In [10]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [11]:
```
filepath = "C:\\Users\\Dell\\OneDrive - bjoz\\Desktop\\Machine Learning\\Data
```

In [12]:
```
df = pd.read_csv(filepath)
```

In [13]:
```
df
```

Out[13]:

	Unnamed: 0	acousticness	danceability	duration_ms	energy	instrumentalness	key	livene
0	0	0.01020	0.833	204600	0.434	0.021900	2	0.16
1	1	0.19900	0.743	326933	0.359	0.006110	1	0.13
2	2	0.03440	0.838	185707	0.412	0.000234	2	0.15
3	3	0.60400	0.494	199413	0.338	0.510000	5	0.09
4	4	0.18000	0.678	392893	0.561	0.512000	5	0.43
...	...	...	...	...	...	...	...	
2012	2012	0.00106	0.584	274404	0.932	0.002690	1	0.12
2013	2013	0.08770	0.894	182182	0.892	0.001670	1	0.05
2014	2014	0.00857	0.637	207200	0.935	0.003990	0	0.21
2015	2015	0.00164	0.557	185600	0.992	0.677000	1	0.09
2016	2016	0.00281	0.446	204520	0.915	0.000039	9	0.21

2017 rows × 17 columns

In [14]:
```
df.drop("Unnamed: 0", axis=1, inplace=True)
```

In [15]:
```
df
```

Out[15]:

	acousticness	danceability	duration_ms	energy	instrumentalness	key	liveness	loudnes
0	0.01020	0.833	204600	0.434	0.021900	2	0.1650	-8.79
1	0.19900	0.743	326933	0.359	0.006110	1	0.1370	-10.40
2	0.03440	0.838	185707	0.412	0.000234	2	0.1590	-7.14
3	0.60400	0.494	199413	0.338	0.510000	5	0.0922	-15.23
4	0.18000	0.678	392893	0.561	0.512000	5	0.4390	-11.64
...	...	...	...	...	...	...	...	.
2012	0.00106	0.584	274404	0.932	0.002690	1	0.1290	-3.50
2013	0.08770	0.894	182182	0.892	0.001670	1	0.0528	-2.66
2014	0.00857	0.637	207200	0.935	0.003990	0	0.2140	-2.46
2015	0.00164	0.557	185600	0.992	0.677000	1	0.0913	-2.73
2016	0.00281	0.446	204520	0.915	0.000039	9	0.2180	-6.22

2017 rows × 16 columns

In [16]:
```
df.head()
```

Out[16]:

	acousticness	danceability	duration_ms	energy	instrumentalness	key	liveness	loudness
0	0.0102	0.833	204600	0.434	0.021900	2	0.1650	-8.795
1	0.1990	0.743	326933	0.359	0.006110	1	0.1370	-10.401
2	0.0344	0.838	185707	0.412	0.000234	2	0.1590	-7.148
3	0.6040	0.494	199413	0.338	0.510000	5	0.0922	-15.236
4	0.1800	0.678	392893	0.561	0.512000	5	0.4390	-11.648

Data Cleaning

In [18]:
```
df.isna().sum()
```

Out[18]:
```
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
target              0
song_title          0
artist              0
dtype: int64
```

In [19]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2017 entries, 0 to 2016
Data columns (total 16 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   acousticness      2017 non-null   float64
 1   danceability      2017 non-null   float64
 2   duration_ms       2017 non-null   int64  
 3   energy            2017 non-null   float64
 4   instrumentalness  2017 non-null   float64
 5   key               2017 non-null   int64  
 6   liveness          2017 non-null   float64
 7   loudness          2017 non-null   float64
 8   mode              2017 non-null   int64  
 9   speechiness       2017 non-null   float64
 10  tempo             2017 non-null   float64
 11  time_signature    2017 non-null   float64
 12  valence           2017 non-null   float64
 13  target            2017 non-null   int64  
 14  song_title        2017 non-null   object 
 15  artist            2017 non-null   object 
dtypes: float64(10), int64(4), object(2)
memory usage: 252.2+ KB
```

In [20]:
```
df.shape
```

Out[20]:
```
(2017, 16)
```

In [21]:
```
df.columns
```

Out[21]:
```
Index(['acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence', 'target',
       'song_title', 'artist'],
      dtype='object')
```

In [22]:
```
len(df.columns)
```

Out[22]:
```
16
```

In [23]:
```
df.describe()
```

Out[23]:

	acousticness	danceability	duration_ms	energy	instrumentalness	key	
count	2017.000000	2017.000000	2.017000e+03	2017.000000	2017.000000	2017.000000	20
mean	0.187590	0.618422	2.463062e+05	0.681577	0.133286	5.342588	
std	0.259989	0.161029	8.198181e+04	0.210273	0.273162	3.648240	
min	0.000003	0.122000	1.604200e+04	0.014800	0.000000	0.000000	
25%	0.009630	0.514000	2.000150e+05	0.563000	0.000000	2.000000	
50%	0.063300	0.631000	2.292610e+05	0.715000	0.000076	6.000000	
75%	0.265000	0.738000	2.703330e+05	0.846000	0.054000	9.000000	
max	0.995000	0.984000	1.004627e+06	0.998000	0.976000	11.000000	

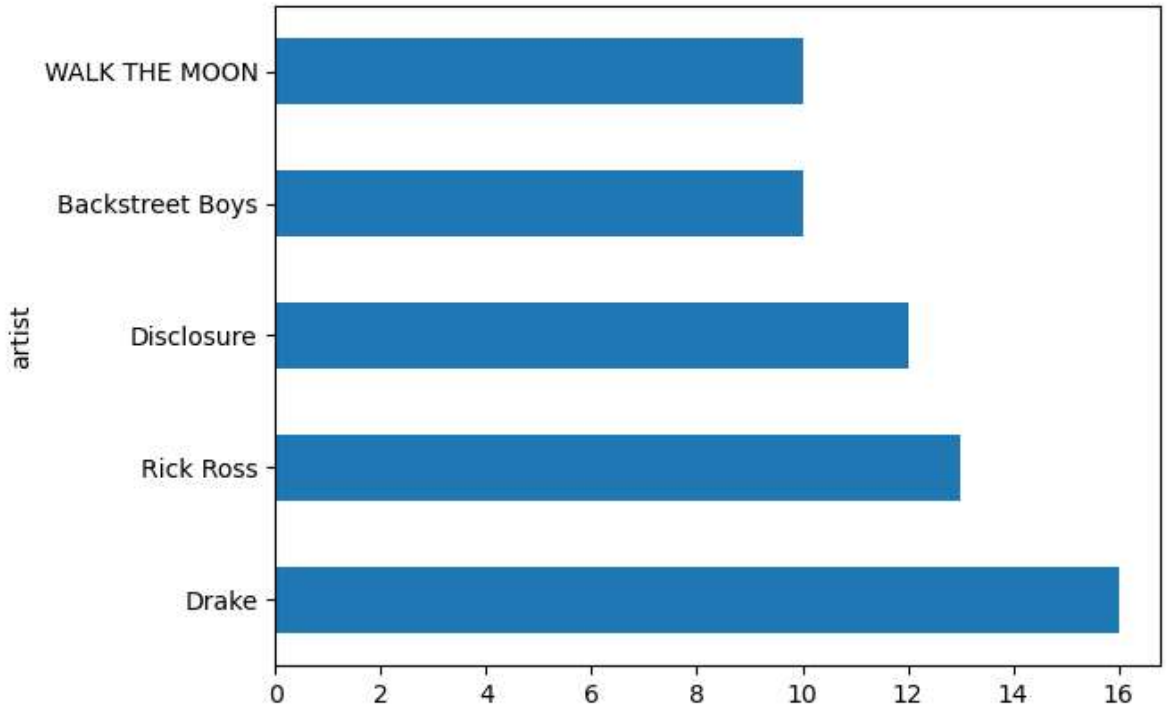
Data Analysis

Top 5 most popular artists

In [25]:
```
top_five_artists=df.groupby("artist").count().sort_values(by="song_title", asc
top_five_artists
```

Out[25]:
```
artist
Drake              16
Rick Ross          13
Disclosure         12
Backstreet Boys    10
WALK THE MOON      10
Name: song_title, dtype: int64
```

In [26]:
```
top_five_artists.plot.barh()
plt.show()
```



Top 5 loudest tracks

In [27]:
```
top_five_loudest_tracks=df[["loudness", "song_title"]].sort_values(by="loudnes
top_five_loudest_tracks
```

Out[27]:

	loudness	song_title
195	-0.307	GodLovesUgly
636	-0.718	The Lion - Original Mix
1443	-0.787	The Wall
2010	-0.935	Hey Baby - Steve Aoki Remix
1299	-0.994	No Absolution
...	...	...
1549	-29.460	Eleanor
1531	-30.447	I Was So Young, and You Were So Beautiful
1598	-31.082	Piano Quartet in E flat, Op.47: 3. Andante can...
1596	-31.367	8 Fantasiestücke, Op.12 : 1. Des Abends
1594	-33.097	Lyric Pieces, Book I Op. 12: I. Arietta

2017 rows × 2 columns

In [38]:
```
df.columns
```

Out[38]:
```
Index(['acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence', 'target',
       'song_title', 'artist'],
      dtype='object')
```

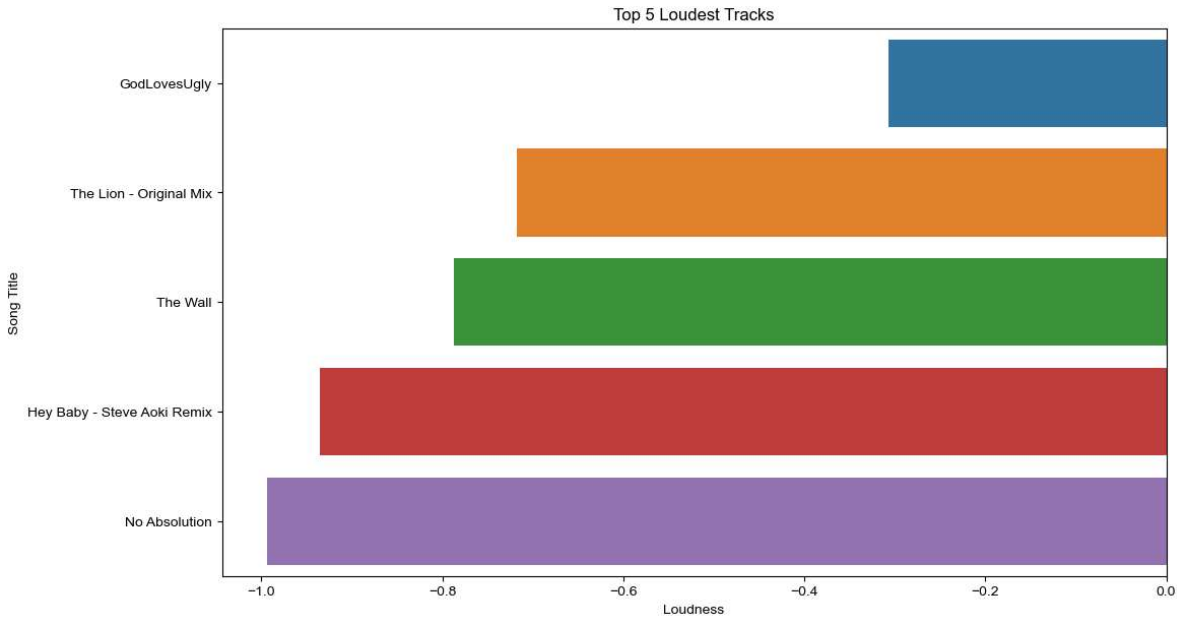
In [42]:
```
print(df.columns)
```

```
Index(['acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence', 'target',
       'song_title', 'artist'],
      dtype='object')
```

In [65]:
```
import matplotlib.pyplot as plt
plt.rcParams["font.family"] = "Arial"
```

In [69]:
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(12, 7))
sns.barplot(x="loudness", y="song_title", data=top_five_loudest_tracks.head())
plt.title("Top 5 Loudest Tracks")
plt.xlabel("Loudness")
plt.ylabel("Song Title")
plt.show()
```



Artist with the most danceability song

In [70]:
```
top_five_artists_danceable_songs = df[["danceability", "song_title", "artist"]]
top_five_artists_danceable_songs
```

Out[70]:

	danceability	song_title	artist
1433	0.984	Flashwind - Radio Edit	Ben Remember
1901	0.967	SexyBack	Justin Timberlake
604	0.962	Check Me Out Like	Blaqstarr
32	0.959	Best Friend	Young Thug
1957	0.959	Ice Ice Baby	Vanilla Ice
...	...	...	...
1598	0.156	Piano Quartet in E flat, Op.47: 3. Andante can...	Robert Schumann
1600	0.152	Trio Sonata in G Major, Wq. 144: I. Adagio	Carl Philipp Emanuel Bach
817	0.148	Mozart: Requiem in D Minor, K. 626: VIII. Lacr...	Nikolaus Harnoncourt
532	0.123	Wake Bake Skate	FIDLAR
729	0.122	Bumpy Road	Destruction Unit

2017 rows × 3 columns

In [33]:
```
print(df.columns)
```

```
Index(['acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence', 'target',
       'song_title', 'artist'],
      dtype='object')
```

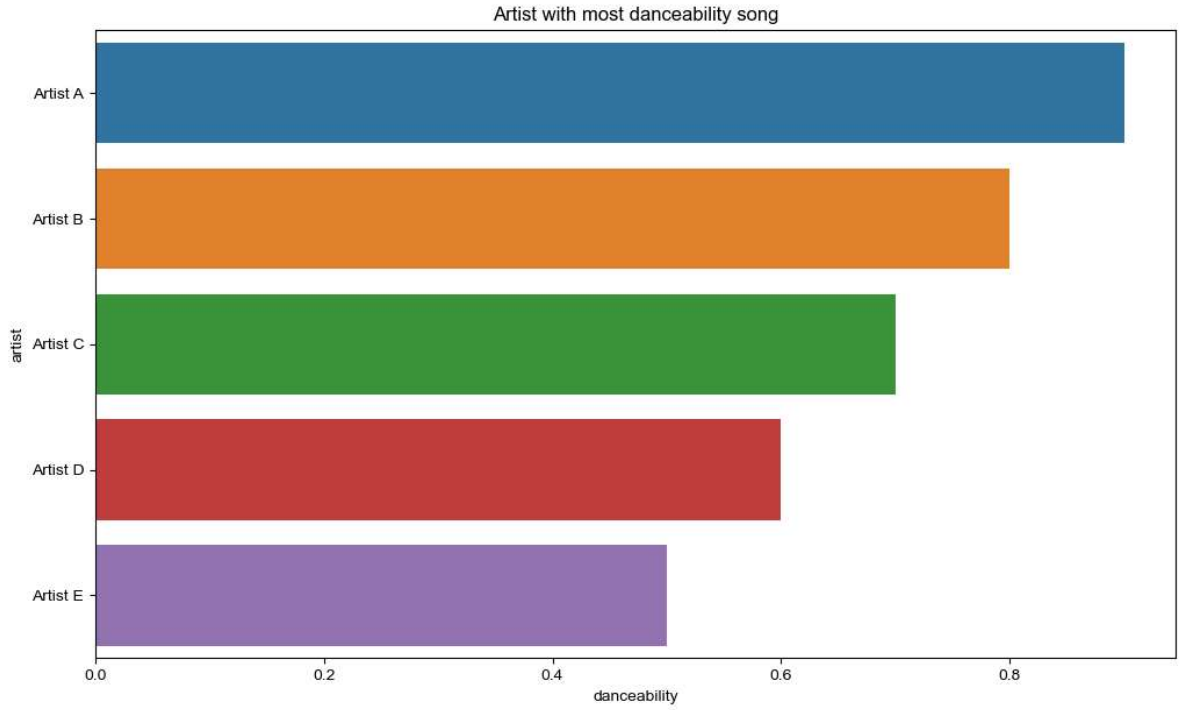
In [37]:
```
df.columns
```

Out[37]:
```
Index(['acousticness', 'danceability', 'duration_ms', 'energy',
       'instrumentalness', 'key', 'liveness', 'loudness', 'mode',
       'speechiness', 'tempo', 'time_signature', 'valence', 'target',
       'song_title', 'artist'],
      dtype='object')
```

In [67]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Define top_five_artists_danceable_songs
top_five_artists_danceable_songs = pd.DataFrame({
    'artist': ['Artist A', 'Artist B', 'Artist C', 'Artist D', 'Artist E'],
    'danceability': [0.9, 0.8, 0.7, 0.6, 0.5]
})

# Create bar plot
plt.figure(figsize=(12, 7))
sns.barplot(x="danceability",  y="artist", data=top_five_artists_danceable_son
plt.title("Artist with most danceability song")
plt.show()
```



Top 10 Instumentalness tracks

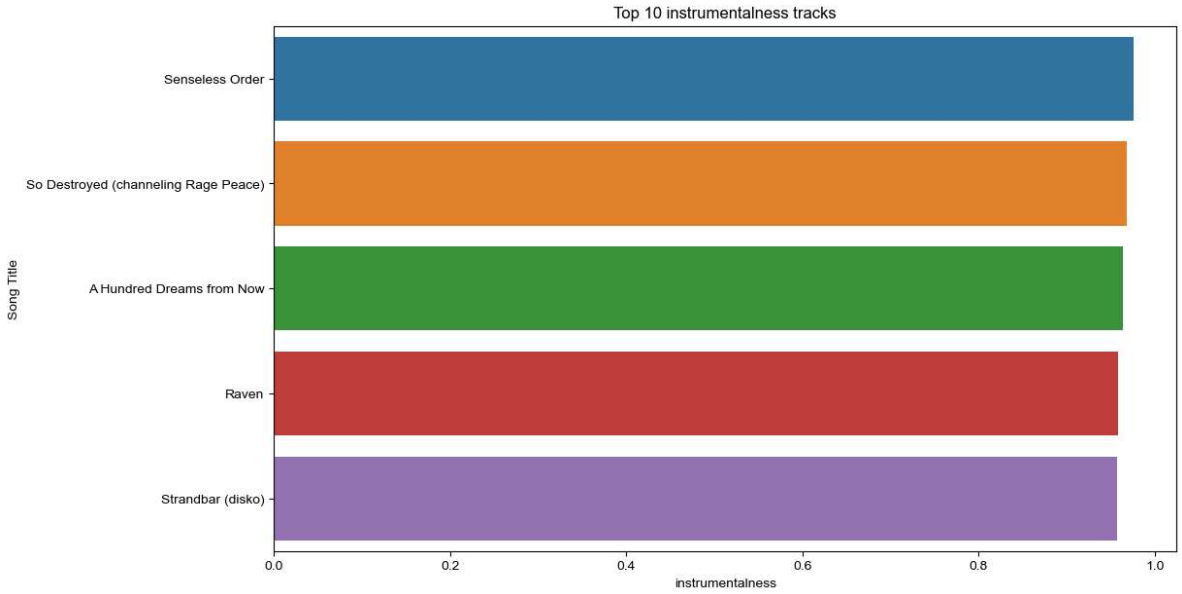
In [50]:
```
top_ten_instrumental_tracks = df[["instrumentalness", "song_title", "artist"]]
```

In [51]:
```
top_ten_instrumental_tracks
```

Out[51]:

	instrumentalness	song_title	artist
1313	0.976	Senseless Order	Signs of the Swarm
271	0.968	So Destroyed (channeling Rage Peace)	Prince Rama
1575	0.964	A Hundred Dreams from Now	Ray Bryant
1619	0.958	Raven	John Dahlbäck
725	0.957	Strandbar (disko)	Todd Terje

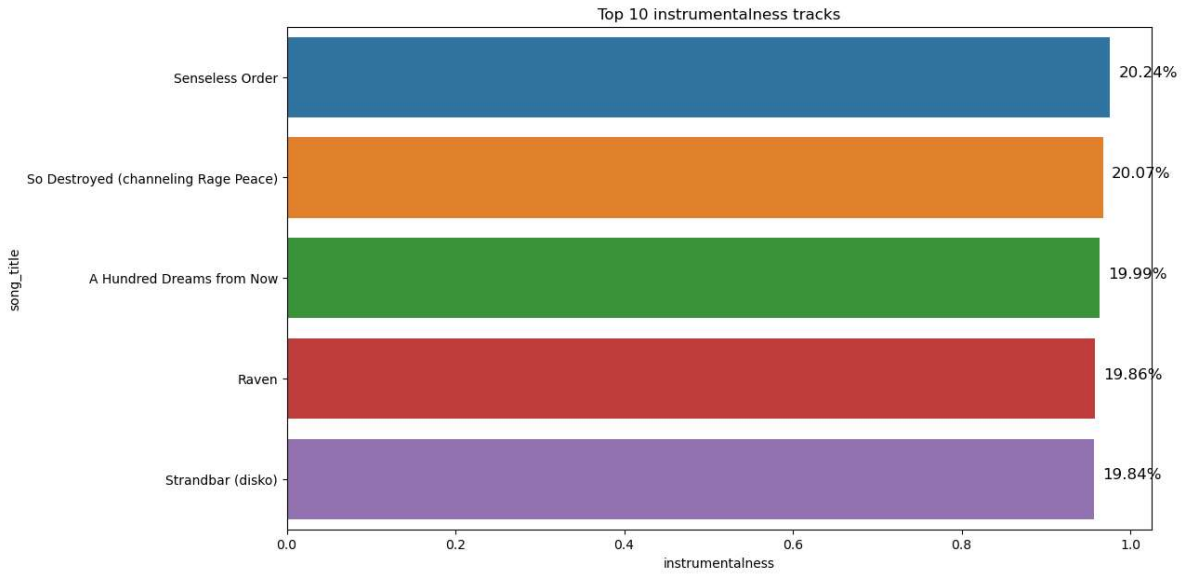
In [68]:
```
plt.figure(figsize=(12, 7))
ax = sns.barplot(x="instrumentalness", y="song_title", data=top_ten_instrument
ax.set_title("Top 10 instrumentalness tracks")
ax.set_ylabel("Song Title")
plt.show()
```



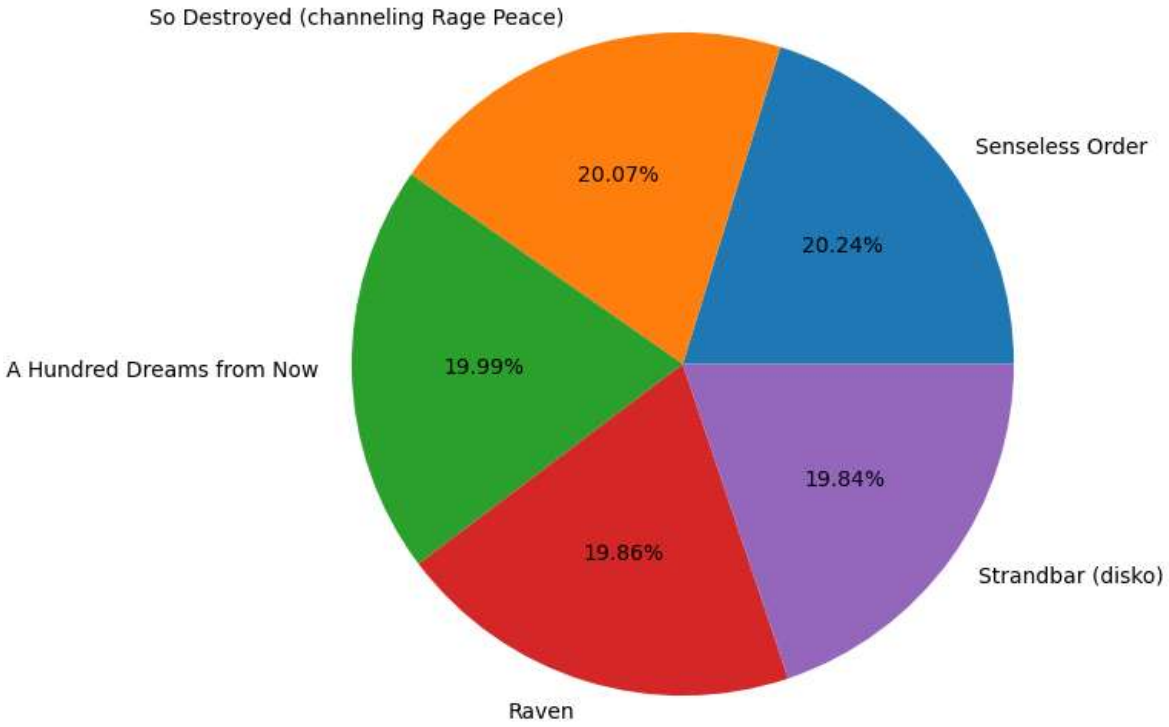
In [55]:
```
plt.figure(figsize=(12, 7))
ax = sns.barplot(x="instrumentalness", y="song_title", data=top_ten_instrument
plt.title("Top 10 instrumentalness tracks")

# Compute percentage values
total = top_ten_instrumental_tracks["instrumentalness"].sum()
for i, v in enumerate(top_ten_instrumental_tracks["instrumentalness"]):
    pct = v / total * 100
    plt.text(v + 0.01, i, f"{pct:.2f}%", color="black", fontsize=12)

plt.show()
```



In [56]:
```
plt.figure(figsize=(12, 7))
plt.pie(x="instrumentalness",  data=top_ten_instrumental_tracks, autopct='%1.2
plt.show()
```



Multiple feature plots

In [64]:
```
interest_feature_cols = ["tempo", "loudness", "acousticness", "danceability",
```

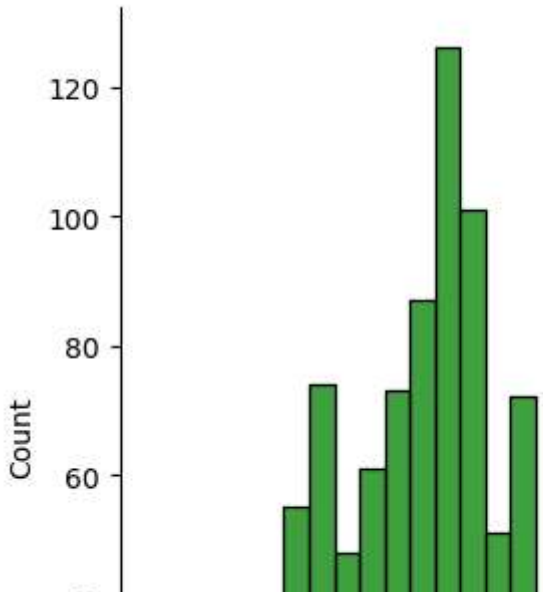
In [ ]:
```
for feature_col in interest_feature_cols:
    pos_data = df[df["target"]==1][feature_col]
    neg_data = df[df["target"]==0][feature_col]

    plt.figure(figsize=(12,7))

    sns.displot(pos_data, bins=30, label='Positive', color='green')
    sns.displot(neg_data, bins=30, label='Negative', color='red')

    plt.legend(loc="upper right")
    plt.title(f"Positive And Negative Histogram Plot For {feature_col}")
    plt.show()
```

```
<Figure size 1200x700 with 0 Axes>
```



In [ ]: