In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
C:\Users\Yash\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required
for this version of SciPy (detected version 1.26.4
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

1.Load the dataframe and ensure data quality by checking for missing values and duplicate rows. Handle missing values and remove duplicate rows if necessary.

In [2]:
```
df = pd.read_csv('spotify.csv')
```

In [3]:
```
df.head()
```

Out[3]:

	Artist	Track Name	Popularity	Duration (ms)	Track ID
0	Drake	Rich Baby Daddy (feat. Sexyy Red & SZA)	92	319191	1yeB8MUNeLo9Ek1UEpsyz6
1	Drake	One Dance	91	173986	1zi7xx7UVEFkmKfv06H8x0
2	Drake	IDGAF (feat. Yeat)	90	260111	2YSzYUF3jWqb9YP9VXmpjE
3	Drake	First Person Shooter (feat. J. Cole)	88	247444	7aqfrAY2p9BUSiupwk3svU
4	Drake	Jimmy Cooks (feat. 21 Savage)	88	218364	3F5CgOj3wFIRv51JsHbxhe

In [4]:
```
df.shape
```

Out[4]:
```
(440, 5)
```

In [5]:
```
df.dtypes
```

Out[5]:
```
Artist          object
Track Name      object
Popularity       int64
Duration (ms)    int64
Track ID        object
dtype: object
```

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

Out[6]:

	Popularity	Duration (ms)
count	440.000000	440.000000
mean	75.736364	206810.040909
std	9.886534	53576.930289
min	29.000000	81666.000000
25%	70.000000	172778.500000
50%	77.000000	201866.000000
75%	83.000000	235119.750000
max	97.000000	501648.000000

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440 entries, 0 to 439
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Artist         440 non-null    object
 1   Track Name     440 non-null    object
 2   Popularity     440 non-null    int64
 3   Duration (ms)  440 non-null    int64
 4   Track ID       440 non-null    object
dtypes: int64(2), object(3)
memory usage: 17.3+ KB
```

In [8]:
```
1 df.isnull().sum()
```

Out[8]:
```
Artist          0
Track Name      0
Popularity      0
Duration (ms)   0
Track ID        0
dtype: int64
```

In [9]:
```
1 df.duplicated().value_counts()
```

Out[9]:
```
False    413
True      27
dtype: int64
```

In [10]:
```
1 df.drop_duplicates(inplace=True)
```

In [11]:
```
1 print("\nDataframe after handling missing values and removing duplicates:")
2 df.head()
```

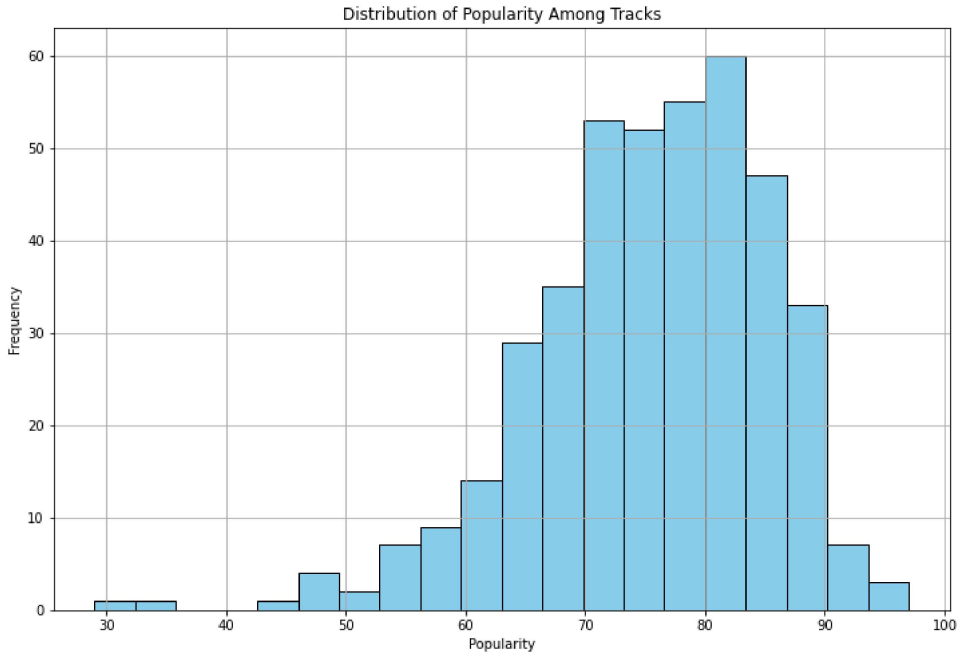
```
Dataframe after handling missing values and removing duplicates:
```

Out[11]:

	Artist	Track Name	Popularity	Duration (ms)	Track ID
0	Drake	Rich Baby Daddy (feat. Sexyy Red & SZA)	92	319191	1yeB8MUNeLo9Ek1UEpsyz6
1	Drake	One Dance	91	173986	1zi7xx7UVEFkmKfv06H8x0
2	Drake	IDGAF (feat. Yeat)	90	260111	2YSzYUF3jWqb9YP9VXmpjE
3	Drake	First Person Shooter (feat. J. Cole)	88	247444	7aqfrAY2p9BUSiupwk3svU
4	Drake	Jimmy Cooks (feat. 21 Savage)	88	218364	3F5CgOj3wFIRv51JsHbxhe

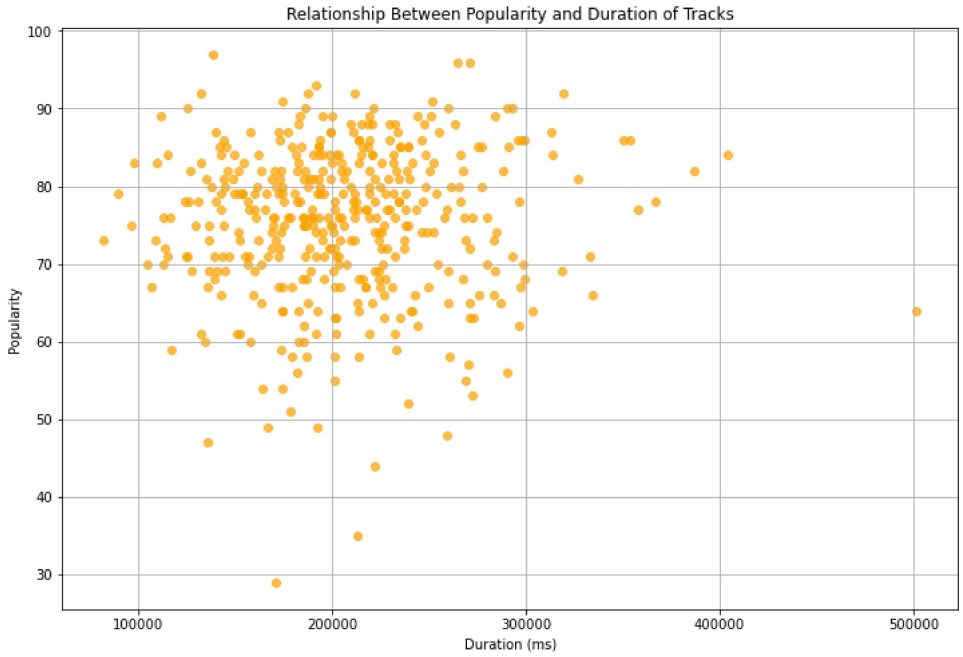
2.What is the distribution of popularity among the tracks in the dataset? Visualize it using a histogram.

In [12]:
```
1 plt.figure(figsize=(12,8))
2 plt.hist(df['Popularity'], bins=20, color='skyblue', edgecolor='black')
3 plt.title('Distribution of Popularity Among Tracks')
4 plt.xlabel('Popularity')
5 plt.ylabel('Frequency')
6 plt.grid(True)
7 plt.show()
```



3.Is there any relationship between the popularity and the duration of tracks? Explore this using a scatter plot.

In [13]:
```
plt.figure(figsize=(12,8))
plt.scatter(df['Duration (ms)'], df['Popularity'], color='orange', alpha=0.7)
plt.title('Relationship Between Popularity and Duration of Tracks')
plt.xlabel('Duration (ms)')
plt.ylabel('Popularity')
plt.grid(True)
plt.show()
```



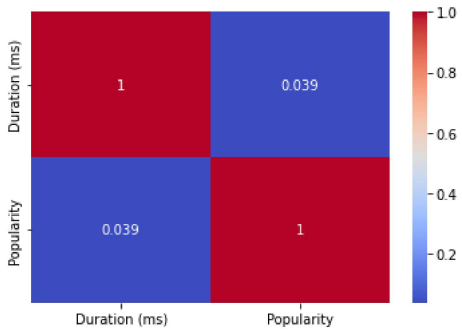
In [14]:
```
correlation = df[['Duration (ms)', 'Popularity']].corr()
correlation
```

Out[14]:

	Duration (ms)	Popularity
Duration (ms)	1.000000	0.038992
Popularity	0.038992	1.000000

In [15]:
```
sns.heatmap(correlation, annot=True, cmap='coolwarm')
```

Out[15]: <AxesSubplot:>



4.Which artist has the highest number of tracks in the dataset? Display the count of tracks for each artist using a countplot.

In [16]:
```
df
```

Out[16]:

	Artist	Track Name	Popularity	Duration (ms)	Track ID
0	Drake	Rich Baby Daddy (feat. Sexyy Red & SZA)	92	319191	1yeB8MUNeLo9Ek1UEpsyz6
1	Drake	One Dance	91	173986	1zi7xx7UVEFkmKfv06H8x0
2	Drake	IDGAF (feat. Yeat)	90	260111	2YSzYUF3jWqb9YP9VXmpjE
3	Drake	First Person Shooter (feat. J. Cole)	88	247444	7aqfrAY2p9BUSiupwk3svU
4	Drake	Jimmy Cooks (feat. 21 Savage)	88	218364	3F5CgOj3wFIRv51JsHbxhe
...	...	...	...	...	...
433	French Montana	Stand United	54	163971	01CHrTerCzyRpMI1MzQ4fz
434	Jason Derulo	Tip Toe (feat. French Montana)	65	187521	0TY3jVGwGDwDabLyQLVRQQ
436	Fat Joe	All The Way Up (feat. Infared)	64	191900	7EzwtgfW7khBrpvaNPtMoT
437	A$AP Ferg	Work REMIX (feat. A$AP Rocky, French Montana, ...	69	283693	7xVLFuuYdAvcTfcP3IG3dS
438	Diddy	Another One Of Me (feat. 21 Savage)	65	220408	4hGmQboiou09EwhcTWa0H6

413 rows × 5 columns

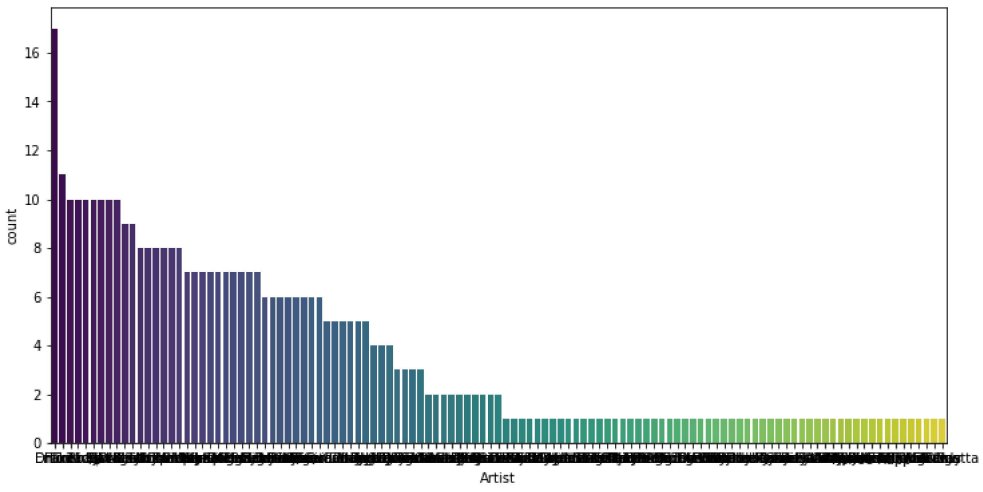
In [17]:
```
artist_track_count = df['Artist'].value_counts()
artist_track_count
```

Out[17]:
```
Drake             17
Eminem            11
Flo Rida          10
Ludacris          10
Timbaland         10
                  ..
Arizona Zervas     1
Fivio Foreign      1
Pressa             1
David Guetta       1
Diddy              1
Name: Artist, Length: 115, dtype: int64
```

In [18]:
```
most_tracks_artist = artist_track_count.idxmax()
highest_track_count = artist_track_count.max()

print(f"The artist with the highest number of tracks is '{most_tracks_artist}' with {highest_track_count} tracks.")

plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='Artist', order=df['Artist'].value_counts().index, palette='viridis')
```

```
The artist with the highest number of tracks is 'Drake' with 17 tracks.
```

Out[18]:
```
<AxesSubplot:xlabel='Artist', ylabel='count'>
```



5.What are the top 5 least popular tracks in the dataset? Provide the artist name and track name for each.

In [19]:
```
df_sorted_by_popularity = df.sort_values(by='Popularity',ascending=True)
```

In [20]:
```
top_5_least_popular_tracks = df_sorted_by_popularity.head(5)
top_5_least_popular_tracks
```

Out[20]:

	Artist	Track Name	Popularity	Duration (ms)	Track ID
207	Pressa	Attachments (feat. Coi Leray)	29	171000	6EfJ8Ct3GHbBz2YlyPnMYb
231	Justin Bieber	Intentions	35	212853	7jAVt70Xdg8EwOtsFB1ZqK
413	French Montana	Splash Brothers	44	221863	3fBsEOnzwtlkpS0LxXAZhN
225	Lil Baby	On Me - Remix	47	135444	2uUFVnVFERNxUdcmvEs7LB
407	Wyclef Jean	911 (feat. Mary J. Blige)	48	259333	28hgx2bWXcaBJeC9zVwrBq

In [21]:
```
for index, row in top_5_least_popular_tracks.iterrows():
    print(f"Artist: {row['Artist']}, Track: {row['Track Name']}, Popularity: {row['Popularity']}")
```

```
Artist: Pressa, Track: Attachments (feat. Coi Leray), Popularity: 29
Artist: Justin Bieber, Track: Intentions, Popularity: 35
Artist: French Montana, Track: Splash Brothers, Popularity: 44
Artist: Lil Baby, Track: On Me - Remix, Popularity: 47
Artist: Wyclef Jean, Track: 911 (feat. Mary J. Blige), Popularity: 48
```

6.Among the top 5 most popular artists, which artist has the highest popularity on average? Calculate and display the average popularity for each artist.

In [22]:
```
avg_popularity_per_artist = df.groupby('Artist')['Popularity'].mean()
avg_popularity_per_artist
```

Out[22]:
```
Artist
*NSYNC                  67.00
2 Chainz                72.00
21 Savage               83.80
A Boogie Wit da Hoodie  80.00
A$AP Ferg               69.00
                        ...
Young Nudy              67.00
Young Thug              73.75
benny blanco            72.00
cassö                   92.00
¥$                      85.10
Name: Popularity, Length: 115, dtype: float64
```

In [23]:
```
sorted_artists_by_avg_popularity = avg_popularity_per_artist.sort_values(ascending=False)
sorted_artists_by_avg_popularity
```

Out[23]:
```
Artist
cassö             92.000000
Trueno            89.000000
David Guetta      87.000000
Travis Scott      86.555556
¥$                85.100000
                    ...
RAYE              55.000000
Wyclef Jean       54.500000
Arizona Zervas    54.000000
Justin Bieber     49.000000
Pressa            29.000000
Name: Popularity, Length: 115, dtype: float64
```

In [24]:
```
top_5_most_popular_artists = sorted_artists_by_avg_popularity.head(5)
top_5_most_popular_artists
```

Out[24]:
```
Artist
cassö           92.000000
Trueno          89.000000
David Guetta    87.000000
Travis Scott    86.555556
¥$              85.100000
Name: Popularity, dtype: float64
```

In [25]:
```
artist_with_highest_avg_popularity = top_5_most_popular_artists.idxmax()
highest_avg_popularity = top_5_most_popular_artists.max()

print(f"\nThe artist with the highest average popularity among the top 5 most popular artists is '{artist_with_highest_avg_p
```

```
The artist with the highest average popularity among the top 5 most popular artists is 'cassö' with an average popularity of 9
2.0.
```

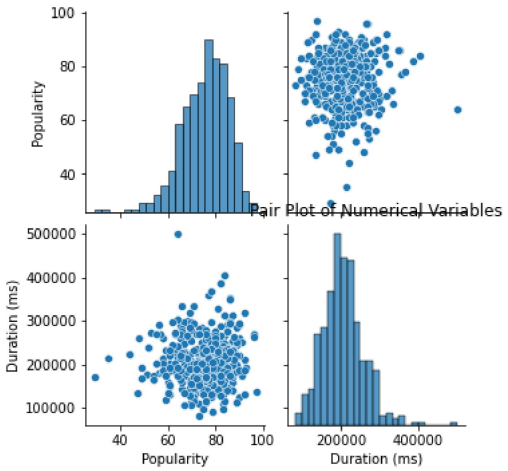
7.For the top 5 most popular artists, what are their most popular tracks? List the track name for each artist.

In [26]:
```
# Get the top 5 most popular artists
top_5_most_popular_artists = sorted_artists_by_avg_popularity.head(5)

# Initialize a dictionary to store the most popular track for each artist
most_popular_tracks_per_artist = {}

# Loop through the top 5 most popular artists
for artist in top_5_most_popular_artists.index:
    # Filter the dataframe to include only tracks by the current artist
    artist_tracks = df[df['Artist'] == artist]
    # Find the most popular track for the current artist
    most_popular_track = artist_tracks.loc[artist_tracks['Popularity'].idxmax(), 'Track Name']
    # Store the most popular track for the current artist in the dictionary
    most_popular_tracks_per_artist[artist] = most_popular_track

# Display the most popular track for each of the top 5 most popular artists
print("Most popular tracks for each of the top 5 most popular artists:")
for artist, track in most_popular_tracks_per_artist.items():
    print(f"Artist: {artist}, Most Popular Track: {track}")

```

```
Most popular tracks for each of the top 5 most popular artists:
Artist: cassö, Most Popular Track: Prada
Artist: Trueno, Most Popular Track: Mamichula - con Nicki Nicole
Artist: David Guetta, Most Popular Track: Baby Don't Hurt Me
Artist: Travis Scott, Most Popular Track: FE!N (feat. Playboi Carti)
Artist: ¥$, Most Popular Track: CARNIVAL
```

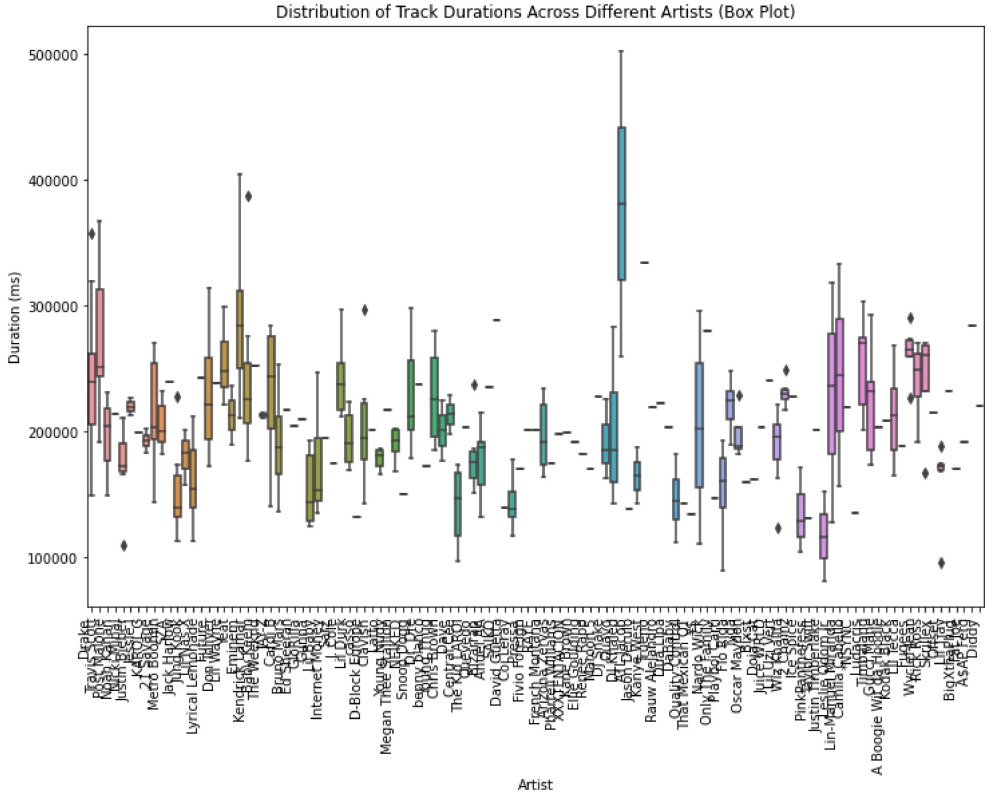
8.Visualize relationships between multiple numerical variables simultaneously using a pair plot.

In [29]:
```
numerical_columns = ['Popularity', 'Duration (ms)']

sns.pairplot(df[numerical_columns])
plt.title('Pair Plot of Numerical Variables')
plt.show()
```

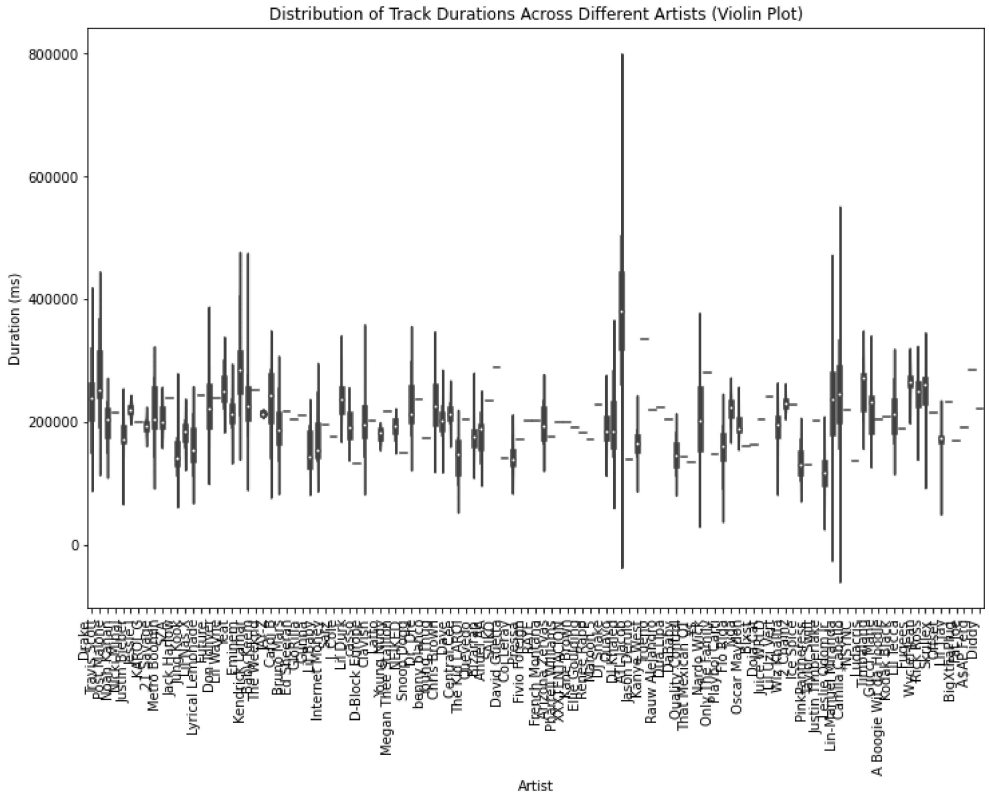


9.Does the duration of tracks vary significantly across different artists? Explore this visually using a box plot or violin plot.

In [32]:
```
# using boxplot

plt.figure(figsize=(12, 8))
sns.boxplot(data=df, x='Artist', y='Duration (ms)')
plt.title('Distribution of Track Durations Across Different Artists (Box Plot)')
plt.xticks(rotation=90, ha='right')
plt.xlabel('Artist')
plt.ylabel('Duration (ms)')
plt.show()
```



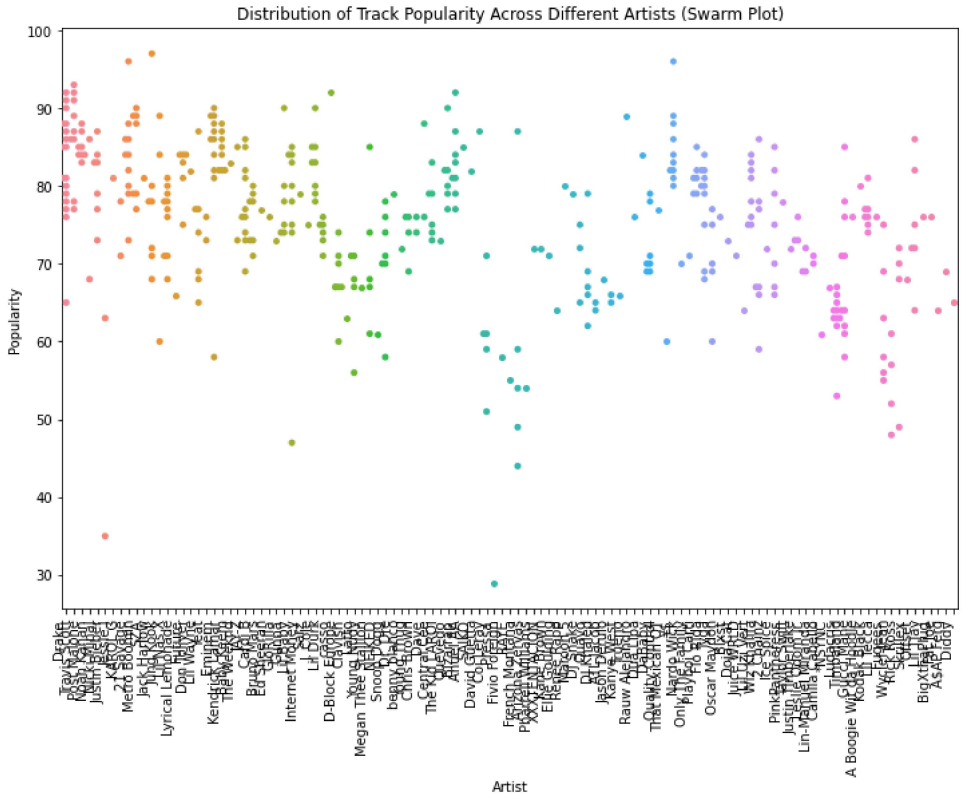
In [33]:
```
# using vioin plot
plt.figure(figsize=(12, 8))
sns.violinplot(data=df, x='Artist', y='Duration (ms)')
plt.title('Distribution of Track Durations Across Different Artists (Violin Plot)')
plt.xticks(rotation=90, ha='right')
plt.xlabel('Artist')
plt.ylabel('Duration (ms)')
plt.show()
```



10.How does the distribution of track popularity vary for different artists? Visualize this using a swarm plot or a violin plot.

In [35]:
```
plt.figure(figsize=(12, 8))
sns.swarmplot(data=df, x='Artist', y='Popularity')
plt.title('Distribution of Track Popularity Across Different Artists (Swarm Plot)')
plt.xticks(rotation=90, ha='right')
plt.xlabel('Artist')
plt.ylabel('Popularity')
plt.show()
```

```
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 17.6% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 11.1% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 37.5% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 14.3% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 20.0% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 28.6% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 33.3% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 18.2% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 25.0% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 10.0% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 16.7% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 12.5% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
C:\Users\Yash\anaconda3\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 30.0% of the points cannot be placed; you m
ay want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```



In [ ]:
```
1
```