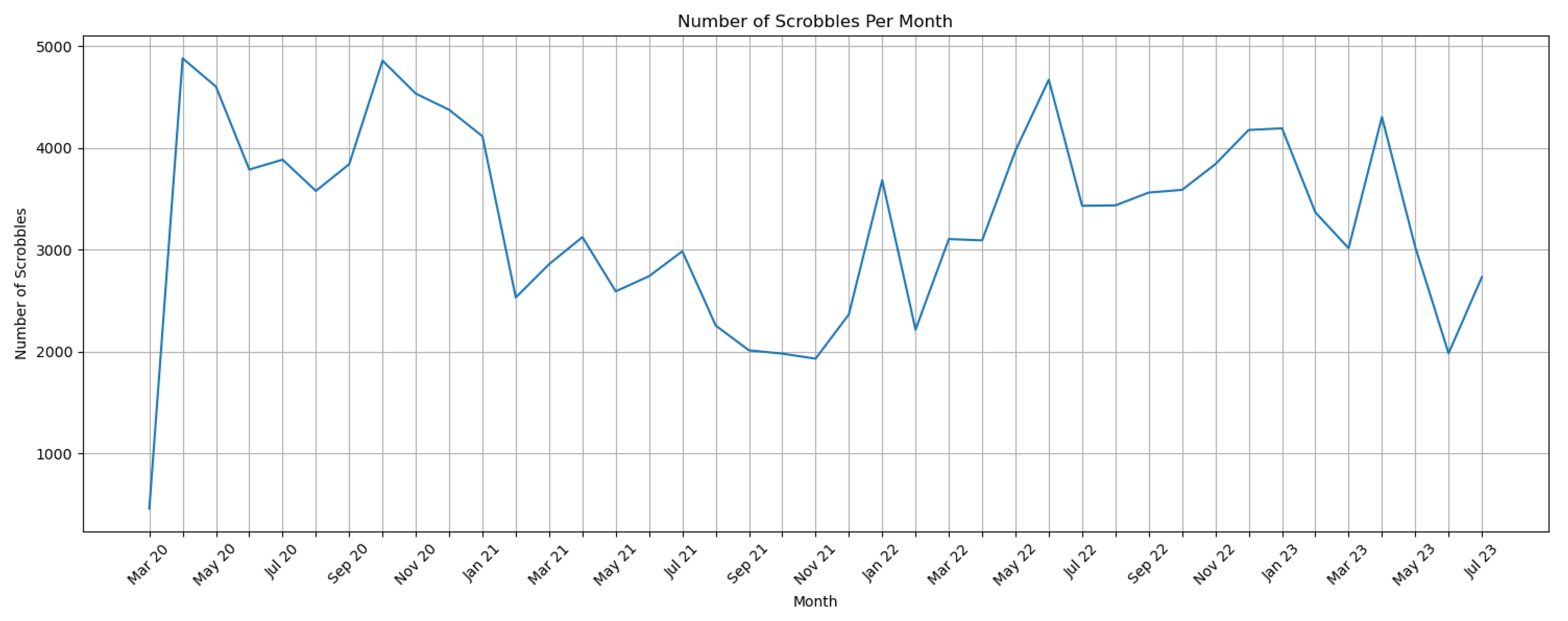
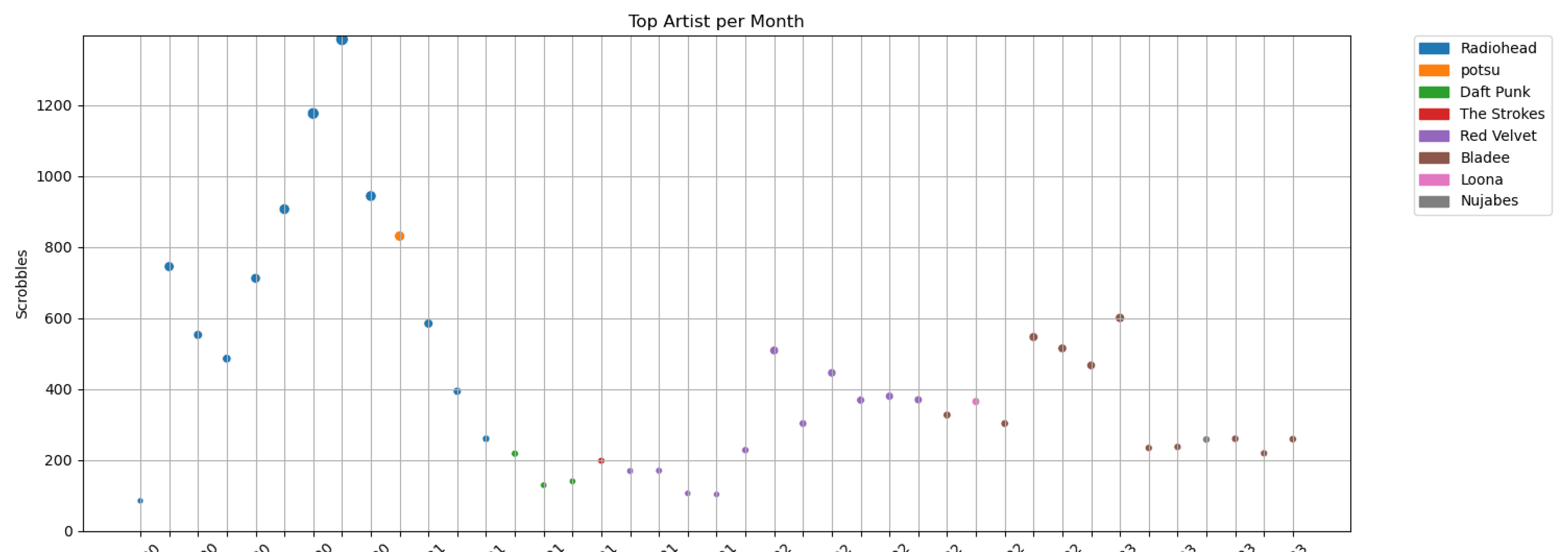
In [1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.patches as mpatches
from datetime import datetime
import re
import numpy as np

def convert_date(date_str):
    return datetime.strptime(date_str, '%d %b %Y, %H:%M')

# Load the data
df = pd.read_csv('lastfm_data2.csv')

# Remove leading and trailing whitespace from column names
df.columns = df.columns.str.strip()

# Drop rows with missing date values
df = df.dropna(subset=['date'])

# Apply the function to the 'date' column
df['date'] = df['date'].apply(convert_date)

# Set the date as the DataFrame's index
df.set_index('date', inplace=True)

# Reset the index of the DataFrame
df_reset = df.reset_index()

# Extract the year and month from the date
df_reset['year'] = df_reset['date'].dt.year
df_reset['month'] = df_reset['date'].dt.month

# Create a 'year-month' column for grouping
df_reset['year_month'] = df_reset['date'].dt.to_period('M')

# Group by 'year_month' and count the number of scrobbles
monthly_scrobbles = df_reset.groupby('year_month').size().reset_index(name='counts')

# Convert 'year_month' to string format in 'Mon YY' format
monthly_scrobbles['year_month'] = monthly_scrobbles['year_month'].dt.strftime('%b %y')

# Plot the result using seaborn
plt.figure(figsize=(15,6))
plot = sns.lineplot(data=monthly_scrobbles, x='year_month', y='counts')
plt.title('Number of Scrobbles Per Month')
plt.xlabel('Month')
plt.ylabel('Number of Scrobbles')
plt.grid(True)

# Get the current x-tick labels and set every 2nd label to visible
for ind, label in enumerate(plot.get_xticklabels()):
    if ind % 2 == 0:  # every 2nd label is kept
        label.set_visible(True)
    else:
        label.set_visible(False)

plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [2]:
```python
# Group by 'year_month' and 'artist' to get scrobbles per artist per month
artist_counts = df_reset.groupby(['year_month', 'artist']).size().reset_index(name='counts')

# Get the top artist for each month
top_artist_per_month = artist_counts.loc[artist_counts.groupby('year_month')['counts'].idxmax()]
```

In [4]:
```python
# Convert 'year_month' back to datetime format for sorting
top_artist_per_month['year_month'] = pd.to_datetime(top_artist_per_month['year_month'].dt.strftime('%Y-%m'), format='%Y-%m')

# Sort the DataFrame by 'year_month' before plotting
top_artist_per_month = top_artist_per_month.sort_values('year_month')

# Convert 'year_month' back to string format for plotting
top_artist_per_month['year_month'] = top_artist_per_month['year_month'].dt.strftime('%b %y')

# Create a dot plot with seaborn
plt.figure(figsize=(15,6))
plot = sns.scatterplot(data=top_artist_per_month, x='year_month', y='counts', size='counts', hue='artist', palette='tab10', alpha=1, legend=False)

# Create a color legend
artists = top_artist_per_month['artist'].unique()
colors = sns.color_palette('tab10', n_colors=len(artists))
patches = [mpatches.Patch(color=colors[i], label=artist) for i, artist in enumerate(artists)]
plt.legend(handles=patches, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.)

# Set the size range for the dots
plot.set_ylim(0, top_artist_per_month['counts'].max() + 10)

# Adjust the x-axis labels to show only every other month
for ind, label in enumerate(plot.get_xticklabels()):
    if ind % 2 == 0:  # every 2nd label is kept
        label.set_visible(True)
    else:
        label.set_visible(False)

plt.title('Top Artist per Month')
plt.xlabel('Month')
plt.ylabel('Scrobbles')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

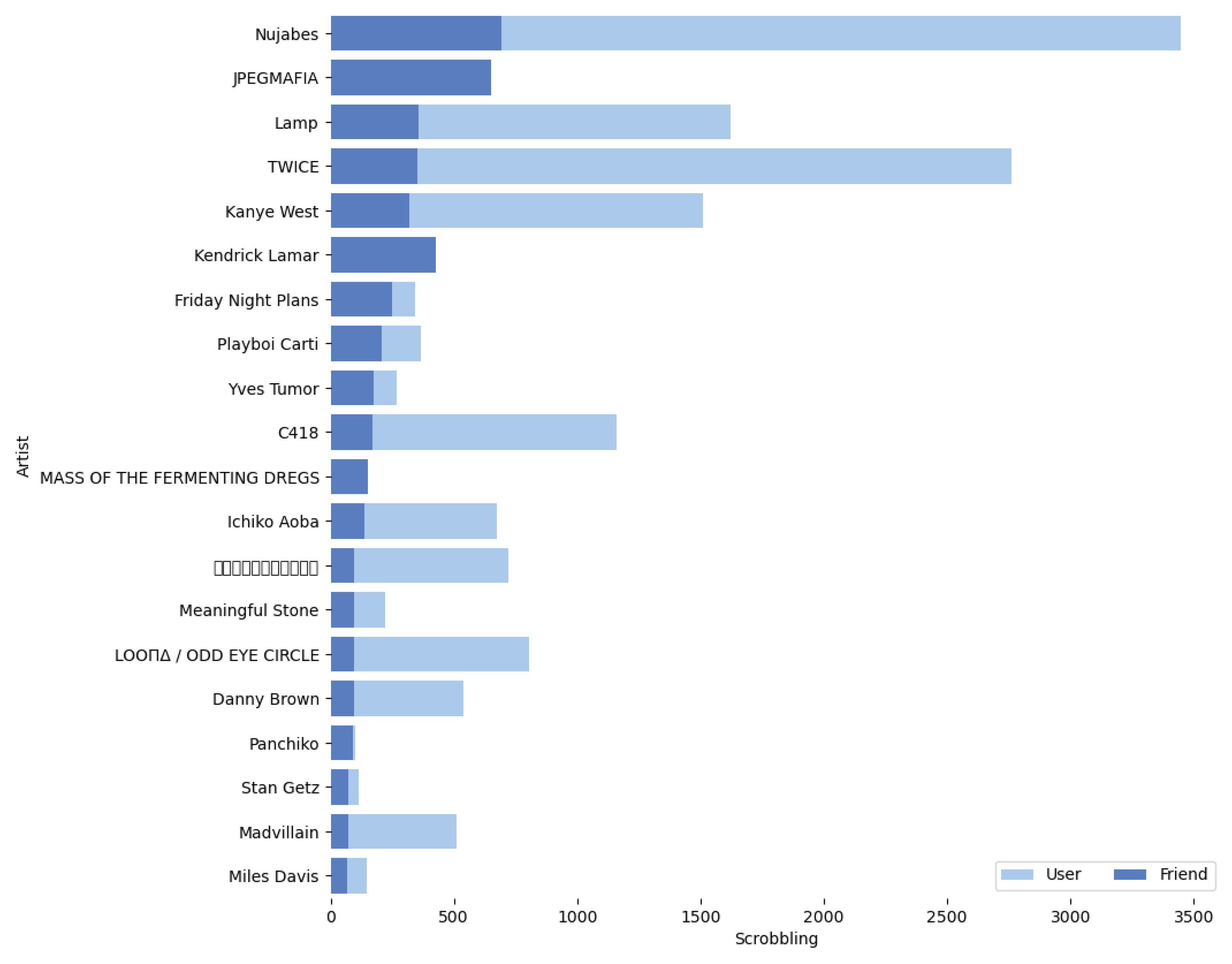


In [8]:
```python
# Load friend's dataset
df_friend = pd.read_csv('gol3m521.csv')

# Check the first few rows of the DataFrame
print(df_friend.head())
```

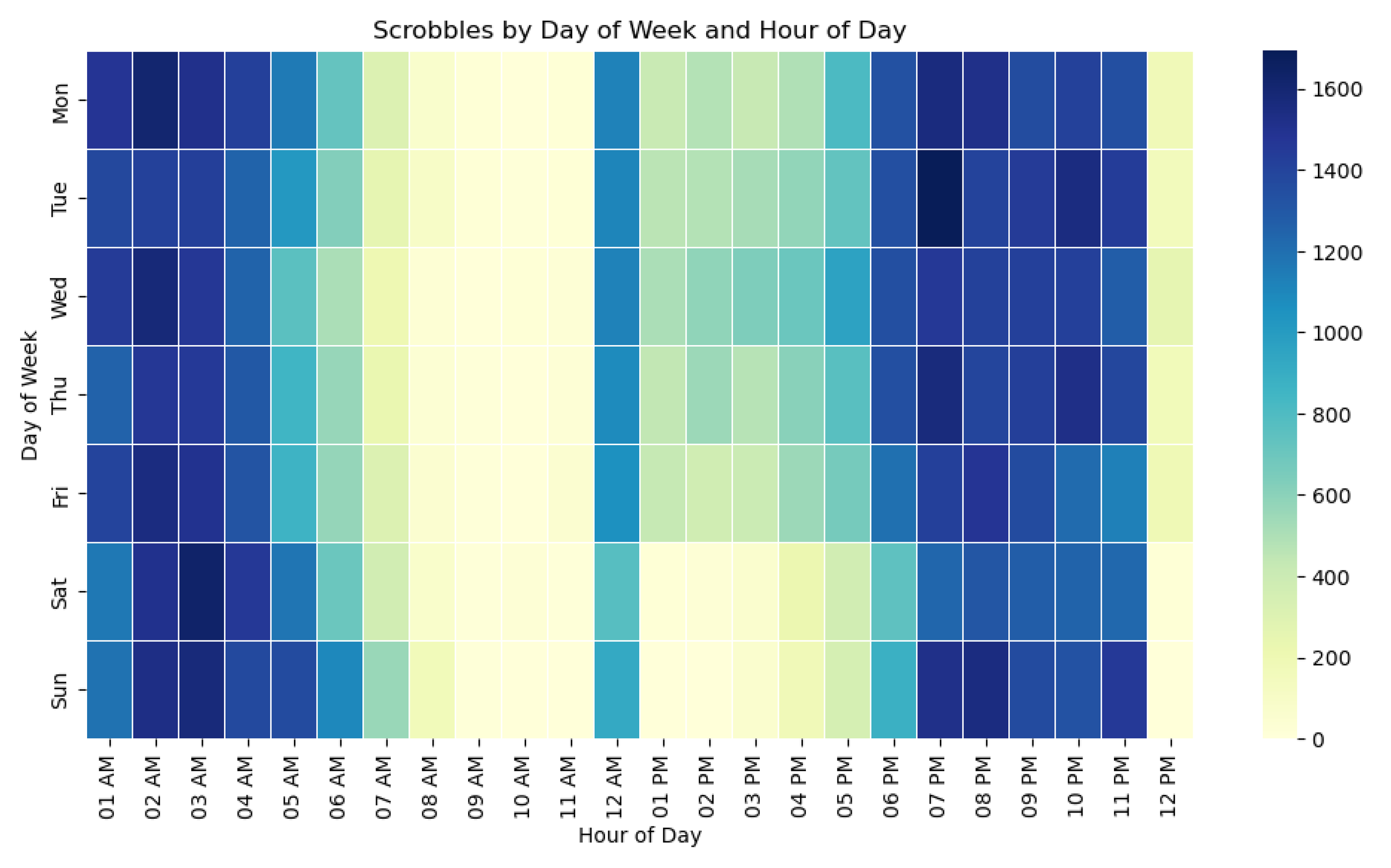
```
           Dijon                              Skin         Skin.1  \
0     Red Velvet  'The ReVe Festival 2022 - Birthday'        BYE BYE
1         Eve Ai                           垂直活著水平留戀著。  Forever Young
2   Zombie Juice               Love Without Conditions            Fly
3     Charli XCX                    how i'm feeling now        anthems
4  Genesis Owusu                 Smiling with No Teeth     Centrefold

   17 Jul 2023 19:17
0  17 Jul 2023 19:13
1  17 Jul 2023 19:07
2  17 Jul 2023 19:04
3  17 Jul 2023 19:02
4  17 Jul 2023 18:57
```

In [9]:
```python
# Find the top artists for user
top_artists_user = df['artist'].value_counts().reset_index()
top_artists_user.columns = ['artist', 'user_count']

# Find the top artists for friend
top_artists_friend = df_friend['Dijon'].value_counts().reset_index()
top_artists_friend.columns = ['artist', 'friend_count']

# Merge the two dataframes on artist
merged_artists = pd.merge(top_artists_user, top_artists_friend, how='inner', on='artist')

# Create a new column 'overlap_count' that is the minimum of 'user_count' and 'friend_count'
merged_artists['overlap_count'] = merged_artists[['user_count', 'friend_count']].min(axis=1)

# Sort the DataFrame by 'overlap_count' in descending order
merged_artists = merged_artists.sort_values('overlap_count', ascending=False)

# Select the top 20 artists with the most overlap
top_overlap = merged_artists.head(20)

# Plot the result
plt.figure(figsize=(10,10))
sns.set_color_codes("pastel")
sns.barplot(x='user_count', y='artist', data=top_overlap, label='User', color='b')
sns.set_color_codes("muted")
sns.barplot(x='friend_count', y='artist', data=top_overlap, label='Friend', color='b')
plt.legend(ncol=2, loc="lower right", frameon=True)
plt.xlabel('Scrobbling')
plt.ylabel('Artist')
sns.despine(left=True, bottom=True)
plt.show()
```

```
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12450 (\N{KATAKANA LETTER A}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12488 (\N{KATAKANA LETTER TO}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12521 (\N{KATAKANA LETTER RA}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12473 (\N{KATAKANA LETTER SU}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12469 (\N{KATAKANA LETTER SA}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12454 (\N{KATAKANA LETTER U}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12531 (\N{KATAKANA LETTER N}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12489 (\N{KATAKANA LETTER DO}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12481 (\N{KATAKANA LETTER TI}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12540 (\N{KATAKANA-HIRAGANA PROLONGED SOUND MARK}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
E:\pythonanoconda\Lib\site-packages\IPython\core\pylabtools.py:152: UserWarning: Glyph 12512 (\N{KATAKANA LETTER MU}) missing from current font.
  fig.canvas.print_figure(bytes_io, **kw)
```

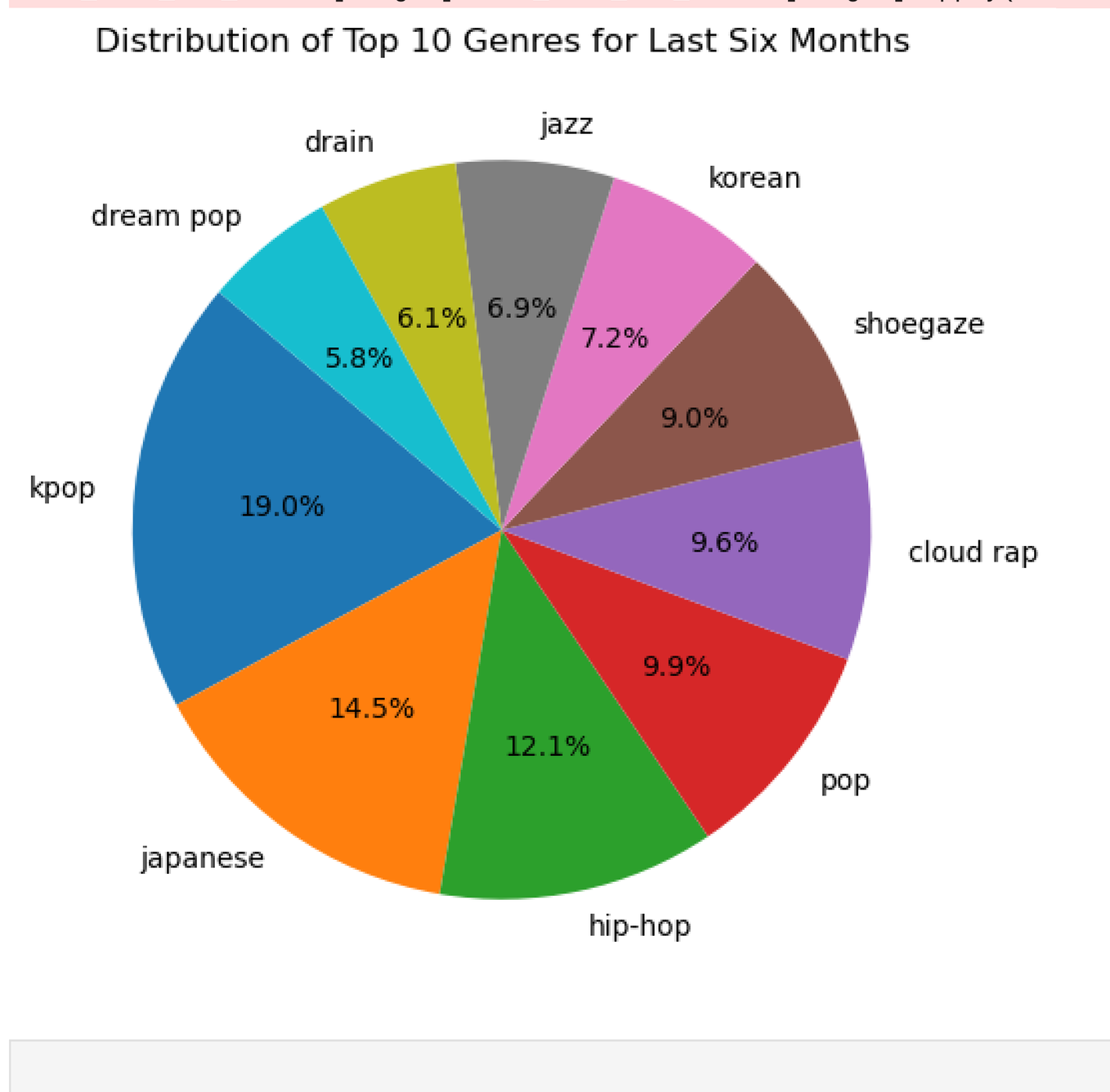


In [7]:
```python
# Define the order of the hours in a day
hour_order = [f"{h:02d} {ap}" for ap in ['AM', 'PM'] for h in list(range(1, 13))]

# Reindex the columns of the pivot table according to the hour_order
pivot = pivot.reindex(columns=hour_order)

# Create the heatmap
plt.figure(figsize=(12, 6))
sns.heatmap(pivot, cmap='YlGnBu', linewidths=0.5)
plt.title('Scrobbles by Day of Week and Hour of Day')
plt.xlabel('Hour of Day')
plt.ylabel('Day of Week')
plt.show()
```



In [18]:
```python
# Get the current date
now = datetime.now()

# Convert 'date' back to datetime format for filtering
df_reset['date'] = pd.to_datetime(df_reset['date'])

# Filter the data for the last six months
df_last_six_months = df_reset[df_reset['date'] > now - pd.DateOffset(months=6)]

# Apply the genre standardization function to the 'tags' column
df_last_six_months['tags'] = df_last_six_months['tags'].apply(standardize_genres)

# Split the 'tags' column into separate genres and stack them into a single column
genres = df_last_six_months['tags'].str.split(', ', expand=True).stack()

# Count the number of occurrences of each genre
genre_counts = genres.value_counts()

# Select the top 10 genres
top_genres = genre_counts.head(10)

# Calculate the proportion of each genre
genre_proportions = top_genres / top_genres.sum()

# Create a pie chart of the top 10 genres
plt.figure(figsize=(10, 6))
plt.pie(genre_proportions, labels=genre_proportions.index, autopct='%1.1f%%', startangle=140)
plt.title('Distribution of Top 10 Genres for Last Six Months')
plt.show()
```

```
C:\Users\Dhruv\AppData\Local\Temp\ipykernel_19092\1256002344.py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_last_six_months['tags'] = df_last_six_months['tags'].apply(standardize_genres)
```



In [ ]: