In [1]:
```
import pandas as pd
```

In [5]:
```
pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
```

Out[5]:

	ID	gender	minority	deprived
0	1087360	0	0	0
1	1088938	0	1	0
2	1088953	0	1	0
3	1088961	0	1	0
4	1089147	0	1	0
...	...	...	...	...
120110	1091072	1	0	1
120111	1091077	1	0	1
120112	1091104	1	0	1
120113	1091112	1	0	1
120114	1091115	1	0	1

120115 rows × 4 columns

In [6]:
```
pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
```

Out[6]:

	ID	C_we	C_wk	G_we	G_wk	S_we	S_wk	T_we	T_wk
0	1000002	0.5	0.5	0.0	0.0	1.0	0.5	1.0	0.5
1	1000003	1.0	0.5	0.0	0.0	2.0	2.0	3.0	2.0
2	1000004	3.0	1.0	2.0	0.0	3.0	2.0	3.0	2.0
3	1000005	4.0	2.0	0.0	0.0	4.0	3.0	7.0	5.0
4	1000006	1.0	0.5	0.0	0.0	2.0	2.0	1.0	0.5
...	...	...	...	...	...	...	...	...	...
113354	1120111	7.0	6.0	7.0	6.0	3.0	1.0	3.0	2.0
113355	1120112	3.0	4.0	7.0	7.0	6.0	7.0	7.0	7.0
113356	1120113	2.0	0.0	4.0	2.0	0.0	0.0	4.0	3.0
113357	1120114	4.0	2.0	5.0	3.0	0.5	0.5	7.0	3.0
113358	1120115	0.0	0.0	7.0	6.0	0.0	0.0	0.0	0.0

113359 rows × 9 columns

In [7]:
```
pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")
```

Out[7]:

	ID	Optm	Usef	Relx	Intp	Engs	Dealpr	Thcklr	Goodme	Clsep	Conf	M
0	1087360	5	3	2	1	3	5	4	1	5	2	
1	1094049	2	3	3	2	3	1	2	1	5	1	
2	1094067	4	3	4	4	4	4	4	3	4	4	
3	1097484	3	3	3	3	3	3	3	3	3	3	
4	1102259	5	4	3	5	2	3	4	4	4	4	
...	...	...	...	...	...	...	...	...	...	...	...	...
102575	1091072	5	5	2	1	5	3	2	1	1	2	
102576	1091077	3	2	4	4	4	4	2	2	4	2	
102577	1091104	2	1	4	4	5	4	4	4	4	4	
102578	1091112	3	2	4	3	4	3	4	4	4	3	
102579	1091115	4	3	2	4	2	4	5	2	5	3	

102580 rows × 15 columns

In [3]:
```
import pandas as pd
```

In [11]:
```
pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
```

Out[11]:

	ID	gender	minority	deprived
0	1087360	0	0	0
1	1088938	0	1	0
2	1088953	0	1	0
3	1088961	0	1	0
4	1089147	0	1	0
...	...	...	...	...
120110	1091072	1	0	1
120111	1091077	1	0	1
120112	1091104	1	0	1
120113	1091112	1	0	1
120114	1091115	1	0	1

120115 rows × 4 columns

In [12]:
```
pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
```

Out[12]:

	ID	C_we	C_wk	G_we	G_wk	S_we	S_wk	T_we	T_wk
0	1000002	0.5	0.5	0.0	0.0	1.0	0.5	1.0	0.5
1	1000003	1.0	0.5	0.0	0.0	2.0	2.0	3.0	2.0
2	1000004	3.0	1.0	2.0	0.0	3.0	2.0	3.0	2.0
3	1000005	4.0	2.0	0.0	0.0	4.0	3.0	7.0	5.0
4	1000006	1.0	0.5	0.0	0.0	2.0	2.0	1.0	0.5
...	...	...	...	...	...	...	...	...	...
113354	1120111	7.0	6.0	7.0	6.0	3.0	1.0	3.0	2.0
113355	1120112	3.0	4.0	7.0	7.0	6.0	7.0	7.0	7.0
113356	1120113	2.0	0.0	4.0	2.0	0.0	0.0	4.0	3.0
113357	1120114	4.0	2.0	5.0	3.0	0.5	0.5	7.0	3.0
113358	1120115	0.0	0.0	7.0	6.0	0.0	0.0	0.0	0.0

113359 rows × 9 columns

In [13]:
```
pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")
```

Out[13]:

	ID	Optm	Usef	Relx	Intp	Engs	Dealpr	Thcklr	Goodme	Clsep	Conf	M
0	1087360	5	3	2	1	3	5	4	1	5	2	
1	1094049	2	3	3	2	3	1	2	1	5	1	
2	1094067	4	3	4	4	4	4	4	3	4	4	
3	1097484	3	3	3	3	3	3	3	3	3	3	
4	1102259	5	4	3	5	2	3	4	4	4	4	
...	...	...	...	...	...	...	...	...	...	...	...	
102575	1091072	5	5	2	1	5	3	2	1	1	2	
102576	1091077	3	2	4	4	4	4	2	2	4	2	
102577	1091104	2	1	4	4	5	4	4	4	4	4	
102578	1091112	3	2	4	3	4	3	4	4	4	3	
102579	1091115	4	3	2	4	2	4	5	2	5	3	

102580 rows × 15 columns

In [22]:
```
dataset1 =  pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
```

In [23]:
```
dataset2 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
```

In [24]:
```
dataset3 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")
```

In [25]:
```
merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')
```

In [26]:
```
print(merged_data.head())
```

```
         ID  gender  minority  deprived  C_we  C_wk  G_we  G_wk  S_we  S_wk  \
0  1087192       0         0         0   2.0   0.5   0.5   0.5   1.0   0.5
1  1087195       0         0         0   2.0   1.0   0.0   0.0   3.0   1.0
2  1087205       0         0         0   1.0   0.5   0.0   0.0   0.5   0.5
3  1087214       0         0         0   2.0   1.0   0.5   0.0   2.0   1.0
4  1087222       0         0         0   1.0   3.0   0.0   0.0   2.0   1.0

   ...  Engs  Dealpr  Thcklr  Goodme  Clsep  Conf  Mkmind  Loved  Intthg  \
0  ...     4       4       4       4      5     4       4      5       4
1  ...     3       4       5       3      5     4       4      5       4
2  ...     3       3       3       3      4     3       3      3       4
3  ...     4       4       4       4      3     5       4      5       4
4  ...     2       3       3       4      4     3       5      5       5

   Cheer
0      4
1      4
2      4
3      4
4      5

[5 rows x 26 columns]
```

In [27]:
```
print(merged_data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 98278 entries, 0 to 98277
Data columns (total 26 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   ID        98278 non-null  int64  
 1   gender    98278 non-null  int64  
 2   minority  98278 non-null  int64  
 3   deprived  98278 non-null  int64  
 4   C_we      98278 non-null  float64
 5   C_wk      98278 non-null  float64
 6   G_we      98278 non-null  float64
 7   G_wk      98278 non-null  float64
 8   S_we      98278 non-null  float64
 9   S_wk      98278 non-null  float64
 10  T_we      98278 non-null  float64
 11  T_wk      98278 non-null  float64
 12  Optm      98278 non-null  int64  
 13  Usef      98278 non-null  int64  
 14  Relx      98278 non-null  int64  
 15  Intp      98278 non-null  int64  
 16  Engs      98278 non-null  int64  
 17  Dealpr    98278 non-null  int64  
 18  Thcklr    98278 non-null  int64  
 19  Goodme    98278 non-null  int64  
 20  Clsep     98278 non-null  int64  
 21  Conf      98278 non-null  int64  
 22  Mkmind    98278 non-null  int64  
 23  Loved     98278 non-null  int64  
 24  Intthg    98278 non-null  int64  
 25  Cheer     98278 non-null  int64  
dtypes: float64(8), int64(18)
memory usage: 19.5 MB
None
```

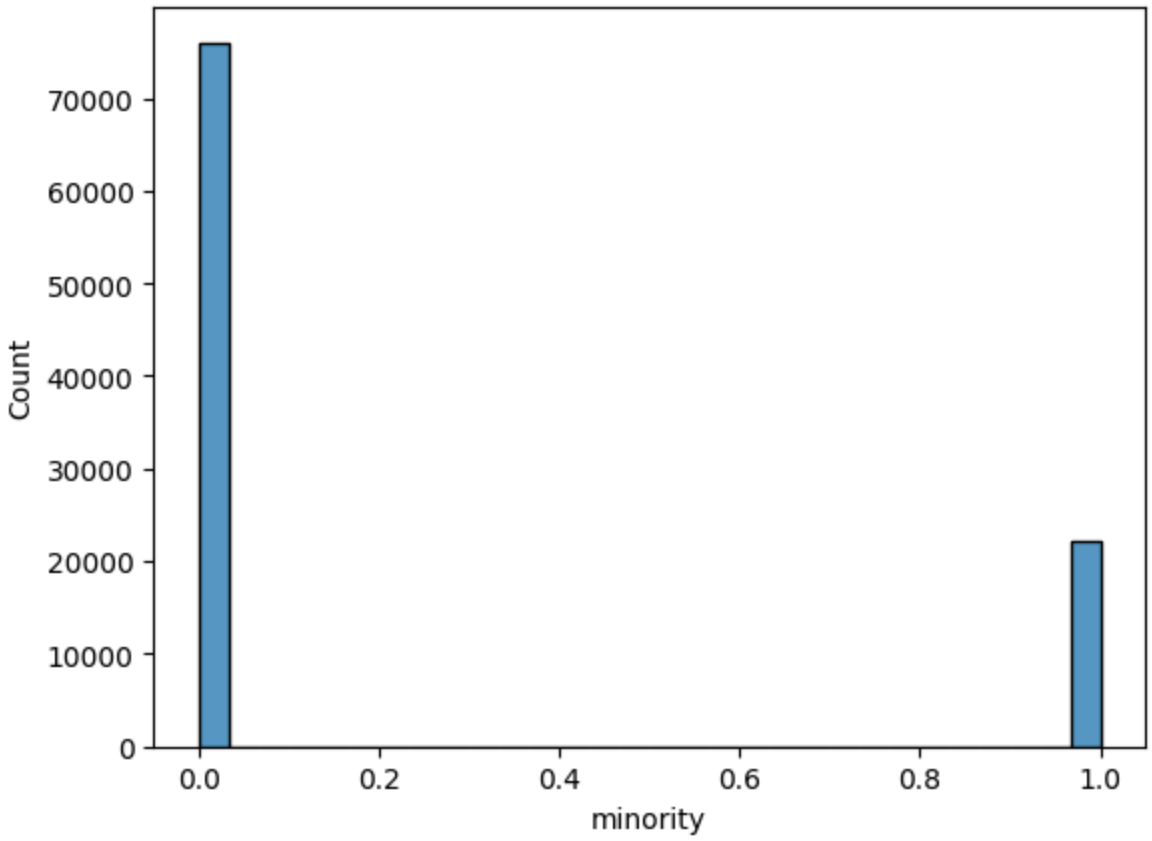
In [28]:
```
print(merged_data.describe())
```

```
                ID        gender      minority      deprived          C_we  \
count  9.827800e+04  98278.000000  98278.000000  98278.000000  98278.000000
mean   1.059895e+06      0.472059      0.226572      0.424022      2.198483
std    3.479310e+04      0.499221      0.418615      0.494196      2.069802
min    1.000002e+06      0.000000      0.000000      0.000000      0.000000
25%    1.029695e+06      0.000000      0.000000      0.000000      0.500000
50%    1.059692e+06      0.000000      0.000000      0.000000      2.000000
75%    1.090143e+06      1.000000      0.000000      1.000000      3.000000
max    1.120115e+06      1.000000      1.000000      1.000000      7.000000

               C_wk          G_we          G_wk          S_we          S_wk  \
count  98278.000000  98278.000000  98278.000000  98278.000000  98278.000000
mean       1.768092      1.726332      0.997828      3.504085      2.889604
std        1.722842      2.159675      1.540496      2.490748      2.326138
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        0.500000      0.000000      0.000000      1.000000      1.000000
50%        1.000000      0.500000      0.000000      3.000000      2.000000
75%        3.000000      3.000000      2.000000      6.000000      5.000000
max        7.000000      7.000000      7.000000      7.000000      7.000000

       ...          Engs        Dealpr        Thcklr        Goodme  \
count  ...  98278.000000  98278.000000  98278.000000  98278.000000
mean   ...      3.046155      3.370693      3.488726      3.271780
std    ...      1.075498      1.047807      1.017481      1.125303
min    ...      1.000000      1.000000      1.000000      1.000000
25%    ...      2.000000      3.000000      3.000000      3.000000
50%    ...      3.000000      3.000000      4.000000      3.000000
75%    ...      4.000000      4.000000      4.000000      4.000000
max    ...      5.000000      5.000000      5.000000      5.000000

              Clsep          Conf        Mkmind         Loved        Intthg  \
count  98278.000000  98278.000000  98278.000000  98278.000000  98278.000000
mean       3.557348      3.306732      3.851533      3.898950      3.477604
std        1.029892      1.115466      0.973831      1.069087      1.071202
min        1.000000      1.000000      1.000000      1.000000      1.000000
25%        3.000000      3.000000      3.000000      3.000000      3.000000
50%        4.000000      3.000000      4.000000      4.000000      4.000000
75%        4.000000      4.000000      5.000000      5.000000      4.000000
max        5.000000      5.000000      5.000000      5.000000      5.000000

              Cheer
count  98278.000000
mean       3.496713
std        1.011319
min        1.000000
25%        3.000000
50%        4.000000
75%        4.000000
max        5.000000

[8 rows x 26 columns]
```

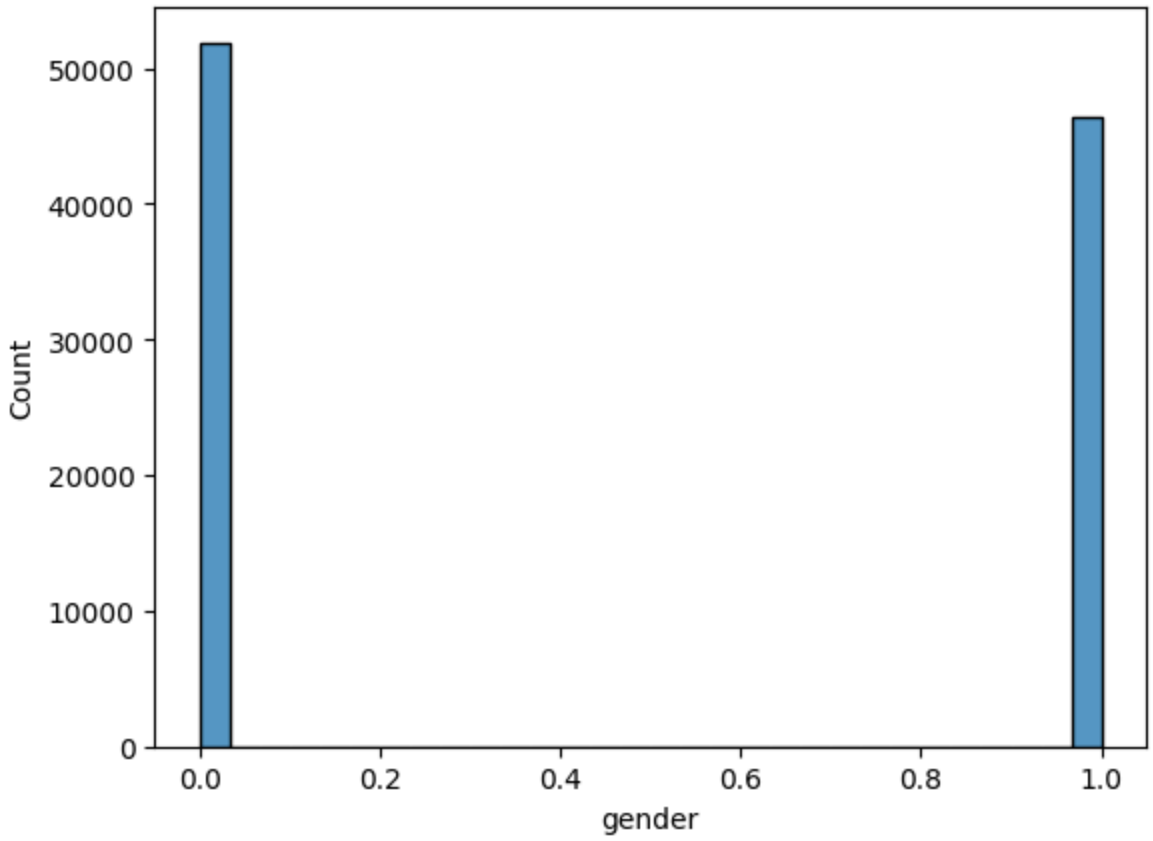
In [36]:
```
import matplotlib.pyplot as plt
```

In [41]:
```
import seaborn as sns
```

In [43]:
```
sns.histplot(merged_data['minority'], bins=30)
plt.show()
```

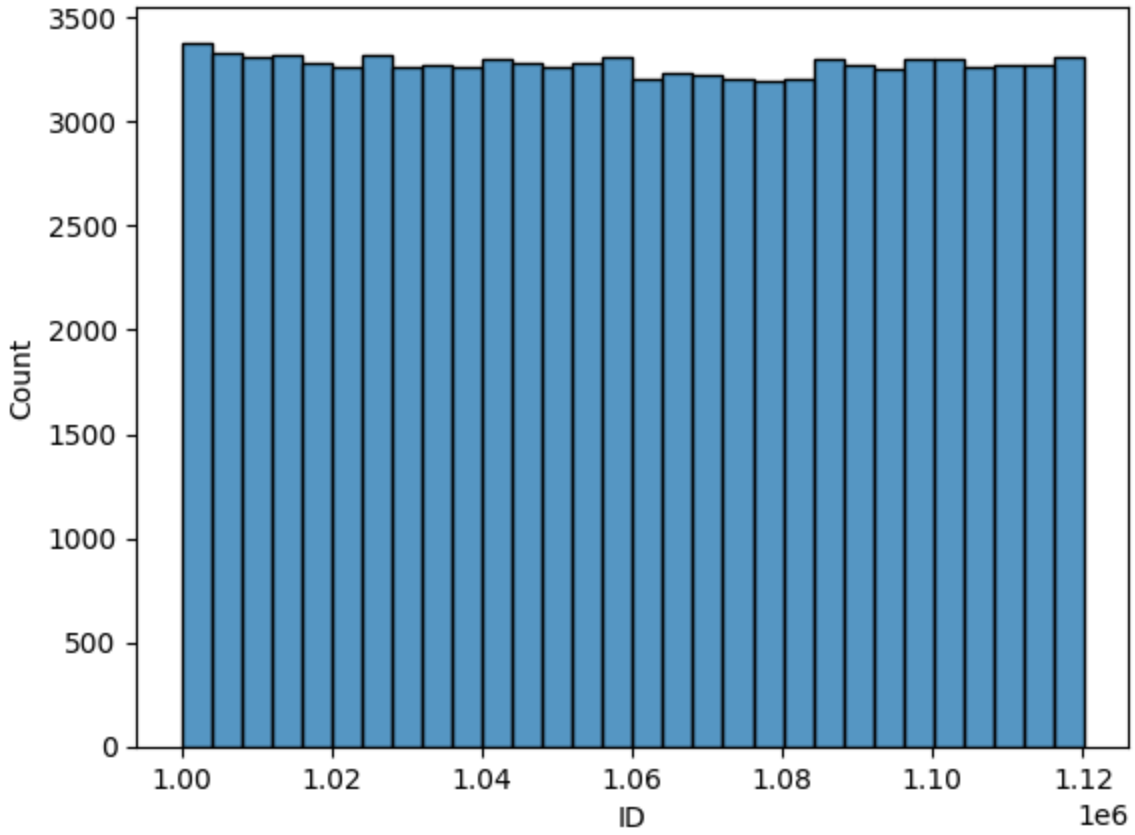


In [98]:
```
sns.histplot(merged_data['gender'], bins=30)
plt.show()
gender_counts = merged_data['gender'].value_counts()
gender_table = gender_counts.to_markdown(numalign='left', stralign='left')

print(gender_table)
```



gender	count
0	51885
1	46393

In [45]:
```
sns.histplot(merged_data['ID'], bins=30)
plt.show()
```



In [50]:
```
summary_stats = merged_data[['minority']].describe()
print(summary_stats)
```

```
           minority
count  98278.000000
mean       0.226572
std        0.418615
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        1.000000
```

In [52]:
```
summary_stats = merged_data[['ID']].describe()
print(summary_stats)
```

```
                 ID
count  9.827800e+04
mean   1.059895e+06
std    3.479310e+04
min    1.000002e+06
25%    1.029695e+06
50%    1.059692e+06
75%    1.090143e+06
max    1.120115e+06
```

In [55]:
```
summary_stats = merged_data[['minority', 'gender']].describe()  # Use lowercase 'ge
print(summary_stats)
```

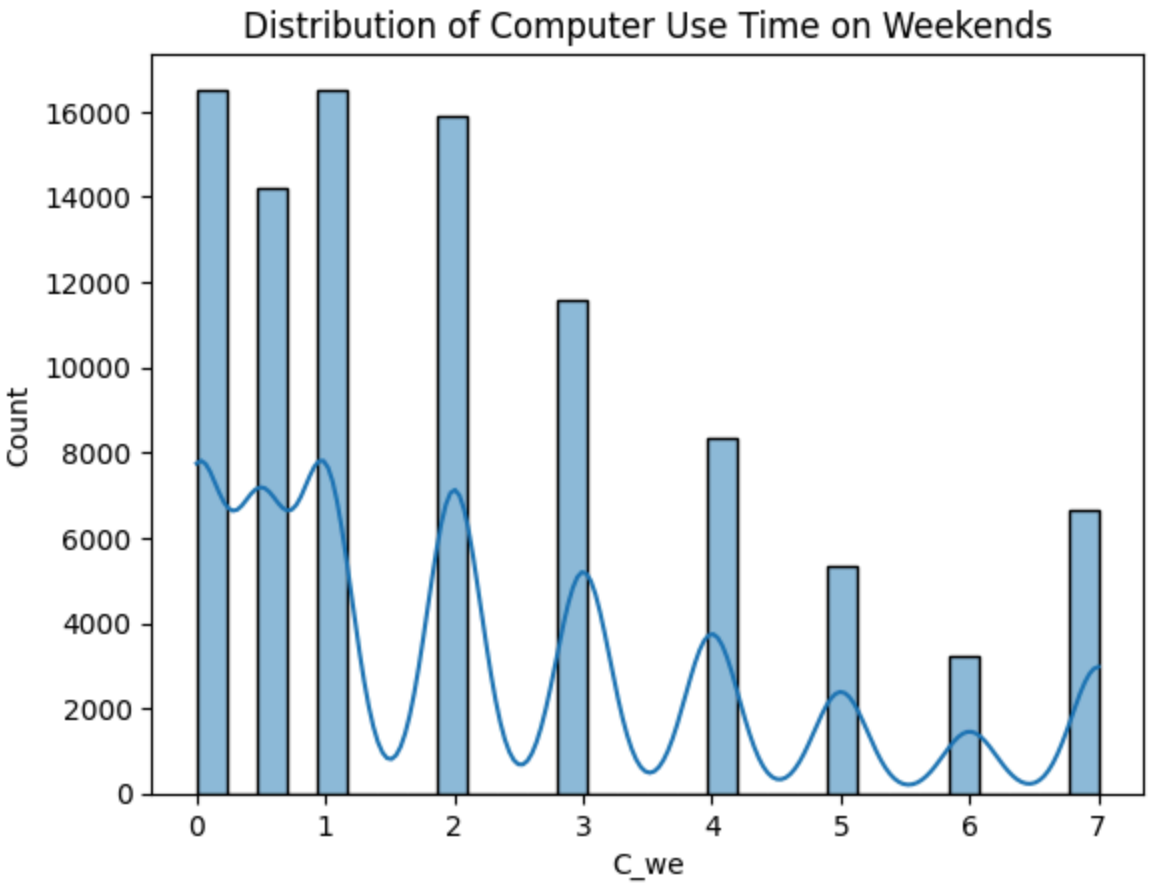
```
           minority        gender
count  98278.000000  98278.000000
mean       0.226572      0.472059
std        0.418615      0.499221
min        0.000000      0.000000
25%        0.000000      0.000000
50%        0.000000      0.000000
75%        0.000000      1.000000
max        1.000000      1.000000
```

In [59]:
```
screen_time_columns = ['C_we', 'C_wk', 'G_we', 'G_wk', 'S_we', 'S_wk', 'T_we', 'T_w
screen_time_summary = merged_data[screen_time_columns].describe()
```

In [60]:
```
import seaborn as sns
```

In [61]:
```
import matplotlib.pyplot as plt
```

In [62]:
```
sns.histplot(merged_data['C_we'], bins=30, kde=True)
plt.title('Distribution of Computer Use Time on Weekends')
plt.show()
```



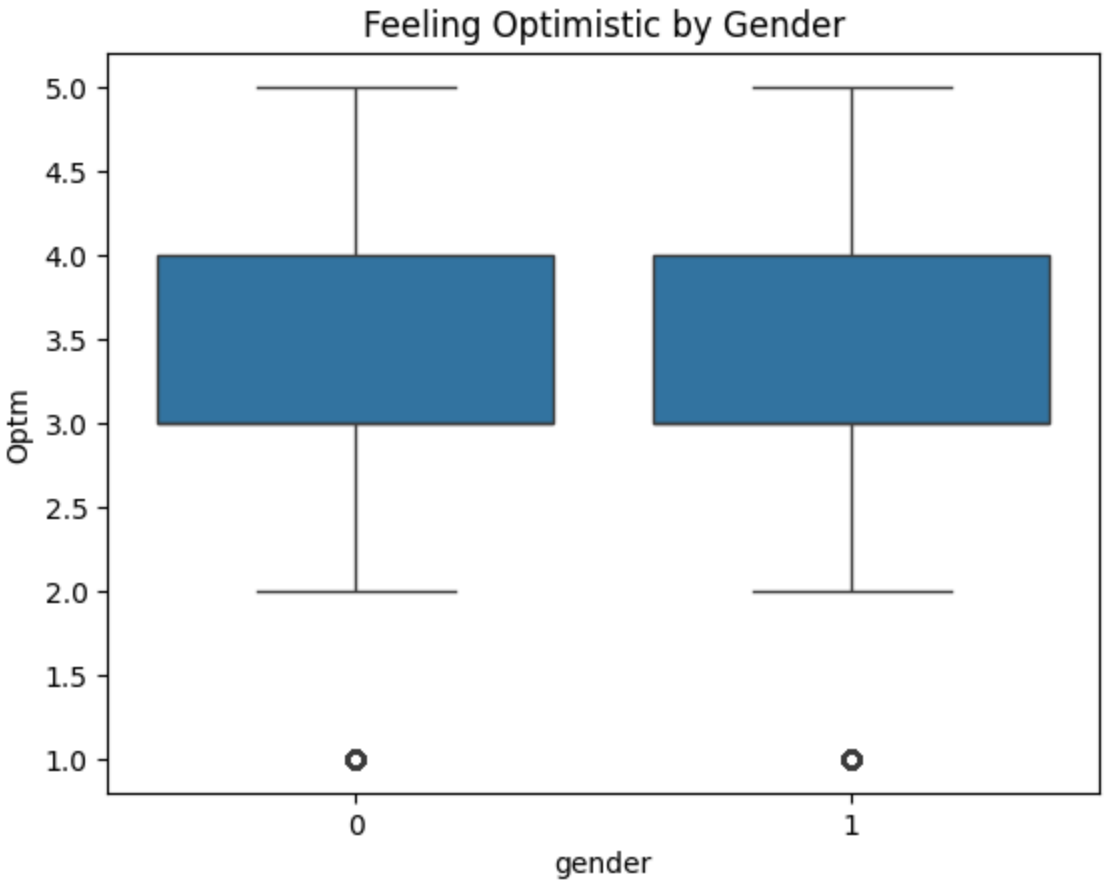
In [67]:
```
wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr', 'Thkclr', 'G
```

In [70]:
```
print(merged_data.columns)


wellbeing_columns = [col for col in merged_data.columns if col in ['Optm', 'Usef',
```

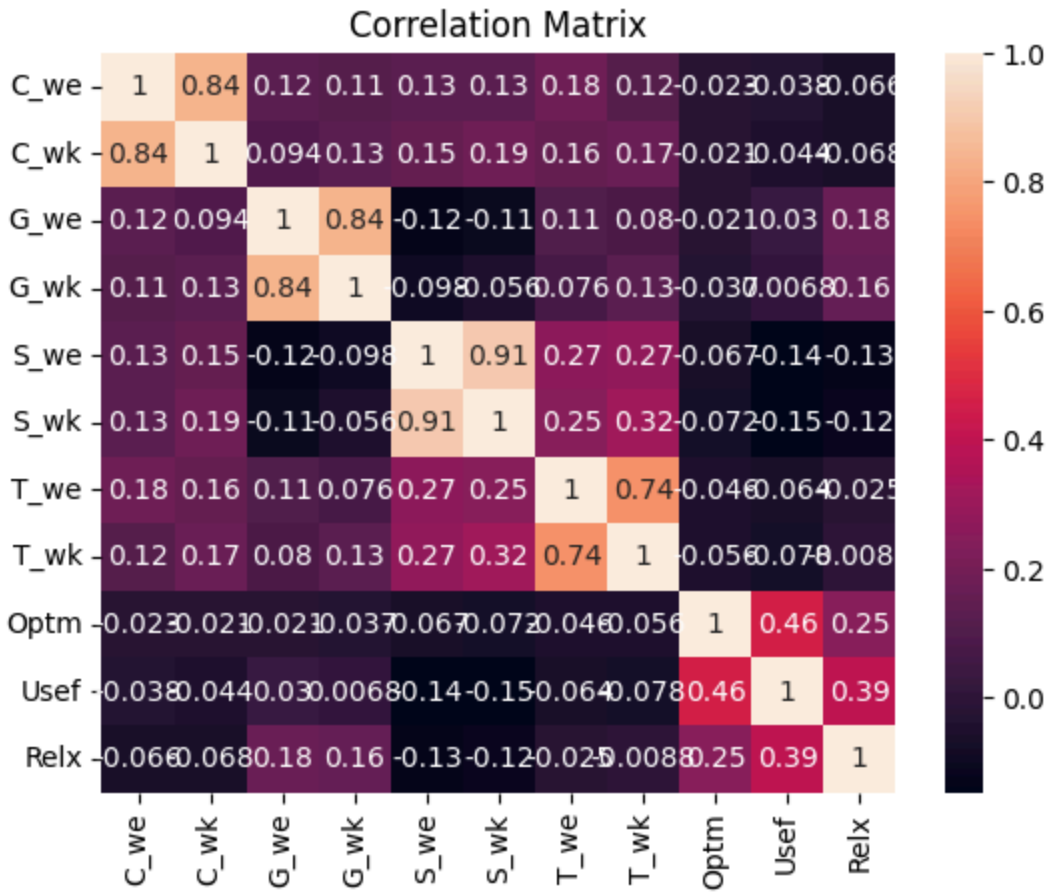
```
wellbeing_summary = merged_data[wellbeing_columns].describe()
```

```
Index(['ID', 'gender', 'minority', 'deprived', 'C_we', 'C_wk', 'G_we', 'G_wk',
       'S_we', 'S_wk', 'T_we', 'T_wk', 'Optm', 'Usef', 'Relx', 'Intp', 'Engs',
       'Dealpr', 'Thcklr', 'Goodme', 'Clsep', 'Conf', 'Mkmind', 'Loved',
       'Intthg', 'Cheer'],
      dtype='object')
```

In [71]:
```
sns.boxplot(x='gender', y='Optm', data=merged_data)
plt.title('Feeling Optimistic by Gender')
plt.show()
```



In [72]:
```
correlation_matrix = merged_data[screen_time_columns + wellbeing_columns].corr()
sns.heatmap(correlation_matrix, annot=True)
plt.title('Correlation Matrix')
plt.show()
```



In [77]:
```
from scipy.stats import ttest_ind

high_tv_use = merged_data[merged_data['T_wk'] > 2]['Optm']  # High TV use group
low_tv_use = merged_data[merged_data['T_wk'] <= 2]['Optm']  # Low TV use group
t_stat, p_value = ttest_ind(high_tv_use, low_tv_use)
print(f"T-statistic: {t_stat}, P-value: {p_value}")
```

```
T-statistic: -14.435487210913344, P-value: 3.459051603771344e-47
```

In [169…
```
import statsmodels.api as sm


X = merged_data[['C_we', 'C_wk', 'G_we', 'G_wk', 'S_we', 'S_wk', 'T_we', 'T_wk']]
y = merged_data['Optm']
X = sm.add_constant(X)

model = sm.OLS(y, X).fit()
print(model.summary())
```

OLS Regression Results

Dep. Variable:	Optm	R-squared:	0.008
Model:	OLS	Adj. R-squared:	0.008
Method:	Least Squares	F-statistic:	101.5
Date:	Thu, 12 Sep 2024	Prob (F-statistic):	3.19e-169
Time:	21:02:28	Log-Likelihood:	-1.3878e+05
No. Observations:	98278	AIC:	2.776e+05
Df Residuals:	98269	BIC:	2.777e+05
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	3.4360	0.008	429.741	0.000	3.420	3.452
C_we	-0.0121	0.003	-4.195	0.000	-0.018	-0.006
C_wk	0.0131	0.003	3.733	0.000	0.006	0.020
G_we	0.0109	0.003	3.981	0.000	0.006	0.016
G_wk	-0.0372	0.004	-9.649	0.000	-0.045	-0.030
S_we	-0.0056	0.003	-1.821	0.069	-0.012	0.000
S_wk	-0.0218	0.003	-6.499	0.000	-0.028	-0.015
T_we	-0.0044	0.003	-1.720	0.085	-0.009	0.001
T_wk	-0.0151	0.003	-5.077	0.000	-0.021	-0.009

Omnibus:	1536.317	Durbin-Watson:	1.966
Prob(Omnibus):	0.000	Jarque-Bera (JB):	1496.835
Skew:	-0.275	Prob(JB):	0.00
Kurtosis:	2.747	Cond. No.	20.5

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.

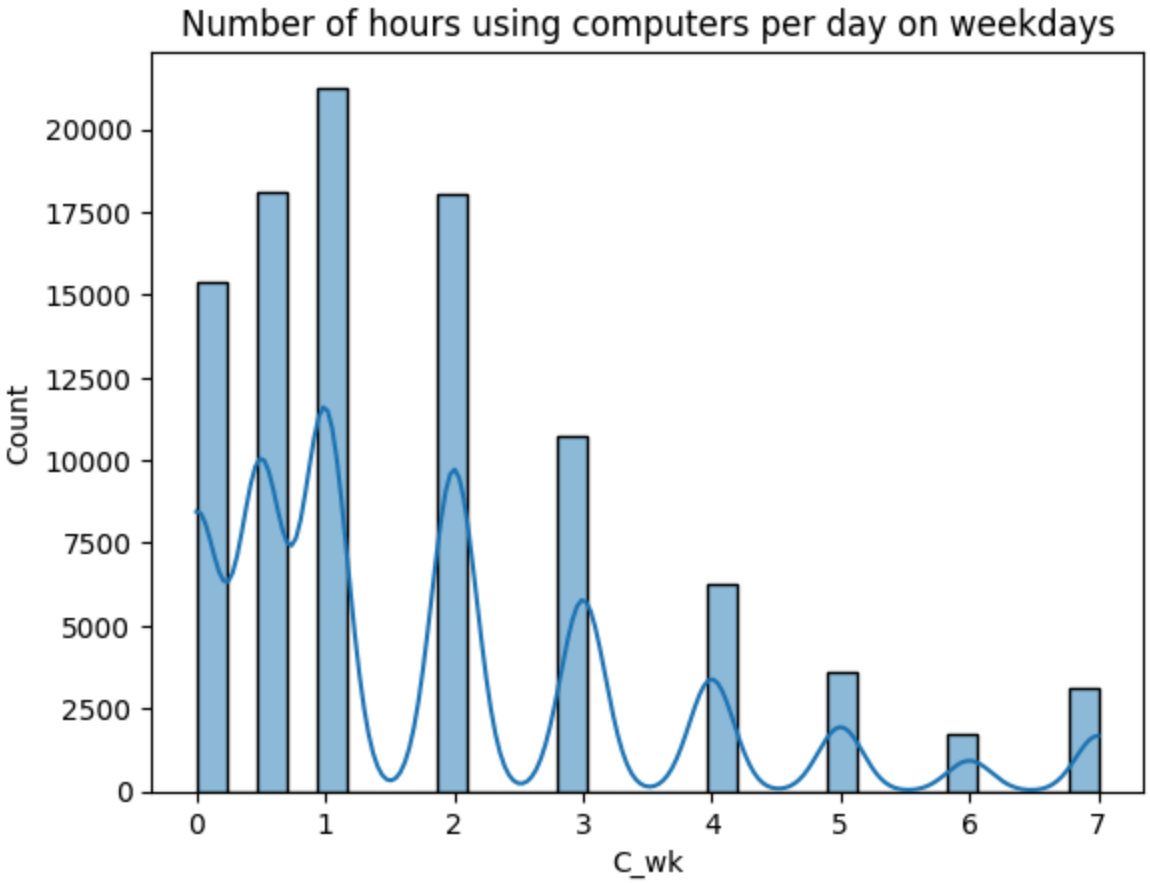
In [170…
```
merged_data['total_screen_time_weekday'] = merged_data['C_wk'] + merged_data['G_wk'
merged_data['total_screen_time_weekend'] = merged_data['C_we'] + merged_data['G_we'
```

In [86]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score


X = merged_data[['total_screen_time_weekday', 'total_screen_time_weekend']]
y = merged_data['Optm']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta


model = LinearRegression()
model.fit(X_train, y_train)


y_pred = model.predict(X_test)
```

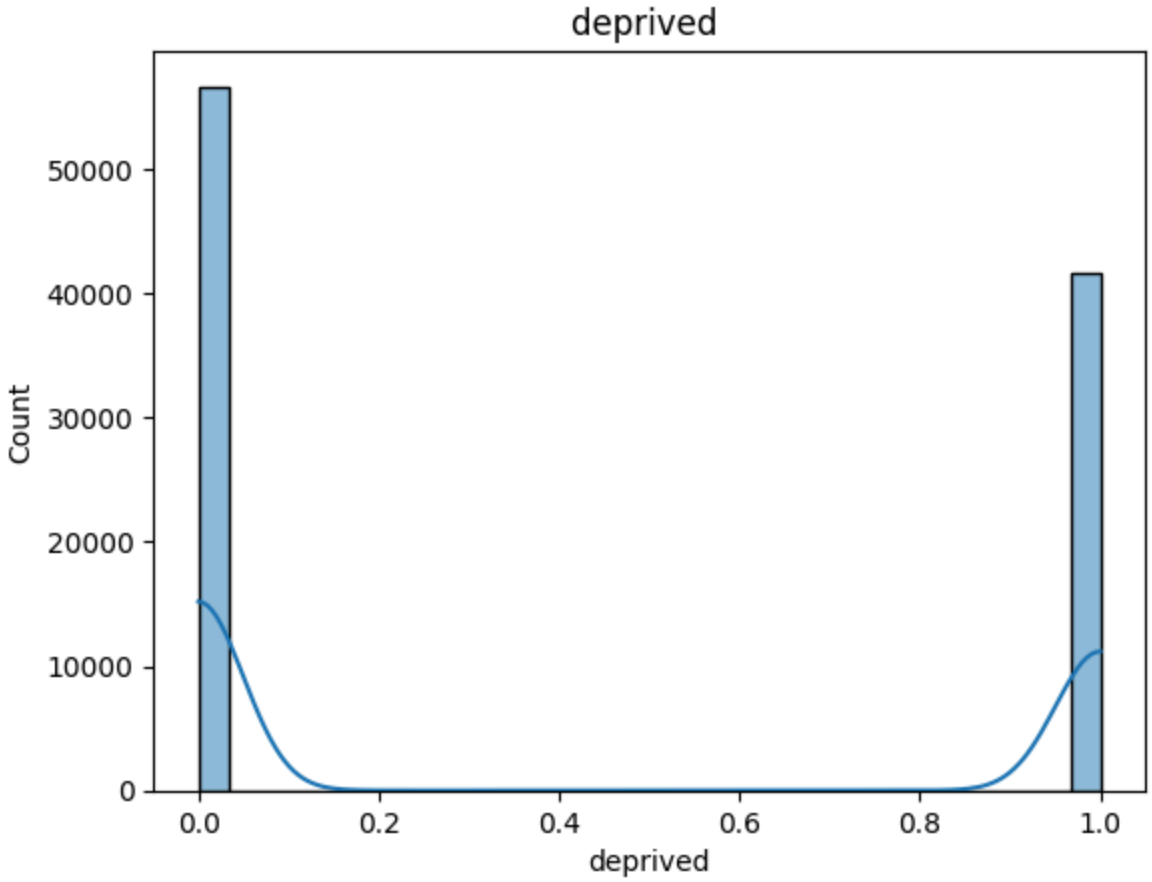
```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse}, R-squared: {r2}")
```

```
Mean Squared Error: 0.9862678657774571, R-squared: 0.004129078345420645
```

In [87]:
```
sns.histplot(merged_data['C_wk'], bins=30, kde=True)
plt.title('Number of hours using computers per day on weekdays')
plt.show()
```

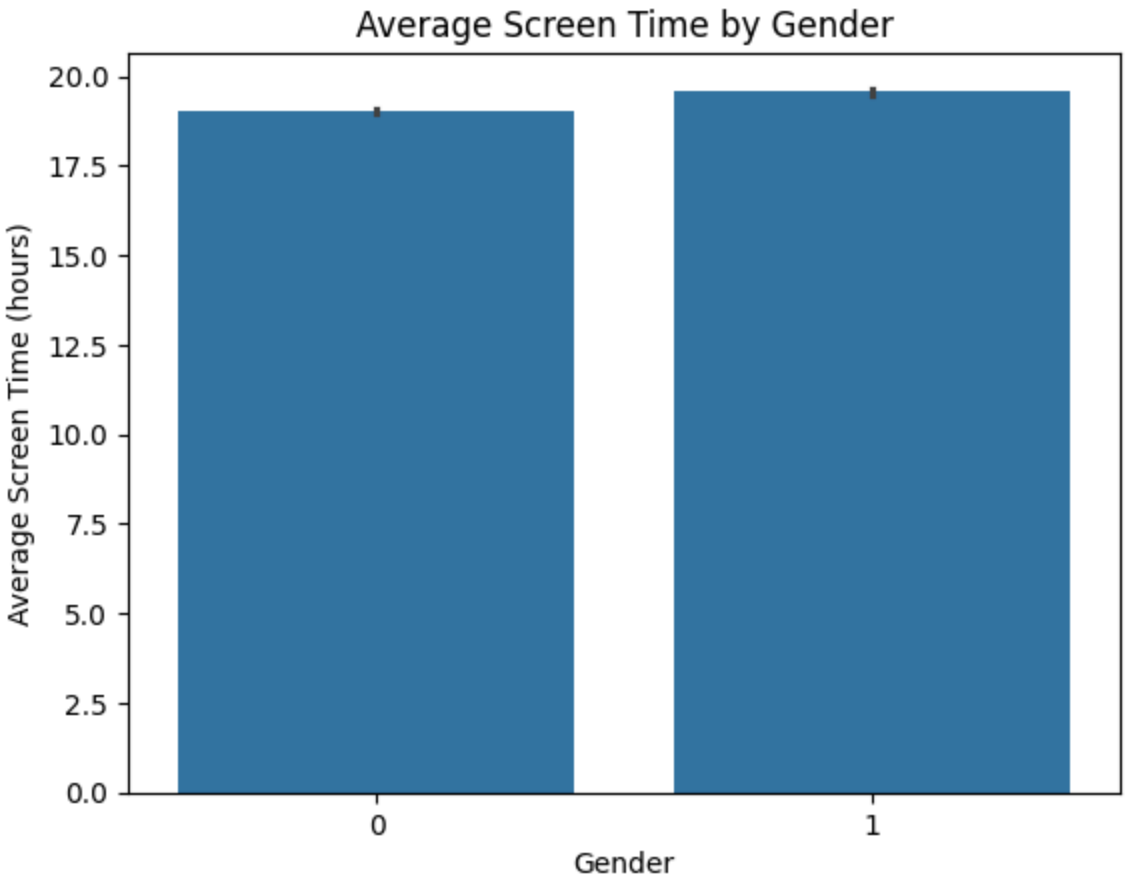


In [95]:
```
sns.histplot(merged_data['deprived' ], bins=30, kde=True)
plt.title('deprived ')
plt.show()
```



In [171…

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
merged_data['total_screen_time'] = merged_data[['C_we', 'C_wk', 'G_we', 'G_wk', 'S_
average_screen_time_by_gender = merged_data.groupby('gender')['total_screen_time'].
sns.barplot(x='gender', y='total_screen_time', data=merged_data)
plt.title('Average Screen Time by Gender')
plt.xlabel('Gender')
plt.ylabel('Average Screen Time (hours)')
plt.show()
```



In [100…
```
wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr', 'Thkclr', 'G
```

In [102…
```
wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr', 'Thkclr', 'G
existing_columns = list(set(wellbeing_columns) & set(merged_data.columns))  # Find
wellbeing_summary = merged_data[existing_columns].describe()
print(wellbeing_summary)
```

	Loved	Cheer	Optm	Intp	Conf	\
count	98278.000000	98278.000000	98278.000000	98278.000000	98278.000000	
mean	3.898950	3.496713	3.276919	3.271658	3.306732	
std	1.069087	1.011319	0.997319	1.017242	1.115466	
min	1.000000	1.000000	1.000000	1.000000	1.000000	
25%	3.000000	3.000000	3.000000	3.000000	3.000000	
50%	4.000000	4.000000	3.000000	3.000000	3.000000	
75%	5.000000	4.000000	4.000000	4.000000	4.000000	
max	5.000000	5.000000	5.000000	5.000000	5.000000	

	Dealpr	Intthg	Engs	Mkmind	Relx	\
count	98278.000000	98278.000000	98278.000000	98278.000000	98278.000000	
mean	3.370693	3.477604	3.046155	3.851533	3.096502	
std	1.047807	1.071202	1.075498	0.973831	1.014054	
min	1.000000	1.000000	1.000000	1.000000	1.000000	
25%	3.000000	3.000000	2.000000	3.000000	2.000000	
50%	3.000000	4.000000	3.000000	4.000000	3.000000	
75%	4.000000	4.000000	4.000000	5.000000	4.000000	
max	5.000000	5.000000	5.000000	5.000000	5.000000	

	Goodme	Usef	Clsep
count	98278.000000	98278.000000	98278.000000
mean	3.271780	3.107593	3.557348
std	1.125303	0.951901	1.029892
min	1.000000	1.000000	1.000000
25%	3.000000	3.000000	3.000000
50%	3.000000	3.000000	4.000000
75%	4.000000	4.000000	4.000000
max	5.000000	5.000000	5.000000

In [103…

```
import pandas as pd
wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr', 'Thkclr', 'G
existing_columns = [col for col in wellbeing_columns if col in merged_data.columns]
if existing_columns:
    wellbeing_summary = merged_data[existing_columns].describe()
    wellbeing_summary_table = wellbeing_summary.to_markdown(numalign='left', strali
    print(wellbeing_summary_table)
else:
    print("None of the specified wellbeing columns were found in the data.")
```

	Optm	Usef	Relx	Intp	Engs	Dealpr	Goodme	Clsep	Conf	Mkmind	Loved	Intthg	Cheer
count	98278	98278	98278	98278	98278	98278	98278	98278	98278	98278	98278	98278	98278
mean	3.27692	3.10759	3.0965	3.27166	3.04615	3.37069	3.27178	3.55735	3.30673	3.85153	3.89895	3.4776	3.49671
std	0.997319	0.951901	1.01405	1.01724	1.0755	1.04781	1.1253	1.02989	1.11547	0.973831	1.06909	1.0712	1.01132
min	1	1	1	1	1	1	1	1	1	1	1	1	1
25%	3	3	2	3	2	3	3	3	3	3	3	3	3
50%	3	3	3	3	3	3	3	4	3	4	4	4	4
75%	4	4	4	4	4	4	4	4	4	5	5	4	4
max	5	5	5	5	5	5	5	5	5	5	5	5	5

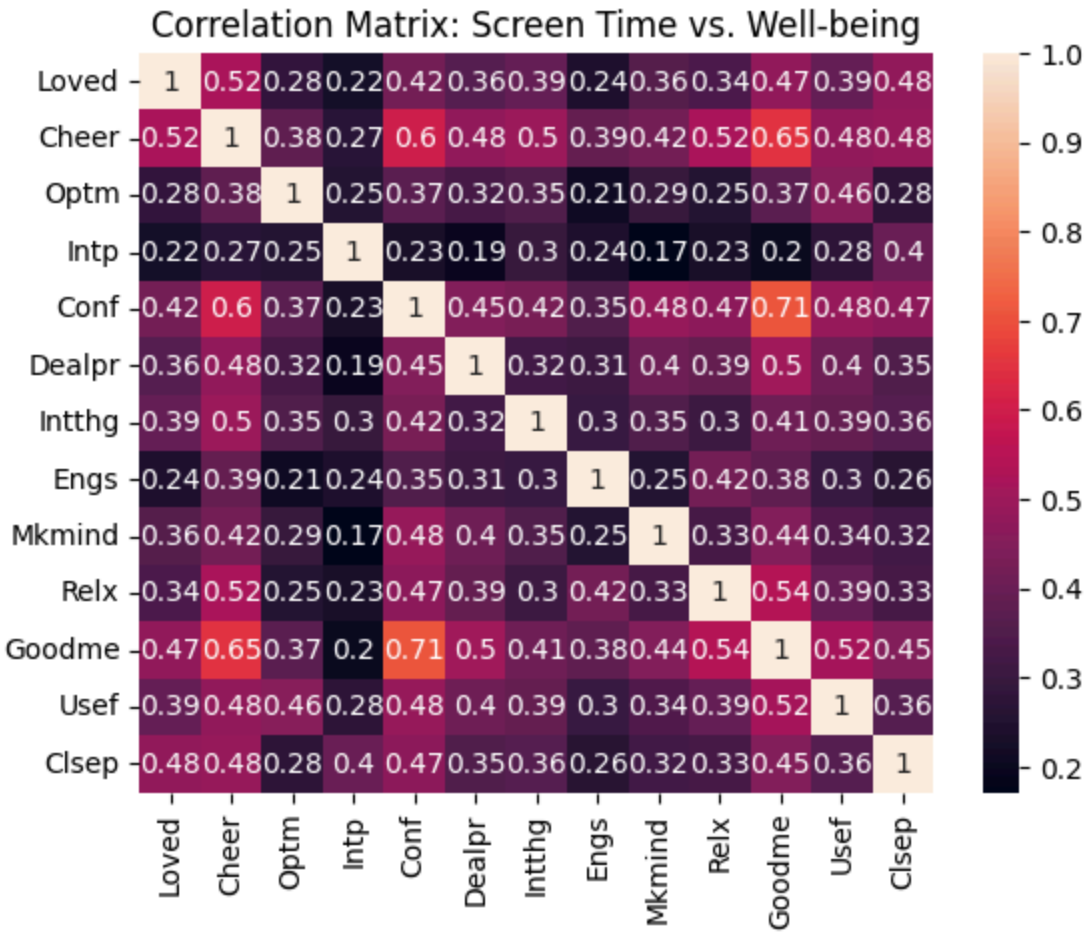
In [105…

```
average_screen_time_by_gender = merged_data.groupby('gender')['total_screen_time'].
print(average_screen_time_by_gender)
```

```
gender
0    19.026559
1    19.570873
Name: total_screen_time, dtype: float64
```

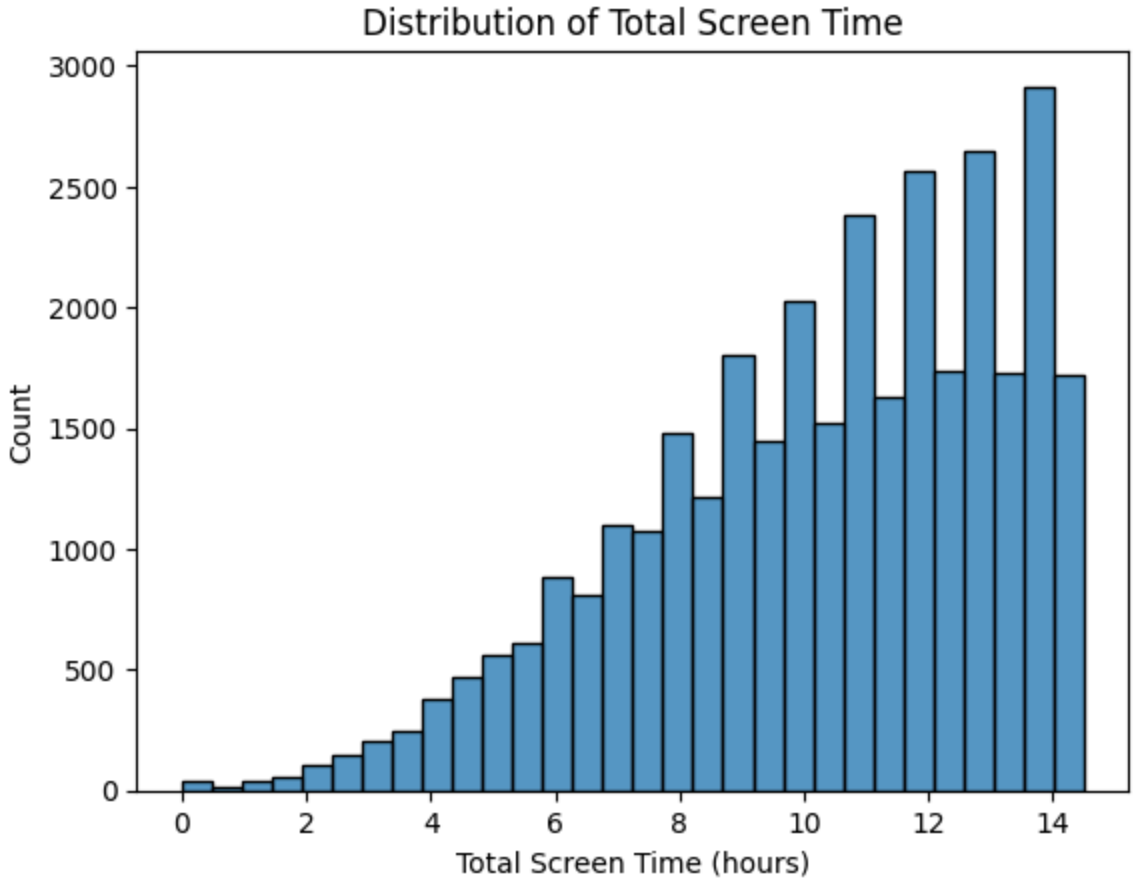
In [107…

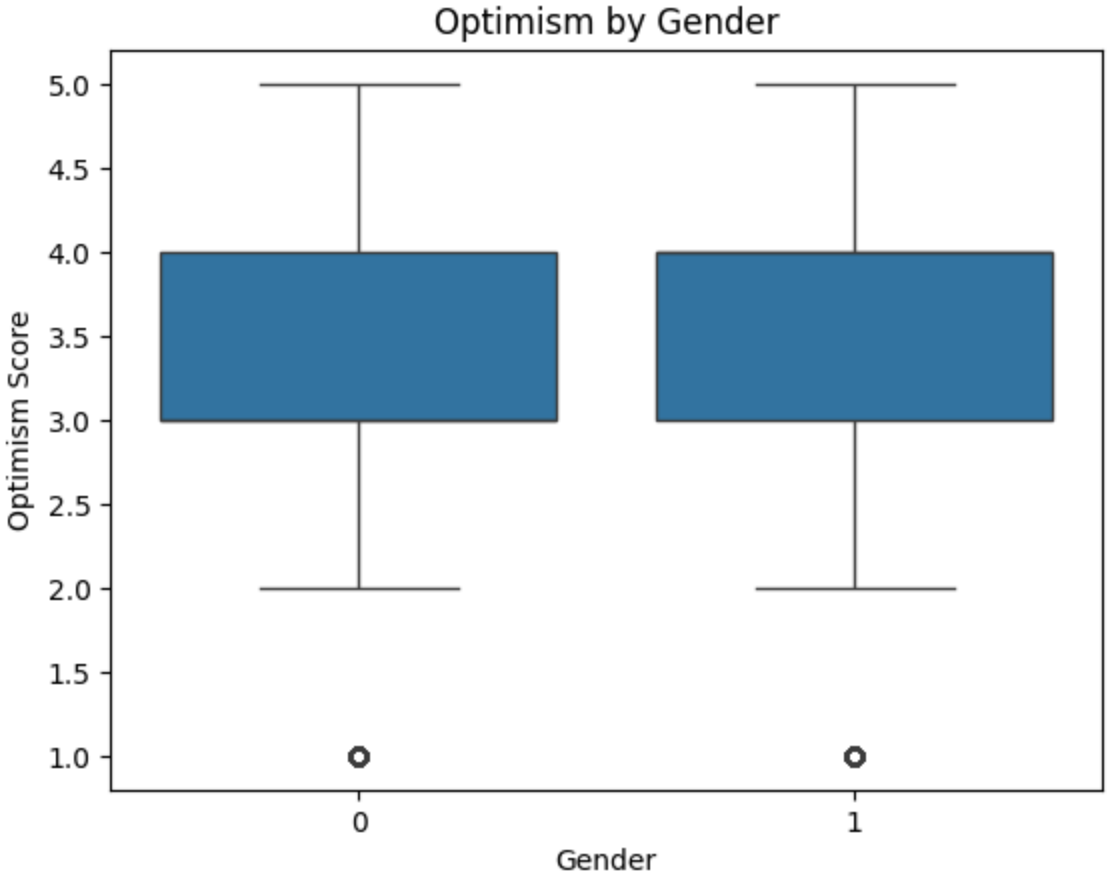
```
wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr', 'Thkclr', 'G
existing_columns = list(set(wellbeing_columns) & set(merged_data.columns))
correlation_matrix = merged_data[existing_columns].corr()
sns.heatmap(correlation_matrix, annot=True)
plt.title('Correlation Matrix: Screen Time vs. Well-being')
plt.show()
```



In [119…
```
merged_data['weekday_weekend'] = merged_data[['C_wk', 'G_wk', 'S_wk', 'T_wk']].sum(
```

In [126…
```
import seaborn as sns
sns.histplot(merged_data['total_screen_time'], bins=30)
plt.title('Distribution of Total Screen Time')
plt.xlabel('Total Screen Time (hours)')
plt.ylabel('Count')
plt.show()
sns.boxplot(x='gender', y='Optm', data=merged_data)
plt.title('Optimism by Gender')
plt.xlabel('Gender')
plt.ylabel('Optimism Score')
plt.show()
```


Distribution of Total Screen Time
Count
Total Screen Time (hours)
0
500
1000
1500
2000
2500
3000
0
2
4
6
8
10
12
14


Optimism by Gender
Optimism Score
Gender
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
0
1

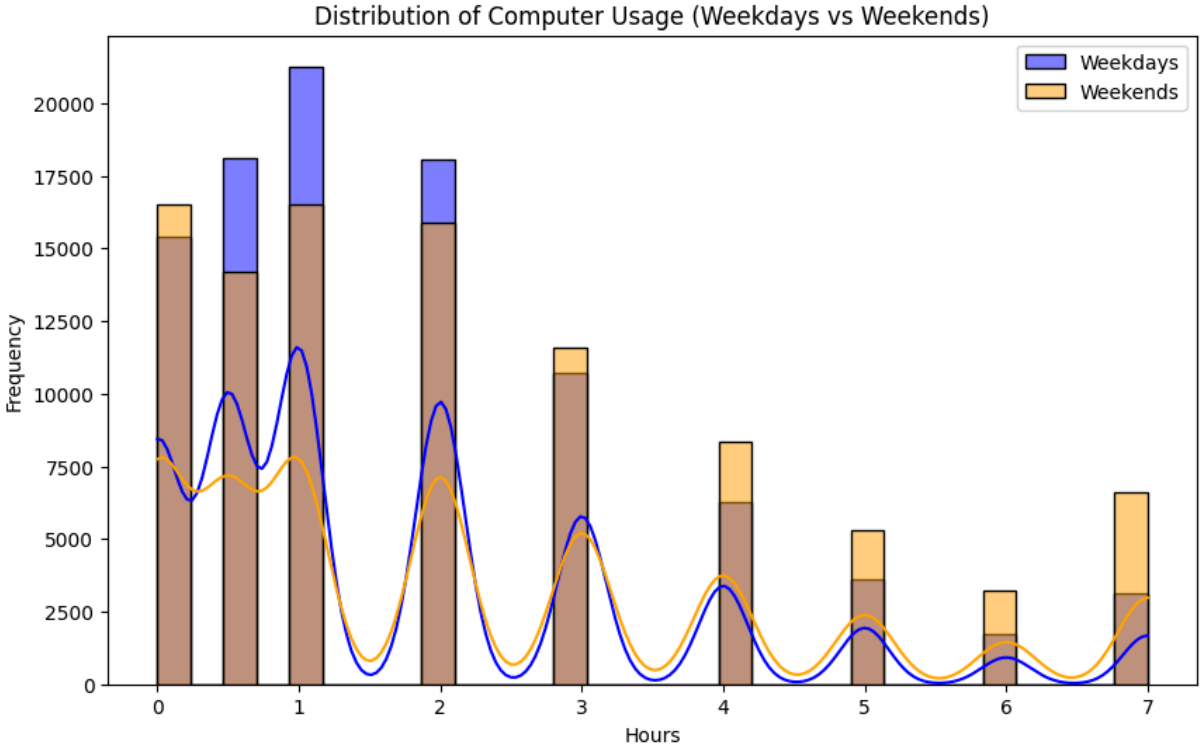
In [172…

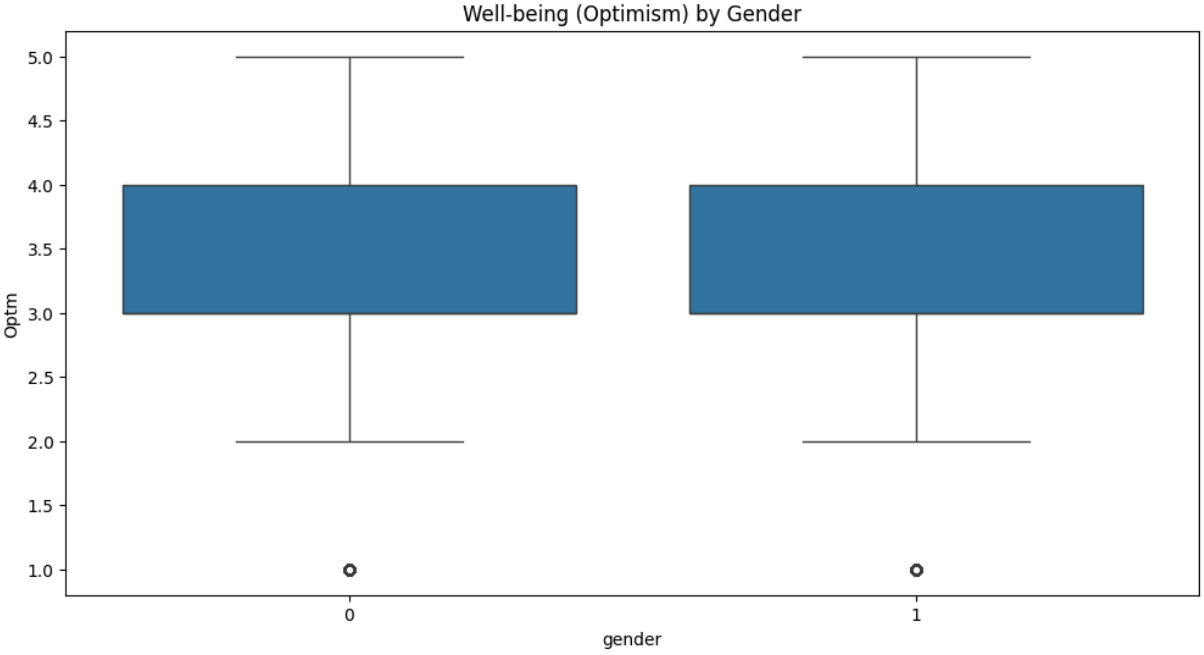
```
print(merged_data[['gender', 'minority', 'deprived']].describe())
print(merged_data[['C_we', 'C_wk', 'G_we', 'G_wk', 'S_we', 'S_wk', 'T_we', 'T_wk']]
print(merged_data.iloc[:, 15:].describe())
plt.figure(figsize=(10, 6))
sns.histplot(merged_data['C_wk'], bins=30, kde=True, color='blue', label='Weekdays'
sns.histplot(merged_data['C_we'], bins=30, kde=True, color='orange', label='Weekend
plt.xlabel('Hours')
plt.ylabel('Frequency')
plt.title('Distribution of Computer Usage (Weekdays vs Weekends)')
plt.legend()
plt.show()
plt.figure(figsize=(12, 6))
sns.boxplot(x='gender', y='Optm', data=merged_data)
plt.title('Well-being (Optimism) by Gender')
plt.show()
```

```
              gender      minority      deprived
count  98278.000000  98278.000000  98278.000000
mean       0.472059      0.226572      0.424022
std        0.499221      0.418615      0.494196
min        0.000000      0.000000      0.000000
25%        0.000000      0.000000      0.000000
50%        0.000000      0.000000      0.000000
75%        1.000000      0.000000      1.000000
max        1.000000      1.000000      1.000000
               C_we          C_wk          G_we          G_wk          S_we  \
count  98278.000000  98278.000000  98278.000000  98278.000000  98278.000000
mean       2.198483      1.768092      1.726332      0.997828      3.504085
std        2.069802      1.722842      2.159675      1.540496      2.490748
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        0.500000      0.500000      0.000000      0.000000      1.000000
50%        2.000000      1.000000      0.500000      0.000000      3.000000
75%        3.000000      3.000000      3.000000      2.000000      6.000000
max        7.000000      7.000000      7.000000      7.000000      7.000000

               S_wk          T_we          T_wk
count  98278.000000  98278.000000  98278.000000
mean       2.889604      3.647439      2.551644
std        2.326138      1.944612      1.686233
min        0.000000      0.000000      0.000000
25%        1.000000      2.000000      1.000000
50%        2.000000      4.000000      2.000000
75%        5.000000      5.000000      4.000000
max        7.000000      7.000000      7.000000
               Intp          Engs        Dealpr        Thcklr        Goodme  \
count  98278.000000  98278.000000  98278.000000  98278.000000  98278.000000
mean       3.271658      3.046155      3.370693      3.488726      3.271780
std        1.017242      1.075498      1.047807      1.017481      1.125303
min        1.000000      1.000000      1.000000      1.000000      1.000000
25%        3.000000      2.000000      3.000000      3.000000      3.000000
50%        3.000000      3.000000      3.000000      4.000000      3.000000
75%        4.000000      4.000000      4.000000      4.000000      4.000000
max        5.000000      5.000000      5.000000      5.000000      5.000000

              Clsep          Conf        Mkmind         Loved        Intthg  \
count  98278.000000  98278.000000  98278.000000  98278.000000  98278.000000
mean       3.557348      3.306732      3.851533      3.898950      3.477604
std        1.029892      1.115466      0.973831      1.069087      1.071202
min        1.000000      1.000000      1.000000      1.000000      1.000000
25%        3.000000      3.000000      3.000000      3.000000      3.000000
50%        4.000000      3.000000      4.000000      4.000000      4.000000
75%        4.000000      4.000000      5.000000      5.000000      4.000000
max        5.000000      5.000000      5.000000      5.000000      5.000000

              Cheer  total_screen_time_weekday  total_screen_time_weekend  \
count  98278.000000               98278.000000               98278.000000
mean       3.496713                   8.207167                  11.076340
std        1.011319                   4.416450                   4.997821
min        1.000000                   0.000000                   0.000000
25%        3.000000                   5.000000                   7.500000
50%        4.000000                   7.500000                  10.500000
75%        4.000000                  11.000000                  14.000000
```

```
max        5.000000                  28.000000                  28.000000
```

```
       total_screen_time
count       98278.000000
mean           19.283507
std             8.995372
min             0.000000
25%            13.000000
50%            18.000000
75%            24.500000
max            56.000000
```





In [129…

```
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import statsmodels.api as sm
import statsmodels.formula.api as smf

dataset1 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
dataset2 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
dataset3 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")

merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')

print(merged_data.head())
```

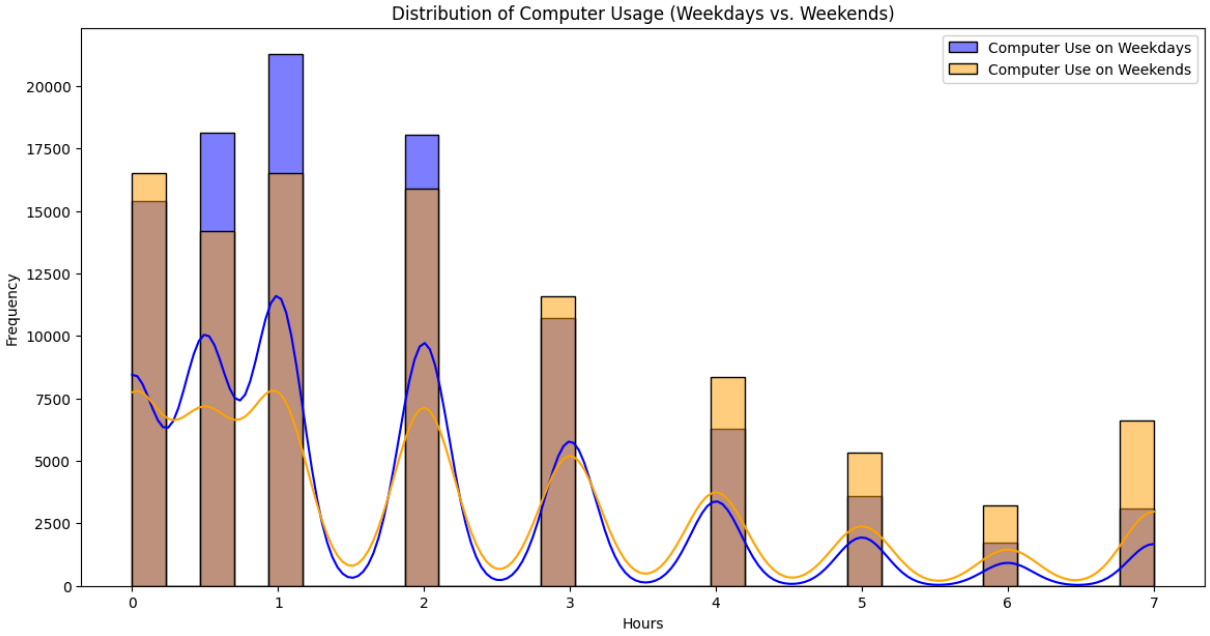
```
        ID  gender  minority  deprived  C_we  C_wk  G_we  G_wk  S_we  S_wk  \
0  1087192       0         0         0   2.0   0.5   0.5   0.5   1.0   0.5
1  1087195       0         0         0   2.0   1.0   0.0   0.0   3.0   1.0
2  1087205       0         0         0   1.0   0.5   0.0   0.0   0.5   0.5
3  1087214       0         0         0   2.0   1.0   0.5   0.0   2.0   1.0
4  1087222       0         0         0   1.0   3.0   0.0   0.0   2.0   1.0

   ...  Engs  Dealpr  Thcklr  Goodme  Clsep  Conf  Mkmind  Loved  Intthg  \
0  ...     4       4       4       4      5     4       4      5       4
1  ...     3       4       5       3      5     4       4      5       4
2  ...     3       3       3       3      4     3       3      3       4
3  ...     4       4       4       4      3     5       4      5       4
4  ...     2       3       3       4      4     3       5      5       5

   Cheer
0      4
1      4
2      4
3      4
4      5

[5 rows x 26 columns]
```

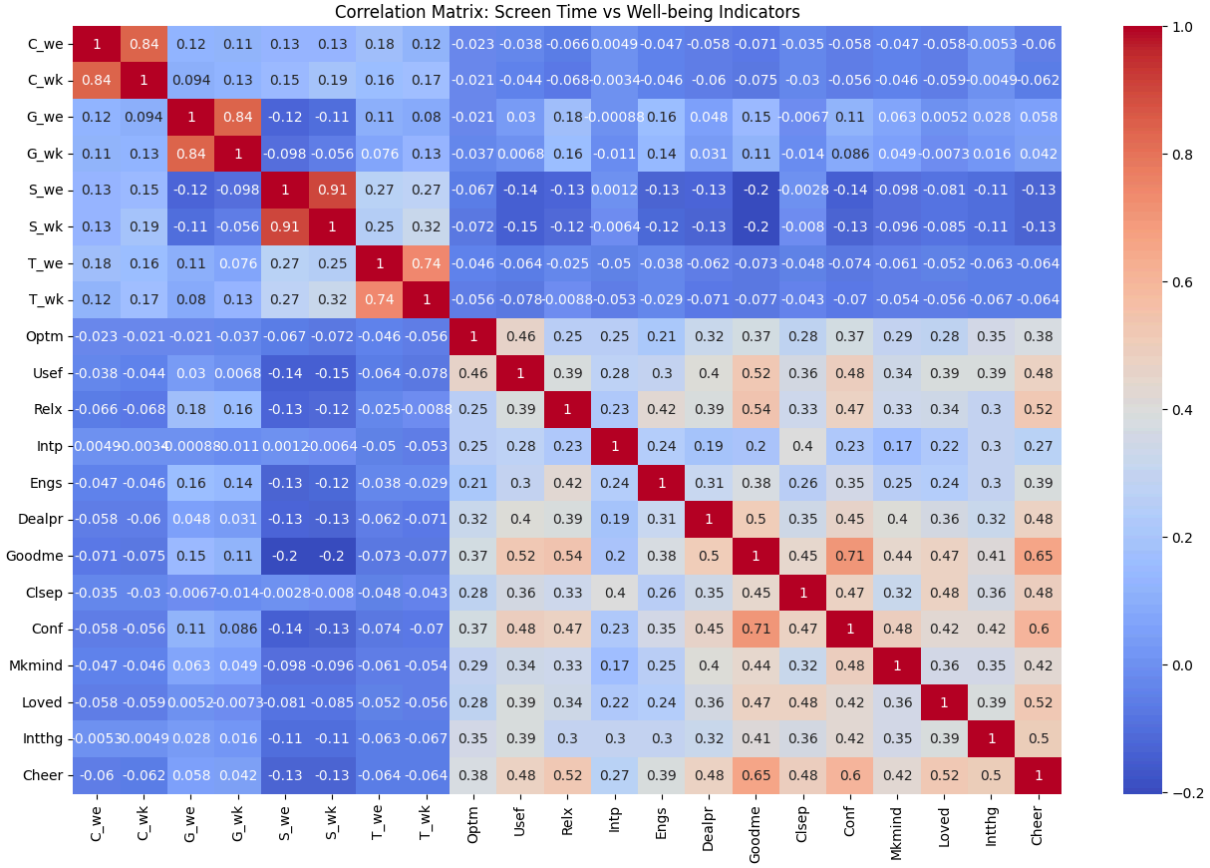
In [147…

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import statsmodels.api as sm
import statsmodels.formula.api as smf


dataset1 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
dataset2 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
dataset3 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")

merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')
```

```
print("Available columns in merged_data:")
print(merged_data.columns.tolist())

wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr',  'Goodme', '
screen_time_columns = ['C_we', 'C_wk', 'G_we', 'G_wk', 'S_we', 'S_wk', 'T_we', 'T_w


print("\nScreen Time Statistics:")
print(merged_data[screen_time_columns].describe())

print("\nWell-being Indicators Statistics by Gender:")
print(merged_data.groupby('gender')[wellbeing_columns].describe())


plt.figure(figsize=(14, 7))
sns.histplot(merged_data['C_wk'], bins=30, kde=True, color='blue', label='Computer
sns.histplot(merged_data['C_we'], bins=30, kde=True, color='orange', label='Compute
plt.xlabel('Hours')
plt.ylabel('Frequency')
plt.title('Distribution of Computer Usage (Weekdays vs. Weekends)')
plt.legend()
plt.show()


correlation_matrix = merged_data[screen_time_columns + wellbeing_columns].corr()
plt.figure(figsize=(16, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Matrix: Screen Time vs Well-being Indicators')
plt.show()


high_tv_use = merged_data[merged_data['T_wk'] > merged_data['T_wk'].median()]['Optm
low_tv_use = merged_data[merged_data['T_wk'] <= merged_data['T_wk'].median()]['Optm
t_stat, p_value = stats.ttest_ind(high_tv_use, low_tv_use)
print(f"T-test for 'Optm' based on TV usage: T-statistic = {t_stat}, P-value = {p_v


X = merged_data[screen_time_columns]
y = merged_data['Optm']
X = sm.add_constant(X)
model = sm.OLS(y, X).fit()
print(model.summary())
```

```
Available columns in merged_data:
['ID', 'gender', 'minority', 'deprived', 'C_we', 'C_wk', 'G_we', 'G_wk', 'S_we', 'S_
wk', 'T_we', 'T_wk', 'Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr', 'Thcklr', 'Go
odme', 'Clsep', 'Conf', 'Mkmind', 'Loved', 'Intthg', 'Cheer']

Screen Time Statistics:
              C_we          C_wk          G_we          G_wk          S_we  \
count  98278.000000  98278.000000  98278.000000  98278.000000  98278.000000
mean       2.198483      1.768092      1.726332      0.997828      3.504085
std        2.069802      1.722842      2.159675      1.540496      2.490748
min        0.000000      0.000000      0.000000      0.000000      0.000000
25%        0.500000      0.500000      0.000000      0.000000      1.000000
50%        2.000000      1.000000      0.500000      0.000000      3.000000
75%        3.000000      3.000000      3.000000      2.000000      6.000000
max        7.000000      7.000000      7.000000      7.000000      7.000000

              S_wk          T_we          T_wk
count  98278.000000  98278.000000  98278.000000
mean       2.889604      3.647439      2.551644
std        2.326138      1.944612      1.686233
min        0.000000      0.000000      0.000000
25%        1.000000      2.000000      1.000000
50%        2.000000      4.000000      2.000000
75%        5.000000      5.000000      4.000000
max        7.000000      7.000000      7.000000

Well-being Indicators Statistics by Gender:
          Optm                                                  Usef  \
         count      mean       std  min  25%  50%  75%  max    count
gender
0      51885.0  3.200848  1.005478  1.0  3.0  3.0  4.0  5.0  51885.0
1      46393.0  3.361994  0.981163  1.0  3.0  3.0  4.0  5.0  46393.0

                ... Intthg         Cheer                                    \
            mean  ...    75%  max    count      mean       std  min  25%  50%
gender            ...
0       2.956114  ...    4.0  5.0  51885.0  3.329671  1.034856  1.0  3.0  3.0
1       3.277003  ...    4.0  5.0  46393.0  3.683530  0.950168  1.0  3.0  4.0


        75%  max
gender
0       4.0  5.0
1       4.0  5.0

[2 rows x 104 columns]
```


Distribution of Computer Usage (Weekdays vs. Weekends)
Computer Use on Weekdays
Computer Use on Weekends
Frequency
Hours
0
2500
5000
7500
10000
12500
15000
17500
20000
0
1
2
3
4
5
6
7


Correlation Matrix: Screen Time vs Well-being Indicators
C_we 1 0.84 0.12 0.11 0.13 0.13 0.18 0.12 -0.023 -0.038 -0.066 0.0049 -0.047 -0.058 -0.071 -0.035 -0.058 -0.047 -0.058 -0.0053 -0.06
C_wk 0.84 1 0.094 0.13 0.15 0.19 0.16 0.17 -0.021 -0.044 -0.068 -0.0034 -0.046 -0.06 -0.075 -0.03 -0.056 -0.046 -0.059 -0.0049 -0.062
G_we 0.12 0.094 1 0.84 -0.12 -0.11 0.11 0.08 -0.021 0.03 0.18 -0.00088 0.16 0.048 0.15 -0.0067 0.11 0.063 0.0052 0.028 0.058
G_wk 0.11 0.13 0.84 1 -0.098 -0.056 0.076 0.13 -0.037 0.0068 0.16 -0.011 0.14 0.031 0.11 -0.014 0.086 0.049 -0.0073 0.016 0.042
S_we 0.13 0.15 -0.12 -0.098 1 0.91 0.27 0.27 -0.067 -0.14 -0.13 0.0012 -0.13 -0.13 -0.2 -0.0028 -0.14 -0.098 -0.081 -0.11 -0.13
S_wk 0.13 0.19 -0.11 -0.056 0.91 1 0.25 0.32 -0.072 -0.15 -0.12 -0.0064 -0.12 -0.13 -0.2 -0.008 -0.13 -0.096 -0.085 -0.11 -0.13
T_we 0.18 0.16 0.11 0.076 0.27 0.25 1 0.74 -0.046 -0.064 -0.025 -0.05 -0.038 -0.062 -0.073 -0.048 -0.074 -0.061 -0.052 -0.063 -0.064
T_wk 0.12 0.17 0.08 0.13 0.27 0.32 0.74 1 -0.056 -0.078 -0.0088 -0.053 -0.029 -0.071 -0.077 -0.043 -0.07 -0.054 -0.056 -0.067 -0.064
Optm -0.023 -0.021 -0.021 -0.037 -0.067 -0.072 -0.046 -0.056 1 0.46 0.25 0.25 0.21 0.32 0.37 0.28 0.37 0.29 0.28 0.35 0.38
Usef -0.038 -0.044 0.03 0.0068 -0.14 -0.15 -0.064 -0.078 0.46 1 0.39 0.28 0.3 0.4 0.52 0.36 0.48 0.34 0.39 0.39 0.48
Relx -0.066 -0.068 0.18 0.16 -0.13 -0.12 -0.025 -0.0088 0.25 0.39 1 0.23 0.42 0.39 0.54 0.33 0.47 0.33 0.34 0.3 0.52
Intp -0.0049 -0.0034 -0.00088 -0.011 0.0012 -0.0064 -0.05 -0.053 0.25 0.28 0.23 1 0.24 0.19 0.2 0.4 0.23 0.17 0.22 0.3 0.27
Engs -0.047 -0.046 0.16 0.14 -0.13 -0.12 -0.038 -0.029 0.21 0.3 0.42 0.24 1 0.31 0.38 0.26 0.35 0.25 0.24 0.3 0.39
Dealpr -0.058 -0.06 0.048 0.031 -0.13 -0.13 -0.062 -0.071 0.32 0.4 0.39 0.19 0.31 1 0.5 0.35 0.45 0.4 0.36 0.32 0.48
Goodme -0.071 -0.075 0.15 0.11 -0.2 -0.2 -0.073 -0.077 0.37 0.52 0.54 0.2 0.38 0.5 1 0.45 0.71 0.44 0.47 0.41 0.65
Clsep -0.035 -0.03 -0.0067 -0.014 -0.0028 -0.008 -0.048 -0.043 0.28 0.36 0.33 0.4 0.26 0.35 0.45 1 0.47 0.32 0.48 0.36 0.48
Conf -0.058 -0.056 0.11 0.086 -0.14 -0.13 -0.074 -0.07 0.37 0.48 0.47 0.23 0.35 0.45 0.71 0.47 1 0.48 0.42 0.42 0.6
Mkmind -0.047 -0.046 0.063 0.049 -0.098 -0.096 -0.061 -0.054 0.29 0.34 0.33 0.17 0.25 0.4 0.44 0.32 0.48 1 0.36 0.35 0.42
Loved -0.058 -0.059 0.0052 -0.0073 -0.081 -0.085 -0.052 -0.056 0.28 0.39 0.34 0.22 0.24 0.36 0.47 0.48 0.42 0.36 1 0.39 0.52
Intthg -0.0053 -0.0049 0.028 0.016 -0.11 -0.11 -0.063 -0.067 0.35 0.39 0.3 0.3 0.3 0.32 0.41 0.36 0.42 0.35 0.39 1 0.5
Cheer -0.06 -0.062 0.058 0.042 -0.13 -0.13 -0.064 -0.064 0.38 0.48 0.52 0.27 0.39 0.48 0.65 0.48 0.6 0.42 0.52 0.5 1
C_we C_wk G_we G_wk S_we S_wk T_we T_wk Optm Usef Relx Intp Engs Dealpr Goodme Clsep Conf Mkmind Loved Intthg Cheer
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

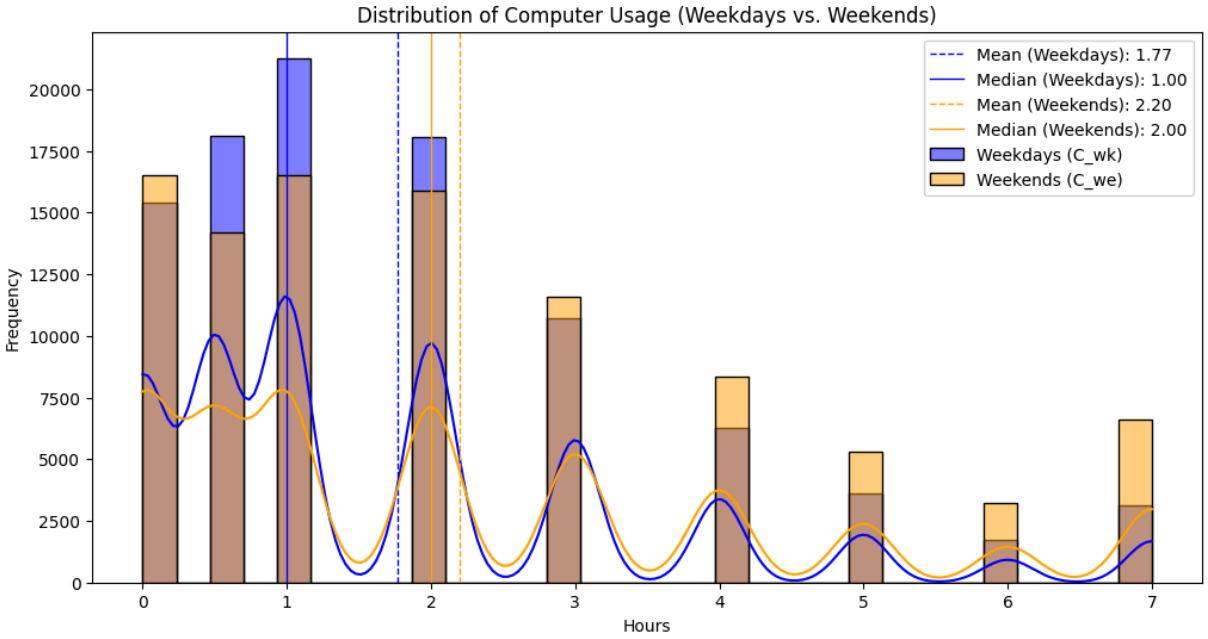
```
T-test for 'Optm' based on TV usage: T-statistic = -14.435487210913344, P-value = 3.
459051603771344e-47
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Optm   R-squared:                       0.008
Model:                            OLS   Adj. R-squared:                  0.008
Method:                 Least Squares   F-statistic:                     101.5
Date:                Thu, 12 Sep 2024   Prob (F-statistic):          3.19e-169
Time:                        05:29:06   Log-Likelihood:            -1.3878e+05
No. Observations:               98278   AIC:                         2.776e+05
Df Residuals:                   98269   BIC:                         2.777e+05
Df Model:                           8
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          3.4360      0.008    429.741      0.000       3.420       3.452
C_we          -0.0121      0.003     -4.195      0.000      -0.018      -0.006
C_wk           0.0131      0.003      3.733      0.000       0.006       0.020
G_we           0.0109      0.003      3.981      0.000       0.006       0.016
G_wk          -0.0372      0.004     -9.649      0.000      -0.045      -0.030
S_we          -0.0056      0.003     -1.821      0.069      -0.012       0.000
S_wk          -0.0218      0.003     -6.499      0.000      -0.028      -0.015
T_we          -0.0044      0.003     -1.720      0.085      -0.009       0.001
T_wk          -0.0151      0.003     -5.077      0.000      -0.021      -0.009
==============================================================================
Omnibus:                     1536.317   Durbin-Watson:                   1.966
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1496.835
Skew:                          -0.275   Prob(JB):                         0.00
Kurtosis:                       2.747   Cond. No.                         20.5
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
```

In [148…

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


dataset1 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
dataset2 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
dataset3 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")


merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')


computer_weekdays = merged_data['C_wk']
computer_weekends = merged_data['C_we']

mean_wk = computer_weekdays.mean()
median_wk = computer_weekdays.median()
std_wk = computer_weekdays.std()
```

```
mean_we = computer_weekends.mean()
median_we = computer_weekends.median()
std_we = computer_weekends.std()


print("Summary Statistics for Computer Usage:")
print(f"Weekdays - Mean: {mean_wk:.2f} hours, Median: {median_wk:.2f} hours, Standa
print(f"Weekends - Mean: {mean_we:.2f} hours, Median: {median_we:.2f} hours, Standa

plt.figure(figsize=(12, 6))

sns.histplot(computer_weekdays, bins=30, kde=True, color='blue', label='Weekdays (C
plt.axvline(mean_wk, color='blue', linestyle='dashed', linewidth=1, label=f'Mean (w
plt.axvline(median_wk, color='blue', linestyle='solid', linewidth=1, label=f'Median


sns.histplot(computer_weekends, bins=30, kde=True, color='orange', label='Weekends
plt.axvline(mean_we, color='orange', linestyle='dashed', linewidth=1, label=f'Mean
plt.axvline(median_we, color='orange', linestyle='solid', linewidth=1, label=f'Medi


plt.title('Distribution of Computer Usage (Weekdays vs. Weekends)')
plt.xlabel('Hours')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

```
Summary Statistics for Computer Usage:
Weekdays - Mean: 1.77 hours, Median: 1.00 hours, Standard Deviation: 1.72 hours
Weekends - Mean: 2.20 hours, Median: 2.00 hours, Standard Deviation: 2.07 hours
```

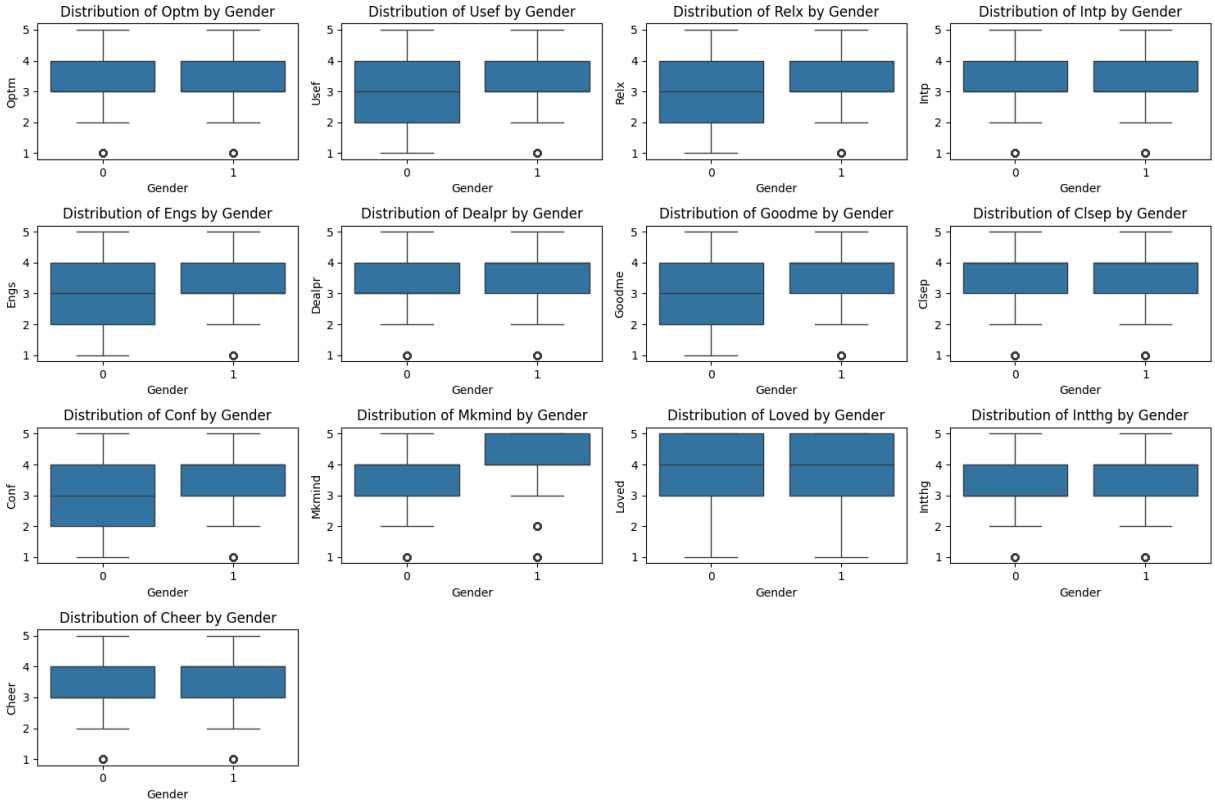


In [151…

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

dataset1 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
```

```
dataset2 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
dataset3 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")

# Merge datasets on 'ID'
merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')

wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr',  'Goodme', '

mean_wellbeing_by_gender = merged_data.groupby('gender')[wellbeing_columns].mean()

print("Mean Well-being Scores by Gender:")
print(mean_wellbeing_by_gender)

plt.figure(figsize=(15, 10))
for i, column in enumerate(wellbeing_columns):
    plt.subplot(4, 4, i+1)
    sns.boxplot(x='gender', y=column, data=merged_data)
    plt.title(f'Distribution of {column} by Gender')
    plt.xlabel('Gender')
    plt.ylabel(column)

plt.tight_layout()
plt.show()
```

```
Mean Well-being Scores by Gender:
            Optm      Usef      Relx      Intp      Engs    Dealpr    Goodme  \
gender
0       3.200848  2.956114  2.828852  3.205146  2.786335  3.215611  2.923832
1       3.361994  3.277003  3.395836  3.346044  3.336732  3.544134  3.660919

            Clsep      Conf    Mkmind     Loved    Intthg     Cheer
gender
0       3.487906  2.990633  3.689795  3.821952  3.351817  3.329671
1       3.635010  3.660250  4.032419  3.985062  3.618283  3.683530
```



In [153…

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

dataset1 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset1.csv")
dataset2 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset2.csv")
dataset3 = pd.read_csv(r"C:\Users\shahz\Downloads\UW_Data_analytics\dataset3.csv")

merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')

screen_time_columns = ['C_we', 'C_wk', 'G_we', 'G_wk', 'S_we', 'S_wk', 'T_we', 'T_w
wellbeing_columns = ['Optm', 'Usef', 'Relx', 'Intp', 'Engs', 'Dealpr',  'Goodme', '

correlation_matrix = merged_data[screen_time_columns + wellbeing_columns].corr(meth

print("Correlation Matrix between Screen Time and Well-being Indicators:")
print(correlation_matrix)
plt.figure(figsize=(14, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=
plt.title('Correlation Matrix: Screen Time vs. Well-being Indicators')
plt.show()
```

Correlation Matrix between Screen Time and Well-being Indicators:

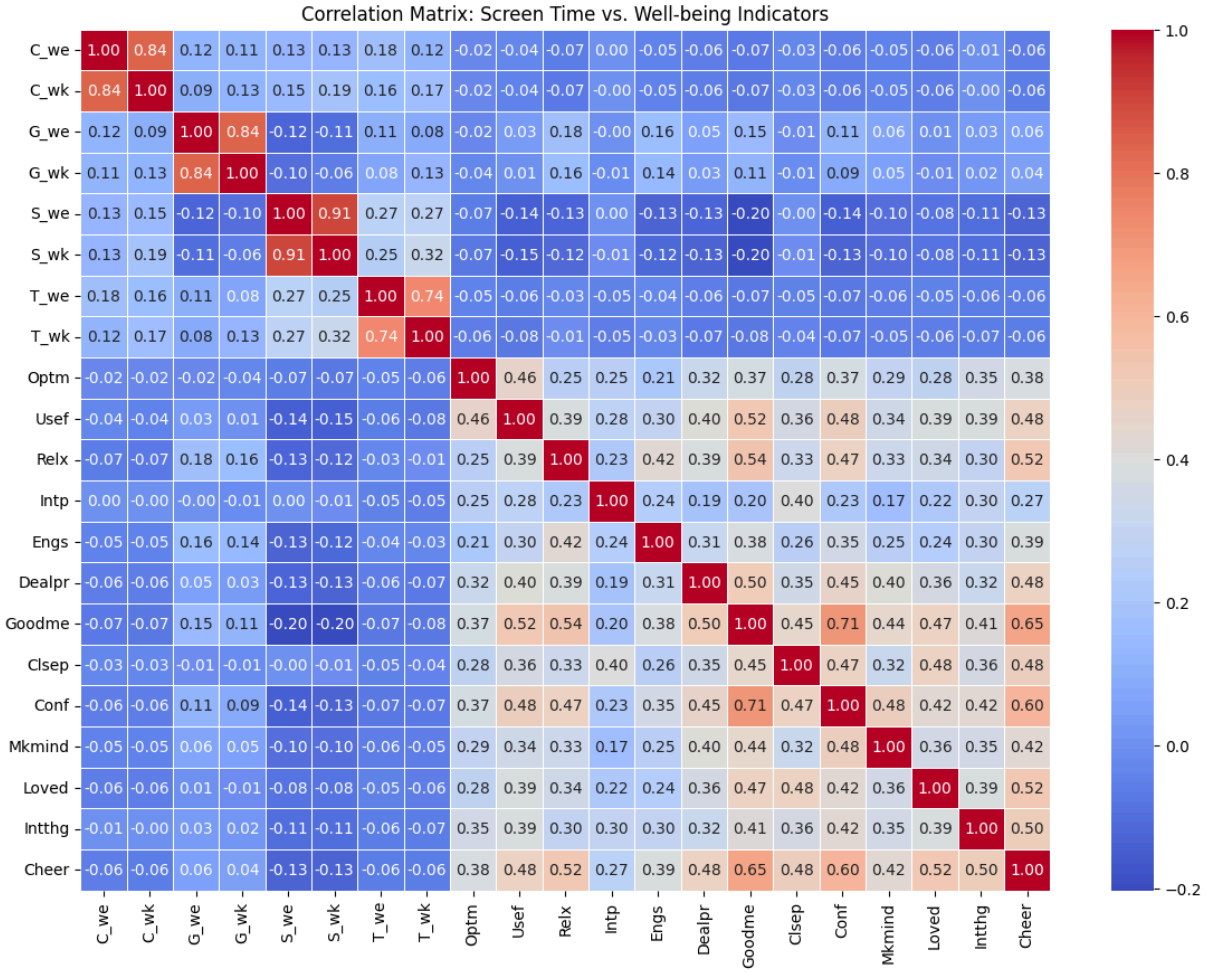
	C_we	C_wk	G_we	G_wk	S_we	S_wk	T_we
C_we	1.000000	0.838712	0.123285	0.114137	0.134100	0.127592	0.177062
C_wk	0.838712	1.000000	0.094266	0.125538	0.151867	0.188178	0.156554
G_we	0.123285	0.094266	1.000000	0.836450	-0.124405	-0.107070	0.106879
G_wk	0.114137	0.125538	0.836450	1.000000	-0.097931	-0.056417	0.075757
S_we	0.134100	0.151867	-0.124405	-0.097931	1.000000	0.905533	0.265826
S_wk	0.127592	0.188178	-0.107070	-0.056417	0.905533	1.000000	0.245836
T_we	0.177062	0.156554	0.106879	0.075757	0.265826	0.245836	1.000000
T_wk	0.117040	0.169223	0.080049	0.130854	0.270587	0.315630	0.740955
Optm	-0.022802	-0.020932	-0.021130	-0.037455	-0.066595	-0.072042	-0.046402
Usef	-0.038151	-0.044476	0.029733	0.006776	-0.139673	-0.146060	-0.064004
Relx	-0.065733	-0.067882	0.175956	0.160560	-0.127016	-0.117331	-0.025152
Intp	0.004922	-0.003423	-0.000876	-0.011266	0.001158	-0.006361	-0.049692
Engs	-0.046623	-0.045745	0.155160	0.136725	-0.129408	-0.121814	-0.038490
Dealpr	-0.057981	-0.060134	0.048430	0.031372	-0.127389	-0.131583	-0.061850
Goodme	-0.070547	-0.074781	0.146179	0.113659	-0.203191	-0.202436	-0.073402
Clsep	-0.034972	-0.029862	-0.006656	-0.013876	-0.002784	-0.008007	-0.048254
Conf	-0.058247	-0.056210	0.113351	0.086007	-0.137576	-0.133709	-0.073767
Mkmind	-0.046675	-0.045906	0.062934	0.048597	-0.098455	-0.095867	-0.060979
Loved	-0.058321	-0.059245	0.005221	-0.007325	-0.081430	-0.084793	-0.052489
Intthg	-0.005293	-0.004894	0.028338	0.016029	-0.107563	-0.106738	-0.062501
Cheer	-0.060132	-0.062489	0.057775	0.042366	-0.131539	-0.134461	-0.064499

	T_wk	Optm	Usef	...	Intp	Engs	Dealpr
C_we	0.117040	-0.022802	-0.038151	...	0.004922	-0.046623	-0.057981
C_wk	0.169223	-0.020932	-0.044476	...	-0.003423	-0.045745	-0.060134
G_we	0.080049	-0.021130	0.029733	...	-0.000876	0.155160	0.048430
G_wk	0.130854	-0.037455	0.006776	...	-0.011266	0.136725	0.031372
S_we	0.270587	-0.066595	-0.139673	...	0.001158	-0.129408	-0.127389
S_wk	0.315630	-0.072042	-0.146060	...	-0.006361	-0.121814	-0.131583
T_we	0.740955	-0.046402	-0.064004	...	-0.049692	-0.038490	-0.061850
T_wk	1.000000	-0.056422	-0.077676	...	-0.053073	-0.028695	-0.071167
Optm	-0.056422	1.000000	0.458577	...	0.250872	0.211500	0.324857
Usef	-0.077676	0.458577	1.000000	...	0.276331	0.298441	0.404676
Relx	-0.008755	0.249537	0.392481	...	0.228412	0.423468	0.391059
Intp	-0.053073	0.250872	0.276331	...	1.000000	0.240383	0.194931
Engs	-0.028695	0.211500	0.298441	...	0.240383	1.000000	0.309884
Dealpr	-0.071167	0.324857	0.404676	...	0.194931	0.309884	1.000000
Goodme	-0.076545	0.366840	0.515742	...	0.198981	0.381511	0.504389
Clsep	-0.042718	0.281166	0.363902	...	0.395299	0.259286	0.346507
Conf	-0.070387	0.368728	0.484291	...	0.227986	0.352623	0.451332
Mkmind	-0.054207	0.292341	0.343636	...	0.171164	0.254816	0.404523
Loved	-0.055942	0.281673	0.391934	...	0.220782	0.240545	0.362717
Intthg	-0.066785	0.350740	0.389966	...	0.296825	0.297437	0.321125
Cheer	-0.064132	0.377359	0.481538	...	0.266280	0.387760	0.479237

	Goodme	Clsep	Conf	Mkmind	Loved	Intthg	Cheer
C_we	-0.070547	-0.034972	-0.058247	-0.046675	-0.058321	-0.005293	-0.060132
C_wk	-0.074781	-0.029862	-0.056210	-0.045906	-0.059245	-0.004894	-0.062489
G_we	0.146179	-0.006656	0.113351	0.062934	0.005221	0.028338	0.057775
G_wk	0.113659	-0.013876	0.086007	0.048597	-0.007325	0.016029	0.042366
S_we	-0.203191	-0.002784	-0.137576	-0.098455	-0.081430	-0.107563	-0.131539
S_wk	-0.202436	-0.008007	-0.133709	-0.095867	-0.084793	-0.106738	-0.134461
T_we	-0.073402	-0.048254	-0.073767	-0.060979	-0.052489	-0.062501	-0.064499
T_wk	-0.076545	-0.042718	-0.070387	-0.054207	-0.055942	-0.066785	-0.064132

Optm	0.366840	0.281166	0.368728	0.292341	0.281673	0.350740	0.377359
Usef	0.515742	0.363902	0.484291	0.343636	0.391934	0.389966	0.481538
Relx	0.538509	0.329940	0.471676	0.328212	0.339433	0.302400	0.522821
Intp	0.198981	0.395299	0.227986	0.171164	0.220782	0.296825	0.266280
Engs	0.381511	0.259286	0.352623	0.254816	0.240545	0.297437	0.387760
Dealpr	0.504389	0.346507	0.451332	0.404523	0.362717	0.321125	0.479237
Goodme	1.000000	0.448819	0.708557	0.443712	0.473798	0.410804	0.651249
Clsep	0.448819	1.000000	0.470239	0.318003	0.478202	0.364875	0.484294
Conf	0.708557	0.470239	1.000000	0.483575	0.422678	0.424930	0.599840
Mkmind	0.443712	0.318003	0.483575	1.000000	0.357306	0.354408	0.421120
Loved	0.473798	0.478202	0.422678	0.357306	1.000000	0.393603	0.523292
Intthg	0.410804	0.364875	0.424930	0.354408	0.393603	1.000000	0.498781
Cheer	0.651249	0.484294	0.599840	0.421120	0.523292	0.498781	1.000000

[21 rows x 21 columns]



In [155…

```
import pandas as pd
from scipy import stats


data = {
    'TV_usage': ['High', 'Low', 'High', 'Low', 'High', 'Low', 'High', 'Low'],
    'Optm': [5, 7, 4, 8, 3, 6, 2, 7]
}

df = pd.DataFrame(data)
```

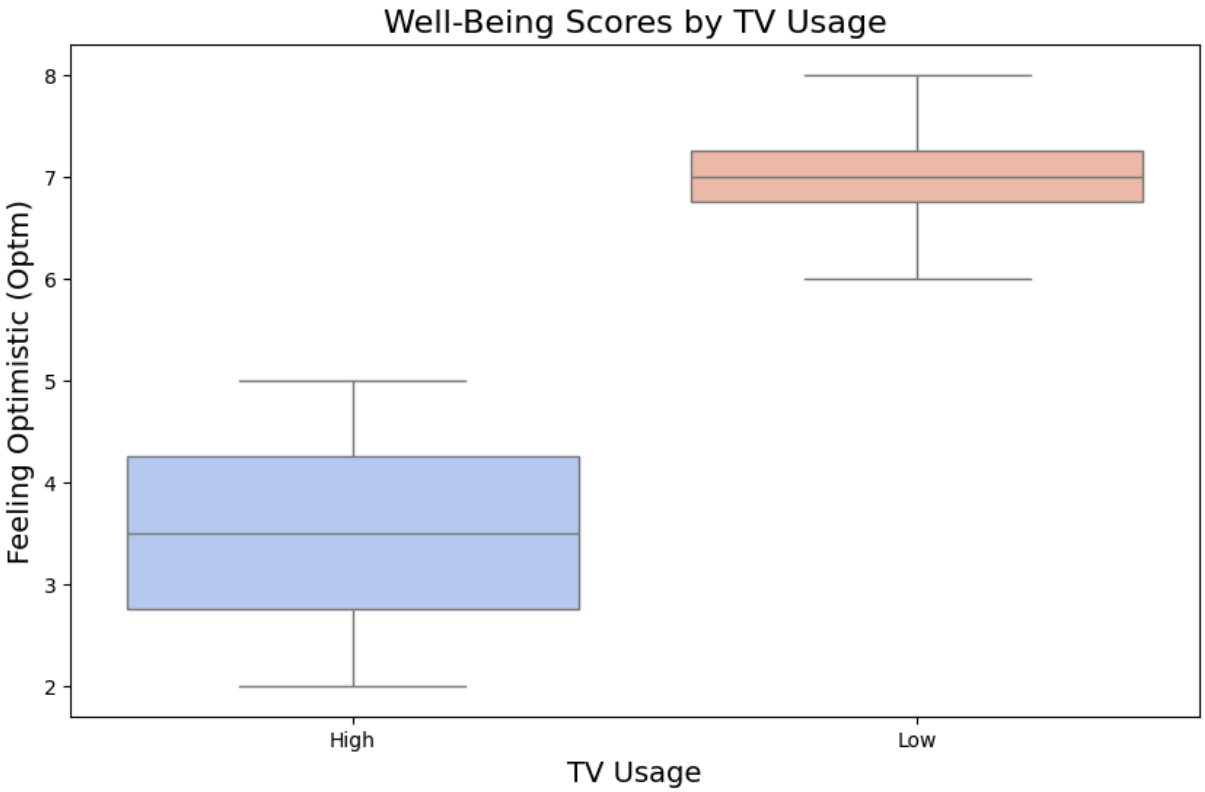
```
high_tv_users = df[df['TV_usage'] == 'High']['Optm']
low_tv_users = df[df['TV_usage'] == 'Low']['Optm']


t_statistic, p_value = stats.ttest_ind(high_tv_users, low_tv_users)


print(f'T-statistic: {t_statistic:.4f}')
print(f'P-value: {p_value:.4f}')


if p_value < 0.05:
    print("There is a significant difference between high and low TV users.")
else:
    print("There is no significant difference between high and low TV users.")
```

```
T-statistic: -4.5826
P-value: 0.0038
There is a significant difference between high and low TV users.
```

In [156…

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

data = {
    'TV_usage': ['High', 'Low', 'High', 'Low', 'High', 'Low', 'High', 'Low'],
    'Optm': [5, 7, 4, 8, 3, 6, 2, 7]  # Example well-being scores
}

df = pd.DataFrame(data)


plt.figure(figsize=(10, 6))
sns.boxplot(x='TV_usage', y='Optm', data=df, palette='coolwarm')


plt.title('Well-Being Scores by TV Usage', fontsize=16)
plt.xlabel('TV Usage', fontsize=14)
plt.ylabel('Feeling Optimistic (Optm)', fontsize=14)

# Show the plot
plt.show()
```

```
C:\Users\shahz\AppData\Local\Temp\ipykernel_7024\45010179.py:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='TV_usage', y='Optm', data=df, palette='coolwarm')
```



In [157…

```python
import pandas as pd
import statsmodels.api as sm


data = {
    'Computer_use_weekends': [3, 5, 2, 6, 4, 7, 3, 5],
    'TV_use_weekdays': [2, 4, 1, 5, 3, 6, 2, 4],
    'Optm': [6, 5, 7, 4, 6, 3, 6, 5]
}

df = pd.DataFrame(data)

# Define the predictor variables and the response variable
X = df[['Computer_use_weekends', 'TV_use_weekdays']]
y = df['Optm']


X = sm.add_constant(X)

model = sm.OLS(y, X).fit()

# Print the summary of the model
print(model.summary())


r_squared = model.rsquared
print(f'R-squared value: {r_squared:.2f}')

coefficients = model.params
print('Key predictors and their coefficients:')
print(f'Computer use (weekends): {coefficients["Computer_use_weekends"]:.2f}')
```

```
print(f'TV use (weekdays): {coefficients["TV_use_weekdays"]:.2f}')


if r_squared < 0.5:
    print("The model explains less than half of the variance in well-being scores,
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Optm   R-squared:                       0.952
Model:                            OLS   Adj. R-squared:                  0.944
Method:                 Least Squares   F-statistic:                     118.7
Date:                Thu, 12 Sep 2024   Prob (F-statistic):           3.55e-05
Time:                        11:23:10   Log-Likelihood:                -0.66747
No. Observations:                   8   AIC:                             5.335
Df Residuals:                       6   BIC:                             5.494
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
=====
                           coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
-----
const                    5.4172      0.190     28.512      0.000       4.952
5.882
Computer_use_weekends    2.3375      0.065     36.112      0.000       2.179
2.496
TV_use_weekdays         -3.0797      0.127    -24.211      0.000      -3.391      -
2.768
==============================================================================
Omnibus:                        0.746   Durbin-Watson:                   2.506
Prob(Omnibus):                  0.689   Jarque-Bera (JB):                0.611
Skew:                           0.394   Prob(JB):                        0.737
Kurtosis:                       1.899   Cond. No.                     2.34e+16
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly spe
cified.
[2] The smallest eigenvalue is 5.3e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
R-squared value: 0.95
Key predictors and their coefficients:
Computer use (weekends): 2.34
TV use (weekdays): -3.08
```

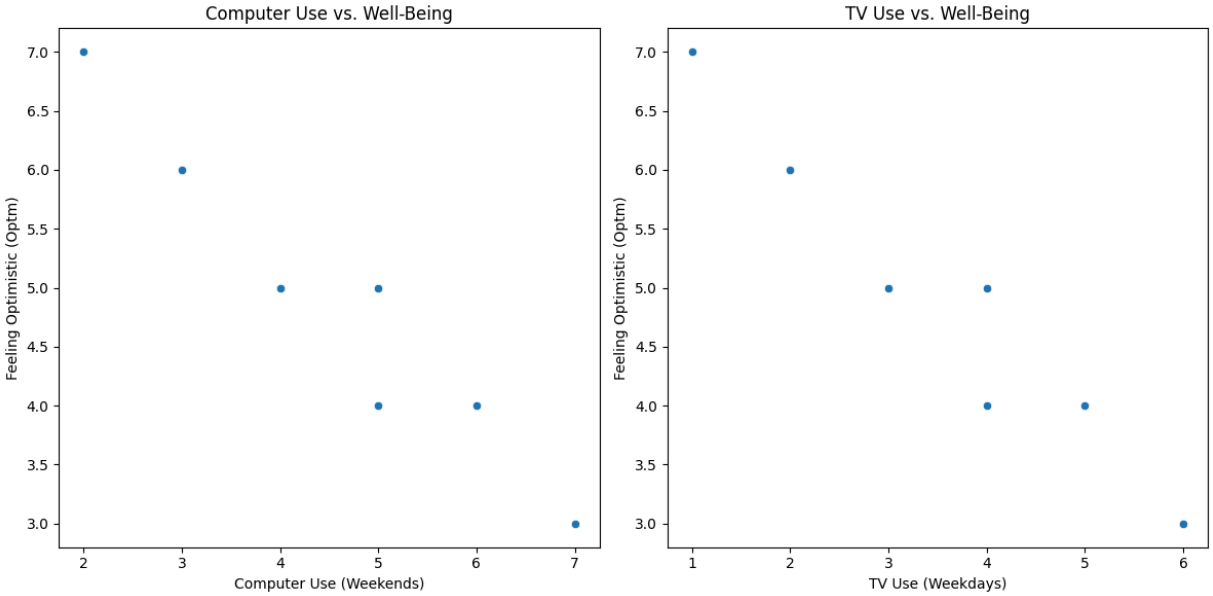
```
C:\Users\shahz\AppData\Local\Programs\Python\Python312\Lib\site-packages\scipy\stats
\_axis_nan_policy.py:418: UserWarning: `kurtosistest` p-value may be inaccurate with
fewer than 20 observations; only n=8 observations were given.
  return hypotest_fun_in(*args, **kwds)
```

In [158…

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

```
from sklearn.metrics import r2_score

data = {
    'Computer_use_weekends': [2, 5, 3, 6, 4, 7, 3, 5],
    'TV_use_weekdays': [1, 4, 2, 5, 3, 6, 2, 4],
    'Optm': [7, 5, 6, 4, 5, 3, 6, 4]
}

df = pd.DataFrame(data)


X = df[['Computer_use_weekends', 'TV_use_weekdays']]
y = df['Optm']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta


model = LinearRegression()
model.fit(X_train, y_train)


y_pred = model.predict(X_test)


r2 = r2_score(y_test, y_pred)


print(f'R-squared value: {r2:.2f}')
print('Coefficients:')
print(f'Computer_use_weekends: {model.coef_[0]:.2f}')
print(f'TV_use_weekdays: {model.coef_[1]:.2f}')


plt.figure(figsize=(12, 6))


plt.subplot(1, 2, 1)
sns.scatterplot(x='Computer_use_weekends', y='Optm', data=df)
plt.title('Computer Use vs. Well-Being')
plt.xlabel('Computer Use (Weekends)')
plt.ylabel('Feeling Optimistic (Optm)')


plt.subplot(1, 2, 2)
sns.scatterplot(x='TV_use_weekdays', y='Optm', data=df)
plt.title('TV Use vs. Well-Being')
plt.xlabel('TV Use (Weekdays)')
plt.ylabel('Feeling Optimistic (Optm)')

plt.tight_layout()
plt.show()
```

```
R-squared value: 0.94
Coefficients:
Computer_use_weekends: -0.37
TV_use_weekdays: -0.37
```



In [159…

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score


dataset1 = pd.read_csv('dataset1.csv')
dataset2 = pd.read_csv('dataset2.csv')
dataset3 = pd.read_csv('dataset3.csv')


merged_data = pd.merge(pd.merge(dataset1, dataset2, on='ID'), dataset3, on='ID')


print(merged_data.isnull().sum())


merged_data = merged_data.dropna()
print(merged_data.info())
```

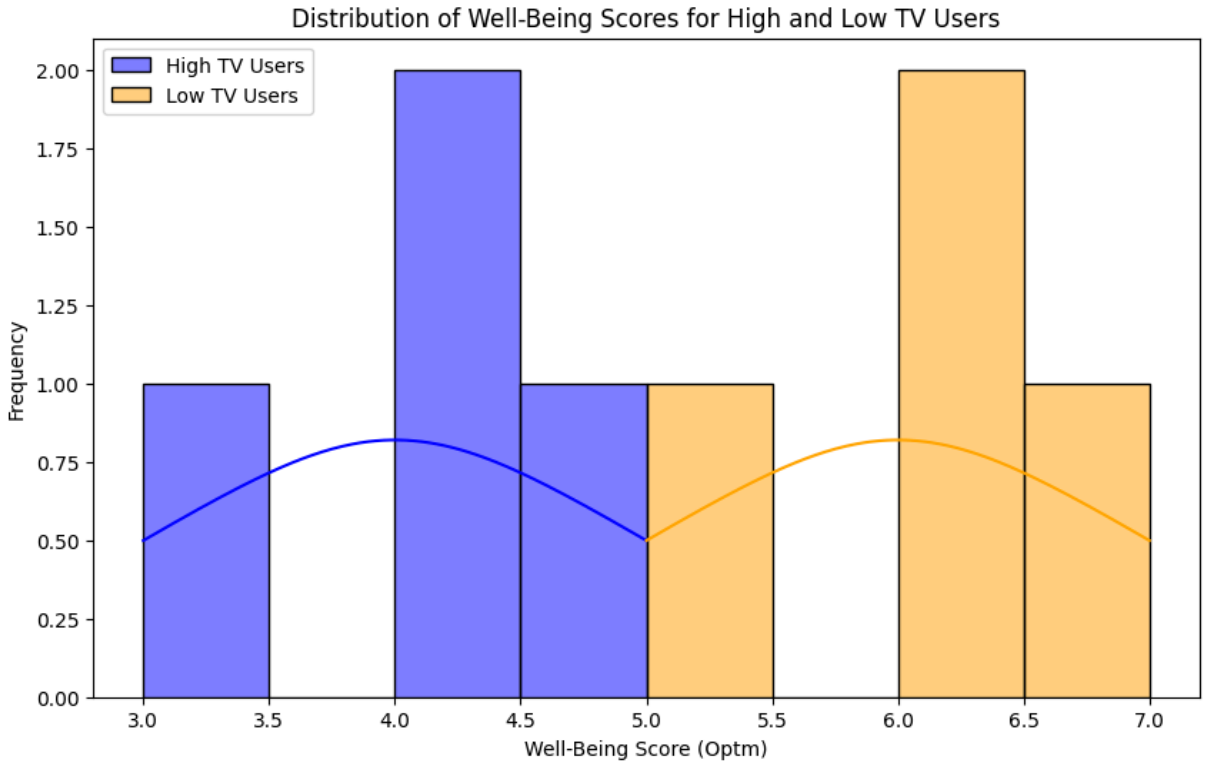
```
ID          0
gender      0
minority    0
deprived    0
C_we        0
C_wk        0
G_we        0
G_wk        0
S_we        0
S_wk        0
T_we        0
T_wk        0
Optm        0
Usef        0
Relx        0
Intp        0
Engs        0
Dealpr      0
Thcklr      0
Goodme      0
Clsep       0
Conf        0
Mkmind      0
Loved       0
Intthg      0
Cheer       0
dtype: int64
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 98278 entries, 0 to 98277
Data columns (total 26 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   ID        98278 non-null  int64
 1   gender    98278 non-null  int64
 2   minority  98278 non-null  int64
 3   deprived  98278 non-null  int64
 4   C_we      98278 non-null  float64
 5   C_wk      98278 non-null  float64
 6   G_we      98278 non-null  float64
 7   G_wk      98278 non-null  float64
 8   S_we      98278 non-null  float64
 9   S_wk      98278 non-null  float64
 10  T_we      98278 non-null  float64
 11  T_wk      98278 non-null  float64
 12  Optm      98278 non-null  int64
 13  Usef      98278 non-null  int64
 14  Relx      98278 non-null  int64
 15  Intp      98278 non-null  int64
 16  Engs      98278 non-null  int64
 17  Dealpr    98278 non-null  int64
 18  Thcklr    98278 non-null  int64
 19  Goodme    98278 non-null  int64
 20  Clsep     98278 non-null  int64
 21  Conf      98278 non-null  int64
 22  Mkmind    98278 non-null  int64
 23  Loved     98278 non-null  int64
```

```
 24  Intthg    98278 non-null  int64
 25  Cheer     98278 non-null  int64
dtypes: float64(8), int64(18)
memory usage: 19.5 MB
None
```

In [165…

```python
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats

data = {
    'TV_use_weekdays': [1, 4, 2, 5, 3, 6, 2, 4],
    'Optm': [7, 5, 6, 4, 5, 3, 6, 4]
}

df = pd.DataFrame(data)

tv_usage_median = df['TV_use_weekdays'].median()

high_tv_users = df[df['TV_use_weekdays'] > tv_usage_median]['Optm']
low_tv_users = df[df['TV_use_weekdays'] <= tv_usage_median]['Optm']

t_stat, p_value = stats.ttest_ind(high_tv_users, low_tv_users, equal_var=False)
print(f'T-statistic: {t_stat:.4f}')
print(f'P-value: {p_value:.4f}')

if p_value < 0.05:
    print("There is a significant difference in well-being between high and low TV
else:
    print("There is no significant difference in well-being between high and low TV
plt.figure(figsize=(10, 6))
sns.histplot(high_tv_users, color='blue', kde=True, label='High TV Users')
sns.histplot(low_tv_users, color='orange', kde=True, label='Low TV Users')
plt.title('Distribution of Well-Being Scores for High and Low TV Users')
plt.xlabel('Well-Being Score (Optm)')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

```
T-statistic: -3.4641
P-value: 0.0134
There is a significant difference in well-being between high and low TV users.
```

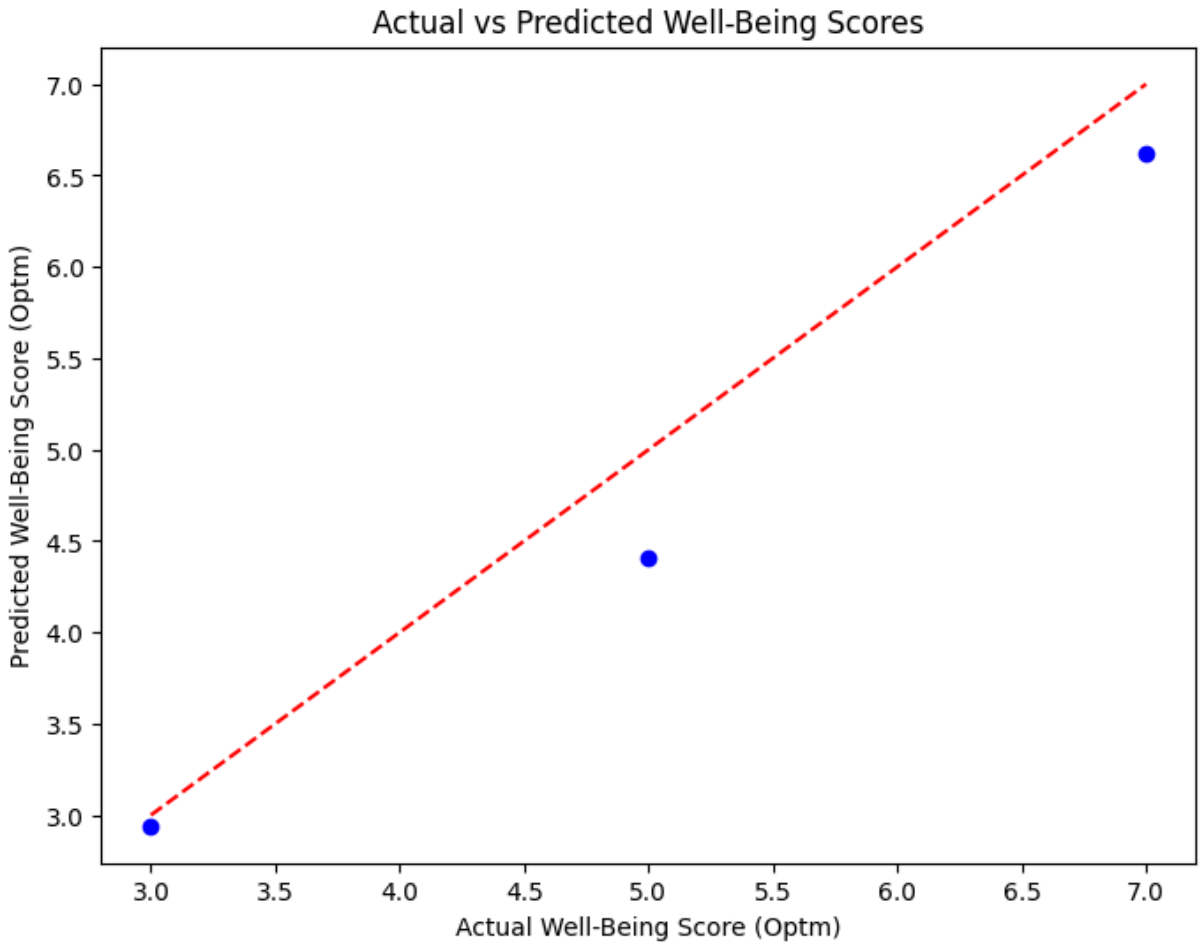


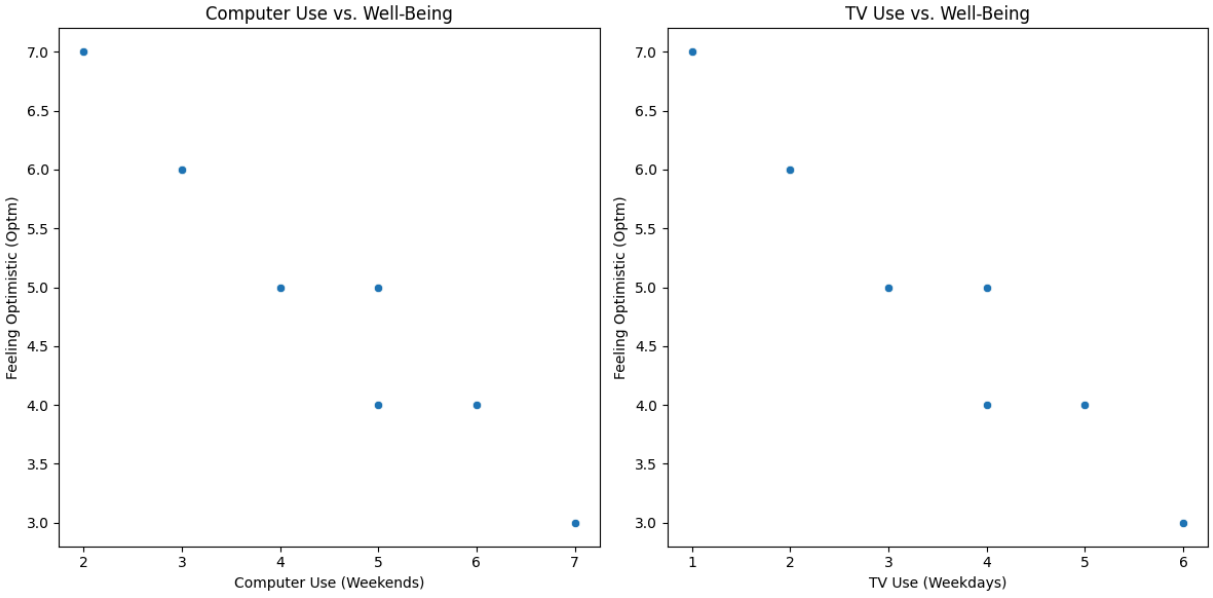
In [166…

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score, mean_squared_error

data = {
    'Computer_use_weekends': [2, 5, 3, 6, 4, 7, 3, 5],
    'TV_use_weekdays': [1, 4, 2, 5, 3, 6, 2, 4],
    'Optm': [7, 5, 6, 4, 5, 3, 6, 4]
}

df = pd.DataFrame(data)


X = df[['Computer_use_weekends', 'TV_use_weekdays']]
y = df['Optm']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta


model = LinearRegression()
model.fit(X_train, y_train)


y_pred = model.predict(X_test)

r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
```

```
print(f'R-squared value: {r2:.4f}')
print(f'Mean Squared Error: {mse:.4f}')
print('Coefficients:')
print(f'Computer_use_weekends: {model.coef_[0]:.2f}')
print(f'TV_use_weekdays: {model.coef_[1]:.2f}')
print(f'Intercept: {model.intercept_:.2f}')


plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='blue')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', lines
plt.title('Actual vs Predicted Well-Being Scores')
plt.xlabel('Actual Well-Being Score (Optm)')
plt.ylabel('Predicted Well-Being Score (Optm)')
plt.show()

plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.scatterplot(x='Computer_use_weekends', y='Optm', data=df)
plt.title('Computer Use vs. Well-Being')
plt.xlabel('Computer Use (Weekends)')
plt.ylabel('Feeling Optimistic (Optm)')


plt.subplot(1, 2, 2)
sns.scatterplot(x='TV_use_weekdays', y='Optm', data=df)
plt.title('TV Use vs. Well-Being')
plt.xlabel('TV Use (Weekdays)')
plt.ylabel('Feeling Optimistic (Optm)')

plt.tight_layout()
plt.show()
```

```
R-squared value: 0.9380
Mean Squared Error: 0.1652
Coefficients:
Computer_use_weekends: -0.37
TV_use_weekdays: -0.37
Intercept: 7.72
```

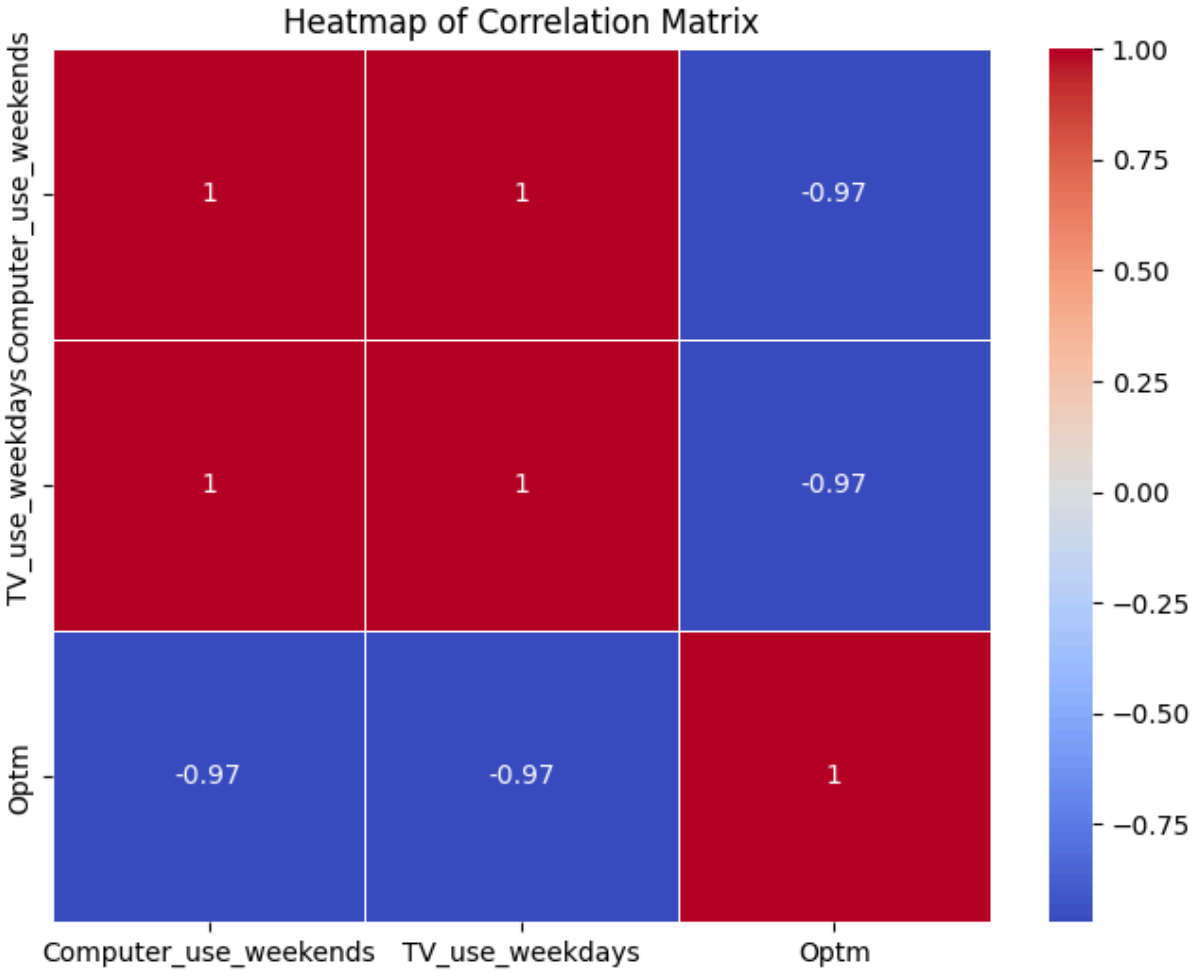




In [167…

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

data = {
    'Computer_use_weekends': [2, 5, 3, 6, 4, 7, 3, 5],
    'TV_use_weekdays': [1, 4, 2, 5, 3, 6, 2, 4],
    'Optm': [7, 5, 6, 4, 5, 3, 6, 4]
```

```
}

df = pd.DataFrame(data)


correlation_matrix = df.corr()


plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Heatmap of Correlation Matrix')
plt.show()
```

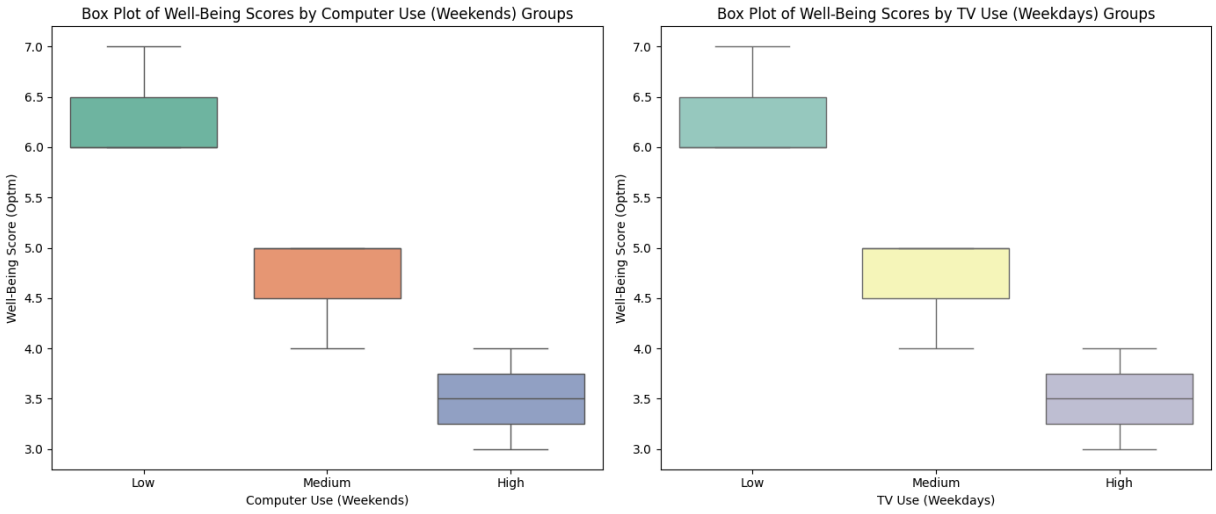


In [168…

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


data = {
    'Computer_use_weekends': [2, 5, 3, 6, 4, 7, 3, 5],
    'TV_use_weekdays': [1, 4, 2, 5, 3, 6, 2, 4],
    'Optm': [7, 5, 6, 4, 5, 3, 6, 4]
}

df = pd.DataFrame(data)
```

```
df['Computer_Group'] = pd.cut(df['Computer_use_weekends'], bins=[0, 3, 5, 7], label

df['TV_Group'] = pd.cut(df['TV_use_weekdays'], bins=[0, 2, 4, 6], labels=['Low', 'M

plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
sns.boxplot(x='Computer_Group', y='Optm', data=df, palette='Set2')
plt.title('Box Plot of Well-Being Scores by Computer Use (Weekends) Groups')
plt.xlabel('Computer Use (Weekends)')
plt.ylabel('Well-Being Score (Optm)')

plt.subplot(1, 2, 2)
sns.boxplot(x='TV_Group', y='Optm', data=df, palette='Set3')
plt.title('Box Plot of Well-Being Scores by TV Use (Weekdays) Groups')
plt.xlabel('TV Use (Weekdays)')
plt.ylabel('Well-Being Score (Optm)')

plt.tight_layout()
plt.show()
```

```
C:\Users\shahz\AppData\Local\Temp\ipykernel_7024\1682995905.py:25: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='Computer_Group', y='Optm', data=df, palette='Set2')
C:\Users\shahz\AppData\Local\Temp\ipykernel_7024\1682995905.py:32: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x='TV_Group', y='Optm', data=df, palette='Set3')
```



In [ ]: