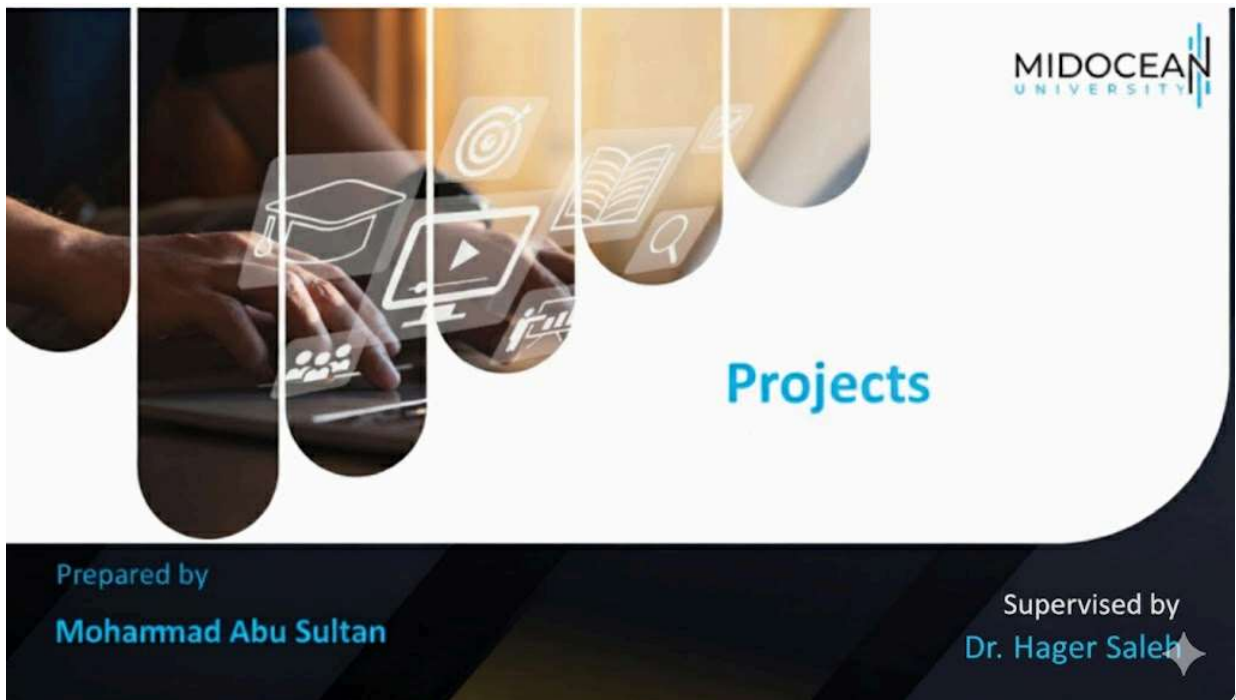


```
from IPython.display import Image, display
display(Image(filename='ttttt.jpg'))
```



```
from google.colab import drive
drive.mount('/content/drive')
```

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount(

```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

✓ Mental Health and Social Media Balance Dataset

```
df = pd.read_csv('/content/drive/MyDrive/MyProject/Mental_Health.csv')
```

✓ Dataset description

```
df.head()
```

	User_ID	Age	Gender	Daily_Screen_Time(hrs)	Sleep_Quality(1-10)	Stress_Level(1-10)
0	U001	44	Male	3.1	7.0	6.0
1	U002	30	Other	5.1	7.0	8.0
2	U003	23	Other	7.4	6.0	7.0
3	U004	36	Female	5.7	7.0	8.0
4	U005	34	Female	7.0	4.0	7.0

Next steps:

[Generate code with df](#)

[New interactive sheet](#)

Getting features type

```
df.dtypes
```

	0
User_ID	object
Age	int64
Gender	object
Daily_Screen_Time(hrs)	float64
Sleep_Quality(1-10)	float64
Stress_Level(1-10)	float64
Days_Without_Social_Media	float64
Exercise_Frequency(week)	float64
Social_Media_Platform	object
Happiness_Index(1-10)	float64

dtype: object

Print columns

```
df.columns
```

```
Index(['User_ID', 'Age', 'Gender', 'Daily_Screen_Time(hrs)',
      'Sleep_Quality(1-10)', 'Stress_Level(1-10)',
      'Days_Without_Social_Media', 'Exercise_Frequency(week)',
      'Social_Media_Platform', 'Happiness_Index(1-10)'],
      dtype='object')
```

✓ Getting the count of missing values

```
df.isnull().sum()
```

	0
User_ID	0
Age	0
Gender	0
Daily_Screen_Time(hrs)	0
Sleep_Quality(1-10)	0
Stress_Level(1-10)	0
Days_Without_Social_Media	0
Exercise_Frequency(week)	0
Social_Media_Platform	0
Happiness_Index(1-10)	0

dtype: int64

✓ Pre-processing on dataset

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for column in df.select_dtypes(include=['object']).columns:
    df[column] = label_encoder.fit_transform(df[column])
df.head()
```

	User_ID	Age	Gender	Daily_Screen_Time(hrs)	Sleep_Quality(1-10)	Stress_Level(1-10)
0	0	44	1	3.1	7.0	6.0
1	1	30	2	5.1	7.0	8.0
2	2	23	2	7.4	6.0	7.0
3	3	36	0	5.7	7.0	8.0
4	4	34	0	7.0	4.0	7.0

