In [39]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import seaborn as sns
```

In [40]:
```
df1=pd.read_csv(r"C:\Users\arumu\Downloads\spotify dataset.csv")
df1
```

Out[40]:

	track_id	track_name	track_artist	track_popularity	track_album_id	track_album_name	tra
0	6f807x0ima9a1j3VPbc7VN	I Don't Care (with Justin Bieber) - Loud Luxur...	Ed Sheeran	66	2oCs0DGTsRO98Gh5ZSl2Cx	I Don't Care (with Justin Bieber) [Loud Luxury...	
1	0r7CVbZTWZgbTCYdfa2P31	Memories - Dillon Francis Remix	Maroon 5	67	63rPSO264uRjW1X5E6cWv6	Memories (Dillon Francis Remix)	
2	1z1Hg7Vb0AhHDiEmnDE79l	All the Time - Don Diablo Remix	Zara Larsson	70	1HoSmj2eLcsrR0vE9gThr4	All the Time (Don Diablo Remix)	
3	75FpbthrwQmzHlBJLuGdC7	Call You Mine - Keanu Silva Remix	The Chainsmokers	60	1nqYsOef1yKKuGOVchbsk6	Call You Mine - The Remixes	
4	1e8PAfcKUYoKkxPhrHqw4x	Someone You Loved - Future Humans Remix	Lewis Capaldi	69	7m7vv9wlQ4i0LFuJiE2zsQ	Someone You Loved (Future Humans Remix)	
...	...	...	...	...	...	...	
32828	7bxnKAamR3snQ1VGLuVfC1	City Of Lights - Official Radio Edit	Lush & Simon	42	2azRoBBWEEEYhqV6sb7JrT	City Of Lights (Vocal Mix)	
32829	5Aevni09Em4575077nkWHz	Closer - Sultan & Ned Shepard Remix	Tegan and Sara	20	6kD6KLxj7s8eCE3ABvAyf5	Closer Remixed	
32830	7ImMqPP3Q1yfUHvsdn7wEo	Sweet Surrender - Radio Edit	Starkillers	14	0ltWNSY9JgxoIZO4VzuCa6	Sweet Surrender (Radio Edit)	
32831	2m69mhnfQ1Oq6lGtXuYhgX	Only For You - Maor Levi Remix	Mat Zo	15	1fGrOkHnHJcStl14zNx8Jy	Only For You (Remixes)	
32832	29zWqhca3zt5NsckZqDf6c	Typhoon - Original Mix	Julian Calor	27	0X3mUOm6MhxR7PzxG95rAo	Typhoon/Storm	

32833 rows × 23 columns

In [41]:
```
features = ['playlist_name', 'playlist_genre', 'playlist_subgenre', 'danceability',
            'energy', 'loudness', 'valence', 'tempo']
```

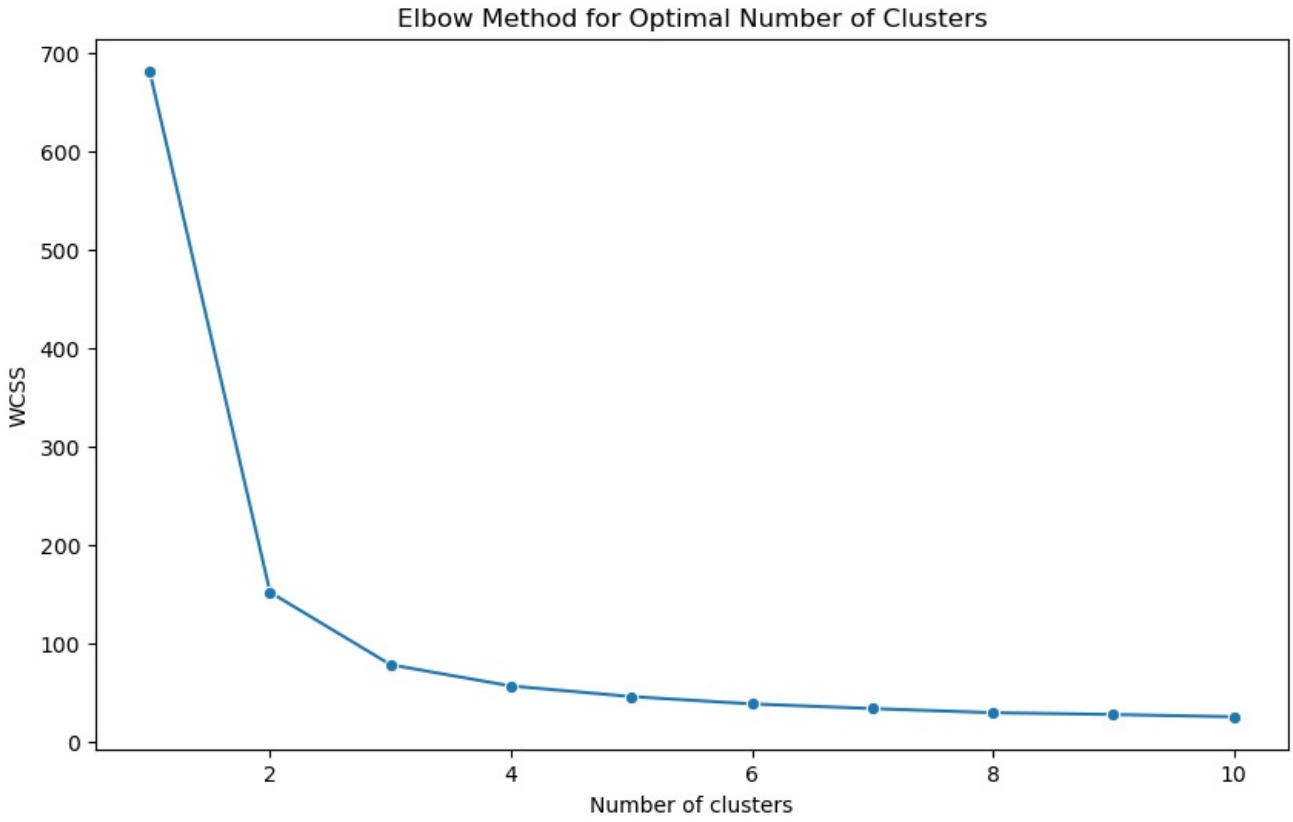
In [43]:
```
from sklearn.datasets import load_iris
data = load_iris().data
```

In [46]:
```
from sklearn.decomposition import PCA
pca = PCA(n_components=4)  # Example for reducing to 2 components
data_pca = pca.fit_transform(data)
```

In [ ]:
```
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=42)
    kmeans.fit(data_pca)
    wcss.append(kmeans.inertia_)
```

In [48]:
```
plt.figure(figsize=(10, 6))
sns.lineplot(x=range(1, 11), y=wcss, marker='o')
plt.title('Elbow Method for Optimal Number of Clusters')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
```

```
plt.show()
```



In [ ]:
```
optimal_clusters = 100# Adjust this based on the elbow plot
kmeans = KMeans(n_clusters=optimal_clusters, init='k-means++', max_iter=300, n_init=10, random_state=42)
clusters = kmeans.fit_predict(data_pca)
```

In [51]:
```
plt.figure(figsize=(10, 6))
sns.scatterplot(x=data_pca[:, 0], y=data_pca[:, 1], hue=clusters, palette='viridis', legend='full')
plt.title('Clusters based on Playlist Genres and Musical Features')
plt.xlabel('PCA Component 1')
plt.ylabel('PCA Component 2')
plt.show()
```