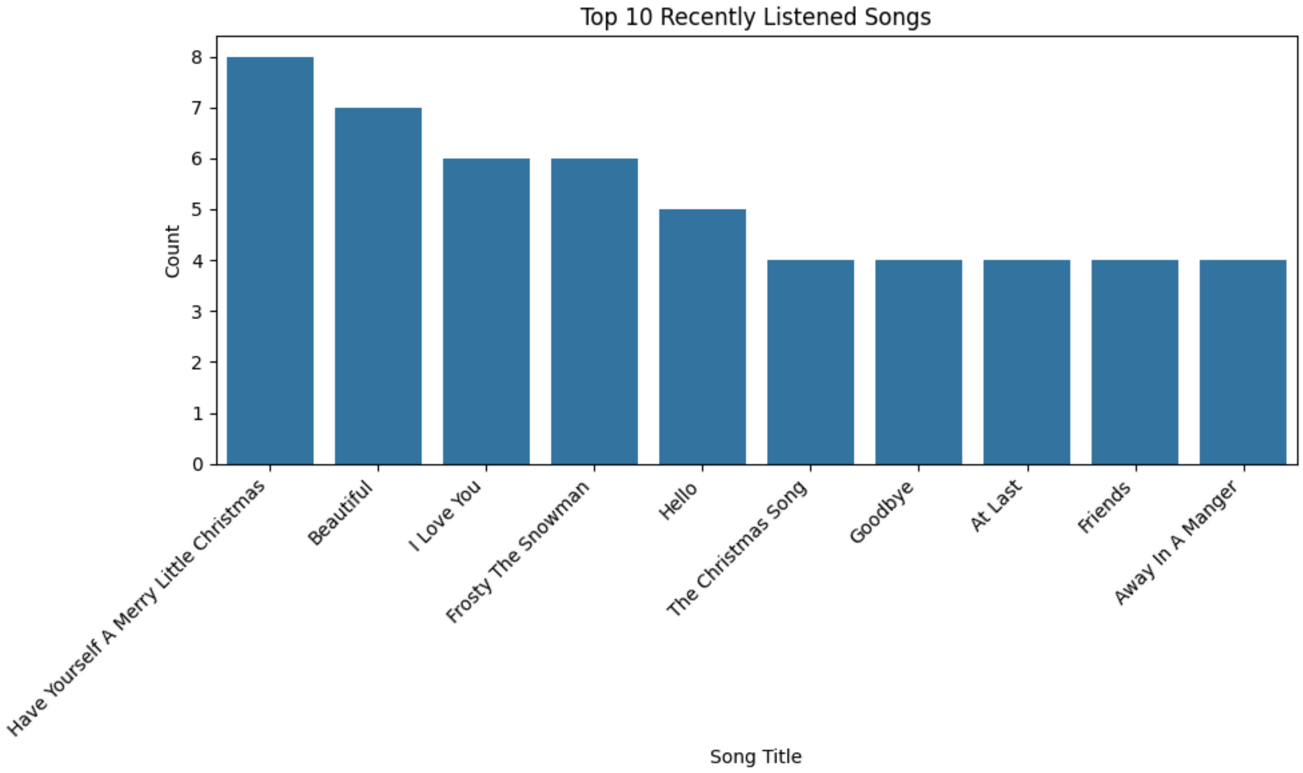
```
import pandas as pd
```

```
import pandas as pd

# Read the CSV file into a DataFrame
df = pd.read_csv("music.csv")
```

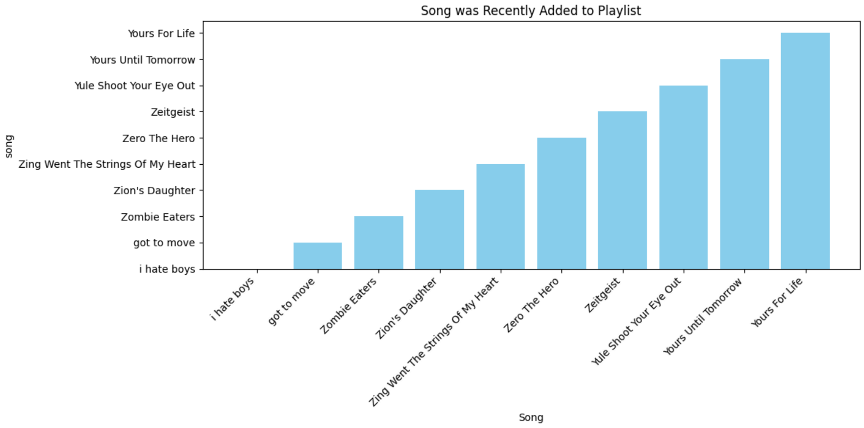
```
df = pd.read_csv("music.csv")
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load your dataset containing recently listened music
# Replace 'your_dataset.csv' with the actual file path or data source
recently_listened_data = pd.read_csv('music.csv')

# Preprocess the dataset if necessary

# Assuming your dataset has a column named 'song' indicating the song title

# Count the occurrences of each song
recently_listened_counts = recently_listened_data['song'].value_counts()

# Plotting the bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x=recently_listened_counts.index[:10], y=recently_listened_counts.values[:10])
plt.title('Top 10 Recently Listened Songs')
plt.xlabel('Song Title')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

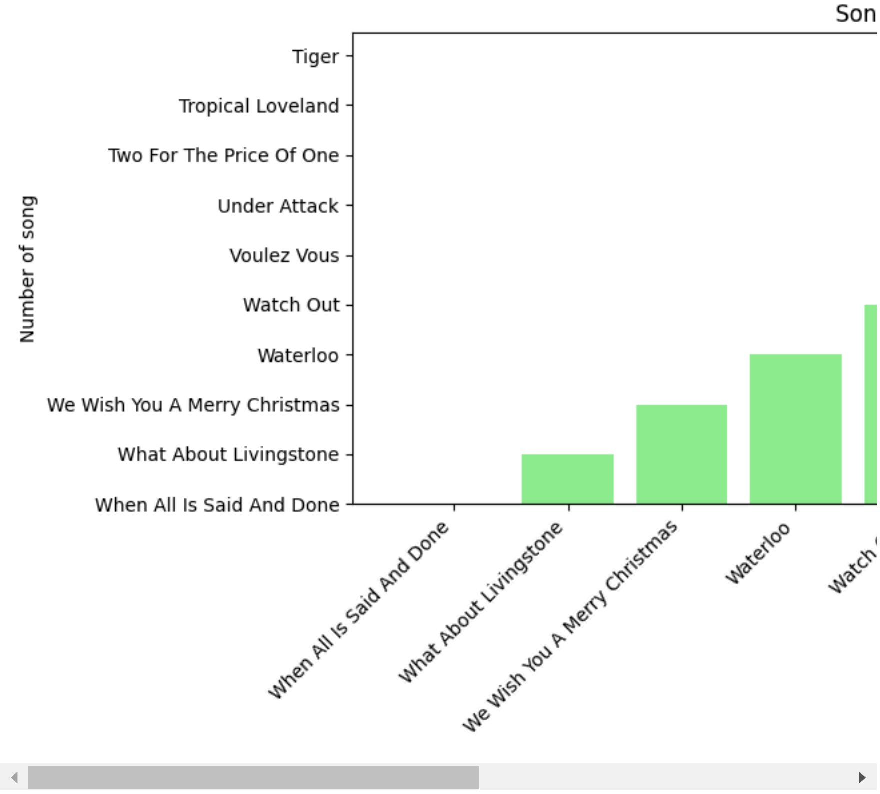


```
import pandas as pd
import matplotlib.pyplot as plt

# Load your dataset containing information about recently added songs to a playlist
# Replace 'your_dataset.csv' with the actual file path or data source
recently_added_data = pd.read_csv('music.csv')

# Assuming your dataset has columns 'song' and 'times_added'

# Sort the data by the number of times added in descending order
recently_added_data_sorted = recently_added_data.sort_values(by='song', ascending=False)
```

```
# Selecting top N songs for visualization (optional)
top_n = 10
top_n_songs = recently_added_data_sorted.head(top_n)

# Plotting the bar chart
plt.figure(figsize=(12, 6))
plt.bar(top_n_songs['song'], top_n_songs['song'], color='skyblue')
plt.title('Song was Recently Added to Playlist')
plt.xlabel('Song')
plt.ylabel('song')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



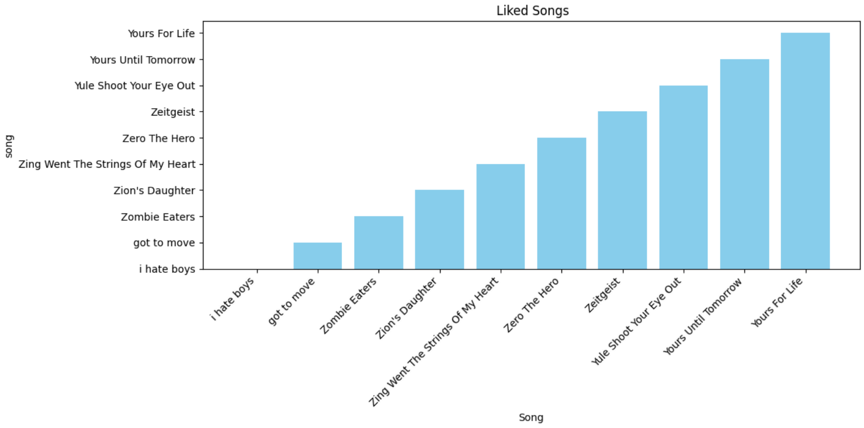
```
import pandas as pd
import matplotlib.pyplot as plt

# Load your dataset containing information about recently added songs to a playlist
# Replace 'your_dataset.csv' with the actual file path or data source
recently_added_data = pd.read_csv('music.csv')

# Assuming your dataset has columns 'song' and 'times_added'

# Selecting the range of rows from 100 to 1500
recently_added_data_selected = recently_added_data.iloc[100:110]

# Sort the selected data by the number of times added in descending order
recently_added_data_sorted = recently_added_data_selected.sort_values(by='song', ascending=False)

# Plotting the bar chart
plt.figure(figsize=(12, 6))
plt.bar(recently_added_data_sorted['song'], recently_added_data_sorted['song'], color='lightgreen')
plt.title(' Song was Recently liked')
plt.xlabel('Song')
plt.ylabel('Number of song')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



```
import pandas as pd

# Load your dataset containing information about songs
# Replace 'your_dataset.csv' with the actual file path or data source
songs_data = pd.read_csv('music.csv')

# Assuming your dataset has a column 'song' containing song titles

random_songs = songs_data['song'].sample(n=25, random_state=42)

# Randomly select one song from the dataset
random_song = songs_data['song'].sample(n=1).iloc[0]

print("Randomly selected song:", random_song)
```

```
Randomly selected song: Ring Them Bells
```

```
import pandas as pd
import matplotlib.pyplot as plt

# Load your dataset containing information about recently liked music
# Replace 'your_dataset.csv' with the actual file path or data source
recently_liked_data = pd.read_csv('music.csv')

# Assuming your dataset has columns 'song' and 'likes'

# Sort the data by the number of likes in descending order
# Sort the data by the number of likes in descending order
recently_liked_data_sorted = recently_liked_data.sort_values(by='song', ascending=False)


# Selecting top N songs for visualization (optional)
top_n = 10
top_n_songs = recently_liked_data_sorted.head(top_n)

# Plotting the bar chart
plt.figure(figsize=(12, 6))
plt.bar(top_n_songs['song'], top_n_songs['song'], color='skyblue')
plt.title(' Liked Songs')
plt.xlabel('Song')
plt.ylabel('song')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```



```
df.head(5)
```

	artist	song	link	text
0	ABBA	Ahe's My Kind Of Girl	/a/abba/ahes+my+kind+of+girl_20598417.html	Look at her face, it's a wonderful face \r\nA...
1	ABBA	Andante, Andante	/a/abba/andante+andante_20002708.html	Take it easy with me, please \r\nTouch me gen...
2	ABBA	As Good As New	/a/abba/as+good+as+new_20003033.html	I'll never know why I had to go \r\nWhy I had

```
df.tail(5)
```

	artist	song	link	text
6606	Hanson	My Own Sweet Time	/h/hanson/my+own+sweet+time_20579405.html	Hello, goodbye my friend \r\nFeels like the s...
6607	Hanson	Need You Now	/h/hanson/need+you+now_20579404.html	Your deep brown eyes \r\nThey watch me as I s...
6608	Hanson	Never	/h/hanson/never+let+go_20287064.html	Just lay down \r\nAnd let your

```
df.shape
```

```
(6611, 4)
```

```
df.isnull().sum()
```

```
artist    0
song      0
link      1
text      1
dtype: int64
```

```
df =df.sample(5000).drop('link', axis=1).reset_index(drop=True)
```

```
df.head(10)
```

	artist	song	text
0	Ariel Rivera	Minamahal Pala Kita	sana'y kaya kong gawin na malimutan ka \r\nsa...
1	Christina Aguilera	Moves Like Jagger	Just shoot for the stars if it feels right \r...
2	Ed Sheeran	Touch And Go	[Intro] \r\nOh oh oh \r\nOh oh oh \r\nOh oh...
3	Dolly Parton	Heartbreak Express	(Dolly Parton) \r\nPackin' my suitcase writin...
4	Annie	We Got Annie	Grace: We got Annie. \r\n \r\nGardener: We'v...
5	Hank Williams Jr.	Just Enough To Get In Trouble	I've been down my share of \r\nCountry roads,...
6	Drake	Trust Issues	All I care about is money and the city that I'...
7	Elton John	Chloe	How come you're so understanding \r\nWhen I t...

```
df['text'][0]
```

```
'sana'y kaya kong gawin na malimutan ka  \r\nsana'y maitago ko ang luha ng mga mata
\r\nsana'y kayang tiisin na magmula ngayo'y di na magkita pa  \r\nngayong wala ka na
\r\n  \r\nii stanza  \r\nang akala ko noon ay di kita mahal  \r\nat ang pagtingin sa
yo'y isang laro lamang  \r\nlabis na sinaktan ka at sa ngayo'y nagpapaalam na...  \r
\ndi mapigilan pa  \r\n  \r\nchorus:  \r\nminamahal pala kita  \r\nngayon ko lamang
nadama  \r\nhindi mo na kaya mapapatawad pa  \r\npatawad na  \r\nat kung mahal mo na
```

```
df.shape
```

```
(5000, 3)
```

```
df['text'] = df['text'].str.lower().replace(r'^\w\s', ' ').replace(r'\n', ' ', regex = True)
```

```
import nltk
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()

def tokenization(txt):
    tokens = nltk.word_tokenize(txt)
    stemming = [stemmer.stem(w) for w in tokens]
    return " ".join(stemming)
```

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
```

```
tfidvector = TfidfVectorizer(analyzer='word',stop_words='english')
matrix = tfidvector.fit_transform(df['text'])
similarity = cosine_similarity(matrix)
```

```
similarity[0]
```

```
array([1.        , 0.        , 0.00852145, ..., 0.        , 0.        ,
       0.        ])
```

```
df[df['song'] == 'Crying Over You']
```

	artist	song	text
4480	ABBA	Crying Over You	i'm waitin' for you baby \r i'm sitting all a...

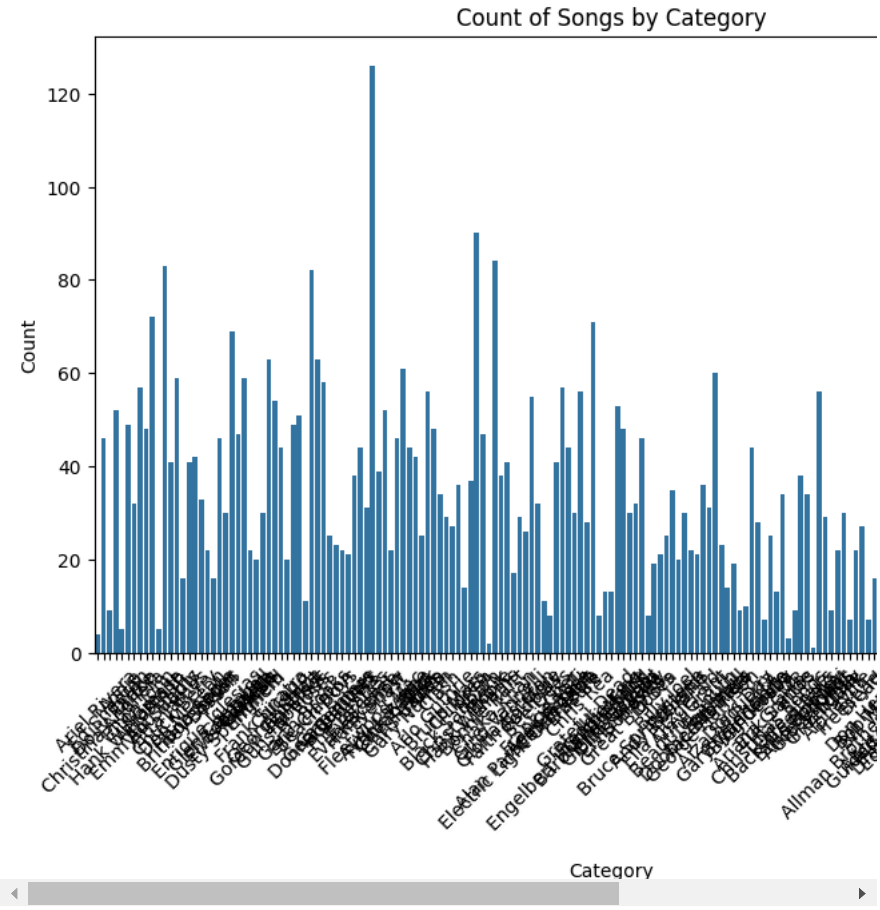
```
def recommendation(song_df):
      idx = df[df['song'] == song_df].index[0]
      distances = sorted(list(enumerate(similarity[idx])),reverse=True,key=lambda x:x[1])

      songs = []
      for m_id in distances[1:21]:
          songs.append(df.iloc[m_id[0]].song)

      return songs
```

```
recommendation('Crying Over You')
```

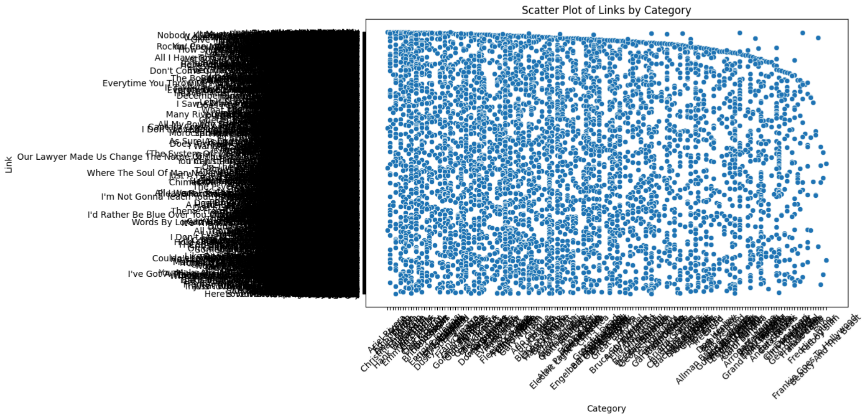
```
["Cryin'",
 'I Want You To Want Me',
 'Am I Blue',
 "For Cryin' Out Loud",
 'I See You Lord',
 'My Sweet Lord',
 'Lovesick Blues',
 'Baby Blue',
 'Midnight Blue',
 "Fixin' To Die",
 'Gospel Plow',
 'Almost Blue',
 'Stand Alone',
 "I'd Still Want You",
 'Am I Blue?',
 'Better',
 'All About You',
 'Oh Me, Oh My Sweet Baby',
 "Just Waitin'",
 'Dump The Dude']
```

```
import pickle
pickle.dump(similarity,open('similarity.pkl','wb'))
pickle.dump(df,open('df.pkl','wb'))
```

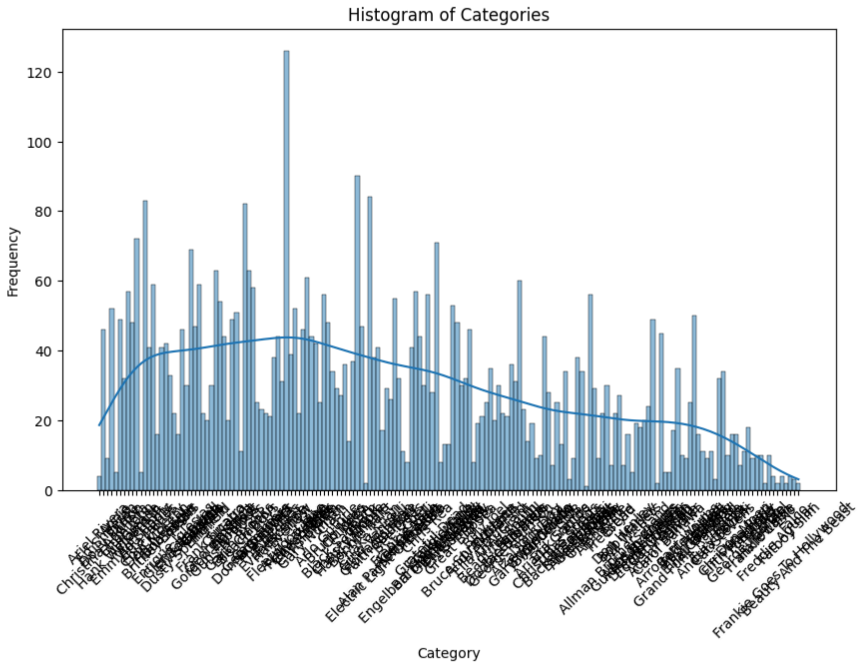
```
import matplotlib.pyplot as plt
import seaborn as sns

# Bar plot
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='artist')
plt.title('Count of Songs by Category')
plt.xlabel('Category')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



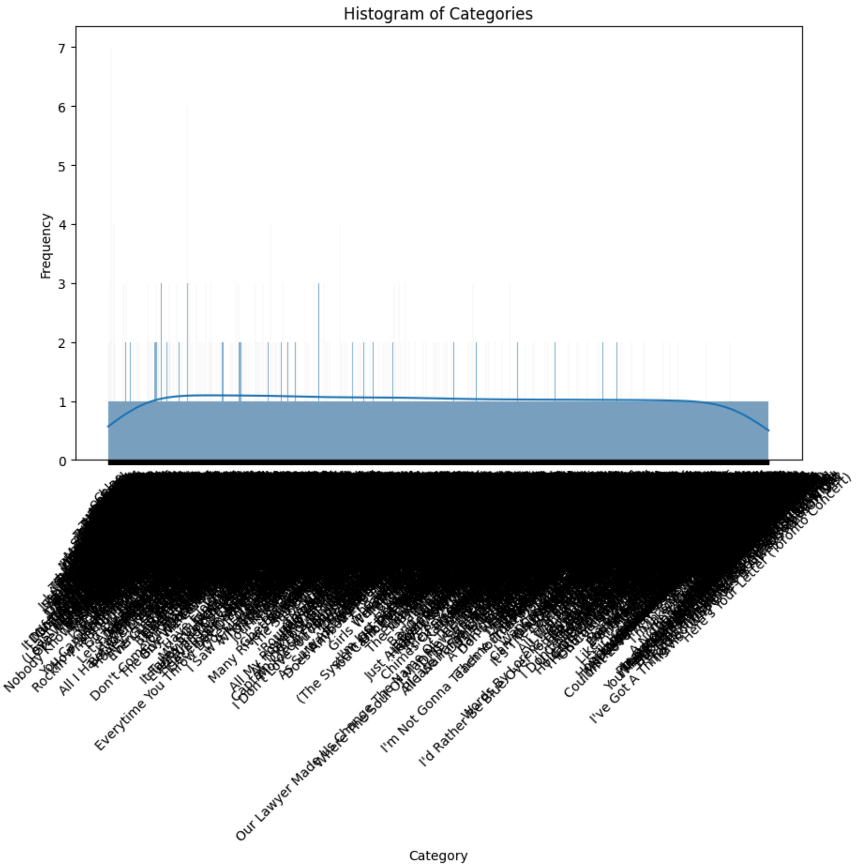
```
import matplotlib.pyplot as plt
import seaborn as sns

# Scatter plot
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='artist', y='song')
plt.title('Scatter Plot of Links by Category')
plt.xlabel('Category')
plt.ylabel('Link')
plt.xticks(rotation=45)
plt.show()
```



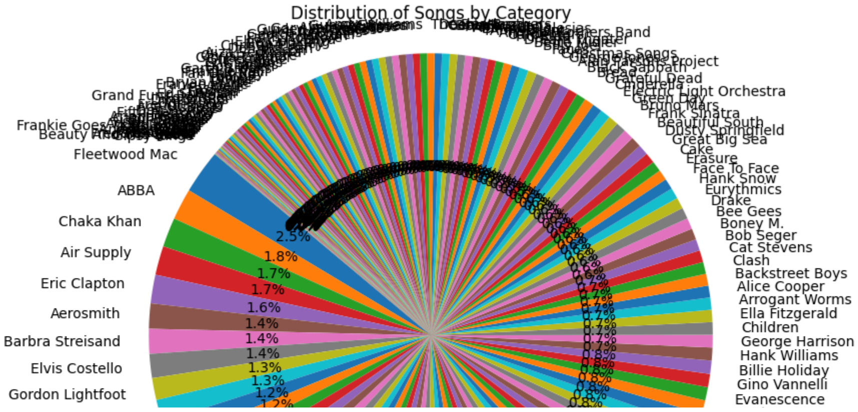
```
import matplotlib.pyplot as plt
import seaborn as sns

# Histogram
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='artist', bins=10, kde=True)
plt.title('Histogram of Categories')
plt.xlabel('Category')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```



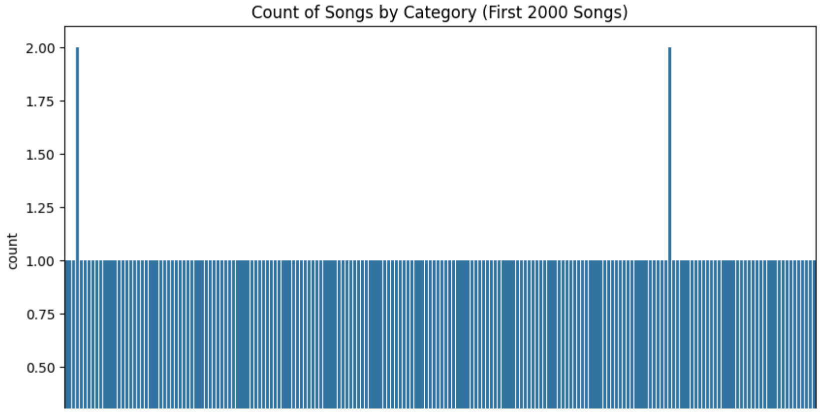
```
import matplotlib.pyplot as plt
import seaborn as sns

# Selecting the first 2000 songs
df_subset = df.head(2000)

# Histogram
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='song', bins=10, kde=True)
plt.title('Histogram of Categories')
plt.xlabel('Category')
plt.ylabel('Frequency')
plt.xticks(rotation=45)
plt.show()
```



```
import matplotlib.pyplot as plt

# Count the number of songs in each category
category_counts = df['artist'].value_counts()

# Create a pie chart
plt.figure(figsize=(8, 8))
plt.pie(category_counts, labels=category_counts.index, autopct='%1.1f%%', startangle=140)
plt.title('Distribution of Songs by Category')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```



```
import matplotlib.pyplot as plt
import seaborn as sns

# Selecting the first 2000 songs
df_subset = df.head(200)

# Bar plot
plt.figure(figsize=(10, 6))
sns.countplot(data=df_subset, x='song')
plt.title('Count of Songs by Category (First 2000 Songs)')
plt.xticks(rotation=45)
plt.show()

# Pie chart
plt.figure(figsize=(8, 8))
category_counts = df_subset['song'].value_counts()
plt.pie(category_counts, labels=category_counts.index, autopct='%1.1f%%', startangle=140)
plt.title('Distribution of Songs by Category (First 2000 Songs)')
plt.axis('equal')
plt.show()
```


Count of Songs by Category (First 2000 Songs)
count