WEEK 3

Univariate and Bivariate Visual Analysis

```
from google.colab import files
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from IPython.display import display, Markdown
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
plt.rcParams["figure.dpi"] = 110
uploaded = files.upload()
df = pd.read_csv(list(uploaded.keys())[0])
```

```
Choose Files Indian_Kids...cleaned.csv
Indian_Kids_Screen_Time_cleaned.csv(text/csv) - 850041 bytes, last modified: 6/10/2025 - 100% done
Saving Indian_Kids_Screen_Time_cleaned.csv to Indian_Kids_Screen_Time_cleaned.csv
```

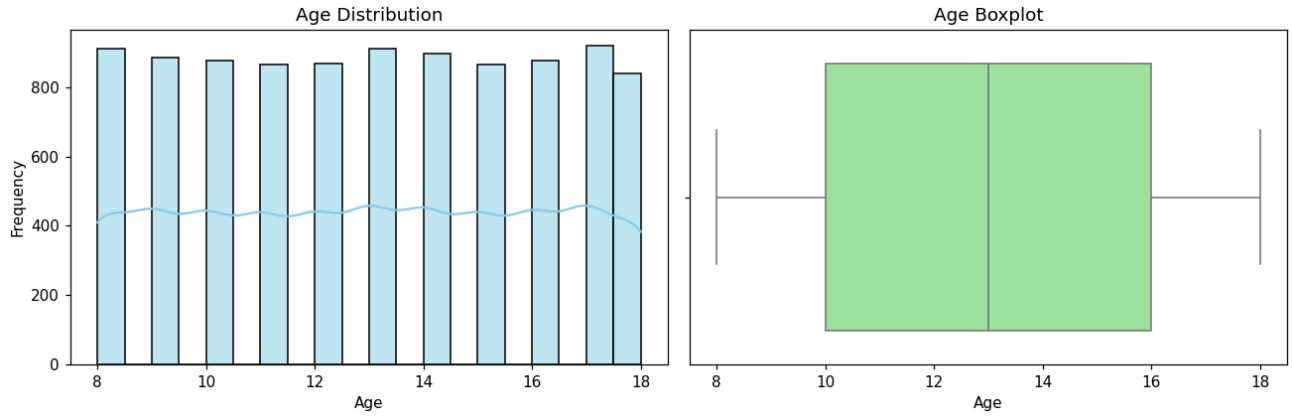
Univariate Analysis: Numerical Columns

```
num_cols = df.select_dtypes(include=['int64','float64']).columns.tolist()
cat_cols = df.select_dtypes(include=['object','category']).columns.tolist()
```

```
def print_heading(text):
    display(Markdown(f"## {text}"))
```

```
def print_heading(text):
    display(Markdown(f"## {text}"))
for col in num_cols:
    print_heading(f"Numerical Column: {col}")
    plt.figure(figsize=(12,4))

    # Histogram with KDE
    plt.subplot(1,2,1)
    sns.histplot(df[col], kde=True, color='skyblue', bins=20)
    plt.title(f"{col} Distribution")
    plt.xlabel(col)
    plt.ylabel("Frequency")

    # Boxplot
    plt.subplot(1,2,2)
    sns.boxplot(x=df[col], color='lightgreen')
    plt.title(f"{col} Boxplot")

    plt.tight_layout()
    plt.show()

    # Display basic descriptive statistics
    display(df[col].describe())
```

Numerical Column: Age



	Age
count	9712.000000
mean	12.979201
std	3.162437
min	8.000000
25%	10.000000
50%	13.000000
75%	16.000000
max	18.000000

dtype: float64

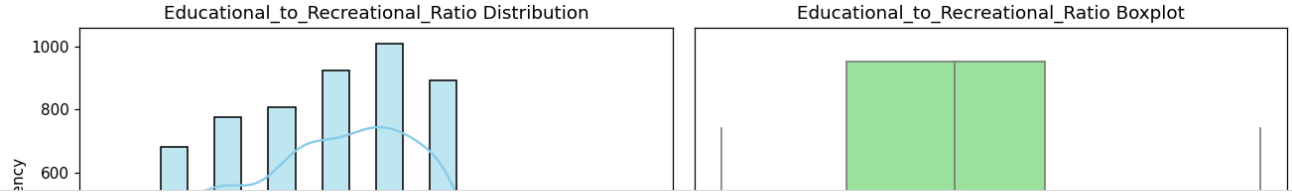
Numerical Column: Avg_Daily_Screen_Time_hr

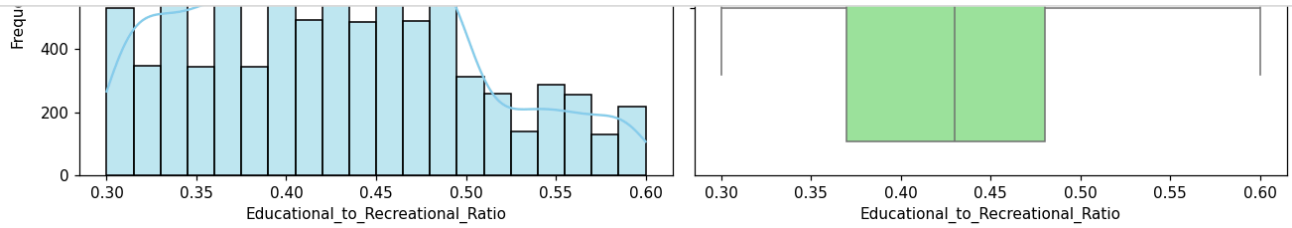

```
for col in cat_cols:
    if col == "Health_Impacts":
        continue  # Skip this column
    print_heading(f"Categorical Column: {col}")
    counts = df[col].value_counts()
    plt.figure(figsize=(12,5))
    # Bar chart
    plt.subplot(1,2,1)
    sns.barplot(x=counts.index, y=counts.values, palette='pastel')
    plt.title(f"{col} Countplot")
    plt.xticks(rotation=45)
    # Pie chart
    plt.subplot(1,2,2)
    plt.pie(counts.values, labels=counts.index, autopct='%1.1f%%', colors=plt.cm.Pastel1.colors)
    plt.title(f"{col} Pie Chart")
    plt.tight_layout()
    plt.show()
    # Display value counts
    display(counts)
```

25%	3.410000
50%	4.440000
75%	5.380000
max	13.890000

dtype: float64

Numerical Column: Educational_to_Recreational_Ratio

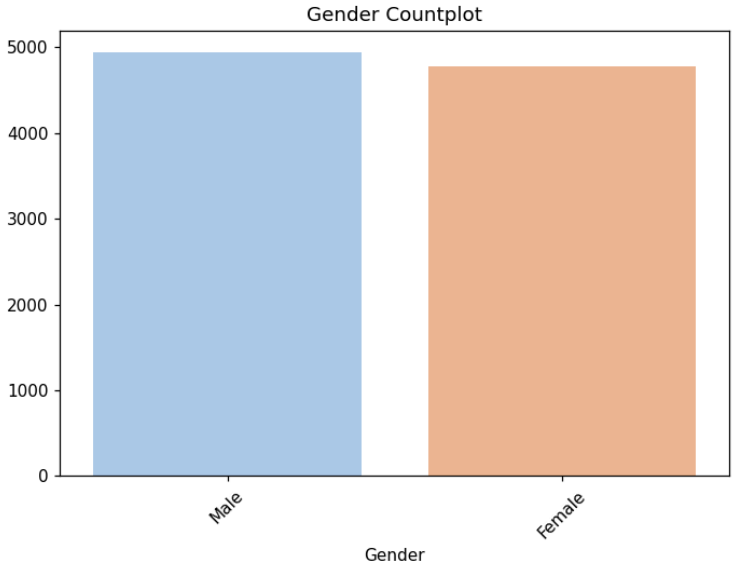


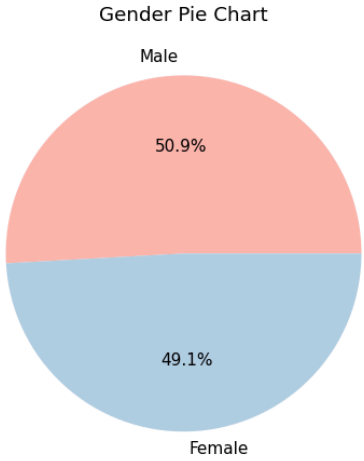


	Educational_to_Recreational_Ratio
count	9712.000000
mean	0.427226
std	0.073221
min	0.300000
25%	0.370000
50%	0.430000
75%	0.480000
max	0.600000

dtype: float64

Categorical Column: Gender

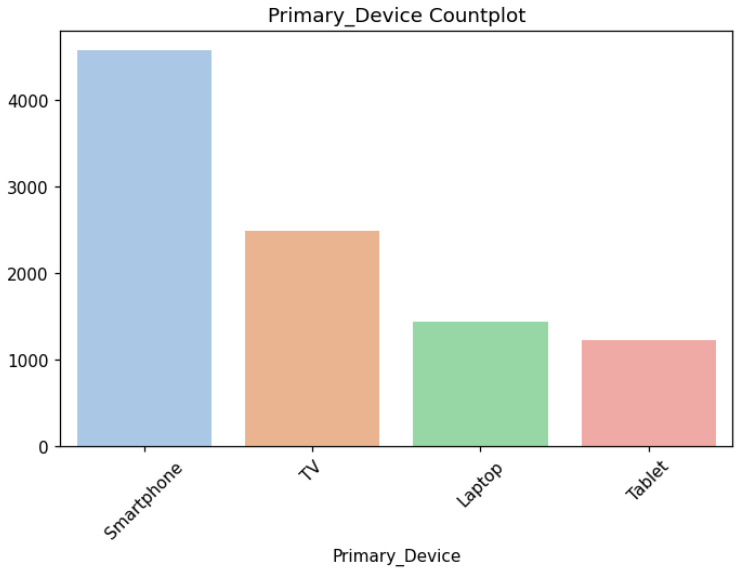


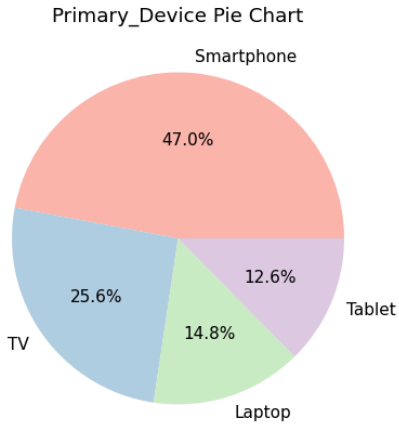


Gender	count
Male	4942
Female	4770

dtype: int64

Categorical Column: Primary_Device

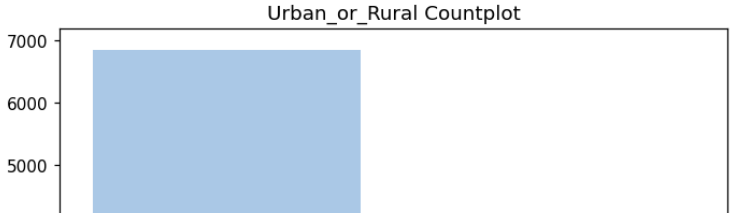


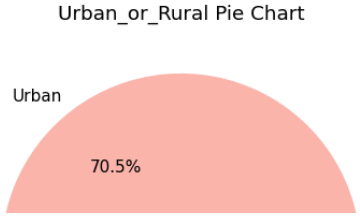


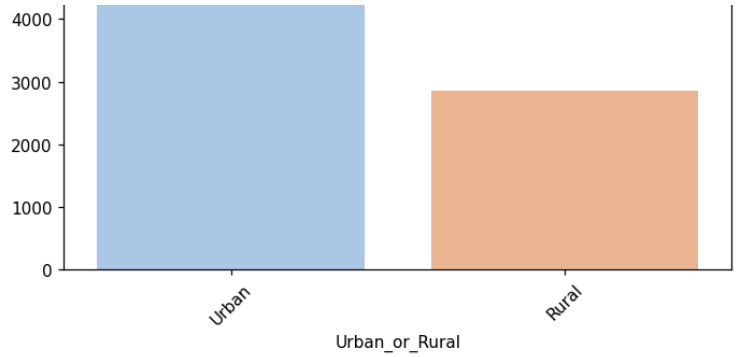
Primary_Device	count
Smartphone	4568
TV	2487
Laptop	1433
Tablet	1224

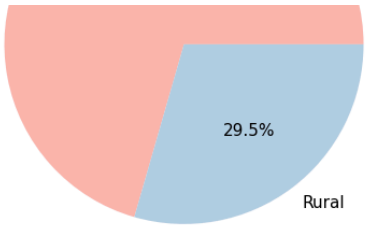
dtype: int64

Categorical Column: Urban_or_Rural





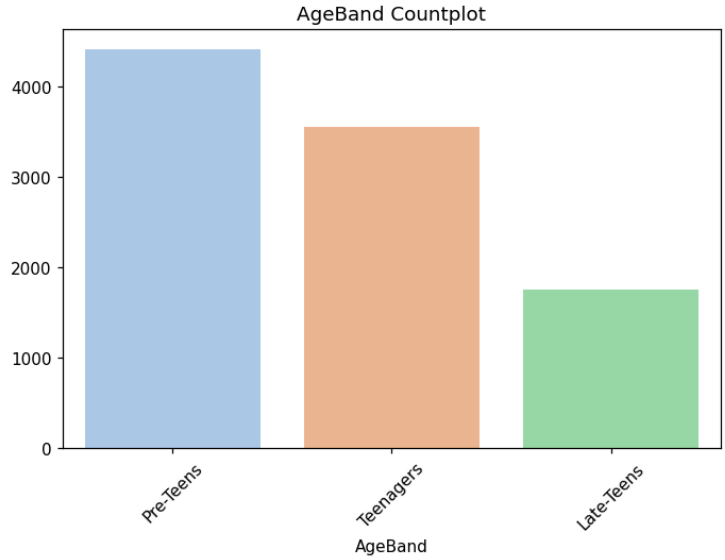


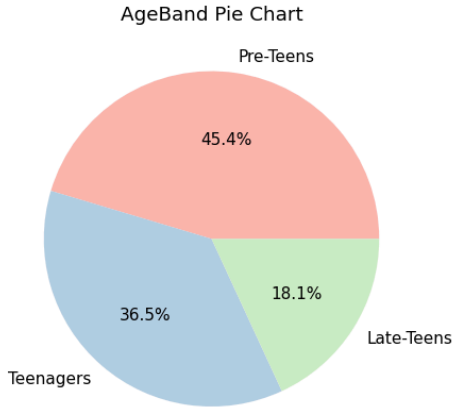


Urban_or_Rural	count
Urban	6851
Rural	2861

dtype: int64

Categorical Column: AgeBand

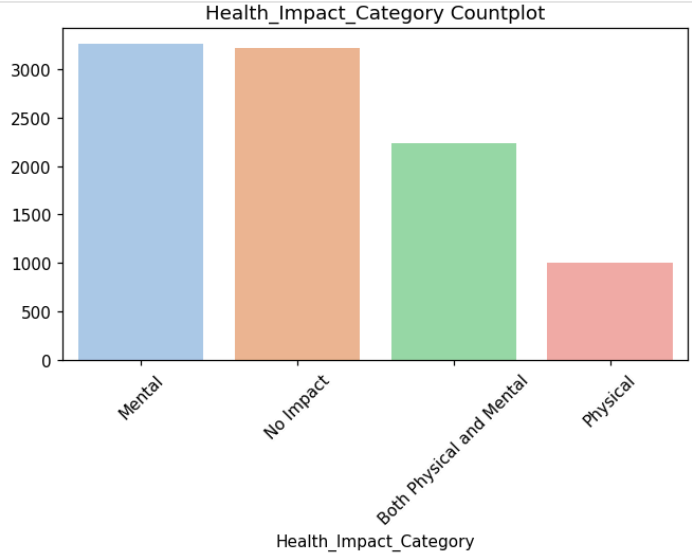


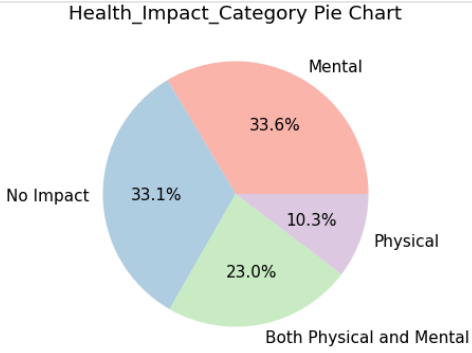


AgeBand	count
Pre-Teens	4407
Teenagers	3546
Late-Teens	1759

dtype: int64

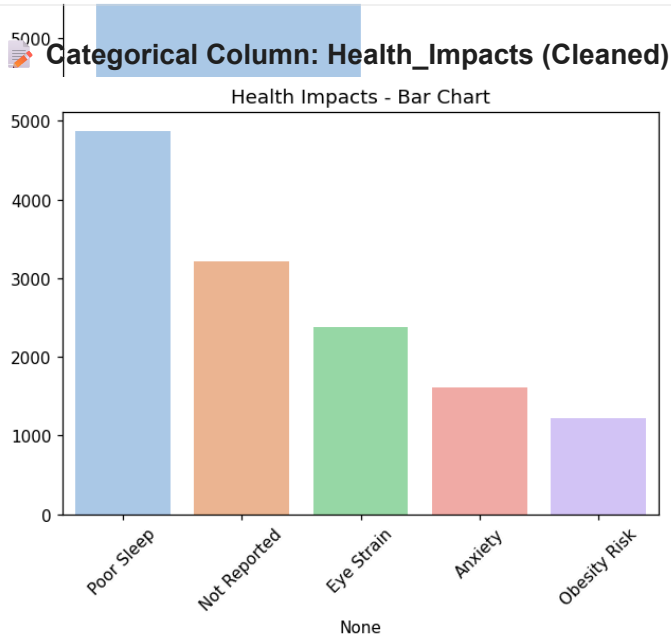
Categorical Column: Health_Impact_Category



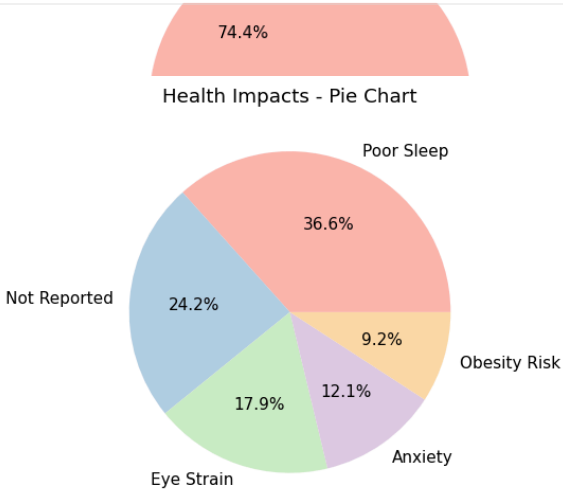


```
# Split multiple impacts into individual categories
impact_counts = df['Health_Impacts'].str.get_dummies(sep=', ').sum().sort_values(ascending=False)
# Heading
display(Markdown("## 📝 Categorical Column: Health_Impacts (Cleaned)"))
# Bar chart
plt.figure(figsize=(12,5))
plt.subplot(1,2,1)
sns.barplot(x=impact_counts.index, y=impact_counts.values, palette='pastel')
plt.title("Health Impacts - Bar Chart")
plt.xticks(rotation=45)
# Pie chart
plt.subplot(1,2,2)
plt.pie(impact_counts.values, labels=impact_counts.index, autopct='%1.1f%%', colors=plt.cm.Pastel1.colors)
plt.title("Health Impacts - Pie Chart")
plt.tight_layout()
plt.show()
# Display counts
display(impact_counts)
```

📝 Categorical Column: Health_Impacts (Cleaned)



None



dtype: int64

	0
Poor Sleep	4868
Not Reported	3218
Eye Strain	2382
Anxiety	1605
Obesity Risk	1217

dtype: int64

Bivarant analaysis

```
cat_cols = ['Gender', 'AgeBand', 'Urban_or_Rural']
target_col = 'Avg_Daily_Screen_Time_hr'
for col in cat_cols:
    print_heading(f"Screen Time vs {col}")

    # Bar chart: Average screen time per category
    avg_screen = df.groupby(col)[target_col].mean()
    plt.figure(figsize=(6,4))
    sns.barplot(x=avg_screen.index, y=avg_screen.values, palette='pastel')
    plt.title(f"Average Screen Time by {col}")
    plt.ylabel("Avg Daily Screen Time (hrs)")
    plt.show()
    # Boxplot: Spread and outliers
    plt.figure(figsize=(6,4))
```

```
    sns.boxplot(x=col, y=target_col, data=df, palette='pastel')
    plt.title(f"Screen Time Boxplot by {col}")
    plt.ylabel("Avg Daily Screen Time (hrs)")
    plt.show()
```