```
#import the libraries

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

# Load the CSV files

morning_df=pd.read_csv(r"C:\Users\Saikiran\Downloads\Morning-28th-
June.csv")
afternoon_df=pd.read_csv(r"C:\Users\Saikiran\Downloads\Afternoon-28th-
June.csv")
evening_df=pd.read_csv(r"C:\Users\Saikiran\Downloads\Evening-28th-
June.csv")

# Display the first few rows of each dataframe to understand the
structure

morning.head()

             Name (Original Name)       User Email               Join
Time  \
0  Krishna Singh (Akshita Roshan)  akshita@agie.ai  06/28/2024
09:52:45 AM
1                    Chetan Kumar              NaN  06/28/2024
09:52:52 AM
2                  Akshita Roshan              NaN  06/28/2024
09:52:52 AM
3               Kshitij Tardalkar              NaN  06/28/2024
09:52:56 AM
4                    Chetan Kumar              NaN  06/28/2024
09:52:57 AM

               Leave Time  Duration (Minutes) Guest In Waiting Room
0  06/28/2024 10:17:06 AM                  25    No              No
1  06/28/2024 09:52:57 AM                   1   Yes             Yes
2  06/28/2024 09:53:08 AM                   1   Yes             Yes
3  06/28/2024 09:56:52 AM                   4   Yes             Yes
4  06/28/2024 10:16:12 AM                  24   Yes              No

afternoon.head()

            Name (Original Name)       User Email               Join
Time  \
0  Krisha Singh (Akshita Roshan)  akshita@agie.ai  06/28/2024 03:57:40
PM
1                         Tanuja              NaN  06/28/2024 03:57:49
PM
2                      Yash Goel              NaN  06/28/2024 03:57:53
PM
3                        Darshan              NaN  06/28/2024 03:57:53
```

```
PM
4                    sneha pawar              NaN  06/28/2024 03:57:56
PM

               Leave Time  Duration (Minutes) Guest In Waiting Room
0  06/28/2024 04:19:20 PM                  22    No              No
1  06/28/2024 03:58:04 PM                   1   Yes             Yes
2  06/28/2024 03:58:09 PM                   1   Yes             Yes
3  06/28/2024 03:58:09 PM                   1   Yes             Yes
4  06/28/2024 03:58:20 PM                   1   Yes             Yes

evening.head()

             Name (Original Name)       User Email               Join
Time  \
0                     Agrima Jain              NaN  06/28/2024
08:56:01 PM
1           AIML-19-SHOUNAK_SARKAR              NaN  06/28/2024
08:56:04 PM
2                  Akshita Roshan              NaN  06/28/2024
08:56:07 PM
3             Revanth Christober M              NaN  06/28/2024
08:56:08 PM
4  Krishna Singh (Akshita Roshan)  akshita@agie.ai  06/28/2024
08:56:10 PM

               Leave Time  Duration (Minutes) Guest  \
0  06/28/2024 08:59:15 PM                   4   Yes
1  06/28/2024 08:57:19 PM                   2   Yes
2  06/28/2024 08:57:20 PM                   2   Yes
3  06/28/2024 08:57:24 PM                   2   Yes
4  06/28/2024 09:19:37 PM                  24    No

  Recording Disclaimer Response In Waiting Room
0                  No Response              No
1                  No Response             Yes
2                  No Response             Yes
3                  No Response             Yes
4                           OK              No

#Get the columns present in the CSV files
morning_columns = morning_df.columns.tolist()
afternoon_columns = afternoon_df.columns.tolist()
evening_columns = evening_df.columns.tolist()

# Print the columns
print("Morning CSV Columns:", morning_columns)
print("Afternoon CSV Columns:", afternoon_columns)
print("Evening CSV Columns:", evening_columns)
```

```
Morning CSV Columns: ['Name (Original Name)', 'User Email', 'Join
Time', 'Leave Time', 'Duration (Minutes)', 'Guest', 'In Waiting Room']
Afternoon CSV Columns: ['Name (Original Name)', 'User Email', 'Join
Time', 'Leave Time', 'Duration (Minutes)', 'Guest', 'In Waiting Room']
Evening CSV Columns: ['Name (Original Name)', 'User Email', 'Join
Time', 'Leave Time', 'Duration (Minutes)', 'Guest', 'Recording
Disclaimer Response', 'In Waiting Room']

# Display the first few rows of each DataFrame
print("First few rows of Morning CSV:")
morning_df.head()

First few rows of Morning CSV:

             Name (Original Name)       User Email               Join
Time  \
0  Krishna Singh (Akshita Roshan)  akshita@agie.ai  06/28/2024
09:52:45 AM
1                    Chetan Kumar              NaN  06/28/2024
09:52:52 AM
2                  Akshita Roshan              NaN  06/28/2024
09:52:52 AM
3               Kshitij Tardalkar              NaN  06/28/2024
09:52:56 AM
4                    Chetan Kumar              NaN  06/28/2024
09:52:57 AM

               Leave Time  Duration (Minutes) Guest In Waiting Room
0  06/28/2024 10:17:06 AM                  25    No              No
1  06/28/2024 09:52:57 AM                   1   Yes             Yes
2  06/28/2024 09:53:08 AM                   1   Yes             Yes
3  06/28/2024 09:56:52 AM                   4   Yes             Yes
4  06/28/2024 10:16:12 AM                  24   Yes              No

print("First few rows of Afternoon CSV:")
afternoon_df.head()

First few rows of Afternoon CSV:

            Name (Original Name)       User Email               Join
Time  \
0  Krisha Singh (Akshita Roshan)  akshita@agie.ai  06/28/2024 03:57:40
PM
1                         Tanuja              NaN  06/28/2024 03:57:49
PM
2                      Yash Goel              NaN  06/28/2024 03:57:53
PM
3                        Darshan              NaN  06/28/2024 03:57:53
PM
4                    sneha pawar              NaN  06/28/2024 03:57:56
PM
```

```
               Leave Time  Duration (Minutes) Guest In Waiting Room
0  06/28/2024 04:19:20 PM                  22    No              No
1  06/28/2024 03:58:04 PM                   1   Yes             Yes
2  06/28/2024 03:58:09 PM                   1   Yes             Yes
3  06/28/2024 03:58:09 PM                   1   Yes             Yes
4  06/28/2024 03:58:20 PM                   1   Yes             Yes
```

```
print("First few rows of Evening CSV:")
evening_df.head()
```

```
First few rows of Evening CSV:
```

```
             Name (Original Name)       User Email               Join
Time  \
0                    Agrima Jain              NaN  06/28/2024
08:56:01 PM
1           AIML-19-SHOUNAK_SARKAR              NaN  06/28/2024
08:56:04 PM
2                 Akshita Roshan              NaN  06/28/2024
08:56:07 PM
3             Revanth Christober M              NaN  06/28/2024
08:56:08 PM
4  Krishna Singh (Akshita Roshan)  akshita@agie.ai  06/28/2024
08:56:10 PM

               Leave Time  Duration (Minutes) Guest  \
0  06/28/2024 08:59:15 PM                   4   Yes
1  06/28/2024 08:57:19 PM                   2   Yes
2  06/28/2024 08:57:20 PM                   2   Yes
3  06/28/2024 08:57:24 PM                   2   Yes
4  06/28/2024 09:19:37 PM                  24    No

  Recording Disclaimer Response In Waiting Room
0                   No Response              No
1                   No Response             Yes
2                   No Response             Yes
3                   No Response             Yes
4                            OK              No
```

```
print("Summary statistics for Morning CSV:")
morning_df.describe()
```

```
Summary statistics for Morning CSV:
```

```
       Duration (Minutes)
count          724.000000
mean            14.006906
std             13.083991
min              1.000000
25%              4.000000
```

```
50%                13.000000
75%                16.000000
max                97.000000

print("Summary statistics for Afternoon CSV:")
afternoon_df.describe()

Summary statistics for Afternoon CSV:

       Duration (Minutes)
count          221.000000
mean            21.642534
std             23.234649
min              0.000000
25%              5.000000
50%             13.000000
75%             24.000000
max             74.000000

print("Summary statistics for Evening CSV:")
evening_df.describe()

Summary statistics for Evening CSV:

       Duration (Minutes)
count          579.000000
mean            17.309154
std             17.728326
min              0.000000
25%              3.000000
50%             13.000000
75%             20.000000
max             85.000000

# Check for missing values
print("Missing values in Morning CSV:")
print(morning_df.isnull().sum(), "\n")

Missing values in Morning CSV:
Name (Original Name)      0
User Email              708
Join Time                 0
Leave Time                0
Duration (Minutes)        0
Guest                     0
In Waiting Room           0
dtype: int64


print("Missing values in Afternoon CSV:")
print(afternoon_df.isnull().sum(), "\n")
```

```
Missing values in Afternoon CSV:
Name (Original Name)      0
User Email              215
Join Time                 0
Leave Time                0
Duration (Minutes)        0
Guest                     0
In Waiting Room           0
dtype: int64


print("Missing values in Evening CSV:")
print(evening_df.isnull().sum(), "\n")

Missing values in Evening CSV:
Name (Original Name)                0
User Email                        567
Join Time                           0
Leave Time                          0
Duration (Minutes)                  0
Guest                               0
Recording Disclaimer Response       0
In Waiting Room                     0
dtype: int64


#Data Grouping and Summarization:

# Combine the dataframes
combined_df = pd.concat([morning_df, afternoon_df, evening_df])

# Group the data by 'Name (Original Name)' and calculate the total
duration for each participant
grouped_df = combined_df.groupby('Name (Original
Name)').agg({'Duration (Minutes)': 'sum'}).reset_index()

# Print the grouped data
print("Total duration for each participant:")
print(grouped_df)

Total duration for each participant:
                         Name (Original Name)  Duration (Minutes)
0                            '-Meet Vaghasiya-                  18
1                 00003121002_Sarath Rajendran                   4
2                                091101 10101                   1
3                      16010320032_Soham_Khadke                  98
4                                    17 SUMEN                   2
..                                        ...                 ...
366                                sneha pawar                  95
367                               sushain devi                  15
368  tera sribindhu (2005969 SRIKAR REDDY T.)                  44
```

```
369                                              v.karthikeya             2
370                                                    20          سيد سامي

[371 rows x 2 columns]

#analysis task

# Calculate the total duration of attendance for each batch
total_morning_duration = morning_df['Duration (Minutes)'].sum()
total_afternoon_duration = afternoon_df['Duration (Minutes)'].sum()
total_evening_duration = evening_df['Duration (Minutes)'].sum()
# Print the total durations
print("Total duration for Morning batch:", total_morning_duration,
"minutes")
print("Total duration for Afternoon batch:", total_afternoon_duration,
"minutes")
print("Total duration for Evening batch:", total_evening_duration,
"minutes")

Total duration for Morning batch: 10141 minutes
Total duration for Afternoon batch: 4783 minutes
Total duration for Evening batch: 10022 minutes

# Calculate the average duration of attendance for each batch
average_morning_duration = morning_df['Duration (Minutes)'].mean()
average_afternoon_duration = afternoon_df['Duration (Minutes)'].mean()
average_evening_duration = evening_df['Duration (Minutes)'].mean()
# Print the average durations
print("Average duration for Morning batch:", average_morning_duration,
"minutes")
print("Average duration for Afternoon batch:",
average_afternoon_duration, "minutes")
print("Average duration for Evening batch:", average_evening_duration,
"minutes")

Average duration for Morning batch: 14.006906077348066 minutes
Average duration for Afternoon batch: 21.642533936651585 minutes
Average duration for Evening batch: 17.30915371329879 minutes

# Calculate the standard deviation of attendance duration for each
batch
std_dev_morning_duration = morning_df['Duration (Minutes)'].std()
std_dev_afternoon_duration = afternoon_df['Duration (Minutes)'].std()
std_dev_evening_duration = evening_df['Duration (Minutes)'].std()

# Print the standard deviations
print("Standard deviation for Morning batch:",
std_dev_morning_duration, "minutes")
print("Standard deviation for Afternoon batch:",
std_dev_afternoon_duration, "minutes")
```

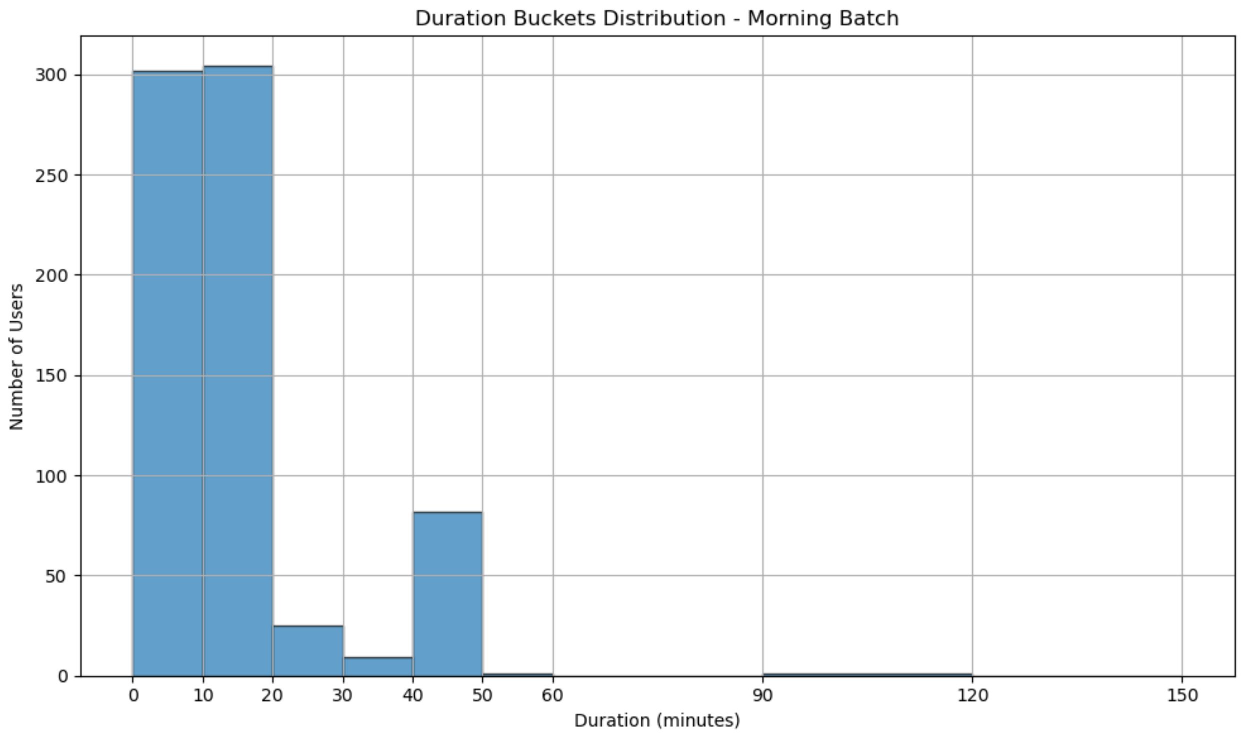
```
print("Standard deviation for Evening batch:",
std_dev_evening_duration, "minutes")
```

```
Standard deviation for Morning batch: 13.083991119280979 minutes
Standard deviation for Afternoon batch: 23.234648908765884 minutes
Standard deviation for Evening batch: 17.72832569594548 minutes
```

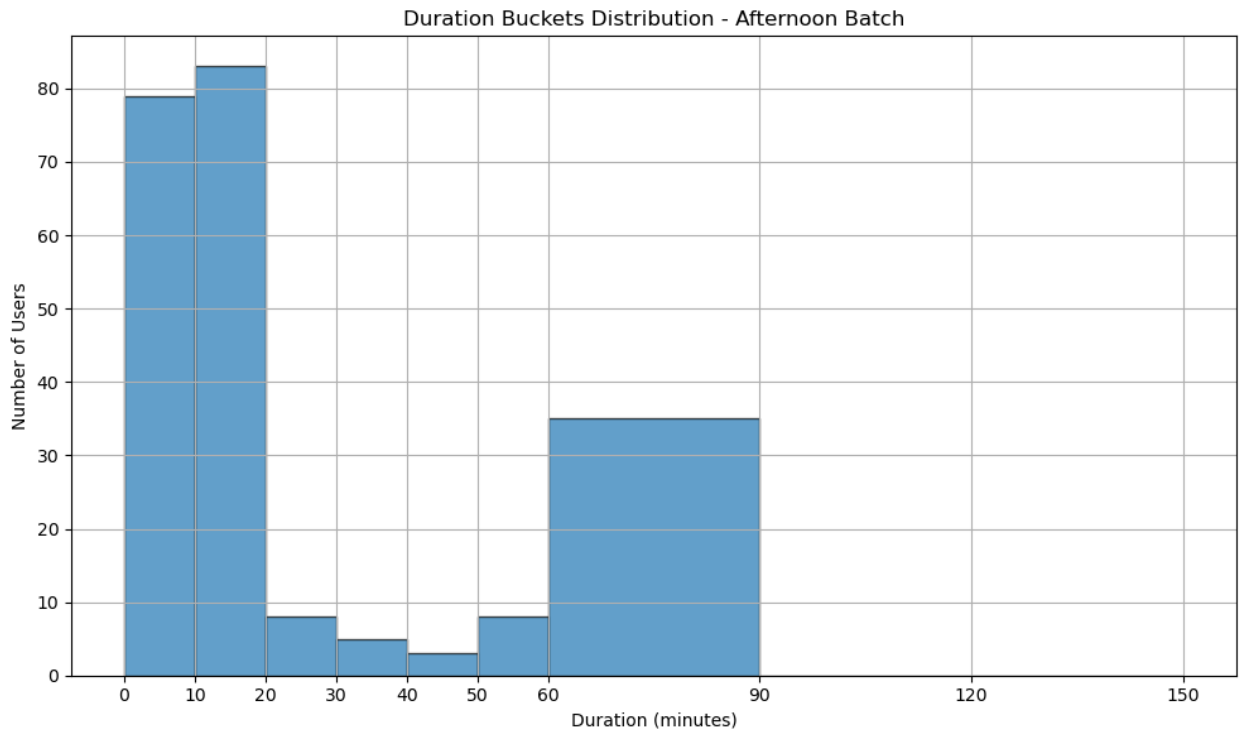
```
# Calculate the average duration for each batch
average_morning_duration = morning_df['Duration (Minutes)'].mean()
average_afternoon_duration = afternoon_df['Duration (Minutes)'].mean()
average_evening_duration = evening_df['Duration (Minutes)'].mean()

# Calculate the percentage of users who attended more than the average
duration for each batch
morning_above_average = (morning_df['Duration (Minutes)'] >
average_morning_duration).mean() * 100
afternoon_above_average = (afternoon_df['Duration (Minutes)'] >
average_afternoon_duration).mean() * 100
evening_above_average = (evening_df['Duration (Minutes)'] >
average_evening_duration).mean() * 100

# Print the percentages
print(f"Percentage of Morning attendees above average duration:
{morning_above_average:.2f}%")
print(f"Percentage of Afternoon attendees above average duration:
{afternoon_above_average:.2f}%")
print(f"Percentage of Evening attendees above average duration:
{evening_above_average:.2f}%")
```

```
Percentage of Morning attendees above average duration: 37.43%
Percentage of Afternoon attendees above average duration: 25.79%
Percentage of Evening attendees above average duration: 29.53%
```

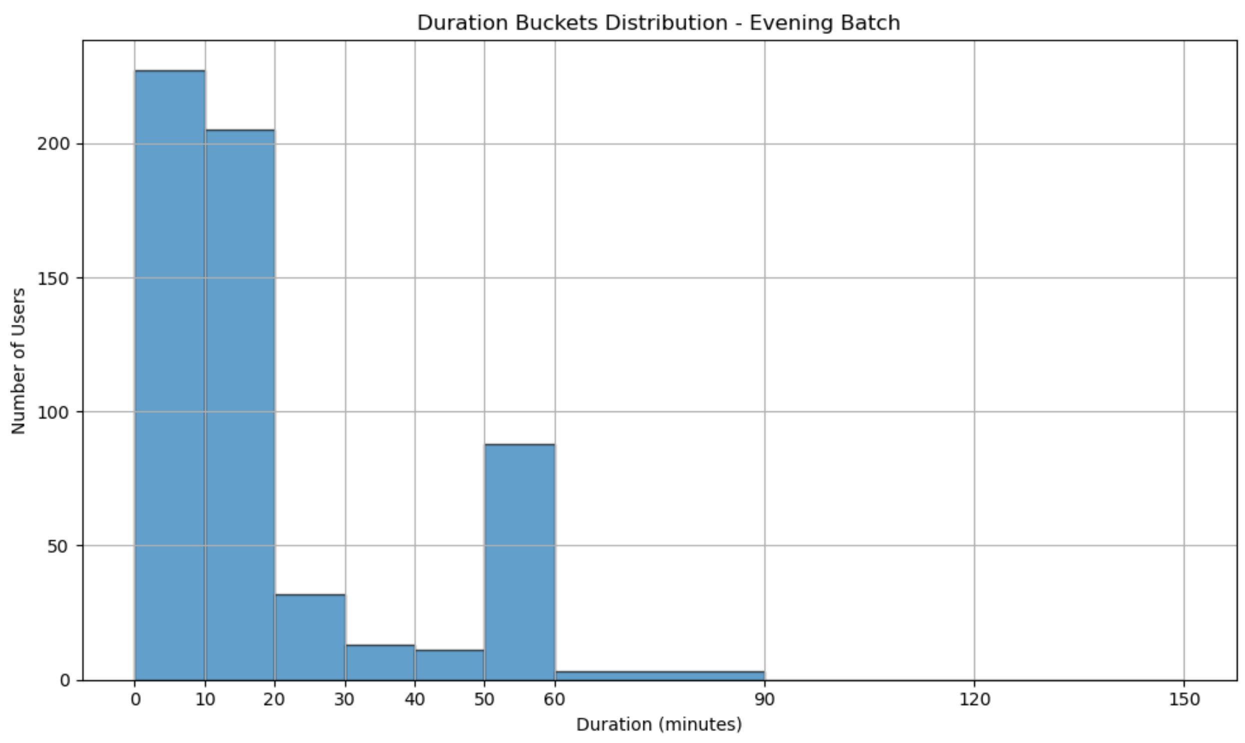
```
#Duration Buckets Distribution: Plot a histogram showing the number of
users in different duration buckets (e.g., 0-10 mins, 10-20 mins,
etc.) for each batch.

# Define the duration buckets
duration_bins = [0, 10, 20, 30, 40, 50, 60, 90, 120, 150]

# Plot histogram for Morning batch
plt.figure(figsize=(10, 6))
plt.hist(morning_df['Duration (Minutes)'], bins=duration_bins,
edgecolor='black', alpha=0.7)
plt.title('Duration Buckets Distribution - Morning Batch')
plt.xlabel('Duration (minutes)')
plt.ylabel('Number of Users')
plt.xticks(duration_bins)
plt.grid(True)
plt.tight_layout()
plt.show()
```



```
# Plot histogram for Afternoon batch
plt.figure(figsize=(10, 6))
plt.hist(afternoon_df['Duration (Minutes)'], bins=duration_bins,
edgecolor='black', alpha=0.7)
plt.title('Duration Buckets Distribution - Afternoon Batch')
plt.xlabel('Duration (minutes)')
plt.ylabel('Number of Users')
plt.xticks(duration_bins)
plt.grid(True)
plt.tight_layout()
plt.show()
```



```python
# Plot histogram for Evening batch
plt.figure(figsize=(10, 6))
plt.hist(evening_df['Duration (Minutes)'], bins=duration_bins,
edgecolor='black', alpha=0.7)
plt.title('Duration Buckets Distribution - Evening Batch')
plt.xlabel('Duration (minutes)')
plt.ylabel('Number of Users')
plt.xticks(duration_bins)
plt.grid(True)
plt.tight_layout()
plt.show()
```



```python
#Comparative Analysis: Compare the total duration, average duration,
standard deviation, and percentage of users attending more than the
average duration across all three batches.

# Calculate total duration for each batch
total_morning_duration = morning_df['Duration (Minutes)'].sum()
total_afternoon_duration = afternoon_df['Duration (Minutes)'].sum()
total_evening_duration = evening_df['Duration (Minutes)'].sum()

# Calculate average duration for each batch
average_morning_duration = morning_df['Duration (Minutes)'].mean()
average_afternoon_duration = afternoon_df['Duration (Minutes)'].mean()
average_evening_duration = evening_df['Duration (Minutes)'].mean()

# Calculate standard deviation for each batch
std_dev_morning_duration = morning_df['Duration (Minutes)'].std()
std_dev_afternoon_duration = afternoon_df['Duration (Minutes)'].std()
std_dev_evening_duration = evening_df['Duration (Minutes)'].std()

# Calculate percentage of users attending more than the average
duration for each batch
morning_above_average = (morning_df['Duration (Minutes)'] >
average_morning_duration).mean() * 100
afternoon_above_average = (afternoon_df['Duration (Minutes)'] >
average_afternoon_duration).mean() * 100
evening_above_average = (evening_df['Duration (Minutes)'] >
average_evening_duration).mean() * 100
```

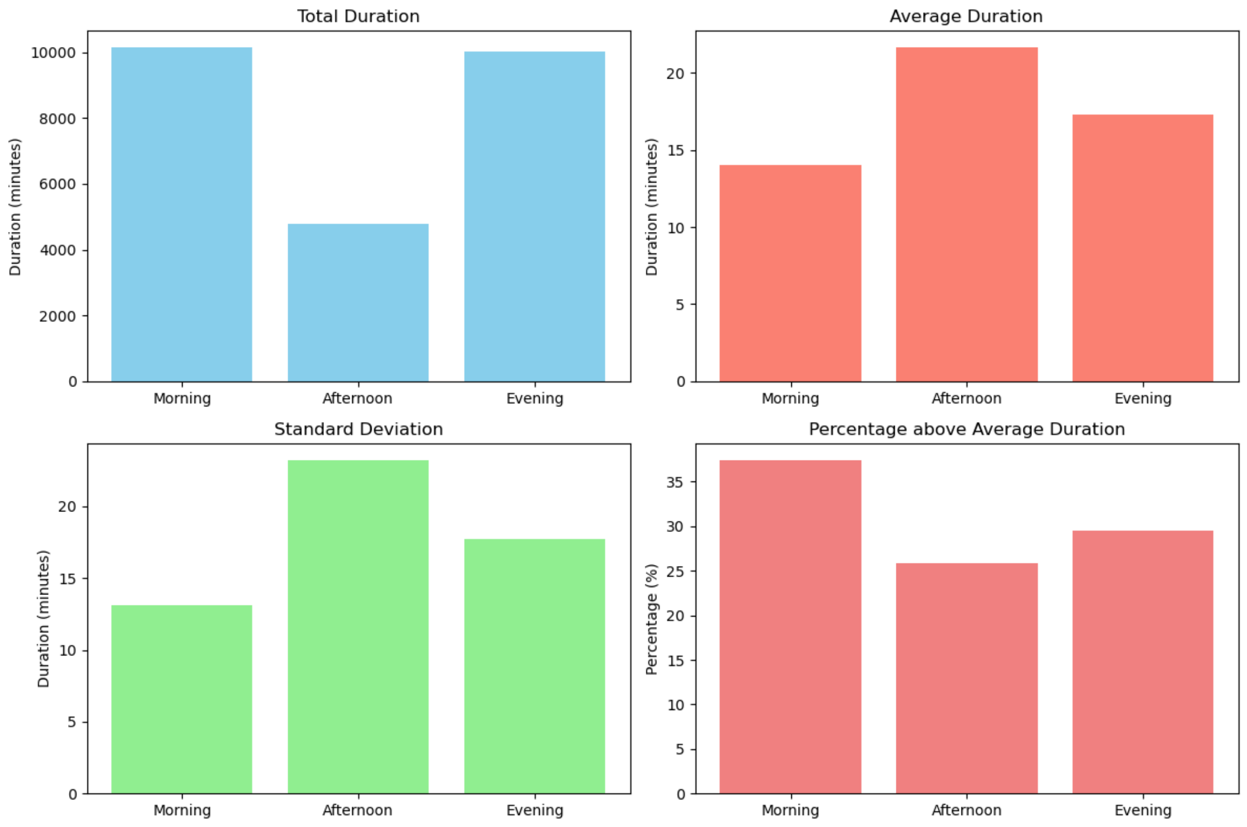
```
# Print the comparative analysis results
print("Comparative Analysis for Morning Batch:")
print(f"Total Duration: {total_morning_duration} minutes")
print(f"Average Duration: {average_morning_duration:.2f} minutes")
print(f"Standard Deviation: {std_dev_morning_duration:.2f} minutes")
print(f"Percentage above Average Duration: {morning_above_average:.2f}
%\n")

Comparative Analysis for Morning Batch:
Total Duration: 10141 minutes
Average Duration: 14.01 minutes
Standard Deviation: 13.08 minutes
Percentage above Average Duration: 37.43%


print("Comparative Analysis for Afternoon Batch:")
print(f"Total Duration: {total_afternoon_duration} minutes")
print(f"Average Duration: {average_afternoon_duration:.2f} minutes")
print(f"Standard Deviation: {std_dev_afternoon_duration:.2f} minutes")
print(f"Percentage above Average Duration:
{afternoon_above_average:.2f}%\n")

Comparative Analysis for Afternoon Batch:
Total Duration: 4783 minutes
Average Duration: 21.64 minutes
Standard Deviation: 23.23 minutes
Percentage above Average Duration: 25.79%


print("Comparative Analysis for Evening Batch:")
print(f"Total Duration: {total_evening_duration} minutes")
print(f"Average Duration: {average_evening_duration:.2f} minutes")
print(f"Standard Deviation: {std_dev_evening_duration:.2f} minutes")
print(f"Percentage above Average Duration: {evening_above_average:.2f}
%")

Comparative Analysis for Evening Batch:
Total Duration: 10022 minutes
Average Duration: 17.31 minutes
Standard Deviation: 17.73 minutes
Percentage above Average Duration: 29.53%

#Report Genaration

# Import necessary libraries
import pandas as pd

# Function to generate detailed summary report for a batch
def generate_summary_report(df, batch_name):
    # Calculate total duration
    total_duration = df['Duration (Minutes)'].sum()
    # Calculate average duration
    average_duration = df['Duration (Minutes)'].mean()

    # Calculate standard deviation
    std_dev_duration = df['Duration (Minutes)'].std()

    # Calculate percentage of users attending more than average
duration
    above_average_percentage = (df['Duration (Minutes)'] >
average_duration).mean() * 100

    # Generate and return detailed summary report
    summary_report = f"Summary Report for {batch_name} Batch:\n"
    summary_report += f"Total Duration: {total_duration} minutes\n"
    summary_report += f"Average Duration: {average_duration:.2f}
minutes\n"
    summary_report += f"Standard Deviation: {std_dev_duration:.2f}
minutes\n"
    summary_report += f"Percentage above Average Duration:
{above_average_percentage:.2f}%\n"

    return summary_report

# Load the CSV files
morning_df=pd.read_csv(r"C:\Users\Saikiran\Downloads\Morning-28th-
June.csv")
afternoon_df=pd.read_csv(r"C:\Users\Saikiran\Downloads\Afternoon-28th-
June.csv")
evening_df=pd.read_csv(r"C:\Users\Saikiran\Downloads\Evening-28th-
June.csv")

# Generate detailed summary reports for each batch
morning_report = generate_summary_report(morning_df, "Morning")
afternoon_report = generate_summary_report(afternoon_df, "Afternoon")
evening_report = generate_summary_report(evening_df, "Evening")

# Print the detailed summary reports
print(morning_report)
print(afternoon_report)
print(evening_report)
```

```
Summary Report for Morning Batch:
Total Duration: 10141 minutes
Average Duration: 14.01 minutes
Standard Deviation: 13.08 minutes
Percentage above Average Duration: 37.43%

Summary Report for Afternoon Batch:
Total Duration: 4783 minutes
```

```
Average Duration: 21.64 minutes
Standard Deviation: 23.23 minutes
Percentage above Average Duration: 25.79%

Summary Report for Evening Batch:
Total Duration: 10022 minutes
Average Duration: 17.31 minutes
Standard Deviation: 17.73 minutes
Percentage above Average Duration: 29.53%
```

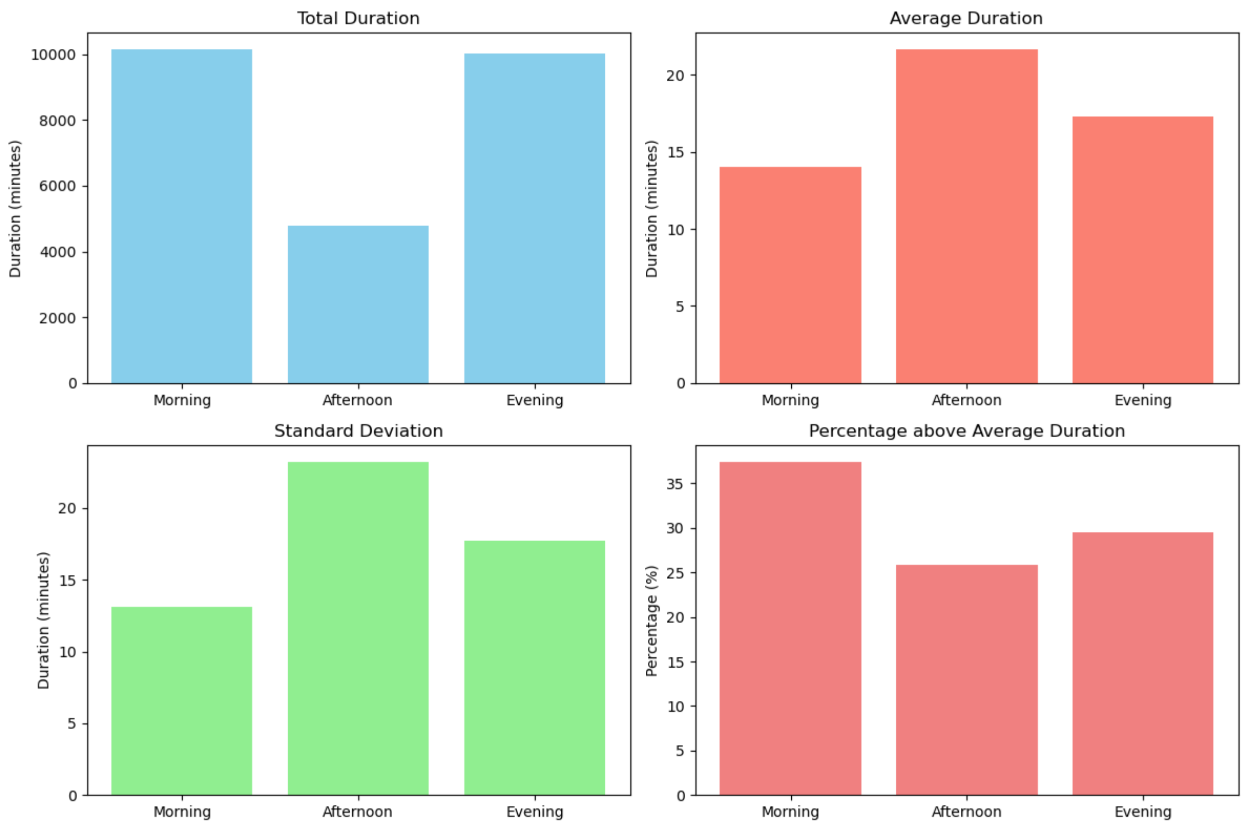
```
import matplotlib.pyplot as plt
import pandas as pd

# Function to generate detailed summary report for a batch
def generate_summary_report(df, batch_name):
    # Calculate total duration
    total_duration = df['Duration (Minutes)'].sum()

    # Calculate average duration
    average_duration = df['Duration (Minutes)'].mean()

    # Calculate standard deviation
    std_dev_duration = df['Duration (Minutes)'].std()

    # Calculate percentage of users attending more than average
duration
    above_average_percentage = (df['Duration (Minutes)'] >
average_duration).mean() * 100

    # Generate and return detailed summary report
    summary_report = {
        'Batch': batch_name,
        'Total Duration': total_duration,
        'Average Duration': average_duration,
        'Standard Deviation': std_dev_duration,
        'Percentage above Average Duration': above_average_percentage
    }

    return summary_report


# Generate detailed summary reports for each batch
morning_report = generate_summary_report(morning_df, "Morning")
afternoon_report = generate_summary_report(afternoon_df, "Afternoon")
evening_report = generate_summary_report(evening_df, "Evening")

# Create lists to store summary reports
summary_reports = [morning_report, afternoon_report, evening_report]
```

```
# Extract metrics for plotting
batches = [report['Batch'] for report in summary_reports]
total_durations = [report['Total Duration'] for report in
summary_reports]
average_durations = [report['Average Duration'] for report in
summary_reports]
std_deviations = [report['Standard Deviation'] for report in
summary_reports]
percentages_above_average = [report['Percentage above Average
Duration'] for report in summary_reports]

# Plotting
plt.figure(figsize=(12, 8))

# Plot 1: Bar plot for Total Duration
plt.subplot(221)
plt.bar(batches, total_durations, color='skyblue')
plt.title('Total Duration')
plt.ylabel('Duration (minutes)')

# Plot 2: Bar plot for Average Duration
plt.subplot(222)
plt.bar(batches, average_durations, color='salmon')
plt.title('Average Duration')
plt.ylabel('Duration (minutes)')

# Plot 3: Bar plot for Standard Deviation
plt.subplot(223)
plt.bar(batches, std_deviations, color='lightgreen')
plt.title('Standard Deviation')
plt.ylabel('Duration (minutes)')

# Plot 4: Bar plot for Percentage above Average Duration
plt.subplot(224)
plt.bar(batches, percentages_above_average, color='lightcoral')
plt.title('Percentage above Average Duration')
plt.ylabel('Percentage (%)')

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



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

# Function to generate detailed summary report for a batch
def generate_summary_report(df, batch_name):
    # Calculate total duration
    total_duration = df['Duration (Minutes)'].sum()

    # Calculate average duration
    average_duration = df['Duration (Minutes)'].mean()

    # Calculate standard deviation
    std_dev_duration = df['Duration (Minutes)'].std()

    # Calculate percentage of users attending more than average
duration
    above_average_percentage = (df['Duration (Minutes)'] >
average_duration).mean() * 100

    # Generate and return detailed summary report
    summary_report = {
        'Batch': batch_name,
        'Total Duration': total_duration,
        'Average Duration': average_duration,
        'Standard Deviation': std_dev_duration,
        'Percentage above Average Duration': above_average_percentage
    }

    return summary_report


# Generate detailed summary reports for each batch
morning_report = generate_summary_report(morning_df, "Morning")
afternoon_report = generate_summary_report(afternoon_df, "Afternoon")
evening_report = generate_summary_report(evening_df, "Evening")

# Create lists to store summary reports
summary_reports = [morning_report, afternoon_report, evening_report]
print(summary_reports)

# Extract metrics for plotting
batches = [report['Batch'] for report in summary_reports]
total_durations = [report['Total Duration'] for report in
summary_reports]
average_durations = [report['Average Duration'] for report in
summary_reports]
std_deviations = [report['Standard Deviation'] for report in
summary_reports]
percentages_above_average = [report['Percentage above Average
Duration'] for report in summary_reports]

# Plotting
plt.figure(figsize=(12, 8))

# Plot 1: Bar plot for Total Duration
plt.subplot(221)
plt.bar(batches, total_durations, color='skyblue')
plt.title('Total Duration')
plt.ylabel('Duration (minutes)')

# Plot 2: Bar plot for Average Duration
plt.subplot(222)
plt.bar(batches, average_durations, color='salmon')
plt.title('Average Duration')
plt.ylabel('Duration (minutes)')

# Plot 3: Bar plot for Standard Deviation
plt.subplot(223)
plt.bar(batches, std_deviations, color='lightgreen')
plt.title('Standard Deviation')
plt.ylabel('Duration (minutes)')

# Plot 4: Bar plot for Percentage above Average Duration
plt.subplot(224)
plt.bar(batches, percentages_above_average, color='lightcoral')
plt.title('Percentage above Average Duration')
plt.ylabel('Percentage (%)')

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

```
[{'Batch': 'Morning', 'Total Duration': 10141, 'Average Duration':
14.006906077348066, 'Standard Deviation': 13.083991119280979,
'Percentage above Average Duration': 37.430939226519335}, {'Batch':
'Afternoon', 'Total Duration': 4783, 'Average Duration':
21.642533936651585, 'Standard Deviation': 23.234648908765884,
'Percentage above Average Duration': 25.791855203619914}, {'Batch':
'Evening', 'Total Duration': 10022, 'Average Duration':
17.30915371329879, 'Standard Deviation': 17.72832569594548,
'Percentage above Average Duration': 29.533678756476682}]
```


Total Duration
Duration (minutes)
Morning
Afternoon
Evening
Average Duration
Duration (minutes)
Morning
Afternoon
Evening
Standard Deviation
Duration (minutes)
Morning
Afternoon
Evening
Percentage above Average Duration
Percentage (%)
Morning
Afternoon
Evening