INTRODUCTION

YouTube has emerged as one of the leading platforms in the modern digital space for content creation, audience building, and monetization. Therefore, a deep understanding of the underlying factors driving performance is needed to optimize the content strategy for long-term growth. The project focuses on deep YouTube channel analysis using Python by developing a data-driven approach to unearth trends, engagement patterns, and monetization insights.

The analysis examines key performance indicators such as views, impressions, subscriber growth, engagement metrics, and revenue trends. By utilizing advanced analytical methods, including time-series forecasting with the FB Prophet model, this study identifies optimal content release schedules, seasonal variations, and factors influencing audience retention. it explores the impact of video duration on engagement and the relationship between ad impressions and revenue generation.

With findings revealing critical insights such as peak performance periods, declining engagement trends post-2017, and the influence of video length on interaction this project provides actionable recommendations for enhancing content strategy. It intends to arm creators with data-supported strategies to push reach, leverage audience engagement, and optimize their monetization further to sustainable successes on the platform.

In [322]:
```python
#importing important libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from pandas import set_option
import statsmodels.api as sm
from prophet import Prophet
from statsmodels.tools.eval_measures import rmse
from prophet.plot import plot_plotly, plot_components_plotly
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import warnings
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
```

In [3]:
```python
#loading data
data = "C:\\Users\\HP\\Documents\\WORKSPACE\\youtube_channel_real_performance_analyti
df = pd.read_csv(data)
```

In [6]:
```python
# Set pandas options to display all columns
pd.set_option('display.max_columns', None)
```

In [7]:
```
df.head()
```
Out[7]:

	ID	Video Duration	Video Publish Time	Days Since Publish	Day	Month	Year	Day of Week	Revenue per 1000 Views (USD)	Monetized Playbacks (Estimate)	Playback-Based CPM (USD)	CPM (USD
0	0	201.0	2016-06-02 00:00:00	0	2	6	2016	Thursday	0.024	723.0	1.386	1.02
1	1	391.0	2016-06-10 00:00:00	8	10	6	2016	Friday	0.056	727.0	1.612	1.36
2	2	133.0	2016-06-14 00:00:00	4	14	6	2016	Tuesday	0.014	76.0	2.105	1.81
3	3	14.0	2016-06-29 00:00:00	15	29	6	2016	Wednesday	0.004	18.0	1.667	0.85
4	4	45.0	2016-07-01 00:00:00	2	1	7	2016	Friday	0.000	0.0	0.000	0.00

In [11]:
```
# Convert 'Date' column to datetime format
df['Video Publish Time'] = pd.to_datetime(df['Video Publish Time'])
```

In [12]:
```
# Function to extract the month
def extract_month(date):
    return date.strftime('%B')  # '%B' format returns the full month name
```

In [14]:
```
# Apply the extract_month function to create a new 'Month' column
df['Months'] = df['Video Publish Time'].apply(extract_month)
```

In [15]:
```
df.head()
```
Out[15]:

	ID	Video Duration	Video Publish Time	Days Since Publish	Day	Month	Year	Day of Week	Revenue per 1000 Views (USD)	Monetized Playbacks (Estimate)	Playback-Based CPM (USD)	CPM (USD)
0	0	201.0	2016-06-02	0	2	6	2016	Thursday	0.024	723.0	1.386	1.021
1	1	391.0	2016-06-10	8	10	6	2016	Friday	0.056	727.0	1.612	1.361
2	2	133.0	2016-06-14	4	14	6	2016	Tuesday	0.014	76.0	2.105	1.818
3	3	14.0	2016-06-29	15	29	6	2016	Wednesday	0.004	18.0	1.667	0.857
4	4	45.0	2016-07-01	2	1	7	2016	Friday	0.000	0.0	0.000	0.000

In [16]:
```
df.describe()
```
Out[16]:
```
                      Video        Days                                             Revenue      Monet
               ID  Duration       Since         Day       Month         Year    per 1000      Playba
                                Publish                                            Views      (Estim
                                                                                   (USD)
count  364.000000  364.000000  364.000000  364.000000  364.000000   364.000000  364.000000   364.00
mean   181.500000  664.239011    8.406593   15.807692    6.642857  2018.736264    0.112283   8934.71
std    105.221988  330.646183   15.371239    8.924004    3.421521     2.530629    0.179668  12967.84
min      0.000000    9.000000    0.000000    1.000000    1.000000  2016.000000    0.000000      0.00
25%     90.750000  496.000000    3.000000    8.000000    4.000000  2017.000000    0.015000    349.50
50%    181.500000  613.000000    5.000000   16.000000    7.000000  2018.000000    0.050000   3564.00
75%    272.250000  786.500000    9.000000   23.000000   10.000000  2021.000000    0.143500  13579.25
max    363.000000 2311.000000  260.000000   31.000000   12.000000  2024.000000    1.600000 100566.00
```

In [74]:
```
#Features I will be working with
chosen_features= ["Video Duration","Days Since Publish","Months","Day of Week", "Year
                  "CPM (USD)", "Ad Impressions","Estimated AdSense Revenue (USD)","D
                  "Watch Page Ads Revenue (USD)","New Comments","Shares","Like Rate (
                  "Unsubscribes","New Subscribers","End Screen Click-Through Rate (%)
                  "Teaser Click-Through Rate (%)","Teaser Impressions","Teaser Clicks
                  "Card Clicks","Views per Playlist Start", "Playlist Views","Playlis
                  "Clip Views","YouTube Premium Watch Time (hours)","YouTube Premium
                  "Average View Percentage (%)","Average View Duration","Views","Wat
                  "Impressions","Video Thumbnail CTR (%)"

                 ]
```

In [75]:
```
# Select the chosen features from the DataFrame
df1 = df[chosen_features]
```

In [76]:
```
df1.head()
```
Out[76]:
```
                                                   Revenue                Playback-                      Es
     Video   Days                   Day of         per 1000   Monetized      Based     CPM          Ad    A
  Duration  Since  Months             Week  Year      Views   Playbacks        CPM   (USD)  Impressions   F
          Publish                                     (USD)  (Estimate)      (USD)
0    201.0      0    June         Thursday  2016      0.024       723.0      1.386   1.021        981.0
1    391.0      8    June           Friday  2016      0.056       727.0      1.612   1.361        861.0
2    133.0      4    June          Tuesday  2016      0.014        76.0      2.105   1.818         88.0
3     14.0     15    June        Wednesday  2016      0.004        18.0      1.667   0.857         35.0
4     45.0      2    July           Friday  2016      0.000         0.0      0.000   0.000          0.0
```

In [ ]:

```
In [77]: df1.describe().T
```

Out[77]:

	count	mean	std	min	25%	50%	75
Video Duration	364.0	664.239011	3.306462e+02	9.0000	496.000000	613.00000	7.865000e+
Days Since Publish	364.0	8.406593	1.537124e+01	0.0000	3.000000	5.00000	9.000000e+
Year	364.0	2018.736264	2.530629e+00	2016.0000	2017.000000	2018.00000	2.021000e+
Revenue per 1000 Views (USD)	364.0	0.112283	1.796680e-01	0.0000	0.015000	0.05000	1.435000e-
Monetized Playbacks (Estimate)	364.0	8934.711538	1.296784e+04	0.0000	349.500000	3564.00000	1.357925e+
Playback-Based CPM (USD)	364.0	1.553734	1.078549e+00	0.0000	0.817750	1.54050	2.025250e+
CPM (USD)	364.0	1.059005	7.283799e-01	0.0000	0.522000	1.15550	1.458750e+
Ad Impressions	364.0	15117.953297	2.433992e+04	0.0000	368.500000	5892.00000	1.937200e+
Estimated AdSense Revenue (USD)	364.0	7.811525	1.277881e+01	0.0000	0.167000	3.32950	1.019550e+
DoubleClick Revenue (USD)	364.0	0.635734	1.393079e+00	0.0000	0.000000	0.00000	5.265000e-
YouTube Ads Revenue (USD)	364.0	15.363964	2.361417e+01	0.0000	0.344500	7.52750	2.007175e+
Watch Page Ads Revenue (USD)	364.0	8.447223	1.299339e+01	0.0000	0.171500	4.13950	1.104275e+
New Comments	364.0	333.846154	2.919346e+02	0.0000	104.750000	262.50000	4.850000e+
Shares	364.0	252.958791	3.630164e+02	1.0000	38.000000	149.00000	3.132500e+
Like Rate (%)	364.0	97.339203	2.308451e+00	76.9800	96.597500	97.74000	9.897500e+
Dislikes	364.0	123.961538	1.283116e+02	2.0000	27.000000	85.50000	1.792500e+
Likes	364.0	5526.733516	4.465211e+03	121.0000	1205.500000	5172.00000	8.504750e+
Unsubscribes	364.0	28.906593	4.859193e+01	0.0000	3.000000	20.00000	4.100000e+
New Subscribers	364.0	349.931319	4.685390e+02	0.0000	44.500000	200.00000	4.640000e+
End Screen Click-Through Rate (%)	364.0	1.551676	3.031638e+00	0.0000	0.000000	0.00000	2.065000e+
End Screen Impressions	364.0	6561.980769	1.482902e+04	0.0000	3.000000	79.50000	1.031875e+
End Screen Clicks	364.0	224.934066	7.875239e+02	0.0000	0.000000	0.00000	1.820000e+
Teaser Click-Through Rate (%)	364.0	0.324725	9.234912e-01	0.0000	0.000000	0.00000	0.000000e+
Teaser Impressions	364.0	4131.986264	1.352398e+04	0.0000	0.000000	0.00000	3.600000e+

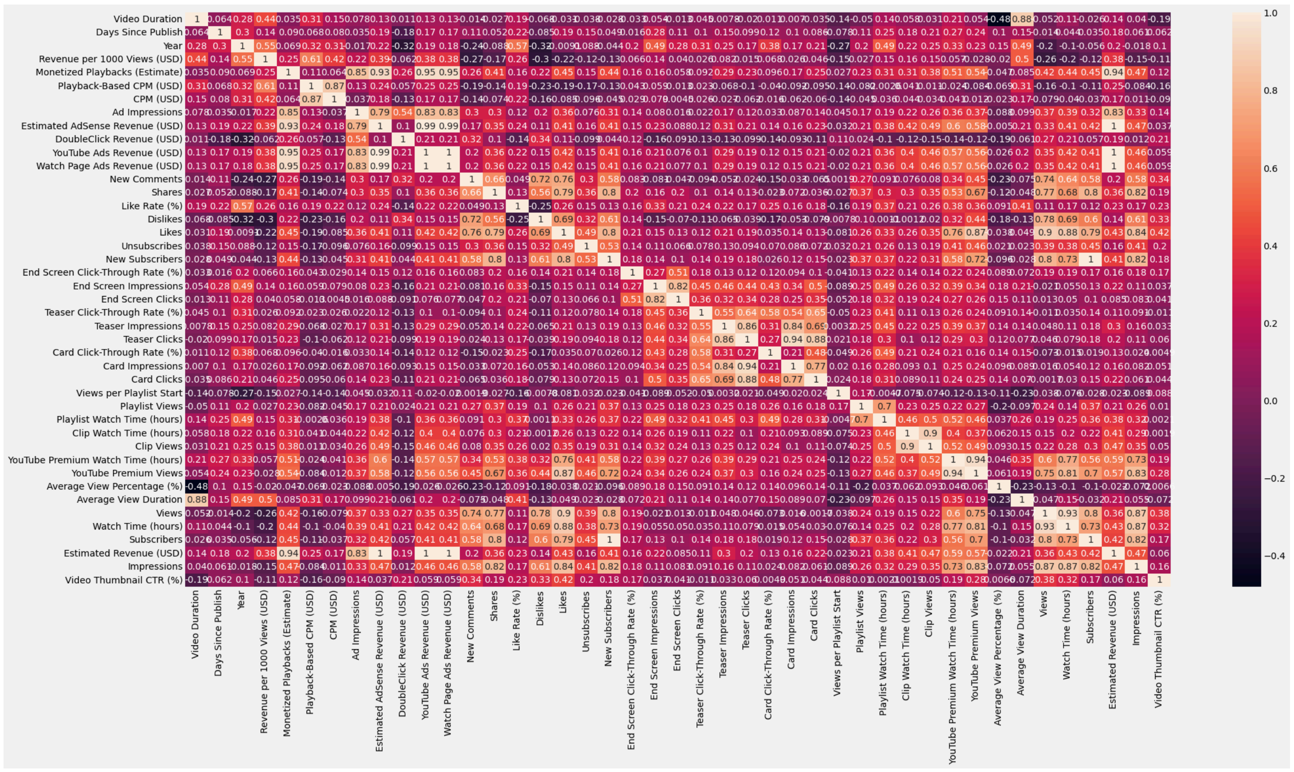
	count	mean	std	min	25%	50%	75
Teaser Clicks	364.0	81.318681	3.520308e+02	0.0000	0.000000	0.00000	0.000000e+
Card Click-Through Rate (%)	364.0	3.291044	1.033844e+01	0.0000	0.000000	0.00000	0.000000e+
Card Impressions	364.0	142.689560	6.691146e+02	0.0000	0.000000	0.00000	1.250000e+
Card Clicks	364.0	19.115385	8.251680e+01	0.0000	0.000000	0.00000	0.000000e+
Views per Playlist Start	364.0	2.644910	1.917453e+00	0.0000	1.720275	2.36350	3.033450e+
Playlist Views	364.0	300.810440	5.559761e+02	0.0000	8.000000	33.00000	4.410000e+
Playlist Watch Time (hours)	364.0	24.830608	3.895840e+01	0.0000	0.409050	2.35050	3.947555e+
Clip Watch Time (hours)	364.0	0.068246	2.093880e-01	0.0000	0.000000	0.00000	3.372500e-
Clip Views	364.0	5.923077	1.429094e+01	0.0000	0.000000	0.00000	5.000000e+
YouTube Premium Watch Time (hours)	364.0	115.206959	1.288271e+02	0.0000	8.999700	83.19235	1.721456e+
YouTube Premium Views	364.0	1303.763736	1.343028e+03	0.0000	170.500000	958.00000	2.011500e+
Average View Percentage (%)	364.0	45.970989	1.245747e+01	15.0500	40.252500	45.46500	5.101000e+
Average View Duration	364.0	285.612637	1.165957e+02	8.0000	233.750000	286.50000	3.442500e+
Views	364.0	128800.101648	1.182098e+05	2461.0000	27160.500000	101950.50000	1.981695e+
Watch Time (hours)	364.0	10058.965455	9.516160e+03	12.6979	2759.404250	8102.96295	1.408450e+
Subscribers	364.0	321.024725	4.445948e+02	-3.0000	41.000000	180.00000	4.212500e+
Estimated Revenue (USD)	364.0	8.852052	1.341465e+01	0.0000	0.443250	4.28500	1.147625e+
Impressions	364.0	959528.618132	1.180519e+06	32228.0000	131708.250000	620406.50000	1.368464e+
Video Thumbnail CTR (%)	364.0	7.914615	2.903383e+00	0.6200	6.420000	8.39500	9.920000e+

```
Based on the ouput of the decriptive statistics some of the insights that I was able to
discover are:

Video Duration:
Mean: ~664 seconds (~11 minutes).
Std Dev: ~330 seconds. The wide range suggests diverse content lengths.
Min & Max: The shortest video is 9 seconds, while the longest is ~38 minutes.
Insight: The median duration is ~10 minutes (613 seconds), indicating a trend toward
mid-length videos. Analyzing audience retention for varying durations could reveal
optimal video lengths for engagement.

 Revenue per 1000 Views (USD):
```

```
Mean: $0.11 per 1,000 views.
Std Dev: ~$0.18. A large variation exists across videos.
Max: $1.60, which shows potential for high revenue if targeted audiences or ad
strategies align.
Insight: You can focus on videos with higher CPMs to understand the drivers behind
their better performance.

Playback-Based CPM (USD):
Mean: $1.55, with a max of $7.39.
Insight: Videos with a high CPM likely cater to niches attractive to advertisers.
Identify these niches for strategic content planning can be a good strategy.

Likes & Dislikes:
Likes: Mean ~5,526 with a max of ~27,222.
Dislikes: Mean ~124 with a max of ~818.
Like Rate (%): Mean ~97.3%. This shows overall positive reception.
Insight: Videos with high likes but low dislikes may highlight effective content
strategies that resonate with the audience. Further analysis can be done to know what
kind of videos or contents you
audience find intresting.

Subscribers:
Mean: ~321 per video, with a max of ~3,728.
Min: Some videos caused a loss of up to 3 subscribers.
Insight: Understanding what content drives subscriber gain vs. loss can improve
audience retention strategies.

Estimated Revenue (USD):
Mean: ~$8.85 per video.
Max: ~$103.
Insight: Videos generating the highest revenue could be studied to replicate success
(topics, posting time, audience).

Impressions:
Mean: ~959,528 impressions per video, with a max of ~12.7M.
```

In [86]:

```python
#heatmap showing the correlation of the variables
plt.figure(figsize = (30,15))
sns.heatmap(df1.corr(), annot = True)
set_option ('display.width', 1000)
```



This correlation heatmap offers valuable insights into how various features within the dataset interact with one another. It helps us understand the strength and direction of the relationships between these features, shedding light on how closely they are connected and influencing each other. We will be analying some of the relationships

Views vs. Likes: High Positive Correlation: Strong correlation between views and likes indicates that videos with more views tend to receive more likes. This is expected since more exposure generally leads to more engagement.

Views vs. Comments: Moderate-to-High Positive Correlation: Videos with higher views often generate more comments. This implies that popular videos tend to spark discussions, likely due to interesting or engaging content.

Likes vs. Comments: Strong Positive Correlation: Videos that receive more likes often receive more comments. Engaging videos inspire viewers to react and discuss further, boosting both metrics.

Views vs. Shares: Moderate Positive Correlation: Videos with more views are often shared more. While the correlation may not be as strong as with likes or comments, it reflects how engaging or impactful videos can drive shares.

Views vs. Watch Time: Strong Positive Correlation: Higher view counts typically lead to increased watch time, as more viewers collectively contribute to the overall watch time. This also reflects how compelling content can retain viewers.

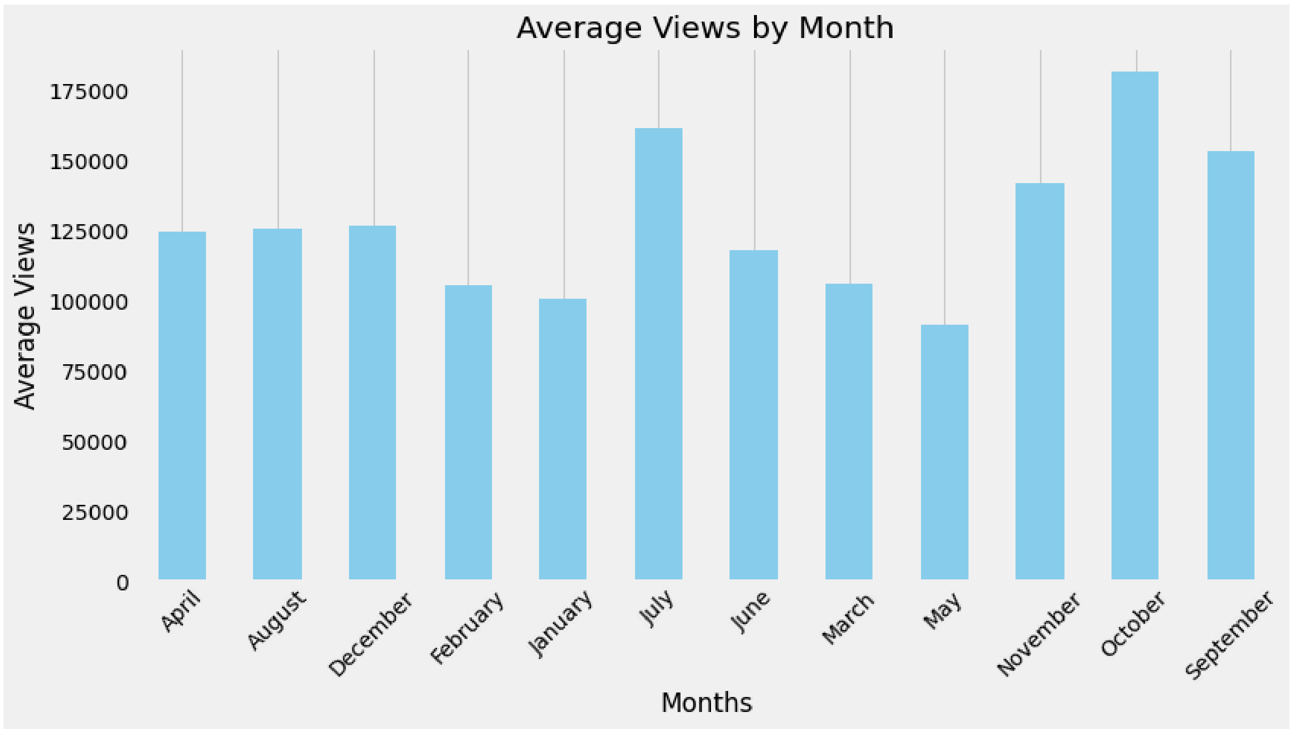
Shares vs. Watch Time: Moderate-to-High Positive Correlation: Videos that are shared more tend to have higher watch time, as shared content likely attracts more viewers who watch a substantial portion of the video.

In [ ]:
```
Analyzing Different== Metrics Accross Days, Month, Week and Year
```

To get more insights from the data we will be exploring some more features like view by week, months and year.

In [44]:
```
# Explore views by month
views_by_month = df1.groupby('Months')['Views'].mean()
```

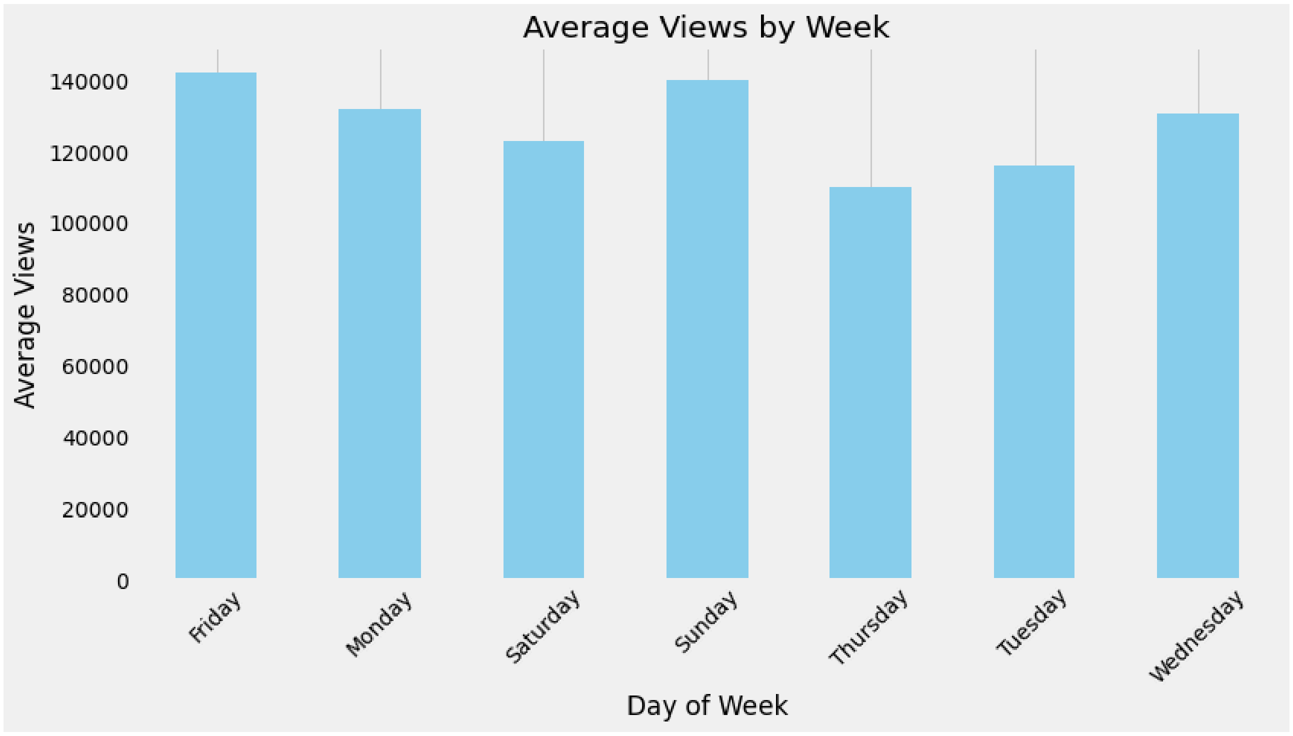
In [54]:
```
# Plot views by month
plt.figure(figsize=(12, 6))
views_by_month.plot(kind='bar', color='skyblue')
plt.title('Average Views by Month')
plt.xlabel('Months')
plt.ylabel('Average Views')
plt.xticks(rotation=45)
plt.grid(axis='y')
```



Based on the chart, July and October emerged as the months with the highest view counts. This raises an important question: What factors contributed to the surge in views during these specific months? To gain deeper insights, we will further explore other features within the dataset to identify potential patterns, trends, or external influences that may have impacted viewer engagement during these periods.

In [50]:
```
# Explore views by week
week_by_month = df1.groupby('Day of Week')['Views'].mean()
```

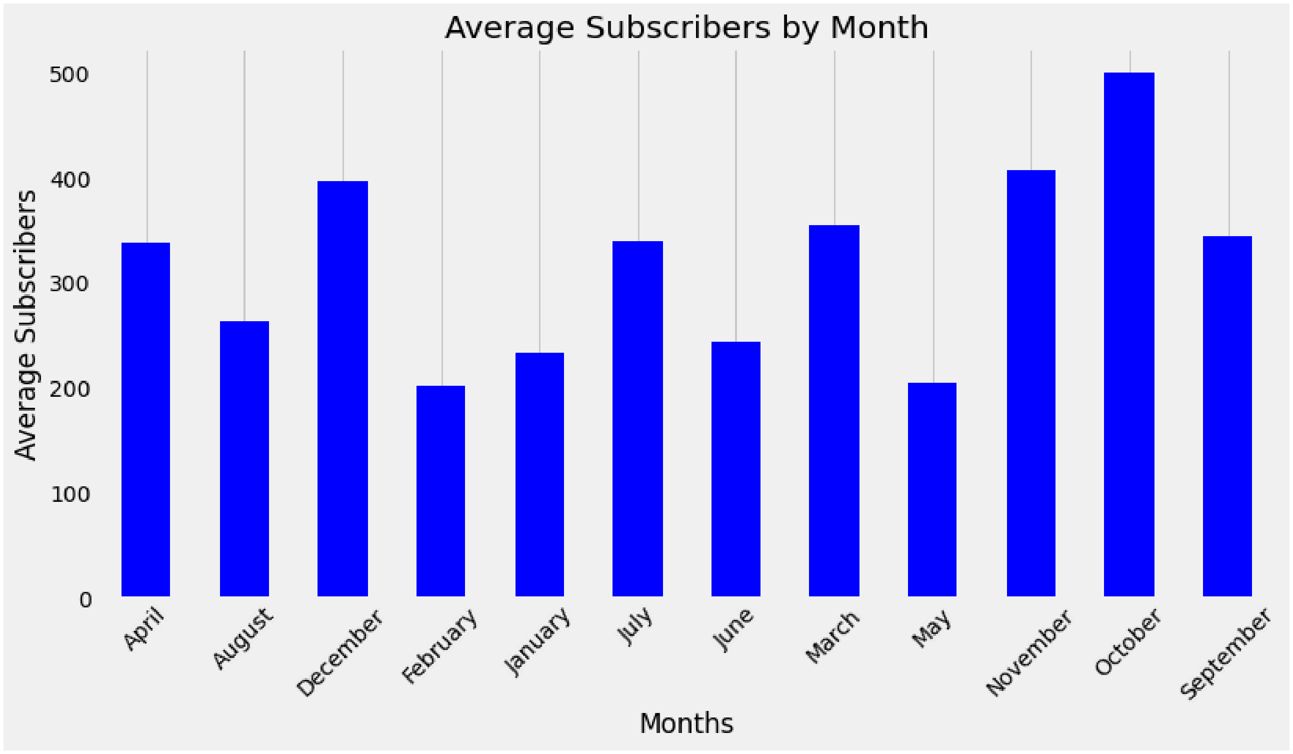
In [56]:
```python
# Plot views by month
plt.figure(figsize=(12, 6))
week_by_month.plot(kind='bar', color='skyblue')
plt.title('Average Views by Week')
plt.xlabel('Day of Week')
plt.ylabel('Average Views')
plt.xticks(rotation=45)
plt.grid(axis='y')
```



In terms of average views by day of the week, an interesting trend emerges: Friday and Sunday appear to attract the highest number of views. This suggests that audience engagement is likely higher on these days, possibly due to factors such as increased free time, weekend relaxation, or specific content preferences aligning with end-of-week entertainment habits. To further understand this trend, we will analyze additional features to determine what drives higher engagement on these particular days.

In [59]:
```python
# Explore subscribers by month
views_by_month = df1.groupby('Months')['Subscribers'].mean()
```

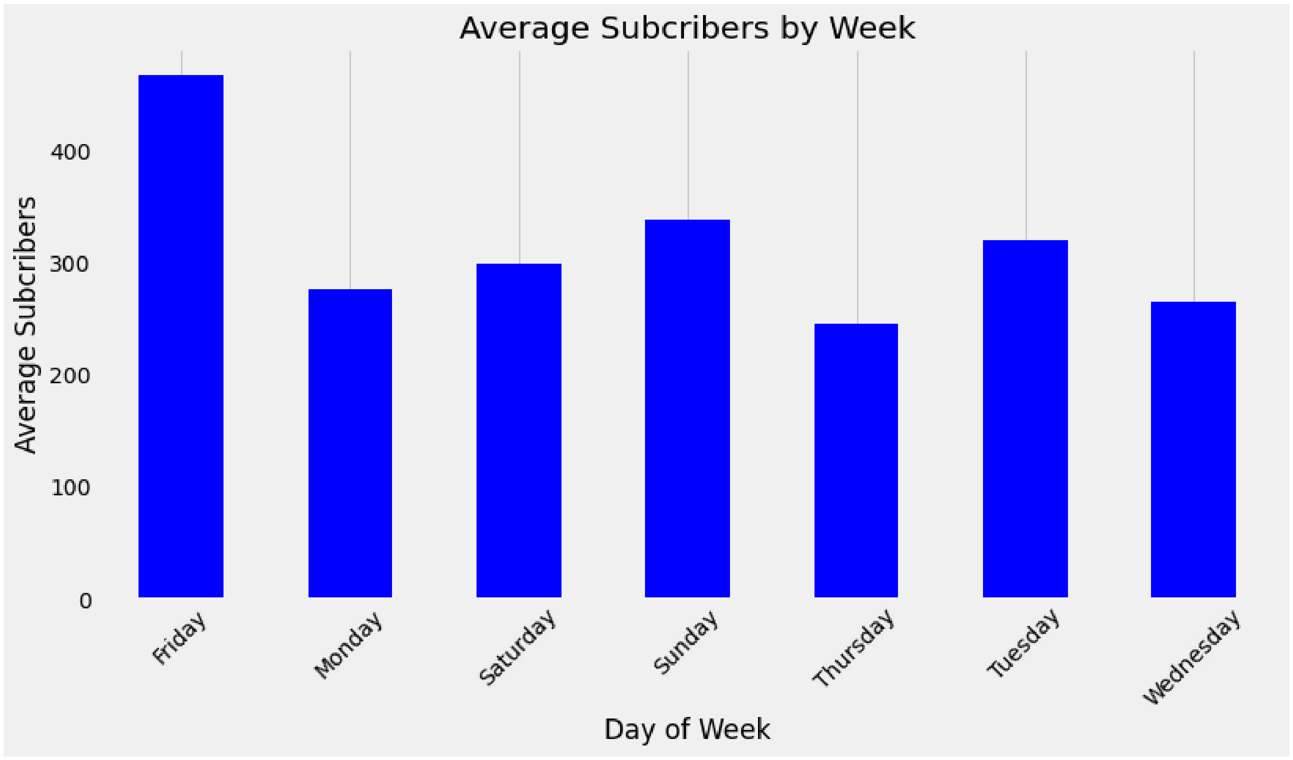
In [60]:
```python
# Plot subscribers by month
plt.figure(figsize=(12, 6))
views_by_month.plot(kind='bar', color='blue')
plt.title('Average Subscribers by Month')
plt.xlabel('Months')
plt.ylabel('Average Subscribers')
plt.xticks(rotation=45)
plt.grid(axis='y')
```



October stands out once again, performing exceptionally well in terms of subscriber growth, mirroring its strong performance in views. This correlation suggests that higher engagement during this month may have contributed to increased subscriptions, possibly due to compelling content, seasonal trends, or marketing efforts. November and December also show strong subscriber growth, which could be influenced by factors such as holiday seasons, end-of-year trends, or increased content consumption during festive periods. Further analysis of video content, audience behavior, and promotional activities during these months can help uncover the key drivers behind this trend.

In [62]:
```python
# Explore subscribers by week
week_by_month = df1.groupby('Day of Week')['Subscribers'].mean()
```

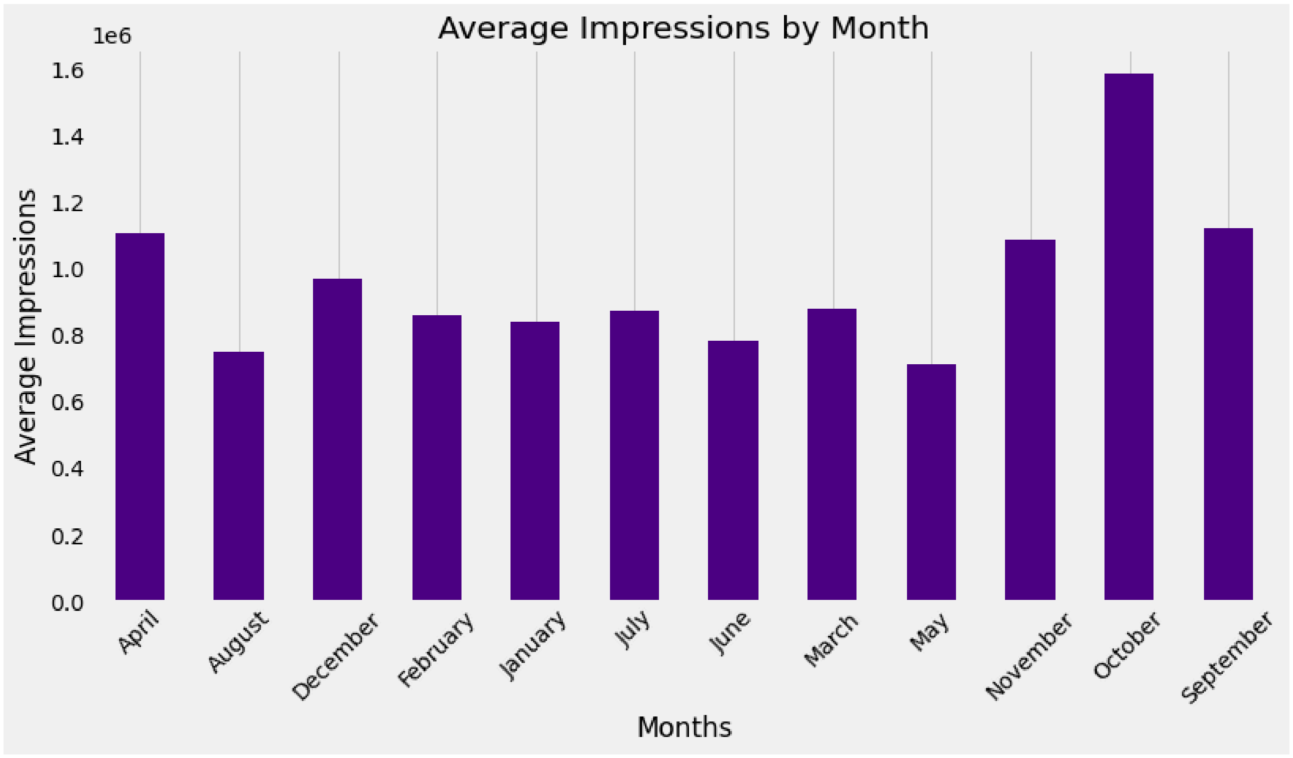
In [63]:
```
# Plot subcribers by week
plt.figure(figsize=(12, 6))
week_by_month.plot(kind='bar', color='blue')
plt.title('Average Subcribers by Week')
plt.xlabel('Day of Week')
plt.ylabel('Average Subcribers')
plt.xticks(rotation=45)
plt.grid(axis='y')
```



Similar to the trend observed in views, subscriber growth also appears to peak on certain days of the week, with Friday standing out once again as a particularly strong day. This suggests that audience engagement tends to be higher toward the end of the workweek, possibly due to increased leisure time, a shift in online behavior, or the release schedule of content. Understanding this pattern can help in optimizing video uploads and promotional strategies to maximize both views and subscriber conversions. Further analysis into content types can provide deeper insights into why Fridays perform exceptionally well.

In [65]:
```
# Explore impressions by month
imp_by_month = df1.groupby('Months')['Impressions'].mean()
```

In [66]:
```
# Plot impressions by month
plt.figure(figsize=(12, 6))
imp_by_month.plot(kind='bar', color='indigo')
plt.title('Average Impressions by Month')
plt.xlabel('Months')
plt.ylabel('Average Impressions')
plt.xticks(rotation=45)
plt.grid(axis='y')
```

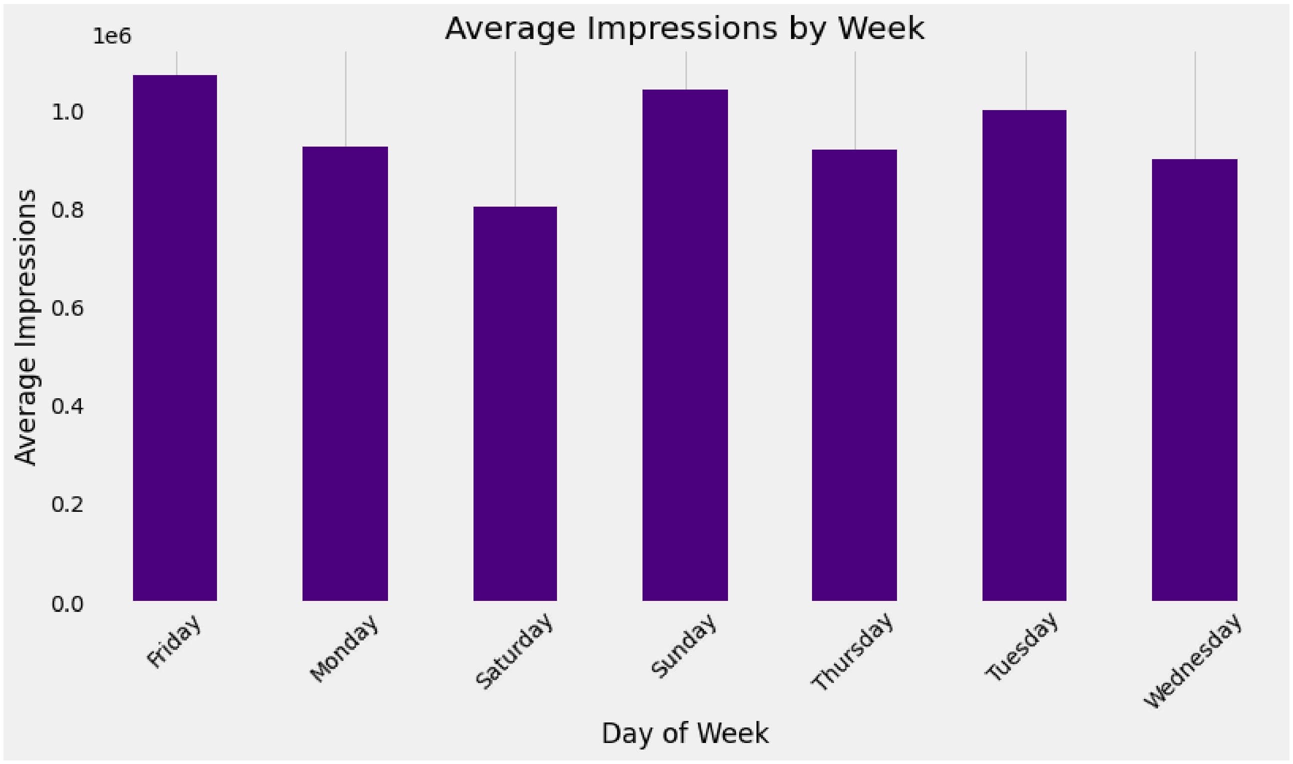


Once again, in terms of impressions by month, October performed exceptionally well, standing out as the month with the highest number of impressions. This reinforces the trend we previously observed with views and subscriber growth, suggesting that October was a particularly strong month for audience engagement. The high impressions indicate that content was widely surfaced to viewers, possibly due to factors such as trending topics, seasonal interests, or improved recommendation algorithms. This trend prompts further investigation into what specific content performed well in October and whether external factors, such as holidays, events, or marketing strategies, played a role in driving traffic.

In [67]:
```
# Explore impressions by week
imp_by_week = df1.groupby('Day of Week')['Impressions'].mean()
```

In [68]:

```python
# Plot Impressions by week
plt.figure(figsize=(12, 6))
imp_by_week.plot(kind='bar', color='indigo')
plt.title('Average Impressions by Week')
plt.xlabel('Day of Week')
plt.ylabel('Average Impressions')
plt.xticks(rotation=45)
plt.grid(axis='y')
```

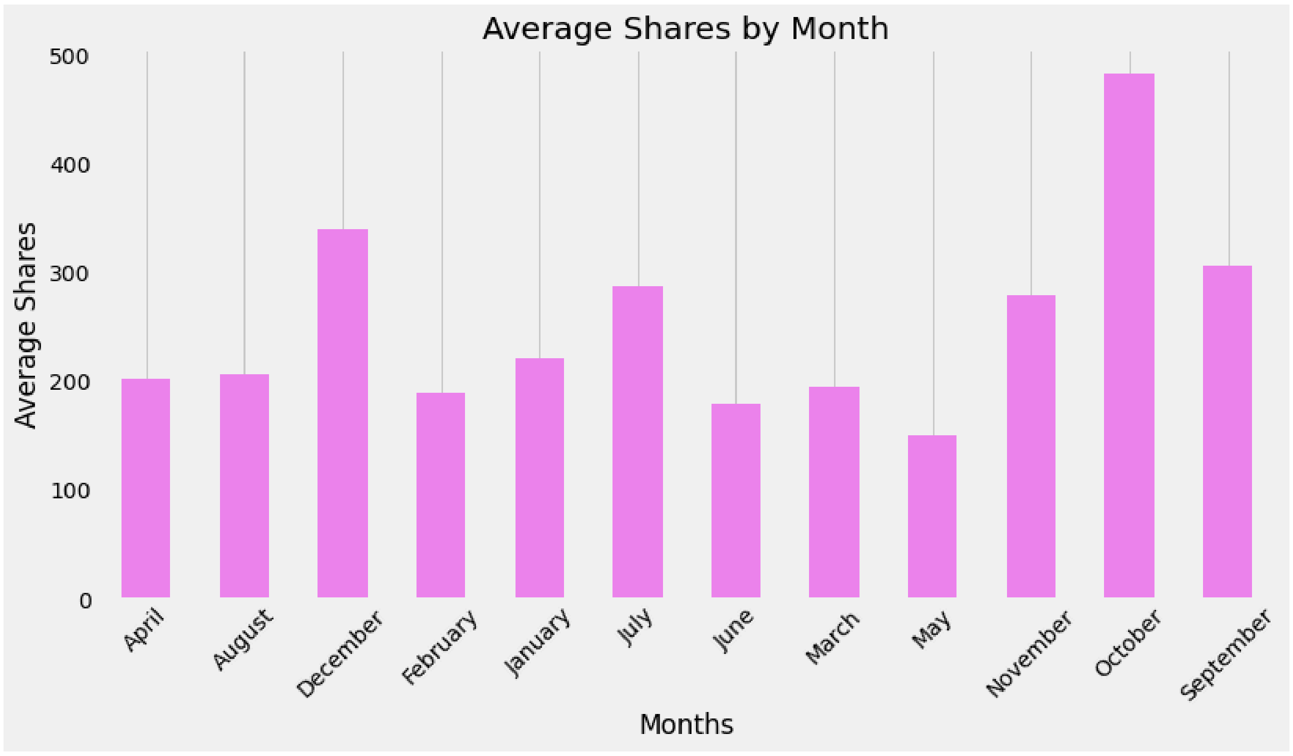


Similar to what we observed with views and subscriber growth, impressions were also notably high on Fridays, with Sundays and Tuesdays performing well too. This trend suggests that these days might be optimal for reaching a larger audience, possibly due to user behavior patterns, such as increased browsing time towards the end of the workweek and on weekends. The strong performance on these days could indicate that viewers are more active and engaged, leading to higher exposure for uploaded content. Further analysis could help determine whether this trend is influenced by specific video topics, promotional strategies, or YouTube's recommendation system.

In [89]:

```python
# Explore shares by month
sh_by_month = df1.groupby('Months')['Shares'].mean()
```

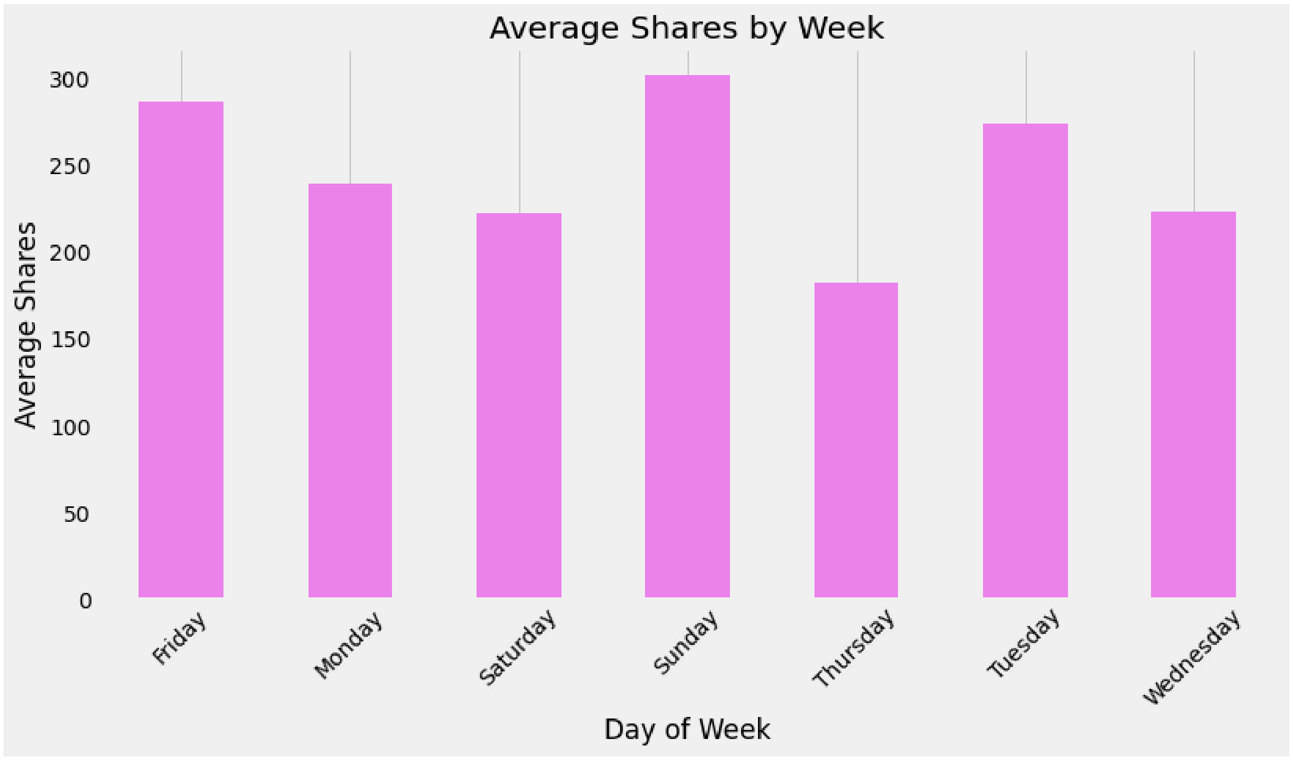
In [90]:
```
# Plot shares by month
plt.figure(figsize=(12, 6))
sh_by_month.plot(kind='bar', color='violet')
plt.title('Average Shares by Month')
plt.xlabel('Months')
plt.ylabel('Average Shares')
plt.xticks(rotation=45)
plt.grid(axis='y')
```



Analyzing the number of shares provides valuable insights into how well the content resonates with the audience and how often viewers find it compelling enough to share with others. In this regard, October significantly outperformed all other months, reinforcing its strong overall performance in terms of views, impressions, and subscriber growth. This suggests that content uploaded during this period was particularly engaging or benefited from external factors such as seasonal trends, viral appeal, or optimized promotion strategies. December also performed well, possibly due to increased online activity during the holiday season.

In [92]:
```
# Explore shares by week
sh_by_week = df1.groupby('Day of Week')['Shares'].mean()
```

In [93]:
```
# Plot Impressions by week
plt.figure(figsize=(12, 6))
sh_by_week.plot(kind='bar', color='violet')
plt.title('Average Shares by Week')
plt.xlabel('Day of Week')
plt.ylabel('Average Shares')
plt.xticks(rotation=45)
plt.grid(axis='y')
```

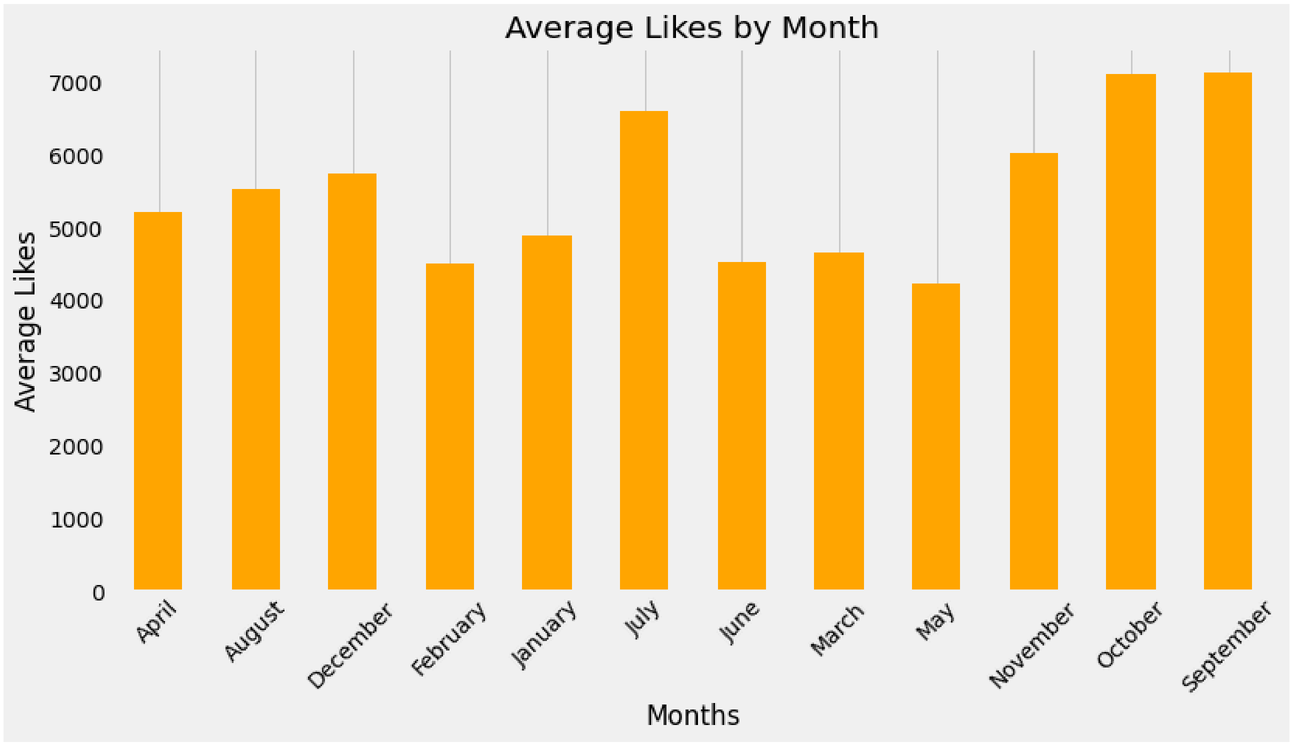


Analyzing the distribution of shares by the day of the week provides key insights into audience engagement and content virality. Notably, Sunday recorded the highest number of shares, suggesting that viewers are more likely to engage with and share content at the end of the week, possibly when they have more leisure time to consume and interact with videos. Friday and Tuesday also performed well in terms of shares, indicating strong midweek and weekend engagement patterns. Unlike other metrics such as views or impressions, which showed consistency on Fridays, shares appear to peak specifically on Sundays. This trend highlights the importance of strategic content scheduling uploading and promoting videos towards the end of the week may increase the likelihood of them being shared, thereby boosting overall reach and engagement.

In [ ]:
```

```

In [95]:
```
# Explore Likes by month
lks_by_month = df1.groupby('Months')['Likes'].mean()
```

In [96]:
```
# Plot Likes by month
plt.figure(figsize=(12, 6))
lks_by_month.plot(kind='bar', color='orange')
plt.title('Average Likes by Month')
plt.xlabel('Months')
plt.ylabel('Average Likes')
plt.xticks(rotation=45)
plt.grid(axis='y')
```

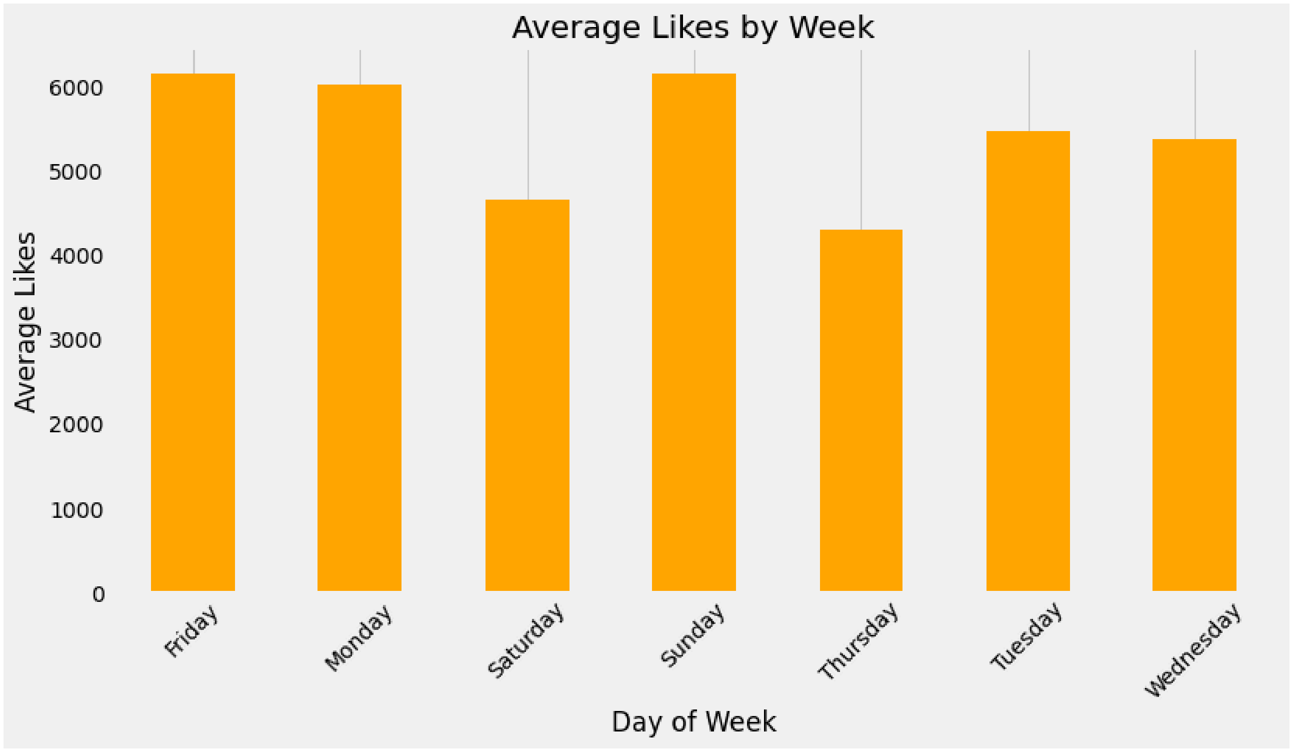


Examining the distribution of likes by month reveals that October and September stood out as the top-performing months, with July also coming in strong. This trend aligns with previous observations regarding high views, impressions, and subscriber growth during these months, suggesting a period of heightened audience engagement. The significant number of likes indicates that content released in these months resonated well with viewers, either due to its quality, relevance, or timing.

This pattern raises interesting questions about seasonal trends, content strategy, and audience behavior. Were specific types of videos uploaded in these months that contributed to higher engagement? Did external factors, such as holidays, events, or trends, play a role in boosting interaction? Further analysis of video topics, posting frequency, and audience demographics could help uncover why these months performed exceptionally well in terms of likes and whether similar patterns can be replicated in the future.

In [97]:
```
# Explore Likes by week
lks_by_week = df1.groupby('Day of Week')['Likes'].mean()
```

In [98]:
```
# Plot Impressions by week
plt.figure(figsize=(12, 6))
lks_by_week.plot(kind='bar', color='orange')
plt.title('Average Likes by Week')
plt.xlabel('Day of Week')
plt.ylabel('Average Likes')
plt.xticks(rotation=45)
plt.grid(axis='y')
```



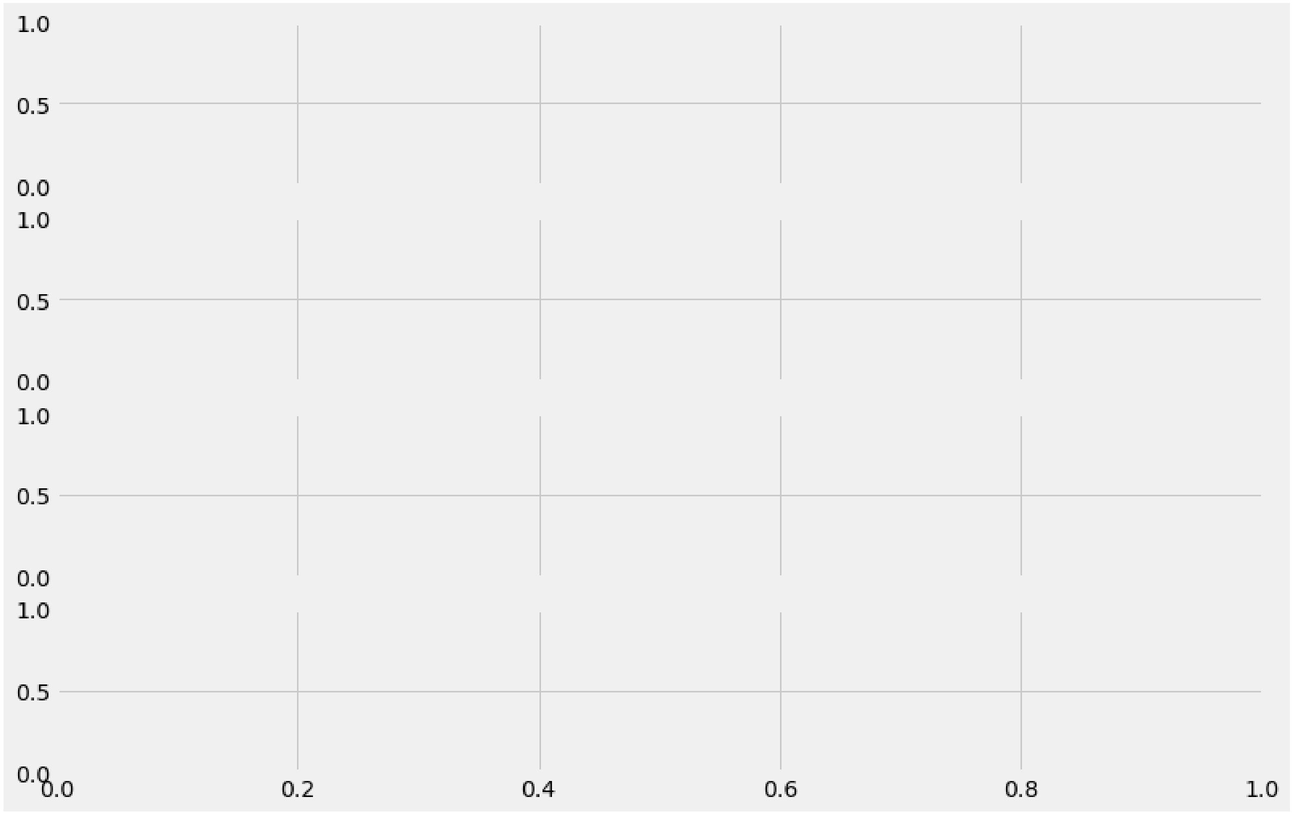
Analyzing likes by day of the week reveals a consistent trend, with Friday once again emerging as one of the top-performing days. This aligns with its strong performance in other metrics, such as views and impressions. Monday and Sunday also show robust engagement, suggesting these days foster higher interaction from viewers.

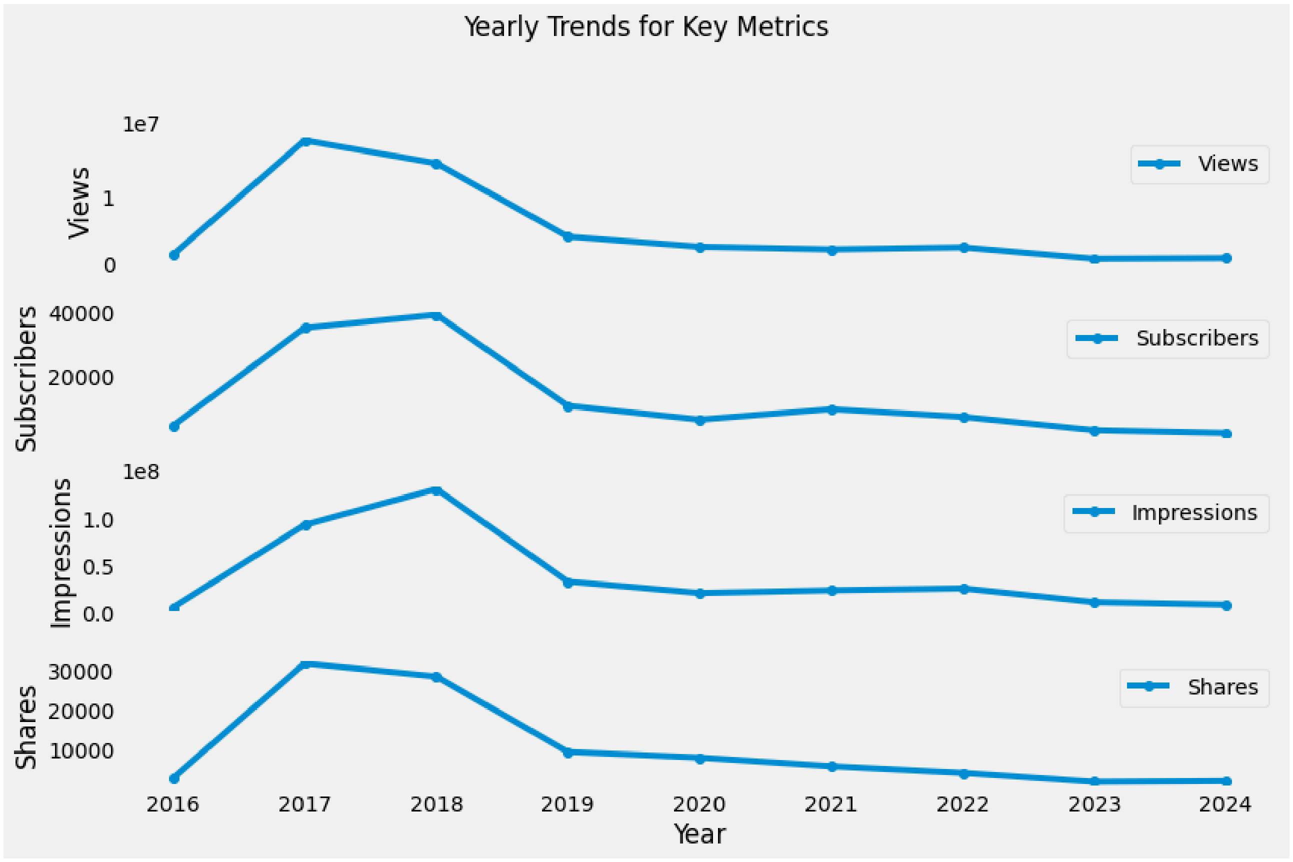
The strong performance on Fridays may reflect a behavioral pattern where audiences are more relaxed and receptive to engaging with content as they transition into the weekend. Similarly, Sunday, often associated with downtime before the workweek, may encourage viewers to catch up on or share content. Meanwhile, Monday's performance could signal an active start to the week, with viewers re-engaging after the weekend.

In [ ]:
```

```

In [100]:
```
# Define the key metrics to analyze over the years
metrics_to_analyze = ['Views', 'Subscribers', 'Impressions', 'Shares']
```

```
In [101]:  # Create subplots for each metric
           fig, axes = plt.subplots(len(metrics_to_analyze), 1, figsize=(12, 8), sharex=True)

           # Create subplots for each metric
           fig, axes = plt.subplots(len(metrics_to_analyze), 1, figsize=(12, 8), sharex=True)

           # Plot each metric over the years
           for i, metric in enumerate(metrics_to_analyze):
               ax = axes[i]
               metric_data = df1.groupby('Year')[metric].sum()
               ax.plot(metric_data.index, metric_data.values, marker='o', linestyle='-', label=
               ax.set_ylabel(metric)
               ax.grid()
               ax.legend()

           # Set the common x-axis label and title
           plt.xlabel('Year')
           plt.suptitle('Yearly Trends for Key Metrics')

           # Display the plots
           plt.tight_layout(rect=[0, 0, 1, 0.95])
           plt.show()
```





Yearly Trends for Key Metrics (2016-2024) Views: Peak: Around 2017, reaching approximately 10 million views. Trend: Gradual decline post-2017, stabilizing around 2020 at approximately 5 million views, and remaining constant through 2024.

Subscribers: Peak: Around 2017, reaching approximately 40,000 subscribers. Trend: Decline after 2017, slight stabilization around 2020, and a gradual decrease continuing through 2024, ending at around 20,000 subscribers.

Impressions: Peak: Around 2018, reaching approximately 1 billion impressions. Trend: Decline post-2018, slight stabilization around 2020, and a gradual decrease continuing through 2024, ending at around 500 million impressions.

Shares: Peak: Around 2017, reaching approximately 30,000 shares. Trend: Significant decline post-2017, sharp drop around 2019, and a gradual decrease continuing through 2024, ending at around 5,000 shares.

Analysis 2017-2018 Peaks: The metrics for views, subscribers, impressions, and shares all peaked around 2017-2018. This suggests a period of high engagement and growth for the YouTube channel during these years.

Post-Peak Decline: After the peak years, there is a noticeable decline in all metrics. This could be due to various factors such as changes in content strategy, audience interest, or external competition.

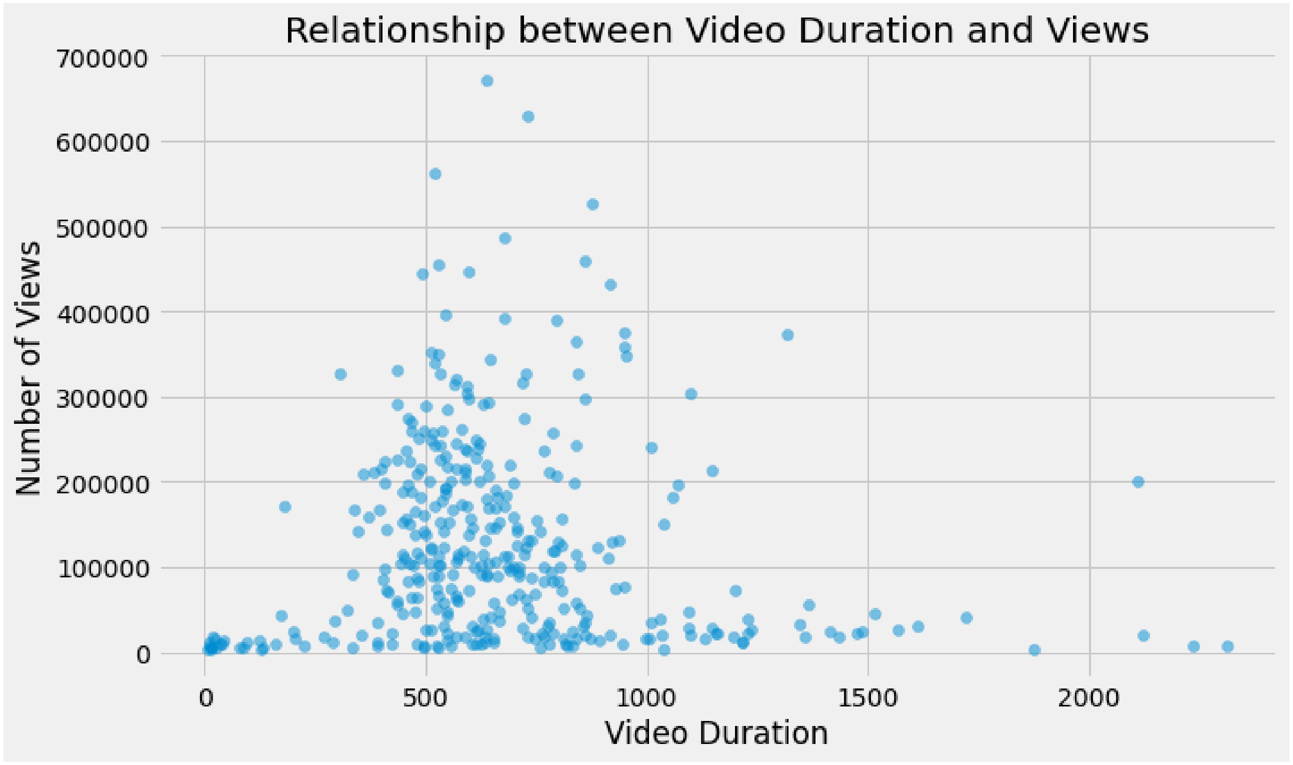
Stabilization Period: Around 2020, there is a slight stabilization in the decline of views, subscribers, and impressions. This indicates that the channel may have found a new equilibrium or consistent audience base.

Shares Decline: The sharp decline in shares around 2019 suggests a significant drop in content virality or audience engagement in sharing the content.

Relationship Between Video Duration and Some Key Metrics

In [102]:

```
# Investigate the relationship between Video Duration and Views
plt.figure(figsize=(10, 6))
# Scatter plot of video duration vs. views
plt.scatter(df1['Video Duration'], df1['Views'], alpha=0.5)
plt.title('Relationship between Video Duration and Views')
plt.xlabel('Video Duration')
plt.ylabel('Number of Views')
plt.grid(True)
```



In [104]:

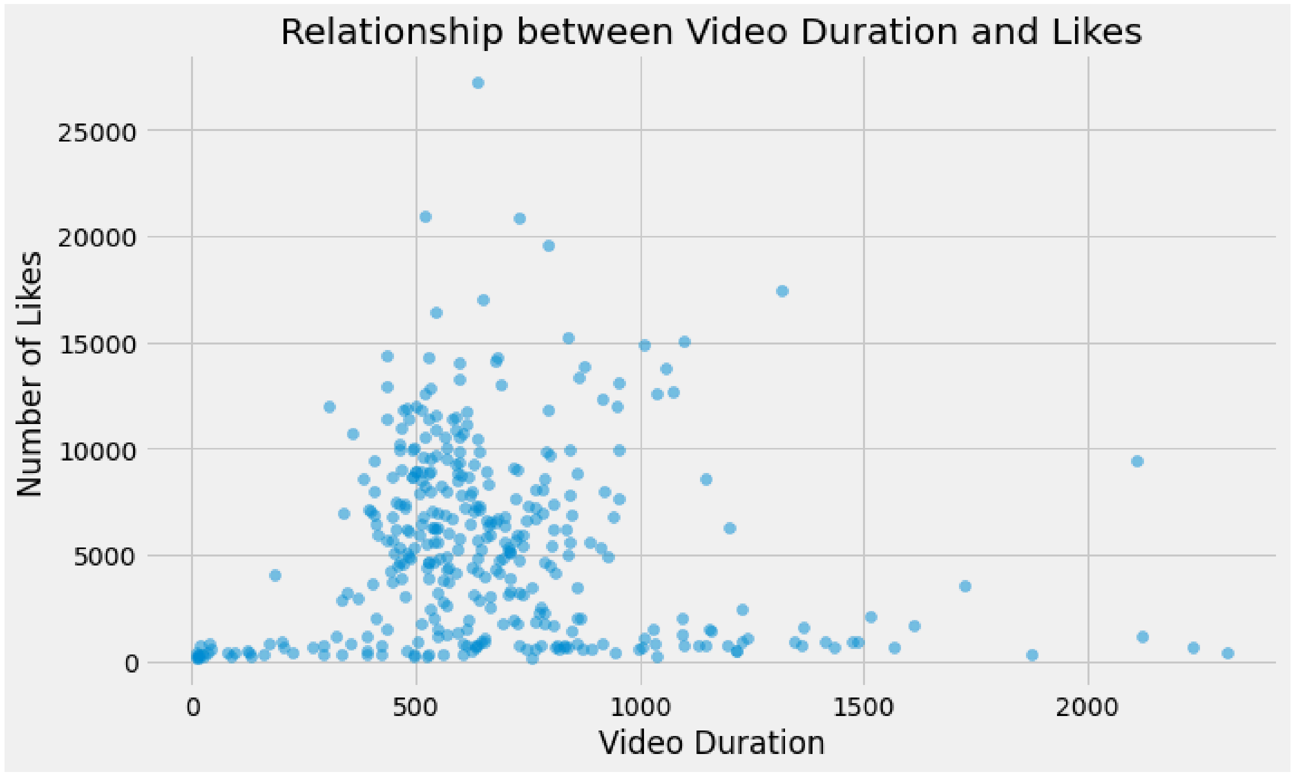
```
correlation_vid_viw = df1['Video Duration'].corr(df1['Views'])
#Print correlation coefficients
print(f'Correlation between Video Duration and Views: {correlation_vid_viw}')
```

```
Correlation between Video Duration and Views: -0.051500282894140256
```

There is a slight negative correlation between video duration and views, with a coefficient of -0.05. This suggests that as video duration increases, the likelihood of resonating with viewers may decrease. While the correlation is not strong, it highlights a potential trend where shorter videos tend to engage audiences more effectively.

This could be due to viewers' limited attention spans or a preference for concise and easily digestible content. Exploring the optimal video length and aligning it with audience preferences might help improve overall viewer engagement and retention.

In [105]:
```python
# Investigate the relationship between Video Duration and Likes
plt.figure(figsize=(10, 6))
# Scatter plot of video duration vs. likes
plt.scatter(df1['Video Duration'], df1['Likes'], alpha=0.5)
plt.title('Relationship between Video Duration and Likes')
plt.xlabel('Video Duration')
plt.ylabel('Number of Likes')
plt.grid(True)
```

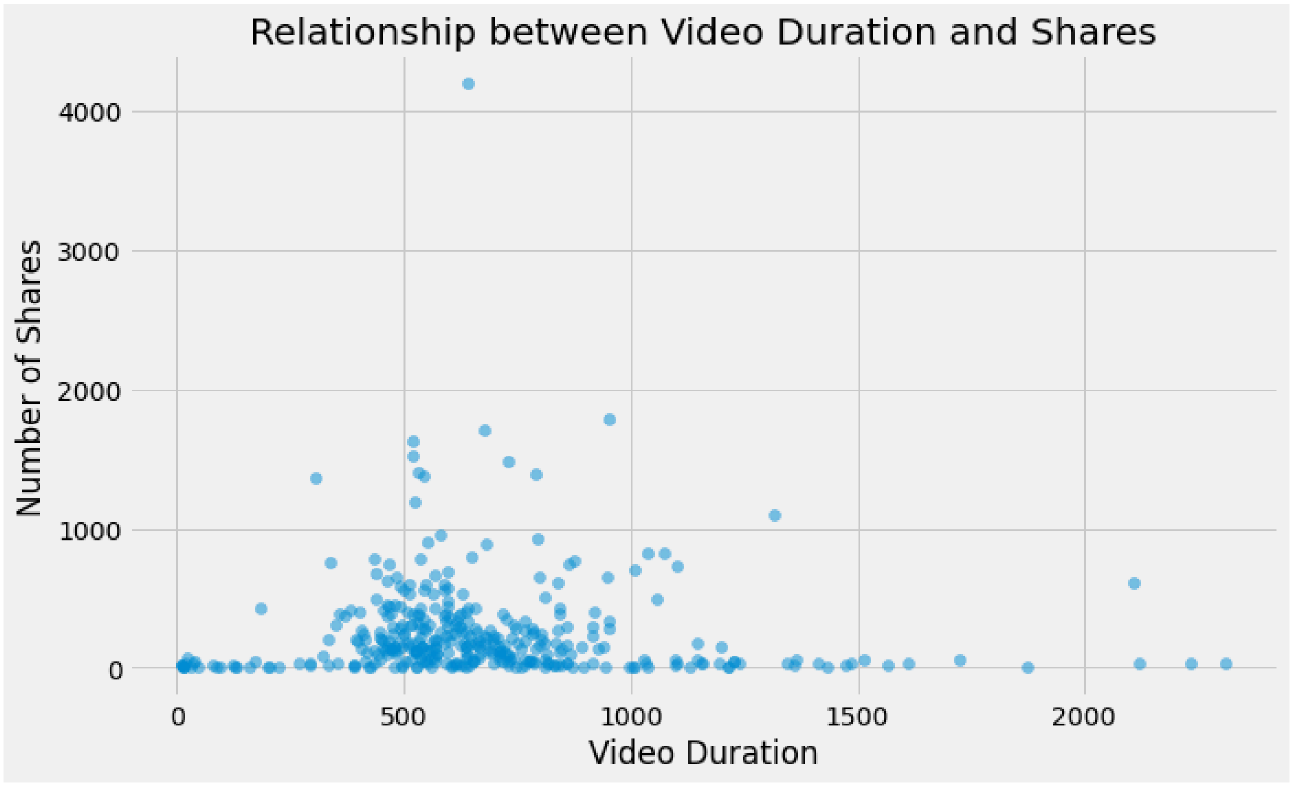


In [133]:
```python
correlation_vid_viw = df1['Video Duration'].corr(df1['Likes'])
#Print correlation coefficients
print(f'Correlation between Video Duration and Likes: {correlation_vid_viw}')
```

```
Correlation between Video Duration and Likes: -0.0311697997552288
```

There is also a slight negative correlation between video duration and likes, with a coefficient of -0.03. This indicates that longer videos may receive fewer likes compared to shorter ones. Although the correlation is weak, it suggests that viewers might prefer shorter content that is more engaging and to the point, leading to higher appreciation in the form of likes.

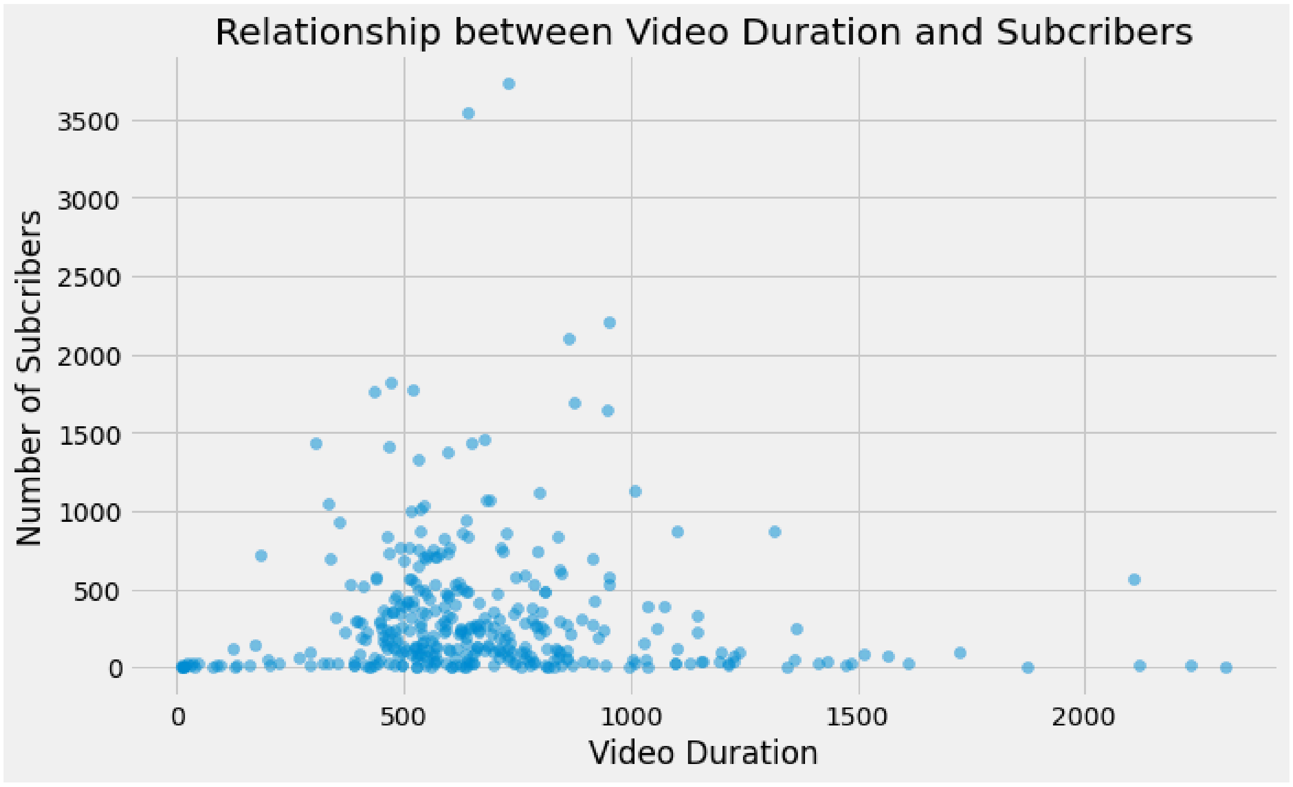
```
In [134]: # Investigate the relationship between Video Duration and Shares
          plt.figure(figsize=(10, 6))
          # Scatter plot of video duration vs. dislikes
          plt.scatter(df1['Video Duration'], df1['Shares'], alpha=0.5)
          plt.title('Relationship between Video Duration and Shares')
          plt.xlabel('Video Duration')
          plt.ylabel('Number of Shares')
          plt.grid(True)
```



```
In [137]: correlation_vid_viw = df1['Video Duration'].corr(df1['Shares'])
          #Print correlation coefficients
          print(f'Correlation between Video Duration and Shares: {correlation_vid_viw}')

          Correlation between Video Duration and Shares: -0.026654528365739272
```

There is a slight negative correlation between video duration and shares, with a coefficient of -0.02. This indicates that longer videos may be shared slightly less frequently than shorter ones, though the relationship is minimal. This trend suggests that viewers might prefer to share concise and easily consumable content, which is often more appealing to a broader audience.

In [117]:
```python
# Investigate the relationship between Video Duration and Subsribers
plt.figure(figsize=(10, 6))
# Scatter plot of video duration vs. subscribers
plt.scatter(df1['Video Duration'], df1['Subscribers'], alpha=0.5)
plt.title('Relationship between Video Duration and Subcribers')
plt.xlabel('Video Duration')
plt.ylabel('Number of Subcribers')
plt.grid(True)
```



In [119]:
```python
correlation_vid_viw = df1['Video Duration'].corr(df1['Subscribers'])
#Print correlation coefficients
print(f'Correlation between Video Duration and Subscribers: {correlation_vid_viw}')
```

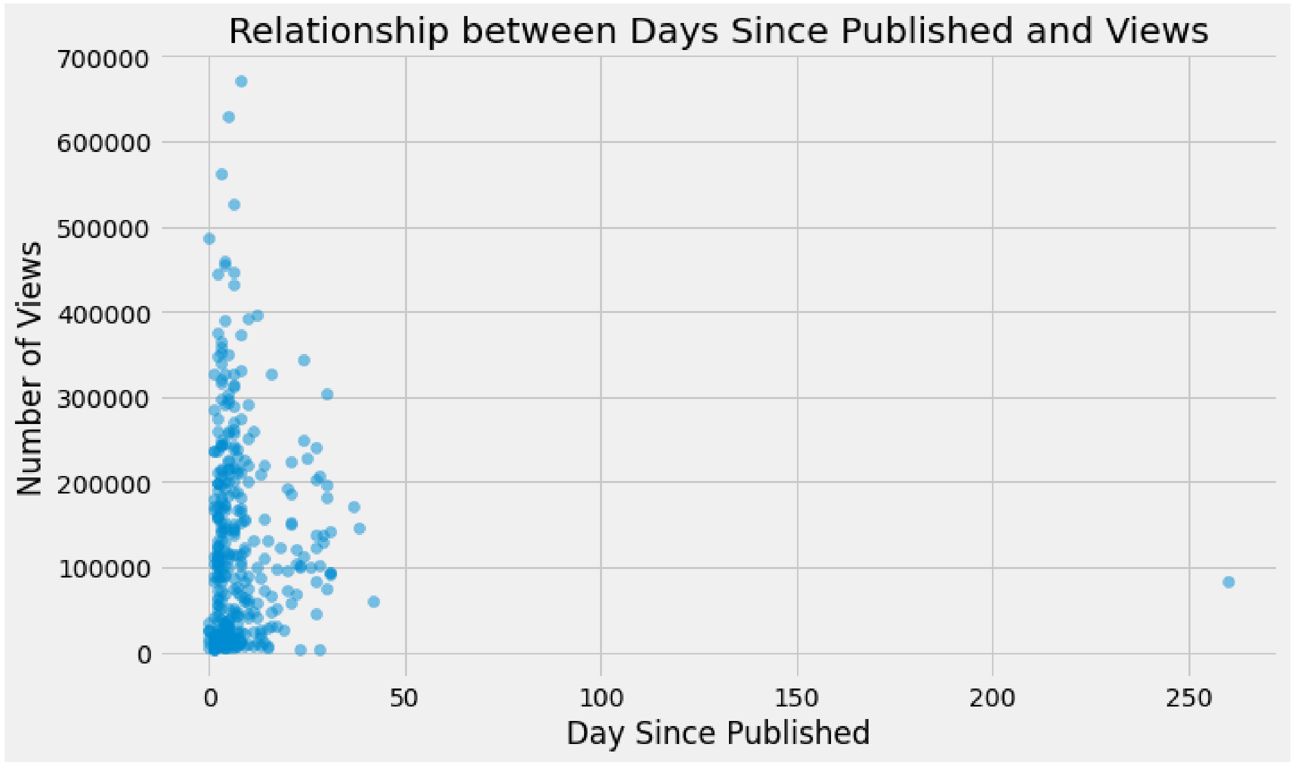
```
Correlation between Video Duration and Subscribers: -0.025583892245784334
```

Based on the relationship between Video duration and Subcribers it shows that this does not have an impact on subcription.

Testing the Relationship between how long a Video was Published and Key Metrics

In [121]:

```
# Investigate the relationship between Days Since Publish and Views
plt.figure(figsize=(10, 6))
# Scatter plot of Days since published vs. Views
plt.scatter(df1['Days Since Publish'], df1['Views'], alpha=0.5)
plt.title('Relationship between Days Since Published and Views')
plt.xlabel('Day Since Published')
plt.ylabel('Number of Views')
plt.grid(True)
```



In [138]:

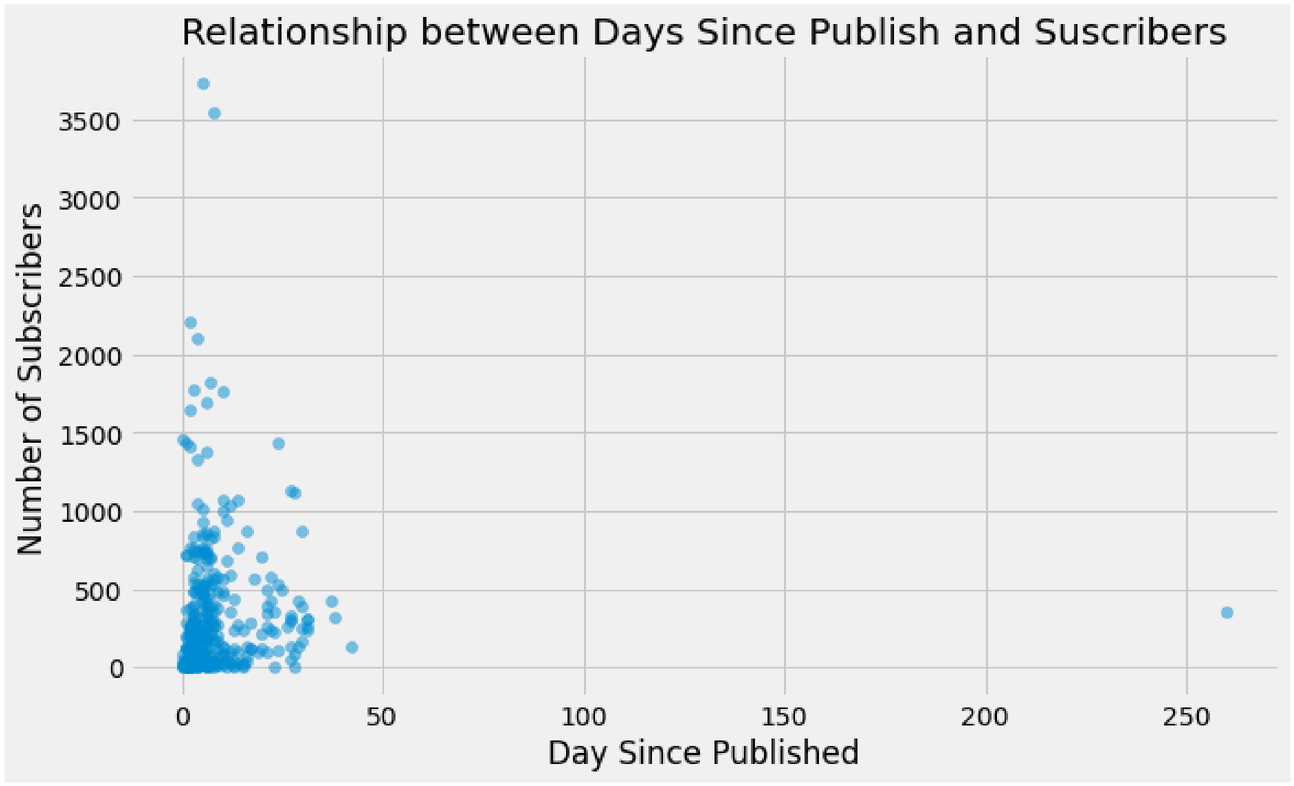
```
correlation_vid_viw = df1['Days Since Publish'].corr(df1['Views'])
#Print correlation coefficients
print(f'Correlation between Days Since Publish and Views: {correlation_vid_viw}')
```

```
Correlation between Days Since Publish and Views: -0.013555213236239119
```

The scatter plot shows the relationship between the number of views and the days since a video was published. With a negative correlation coefficient of -0.01, there is a very weak inverse relationship between these two variables.

Most data points are clustered within the early days after publication, suggesting that videos receive the majority of their views shortly after being uploaded. This trend indicates that audience engagement is highest soon after a video's release, with fewer views accumulating over time.

In [126]:
```
# Investigate the relationship between Days Since Published and Subcribers
plt.figure(figsize=(10, 6))
# Scatter plot of Days since published vs. Subscribers
plt.scatter(df1['Days Since Publish'], df1['Subscribers'], alpha=0.5)
plt.title('Relationship between Days Since Publish and Suscribers')
plt.xlabel('Day Since Published')
plt.ylabel('Number of Subscribers')
plt.grid(True)
```



In [127]:
```
correlation_vid_viw = df1['Days Since Publish'].corr(df1['Subscribers'])
#Print correlation coefficients
print(f'Correlation between Days Since Publish and Subscribers: {correlation_vid_viw
```

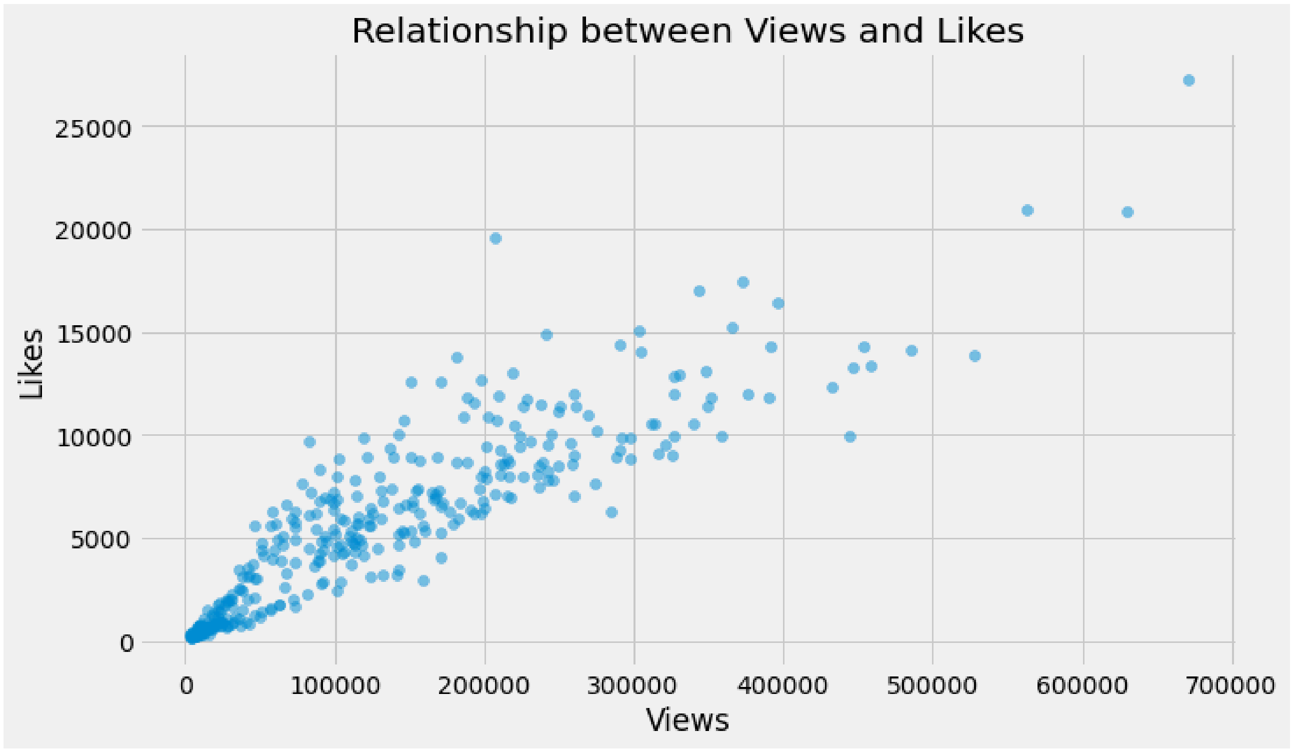
```
Correlation between Days Since Publish and Subscribers: 0.03480800830559202
```

The scatter plot illustrates the relationship between the number of subscribers gained and the days since a video was published. The data points are heavily clustered within the early days of publication, indicating that most new subscribers are acquired shortly after a video's release.

This trend suggests that fresh content plays a key role in attracting new subscribers, reinforcing the importance of consistent and timely uploads. Over time, the number of new subscribers per video declines, implying that older videos contribute less to subscriber growth.

Analyzing the relationship between views and likes will help us understand viewer engagement and determine whether the audience actively interacts with the content when it is posted

In [128]:
```
# Investigate the relationship between Views and Likes
plt.figure(figsize=(10, 6))
# Scatter plot of Days since published vs. Subscribers
plt.scatter(df1['Views'], df1['Likes'], alpha=0.5)
plt.title('Relationship between Views and Likes')
plt.xlabel('Views')
plt.ylabel('Likes')
plt.grid(True)
```



In [129]:
```
correlation_vid_viw = df1['Views'].corr(df1['Likes'])
#Print correlation coefficients
print(f'Correlation between Views and Likes: {correlation_vid_viw}')
```

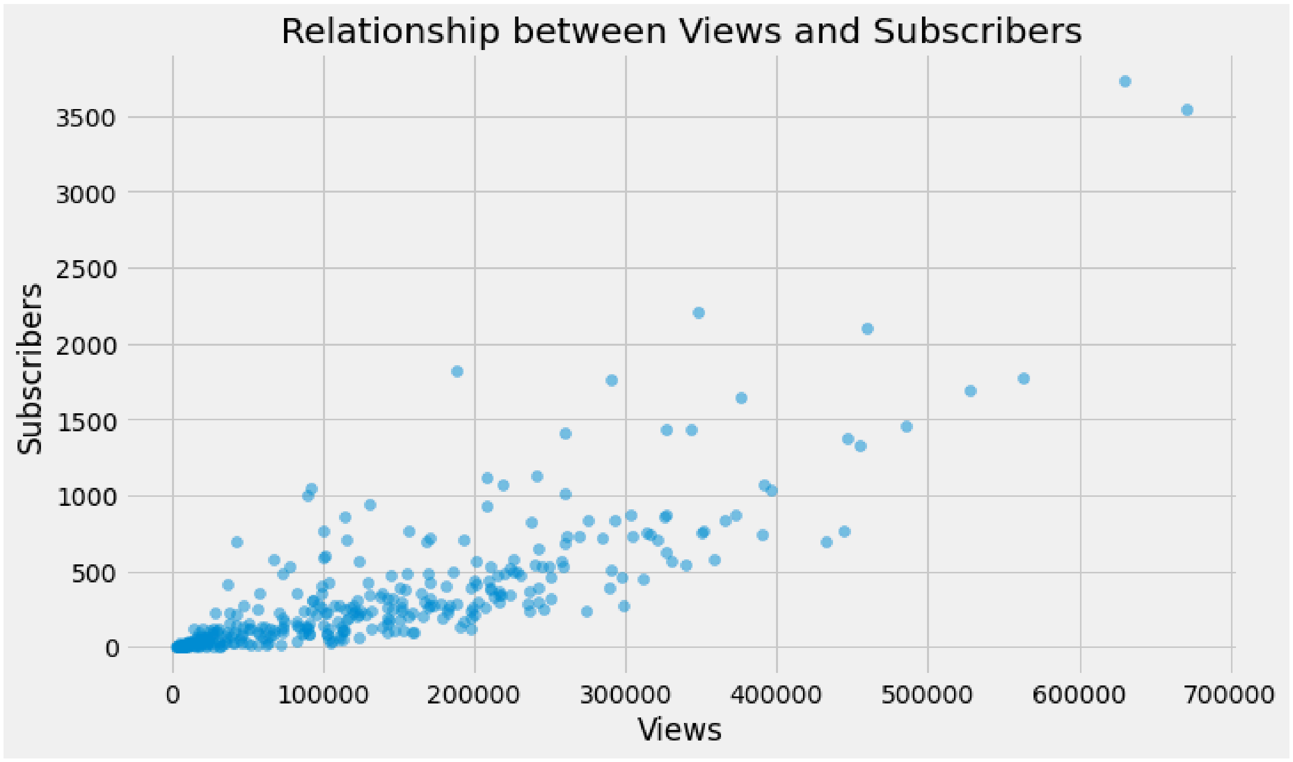
```
Correlation between Views and Likes: 0.8991238497941054
```

The scatter plot displays the relationship between views and likes, with a strong positive correlation coefficient of 0.9. This indicates that as the number of views increases, the number of likes also tends to rise proportionally. The clustering along an upward trend suggests that audience engagement, in terms of likes, scales with video popularity.

However, while the correlation is high, some variability exists, as seen in the spread of data points at higher view counts. This could be due to differences in video content, audience preferences, or external factors such as promotion and timing.

Examining the relationship between views and subscribers provides valuable insights into how effectively content resonates with the audience and influences subscription growth.

In [130]:
```python
# Investigate the relationship between Views and Subscribers
plt.figure(figsize=(10, 6))
# Scatter plot of Views vs. Subscribers
plt.scatter(df1['Views'], df1['Subscribers'], alpha=0.5)
plt.title('Relationship between Views and Subscribers')
plt.xlabel('Views')
plt.ylabel('Subscribers')
plt.grid(True)
```



In [132]:
```python
correlation_vid_viw = df1['Views'].corr(df1['Subscribers'])
#Print correlation coefficients
print(f'Correlation between Views and Subscribers: {correlation_vid_viw}')
```

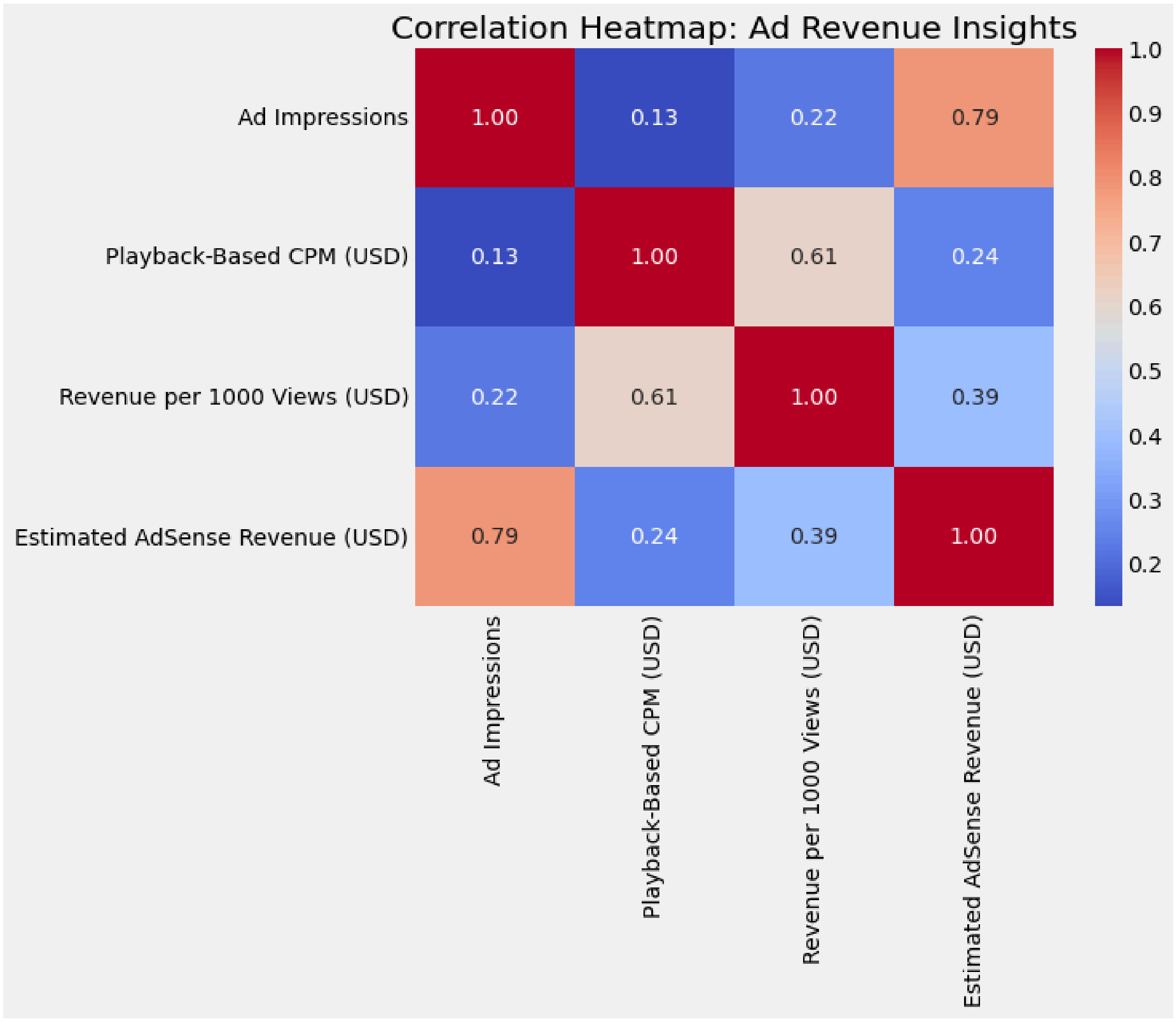
```
Correlation between Views and Subscribers: 0.802406400506519
```

The scatter plot illustrates the relationship between views and subscribers, with a strong positive correlation coefficient of 0.8. This suggests that as the number of views increases, the number of subscribers tends to rise in a similar fashion. The clustering of data points along an upward trend implies that audience growth, in terms of subscribers, is linked to the popularity of the videos.

Although the correlation is strong, there is noticeable variability, particularly at higher view counts. This suggests that factors such as the type of content, audience engagement, and external elements like promotional activities could play a role in the variations in subscriber growth.

In [ ]:

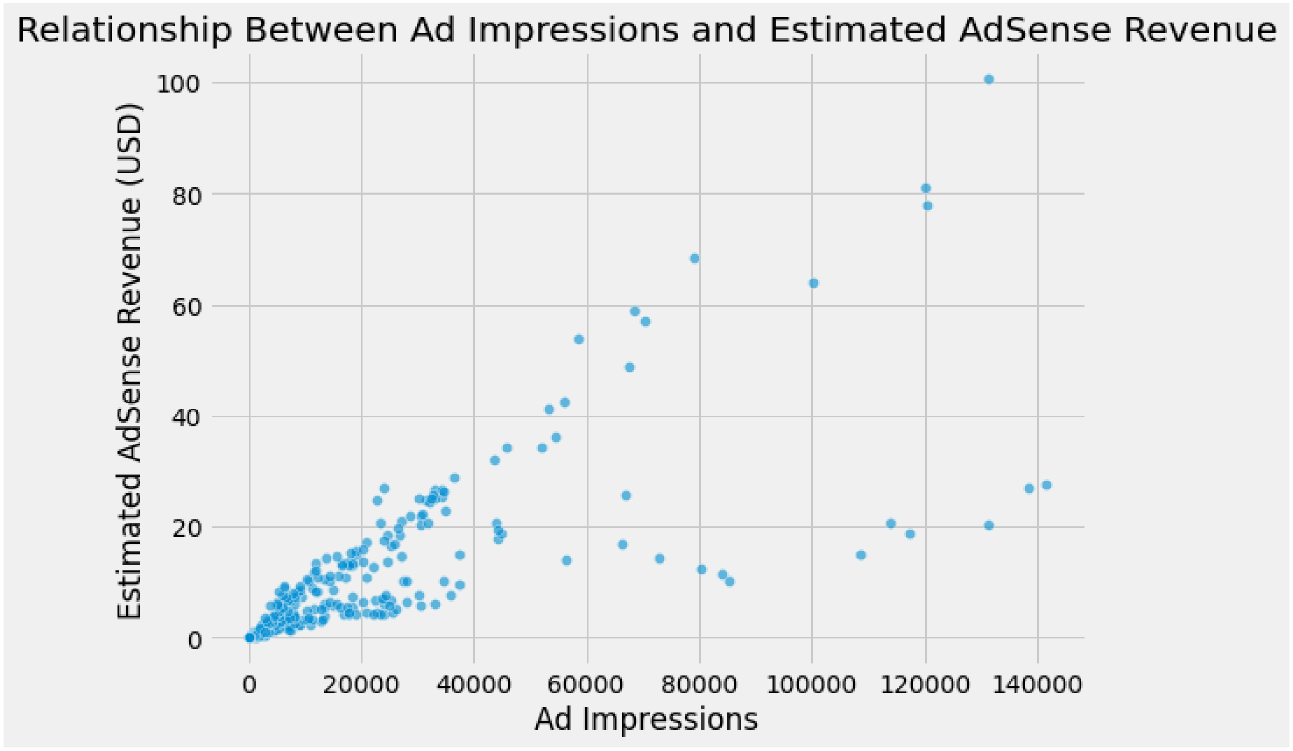
```
In [139]:  # Correlation heatmap for Ad Revenue Insights
           plt.figure(figsize=(8, 6))
           sns.heatmap(df1[["Ad Impressions", "Playback-Based CPM (USD)",
                            "Revenue per 1000 Views (USD)", "Estimated AdSense Revenue (USD)"]]
                       annot=True, cmap="coolwarm", fmt=".2f")
           plt.title("Correlation Heatmap: Ad Revenue Insights")
           plt.show()
```



The correlation heatmap provides key insights into the relationships between ad revenue metrics. Ad Impressions show a strong positive correlation (0.79) with Estimated AdSense Revenue, indicating that higher impressions significantly contribute to total revenue. Playback-Based CPM (USD) has a moderate correlation (0.61) with Revenue per 1000 Views, suggesting that as CPM increases, revenue per 1000 views also rises. However, CPM has a weaker correlation (0.24) with total AdSense revenue, implying that while CPM affects revenue, other factors like impressions play a more dominant role. The relationship between Revenue per 1000 Views and Estimated AdSense Revenue (0.39) suggests that while revenue per 1000 views contributes to overall earnings, it is not the sole determinant. Overall, the heatmap highlights that Ad Impressions are the strongest predictor of total revenue, while CPM and revenue per 1000 views have more moderate influences.

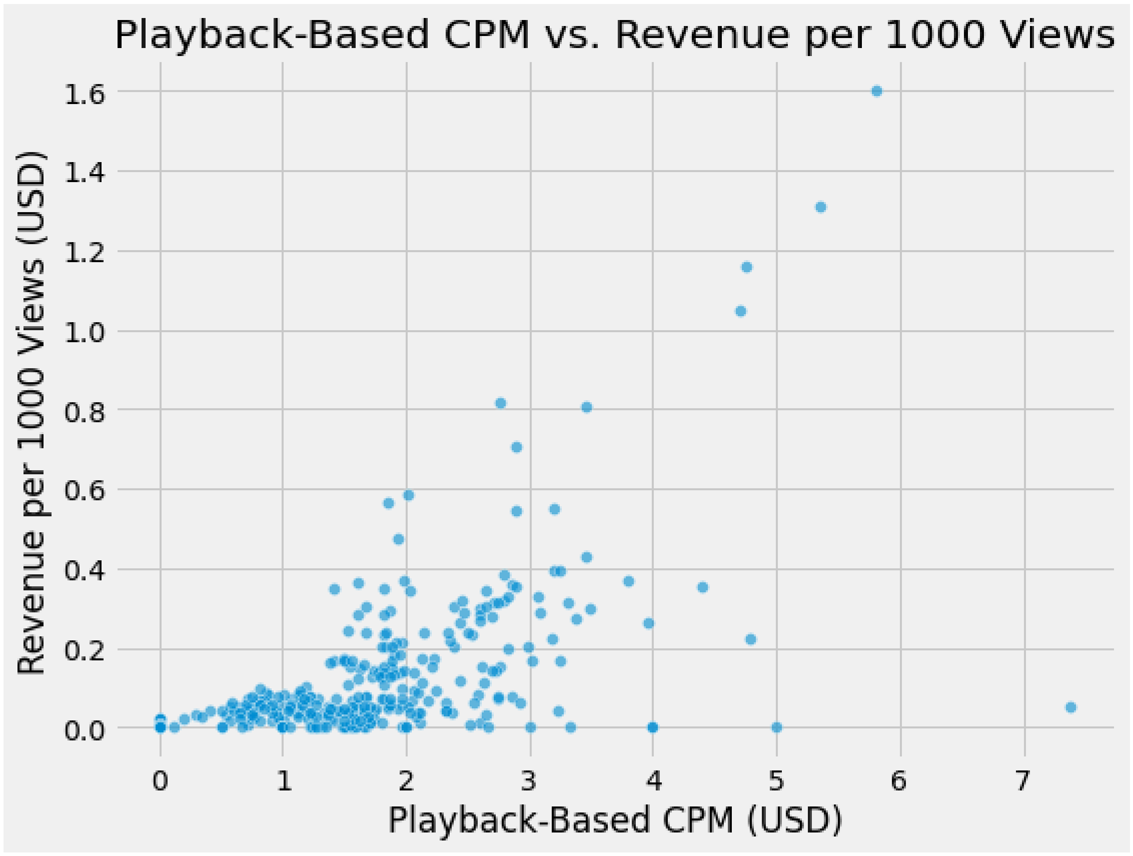
In [140]:

```
# Scatter Plot: Relationship between Ad Impressions and Estimated AdSense Revenue
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df1, x="Ad Impressions", y="Estimated AdSense Revenue (USD)", al
plt.title("Relationship Between Ad Impressions and Estimated AdSense Revenue")
plt.xlabel("Ad Impressions")
plt.ylabel("Estimated AdSense Revenue (USD)")
plt.show()
```



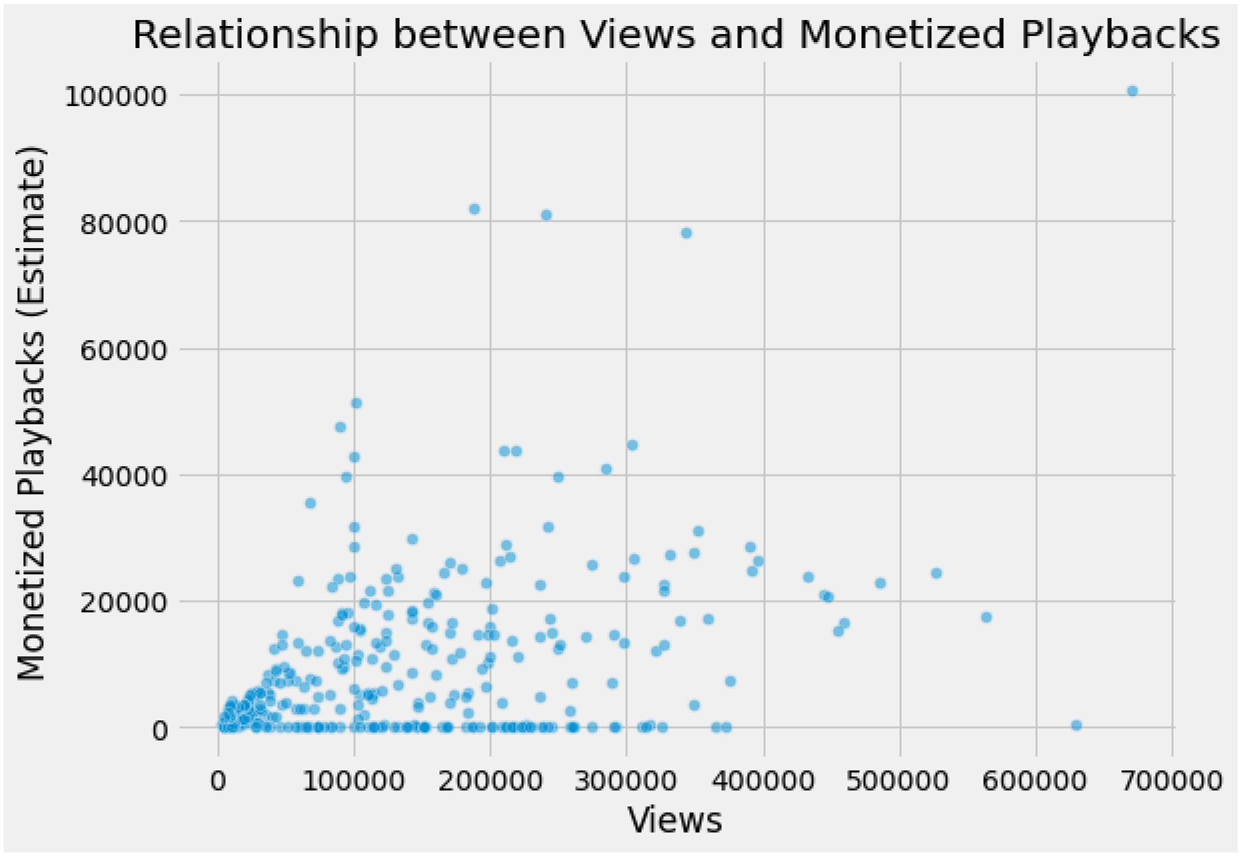
In [ ]:

```
In [141]:  # Scatter Plot: Playback-Based CPM vs. Revenue per 1000 Views
           plt.figure(figsize=(8, 6))
           sns.scatterplot(data=df1, x="Playback-Based CPM (USD)", y="Revenue per 1000 Views (U
           plt.title("Playback-Based CPM vs. Revenue per 1000 Views")
           plt.xlabel("Playback-Based CPM (USD)")
           plt.ylabel("Revenue per 1000 Views (USD)")
           plt.show()
```



```
In [ ]:
```

In [145]:
```
# Scatter plot to visualize relationship
plt.figure(figsize=(8, 6))
sns.scatterplot(x=df1["Views"], y=df1["Monetized Playbacks (Estimate)"], alpha=0.5)
plt.xlabel("Views")
plt.ylabel("Monetized Playbacks (Estimate)")
plt.title("Relationship between Views and Monetized Playbacks")
plt.show()
```



The scatter plot visualizes the relationship between "Views" and "Monetized Playbacks (Estimate)." The correlation coefficient of 0.42 suggests a moderate positive relationship, meaning that as views increase, monetized playbacks tend to increase as well, but not in a strongly linear fashion. The data points are widely scattered, indicating variability in how views translate into monetized playbacks. Some videos with high views have relatively low monetized playbacks, while others show a stronger conversion.

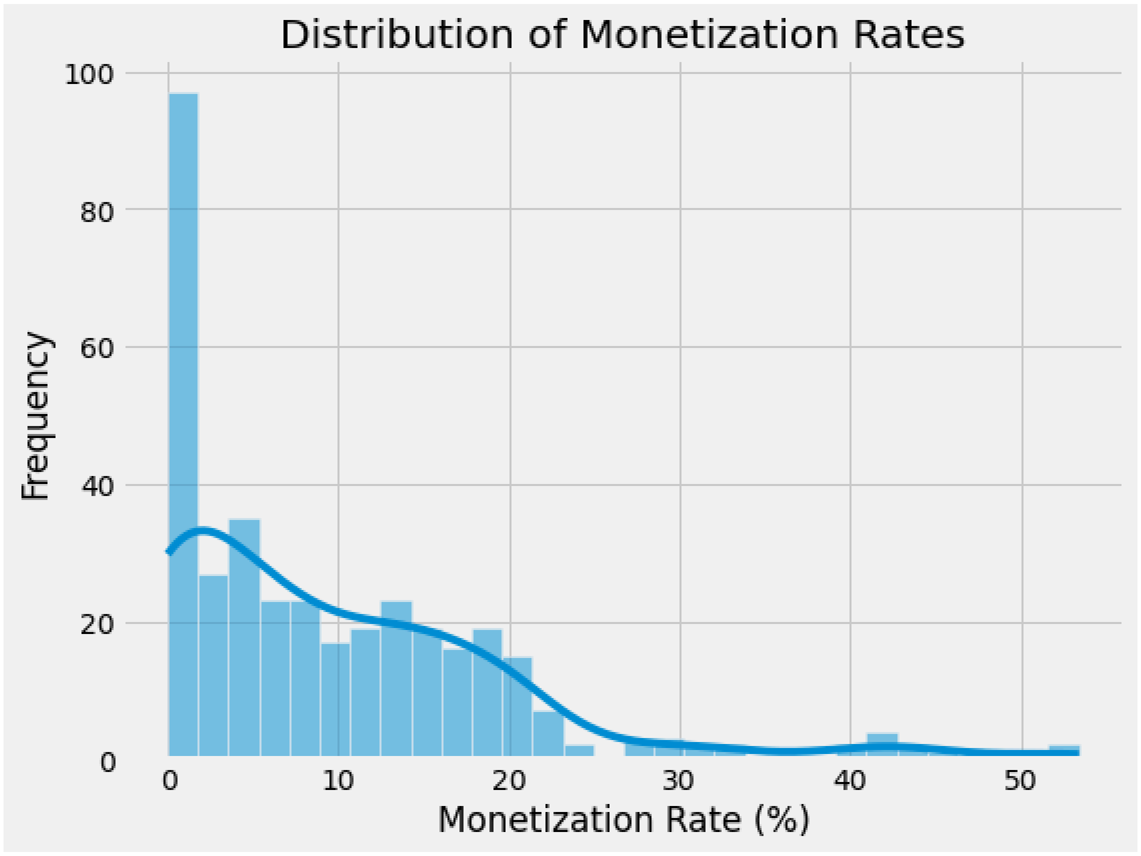
In [146]:
```
# Calculate the correlation coefficient
correlation = df1["Views"].corr(df1["Monetized Playbacks (Estimate)"])
print(f"Correlation Coefficient: {correlation:.2f}")
```

```
Correlation Coefficient: 0.42
```

In [148]:
```
# Calculate the monetization rate (percentage of views that are monetized)
df1["Monetization Rate (%)"] = (df1["Monetized Playbacks (Estimate)"] / df1["Views"]
# Summary statistics for monetization rate
monetization_summary = df1["Monetization Rate (%)"].describe()
print("\nMonetization Rate Statistics:\n", monetization_summary)
```

```
Monetization Rate Statistics:
 count    364.000000
mean       9.701728
std       10.125434
min        0.000000
25%        1.171042
50%        7.034468
75%       14.928226
max       53.413826
Name: Monetization Rate (%), dtype: float64
```

In [149]:
```
# Histogram of monetization rates
plt.figure(figsize=(8, 6))
sns.histplot(df1["Monetization Rate (%)"], bins=30, kde=True)
plt.xlabel("Monetization Rate (%)")
plt.ylabel("Frequency")
plt.title("Distribution of Monetization Rates")
plt.show()
```



The histogram illustrates the distribution of monetization rates (%) across videos. The data is heavily skewed towards the lower end, with a significant number of videos having a very low monetization rate close to 0%. This suggests that many views do not convert into monetized playbacks. However, there are still some videos achieving higher monetization rates, though these are much less frequent. The presence of a long tail indicates that while some videos have strong monetization performance, the majority experience

Regression Model

In [317]:
```
#create a list of variables to test the relationship against estimated revenue
df2 = df1[['Estimated Revenue (USD)','Video Thumbnail CTR (%)','Estimated AdSense Rev
```

In [318]:
```
df2 = df2.reset_index(drop=True)
```

In [319]:
```
# Define the dependent variable (Y) and independent variables (X)
Y = df2['Estimated Revenue (USD)']
X = df2.drop(columns=['Estimated Revenue (USD)'])
```

```
In [320]:  # Add a constant term to the independent variables
           X = sm.add_constant(X)

           # Fit the regression model
           model = sm.OLS(Y, X).fit()

           # Print the model summary
           print(model.summary())
```

```
                            OLS Regression Results
==================================================================================
Dep. Variable:     Estimated Revenue (USD)   R-squared:                     0.999
Model:                                 OLS   Adj. R-squared:                0.999
Method:                      Least Squares   F-statistic:               9.164e+04
Date:                     Wed, 29 Jan 2025   Prob (F-statistic):             0.00
Time:                             16:12:04   Log-Likelihood:              -158.79
No. Observations:                      364   AIC:                           329.6
Df Residuals:                          358   BIC:                           353.0
Df Model:                                5
Covariance Type:                 nonrobust
==================================================================================
================
                                      coef    std err          t      P>|t|
[0.025      0.975]
----------------------------------------------------------------------------------
----------------
const                               0.0197      0.067      0.293      0.769      -
0.112       0.151
Video Thumbnail CTR (%)             0.0169      0.007      2.400      0.017
0.003       0.031
Estimated AdSense Revenue (USD)     0.1466      0.015      9.732      0.000
0.117       0.176
Watch Page Ads Revenue (USD)        0.8891      0.015     60.278      0.000
0.860       0.918
CPM (USD)                           0.0701      0.030      2.324      0.021
0.011       0.129
Revenue per 1000 Views (USD)       -0.2757      0.130     -2.115      0.035      -
0.532      -0.019
==============================================================================
Omnibus:                      243.190   Durbin-Watson:                   1.339
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2589.411
Skew:                           2.725   Prob(JB):                         0.00
Kurtosis:                      14.875   Cond. No.                         146.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly sp
ecified.
```

R-squared (0.999): The model explains 99.9% of the variance in the estimated revenue, indicating a very strong fit between the independent variables and the dependent variable (Estimated Revenue). This suggests that the model does an excellent job of predicting the revenue.

The OLS regression results show a strong model fit, with an R-squared of 0.999, indicating that the independent variables explain nearly all of the variance in estimated revenue. Significant factors include video thumbnail click-through rate (CTR), AdSense revenue, Watch Page Ads revenue, CPM, and revenue per 1000 views, with AdSense revenue and Watch Page Ads revenue having the most substantial positive effects on estimated revenue. Video thumbnail CTR and CPM also have positive effects, while revenue per

1000 views shows a negative relationship with revenue. The model indicates some issues with residual normality and potential autocorrelation, but overall, it suggests that factors related to ads and views strongly

```
In [ ]:
```

Time Series Analyis for Subscribers Using Facebook prophet

```
In [324]: df3 = df[['Video Publish Time', 'Subscribers']]
          df3.columns = ['ds', 'y']  # Rename columns to 'ds' and 'y' for Prophet
```

```
In [325]: df3.head()
```

Out[325]:

	ds	y
0	2016-06-02	51.0
1	2016-06-10	33.0
2	2016-06-14	8.0
3	2016-06-29	2.0
4	2016-07-01	28.0

```
In [328]: #convert the ds column to date format
          df3['ds'] = pd.to_datetime(df3['ds']).dt.date
```

```
In [329]: # Initialize the Prophet model
          model = Prophet()
```

```
In [330]: # Fit the model to your data
          model.fit(df3)

          16:58:16 - cmdstanpy - INFO - Chain [1] start processing
          16:58:17 - cmdstanpy - INFO - Chain [1] done processing

Out[330]: <prophet.forecaster.Prophet at 0x1e2ba9eceb0>
```

```
In [331]: # Create a future dataframe for forecasting
          future = model.make_future_dataframe(periods=365)
```

```
In [332]: # Make predictions
          forecast = model.predict(future)
```

In [333]:
```
forecast.tail(5)
```

Out[333]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terms
717	2025-10-14	-27.799199	-378.583943	611.220221	-49.593841	-6.526004	139.773872	139.
718	2025-10-15	-28.057588	-452.575301	538.887033	-49.935880	-6.733438	66.138055	66.
719	2025-10-16	-28.315978	-399.430671	580.870685	-50.314928	-6.934221	128.088309	128.
720	2025-10-17	-28.574367	-195.721699	721.480822	-50.701980	-7.081552	302.589072	302.
721	2025-10-18	-28.832756	-332.537125	666.426449	-51.088678	-7.228884	155.890610	155.

In [334]:
```
forecast[['ds', 'yhat']]
```

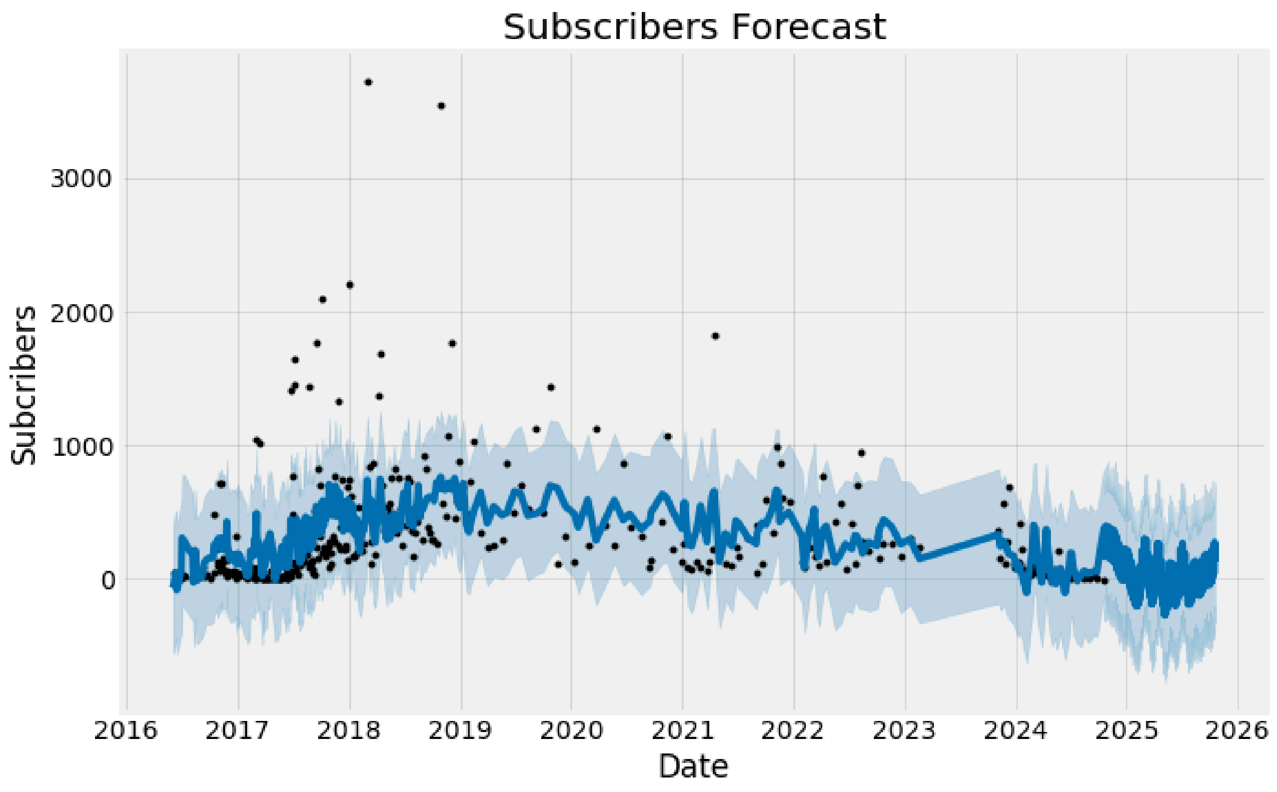
Out[334]:

	ds	yhat
0	2016-06-02	-67.562078
1	2016-06-10	45.138507
2	2016-06-14	-82.812245
3	2016-06-29	67.646835
4	2016-07-01	306.039496
...	...	...
717	2025-10-14	111.974672
718	2025-10-15	38.080467
719	2025-10-16	99.772331
720	2025-10-17	274.014705
721	2025-10-18	127.057853

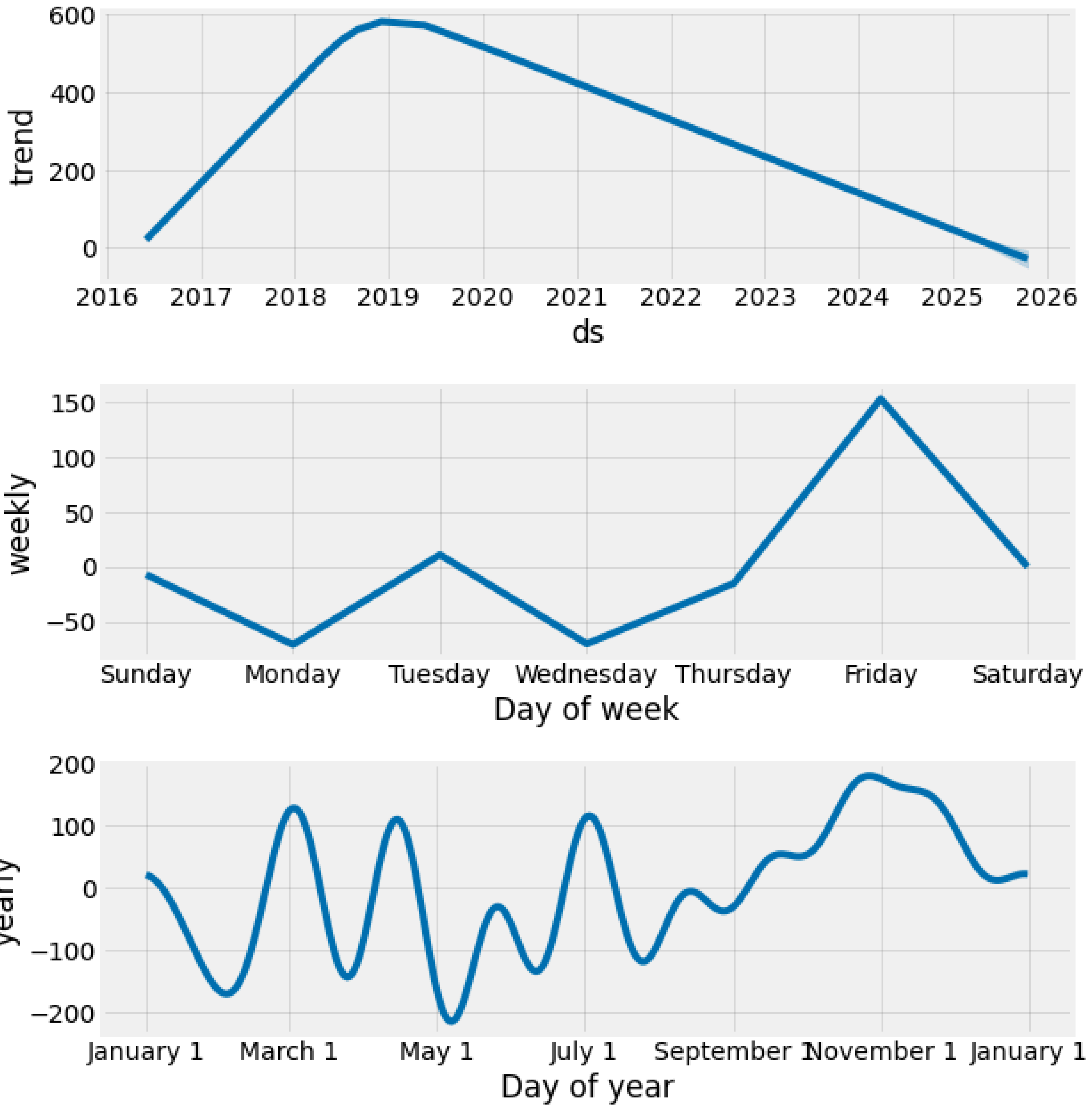
722 rows × 2 columns

In [335]:

```
# Plot the forecast
fig = model.plot(forecast)
plt.title('Subscribers Forecast')
plt.xlabel('Date')
plt.ylabel('Subcribers')
plt.show()
```



In [336]:
```
fig2 = model.plot_components(forecast)
```



In [ ]:

In [337]:
```
#removing the outliers from the dataset and comparing the result
Q1 = df1['Subscribers'].quantile(0.25)
Q3 = df1['Subscribers'].quantile(0.75)
IQR = Q3 - Q1
```

In [338]:
```
lower_bound = Q1 - 0.2 * IQR
upper_bound = Q3 + 0.2 * IQR
```

In [339]:
```
df_no_outliers = df[(df['Subscribers'] >= lower_bound) & (df['Subscribers'] <= upper_
```

In [341]:
```
df['sub_no_outliers'] = df_no_outliers['Subscribers']
```

In [343]:
```
df3 = df[['Video Publish Time', 'sub_no_outliers']].dropna()
df3.columns = ['ds', 'y']  # Rename columns to 'ds' and 'y' for Prophet
```

In [344]:
```
df3.head()
```

Out[344]:

	ds	y
0	2016-06-02	51.0
1	2016-06-10	33.0
2	2016-06-14	8.0
3	2016-06-29	2.0
4	2016-07-01	28.0

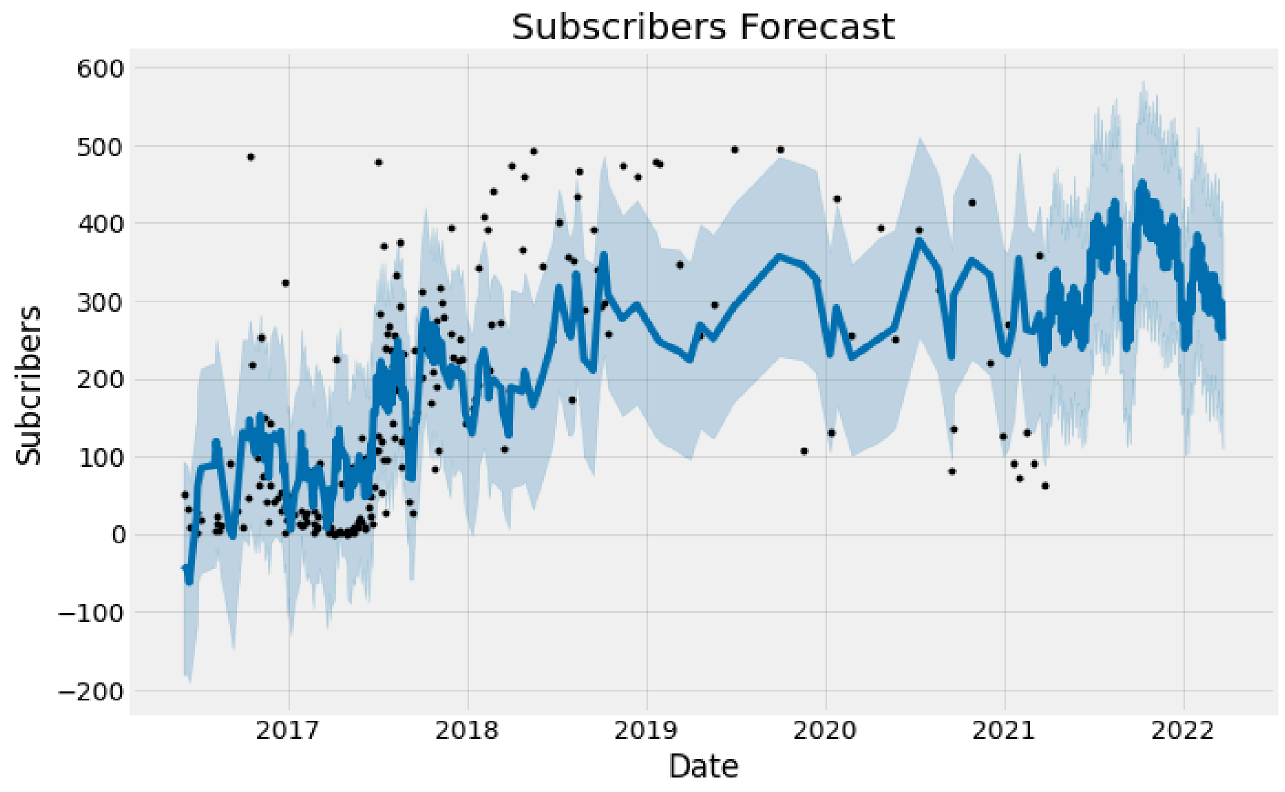
In [345]:
```
#convert the ds column to date format
df3['ds'] = pd.to_datetime(df3['ds']).dt.date
```

In [347]:
```
train = df3.iloc[:len(df3)-365]
test = df3.iloc[len(df3)-365:]
```

In [348]:
```
m = Prophet()
m.fit(train)
future = m.make_future_dataframe(periods = 365)
forecast = m.predict(future)
```

```
17:13:12 - cmdstanpy - INFO - Chain [1] start processing
17:13:12 - cmdstanpy - INFO - Chain [1] done processing
```

In [349]:
```
# Plot the forecast
fig = m.plot(forecast)
plt.title('Subscribers Forecast')
plt.xlabel('Date')
plt.ylabel('Subcribers')
plt.show()
```



The FB Prophet model's prediction for subscriber growth reveals distinct temporal patterns. The overall trend suggests a rapid increase in subscribers, peaking around 2019, followed by a gradual decline projected through 2026. Weekly patterns indicate that subscriber activity is lowest on Mondays and highest on Fridays, with moderate fluctuations throughout the week. Seasonal variations show peak subscriber growth around January 1, March 1, and November 1, while declines typically occur around May 1 and July 1. These insights highlight key periods of engagement, which can inform strategic decisions for optimizing content and marketing efforts.

In [ ]:
```

```

Key Insights from YouTube Channel Analysis Peak Engagement Months July and October had the highest view counts. Further analysis is needed to determine whether seasonal trends, content strategy, or external factors contributed.

October also excelled in subscriber growth, impressions, shares, and likes, suggesting strong audience engagement. November and December showed strong subscriber growth, possibly due to holiday season content consumption.

Engagement by Day of the Week Fridays consistently perform well across views, subscribers, impressions, and likes. Sundays and Tuesdays also show strong engagement, particularly in shares and impressions.

Monday exhibits unexpected engagement in likes, possibly due to users catching up on content.

Long-Term Trends (2016-2024) 2017-2018 marked peak engagement (views, subscribers, impressions, and shares).

Post-2018 decline, with stabilization around 2020.

Sharp drop in shares after 2019, indicating reduced content virality.

Video Performance Patterns Shorter videos tend to receive more likes and shares, suggesting that concise content is more engaging. Videos gain the majority of views and subscribers shortly after release, reinforcing the importance of frequent uploads. High correlation between views and likes (0.9) & views and subscribers (0.8), confirming that video popularity drives engagement.

Monetization Insights Ad Impressions strongly predict AdSense revenue (correlation: 0.79).

Moderate correlation (0.42) between views and monetized playbacks, suggesting variability in ad engagement.

Most videos have low monetization rates, highlighting an opportunity to optimize ad strategies.

Predictive Modeling Insights Regression model ($R^2 = 0.999$) strongly predicts estimated revenue, with key drivers being AdSense revenue, Watch Page Ads revenue, CPM, and thumbnail CTR. FB Prophet model shows peak subscriber growth around January 1, March 1, and November 1, with declines in May and July.

```
In [ ]:
```

RECOMMENDATION

From these, it would therefore seem that there is strong optimization of content scheduling and release strategies. The findings show that Friday and Sunday receive the most views, impressions, and subscriber growth, hence uploading and promoting videos towards the week's end would yield a better reach and engagement. October proved to be the strongest month by many measures and was likely to be influenced by seasonal trends, compelling content, or external events. In view of this, replicating such success will involve analyzing the type of videos that performed well during October and then strategically planning for similar themes or promotional efforts in future high-performing months, such as November and December.

The decline in key metrics since 2017, including views, subscribers, impressions, and shares, would suggest a shift in audience behavior, competition, or content strategy effectiveness. Considering the above slump in graph growth, creators should go back to their previous successful videos, analyze viewer responses, and make more polished content catering to those types that did well. The stagnation of views around 2020 reflects a very loyal viewer base; further engagement can be done by incorporating interactive content, community involvement, and new content formats that could appeal to emerging viewer preferences.

Video length and engagement metric analyses showed the slight favorability of shorter content, which happens to attract more likes and shares. However, longer content is still applicable, provided it would push creators to innovate and come up with formats capable of easily catching the viewers' attention in an extremely short time. Besides, knowing how to build anticipation, such as through scheduled premiers, notifications, and encouragement of early interactions, will give channels an edge in knowing that a majority of their views and subscribers happen just right after publishing a video.

Monetization insights really drive home the need for optimization of ad impressions and engagement in driving revenues. Since ad impressions strongly correlate with estimated AdSense revenue, lifting video findability through captivating thumbnails, higher click-through rates, and strategic keyword optimization improves visibility. Moreover, understanding the fluctuations in CPM and revenue per 1000 views will help refine monetization strategies further, keeping the content appealing to advertisers while making it engaging for the audience.

Finally, predictive modeling using the FB Prophet model shows key periods for subscriber growth, peaking in the months of January, March, and November. Content creators can time major campaigns, special releases, or promotional efforts to start in these high-growth months in order to capitalize on audience activity. Additionally, the weekly trend shows that Fridays generally see the best performance, further emphasizing the need to focus content drops and promotional activities on this day of the week. By

```
In [ ]:
```

```
In [ ]:
```

```
In [ ]:
```