🚀 Viral Social Media Trends & Engagement Analysis

About Dataset

This dataset captures the pulse of viral social media trends across TikTok, Instagram, Twitter, and YouTube. It provides insights into the most popular hashtags, content types, and user engagement levels, offering a comprehensive view of how trends unfold across platforms. With regional data and influencer-driven content, this dataset is perfect for:

- Trend analysis 🔍
- Sentiment modeling 💭
- Understanding influencer marketing 📈

Dive in to explore what makes content go viral, the behaviors that drive engagement, and how trends evolve on a global scale! 🌍

Import Library

In [3]:
```
import pandas as pd
```

In [4]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
```

Uploading Csv fle

In [5]:
```
df = pd.read_csv(r"C:\Users\Syed Arif\OneDrive\Desktop\Viral Social Media Trends\Viral_Social_Media_Trends.csv")
```

Data Preprocessing

head()

head is used show to the By default = 5 rows in the dataset

In [6]:
```
df.head()
```

Out[6]:

	Post_ID	Platform	Hashtag	Content_Type	Region	Views	Likes	Shares	Comments
0	Post_1	TikTok	#Challenge	Video	UK	4163464	339431	53135	19346
1	Post_2	Instagram	#Education	Shorts	India	4155940	215240	65860	27239
2	Post_3	Twitter	#Challenge	Video	Brazil	3666211	327143	39423	36223
3	Post_4	YouTube	#Education	Shorts	Australia	917951	127125	11687	36806
4	Post_5	TikTok	#Dance	Post	Brazil	64866	171361	69581	6376

.tail()

tail is used to show rows by Descending order

In [7]:
```
df.tail()
```

Out[7]:

	Post_ID	Platform	Hashtag	Content_Type	Region	Views	Likes	Shares	Comme
4995	Post_4996	YouTube	#Education	Video	Japan	4258236	381509	59209	46
4996	Post_4997	Instagram	#Education	Video	UK	4841525	166266	54841	48
4997	Post_4998	YouTube	#Fashion	Shorts	USA	2004080	32401	75229	338
4998	Post_4999	Instagram	#Gaming	Shorts	Canada	2221665	46996	12954	428
4999	Post_5000	Instagram	#Fashion	Reel	Japan	2224007	72744	29945	463

shape

It show the total no of rows & Column in the dataset

In [8]:
```
df.shape
```

Out[8]: (5000, 10)

Columns

It show the no of each Column

In [9]:
```
df.columns
```

Out[9]:
```
Index(['Post_ID', 'Platform', 'Hashtag', 'Content_Type', 'Region', 'Views',
       'Likes', 'Shares', 'Comments', 'Engagement_Level'],
      dtype='object')
```

.dtypes

This Attribute show the data type of each column

In [10]:
```
df.dtypes
```

Out[10]:
```
Post_ID            object
Platform           object
Hashtag            object
Content_Type       object
Region             object
Views               int64
Likes               int64
Shares              int64
Comments            int64
Engagement_Level   object
dtype: object
```

.unique()

In a column, It show the unique value of specific column.

In [12]:
```
df["Content_Type"].unique()
```

Out[12]:
```
array(['Video', 'Shorts', 'Post', 'Tweet', 'Live Stream', 'Reel'],
      dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

In [13]:
```
df.nunique()
```

Out[13]:
```
Post_ID            5000
Platform              4
Hashtag              10
Content_Type          6
Region                8
Views              4996
Likes              4972
Shares             4877
Comments           4735
Engagement_Level      3
dtype: int64
```

.describe()

It show the Count, mean , median etc

In [14]:
```
df.describe()
```

Out[14]:

	Views	Likes	Shares	Comments
count	5.000000e+03	5000.000000	5000.000000	5000.000000
mean	2.494066e+06	251475.029800	50519.562000	24888.393800
std	1.459490e+06	144349.583384	29066.362671	14284.504319
min	1.266000e+03	490.000000	52.000000	18.000000
25%	1.186207e+06	126892.250000	25029.000000	12305.250000
50%	2.497373e+06	249443.000000	50839.500000	25004.000000
75%	3.759781e+06	373970.750000	75774.250000	37072.750000
max	4.999430e+06	499922.000000	99978.000000	49993.000000

.value_counts

It Shows all the unique values with their count

In [15]:
```
df["Content_Type"].value_counts()
```

Out[15]:
```
Content_Type
Live Stream    855
Post           853
Reel           841
Tweet          836
Video          828
Shorts         787
Name: count, dtype: int64
```

isnull()

It shows the how many null values

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

Out[16]:

	Post_ID	Platform	Hashtag	Content_Type	Region	Views	Likes	Shares	Comments	Enga
0	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	
4995	False	False	False	False	False	False	False	False	False	
4996	False	False	False	False	False	False	False	False	False	
4997	False	False	False	False	False	False	False	False	False	
4998	False	False	False	False	False	False	False	False	False	
4999	False	False	False	False	False	False	False	False	False	

5000 rows × 10 columns

.info()

To Show Data type of each column

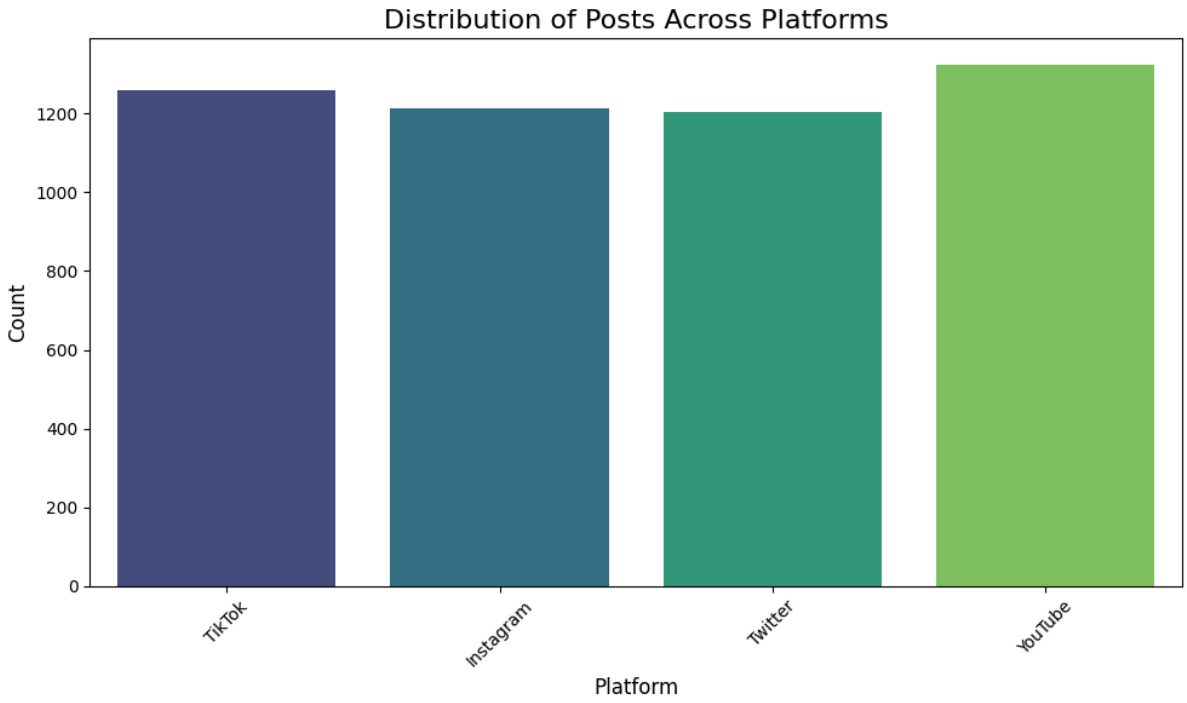
In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Post_ID          5000 non-null   object
 1   Platform         5000 non-null   object
 2   Hashtag          5000 non-null   object
 3   Content_Type     5000 non-null   object
 4   Region           5000 non-null   object
 5   Views            5000 non-null   int64 
 6   Likes            5000 non-null   int64 
 7   Shares           5000 non-null   int64 
 8   Comments         5000 non-null   int64 
 9   Engagement_Level 5000 non-null   object
dtypes: int64(4), object(6)
memory usage: 390.8+ KB
```

Distribution of Platforms

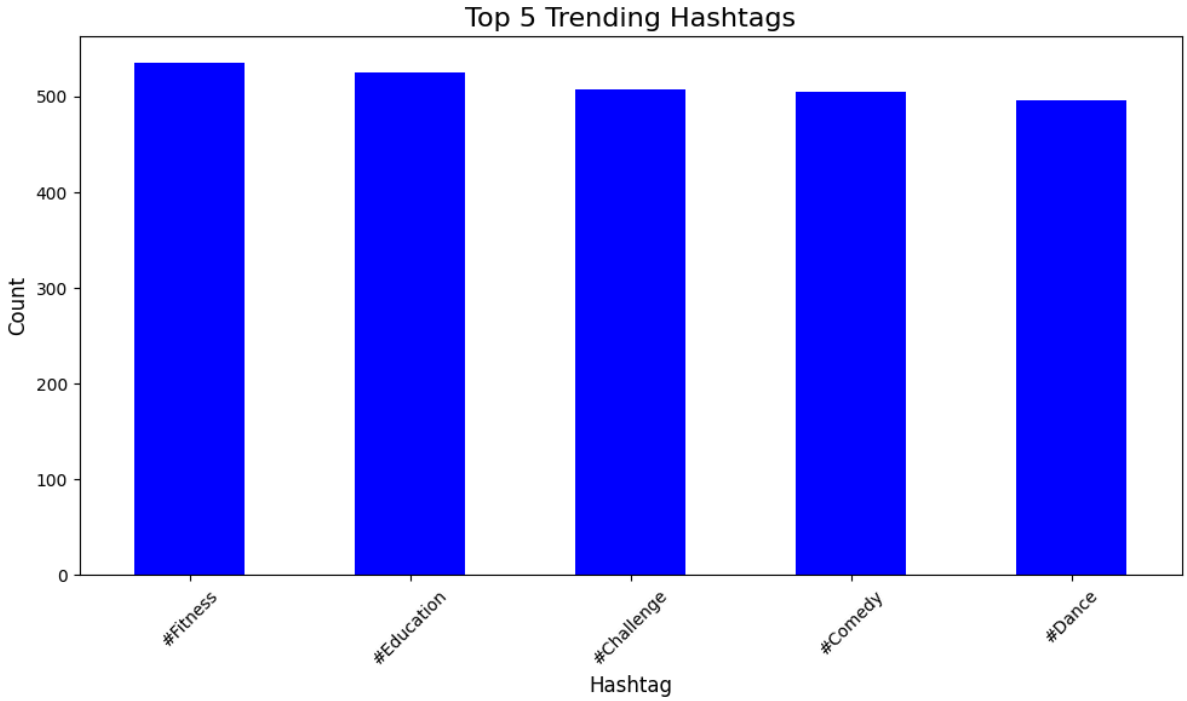
In [18]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x="Platform", palette="viridis")
plt.title("Distribution of Posts Across Platforms", fontsize=16)
plt.xlabel("Platform", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_21040\2083765734.py:2: Future
Warning:

Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.

  sns.countplot(data=df, x="Platform", palette="viridis")
```



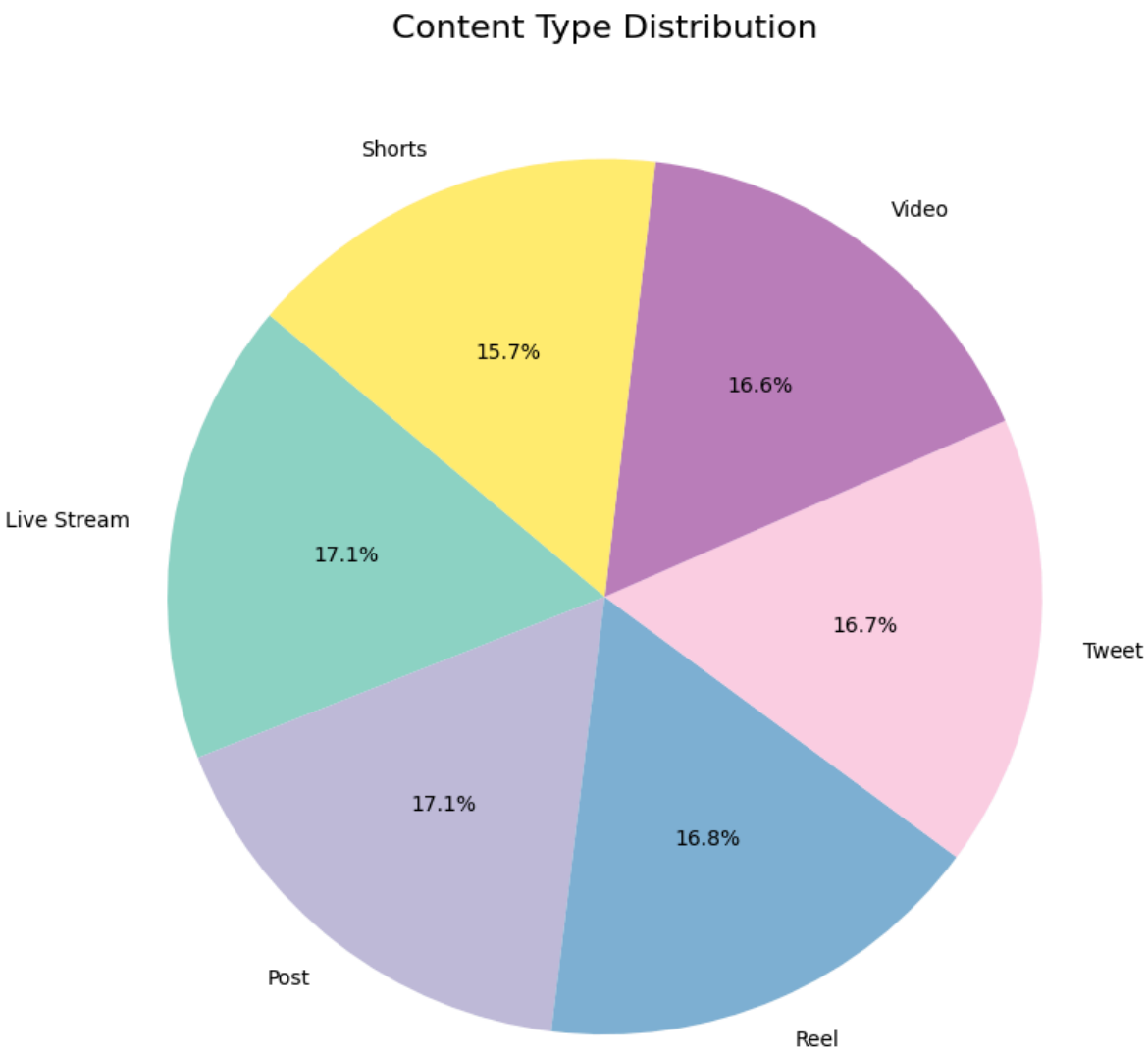
Top 5 Trending Hashtags

In [28]:
```
top_hashtags = df["Hashtag"].value_counts().head(5)
plt.figure(figsize=(10, 6))
top_hashtags.plot(kind="bar", color="blue")
plt.title("Top 5 Trending Hashtags", fontsize=16)
plt.xlabel("Hashtag", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



Content Type Distribution

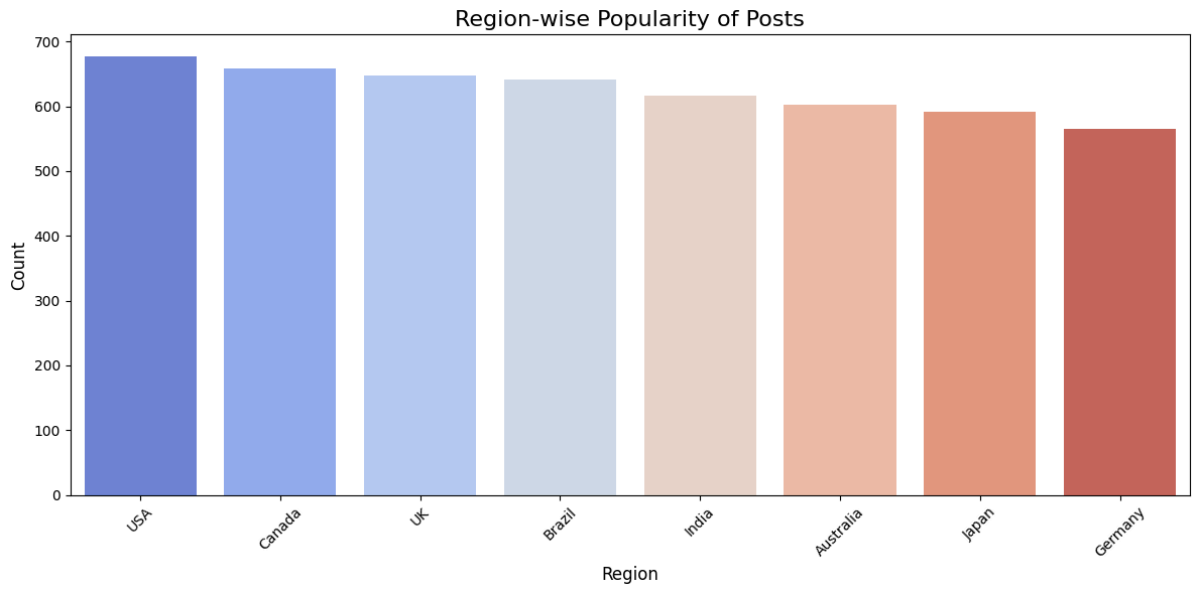
In [29]:
```
content_type_counts = df["Content_Type"].value_counts()
plt.figure(figsize=(8, 8))
content_type_counts.plot(kind="pie", autopct="%1.1f%%", startangle=140, cmap
="Set3")
plt.title("Content Type Distribution", fontsize=16)
plt.ylabel("")
plt.tight_layout()
plt.show()
```



Region-wise Popularity of Posts

In [30]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Region", order=df["Region"].value_counts().index, pa
lette="coolwarm")
plt.title("Region-wise Popularity of Posts", fontsize=16)
plt.xlabel("Region", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

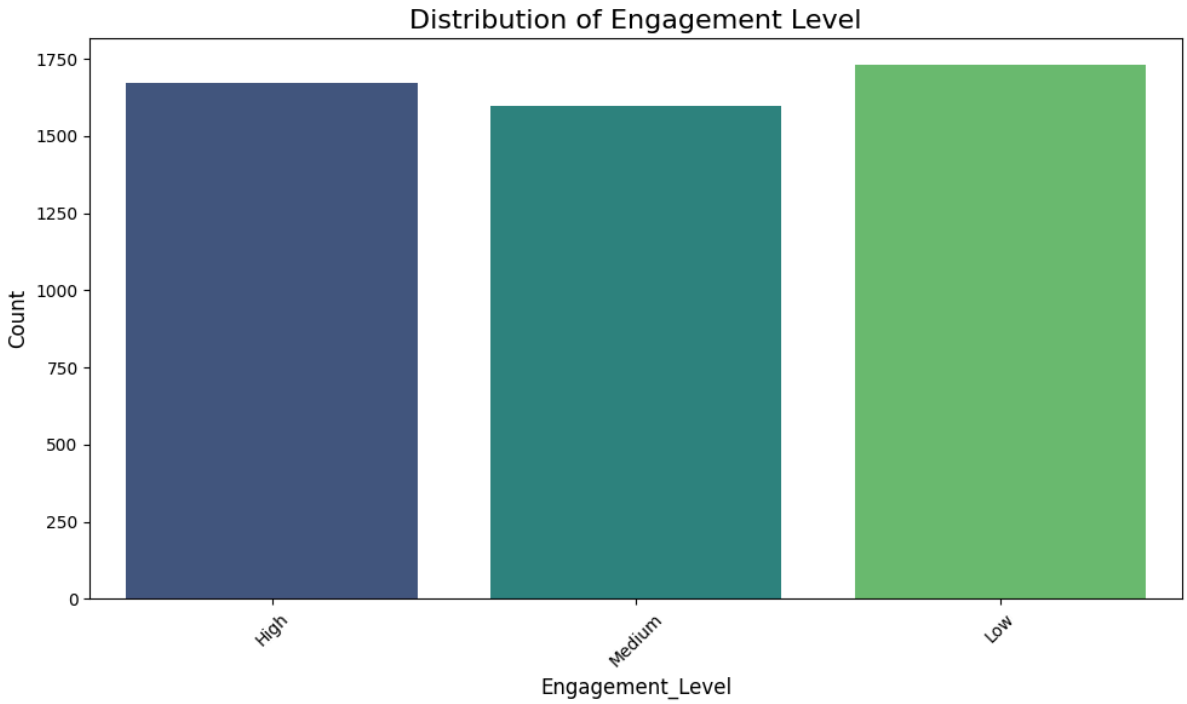
```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_21040\4284369253.py:2: Future
Warning:

Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.

  sns.countplot(data=df, x="Region", order=df["Region"].value_counts().index,
palette="coolwarm")
```



Engagement Level Distribution

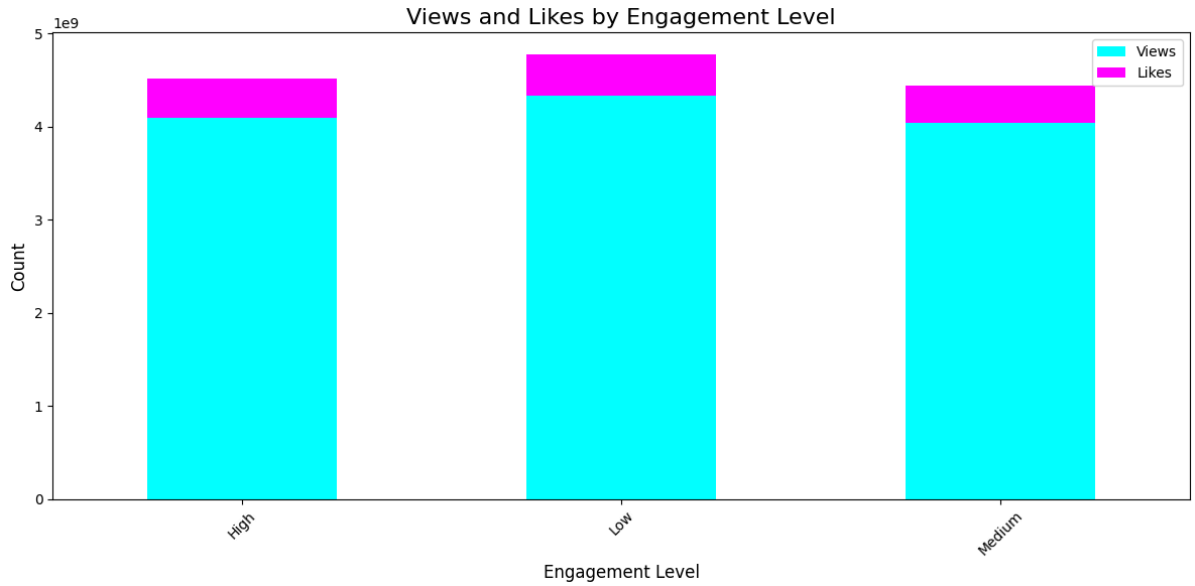
In [36]:
```
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x="Engagement_Level", palette="viridis")
plt.title("Distribution of Engagement Level", fontsize=16)
plt.xlabel("Engagement_Level", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_21040\2727971578.py:2: Future
Warning:

Passing `palette` without assigning `hue` is deprecated and will be removed i
n v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the sa
me effect.

  sns.countplot(data=df, x="Engagement_Level", palette="viridis")
```



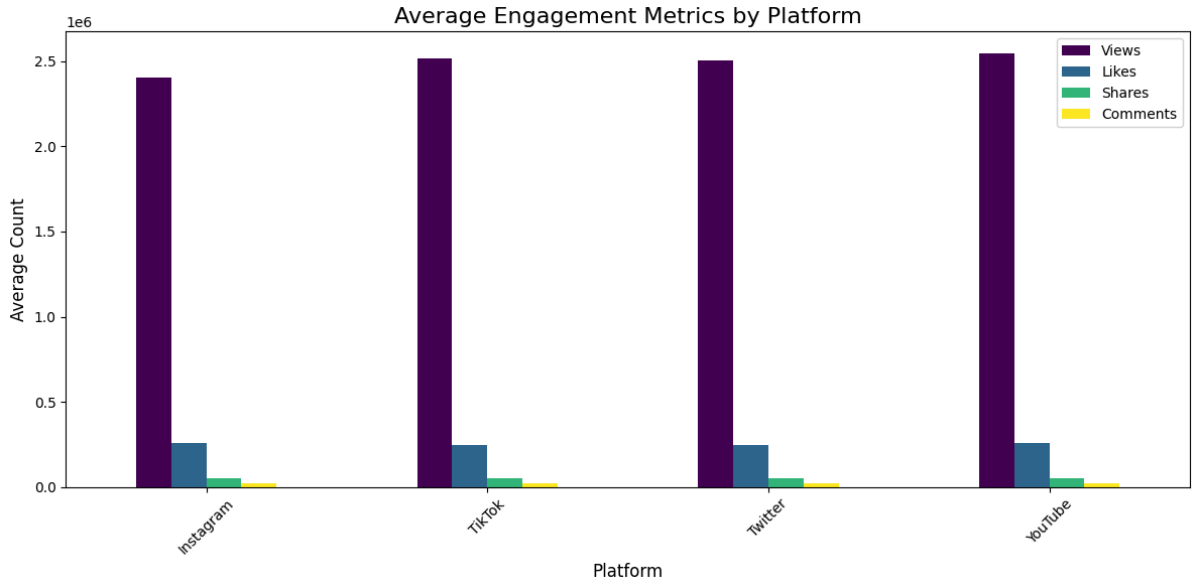
Views vs Likes

In [43]:
```
plt.figure(figsize=(12, 6))
df_grouped = df.groupby('Engagement_Level')[['Views', 'Likes']].sum()
df_grouped.plot(kind="bar", stacked=True, colormap="cool", ax=plt.gca())

plt.title("Views and Likes by Engagement Level", fontsize=16)
plt.xlabel("Engagement Level", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



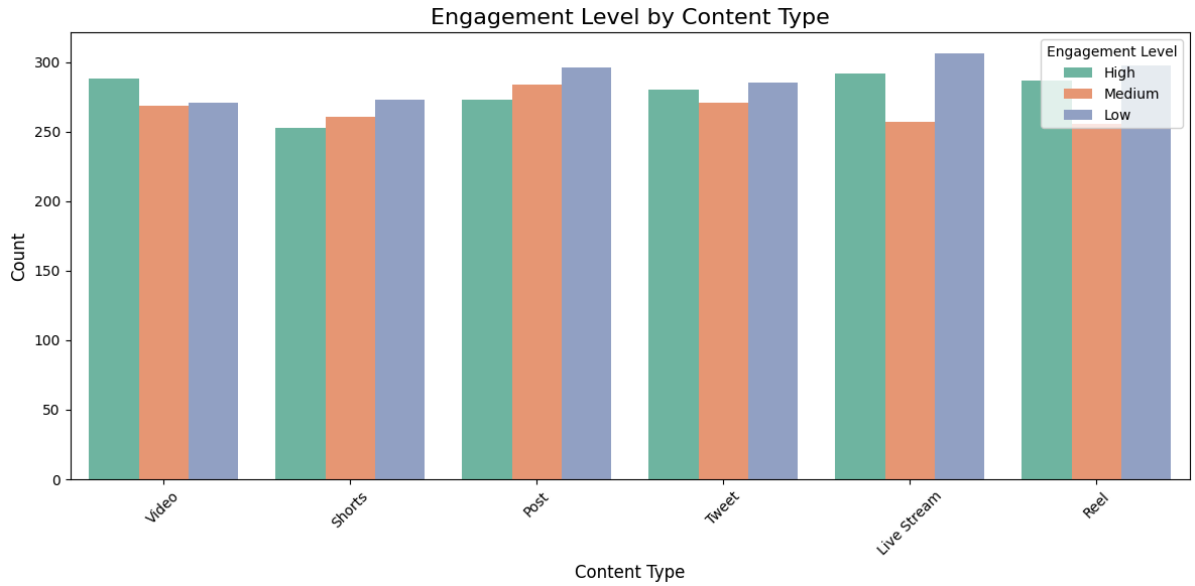
Average Engagement Metrics by Platform

In [44]:
```
platform_engagement = df.groupby("Platform")[["Views", "Likes", "Shares", "Com
ments"]].mean()
platform_engagement.plot(kind="bar", figsize=(12, 6), colormap="viridis")
plt.title("Average Engagement Metrics by Platform", fontsize=16)
plt.xlabel("Platform", fontsize=12)
plt.ylabel("Average Count", fontsize=12)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



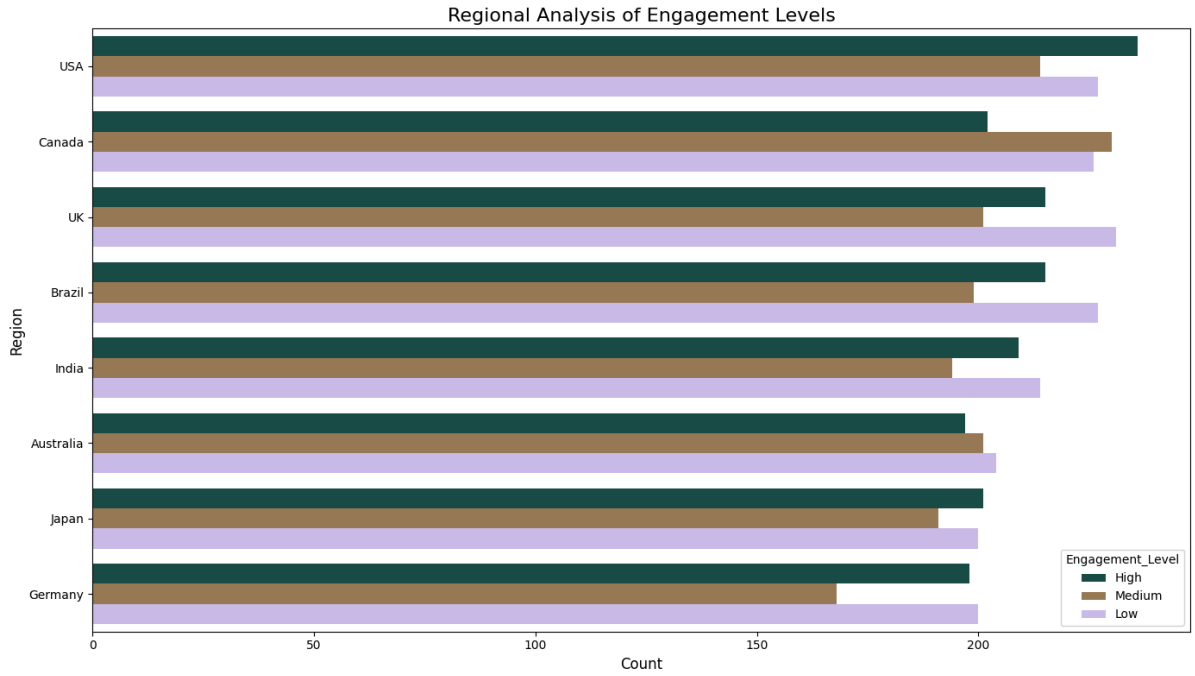
Engagement Level by Content Type

In [45]:
```
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x="Content_Type", hue="Engagement_Level", palette="Set2")
plt.title("Engagement Level by Content Type", fontsize=16)
plt.xlabel("Content Type", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.legend(title="Engagement Level", loc="upper right")
plt.tight_layout()
plt.show()
```



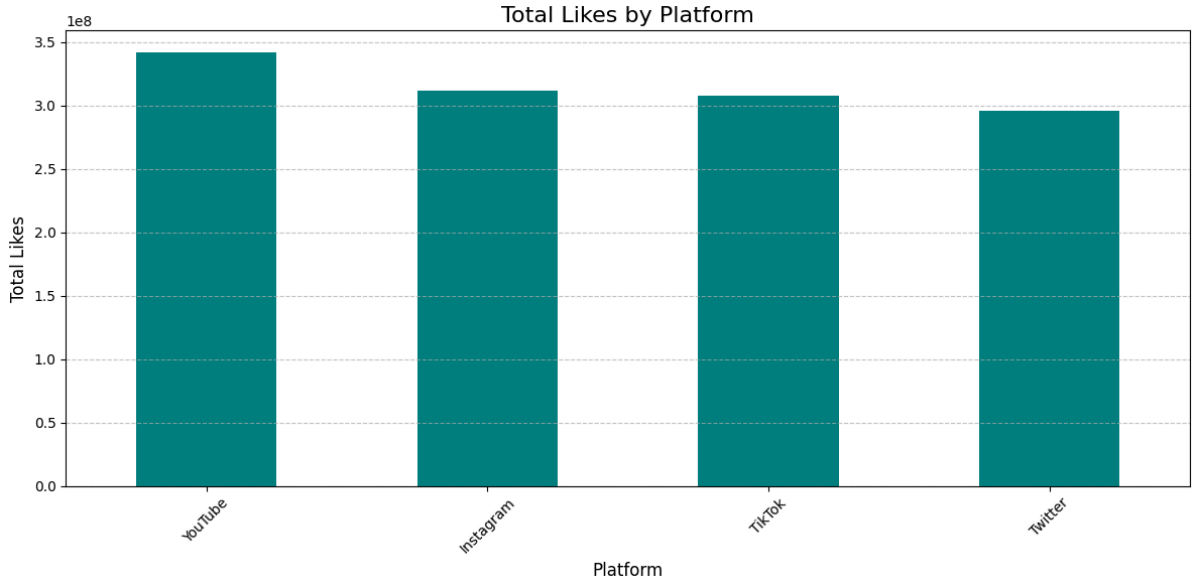
Regional Analysis of Engagement Levels

In [46]:
```
plt.figure(figsize=(14, 8))
sns.countplot(data=df, y="Region", hue="Engagement_Level", palette="cubehelix", order=df["Region"].value_counts().index)
plt.title("Regional Analysis of Engagement Levels", fontsize=16)
plt.xlabel("Count", fontsize=12)
plt.ylabel("Region", fontsize=12)
plt.tight_layout()
plt.show()
```



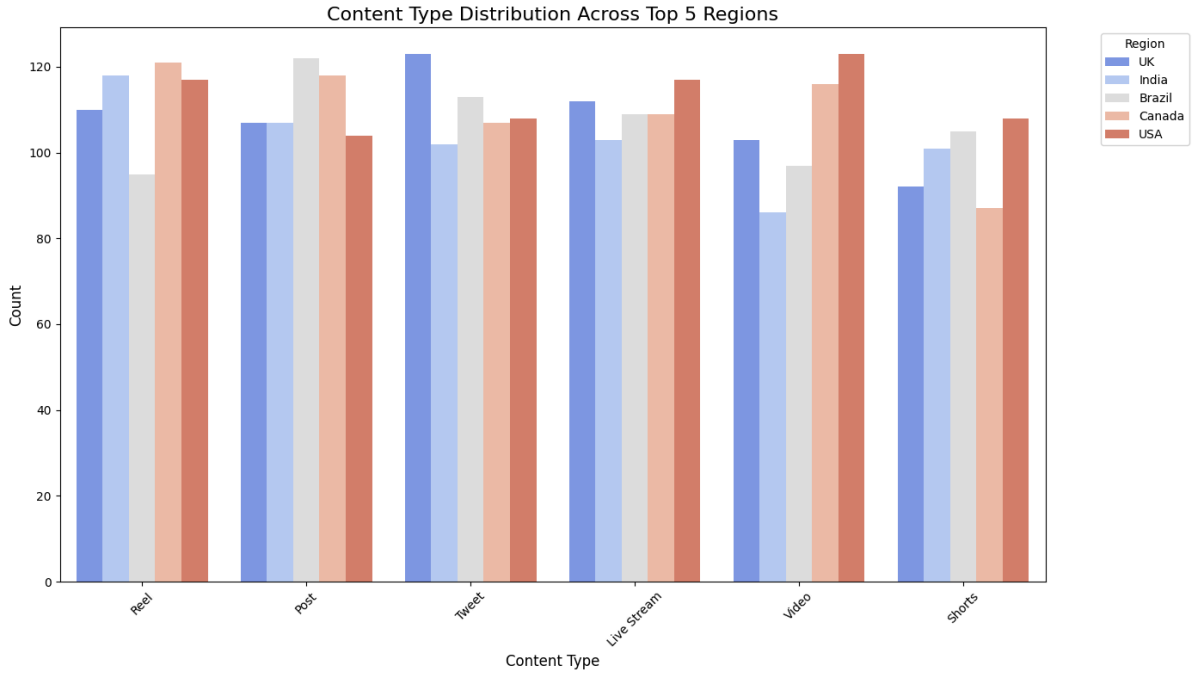
Total Likes by Platform

In [48]:
```
platform_likes = df.groupby("Platform")["Likes"].sum().sort_values(ascending=F
alse)
plt.figure(figsize=(12, 6))
platform_likes.plot(kind="bar", color="teal")
plt.title("Total Likes by Platform", fontsize=16)
plt.xlabel("Platform", fontsize=12)
plt.ylabel("Total Likes", fontsize=12)
plt.xticks(rotation=45)
plt.grid(axis="y", linestyle="--", alpha=0.7)
plt.tight_layout()
plt.show()
```



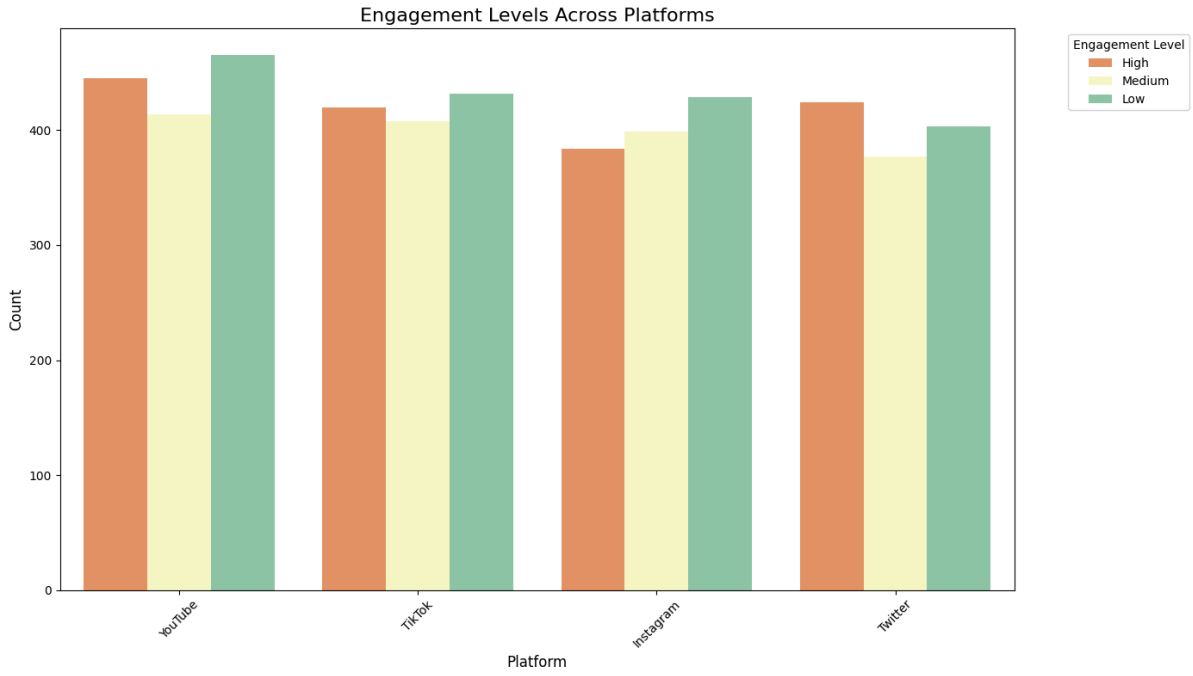
Content Type Distribution by Region

In [50]:
```
# Get the top 5 regions by count
top_regions = df["Region"].value_counts().head(5).index

# Filter the DataFrame to only include these top 5 regions
df_top_regions = df[df["Region"].isin(top_regions)]

# Plotting the count plot for the top 5 regions
plt.figure(figsize=(14, 8))
sns.countplot(
    data=df_top_regions,
    x="Content_Type",
    hue="Region",
    palette="coolwarm",
    order=df_top_regions["Content_Type"].value_counts().index,
)

# Customizing the plot
plt.title("Content Type Distribution Across Top 5 Regions", fontsize=16)
plt.xlabel("Content Type", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)
plt.legend(title="Region", bbox_to_anchor=(1.05, 1), loc="upper left")
plt.tight_layout()
plt.show()
```



Engagement Levels Across Platforms

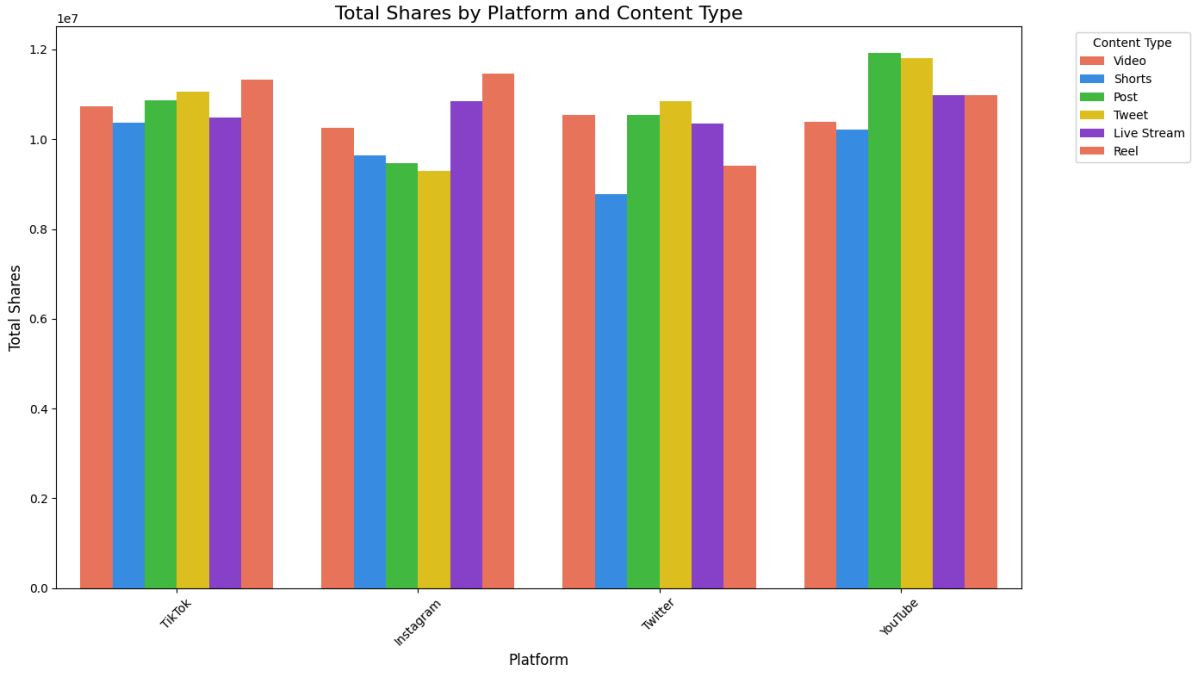
In [52]:
```
plt.figure(figsize=(14, 8))
sns.countplot(
    data=df,
    x="Platform",  # Change to 'Platform' on x-axis
    hue="Engagement_Level",  # Use 'Engagement_Level' for hue
    palette="Spectral",
    order=df["Platform"].value_counts().index,  # Optionally order by count of platforms
)

plt.title("Engagement Levels Across Platforms", fontsize=16)
plt.xlabel("Platform", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.xticks(rotation=45)  # Adjust rotation if necessary
plt.legend(title="Engagement Level", bbox_to_anchor=(1.05, 1), loc="upper left")
plt.tight_layout()
plt.show()
```



Total Shares by Platform and Content Type

In [54]:
```
# Define a custom palette for Content_Type
content_type_colors = ['#FF6347', '#1E90FF', '#32CD32', '#FFD700', '#8A2BE2']
# List of colors (you can adjust this)

plt.figure(figsize=(14, 8))
sns.barplot(
    data=df,
    x="Platform",
    y="Shares",
    hue="Content_Type",
    ci=None,
    palette=content_type_colors,  # Use custom color palette
    estimator=sum,
)

plt.title("Total Shares by Platform and Content Type", fontsize=16)
plt.xlabel("Platform", fontsize=12)
plt.ylabel("Total Shares", fontsize=12)
plt.xticks(rotation=45)
plt.legend(title="Content Type", bbox_to_anchor=(1.05, 1), loc="upper left")
plt.tight_layout()
plt.show()
```

```
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_21040\2914511461.py:5: Future
Warning:

The `ci` parameter is deprecated. Use `errorbar=None` for the same effect.

  sns.barplot(
C:\Users\Syed Arif\AppData\Local\Temp\ipykernel_21040\2914511461.py:5: UserWa
rning:
The palette list has fewer values (5) than needed (6) and will cycle, which m
ay produce an uninterpretable plot.
  sns.barplot(
```



WordCloud of Channels

In [47]:
```
# Import necessary libraries
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Generate a word cloud based on the frequency of channel names
wordcloud = WordCloud(width=1000, height=600, background_color='white', colorm
ap='viridis').generate_from_frequencies(df['Platform'].value_counts())

# Plot the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.title('Word Cloud of Platform', fontsize=16)
plt.axis('off')  # Disable the axis for better visualization
plt.show()
```

In [ ]: