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

In [5]:
```
df=pd.read_csv("C:/Users/NITISH/Downloads/MOD.csv")
```

In [6]:
```
df
```

Out[6]:

	Age Group	Gender	Relationship Status	City	Captured	Name of App 1	Percentage usage of App 1	Normalized percentage1	I
0	26 - 35	Female	Married with kids	Delhi NCR	24.0	Youtube	11.80%	2.83%	W
1	18 - 25	Female	In a relationship	Mumbai	48.0	Youtube	33.08%	15.88%	
2	26 - 35	Female	In a relationship	Kolkata	24.0	Chrome	20.20%	4.85%	F
3	26 - 35	Female	Married with kids	Delhi NCR	24.0	Gmail	15.11%	3.63%	W
4	18 - 25	Female	In a relationship	Mumbai	24.0	Whatsapp	7.20%	1.73%	
...	...	...	...	...	...	...	...	...	
287	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
288	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
289	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
290	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
291	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

292 rows × 36 columns

In [7]:
```
print(df.head())  # Display the first few rows of the DataFrame
print(df.info())   # Summary of the DataFrame including data types and miss
print(df.describe())  # Summary statistics
```

```
  Age Group  Gender Relationship Status       City  Captured Name of App 1
\
0   26 - 35  Female   Married with kids  Delhi NCR      24.0       Youtube
1   18 - 25  Female   In a relationship     Mumbai      48.0       Youtube
2   26 - 35  Female   In a relationship    Kolkata      24.0        Chrome
3   26 - 35  Female   Married with kids  Delhi NCR      24.0         Gmail
4   18 - 25  Female   In a relationship     Mumbai      24.0      Whatsapp

  Percentage usage of App 1 Normalized percentage1     Name of App 2  \
0                    11.80%                  2.83%          Whatsapp
1                    33.08%                 15.88%  Candy Crush Soda
2                    20.20%                  4.85%          Facebook
3                    15.11%                  3.63%          Whatsapp
4                     7.20%                  1.73%           Netflix

  Percentage usage of App 2  ...                Name of App 8  \
0                     8.10%  ...                       Meesho
1                    18.92%  ...                      Whatsapp
2                     2.80%  ...                          AJIO
3                    11.33%  ...  Google Play Store (Service)
4                     2.90%  ...                           NaN

  Percentage usage of App 8 Normalized percentage8  \
0                     0.70%                  0.17%
1                     1.74%                  0.84%
2                     0.20%                  0.05%
3                     1.25%                  0.30%
4                       NaN                  0.00%

                Name of App 9 Percentage usage of App 9  \
0  Google Play Store (Service)                     0.70%
1                      Netflix                     1.22%
2  Google Play Store (Service)                     0.20%
3                   Truecaller                     1.03%
4                          NaN                       NaN

  Normalized percentage9              Name of App 10  \
0                  0.17%                      Google
1                  0.59%  Google Play Store (Service)
2                  0.05%                   Messenger
3                  0.25%                        AJIO
4                  0.00%                         NaN

  Percentage usage of App 10 Normalized percentage10 Total Apps
0                     0.60%                   0.14%    Youtube
1                     1.14%                   0.55%     Chrome
2                     0.10%                   0.02%      Gmail
3                     0.91%                   0.22%   Whatsapp
4                       NaN                   0.00%  Instagram

[5 rows x 36 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 292 entries, 0 to 291
Data columns (total 36 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Age Group                  221 non-null    object
 1   Gender                     221 non-null    object
 2   Relationship Status        221 non-null    object
 3   City                       221 non-null    object
 4   Captured                   221 non-null    float64
```

```
 5   Name of App 1               220 non-null    object
 6   Percentage usage of App 1   221 non-null    object
 7   Normalized percentage1      221 non-null    object
 8   Name of App 2               221 non-null    object
 9   Percentage usage of App 2   219 non-null    object
 10  Normalized percentage2      221 non-null    object
 11  Name of App 3               221 non-null    object
 12  Percentage usage of App 3   217 non-null    object
 13  Normalized percentage3      221 non-null    object
 14  Name of App 4               221 non-null    object
 15  Percentage usage of App 4   219 non-null    object
 16  Normalized percentage4      221 non-null    object
 17  Name of App 5               221 non-null    object
 18  Percentage usage of App 5   216 non-null    object
 19  Normalized percentage5      221 non-null    object
 20  Name of App 6               206 non-null    object
 21  Percentage usage of App 6   201 non-null    object
 22  Normalized percentage6      221 non-null    object
 23  Name of App 7               187 non-null    object
 24  Percentage usage of App 7   183 non-null    object
 25  Normalized percentage7      221 non-null    object
 26  Name of App 8               153 non-null    object
 27  Percentage usage of App 8   150 non-null    object
 28  Normalized percentage8      221 non-null    object
 29  Name of App 9               128 non-null    object
 30  Percentage usage of App 9   119 non-null    object
 31  Normalized percentage9      221 non-null    object
 32  Name of App 10              106 non-null    object
 33  Percentage usage of App 10  99 non-null     object
 34  Normalized percentage10     221 non-null    object
 35  Total Apps                  291 non-null    object
dtypes: float64(1), object(35)
memory usage: 82.3+ KB
None
         Captured
count  221.000000
mean    40.072398
std     56.072633
min     24.000000
25%     24.000000
50%     24.000000
75%     24.000000
max    240.000000
```

In [8]:
```
df.dropna(inplace=True)
```

In [9]:
```
print(df.columns)
```

```
Index(['Age Group', 'Gender', 'Relationship Status', 'City', 'Captured',
       'Name of App 1', 'Percentage usage of App 1', 'Normalized percentag
e1',
       'Name of App 2', 'Percentage usage of App 2', 'Normalized percentag
e2',
       'Name of App 3', 'Percentage usage of App 3', 'Normalized percentag
e3',
       'Name of App 4', 'Percentage usage of App 4', 'Normalized percentag
e4',
       'Name of App 5', 'Percentage usage of App 5', 'Normalized percentag
e5',
       'Name of App 6', 'Percentage usage of App 6', 'Normalized percentag
e6',
       'Name of App 7', 'Percentage usage of App 7', 'Normalized percentag
e7',
       'Name of App 8', 'Percentage usage of App 8', 'Normalized percentag
e8',
       'Name of App 9', 'Percentage usage of App 9', 'Normalized percentag
e9',
       'Name of App 10', 'Percentage usage of App 10',
       'Normalized percentage10', 'Total Apps'],
      dtype='object')
```

In [10]:
```
print(df['Age Group'].head())  # Check the first few values of the 'Age Gro
print(df['Age Group'].dtype)
```

```
0    26 - 35
1    18 - 25
2    26 - 35
3    26 - 35
6    26 - 35
Name: Age Group, dtype: object
object
```

In [11]:
```
df['Age Group'] = pd.Categorical(df['Age Group'])
```

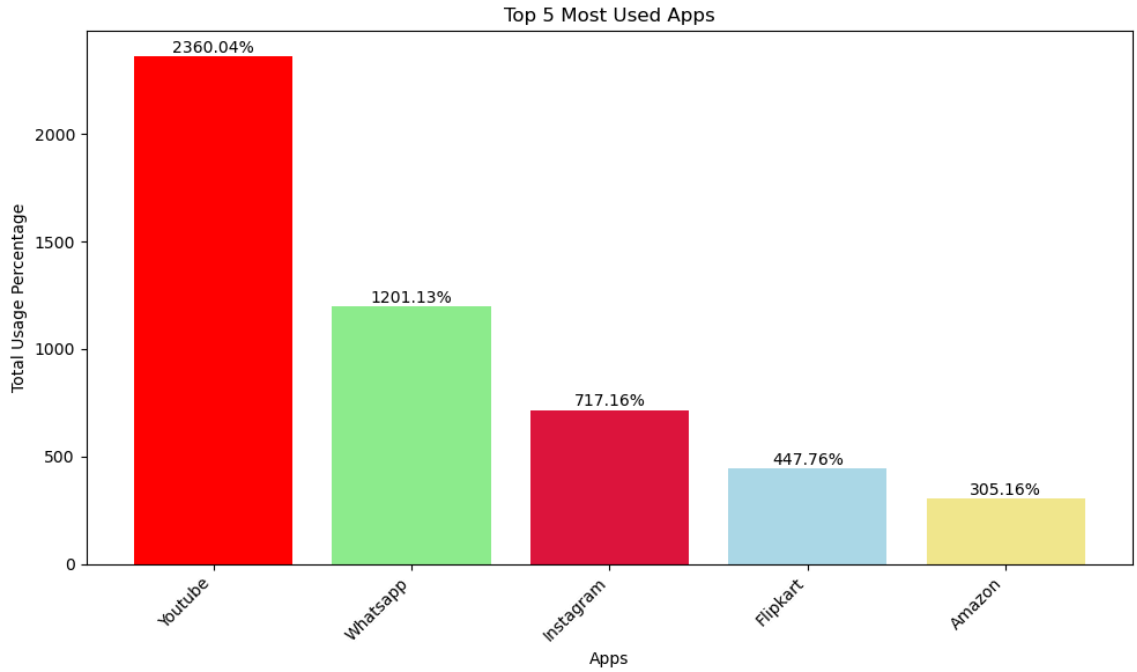
In [12]:
```
print(df.dtypes)
```

```
Age Group                     category
Gender                          object
Relationship Status             object
City                            object
Captured                       float64
Name of App 1                   object
Percentage usage of App 1       object
Normalized percentage1          object
Name of App 2                   object
Percentage usage of App 2       object
Normalized percentage2          object
Name of App 3                   object
Percentage usage of App 3       object
Normalized percentage3          object
Name of App 4                   object
Percentage usage of App 4       object
Normalized percentage4          object
Name of App 5                   object
Percentage usage of App 5       object
Normalized percentage5          object
Name of App 6                   object
Percentage usage of App 6       object
Normalized percentage6          object
Name of App 7                   object
Percentage usage of App 7       object
Normalized percentage7          object
Name of App 8                   object
Percentage usage of App 8       object
Normalized percentage8          object
Name of App 9                   object
Percentage usage of App 9       object
Normalized percentage9          object
Name of App 10                  object
Percentage usage of App 10      object
Normalized percentage10         object
Total Apps                      object
dtype: object
```

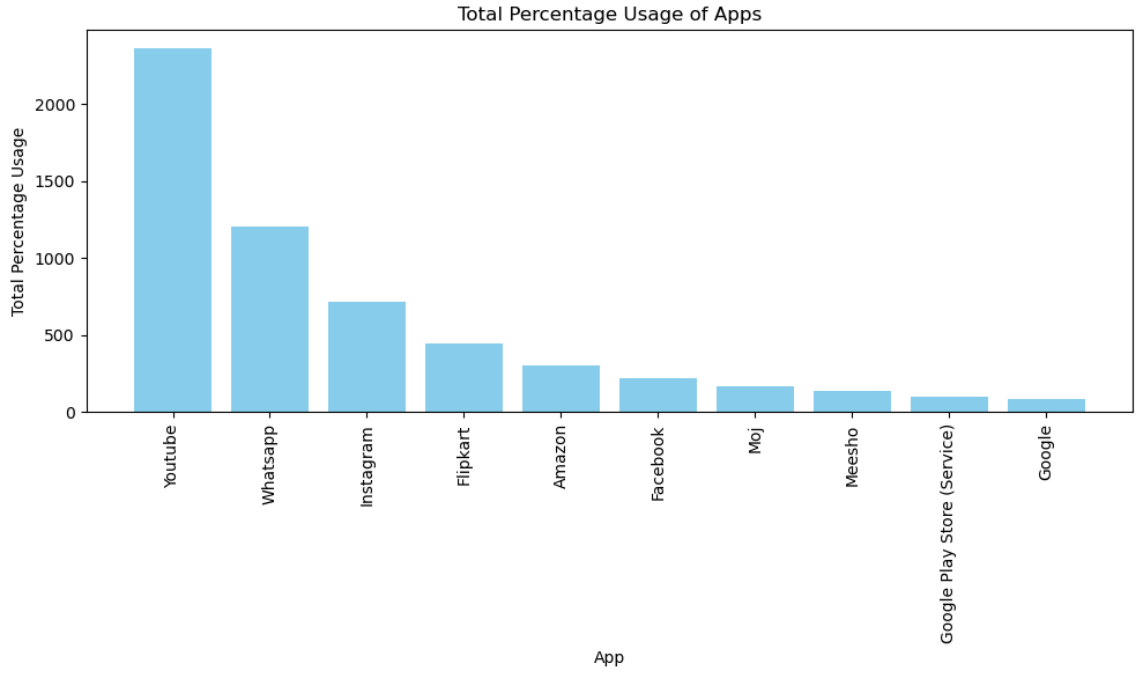
In [16]:
```
# Get the top 5 most used apps1 this shows the very 1st used app on 1st pri
top_5_apps = sorted_apps[:5]

print("Top 5 most used apps:", top_5_apps)
```

```
Top 5 most used apps: ['Youtube', 'Whatsapp', 'Instagram', 'Flipkart', 'Am
azon']
```

In [59]:
```
import matplotlib.pyplot as plt

# Extract the usage percentages of the top 5 apps
usage_percentages = [total_usage[app] for app in top_5_apps]

# Define custom colors for the bars
colors = ['red', 'lightgreen', 'crimson', 'lightblue', 'Khaki']

# Create a bar plot with custom colors
plt.figure(figsize=(10, 6))
bars = plt.bar(top_5_apps, usage_percentages, color=colors)

# Add the values on each bar
for bar, percentage in zip(bars, usage_percentages):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height() + 0.5, f'{

plt.xlabel('Apps')
plt.ylabel('Total Usage Percentage')
plt.title('Top 5 Most Used Apps')
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better read
plt.tight_layout()  # Adjust layout to prevent clipping of labels
plt.show()
```



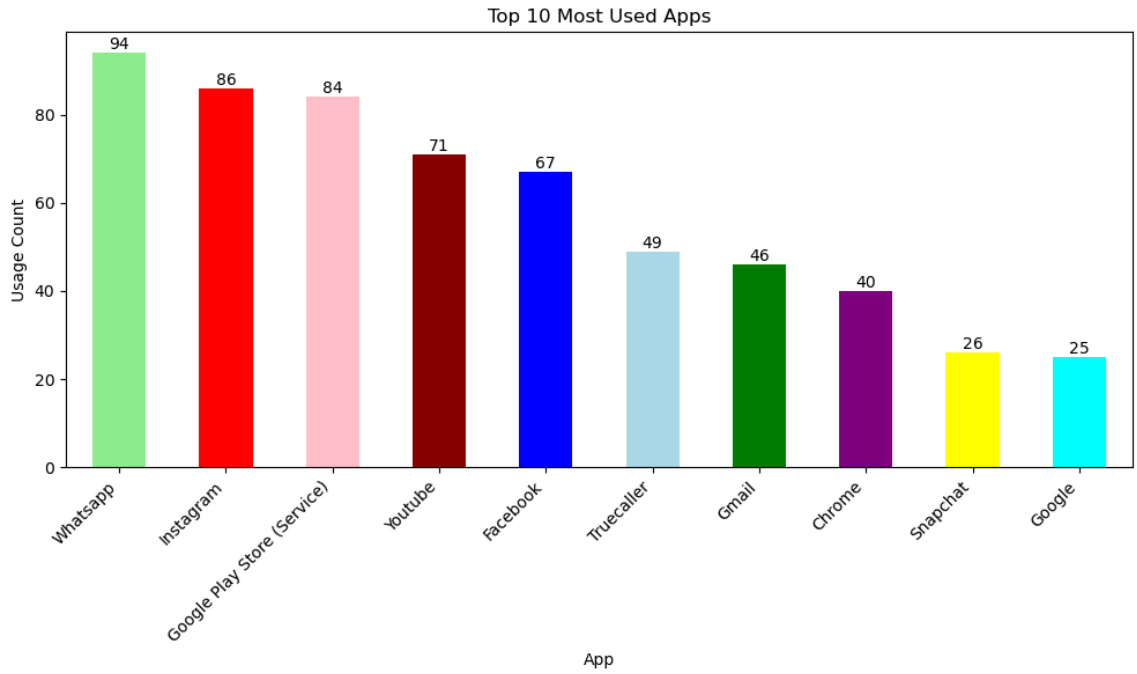
In [63]:
```
#TOP 10 USED APP1

import matplotlib.pyplot as plt

# Convert the total_usage dictionary to a DataFrame for easier plotting
total_usage_df = pd.DataFrame(total_usage.items(), columns=['App', 'Total P

# # Sort the DataFrame by total percentage usage in descending order
total_usage_df.sort_values(by='Total Percentage Usage', ascending=False, in

# # Plot the bar graph
plt.figure(figsize=(10, 6))
plt.bar(total_usage_df['App'], total_usage_df['Total Percentage Usage'], co
plt.xlabel('App')
plt.ylabel('Total Percentage Usage')
plt.title('Total Percentage Usage of Apps')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```



In [22]:
```
# Initialize a dictionary to store the total usage count of each app among
total_usage_count = {}

# Iterate over the rows (respondents) in the DataFrame
for _, row in df.iterrows():
    for col in row.index:
        if 'Name of App' in col:
            app_name = row[col]
            # If the app name is not NaN
            if pd.notna(app_name):
                # Increment the usage count for this app
                total_usage_count[app_name] = total_usage_count.get(app_nam

# Find the most used app among all cities
most_used_app = max(total_usage_count, key=total_usage_count.get)
print("The most used app among all cities is:", most_used_app)
```

```
The most used app among all cities is: Whatsapp
```

In [49]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Count the occurrences of each app name across all respondents and select
top_10_apps = df.filter(like='Name of App').stack().value_counts().head(10)

# Define custom colors for the bars
colors = ['lightgreen', 'red', 'pink', 'darkred', 'blue', 'lightblue', 'gre

# Plot the total usage count of each app with custom colors
plt.figure(figsize=(10, 6))
top_10_apps.plot(kind='bar', color=colors)
plt.title('Top 10 Most Used Apps')
plt.xlabel('App')
plt.ylabel('Usage Count')
plt.xticks(rotation=45, ha='right')  # Rotate x-axis labels for better read

# Add count on each bar
[plt.text(i, count, str(count), ha='center', va='bottom') for i, count in e

plt.tight_layout()
plt.show()

# Find the most used app among the top 10 apps
most_used_app, most_used_app_count = top_10_apps.idxmax(), top_10_apps.max(
print(f"The most used app among the top 10 apps is: {most_used_app} with a
```



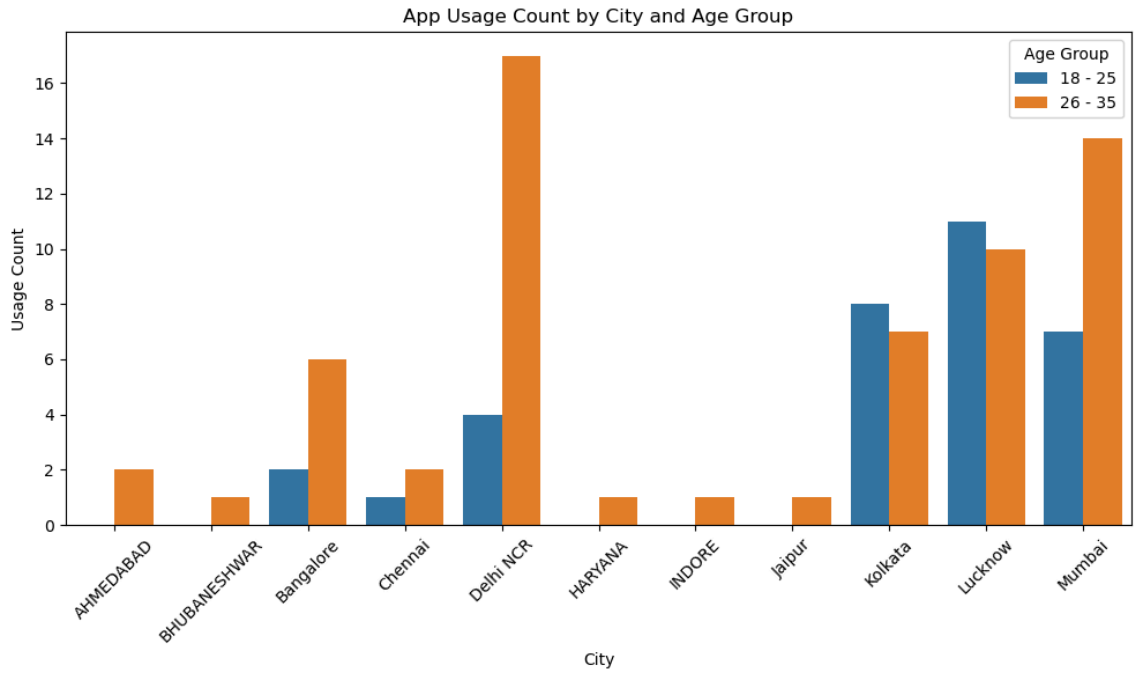

```
The most used app among the top 10 apps is: Whatsapp with a count of 94
```

In [24]:
```
print(df.columns)
```

```
Index(['Age Group', 'Gender', 'Relationship Status', 'City', 'Captured',
       'Name of App 1', 'Percentage usage of App 1', 'Normalized percentag
e1',
       'Name of App 2', 'Percentage usage of App 2', 'Normalized percentag
e2',
       'Name of App 3', 'Percentage usage of App 3', 'Normalized percentag
e3',
       'Name of App 4', 'Percentage usage of App 4', 'Normalized percentag
e4',
       'Name of App 5', 'Percentage usage of App 5', 'Normalized percentag
e5',
       'Name of App 6', 'Percentage usage of App 6', 'Normalized percentag
e6',
       'Name of App 7', 'Percentage usage of App 7', 'Normalized percentag
e7',
       'Name of App 8', 'Percentage usage of App 8', 'Normalized percentag
e8',
       'Name of App 9', 'Percentage usage of App 9', 'Normalized percentag
e9',
       'Name of App 10', 'Percentage usage of App 10',
       'Normalized percentage10', 'Total Apps'],
      dtype='object')
```

In [25]:
```
# 1. Filter the DataFrame to include only the relevant columns
app_data = df[['City', 'Age Group', 'Name of App 1', 'Name of App 2','Name

# 2. Group the data by both "City" and "Age Group" and count the app usage
grouped_data = app_data.groupby(['City', 'Age Group']).count()
```

In [26]:
```
# 3. Stack the grouped data to create a multi-level index
stacked_data = grouped_data.stack()

# 4. Reset the index to make it a regular DataFrame
stacked_data = stacked_data.reset_index()

# 5. Rename the columns for clarity
stacked_data.columns = ['City', 'Age Group', 'App', 'Usage Count']

# 6. Plot the stacked bar chart
plt.figure(figsize=(10, 6))
sns.barplot(x='City', y='Usage Count', hue='Age Group', data=stacked_data)
plt.title('App Usage Count by City and Age Group')
plt.xlabel('City')
plt.ylabel('Usage Count')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [36]:
```
print(grouped_data.columns)
```

```
Index(['index', 'City', 'Age Group', 'Name of App 1', 'Name of App 2',
       'Name of App 3', 'Name of App 4', 'Name of App 5', 'Name of App 6',
       'Name of App 7', 'Name of App 8', 'Name of App 9', 'Name of App 1
0'],
      dtype='object')
```

In [ ]: