jupyter Practicle2 Last Checkpoint: a few seconds ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

In [45]:
```
import pandas as pd
import numpy as np

# Generate sample data
np.random.seed(0)
data = {
    'Student_ID': range(1, 101),
    'Math': np.random.randint(50, 100, 100).astype(object),
    'Physics': np.random.randint(40, 95, 100),
    'English': np.random.randint(30, 90, 100).astype(object),
    'Attendance': np.random.uniform(0.7, 1, 100).astype(object),
    'GPA': np.random.uniform(2.5, 4, 100)
}

# Introduce some missing values and inconsistencies
data['Math'][10] = np.nan
data['Physics'][20] = 200  # introducing an outlier
data['English'][30] = 'A'  # introducing inconsistency

# Create DataFrame
df = pd.DataFrame(data)
```

In [46]:
```
df
```

Out[46]:

	Student_ID	Math	Physics	English	Attendance	GPA
0	1	94	45	63	0.983236	3.573342
1	2	97	81	70	0.882476	3.229239
2	3	50	75	62	0.878997	3.562822
3	4	53	40	66	0.935093	3.247210
4	5	53	71	89	0.850008	3.766825
...	...	...	...	...	...	...
95	96	65	65	30	0.881943	3.574366
96	97	63	56	41	0.765521	2.540603
97	98	71	94	64	0.836551	3.597096
98	99	98	93	35	0.963661	3.650445
99	100	99	59	83	0.84768	2.514650

100 rows × 6 columns

In [47]:
```
missing_values = df.isnull().sum()
print("Missing Values:\n", missing_values)
```

```
Missing Values:
 Student_ID    0
Math           1
Physics        0
English        0
Attendance     0
GPA            0
dtype: int64
```

In [48]:
```
# Check for inconsistencies in 'English_Score'
inconsistent_data = pd.to_numeric(df['English'], errors='coerce').isnull().sum()
print("Inconsistencies in English_Score:", inconsistent_data)
```

```
Inconsistencies in English_Score: 1
```

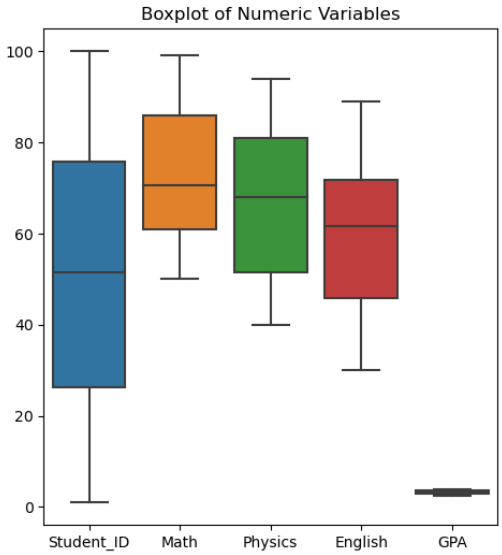
In [49]:
```
# Dealing with missing values and inconsistencies
# For missing values, we can either drop rows or fill them with a suitable value
df['Math'].fillna(df['Math'].mean(), inplace=True)
df['Physics'] = np.where(df['Physics'] > 100, np.nan, df['Physics'])  # replacing outlier with NaN
df['English'] = pd.to_numeric(df['English'], errors='coerce')  # Convert to numeric, converting inconsistencies to NaN
df.dropna(inplace=True)  # Drop rows with NaN values
```

In [50]:
```
# Check if missing values and inconsistencies are handled
print("After handling missing values and inconsistencies:")
print(df.isnull().sum())
```

```
After handling missing values and inconsistencies:
Student_ID    0
Math          0
Physics       0
English       0
Attendance    0
GPA           0
dtype: int64
```

In [51]:
```
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import zscore
numeric_cols = df.select_dtypes(include=np.number).columns.tolist()
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.boxplot(data=df[numeric_cols])
plt.title('Boxplot of Numeric Variables')
```

Out[51]:
```
Text(0.5, 1.0, 'Boxplot of Numeric Variables')
```



In [52]:
```
# Data transformation on 'GPA' variable
# Log transformation to decrease skewness
z_scores = np.abs(zscore(df[numeric_cols]))
threshold = 3
outlier_indices = np.where(z_scores > threshold)[0]
df_cleaned = df.drop(outlier_indices)
df_cleaned.dropna(subset=['GPA'], inplace=True)
df_cleaned['GPA'] = np.log1p(df_cleaned['GPA'])
```

In [53]:
```
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.histplot(df_cleaned['GPA'], kde=True)
plt.title('Before Transformation')

plt.subplot(1, 2, 2)
sns.histplot(np.log1p(df['GPA']), kde=True)
plt.title('After Transformation')

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