INFOSYS SPRINGBOARD

kids' screentime patterns to uncover using data ***VISUALIZATION***

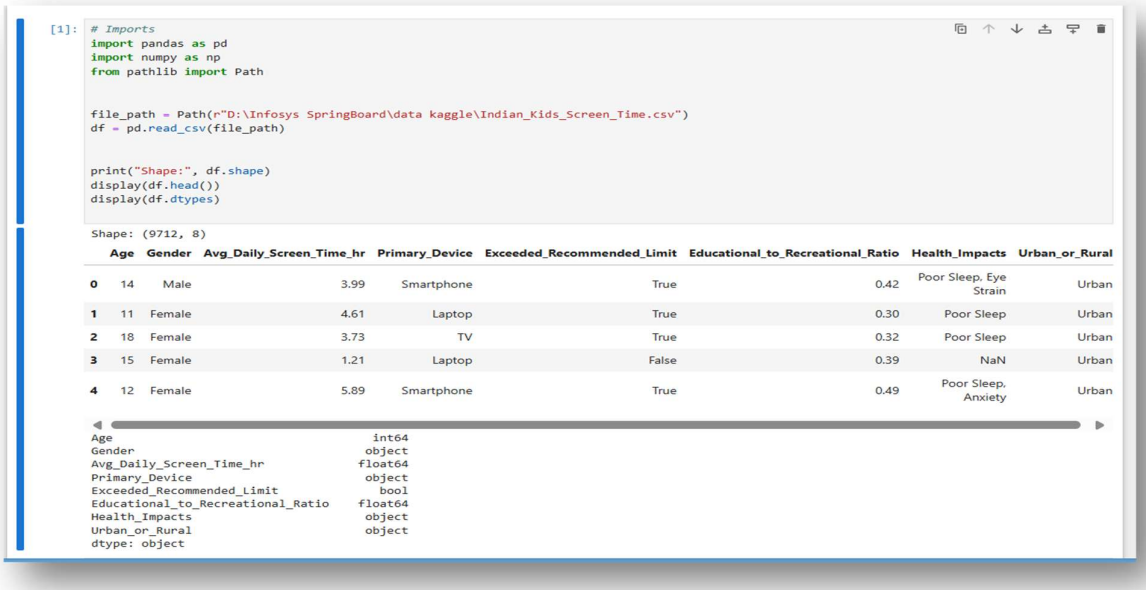
Problem Statement:

Analyse kids' screentime patterns to uncover trends by age, gender, location type (urban/rural), device type, day-of-week, and activity category using data visualization. The goal is to present clear, actionable insights for parents, educators, and policymakers.

Load the dataset

Source: Kaggle — Indian Kids Screentime 2025

https://www.kaggle.com/datasets/ankushpanday2/indian-kids-screentime-2025



```
[1]: # Imports
import pandas as pd
import numpy as np
from pathlib import Path


file_path = Path(r"D:\Infosys SpringBoard\data kaggle\Indian_Kids_Screen_Time.csv")
df = pd.read_csv(file_path)


print("Shape:", df.shape)
display(df.head())
display(df.dtypes)
```

```
Shape: (9712, 8)
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_Impacts	Urban_or_Rural
0	14	Male	3.99	Smartphone	True	0.42	Poor Sleep, Eye Strain	Urban
1	11	Female	4.61	Laptop	True	0.30	Poor Sleep	Urban
2	18	Female	3.73	TV	True	0.32	Poor Sleep	Urban
3	15	Female	1.21	Laptop	False	0.39	NaN	Urban
4	12	Female	5.89	Smartphone	True	0.49	Poor Sleep, Anxiety	Urban

```
Age                                int64
Gender                            object
Avg_Daily_Screen_Time_hr         float64
Primary_Device                    object
Exceeded_Recommended_Limit          bool
Educational_to_Recreational_Ratio float64
Health_Impacts                    object
Urban_or_Rural                    object
dtype: object
```

Explored schema, data types, size, and nulls

Age	Numeric (8-18)
Gender	male or female
Avg_Daily_Screen_Time_hr	hrs:mins
Primary_Device	smartphone, laptop, tv, tablet
Exceeded_Recommended_Limit	Boolean (TRUE / FALSE)
Educational_to_Recreational_Ratio	Decimal
Health_Imapacts	String ▪ Poor sleep ▪ Anxiety ▪ Eye strain ▪ Obesity risk
Urban_or_Rural	String (urban / rural)

Data Cleaning & Preprocessing

- Removed Duplicate rows (from 9712 to 9670)

```
[2]: # Remove exact duplicate rows
df = df.drop_duplicates(keep='first')
print("After dropping exact duplicates, shape:", df.shape)
df.head()
```

```
After dropping exact duplicates, shape: (9668, 8)
```

[2]:

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_Impacts	Urban_or_Rural
0	14	Male	3.99	Smartphone	True	0.42	Poor Sleep, Eye Strain	Urban
1	11	Female	4.61	Laptop	True	0.30	Poor Sleep	Urban
2	18	Female	3.73	TV	True	0.32	Poor Sleep	Urban
3	15	Female	1.21	Laptop	False	0.39	NaN	Urban
4	12	Female	5.89	Smartphone	True	0.49	Poor Sleep, Anxiety	Urban

- Changed datatype of age from String to Numeric

```
Trim whitespace and normalize string columns
str_cols = df.select_dtypes(include=['object']).columns.tolist()
for c in str_cols:
    df[c] = df[c].astype(str).str.strip().replace({'nan': np.nan})
    df[c] = df[c].where(df[c].isna(), df[c].str.lower())


if 'gender' in df.columns:
    gender_map = {
        'boy': 'male', 'm': 'male', 'male': 'male',
        'girl': 'female', 'f': 'female', 'female': 'female'
    }
    df['gender'] = df['gender'].map(gender_map).fillna(df['gender'])  # keep others unchanged
    print("Gender value counts after mapping:")
    display(df['gender'].value_counts(dropna=False))
df.head()
```

	Age	Gender	Avg_Daily_Screen_Time_hr	Primary_Device	Exceeded_Recommended_Limit	Educational_to_Recreational_Ratio	Health_Impacts	Urban_or_Rural
0	14	male	3.99	smartphone	True	0.42	poor sleep, eye strain	urban
1	11	female	4.61	laptop	True	0.30	poor sleep	urban
2	18	female	3.73	tv	True	0.32	poor sleep	urban
3	15	female	1.21	laptop	False	0.39	NaN	urban
4	12	female	5.89	smartphone	True	0.49	poor sleep, anxiety	urban

- Converted daily_avg_screen_time from decimal to hhmm

```
def decimal_hours_to_hhmm(decimal_hours):
    if pd.isna(decimal_hours):
        return np.nan
    try:
        total_minutes = int(round(decimal_hours * 60))
        hours = total_minutes // 60
        minutes = total_minutes % 60
        return f"{hours:d}:{minutes:02d}"
    except Exception:
        return np.nan

if 'Avg_Daily_Screen_Time_hr' in df.columns:

    original_values = df['Avg_Daily_Screen_Time_hr'].copy()


    df['Avg_Daily_Screen_Time_hr'] = df['Avg_Daily_Screen_Time_hr'].apply(decimal_hours_to_hhmm)

    print("Screen time converted from decimal hours to hh:mm format:")
    display(df[['Avg_Daily_Screen_Time_hr']].head(10))
```

```
Screen time converted from decimal hours to hh:mm format:
```

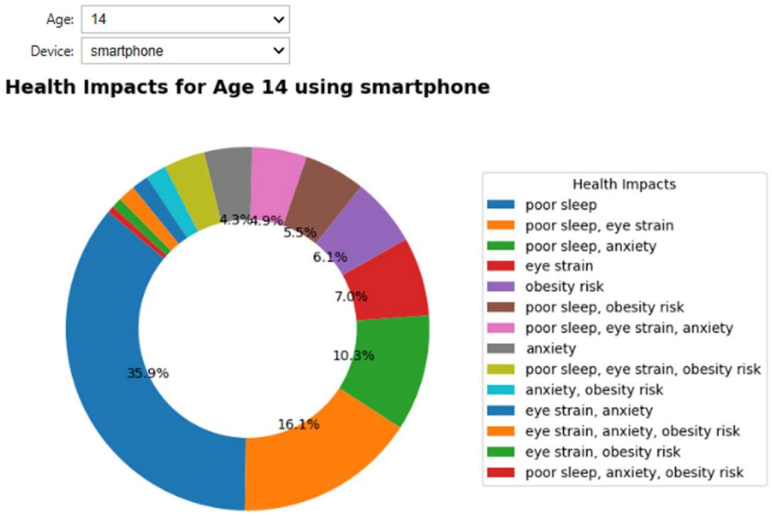
	Avg_Daily_Screen_Time_hr
0	3:59
1	4:37
2	3:44
3	1:13
4	5:53
5	4:53
6	2:58
7	2:44
8	4:37

Goals and workflow:

Parents:

Data-driven insights to help parents make informed decisions about their children's screen time habits and create healthier digital environments at home. Some are listed below with examples, and more will be added.

- Health Impacts, Age, Primary device as Pie Chart



- Educational_to_Recreational_Ratio & Avg_Daily_Screen_Time_hr
- Primary_device with Avg_Daily_Screen_Time_hr

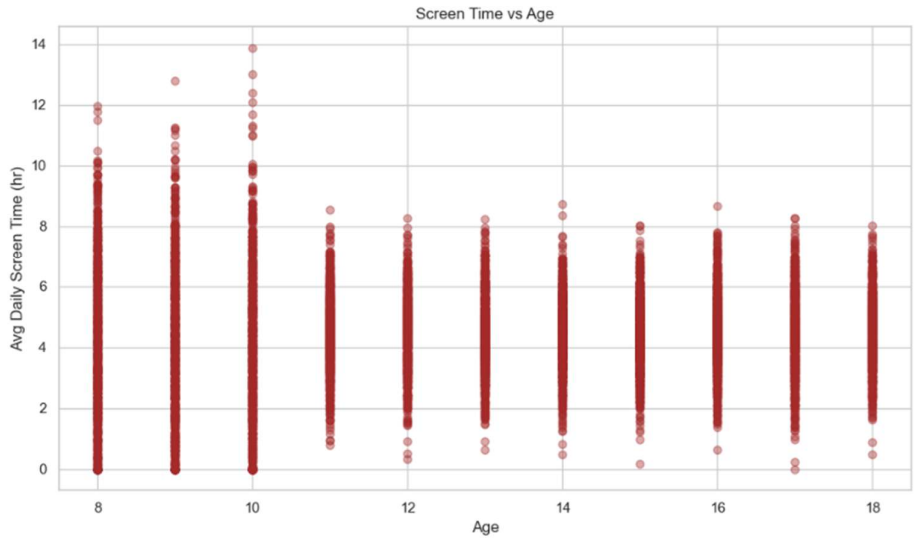
Educator:

Provide educators with insights into how students use their primary device for education.

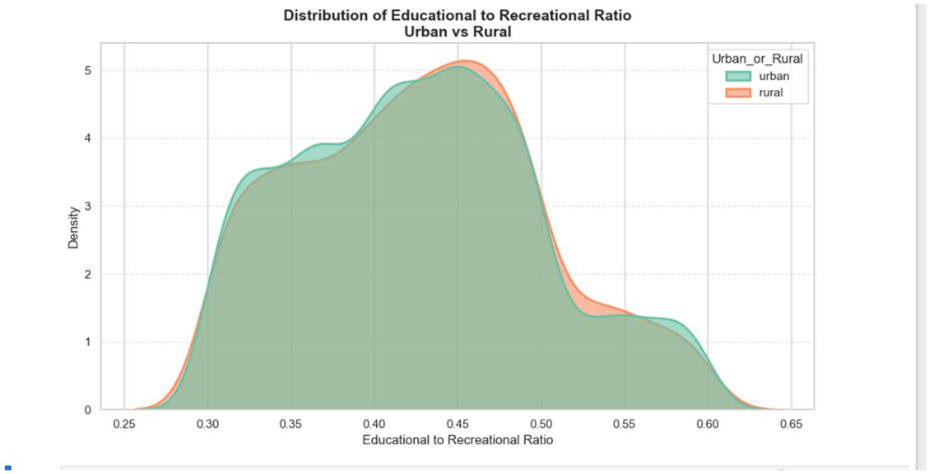
- Screentime vs. Age Correlation

Educator

```
[52]: #Scatter Plot for Screen time and age

[55]:
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.scatter(df['Age'], df['Avg_Daily_Screen_Time_hr'], alpha=0.4, color='brown')
plt.xlabel('Age')
plt.ylabel('Avg Daily Screen Time (hr)')
plt.title('Screen Time vs Age')
plt.grid(True)
plt.tight_layout()
plt.show()
```



- Educational_to_Recreational_Ratio of urban and rural

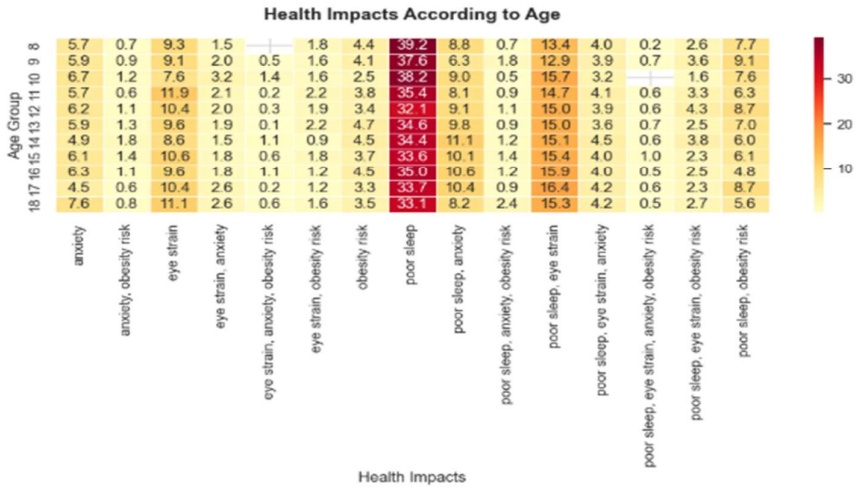


Policymaker:

Provide evidence-based insights to inform public health initiatives, educational policies and health impacts. MORE WILL BE ADDED FURTHER

- Health impacts according to age

```
[62]: #Public Health Priority Matrix

[65]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

heatmap_data = (
    df.groupby(['Age', 'Health_Impacts'])
      .size()
      .reset_index(name='Count')
)

heatmap_data['Percentage'] = heatmap_data.groupby('Age')['Count'].transform(lambda x: 100 * x / x.sum())

pivot_table = heatmap_data.pivot(index='Age', columns='Health_Impacts', values='Percentage')

plt.figure(figsize=(10, 6))
sns.heatmap(pivot_table, annot=True, fmt=".1f", cmap='YlOrRd', linewidths=0.5)

plt.title('Health Impacts According to Age', fontsize=14, fontweight='bold', pad=15)
plt.xlabel('Health Impacts', fontsize=12)
plt.ylabel('Age Group', fontsize=12)
plt.tight_layout()
plt.show()
```



- Screentime vs age