In [ ]:

```
## Introduction

- Exploratory Data Analysis (EDA) is the process of analyzing datasets to summarize
- uncover patterns, detect anomalies, and form hypotheses using statistical and vis
- In this project, EDA is used to understand how students daily habits influence th
- The goal is not prediction, but understanding relationships and trends in the dat




## PROBLEM STATEMENT

- Students follow different daily routines such as study time, sleep duration, mobi
- These habits may significantly impact their academic performance

## Objective:

- To explore how daily study habits affect exam scores and identify the most influe
```

In [ ]:

```
# Dataset Description
- The dataset used in this project is self-created, containing realistic academic a

# Dataset Details:
- File name: study_habits.csv
- Number of records: 30-50 students
- Each row represents one student
- Each column represents a habit or performance metric
```

In [ ]:

```
# Features Explanation:

 | Column Name        | Description                        |
| ------------------ | ---------------------------------- |
| student_id         | Unique identifier for each student |
| study_hours        | Average hours studied per day      |
| sleep_hours        | Average daily sleep duration       |
| mobile_usage_hours | Daily mobile phone usage           |
| attendance_percent | Class attendance percentage        |
| mock_test_score    | Internal assessment score          |
| exam_score         | Final examination score            |
| stress_level       | Stress category (Low, Medium, High) |
```

In [ ]:

```
## Import Required Libraries
```

In [5]:

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

In [ ]:

```
- Load & Inspect Data
```

In [8]:

```
## LOAD THE DATASET
```

```
df = pd.read_csv("study_habits.csv")
```

In [ ]:

```

```

In [31]:

```
df
```

Out[31]:

	student_id	study_hours	sleep_hours	mobile_usage_hours	attendance_percent	mock_t
0	1	2.2	7.7	2.5	66	
1	2	5.8	6.9	1.0	70	
2	3	4.0	5.7	2.7	64	
3	4	2.3	5.1	1.9	63	
4	5	4.0	6.4	5.1	62	
5	6	3.3	7.0	1.4	74	
6	7	4.0	7.0	2.5	75	
7	8	3.7	7.5	3.8	60	
8	9	3.1	7.9	4.1	70	
9	10	5.3	5.8	3.9	92	
10	11	2.6	7.0	2.8	65	
11	12	1.4	5.0	2.7	74	
12	13	3.4	6.9	3.6	80	
13	14	4.5	6.5	3.1	73	
14	15	2.5	5.8	5.1	78	
15	16	5.3	5.6	5.1	67	
16	17	4.1	7.7	3.8	60	
17	18	5.4	6.2	3.0	62	
18	19	4.5	7.3	1.4	82	
19	20	5.1	6.9	1.7	76	
20	21	4.5	5.3	1.1	94	
21	22	5.9	5.6	1.9	77	
22	23	5.5	7.3	1.6	65	
23	24	3.0	7.0	5.3	94	
24	25	6.0	7.1	4.8	74	
25	26	2.0	6.5	4.2	65	
26	27	1.1	7.7	3.8	92	
27	28	4.0	5.9	2.9	83	
28	29	3.0	6.7	1.5	81	
29	30	5.5	5.3	4.7	84	

	student_id	study_hours	sleep_hours	mobile_usage_hours	attendance_percent	mock_t
30	31	5.1	5.6	1.1	79	
31	32	2.4	6.4	3.0	73	
32	33	1.9	5.1	4.9	84	
33	34	3.9	5.5	3.1	95	
34	35	2.1	5.7	3.3	66	
35	36	5.8	5.1	5.0	74	
36	37	4.7	5.5	5.4	61	
37	38	5.1	7.4	1.9	68	
38	39	2.7	7.0	3.6	69	
39	40	4.4	6.0	1.7	75	
40	41	3.3	6.1	3.3	76	
41	42	2.1	5.6	4.8	74	
42	43	3.5	6.3	2.1	83	
43	44	2.7	6.4	2.3	67	
44	45	5.4	7.2	5.2	87	
45	46	3.6	7.1	2.6	90	
46	47	2.6	6.2	2.9	90	
47	48	3.9	7.2	2.4	67	
48	49	1.4	8.0	3.3	67	
49	50	2.3	7.2	3.7	61	

In [ ]:

```
```

In [32]:

```
print(df.head())
```

```
   student_id  study_hours  sleep_hours  mobile_usage_hours  \
0           1          2.2          7.7                 2.5   
1           2          5.8          6.9                 1.0   
2           3          4.0          5.7                 2.7   
3           4          2.3          5.1                 1.9   
4           5          4.0          6.4                 5.1   

   attendance_percent  mock_test_score  exam_score stress_level  
0                  66               86          51       Medium  
1                  70               58          50         High  
2                  64               44          47         High  
3                  63               76          57       Medium  
4                  62               89          92       Medium  
```

In [33]:

```python
print(df.tail())
```

```
    student_id  study_hours  sleep_hours  mobile_usage_hours  \
45          46          3.6          7.1                 2.6   
46          47          2.6          6.2                 2.9   
47          48          3.9          7.2                 2.4   
48          49          1.4          8.0                 3.3   
49          50          2.3          7.2                 3.7   

    attendance_percent  mock_test_score  exam_score stress_level  
45                  90               44          77          Low  
46                  90               68          91          Low  
47                  67               80          95       Medium  
48                  67               56          66          Low  
49                  61               59          47          Low  
```

In [34]:

```python
df.shape
```

Out[34]:

```
(50, 8)
```

In [35]:

```python
df.describe(include="all")
```

Out[35]:

	student_id	study_hours	sleep_hours	mobile_usage_hours	attendance_percent	mo
count	50.00000	50.000000	50.000000	50.000000	50.000000	
unique	NaN	NaN	NaN	NaN	NaN	
top	NaN	NaN	NaN	NaN	NaN	
freq	NaN	NaN	NaN	NaN	NaN	
mean	25.50000	3.718000	6.458000	3.172000	74.560000	
std	14.57738	1.357622	0.845417	1.290632	10.128078	
min	1.00000	1.100000	5.000000	1.000000	60.000000	
25%	13.25000	2.600000	5.700000	2.150000	66.250000	
50%	25.50000	3.800000	6.450000	3.050000	74.000000	
75%	37.75000	5.000000	7.100000	4.050000	81.750000	
max	50.00000	6.000000	8.000000	5.400000	95.000000	

In [18]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50 entries, 0 to 49
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   student_id          50 non-null     int64  
 1   study_hours         50 non-null     float64
 2   sleep_hours         50 non-null     float64
 3   mobile_usage_hours  50 non-null     float64
 4   attendance_percent  50 non-null     int64  
 5   mock_test_score     50 non-null     int64  
 6   exam_score          50 non-null     int64  
 7   stress_level        50 non-null     object 
dtypes: float64(3), int64(4), object(1)
memory usage: 3.3+ KB
```

In [ ]:

In [ ]:

```
## DATA CLEANING

- Data cleaning ensures accuracy and reliability of analysis.

# Activities Performed

- Checked for missing values
- Identified duplicate records
- Verified realistic value ranges
- Ensured correct data types

# Importance:
```

```
- Dirty data leads to misleading insights
- Removing duplicates avoids biased results
- Clean data improves analysis quality
```

In [36]:

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

Out[36]:

```
student_id             0
study_hours            0
sleep_hours            0
mobile_usage_hours     0
attendance_percent     0
mock_test_score        0
exam_score             0
stress_level           0
dtype: int64
```

In [ ]:

```
## REMOVE DUPLICATES

- Missing values
- Incorrect data types
- Outliers
- Duplicates
```

In [37]:

```
df.drop_duplicates(inplace=True)
```

In [ ]:

```

```

In [ ]:

```
# Univariate Analysis
- Univariate analysis examines one variable at a time.

# Purpose:
- Understand distribution
- Identify skewness and outliers
- Analyze central tendency (mean, median)

# Examples:
- Distribution of study hours
-Distribution of sleep hours
-Exam score spread
```

In [ ]:

```

```

In [ ]:

```
## Distribution
```

In [ ]:

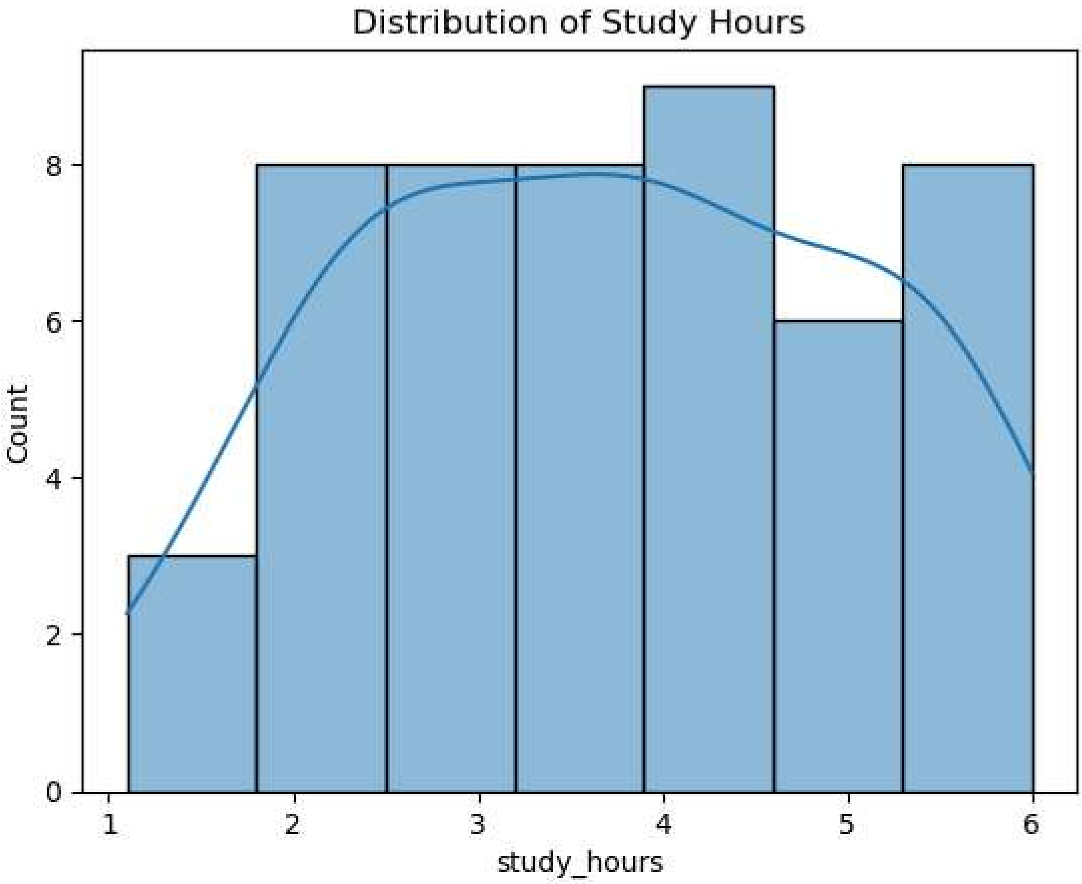
```

```

In [ ]:

```
- Study Hours Distribution
```

In [38]:

```
sns.histplot(df['study_hours'], kde=True)
plt.title("Distribution of Study Hours")
plt.show()
```

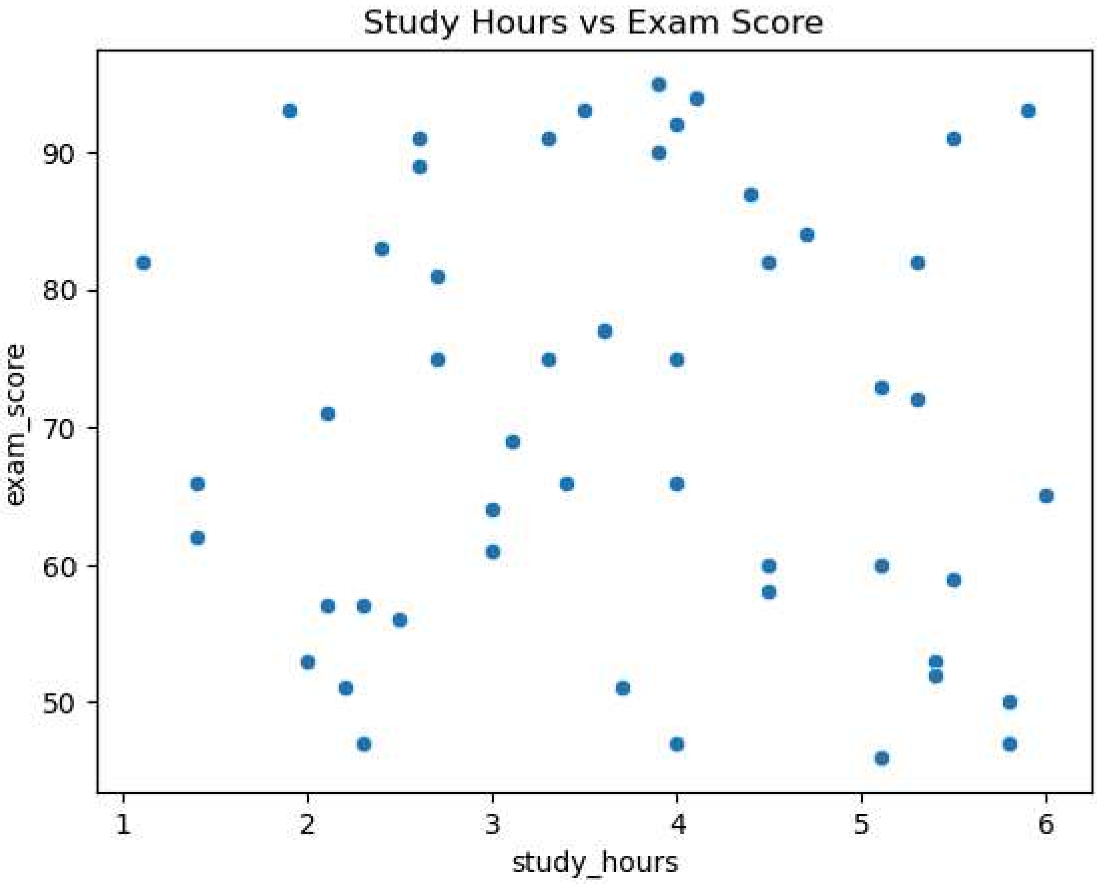


```
In [ ]: # Bivariate Analysis
        - Bivariate analysis studies the relationship between two variables.

        # Key Relationships Analyzed:
        - Study hours vs exam score
        - Mobile usage vs exam score
        - Sleep hours vs exam score

        # Importance:
        -Identifies positive or negative relationships
        -Helps understand which habits impact performance
        -Supports hypothesis formation
```

```
In [ ]: - Study Hours vs Exam Score
```

```
In [ ]:
```

```
In [23]: sns.scatterplot(x='study_hours', y='exam_score', data=df)
         plt.title("Study Hours vs Exam Score")
         plt.show()
```



In [ ]:

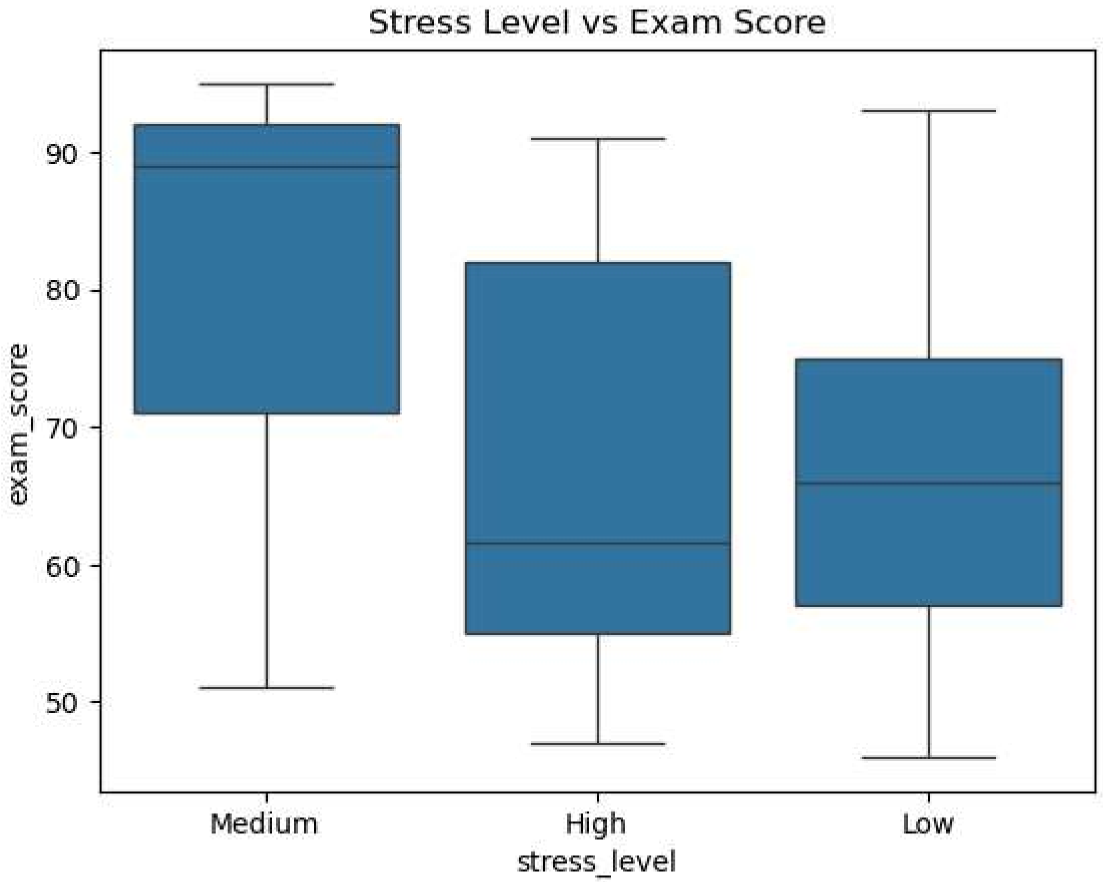
In [ ]:

```
- Mobile Usage vs Exam Score
```

In [ ]:

In [24]:

```
sns.boxplot(x='stress_level', y='exam_score', data=df)
plt.title("Stress Level vs Exam Score")
plt.show()
```



In [ ]:

```
# Multivariate Analysis
- Multivariate analysis examines multiple variables together to identify deeper pat

# Methods Used:
-Correlation matrix
-Heatmap visualization

# Purpose:
-Understand combined effects of multiple habits
-Identify strongest influencing factors
-Detect multicollinearity
This step provides a holistic view of student behavior and performance.
```

In [ ]:

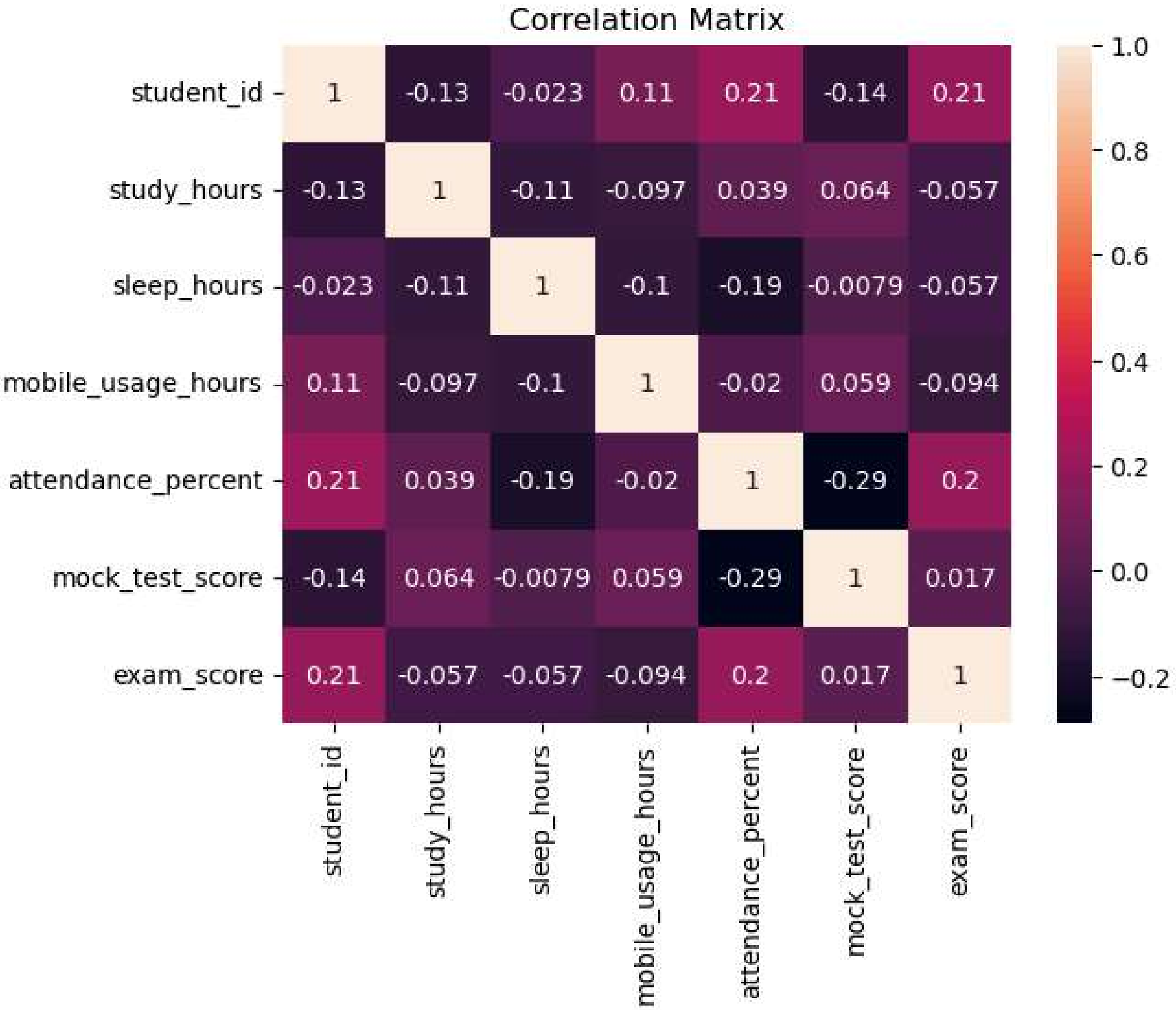
```
- Correlation Heatmap
```

In [ ]:

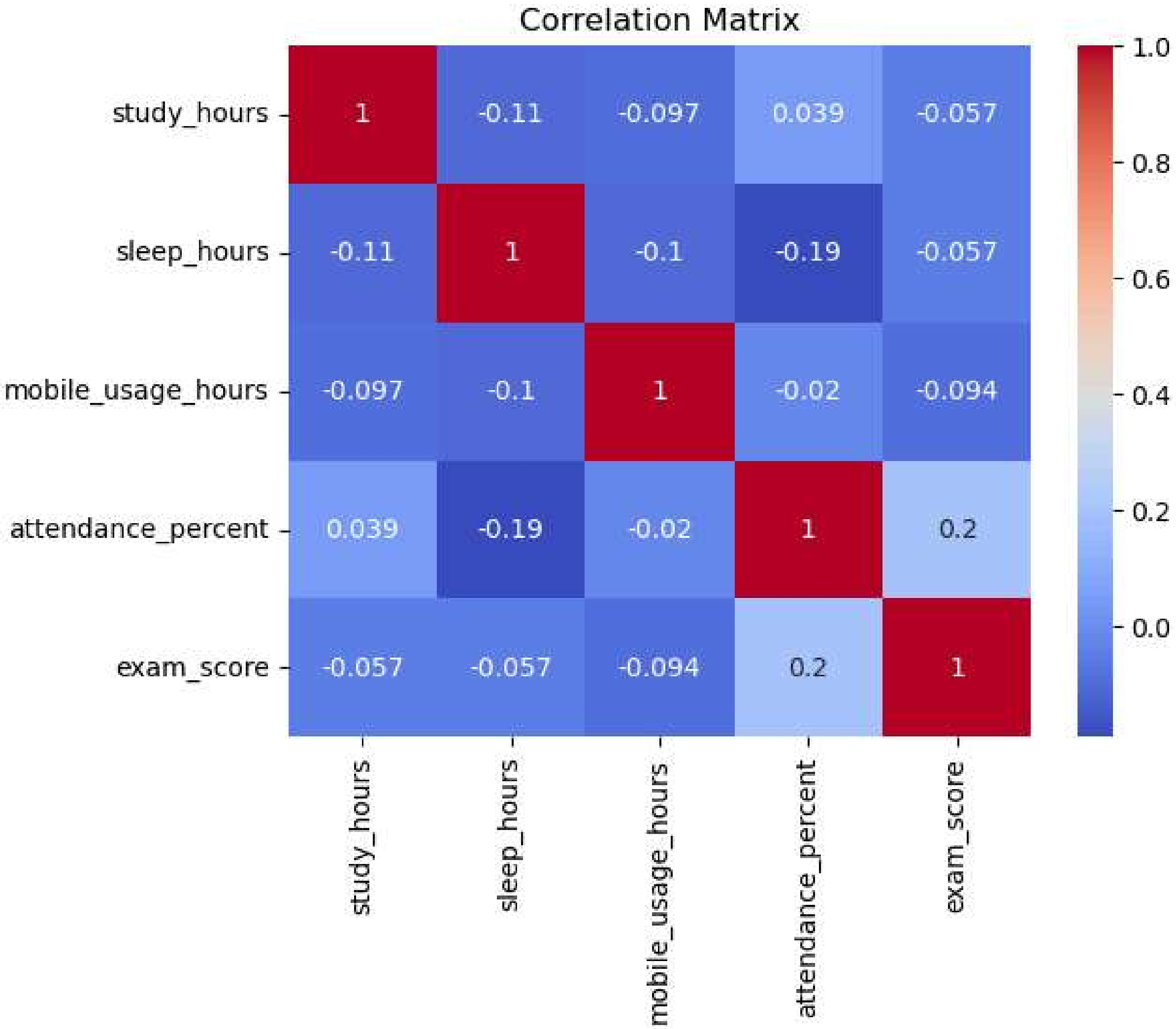
```
```

In [26]:

```
corr = df[['study_hours','sleep_hours','mobile_usage_hours',
           'attendance_percent','exam_score']].corr()

sns.heatmap(df.corr(numeric_only = True),annot = True)
plt.title("Correlation Matrix")
plt.show()
```



In [27]:
```
corr = df[['study_hours','sleep_hours','mobile_usage_hours',
           'attendance_percent','exam_score']].corr()

sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title("Correlation Matrix")
plt.show()
```



In [ ]:

```
# Conclusion
-The exploratory data analysis successfully identified key daily habits influencing
-Study time, attendance, sleep, and controlled mobile usage play a significant role

-This analysis can help educators and students focus on positive habits to improve
```

In [ ]:

```
# Tools & Technologies Used
-Python
-Pandas - data manipulation
-NumPy - numerical operations
-Matplotlib & Seaborn - data visualization
-Jupyter Notebook - analysis environment
```

In [ ]:

```
# Project Significance
-Demonstrates real-world analytical thinking
-Uses a self-created dataset
-Easy to explain in interviews
-Strong foundation for advanced analytics or ML projects
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: