Dating_Recommendations_using_Python

In [2]:

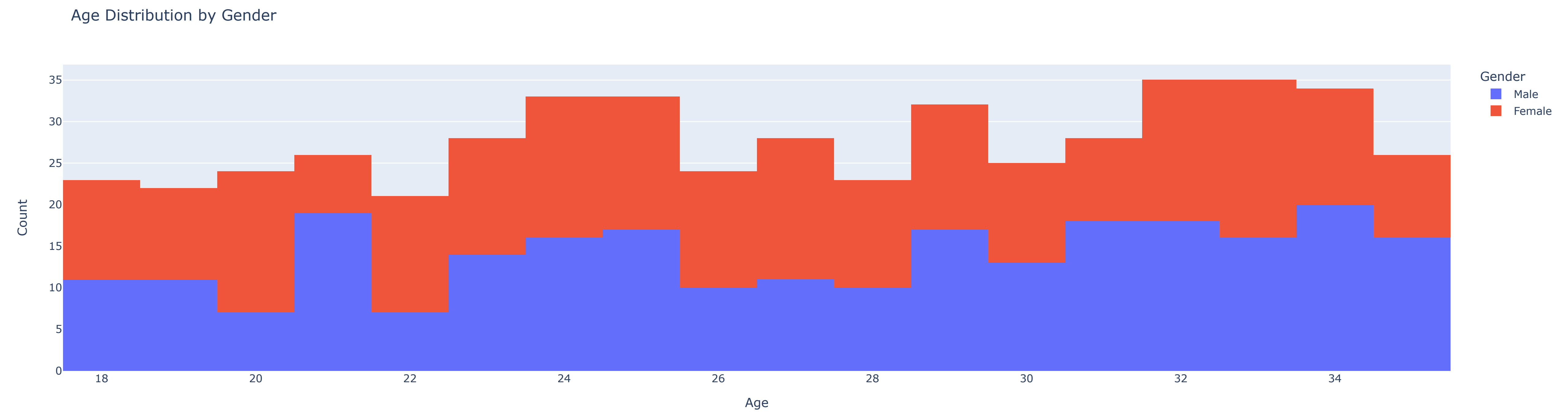
```python
import pandas as pd
import plotly.express as px

data =  pd.read_csv("dating_app_dataset.csv")
print(data.head())
```

```
   User ID  Age  Gender    Height  \
0        1   30    Male  5.240385   
1        2   27  Female  4.937625   
2        3   29  Female  5.806296   
3        4   29  Female  5.101402   
4        5   32    Male  5.986670   

                                           Interests             Looking For  \
0  ['Sports', 'Cooking', 'Hiking', 'Music', 'Movi...           Casual Dating   
1                              ['Sports', 'Reading']              Friendship   
2                                         ['Sports']           Casual Dating   
3                                        ['Reading']                Marriage   
4  ['Sports', 'Hiking', 'Music', 'Movies', 'Readi...  Long-term Relationship   

  Children    Education Level              Occupation  Swiping History  \
0       No        High School                 Student               96   
1      Yes    Master's Degree                  Artist               96   
2       No  Bachelor's Degree  Social Media Influencer               64   
3       No              Ph.D.                  Doctor               67   
4      Yes              Ph.D.                Engineer               93   

  Frequency of Usage  
0             Weekly  
1            Monthly  
2              Daily  
3              Daily  
4            Monthly  
```

Age distribution by gender

In [3]:

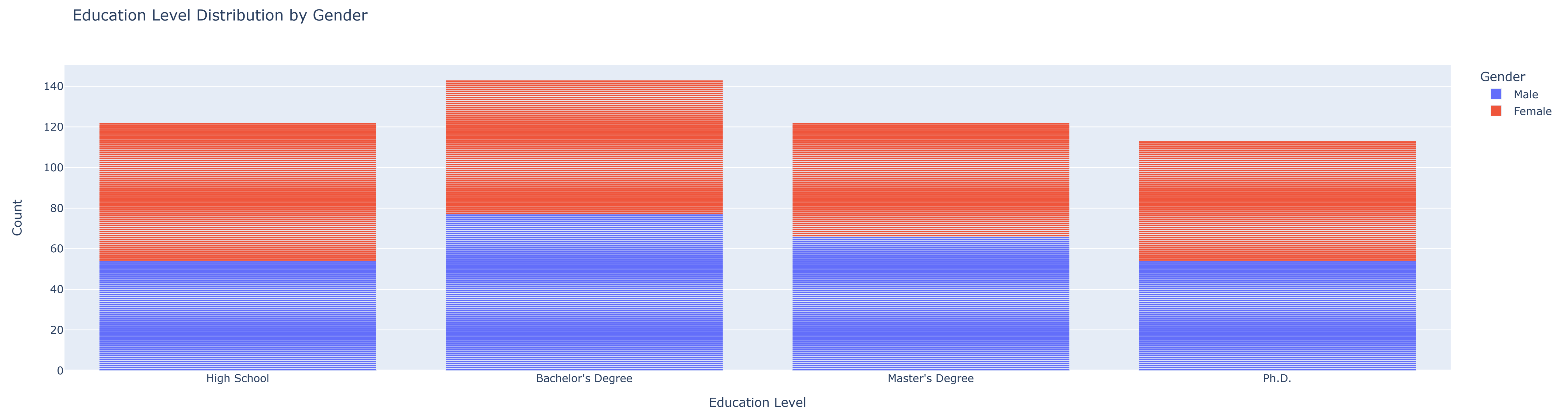
```python
fig = px.histogram(data, x="Age", color="Gender", nbins=20,
                   title="Age Distribution by Gender")
fig.update_layout(xaxis_title="Age", yaxis_title="Count")
fig.show()
```



Education level distribution by gender

In [4]:

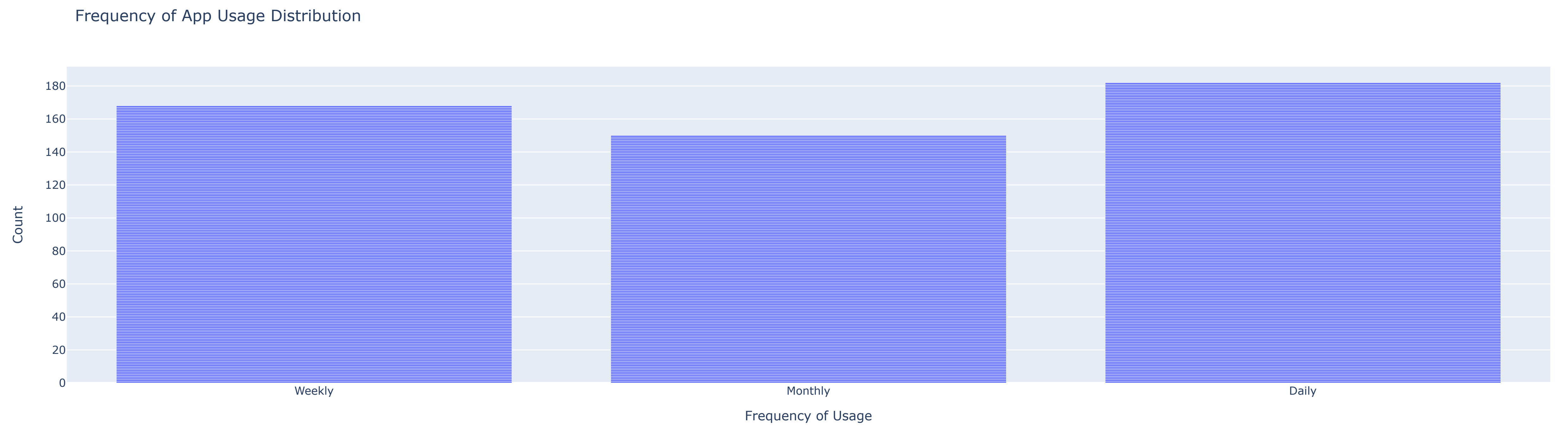
```python
education_order = ["High School", "Bachelor's Degree", "Master's Degree", "Ph.D."]
fig = px.bar(data, x="Education Level", color="Gender",
             category_orders={"Education Level": education_order},
             title="Education Level Distribution by Gender")

fig.update_layout(xaxis_title="Education Level", yaxis_title="Count")
fig.show()
```



Frequency of app usage distribution

In [5]:

```python
fig = px.bar(data, x="Frequency of Usage",
             title="Frequency of App Usage Distribution")
fig.update_layout(xaxis_title="Frequency of Usage",
                  yaxis_title="Count")
fig.show()
```



Separate data into male and female

In [6]:

```python
male_profiles = data[data['Gender'] == 'Male']
female_profiles = data[data['Gender'] == 'Female']
```

In [7]:

```python
def calculate_match_score(profile1, profile2):
    # Shared interests score (1 point per shared interest)
    interests1 = set(eval(profile1['Interests']))
    interests2 = set(eval(profile2['Interests']))
    shared_interests_score = len(interests1.intersection(interests2))

    # Age difference score (higher age difference, lower score)
    age_difference_score = max(0, 10 - abs(profile1['Age'] - profile2['Age']))

    # Swiping history score (higher swiping history, higher score)
    swiping_history_score = min(profile1['Swiping History'], profile2['Swiping History']) / 100

    # Relationship type score (1 point for matching types)
    relationship_type_score = 0
    if profile1['Looking For'] == profile2['Looking For']:
        relationship_type_score = 1

    # Total match score
    total_score = (
        shared_interests_score + age_difference_score + swiping_history_score + relationship_type_score
    )

    return total_score

# Example: Calculate match score between two profiles
profile1 = male_profiles.iloc[0]
profile2 = female_profiles.iloc[0]
match_score = calculate_match_score(profile1, profile2)
print(f"Match score between User {profile1['User ID']} and User {profile2['User ID']} : {match_score}")
```

```
Match score between User 1 and User 2 : 9.96
```

In [9]:

```python
def recommend_profiles(male_profiles, female_profiles):
    recommendations = []

    for _, male_profile in male_profiles.iterrows():
        best_match = None
        best_score = -1

        for _, female_profile in female_profiles.iterrows():
            score = calculate_match_score(male_profile, female_profile)

            if score > best_score:
                best_match = female_profile
                best_score = score

        recommendations.append((male_profile, best_match, best_score))

    return recommendations

# Generate recommendations
recommendations = recommend_profiles(male_profiles, female_profiles)

# Sort recommendations by match score in descending order
recommendations.sort(key=lambda x: x[2], reverse=True)

# Display the top recommendations
for idx, (male_profile, female_profile, match_score) in enumerate(recommendations[:10]):
    print(f"Recommendation {idx + 1}:")
    print(f"Male Profile (User {male_profile['User ID']}): Age {male_profile['Age']}, Interests {male_profile['Interests']}")
    print(f"Female Profile (User {female_profile['User ID']}): Age {female_profile['Age']}, Interests {female_profile['Interests']}")
    print(f"Match Score: {match_score}")
    print()
```

```
Recommendation 1:
Male Profile (User 36): Age 19, Interests ['Movies', 'Cooking', 'Hiking', 'Reading', 'Sports', 'Travel', 'Music']
Female Profile (User 451): Age 19, Interests ['Reading', 'Music', 'Cooking', 'Hiking', 'Travel', 'Sports', 'Movies']
Match Score: 18.79

Recommendation 2:
Male Profile (User 274): Age 29, Interests ['Reading', 'Movies', 'Travel', 'Music', 'Hiking', 'Cooking', 'Sports']
Female Profile (User 300): Age 29, Interests ['Cooking', 'Reading', 'Music', 'Hiking', 'Travel', 'Sports', 'Movies']
Match Score: 18.73

Recommendation 3:
Male Profile (User 456): Age 29, Interests ['Cooking', 'Hiking', 'Sports', 'Travel', 'Music', 'Movies', 'Reading']
Female Profile (User 65): Age 29, Interests ['Travel', 'Movies', 'Reading', 'Sports', 'Music', 'Cooking', 'Hiking']
Match Score: 18.69

Recommendation 4:
Male Profile (User 147): Age 34, Interests ['Reading', 'Travel', 'Movies', 'Hiking', 'Cooking', 'Music', 'Sports']
Female Profile (User 287): Age 34, Interests ['Reading', 'Hiking', 'Cooking', 'Music', 'Movies', 'Travel', 'Sports']
Match Score: 18.66

Recommendation 5:
Male Profile (User 321): Age 20, Interests ['Sports', 'Reading', 'Cooking', 'Travel', 'Movies', 'Hiking', 'Music']
Female Profile (User 168): Age 20, Interests ['Cooking', 'Sports', 'Music', 'Reading', 'Travel', 'Hiking', 'Movies']
Match Score: 18.58

Recommendation 6:
Male Profile (User 323): Age 30, Interests ['Hiking', 'Travel', 'Movies', 'Reading', 'Sports', 'Cooking', 'Music']
Female Profile (User 497): Age 30, Interests ['Hiking', 'Reading', 'Travel', 'Sports', 'Music', 'Cooking', 'Movies']
Match Score: 18.57

Recommendation 7:
Male Profile (User 181): Age 25, Interests ['Sports', 'Music', 'Hiking', 'Travel', 'Cooking', 'Movies', 'Reading']
Female Profile (User 175): Age 25, Interests ['Sports', 'Music', 'Travel', 'Hiking', 'Movies', 'Reading', 'Cooking']
Match Score: 18.34

Recommendation 8:
Male Profile (User 489): Age 33, Interests ['Travel', 'Hiking', 'Reading', 'Sports', 'Music', 'Movies', 'Cooking']
Female Profile (User 99): Age 33, Interests ['Reading', 'Cooking', 'Sports', 'Hiking', 'Movies', 'Music', 'Travel']
Match Score: 18.3

Recommendation 9:
Male Profile (User 280): Age 29, Interests ['Travel', 'Hiking', 'Music', 'Sports', 'Reading', 'Cooking', 'Movies']
Female Profile (User 300): Age 29, Interests ['Cooking', 'Reading', 'Music', 'Hiking', 'Travel', 'Sports', 'Movies']
Match Score: 18.29

Recommendation 10:
Male Profile (User 92): Age 22, Interests ['Music', 'Hiking', 'Cooking', 'Travel', 'Movies', 'Reading', 'Sports']
Female Profile (User 205): Age 22, Interests ['Hiking', 'Movies', 'Reading', 'Travel', 'Sports', 'Cooking', 'Music']
Match Score: 18.2
```

THANK YOU!

GitHub Link: https://github.com/anujtiwari21?tab=repositories