In [1]:
```
# Import necessary libraries
import os
import pandas as pd   # For handling the data
import numpy as np    # For calculations
import matplotlib.pyplot as plt  # For making charts
import seaborn as sns  # For making fancy charts
import matplotlib.ticker as mticker
```

In [2]:
```
current_directory = os.getcwd()
print("Current Directory:", current_directory)
```

```
Current Directory: /Users/amarachiordor/Downloads/Task 2
```

In [3]:
```
files = os.listdir()
print("Files in Directory:", files)
```

```
Files in Directory: ['Stage 2 Task - Exploratory Data Analysis.ipynb', 'ma
rketing_campaign_dataset.xlsx', '.ipynb_checkpoints']
```

In [4]:
```
# Load Excel file
df = pd.read_excel(r"/Users/amarachiordor/Downloads/Task 2/marketing_camp
```

In [5]:
```
# Display the first few rows
print(df.head())
```

```
   Campaign_ID              Company Campaign_Type Target_Audience Duration
\
0            1  Innovate Industries         Email       Men 18-24  30 days
1            2       NexGen Systems         Email     Women 35-44  60 days
2            3    Alpha Innovations    Influencer       Men 25-34  30 days
3            4   DataTech Solutions       Display        All Ages  60 days
4            5       NexGen Systems         Email       Men 25-34  15 days

  Channel_Used  Conversion_Rate  Acquisition_Cost   ROI     Location  \
0   Google Ads             0.04             16174  6.29      Chicago
1   Google Ads             0.12             11566  5.61     New York
2      YouTube             0.07             10200  7.18  Los Angeles
3      YouTube             0.11             12724  5.55        Miami
4      YouTube             0.05             16452  6.50  Los Angeles

                 Date  Clicks  Impressions  Engagement_Score  \
0 2021-01-01 00:00:00     506         1922                 6
1 2021-02-01 00:00:00     116         7523                 7
2 2021-03-01 00:00:00     584         7698                 1
3 2021-04-01 00:00:00     217         1820                 7
4 2021-05-01 00:00:00     379         4201                 3

      Customer_Segment
0    Health & Wellness
1          Fashionistas
2  Outdoor Adventurers
3    Health & Wellness
4    Health & Wellness
```

In [6]:
```
#Check Column Names and Data Types
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200005 entries, 0 to 200004
Data columns (total 15 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   Campaign_ID       200005 non-null  int64  
 1   Company           200005 non-null  object 
 2   Campaign_Type     200005 non-null  object 
 3   Target_Audience   200005 non-null  object 
 4   Duration          200005 non-null  object 
 5   Channel_Used      200005 non-null  object 
 6   Conversion_Rate   200005 non-null  float64
 7   Acquisition_Cost  200005 non-null  int64  
 8   ROI               200005 non-null  float64
 9   Location          200005 non-null  object 
 10  Date              200005 non-null  object 
 11  Clicks            200005 non-null  int64  
 12  Impressions       200005 non-null  int64  
 13  Engagement_Score  200005 non-null  int64  
 14  Customer_Segment  200005 non-null  object 
dtypes: float64(2), int64(5), object(8)
memory usage: 22.9+ MB
```

In [7]:
```
# Check Shape of the Data
df.shape
```

Out[7]:
```
(200005, 15)
```

In [8]:
```
# Check for Duplicates
df.duplicated().sum()
```

Out[8]:
```
0
```

In [9]:
```
# Summary of Numerical Columns
df.describe()
```

Out[9]:

	Campaign_ID	Conversion_Rate	Acquisition_Cost	ROI	
count	200005.000000	200005.000000	200005.000000	200005.000000	200005
mean	100003.000000	0.080069	12504.441794	5.002416	549
std	57736.614632	0.040602	4337.663210	1.734485	260
min	1.000000	0.010000	5000.000000	2.000000	100
25%	50002.000000	0.050000	8740.000000	3.500000	325
50%	100003.000000	0.080000	12497.000000	5.010000	550
75%	150004.000000	0.120000	16264.000000	6.510000	775
max	200005.000000	0.150000	20000.000000	8.000000	1000

In [10]:
```
# Summary of Categorical Columns

df.describe(include='object')
```

Out[10]:

	Company	Campaign_Type	Target_Audience	Duration	Channel_Used	Loc
count	200005	200005	200005	200005	200005	20
unique	5	5	5	4	6	
top	TechCorp	Influencer	Men 18-24	30 days	Email	
freq	40238	40170	40259	50257	33599	4

In [11]:
```python
print(df.columns)
```

```
Index(['Campaign_ID', 'Company', 'Campaign_Type', 'Target_Audience',
       'Duration', 'Channel_Used', 'Conversion_Rate', 'Acquisition_Cost',
       'ROI', 'Location', 'Date', 'Clicks', 'Impressions', 'Engagement_Sco
re',
       'Customer_Segment'],
      dtype='object')
```

In [12]:
```python
# Unique target audiences
unique_audiences = df["Target_Audience"].unique()
print("Unique Target Audiences:", unique_audiences)
```

```
Unique Target Audiences: ['Men 18-24' 'Women 35-44' 'Men 25-34' 'All Ages'
'Women 25-34']
```

In [13]:
```python
# Unique marketing channels
unique_channels = df["Channel_Used"].unique()
print("Unique Marketing Channels:", unique_channels)
```

```
Unique Marketing Channels: ['Google Ads' 'YouTube' 'Instagram' 'Website'
'Facebook' 'Email']
```

In [14]:
```python
# Function to detect outliers using Interquartile Range (IQR)
def detect_outliers(column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = df[(df[column] < lower_bound) | (df[column] > upper_bound)
    return outliers

# Find outliers in key numeric columns
outliers_impressions = detect_outliers("Impressions")
outliers_clicks = detect_outliers("Clicks")
outliers_acquisition_cost = detect_outliers("Acquisition_Cost")
outliers_ROI = detect_outliers("ROI")
outliers_Engagement_Score = detect_outliers("Engagement_Score")

# Print how many outliers were found
print(f"Outliers in Impressions: {len(outliers_impressions)}")
print(f"Outliers in Clicks: {len(outliers_clicks)}")
print(f"Outliers in Acquisition Cost: {len(outliers_acquisition_cost)}")
print(f"Outliers in ROI: {len(outliers_ROI)}")
print(f"Outliers in Engagement Score: {len(outliers_Engagement_Score)}")
```

```
Outliers in Impressions: 0
Outliers in Clicks: 0
Outliers in Acquisition Cost: 0
Outliers in ROI: 0
Outliers in Engagement Score: 0
```

In [15]:
```
# Calculate Cost Per Click and click Through rate and creating a new colu
df["Cost_Per_Click"] = df["Acquisition_Cost"] / df["Clicks"]
df["Click_Through_Rate"] = (df["Clicks"] / df["Impressions"]) * 100

df.head()
```

Out[15]:

	Campaign_ID	Company	Campaign_Type	Target_Audience	Duration	Channel_
0	1	Innovate Industries	Email	Men 18-24	30 days	Goog
1	2	NexGen Systems	Email	Women 35-44	60 days	Goog
2	3	Alpha Innovations	Influencer	Men 25-34	30 days	Yo
3	4	DataTech Solutions	Display	All Ages	60 days	Yo
4	5	NexGen Systems	Email	Men 25-34	15 days	Yo

In [16]:
```
df.shape
```

Out[16]:
```
(200005, 17)
```

In [17]:
```
df["Conversion_Rate"] = (df["Conversion_Rate"]) * 100  # Already in perce

df[["Click_Through_Rate", "Cost_Per_Click", "Conversion_Rate"]].describe(
```

Out[17]:

	Click_Through_Rate	Cost_Per_Click	Conversion_Rate
count	200005.000000	200005.000000	200005.000000
mean	14.040504	32.008319	8.006885
std	13.087980	26.925841	4.060177
min	1.005429	5.021084	1.000000
25%	5.860637	15.092037	5.000000
50%	9.978960	22.773973	8.000000
75%	16.969848	38.598253	12.000000
max	99.202393	199.960000	15.000000

In [18]:
```
df.head()
```

Out[18]:

	Campaign_ID	Company	Campaign_Type	Target_Audience	Duration	Channel_
0	1	Innovate Industries	Email	Men 18-24	30 days	Goog
1	2	NexGen Systems	Email	Women 35-44	60 days	Goog
2	3	Alpha Innovations	Influencer	Men 25-34	30 days	Yo
3	4	DataTech Solutions	Display	All Ages	60 days	Yo
4	5	NexGen Systems	Email	Men 25-34	15 days	Yo

In [19]:

```python
# Grouping by channel and calculating key performance metrics
channel_performance = df.groupby("Channel_Used").agg({
    "ROI": "mean",
    "Click_Through_Rate": "mean",
    "Conversion_Rate": "mean",
    "Acquisition_Cost": "mean"
}).reset_index()

# Sorting by ROI
channel_performance = channel_performance.sort_values(by="ROI", ascending

# Display results
print(channel_performance)
```

```
  Channel_Used       ROI  Click_Through_Rate  Conversion_Rate  \
1     Facebook  5.018672           14.049724         7.998995
4      Website  5.014114           14.096941         8.018195
2   Google Ads  5.003126           13.918943         8.018062
0        Email  4.996487           14.054269         8.028156
5      YouTube  4.993720           14.119755         7.988980
3    Instagram  4.988706           14.003691         7.988650

   Acquisition_Cost
1      12510.768617
4      12487.842001
2      12528.245036
0      12526.387809
5      12481.570688
3      12491.760002
```

In [20]:

```python
# Top-performing campaigns (highest ROI)
top_campaigns = df.groupby(["Company", "Campaign_ID"]).agg({
    "ROI": "max",
    "Click_Through_Rate": "mean",
    "Conversion_Rate": "mean",
}).reset_index()

# Sorting by ROI
top_campaigns = top_campaigns.sort_values("ROI", ascending=False).head(5)
```

```
# Display
print("Top Campaigns:")
print(top_campaigns)
```

```
Top Campaigns:
                     Company  Campaign_ID  ROI  Click_Through_Rate  \
112313  Innovate Industries       162525  8.0            9.705248
104621  Innovate Industries       124541  8.0           10.181311
149029       NexGen Systems       146166  8.0            8.454488
77076     DataTech Solutions       185179  8.0           52.676240
153137       NexGen Systems       167119  8.0           10.883464

        Conversion_Rate
112313              9.0
104621              4.0
149029             15.0
77076               7.0
153137              2.0
```

In [21]:
```
# Worst-performing campaigns (lowest ROI)
bottom_campaigns = df.groupby(["Company", "Campaign_ID"]).agg({
    "ROI": "min",
    "Click_Through_Rate": "mean",
    "Conversion_Rate": "mean"
}).reset_index()

# Sorting by ROI
bottom_campaigns = bottom_campaigns.sort_values("ROI").head(5)

print("\nWorst Campaigns:")
print(bottom_campaigns)
```

```
Worst Campaigns:
               Company  Campaign_ID  ROI  Click_Through_Rate  Conversion_R
ate
153246  NexGen Systems       167572  2.0            8.231597                1
4.0
161330        TechCorp         7600  2.0           22.053676                1
0.0
167871        TechCorp        40005  2.0            9.545615
5.0
122158  NexGen Systems        12211  2.0           11.666442
4.0
193907        TechCorp       169511  2.0            5.524862                1
2.0
```

In [22]:
```
# Group by Location
location_performance = df.groupby("Location").agg({
    "ROI": "mean",
    "Click_Through_Rate": "mean",
    "Conversion_Rate": "mean",
}).reset_index()

# Sorting by ROI
location_performance = location_performance.sort_values(by="ROI", ascendi

# Display results
print(location_performance)# Grouping by marketing channel
```

```
      Location       ROI  Click_Through_Rate  Conversion_Rate
3        Miami  5.012282           14.024957         8.004743
2  Los Angeles  5.010876           14.067175         8.001302
1      Houston  5.007174           14.059033         7.994893
0      Chicago  5.001555           14.045011         8.013071
4     New York  4.980185           14.006619         8.020337
```

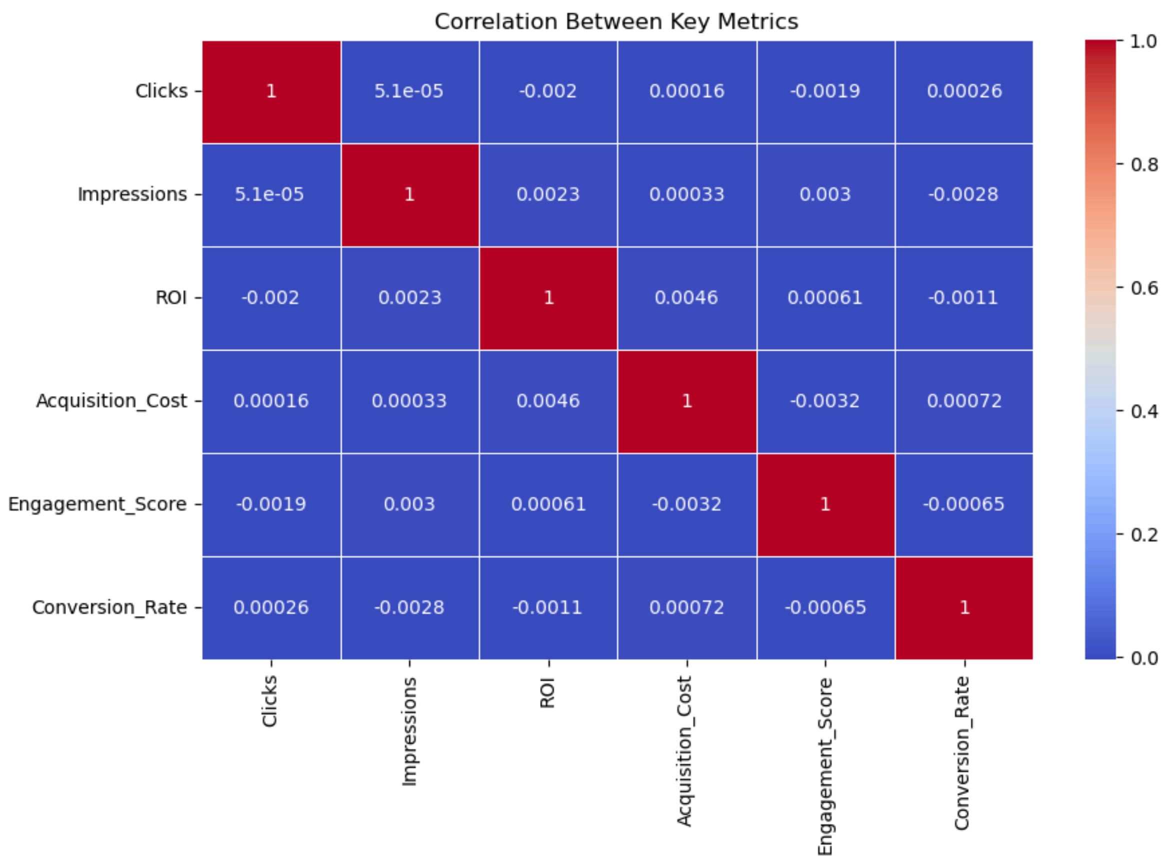
In [23]:
```
df["Date"] = pd.to_datetime(df["Date"], format="%d/%m/%Y")
```

In [24]:
```
df["Month_Year"] = df["Date"].dt.strftime("%B %Y")
df.head
```

Out[24]:
```
<bound method NDFrame.head of         Campaign_ID             Company C
ampaign_Type Target_Audience  \
0                 1  Innovate Industries         Email       Men 18-24
1                 2       NexGen Systems         Email     Women 35-44
2                 3    Alpha Innovations    Influencer       Men 25-34
3                 4   DataTech Solutions       Display        All Ages
4                 5       NexGen Systems         Email       Men 25-34
...             ...                  ...           ...             ...
200000       200001             TechCorp       Display        All Ages
200001       200002   DataTech Solutions         Email       Men 25-34
200002       200003   DataTech Solutions  Social Media       Men 18-24
200003       200004  Innovate Industries    Influencer        All Ages
200004       200005  Innovate Industries  Social Media     Women 35-44

       Duration Channel_Used  Conversion_Rate  Acquisition_Cost   ROI  \
0       30 days   Google Ads              4.0             16174  6.29
1       60 days   Google Ads             12.0             11566  5.61
2       30 days      YouTube              7.0             10200  7.18
3       60 days      YouTube             11.0             12724  5.55
4       15 days      YouTube              5.0             16452  6.50
...         ...          ...              ...               ...   ...
200000  30 days   Google Ads              6.0             18365  2.84
200001  15 days     Facebook              2.0              8168  4.14
200002  45 days      Website              5.0             13397  3.25
200003  30 days      YouTube             10.0             18508  3.86
200004  45 days   Google Ads              1.0             13835  6.64

           Location       Date  Clicks  Impressions  Engagement_Score  \
0           Chicago 2021-01-01     506         1922                 6
1          New York 2021-02-01     116         7523                 7
2       Los Angeles 2021-03-01     584         7698                 1
3             Miami 2021-04-01     217         1820                 7
4       Los Angeles 2021-05-01     379         4201                 3
...             ...        ...     ...          ...               ...
200000      Chicago 2021-07-12     858         5988                 1
200001      Chicago 2021-08-12     228         3068                 7
200002     New York 2021-09-12     723         9548                 3
200003      Houston 2021-10-12     528         2763                 1
200004      Chicago 2021-11-12     924         7287                 8

           Customer_Segment  Cost_Per_Click  Click_Through_Rate  \
0         Health & Wellness       31.964427           26.326743
1              Fashionistas       99.706897            1.541938
2       Outdoor Adventurers       17.465753            7.586386
3         Health & Wellness       58.635945           11.923077
4         Health & Wellness       43.408971            9.021662
...                     ...             ...                 ...
200000     Tech Enthusiasts       21.404429           14.328657
200001              Foodies       35.824561            7.431551
200002     Tech Enthusiasts       18.529737            7.572266
200003              Foodies       35.053030           19.109663
200004     Tech Enthusiasts       14.972944           12.680115

            Month_Year
0         January 2021
1        February 2021
2           March 2021
3           April 2021
4             May 2021
...                ...
```

```
200000       July 2021
200001     August 2021
200002  September 2021
200003    October 2021
200004   November 2021

[200005 rows x 18 columns]>
```

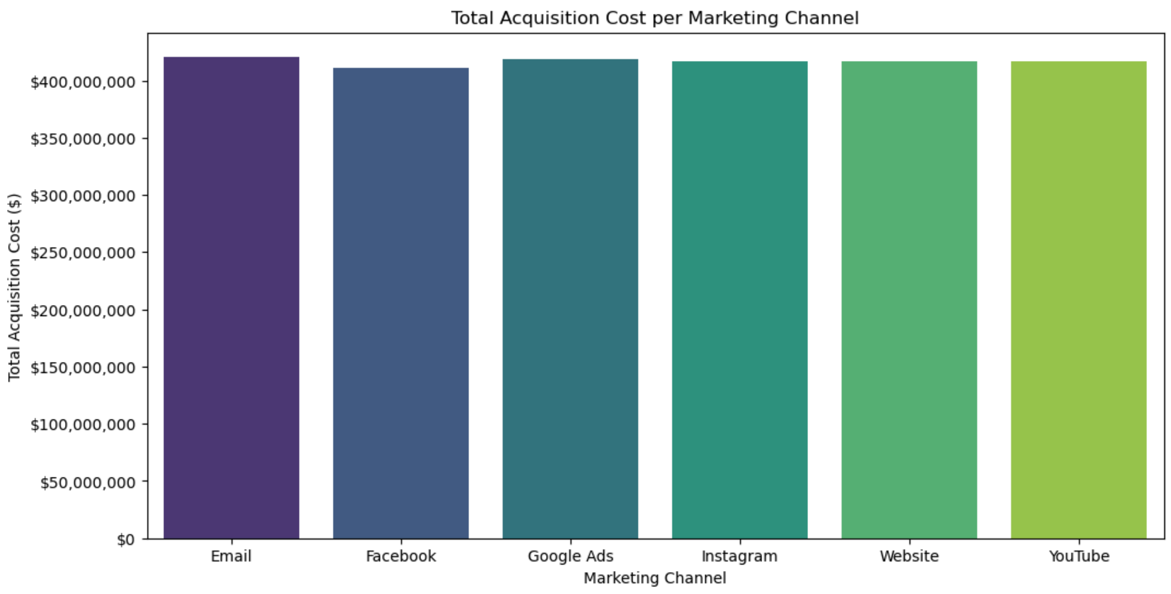
In [25]:
```
#How different metrics relate to each other
correlation_matrix = df[['Clicks', 'Impressions', 'ROI', 'Acquisition_Cos

# Plot Heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0
plt.title("Correlation Between Key Metrics")
plt.show()
```



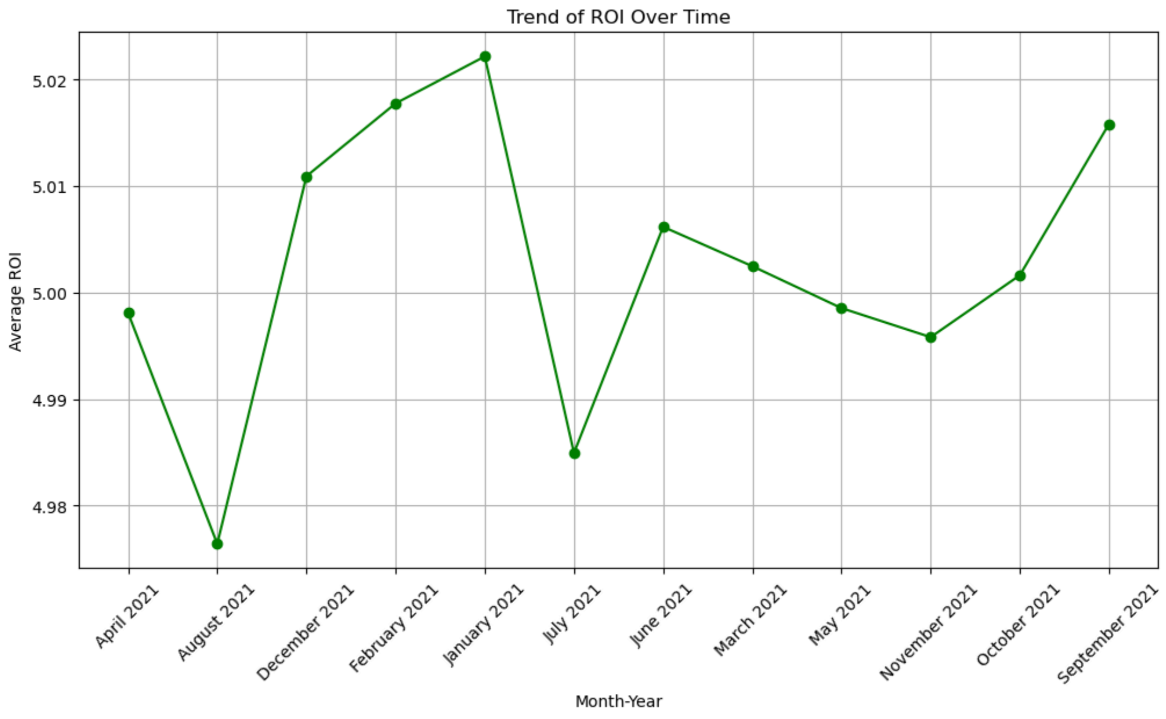
	Clicks	Impressions	ROI	Acquisition_Cost	Engagement_Score	Conversion_Rate
Clicks	1	5.1e-05	-0.002	0.00016	-0.0019	0.00026
Impressions	5.1e-05	1	0.0023	0.00033	0.003	-0.0028
ROI	-0.002	0.0023	1	0.0046	0.00061	-0.0011
Acquisition_Cost	0.00016	0.00033	0.0046	1	-0.0032	0.00072
Engagement_Score	-0.0019	0.003	0.00061	-0.0032	1	-0.00065
Conversion_Rate	0.00026	-0.0028	-0.0011	0.00072	-0.00065	1

In [57]:
```
#Total Spending Cost per Marketing Channel
plt.figure(figsize=(12, 6))

# Group by Marketing Channel and sum Acquisition Cost
df_grouped = df.groupby("Channel_Used")["Acquisition_Cost"].sum().reset_i

# Corrected bar plot syntax
sns.barplot(x="Channel_Used", y="Acquisition_Cost", hue="Channel_Used", d

plt.title("Total Acquisition Cost per Marketing Channel")
plt.xlabel("Marketing Channel")
plt.ylabel("Total Acquisition Cost ($)")

# Format Y-axis labels to currency
formatter = mticker.StrMethodFormatter('${x:,.0f}')
plt.gca().yaxis.set_major_formatter(formatter)
```

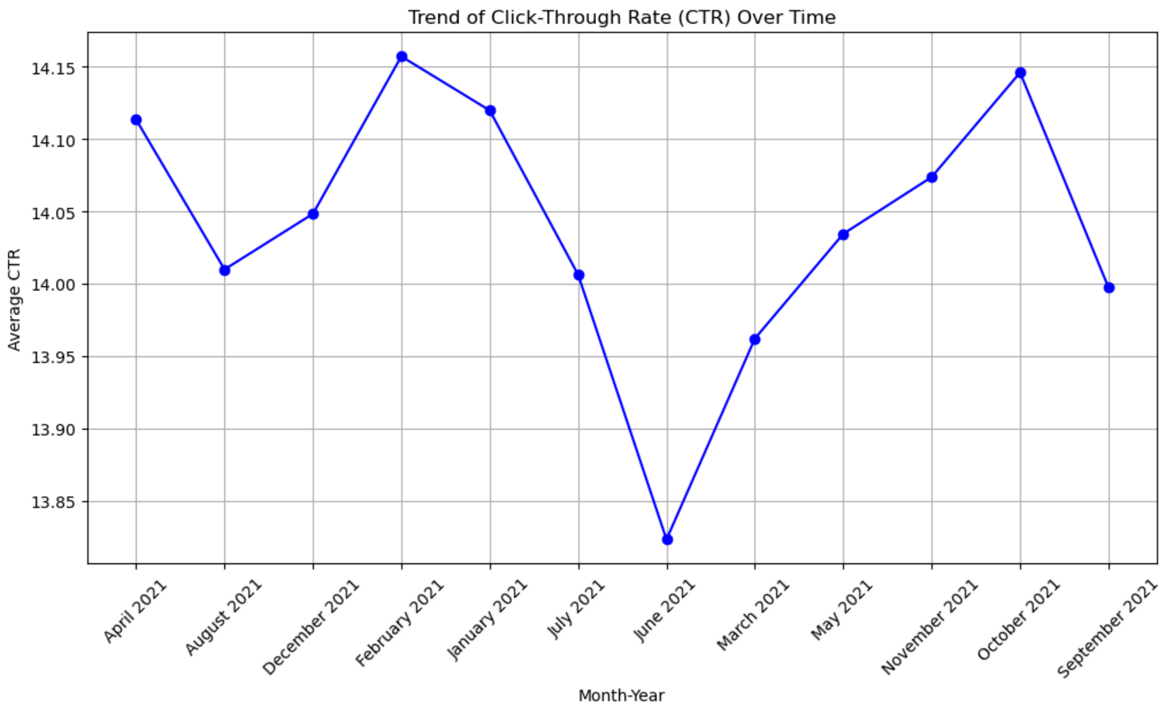
```
# Show the plot
plt.show()
```



In [27]:
```
# Group by Date and calculate mean ROI
roi_over_time = df.groupby("Month_Year")["ROI"].mean()

# Plot Line Chart
plt.figure(figsize=(12, 6))
plt.plot(roi_over_time.index, roi_over_time, marker="o", linestyle="-", c
plt.title("Trend of ROI Over Time")
plt.xlabel("Month-Year")
plt.ylabel("Average ROI")
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.grid(True)
plt.show()
```



In [28]:
```
ctr_over_time = df.groupby("Month_Year")["Click_Through_Rate"].mean()

# Plot Line Chart
plt.figure(figsize=(12, 6))
plt.plot(ctr_over_time.index, ctr_over_time, marker="o", linestyle="-", c
plt.title("Trend of Click-Through Rate (CTR) Over Time")
plt.xlabel("Month-Year")
plt.ylabel("Average CTR")
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability
plt.grid(True)
plt.show()
```


Trend of Click-Through Rate (CTR) Over Time
Average CTR
April 2021
August 2021
December 2021
February 2021
January 2021
July 2021
June 2021
March 2021
May 2021
November 2021
October 2021
September 2021
Month-Year