In [12]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [13]:
```
# Step 1: Generate Random Data
np.random.seed(42)  # for reproducibility

# Creating a sample dataset with 1000 users
n = 1000
data = pd.DataFrame({
    'user_id': range(1, n+1),
    'age': np.random.randint(18, 60, size=n),  # ages between 18 and 60
    'skin_type': np.random.choice(['Oily', 'Dry', 'Combination', 'Sensitive'], size=n),
    'acne_severity': np.random.randint(1, 5, size=n),  # 1: Low, 4: High
    'dark_spots': np.random.choice([0, 1], size=n),  # 0: No, 1: Yes
    'recommended_product': np.random.choice(['Product_A', 'Product_B', 'Product_C', 'Product_D'], size=n),
    'skin_improvement_score': np.random.normal(5, 2, n).clip(0, 10)  # normalized score 0-10
})
```

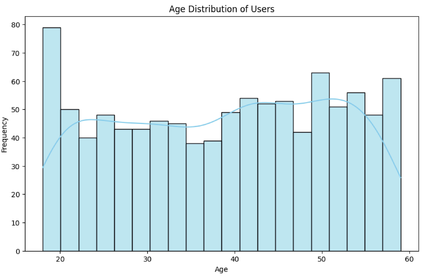
In [21]:
```
# Step 2: Perform EDA

# Summary of the data
print("Data Summary:")
print(data.describe(include='all'))
```

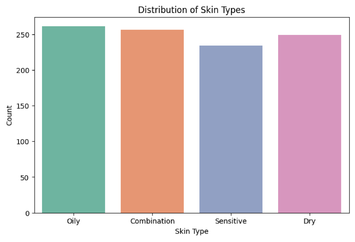
```
Data Summary:
            user_id          age skin_type  acne_severity   dark_spots  \
count   1000.000000  1000.000000      1000    1000.000000  1000.000000
unique          NaN          NaN         4            NaN          NaN
top             NaN          NaN      Oily            NaN          NaN
freq            NaN          NaN       261            NaN          NaN
mean     500.500000    38.745000       NaN       2.461000     0.522000
std      288.819436    12.186734       NaN       1.122381     0.499766
min        1.000000    18.000000       NaN       1.000000     0.000000
25%      250.750000    28.000000       NaN       1.000000     0.000000
50%      500.500000    40.000000       NaN       2.000000     1.000000
75%      750.250000    50.000000       NaN       3.000000     1.000000
max     1000.000000    59.000000       NaN       4.000000     1.000000

       recommended_product  skin_improvement_score
count                 1000             1000.000000
unique                   4                     NaN
top                Product_A                     NaN
freq                   273                     NaN
mean                   NaN                5.055599
std                    NaN                1.998775
min                    NaN                0.000000
25%                    NaN                3.692408
50%                    NaN                5.082674
75%                    NaN                6.394781
max                    NaN               10.000000
```

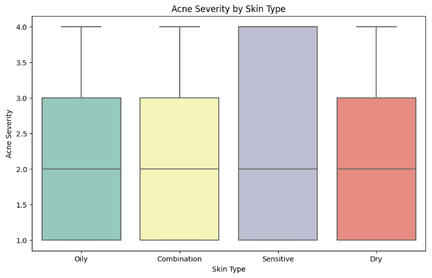
In [23]:
```
# Distribution of Age
plt.figure(figsize=(10, 6))
sns.histplot(data['age'], kde=True, bins=20, color='skyblue')
plt.title('Age Distribution of Users')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



In [22]:
```
# Skin Type Distribution
plt.figure(figsize=(8, 5))
sns.countplot(data=data, x='skin_type', palette='Set2')
plt.title('Distribution of Skin Types')
plt.xlabel('Skin Type')
plt.ylabel('Count')
plt.show()
```



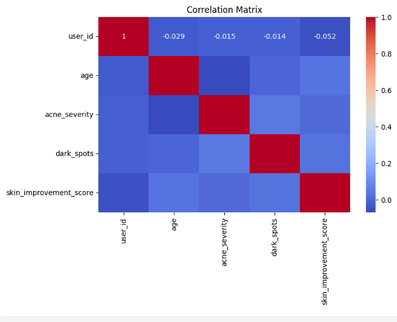
In [24]:
```
# Acne Severity by Skin Type
plt.figure(figsize=(10, 6))
sns.boxplot(data=data, x='skin_type', y='acne_severity', palette='Set3')
plt.title('Acne Severity by Skin Type')
plt.xlabel('Skin Type')
plt.ylabel('Acne Severity')
plt.show()
```



In [18]:
```
# Correlation between Age and Skin Improvement Score
plt.figure(figsize=(8, 6))
sns.scatterplot(data=data, x='age', y='skin_improvement_score', hue='skin_type', palette='viridis')
plt.title('Age vs. Skin Improvement Score')
plt.xlabel('Age')
plt.ylabel('Skin Improvement Score')
plt.show()
```

In [19]:
```
# Distribution of Skin Improvement Scores by Recommended Product
plt.figure(figsize=(12, 6))
sns.violinplot(data=data, x='recommended_product', y='skin_improvement_score', palette='muted')
plt.title('Skin Improvement Scores by Recommended Product')
plt.xlabel('Recommended Product')
plt.ylabel('Skin Improvement Score')
plt.show()
```

In [25]:
```
# Heatmap of Correlations
plt.figure(figsize=(8, 5))
# Select only numerical columns for correlation calculation
numerical_data = data.select_dtypes(include=np.number)
sns.heatmap(numerical_data.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix')
plt.show()
```



In [ ]: