))
sns.countplot(x='key', data=tracks, hue='key', palette='muted', legend=False)
plt.title('Most Common Music Keys')
plt.xlabel('Key (0=C, 1=C#/Db, ..., 11=B)')
plt.tight_layout()
plt.show()
```

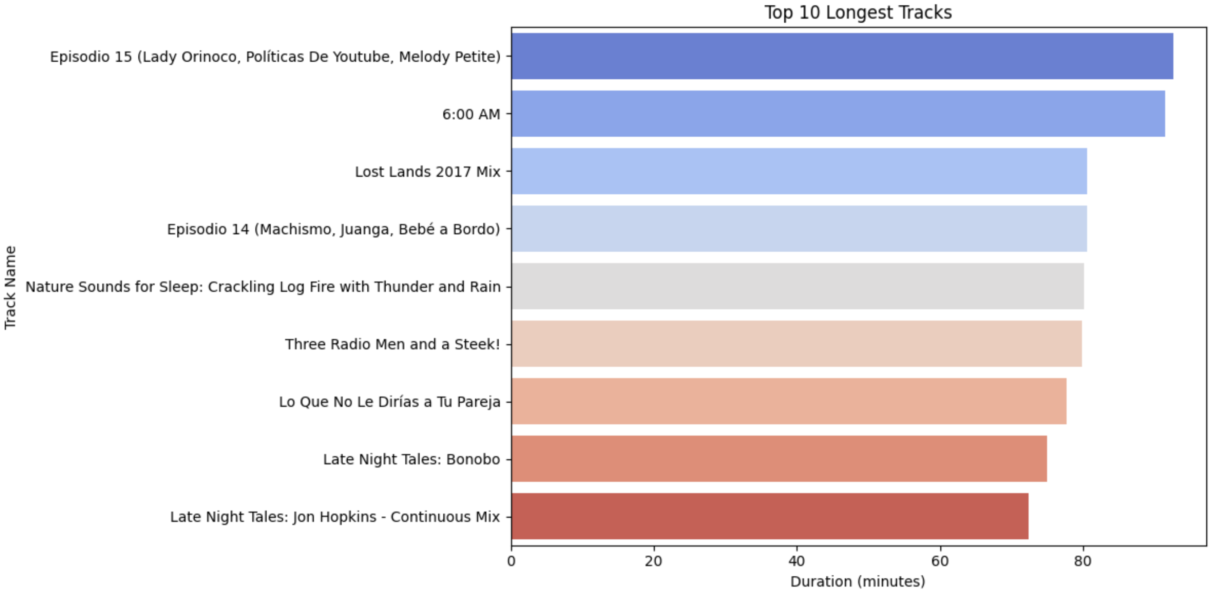


In [45]:

```
# Convert duration from milliseconds to minutes
features['duration_min'] = features['duration_ms'] / 60000

# Get top 10 longest tracks
longest_tracks = features.sort_values(by='duration_min', ascending=False).head(10)

# Plot with hue to avoid warning
plt.figure(figsize=(12,6))
sns.barplot(
    x='duration_min',
    y='track_name',
    data=longest_tracks,
    hue='track_name',       # add hue
    palette='coolwarm',
    dodge=False,
    legend=False            # hide redundant legend
)
plt.title('Top 10 Longest Tracks')
plt.xlabel('Duration (minutes)')
plt.ylabel('Track Name')
plt.tight_layout()
plt.show()
```



In [41]:

```
print(tracks.columns.tolist())
```

```
['id', 'name', 'popularity', 'explicit', 'artists', 'id_artists', 'danceability', 'e
nergy', 'key', 'loudness', 'mode', 'speechiness', 'acousticness', 'instrumentalness'
, 'liveness', 'valence', 'tempo', 'time_signature', 'duration', 'dates', 'duration_m
in']
```

In [42]:

```
print(features.columns.tolist())
```

```
['genre', 'artist_name', 'track_name', 'track_id', 'popularity', 'acousticness', 'da
nceability', 'duration_ms', 'energy', 'instrumentalness', 'key', 'liveness', 'loudne
ss', 'mode', 'speechiness', 'tempo', 'time_signature', 'valence']
```

In [ ]: