Task 1 - Youtubers Streamer Analysis

In [2]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
df=pd.read_csv("D:\INTERNSHIPS\InternCareer\youtubers_df.csv")
```

```
<>:1: SyntaxWarning: invalid escape sequence '\I'
<>:1: SyntaxWarning: invalid escape sequence '\I'
C:\Users\lesego\AppData\Local\Temp\ipykernel_13536\929674512.py:1: SyntaxWarning:
invalid escape sequence '\I'
  df=pd.read_csv("D:\INTERNSHIPS\InternCareer\youtubers_df.csv")
```

In [4]:
```
df.head()
```

Out[4]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	C
0	1	tseries	Música y baile	249500000.0	India	86200.0	2700.0	
1	2	MrBeast	Videojuegos, Humor	183500000.0	Estados Unidos	117400000.0	5300000.0	
2	3	CoComelon	Educación	165500000.0	Unknown	7000000.0	24700.0	
3	4	SETIndia	NaN	162600000.0	India	15600.0	166.0	
4	5	KidsDianaShow	Animación, Juguetes	113500000.0	Unknown	3900000.0	12400.0	

In [5]:
```
df.tail()
```

Out[5]:

	Rank	Username	Categories	Suscribers	Country	Visits	Likes	C
995	996	hamzymukbang	NaN	11700000.0	Estados Unidos	397400.0	14000.0	
996	997	Adaahqueen	NaN	11700000.0	India	1100000.0	92500.0	
997	998	LittleAngelIndonesia	Música y baile	11700000.0	Unknown	211400.0	745.0	
998	999	PenMultiplex	NaN	11700000.0	India	14000.0	81.0	
999	1000	OneindiaHindi	Noticias y Política	11700000.0	India	2200.0	31.0	

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Rank         1000 non-null   int64  
 1   Username     1000 non-null   object 
 2    Categories  694 non-null    object 
 3    Suscribers  1000 non-null   float64
 4   Country      1000 non-null   object 
 5   Visits       1000 non-null   float64
 6   Likes        1000 non-null   float64
 7   Comments     1000 non-null   float64
 8   Links        1000 non-null   object 
dtypes: float64(4), int64(1), object(4)
memory usage: 70.4+ KB
```

In [8]:
```
# Finding missing values.
df.isna().sum()
```

Out[8]:
```
Rank             0
Username         0
Categories     306
Suscribers       0
Country          0
Visits           0
Likes            0
Comments         0
Links            0
dtype: int64
```

In [9]:
```
# Addressing missing values.
df.fillna('Unknown', inplace=True)
```

In [10]:
```
df.isna().sum()
```

Out[10]:
```
Rank           0
Username       0
Categories     0
Suscribers     0
Country        0
Visits         0
Likes          0
Comments       0
Links          0
dtype: int64
```

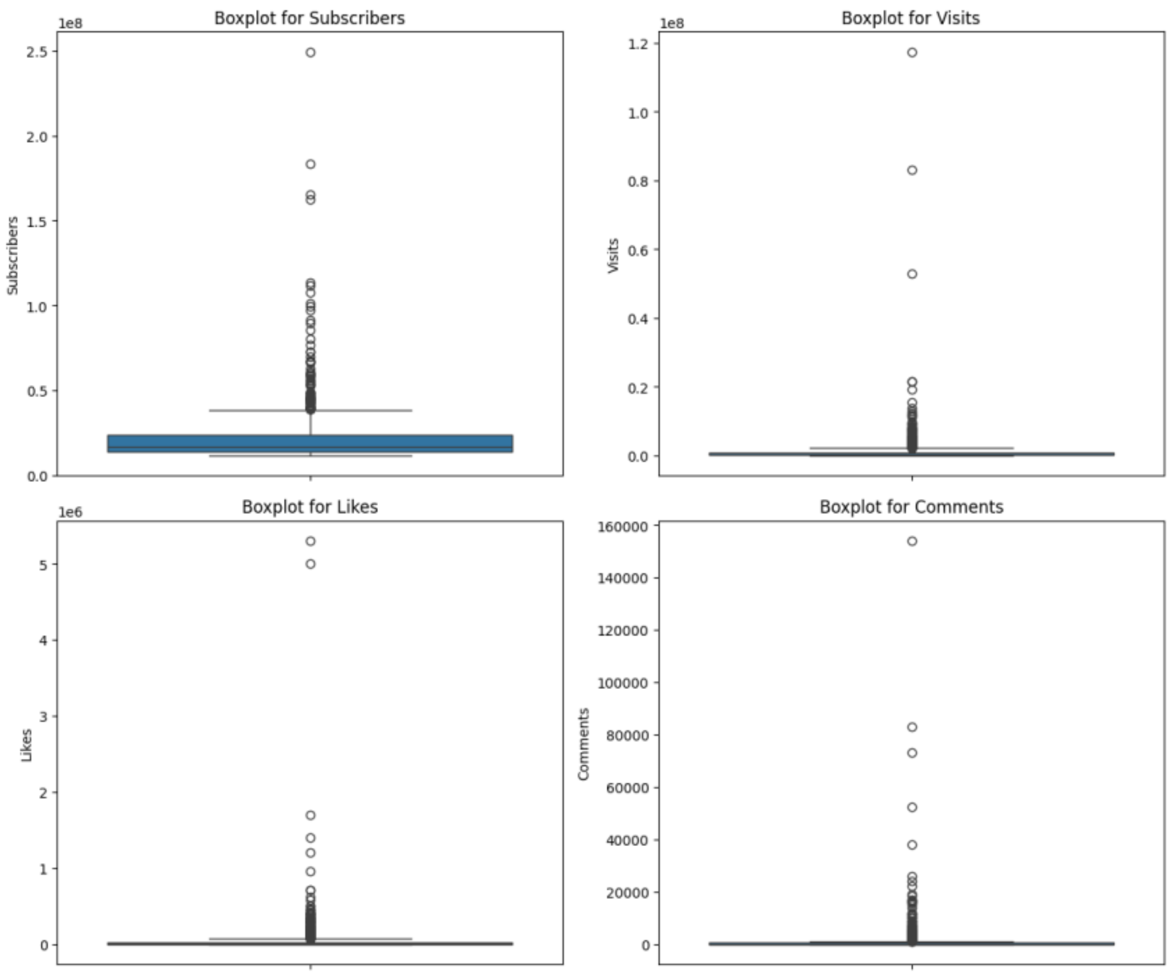
In [11]:
```
df.rename(columns={'Suscribers':'Subscribers'}, inplace=True)
```

In [12]:
```
df.head()
```

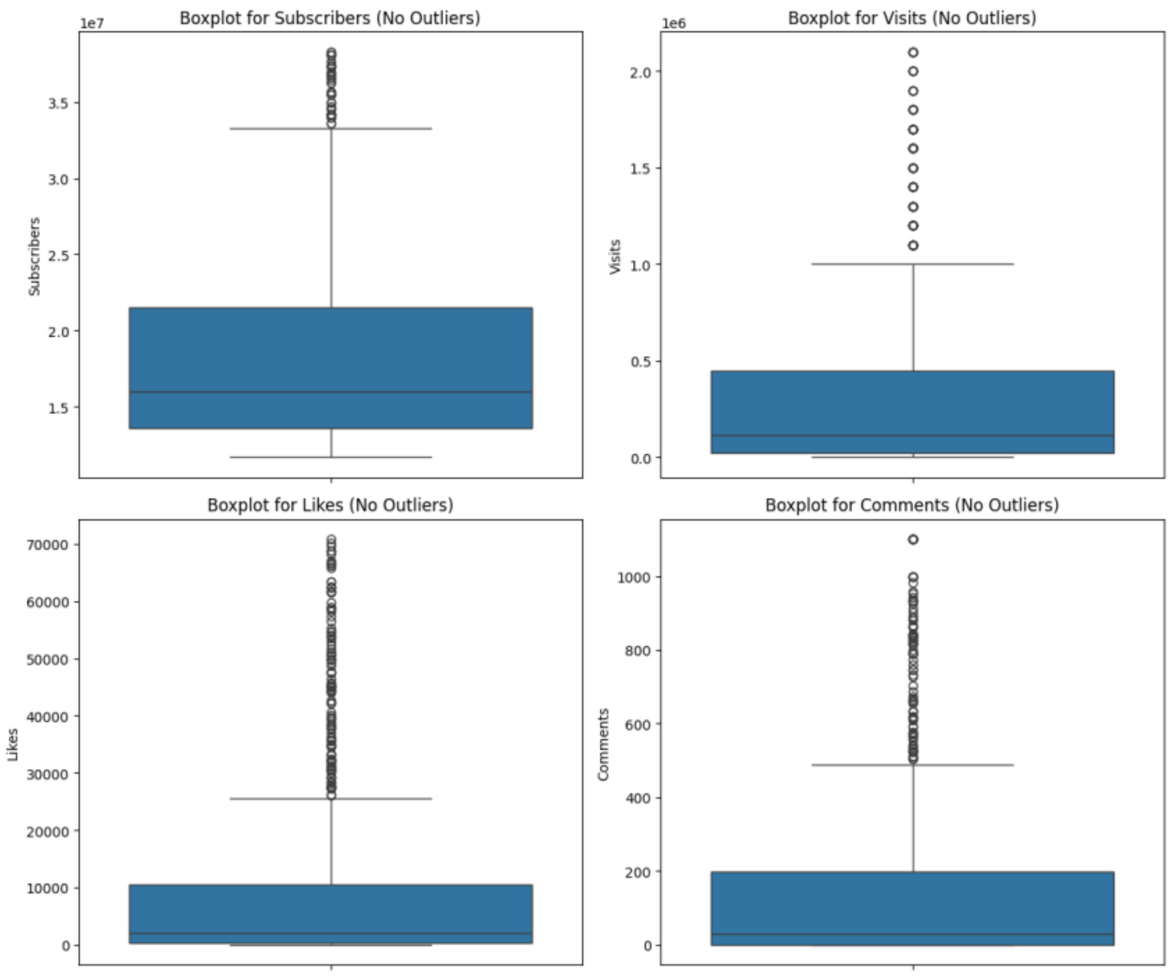
Out[12]:

	Rank	Username	Categories	Subscribers	Country	Visits	Likes	C
0	1	tseries	Música y baile	249500000.0	India	86200.0	2700.0	
1	2	MrBeast	Videojuegos, Humor	183500000.0	Estados Unidos	117400000.0	5300000.0	
2	3	CoComelon	Educación	165500000.0	Unknown	7000000.0	24700.0	
3	4	SETIndia	Unknown	162600000.0	India	15600.0	166.0	
4	5	KidsDianaShow	Animación, Juguetes	113500000.0	Unknown	3900000.0	12400.0	

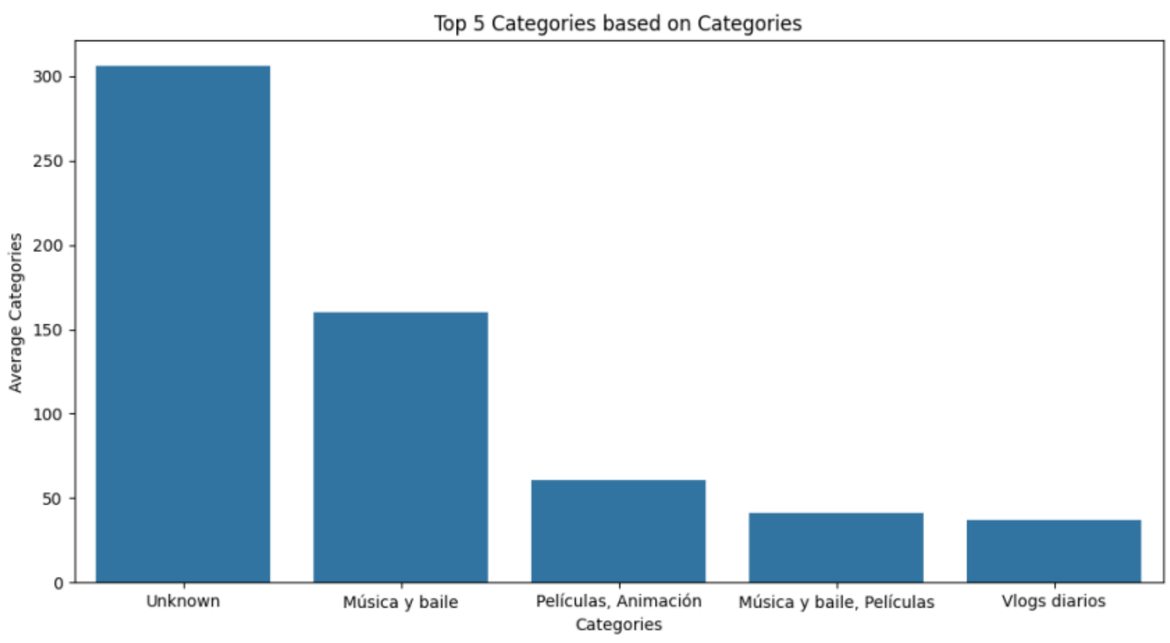
In [13]:
```
# Checking for outliers.
outliers = ['Subscribers','Visits','Likes','Comments']

fig, axes = plt.subplots(2, 2, figsize=(12, 10))

axes = axes.flatten()

for i, col in enumerate (outliers):
    sns.boxplot(y=df[col], ax=axes[i])
    axes[i].set_title(f'Boxplot for {col}')

plt.tight_layout()
plt.show()
```



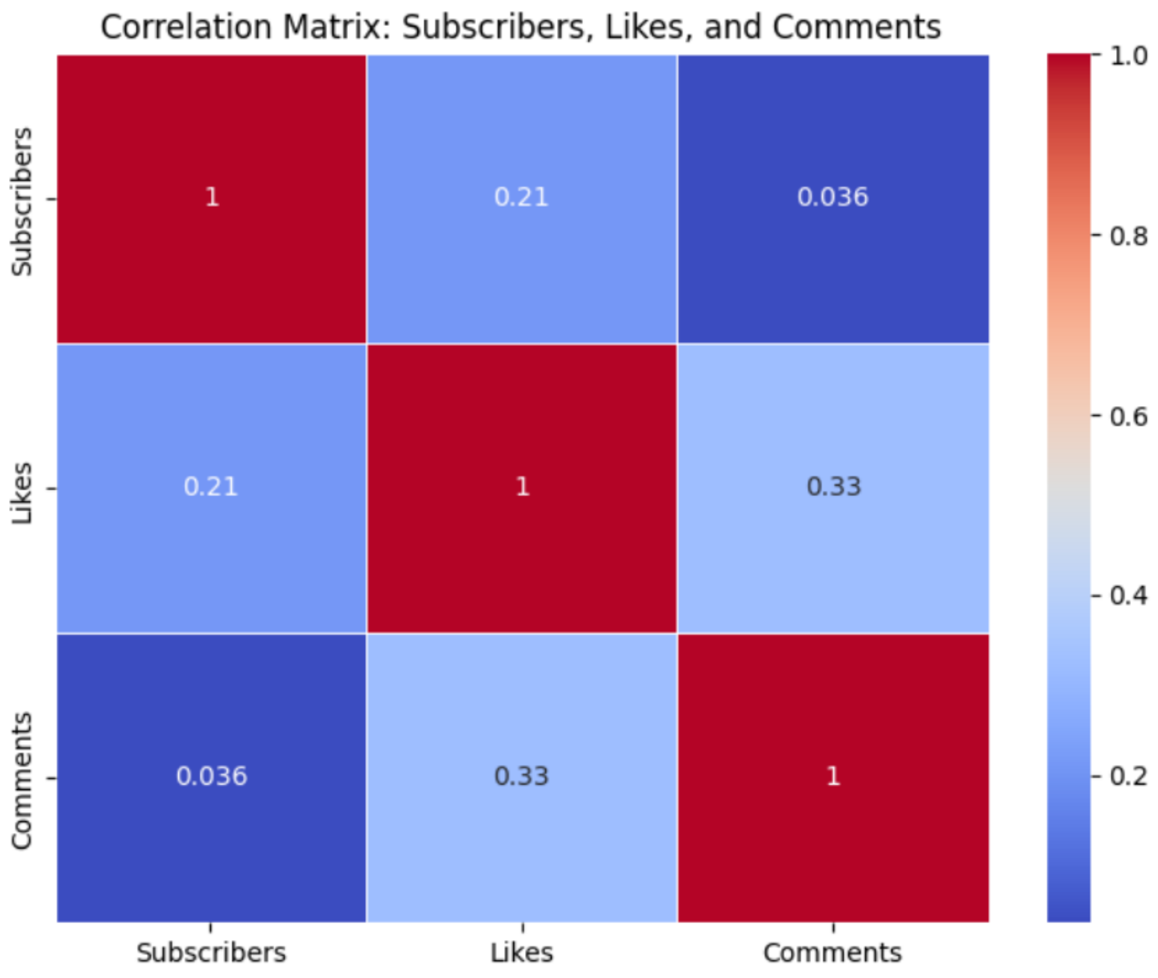
In [14]:
```python
# Removing outliers.
columns_to_check = ['Subscribers', 'Visits', 'Likes', 'Comments']

fig, axes = plt.subplots(2, 2, figsize=(12, 10))

axes = axes.flatten()

for i, col in enumerate(columns_to_check):
    Q1 = df[col].quantile(0.25)
    Q3 = df[col].quantile(0.75)
    IQR =Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    df_no_outliers = df[(df[col] >= lower_bound) & (df[col] <= upper_bound)]
    sns.boxplot(y=df_no_outliers[col], ax=axes[i])
    axes[i].set_title(f'Boxplot for {col} (No Outliers)')

plt.tight_layout()
plt.show()
```



In [22]:
```python
top_5_categories = df['Categories'].value_counts().head(5)
plt.figure(figsize=(12, 6))
sns.barplot(x=top_5_categories.index, y=top_5_categories.values)
plt.title('Top 5 Categories based on Categories')
plt.xlabel('Categories')
plt.ylabel('Average Categories')
plt.show()
```

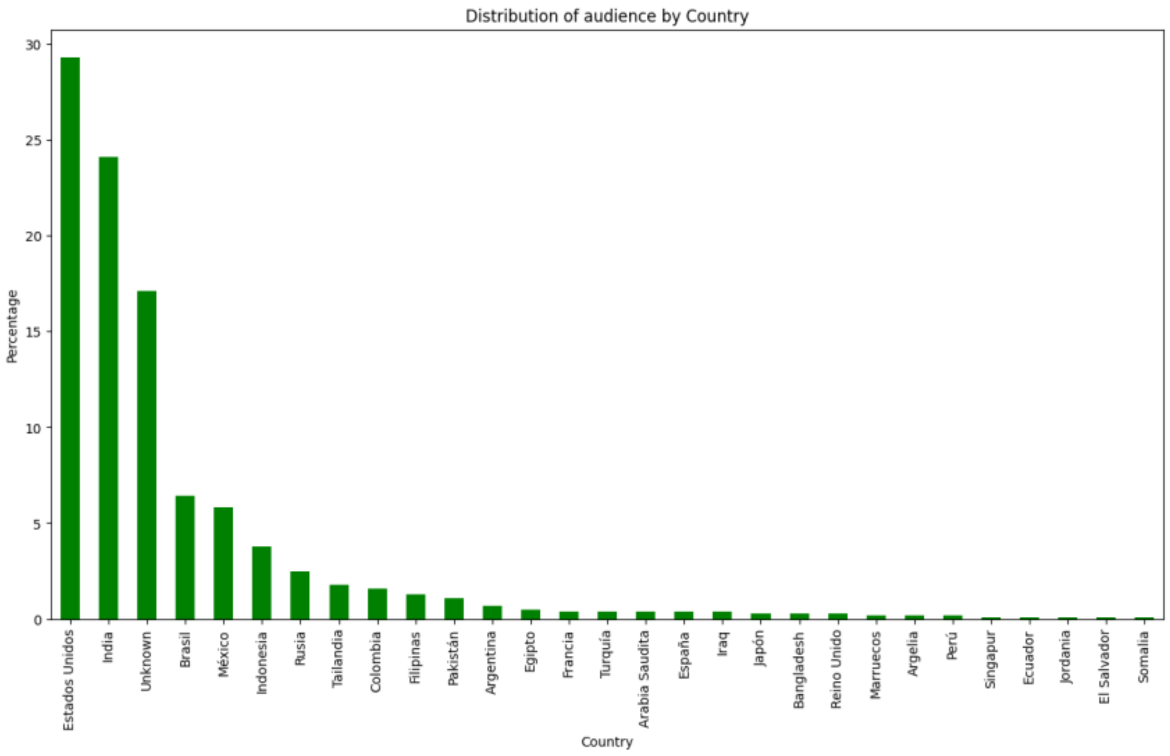


In [23]:
```python
correlation_data = df[['Subscribers', 'Likes', 'Comments']]
correlation_matrix = correlation_data.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix: Subscribers, Likes, and Comments')
plt.show()
```

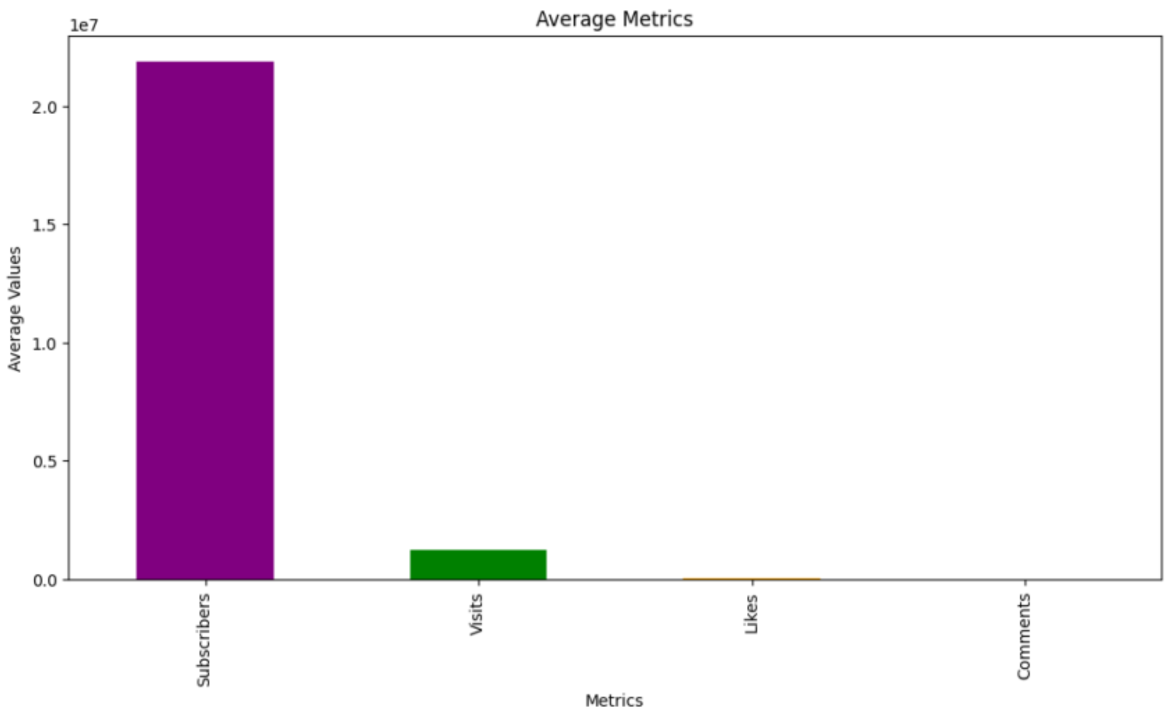


In [31]:
```python
# Audience study
country_distribution = df['Country'].value_counts(normalize=True) *100
```

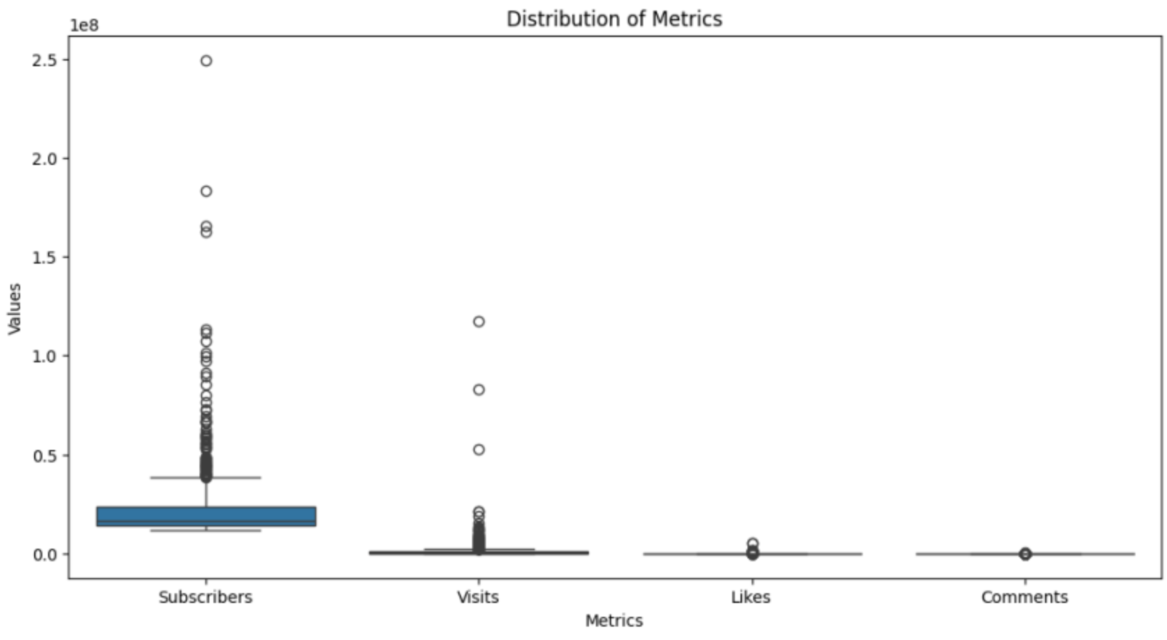
```
plt.figure(figsize=(15, 8))
country_distribution.plot(kind='bar', color='green')
plt.title('Distribution of audience by Country')
plt.xlabel('Country')
plt.ylabel('Percentage')
plt.show()
```



```
In [34]: average_values = df[['Subscribers', 'Visits', 'Likes', 'Comments']].mean()

         plt.figure(figsize=(12, 6))
         average_values.plot(kind='bar', color=['purple', 'green', 'orange', 'skyblue'])
         plt.title('Average Metrics')
         plt.xlabel('Metrics')
         plt.ylabel('Average Values')
         plt.show()
```



In [36]:
```
plt.figure(figsize=(12, 6))
sns.boxplot(data=df[['Subscribers', 'Visits', 'Likes', 'Comments']])
plt.title('Distribution of Metrics')
plt.xlabel('Metrics')
plt.ylabel('Values')
plt.show()
```



In [37]:
```
df.describe()
```

Out[37]:

	Rank	Subscribers	Visits	Likes	Comments
count	1000.000000	1.000000e+03	1.000000e+03	1.000000e+03	1000.000000
mean	500.500000	2.189440e+07	1.209446e+06	5.363259e+04	1288.768000
std	288.819436	1.682775e+07	5.229942e+06	2.580457e+05	6778.188308
min	1.000000	1.170000e+07	0.000000e+00	0.000000e+00	0.000000
25%	250.750000	1.380000e+07	3.197500e+04	4.717500e+02	2.000000
50%	500.500000	1.675000e+07	1.744500e+05	3.500000e+03	67.000000
75%	750.250000	2.370000e+07	8.654750e+05	2.865000e+04	472.000000
max	1000.000000	2.495000e+08	1.174000e+08	5.300000e+06	154000.000000

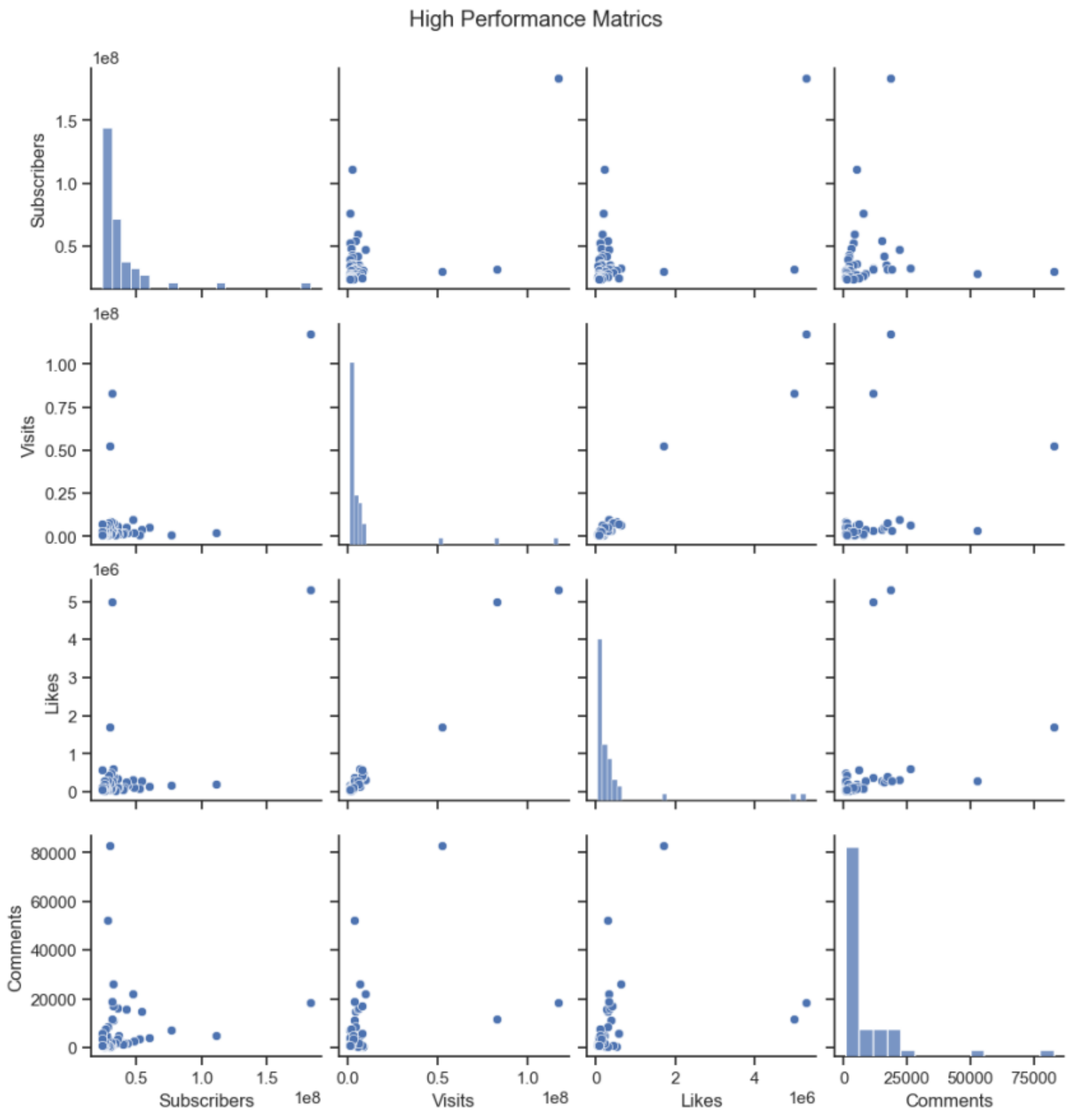
In [40]:

```
content_categories_distribution = df['Categories'].value_counts()
print(content_categories_distribution)
```

```
Categories
Unknown                                306
Música y baile                         160
Películas, Animación                    61
Música y baile, Películas               41
Vlogs diarios                           37
Noticias y Política                     36
Películas, Humor                        34
Animación, Videojuegos                  34
Animación, Juguetes                     29
Animación, Humor                        27
Películas                               24
Educación                               24
Animación                               22
Videojuegos                             19
Videojuegos, Humor                      17
Música y baile, Animación               16
Ciencia y tecnología                    14
Comida y bebida                         12
Humor                                   10
Juguetes                                10
Películas, Juguetes                      9
Películas, Videojuegos                   8
Deportes                                 8
Música y baile, Humor                    6
Juguetes, Coches y vehículos             4
DIY y Life Hacks                         3
Fitness, Salud y autoayuda               3
Videojuegos, Juguetes                    3
Animales y mascotas                      2
Moda                                     2
Coches y vehículos                       2
Educación, Juguetes                      2
Fitness                                  2
Comida y bebida, Juguetes                1
ASMR, Comida y bebida                    1
Animación, Humor, Juguetes               1
Diseño/arte, Belleza                     1
Belleza, Moda                            1
ASMR                                     1
Música y baile, Juguetes                 1
Diseño/arte, DIY y Life Hacks            1
DIY y Life Hacks, Juguetes               1
Diseño/arte                              1
Comida y bebida, Salud y autoayuda       1
Viajes, Espectáculos                     1
Juguetes, DIY y Life Hacks               1
Name: count, dtype: int64
```

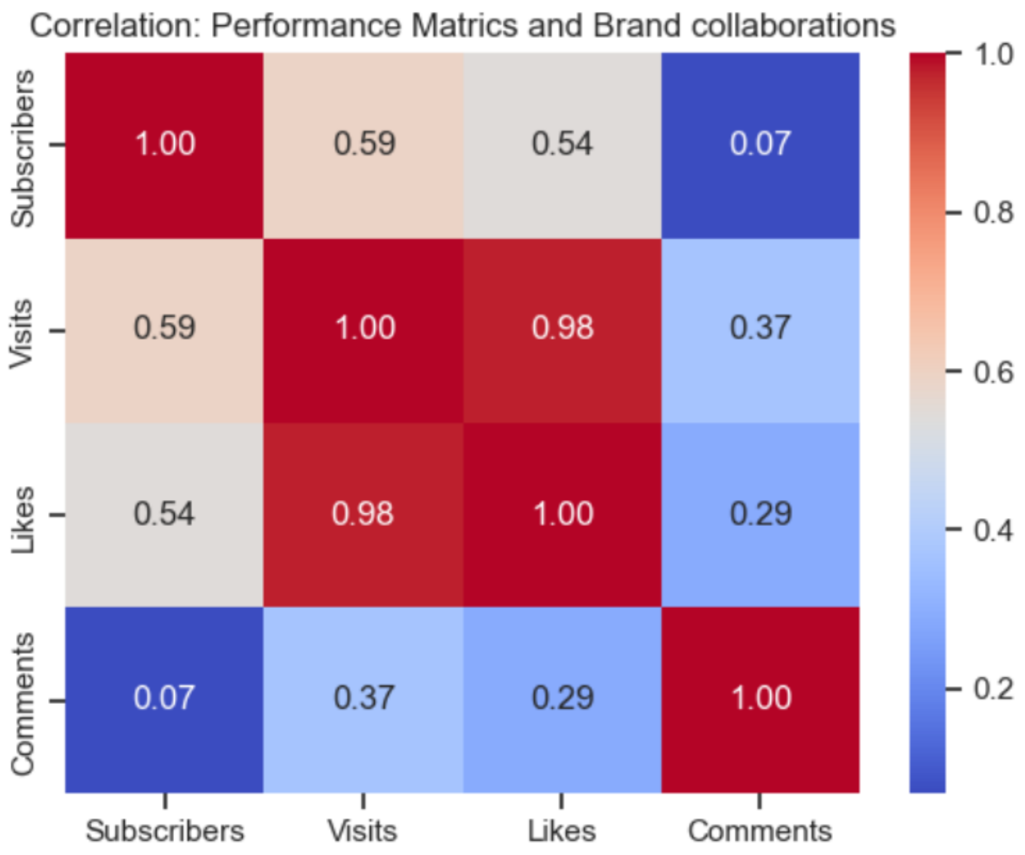
In [95]:
```
# Setting high performance thresholds.
threshold_subscribers = df['Subscribers'].quantile(0.75)
threshold_visits = df['Visits'].quantile(0.75)
threshold_likes = df['Likes'].quantile(0.75)
threshold_comments = df['Comments'].quantile(0.75)

high_performance_users = df[
    (df['Subscribers'] > threshold_subscribers) &
    (df['Visits'] > threshold_visits) &
    (df['Likes'] > threshold_likes) &
    (df['Comments'] > threshold_comments)]
```

```
sns.set(style="ticks")
sns.pairplot(high_performance_users, vars=['Subscribers', 'Visits', 'Likes', 'Co
plt.suptitle('High Performance Matrics', y=1.02)
plt.show()
```



```
In [96]: correlation_matrix = high_performance_users[['Subscribers', 'Visits', 'Likes', '
         sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
         plt.title('Correlation: Performance Matrics and Brand collaborations')
         print(correlation_matrix)
```

```
             Subscribers    Visits     Likes  Comments
Subscribers     1.000000  0.590301  0.542853  0.068471
Visits          0.590301  1.000000  0.977866  0.370393
Likes           0.542853  0.977866  1.000000  0.288881
Comments        0.068471  0.370393  0.288881  1.000000
```



In [116…

```
average_matrics = df[['Subscribers', 'Visits', 'Likes', 'Comments']].mean()

above_average_streamers = df[
    (df['Subscribers'] > average_matrics['Subscribers']) &
    (df['Visits'] > average_matrics['Visits']) &
    (df['Likes'] > average_matrics['Likes']) &
    (df['Comments'] > average_matrics['Comments'])
]

print("Top performing content creaters:")
print(above_average_streamers[['Username', 'Subscribers', 'Visits', 'Likes', 'Co
Top performing content creaters:
                  Username  Subscribers       Visits      Likes  Comments
1                  MrBeast  183500000.0  117400000.0  5300000.0   18500.0
5                PewDiePie  111500000.0    2400000.0   197300.0    4900.0
26             dudeperfect   59700000.0    5300000.0   156500.0    4200.0
34             TaylorSwift   54100000.0    4300000.0   300400.0   15000.0
39             JuegaGerman   48600000.0    2000000.0   117100.0    3000.0
43                A4a4a4a4   47300000.0    9700000.0   330400.0   22000.0
58               Mikecrack   43400000.0    2200000.0   183400.0    1800.0
62          KimberlyLoaiza   42100000.0    5300000.0   271300.0   16000.0
64         luisitocomunica   41100000.0    2500000.0   128900.0    1800.0
70             JessNoLimit   39600000.0    1300000.0    73500.0    1600.0
96          TotalGaming093   36300000.0    1500000.0   129400.0    4900.0
98    TechnoGamerzOfficial   35600000.0    6200000.0   341800.0   16500.0
100             markiplier   35500000.0    2100000.0   126500.0    3800.0
122                AboFlah   32700000.0    3300000.0   382000.0   11400.0
123          MRINDIANHACKER   32600000.0    6500000.0   617400.0   26000.0
131            fedevigevani   32000000.0    7700000.0   412200.0   17000.0
132                  dream   31900000.0    3300000.0   309200.0   19000.0
136               MrBeast2   31300000.0   83100000.0  5000000.0   11600.0
145           jacksepticeye   30400000.0    1600000.0    83400.0    2300.0
153              DaFuqBoom   29800000.0   52700000.0  1700000.0   82800.0
176               CrazyXYZ   27800000.0    4200000.0   284100.0    8600.0
177                 DanTDM   27800000.0    3500000.0   285000.0   52500.0
179            brentrivera   27600000.0    6400000.0   154100.0    5000.0
180               NichLmao   27500000.0    1500000.0    85800.0    1600.0
195             nickiminaj   26100000.0    1600000.0    98300.0    7600.0
206              AlejoIgoa   25700000.0    5700000.0   208400.0    1700.0
207                  ZHCYT   25700000.0    2600000.0   127300.0    2200.0
234                    rug   24300000.0    3200000.0    85300.0    5100.0
238             alanbecker   24300000.0    7600000.0   582600.0    5900.0
241      juandediospantojaa   24000000.0    3000000.0   133200.0    3600.0
266            DrossRotzank   23100000.0    1700000.0   105900.0    3900.0
272            AmiRodrigueZZ   22900000.0    4300000.0   294400.0    1300.0
278             StokesTwins   22700000.0   11700000.0   235000.0   10000.0
281                 SSundee   22700000.0    1700000.0    59800.0    1800.0
282     souravjoshivlogs7028   22700000.0    5600000.0   382300.0    8900.0
288   VillageCookingChannel   22500000.0   21500000.0   321500.0    5900.0
300             alfredolarin   21900000.0   12900000.0   707600.0    2100.0
302               royaltyfam   21900000.0    4700000.0    67000.0    6600.0
```

COMMENT: Conclusion - The first steps were data exploration and data cleaning - Unveiling the data structure, basically preparing it before conducting any form of analysis. The analysis allowed the questions about the data set to be answered, thereby deriving insight into the Performance matrix, trend analysis, audience study, and benchmarking.