Next steps:

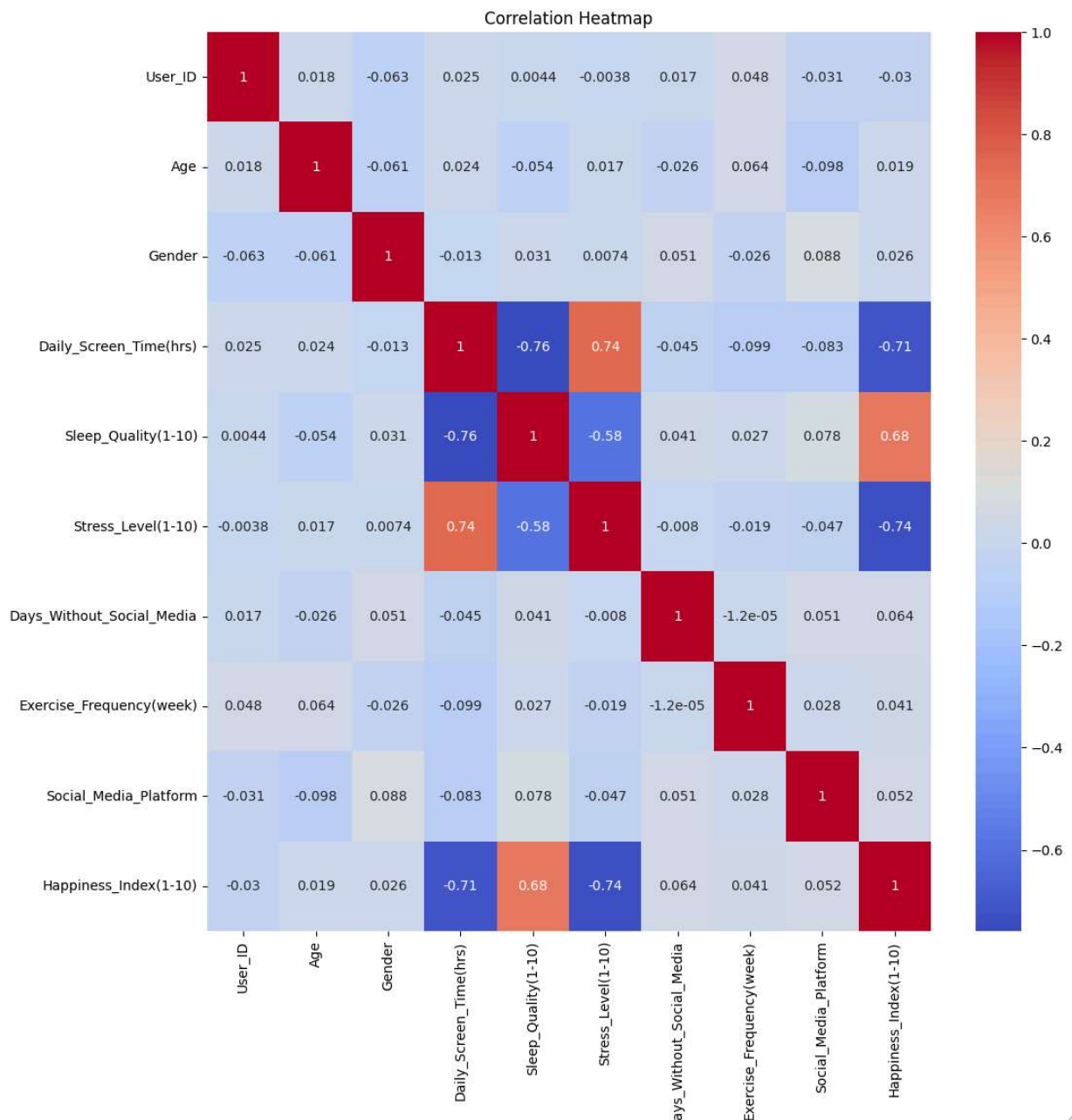
[Generate code with df](#)

[New interactive sheet](#)

✓ Data Visualization

Correlation heatmap

```
plt.figure(figsize=(12, 12))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```



```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
# Assuming df is already loaded from your Mental_Health.csv file

# --- 1. Get the value counts for the selected column ---
gender_counts = df['Gender'].value_counts()
print(gender_counts)
print("\n")

# --- 2. Create the Bar Chart using Seaborn (Recommended) ---
plt.figure(figsize=(8, 5))

# Use countplot for clearer visualization of counts
sns.countplot(x='Gender',
              data=df,
```

```

order=gender_counts.index,
palette=['skyblue', 'pink', 'lightgrey']) # Adjust colors for ger

# Set the title and labels
plt.title('Gender Distribution of Survey Participants', fontsize=15)
plt.xlabel('Gender Category', fontsize=12)
plt.ylabel('Count', fontsize=12)

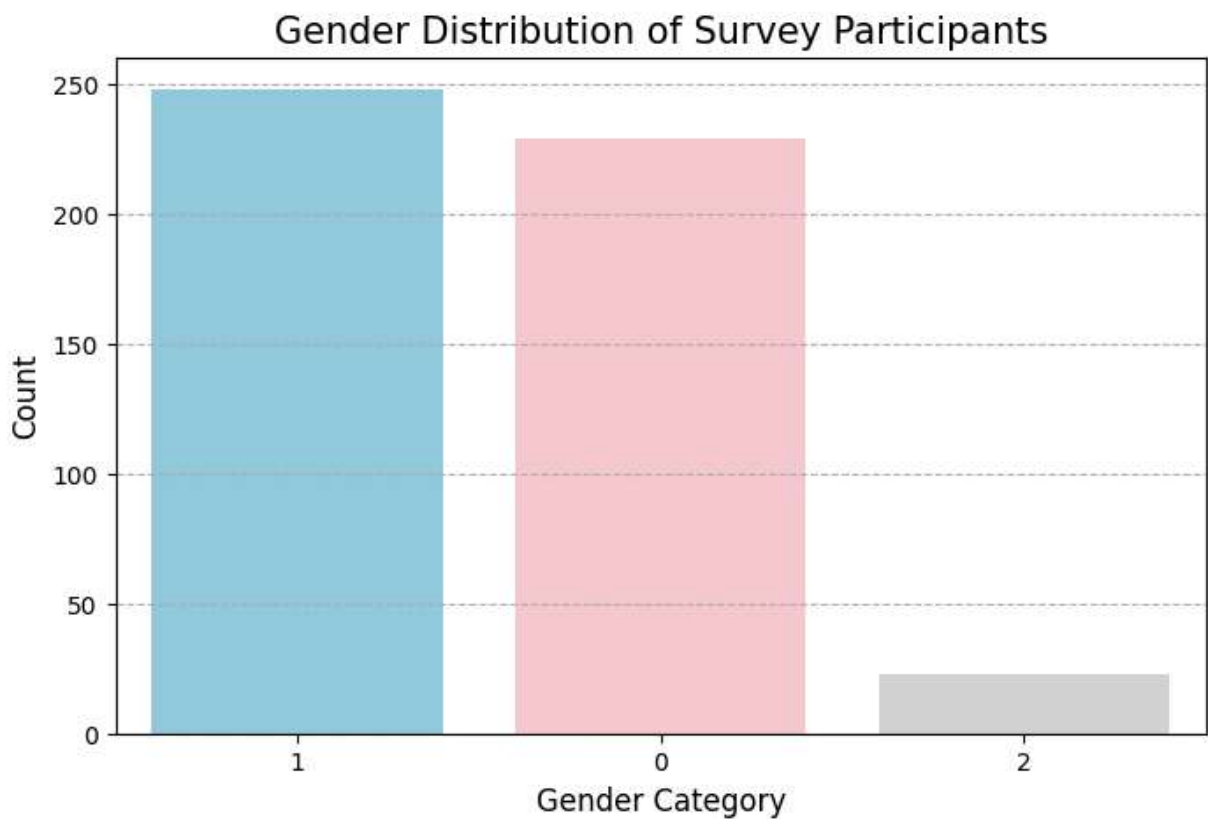
# Optional: Add grid for better readability
plt.grid(axis='y', linestyle='--')
plt.show()

```

```

Gender
1    248
0    229
2     23
Name: count, dtype: int64

```



```

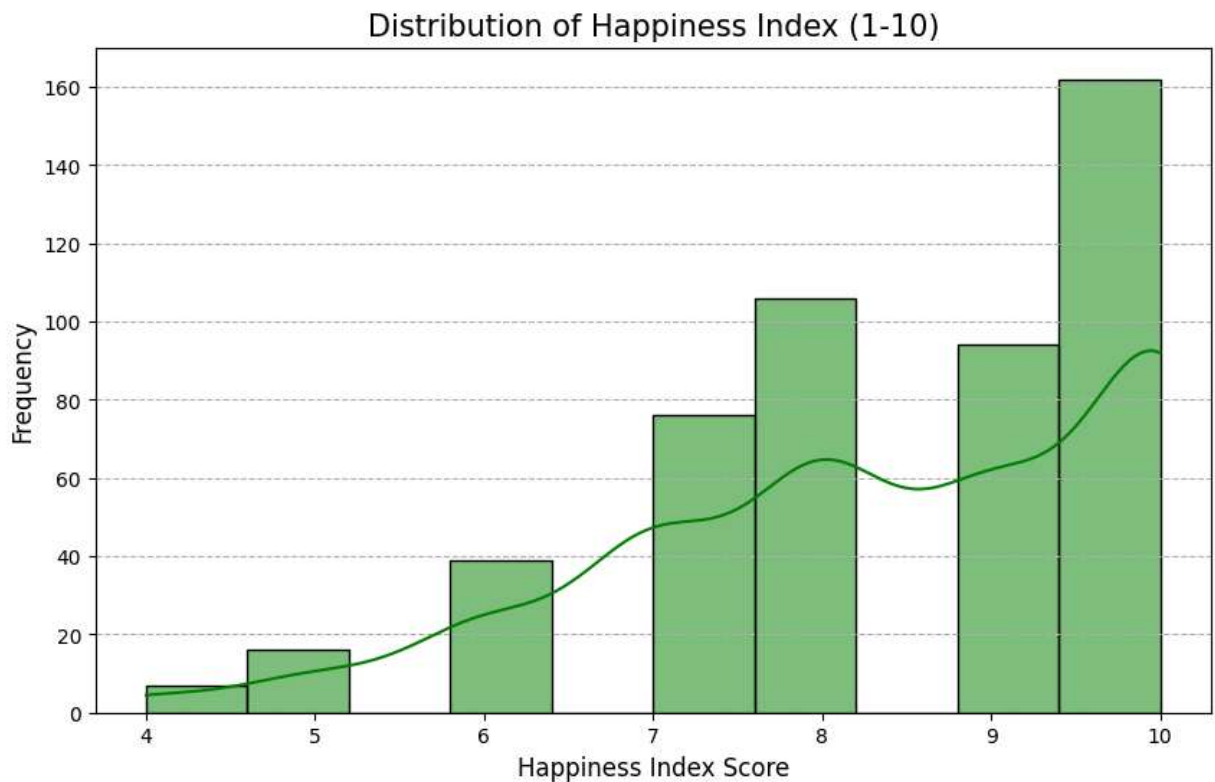
plt.figure(figsize=(10, 6))

sns.histplot(df['Happiness_Index(1-10)'],
             kde=True,
             bins=10,
             color='green',
             edgecolor='black')

plt.title('Distribution of Happiness Index (1-10)', fontsize=15)
plt.xlabel('Happiness Index Score', fontsize=12)

```

```
plt.ylabel('Frequency', fontsize=12)
plt.grid(axis='y', linestyle='--')
plt.show()
```



```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")

# Assuming df is already loaded: df = pd.read_csv('...')
# We will analyze the distribution of the 'Social_Media_Platform' column

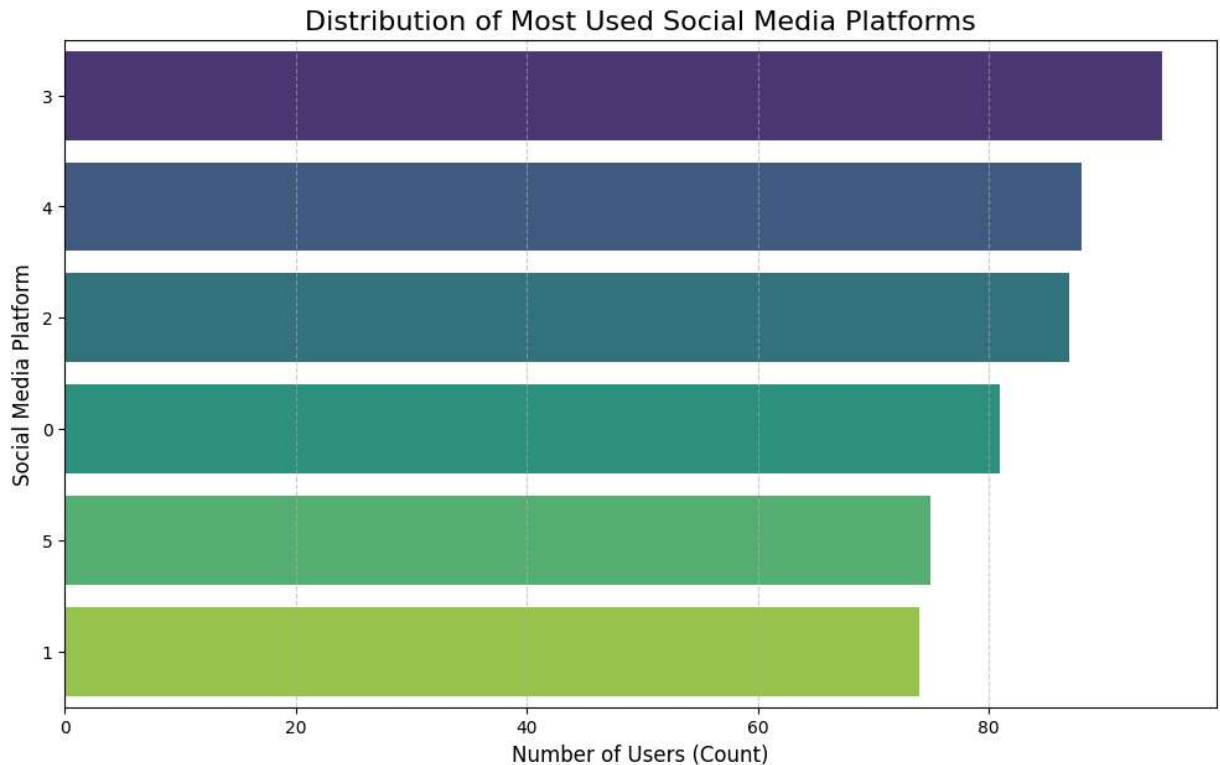
plt.figure(figsize=(12, 7))

# Create the Count Plot (Bar Chart for categorical data)
sns.countplot(y='Social_Media_Platform',
              data=df,
              order=df['Social_Media_Platform'].value_counts().index,
              palette='viridis')

# Set the title and labels
plt.title('Distribution of Most Used Social Media Platforms', fontsize=16)
plt.xlabel('Number of Users (Count)', fontsize=12)
plt.ylabel('Social Media Platform', fontsize=12)

# Add a subtle grid
plt.grid(axis='x', linestyle='--', alpha=0.6)
```

```
plt.show()
```



```
import pandas as pd
from matplotlib import pyplot as plt
import warnings
warnings.filterwarnings("ignore")
# Assuming df is already loaded: df = pd.read_csv('...')

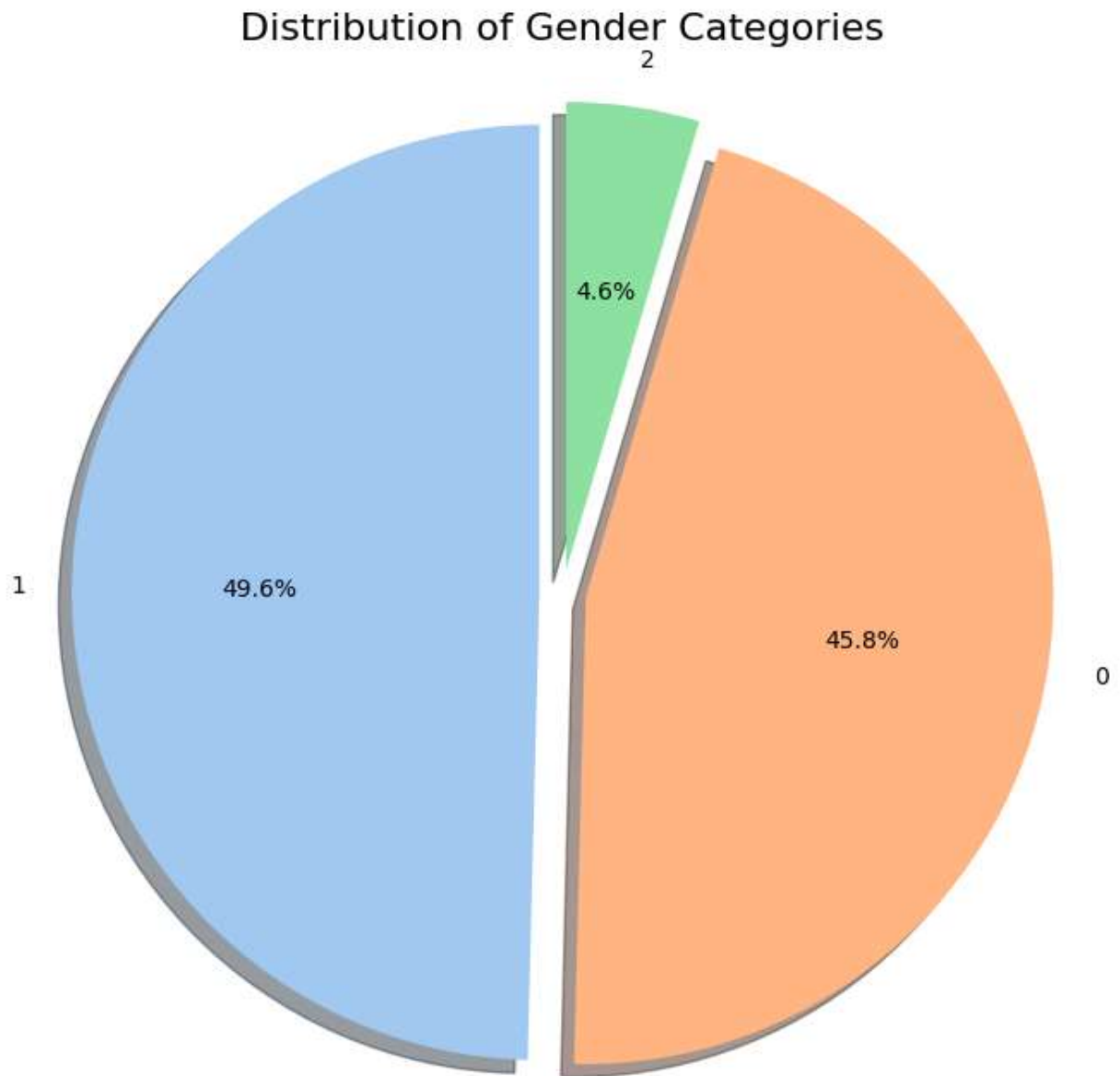
# 1. Prepare data for the Pie Chart
pie_data = df['Gender'].value_counts()
labels = pie_data.index.tolist()
sizes = pie_data.values.tolist()

# 2. Configure the Pie Chart features
# 'explode' is used to slightly offset one slice for emphasis (optional)
# We create an explode list with a small offset for the first category
explode = [0.05] * len(labels)

plt.figure(figsize=(8, 8))

# 3. Create the Pie Chart
plt.pie(sizes,
        labels=labels,
        autopct='%1.1f%%',      # Display the percentage (1 decimal place)
        startangle=90,          # Start the chart from the top
        shadow=True,             # Add a subtle shadow
        explode=explode,         # Apply the explode effect
        colors=sns.color_palette('pastel')) # Use a pastel color scheme
```

```
# 4. Add title and ensure it is a perfect circle
plt.title('Distribution of Gender Categories', fontsize=16)
plt.axis('equal') # Ensures the pie is drawn as a circle, not an ellipse
plt.show()
```



```
import seaborn as sns
from matplotlib import pyplot as plt
# Assuming df is already loaded from your Mental_Health.csv file

# Set the size of the plot
plt.figure(figsize=(8, 5))

# Create the Bar Plot
# x='Gender': The categorical variable on the x-axis
# y='Happiness_Index(1-10)': The quantitative variable for which the average is
# estimator='mean': Calculates the average (mean) for each group
# ci=None: Removes the confidence interval bars for a cleaner look
sns.barplot(x='Gender',
```

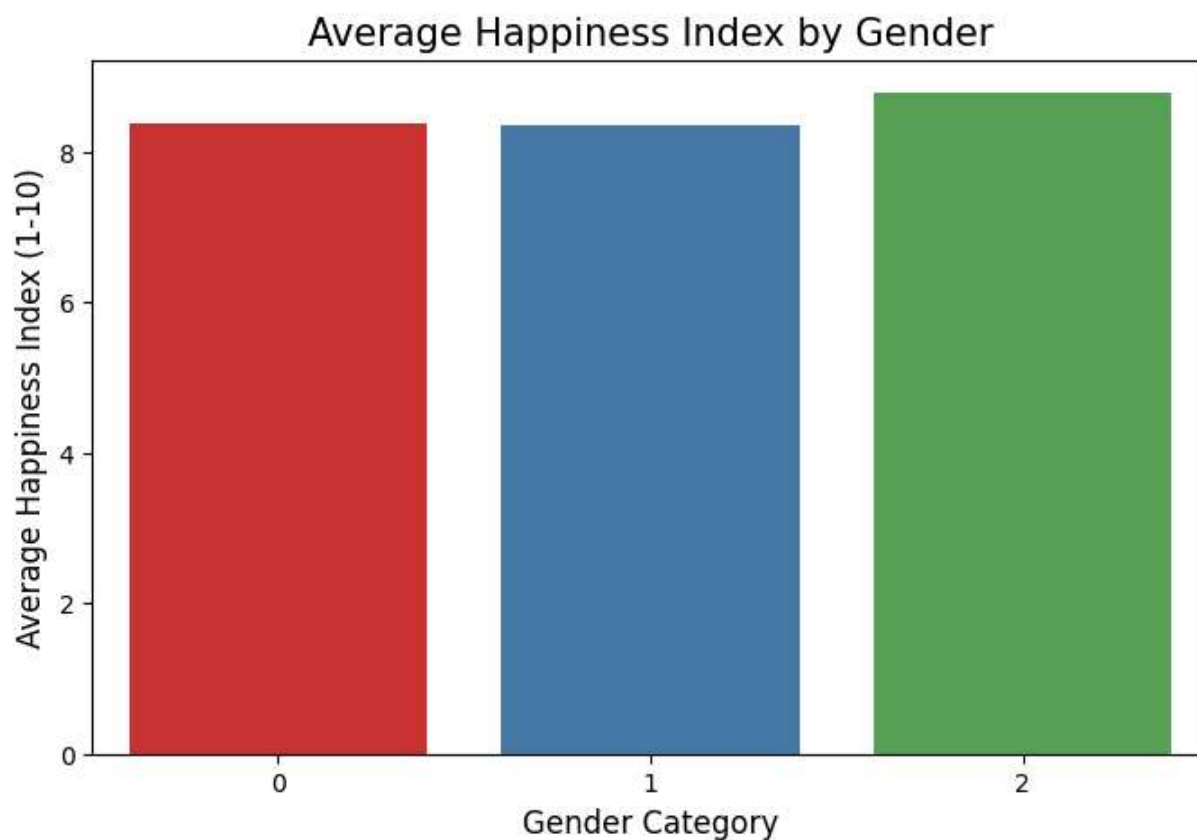
```

y='Happiness_Index(1-10)',
data=df,
estimator='mean',
ci=None,
palette='Set1')

# Set the title and labels
plt.title("Average Happiness Index by Gender", fontsize=15)
plt.xlabel("Gender Category", fontsize=12)
plt.ylabel("Average Happiness Index (1-10)", fontsize=12)

# Display the plot
plt.show()

```



```

import seaborn as sns
from matplotlib import pyplot as plt

plt.figure(figsize=(12, 7))

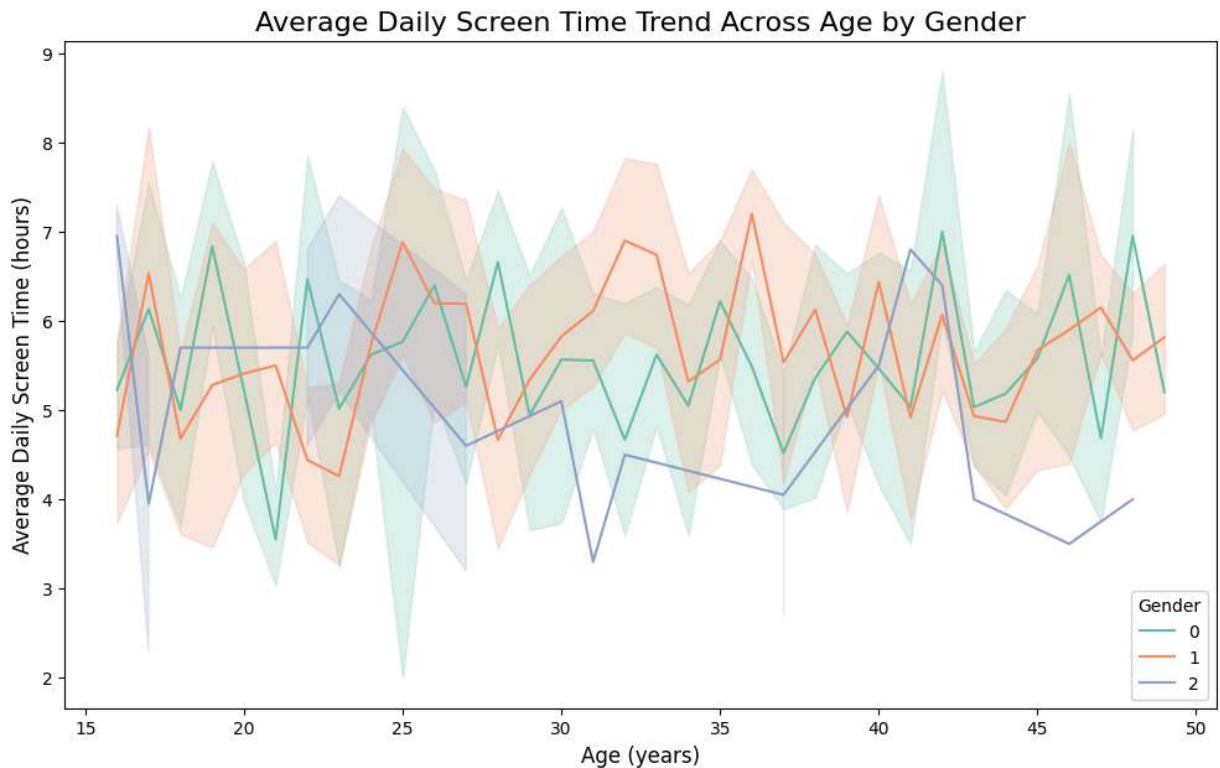
sns.lineplot(x='Age',
             y='Daily_Screen_Time(hrs)',
             hue='Gender',
             data=df,
             palette='Set2')

plt.title("Average Daily Screen Time Trend Across Age by Gender", fontsize=16)
plt.xlabel("Age (years)", fontsize=12)
plt.ylabel("Average Daily Screen Time (hours)", fontsize=12)

```

```
plt.legend(title='Gender')
```

```
plt.show()
```



```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Calculate counts and set configurations
gender_counts = df['Gender'].value_counts()
labels = gender_counts.index.tolist()
colors = ['skyblue', 'lightcoral', 'lightgreen'] # For up to 3 categories

plt.figure(figsize=(16, 8))
plt.suptitle('Gender Distribution', fontsize=16, fontweight='bold')

# --- 1. Donut Chart ---
plt.subplot(1, 2, 1)
plt.title('Donut Chart', fontsize=14)
plt.pie(gender_counts, labels=labels, colors=colors,
        autopct='%.1f%%', startangle=90, textprops={'fontsize': 12})
centre = plt.Circle((0, 0), 0.45, color='white')
plt.gca().add_artist(centre)
plt.axis('equal')

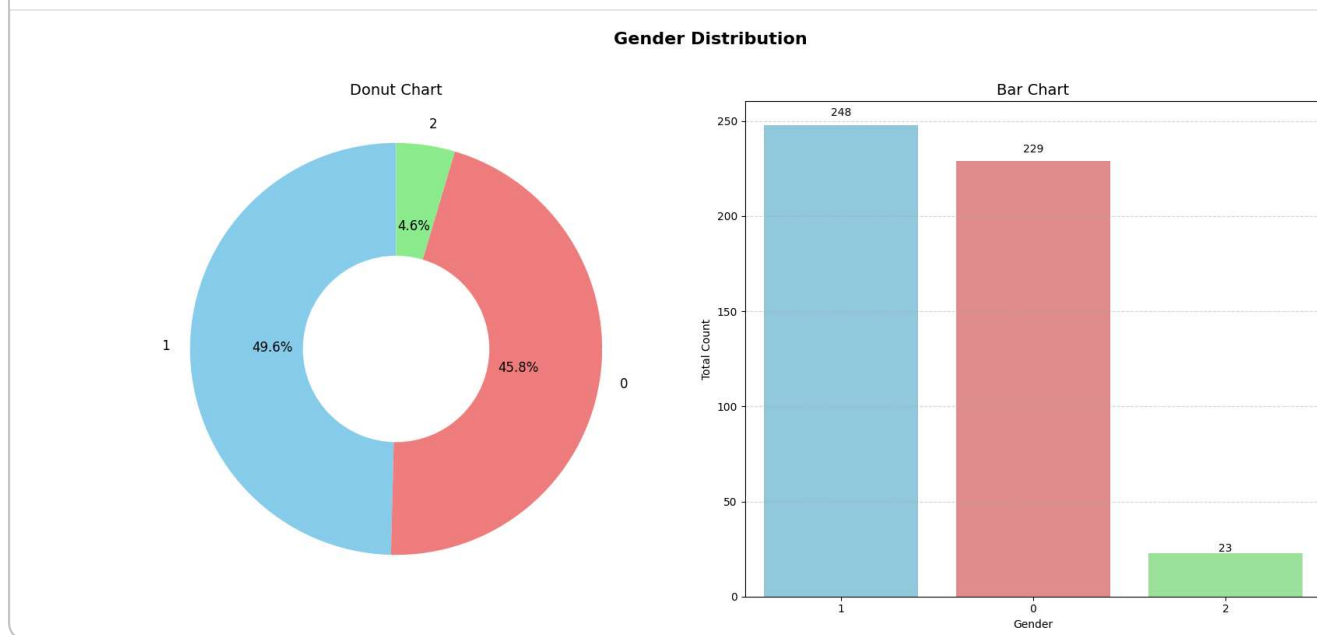
# --- 2. Bar Chart ---
plt.subplot(1, 2, 2)
plt.title('Bar Chart', fontsize=14)
ax = sns.countplot(x='Gender', data=df, palette=colors, order=gender_counts.index)
for bar in ax.patches:
```

```

height = bar.get_height()
ax.text(bar.get_x() + bar.get_width() / 2, height + (height * 0.02),
        f'{int(height)}', ha='center', fontsize=10)
plt.xlabel('Gender')
plt.ylabel('Total Count')
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.tight_layout(rect=[0, 0, 1, 0.96])

plt.show()

```



```

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Assuming df is already loaded from your Mental_Health.csv file

# --- 1. Explicitly Define Categorical Columns and Ensure Dtype ---
# Define the names of the columns that MUST be treated as categorical
explicit_cat_cols = ['Gender', 'Social_Media_Platform']

# Convert these columns to the 'object' dtype explicitly if they are not already
for col in explicit_cat_cols:
    if col in df.columns:
        df[col] = df[col].astype('object')

# Re-define categorical columns based on the corrected dtypes
categorical_cols = [col for col in df.columns if df[col].dtype == 'object']

# --- 2. Create the Multi-Plot Figure ---
if not categorical_cols:
    print("Error: After conversion, still no suitable categorical columns found")
else:
    num_plots = len(categorical_cols)
    num_cols_per_row = 2
    num_rows = (num_plots + num_cols_per_row - 1) // num_cols_per_row

```

```

plt.figure(figsize=(15, 6 * num_rows))

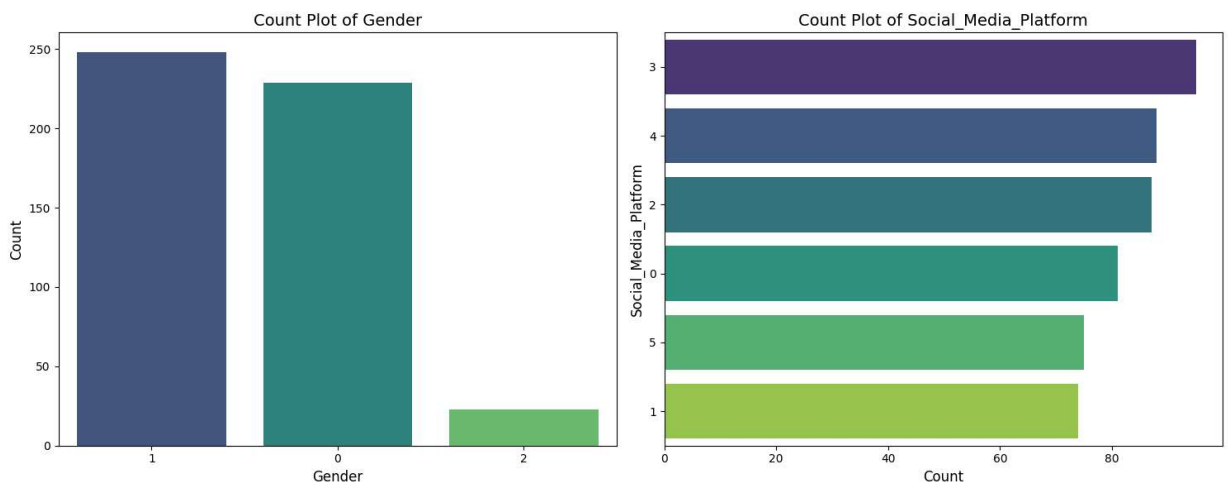
for i, col in enumerate(categorical_cols):
    plt.subplot(num_rows, num_cols_per_row, i + 1)

    # Use horizontal bars for long labels (more than 5 unique values)
    if df[col].nunique() > 5 and len(df[col].unique()) > 1:
        sns.countplot(y=col, data=df, palette='viridis',
                      order=df[col].value_counts().index)
        plt.xlabel('Count', fontsize=12)
        plt.ylabel(col, fontsize=12)
    else:
        sns.countplot(x=col, data=df, palette='viridis',
                      order=df[col].value_counts().index)
        plt.xlabel(col, fontsize=12)
        plt.ylabel('Count', fontsize=12)

    plt.title(f'Count Plot of {col}', fontsize=14)

plt.tight_layout()
plt.show()

```



```

import matplotlib.pyplot as plt
import seaborn as sns
# Assuming df is already loaded from your Mental_Health.csv file

plt.figure(figsize=(10, 6))

# Use Daily_Screen_Time(hrs) for X-axis and Happiness_Index(1-10) for Y-axis
plt.scatter(df["Daily_Screen_Time(hrs)"],
            df["Happiness_Index(1-10)"],
            color='purple',
            alpha=0.7) # Added color and alpha for better visuals

plt.title("Daily Screen Time vs. Happiness Index", fontsize=15)
plt.xlabel("Daily Screen Time (hours)", fontsize=12)

```

```
plt.ylabel("Happiness Index (1-10)", fontsize=12)
```

```
plt.tight_layout()
```

```
plt.show()
```



```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
# Assuming df is already loaded from your Mental_Health.csv file
```

```
plt.figure(figsize=(8, 6))
```

```
# Use seaborn boxplot for easier plotting with DataFrames
```

```
# x='Gender': The categorical variable
```

```
# y='Stress_Level(1-10)': The quantitative variable
```

```
sns.boxplot(x='Gender',
            y='Stress_Level(1-10)',
            data=df,
            palette='coolwarm') # Using a divergent color palette
```

```
# Set the title and labels
```

```
plt.title("Stress Level Distribution by Gender", fontsize=15)
```

```
plt.xlabel("Gender Category", fontsize=12)
```

```
plt.ylabel("Stress Level (1-10)", fontsize=12)
```

```
plt.tight_layout()
```

```
plt.show()
```

