Spotify Dataset EDA

1. Importing the required libraries for EDA

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns                          #visualisation
import matplotlib.pyplot as plt                #visualisation
%matplotlib inline
sns.set(color_codes=True)
```

2. Importing the dataset

In [3]:
```
df = pd.read_csv(r"D:\spotify_dataset.csv")
# To display the top 5 rows
df.head(5)
```

Out[3]:

	Index	Highest Charting Position	Number of Times Charted	Week of Highest Charting	Song Name	Streams	Artist	Artist Followers	
0	1	1	8	2021-07-23--2021-07-30	Beggin'	48,633,449	Måneskin	3377762	3Wrjm47
1	2	2	3	2021-07-23--2021-07-30	STAY (with Justin Bieber)	47,248,719	The Kid LAROI	2230022	5HCyWlXZPF
2	3	1	11	2021-06-25--2021-07-02	good 4 u	40,162,559	Olivia Rodrigo	6266514	4ZtFanR9U
3	4	3	5	2021-07-02--2021-07-09	Bad Habits	37,799,456	Ed Sheeran	83293380	6PQ88X9TkU
4	5	5	1	2021-07-23--2021-07-30	INDUSTRY BABY (feat. Jack Harlow)	33,948,454	Lil Nas X	5473565	27NovPIUIRr

5 rows × 23 columns

In [4]:
```
df.tail(5)                    # To display the botton 5 rows
```

Out[4]:

	Index	Highest Charting Position	Number of Times Charted	Week of Highest Charting	Song Name	Streams	Artist	Artist Followers	
1551	1552	195	1	2019-12-27--2020-01-03	New Rules	4,630,675	Dua Lipa	27167675	2ekn2ttSfG
1552	1553	196	1	2019-12-27--2020-01-03	Cheirosa - Ao Vivo	4,623,030	Jorge & Mateus	15019109	2PWjKmjyTZe
1553	1554	197	1	2019-12-27--2020-01-03	Havana (feat. Young Thug)	4,620,876	Camila Cabello	22698747	1rfofaqEpAC
1554	1555	198	1	2019-12-27--2020-01-03	Surtada - Remix Brega Funk	4,607,385	Dadá Boladão, Tati Zaqui, OIK	208630	5F8ffc8KWk
1555	1556	199	1	2019-12-27--2020-01-03	Lover (Remix) [feat. Shawn Mendes]	4,595,450	Taylor Swift	42227614	3i9UVIdZOE

5 rows × 23 columns

3. Checking the types of data

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

Out[5]:
```
Index                          int64
Highest Charting Position      int64
Number of Times Charted        int64
Week of Highest Charting      object
Song Name                     object
Streams                       object
Artist                        object
Artist Followers              object
Song ID                       object
Genre                         object
Release Date                  object
Weeks Charted                 object
Popularity                    object
Danceability                  object
Energy                        object
Loudness                      object
Speechiness                   object
Acousticness                  object
Liveness                      object
Tempo                         object
Duration (ms)                 object
Valence                       object
Chord                         object
dtype: object
```

4. Feature engineering

Dropping the columns which does not play an important role in the data analysis

In [6]:
```
df = df.drop(['Song ID', 'Energy', 'Loudness', 'Speechiness', 'Acousticness',
'Index', 'Liveness', 'Tempo','Valence','Chord','Danceability','Release Date'],
axis=1)
df.head(5)
```

Out[6]:

	Highest Charting Position	Number of Times Charted	Week of Highest Charting	Song Name	Streams	Artist	Artist Followers	Genre	Week Charte
0	1	8	2021-07-23--2021-07-30	Beggin'	48,633,449	Måneskin	3377762	['indie rock italiano', 'italian pop']	2021-07 23--2021 07 30\n2021 07-16 -2021-07 23.
1	2	3	2021-07-23--2021-07-30	STAY (with Justin Bieber)	47,248,719	The Kid LAROI	2230022	['australian hip hop']	2021-07 23--2021 07 30\n2021 07-16 -2021-07 23.
2	1	11	2021-06-25--2021-07-02	good 4 u	40,162,559	Olivia Rodrigo	6266514	['pop']	2021-07 23--2021 07 30\n2021 07-16 -2021-07 23.
3	3	5	2021-07-02--2021-07-09	Bad Habits	37,799,456	Ed Sheeran	83293380	['pop', 'uk pop']	2021-07 23--2021 07 30\n2021 07-16 -2021-07 23.
4	5	1	2021-07-23--2021-07-30	INDUSTRY BABY (feat. Jack Harlow)	33,948,454	Lil Nas X	5473565	['lgbtq+ hip hop', 'pop rap']	2021-07 23--2021 07-3

Dropping the duplicate rows & NA values(if any)

In [7]:
```
df.shape
```

Out[7]: (1556, 11)

In [8]:
```
duplicate_rows_df = df[df.duplicated()]
print("number of duplicate rows: ", duplicate_rows_df.shape)
```

```
number of duplicate rows:  (0, 11)
```

In [9]:
```
print(df.isnull().sum())
```

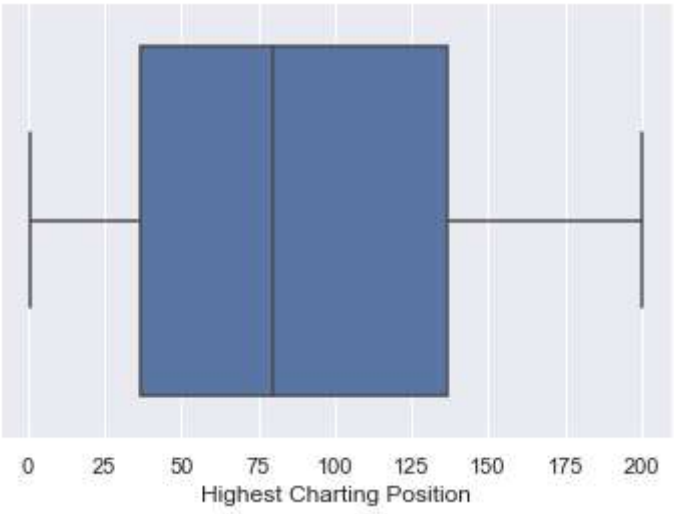
```
Highest Charting Position    0
Number of Times Charted      0
Week of Highest Charting     0
Song Name                    0
Streams                      0
Artist                       0
Artist Followers             0
Genre                        0
Weeks Charted                0
Popularity                   0
Duration (ms)                0
dtype: int64
```

Since there are no duplicate rows and NA values , let us proceed further

Detecting Outliers

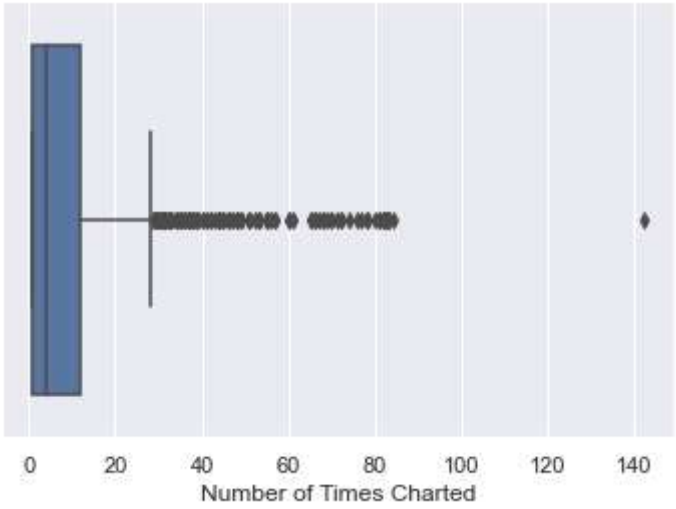
In [10]:
```
sns.boxplot(x=df['Highest Charting Position'])
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d5de33f730>
```



In [11]:
```
sns.boxplot(x=df['Number of Times Charted'])
```

Out[11]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1d5de3e50d0>
```



In [12]:
```
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
Highest Charting Position    100.0
Number of Times Charted       11.0
dtype: float64
```

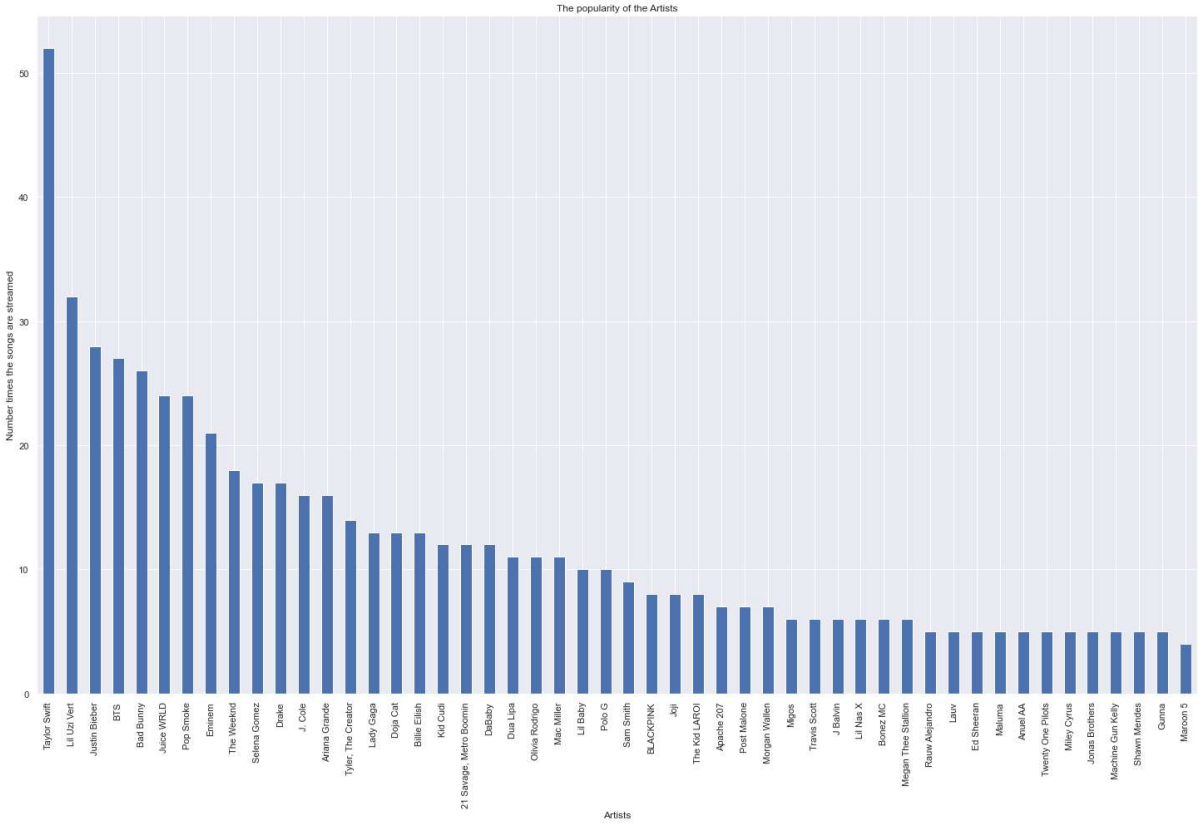
using this technique in order to remove the outliers.

In [13]:
```
df = df[~((df < (Q1 - 1.5 * IQR)) |(df > (Q3 + 1.5 * IQR))).any(axis=1)]
df.shape
```

Out[13]:
```
(1385, 11)
```

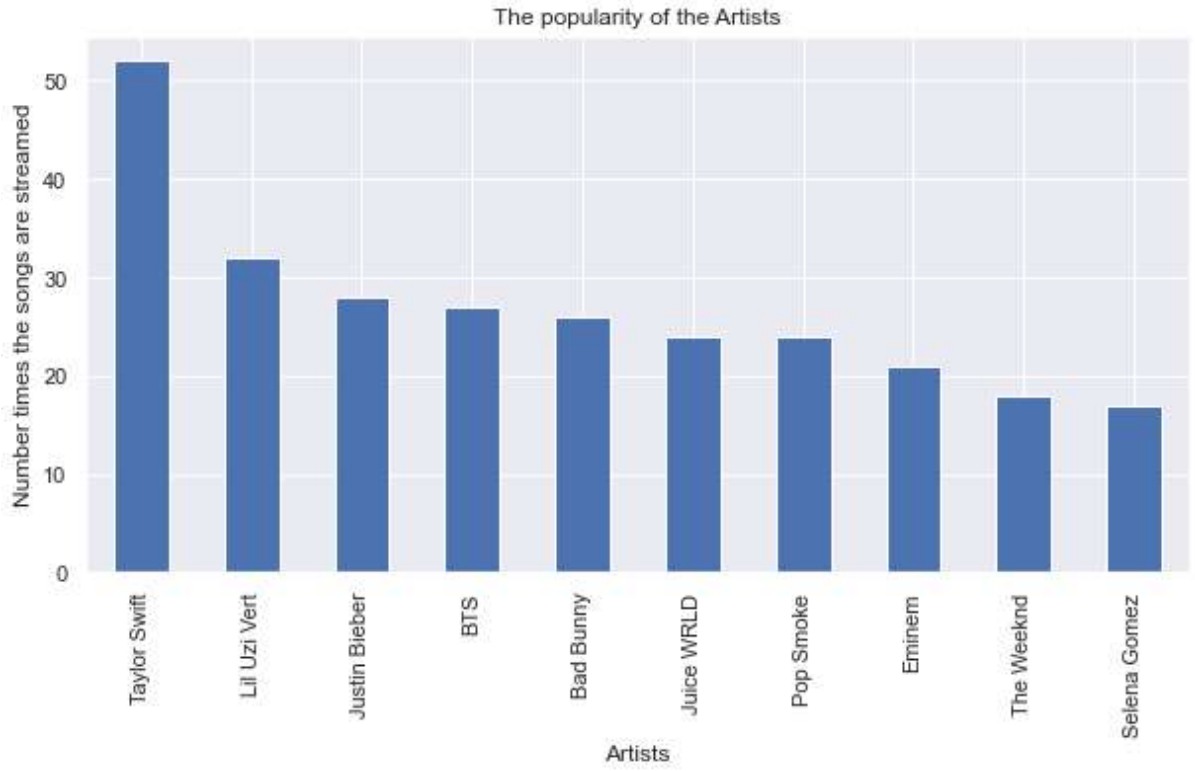
In [ ]:
```

```

In [14]:
```
df.Artist.value_counts().nlargest(50).plot(kind='bar', figsize=(25,15))
plt.title("The popularity of the Artists")
plt.ylabel('Number times the songs are streamed')
plt.xlabel('Artists');
```



For the Top 10

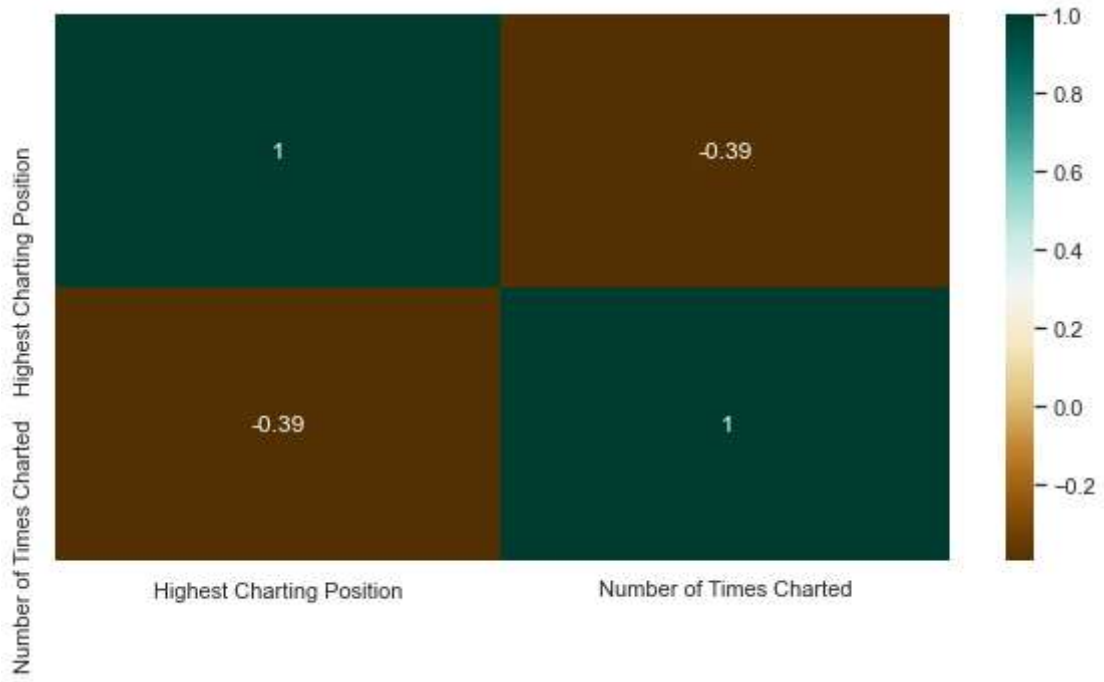
In [15]:
```
df.Artist.value_counts().nlargest(10).plot(kind='bar', figsize=(10,5))
plt.title("The popularity of the Artists")
plt.ylabel('Number times the songs are streamed')
plt.xlabel('Artists');
```



In [16]:
```
plt.figure(figsize=(10,5))
c= df.corr()
sns.heatmap(c,cmap="BrBG",annot=True)
c
```

Out[16]:

	Highest Charting Position	Number of Times Charted
Highest Charting Position	1.000000	-0.394955
Number of Times Charted	-0.394955	1.000000



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: