Project Scope and Plan: LifeLens - Sleep Score Analysis - Final Iteration

Muhaiminul Ashraf

December 4, 2024

1 Project Scope

The aim of this project is to design and implement a health recommendation system that analyzes user sleep data to generate actionable insights and personalized recommendations for improving overall health. By leverage graph-based modeling, the system provides a detailed understanding of sleep patterns, deviations from optimal benchmarks, and actionable recommendations

The project's main objectives are:

- Model user sleep cycles using a graph-based approach to analyze temporal patterns and relationships between consecutive sleep cycles.
- Calculate and compare sleep performance metrics (light, deep, and REM sleep) against scientifically established benchmarks.
- Generated detailed visualizations to compare actual and optimal sleep durations, highlighting areas of improvement.
- Lay the groundwork for personalized health recommendations based on analyzed sleep patterns and overall sleep performance.

The expected outcomes include:

- A functional back-end system for sleep performance analysis and visualization
- Detailed visualizations showcasing deviations from optimal sleep benchmarks.
- Comprehensive documentation and a final project report summarizing findings and insights.

2 Project Plan

2.1 Timeline

The overall timeline for the project is divided into phases:

- **Week 1 (October 7 - October 13)**: Define the scope of the project, establish team roles, and outline skills/tools.
- **Week 2 (October 14 - October 20)**: Begin development, setup the project repository, and start coding basic system functionalities.
- **Week 3 (October 21 - October 27)**: Continue coding, work on back-end integration, and start writing technical documentation.
- **Week 4 (October 28 - November 3)**: Complete the back-end and integrate front-end elements. Begin the PowerPoint presentation.
- **Week 5 (November 4 - November 10)**: Finalize the system, conduct testing, and continue with the report.

- **Week 6 (November 11 - November 17)**: Revise and finalize the technical report and the PowerPoint presentation.
- **Week 7 (November 18 - November 28)**: Final presentation, report submission, and project closure.

2.2 Milestones

Key milestones include:

- Project Scope and Plan (October 7).
- GitHub Repository Setup and Initial Development (October 9).
- Backend Completion (November 3).
- Graph Integration and Visualization (November 10).
- Final System Testing and Report Draft (November 17).
- Final Presentation and Report Submission (November 28).

2.3 Roles

Muhaiminul: Responsible for frontend design of the health recommendation system, graph system design to connect various nodes, and proper algorithm design to rate sleep performance against WHOOP's performance rating as well as average sleep phase performance.

3 Progress Review

3.1 Achievements

During this iteration, significant progress was made in the backend implementation. The key achievements include:

- Development of a graph-based system in `SleepGraph.py`, where nodes represent individual sleep cycles, storing metrics such as light sleep, deep sleep, REM sleep, total sleep, and sleep performance ratings.
- Introduction of a function to compute overall sleep performance based on the ratio of total sleep to sleep need.
- Implementation of functions to compare actual sleep durations for light, deep, and REM phases against optimal benchmarks and calculate differences and performances.
- Creation of the `sleepPlotRender.py` script to generate detailed visualizations of sleep metrics over time, including:
  - Line plots for actual vs. optimal sleep durations.
  - Bar charts for differences between actual and optimal sleep durations.

3.2 Key Backend Features

- Graph-Based Data Representation: Sleep cycles are modeled as a graph with nodes storing detailed attributes and edges connecting consecutive cycles.
- Performance metrics: Functions to evaluate sleep performance and compare actual vs. optimal values for light, deep, and REM sleep.
- Visualization: Visual tools for understanding trends and deviations in sleep performance across multiple cycles.

3.3 Challenges and Solutions

- Handling Missing Data: Ensured that missing values in the dataset are handled gracefully, avoiding disruptions in calculations or visualizations.
- Graph Integration: Designed a flexible graph structure to support both node-level and graph-level analyses, enabling temporal insights into user sleep patterns.

4 Methodology

4.1 Logic Layout

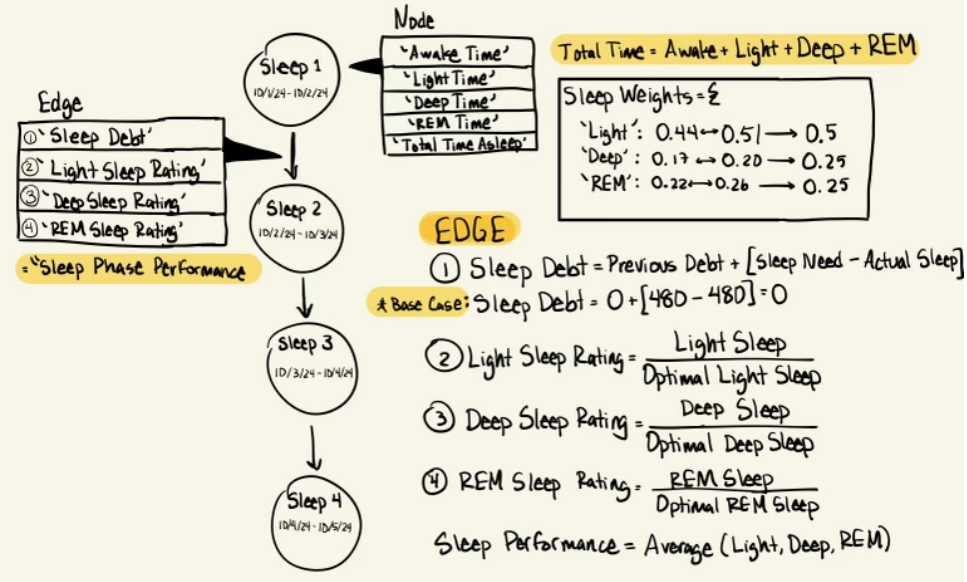

Figure 1: Sleep Nodes and Weights per Sleep Mode

Each sleep node encompasses a nightly sleep cycle. In that node, there are the properties 'Awake Time', 'Light Time', 'Deep Time', 'REM Time', and 'Total Time Asleep'. These values are given to us in the sleepData.csv file. Each edge connecting the different nodes contains the properties 'Sleep Debt', 'Awake Sleep Rating', 'Light Sleep Rating', 'Deep Sleep Rating', and 'REM Sleep Rating'. The ratings are calculated based on the data from the csv file divided by the optimal time spent in each phase. The optimal time spent in each phase is determined by multiplying the total time spent asleep by the sleep weights (defined in the box 'Sleep Weights'). Based on all of these ratings, a performance for the night is calculated. Each of these ratings are weighted equally in calculating the final aggregate sleep phase performance score. Sleep Debt or Sleep Need is the difference between each night's sleep amount and the baseline 8 hours needed for most people. These are carried between nodes to signify the accumulated sleep debt.

4.2 Results

```
Cycle 0:
WHOOP Performance Rating: 60
Sleep Performance: 62.88
Light Sleep: {'Actual': 182, 'Expected': 186.0, 'Difference': -4.0, 'Performance': 97.85}
Deep Sleep: {'Actual': 105, 'Expected': 93.0, 'Difference': 12.0, 'Performance': 100}
REM Sleep: {'Actual': 85, 'Expected': 93.0, 'Difference': -8.0, 'Performance': 91.4}
Average Sleep Phase Performance: {'Average Performance': 96.0}
```

Figure 2: Calculation of Sleep Performance Score vs. WHOOP Performance Score; Sleep Phase Performance.

We first approximated the sleep performance based off the total sleep divided by the sleep need. This gave us approximations that were relatively similar to WHOOP's performance score, thus we can say with some confidence these are the metrics WHOOP uses to calculate sleep scores.

In each phase we calculate the sleep ratings based on the difference between Expected and Actual sleep ratings. If the difference is negative, that means the user got less than optimal, so we create a performance score based on actual divided by expected. Otherwise, if the difference is positive, we can assume the user got more than was expected. We then average the three of these performance metrics to come up with an aggregate rating.

```
Ranking by Overall Sleep Performance:
Rank 1: Node 15 with Overall Sleep Performance: 92.53
Rank 2: Node 14 with Overall Sleep Performance: 92.44
Rank 3: Node 11 with Overall Sleep Performance: 84.41
Rank 4: Node 8 with Overall Sleep Performance: 83.68
Rank 5: Node 5 with Overall Sleep Performance: 80.92
Rank 6: Node 12 with Overall Sleep Performance: 75.64
Rank 7: Node 17 with Overall Sleep Performance: 74.35
Rank 8: Node 6 with Overall Sleep Performance: 72.76
Rank 9: Node 3 with Overall Sleep Performance: 71.78
Rank 10: Node 7 with Overall Sleep Performance: 71.15
Rank 11: Node 16 with Overall Sleep Performance: 71.02
Rank 12: Node 1 with Overall Sleep Performance: 70.8
Rank 13: Node 9 with Overall Sleep Performance: 65.71
Rank 14: Node 0 with Overall Sleep Performance: 62.88
Rank 15: Node 2 with Overall Sleep Performance: 61.55
Rank 16: Node 13 with Overall Sleep Performance: 56.45
Rank 17: Node 10 with Overall Sleep Performance: 54.73
Rank 18: Node 4 with Overall Sleep Performance: 41.96
```

Figure 3: Bubblesort Ranking of Nodes by Sleep Performance

Figure 3 uses bubble-sort to rank nodes by the sleep performance calculated in Figure 2. This gives insight into how days with poor sleep performance must be followed by days with good sleep performance. Nodes 14 and 15 sit at the top of the rankings while Node 10 sits near the bottom of the ranking system.

5 Visualizations and Insights

5.1 Sample Outputs

Below are the visualizations generated from `sleepPlotRender.py`:

- **Light Sleep Analysis**:
  - Actual vs. Optimal Light Sleep durations.
  - Differences displayed as bar charts.
- **Deep Sleep Analysis**:
  - Actual vs. Optimal Deep Sleep durations.
  - Differences displayed as bar charts.
- **REM Sleep Analysis**:
  - Actual vs. Optimal REM Sleep durations.
  - Differences displayed as bar charts.

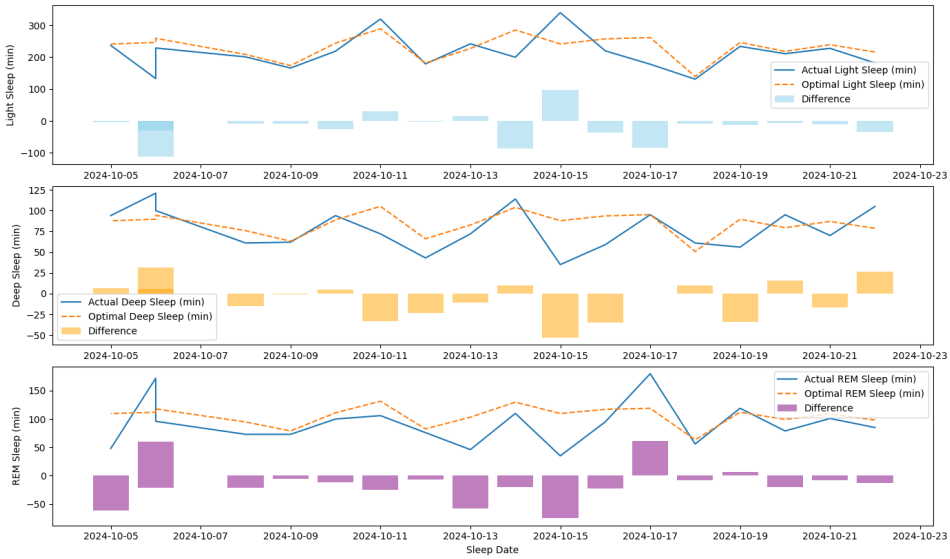

Figure 4: Sleep Performance Metrics as measured by the difference between optimal sleep and measured sleep.

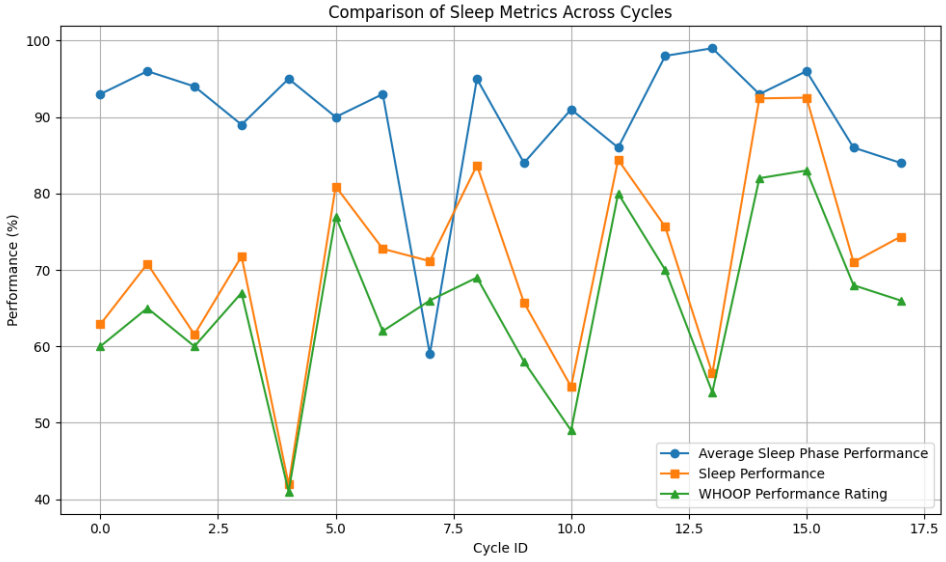

Figure 5: Sleep Performance Metrics as measured by the difference between Whoop's performance rating and the average sleep performance and calculated sleep performance.

- Performance metrics align with WHOOP scores, validating the methodology.
- Visual tools provide actionable insights into sleep patterns.

6 Future Work

- Testing with simulated data to verify the accuracy of the recommendations.
- Refinement of the user interface for seamless data interpretation.

7 Pseudocode for Sleep Analysis

7.1 Graph Construction

```
1. Initialize an empty graph:
    Graph: Nodes = {}, Edges = {}

2. For each row in sleep data:
    a. Extract attributes: Wake, Light Sleep, Deep Sleep, REM Sleep,
       Total Sleep, Sleep Need, Sleep Debt, WHOOP Performance.
    b. Create a Node with these attributes and a unique cycle ID.
    c. Add the Node to the graph.

3. Connect consecutive sleep cycles by adding edges between Nodes.
```

7.2 Calculate Sleep Performance

```
1. For each Node in the graph:
    a. Calculate Total Sleep as the sum of Wake, Light, Deep, and REM.
    b. Retrieve Sleep Need (default to 480 minutes if unavailable).
    c. Compute Overall Sleep Performance as:
        Min(Total Sleep / Sleep Need, 1.0) * 100.
    d. Store the Overall Sleep Performance in the Node.
```

7.3 Phase Comparison

```
1. Define phase weights:
    Light Sleep = 0.50, Deep Sleep = 0.25, REM Sleep = 0.25.

2. For each Node in the graph:
    a. Compute expected durations for Light, Deep, and REM phases:
        Expected Phase = Weight * Total Sleep.
    b. Calculate differences between actual and expected durations.
    c. Compute phase performance:
        If Actual < Expected:
            Performance = (Actual / Expected) * 100.
        Else:
            Performance = 100.
    d. Calculate Average Sleep Phase Performance as the mean of phase performances.
    e. Store phase performances and the average in the Node.
```

7.4 Rank Nodes by Performance

```
1. Initialize an empty list for ranked Nodes.

2. Collect Nodes with valid Overall Sleep Performance values.

3. Use Bubble Sort to rank Nodes:
    a. Compare adjacent Nodes based on performance.
    b. Swap positions if out of order (descending or ascending as needed).

4. Assign ranks to Nodes and store results.
```

7.5 Visualization of Sleep Metrics

```
1. Map each Node ID to corresponding Sleep Date.

2. Create a data structure with columns:
    - Sleep Date
```

```
    - Average Sleep Phase Performance
    - Sleep Performance
    - WHOOP Performance Rating.

3. Populate the data structure using Node attributes.

4. Plot data:
    a. Use Sleep Date as the x-axis.
    b. Plot Average Sleep Phase Performance, Sleep Performance,
       and WHOOP Performance Rating as lines with markers.
    c. Add labels, title, legend, and grid for clarity.
```

8 Appendix

8.1 Code Listings

- `SleepGraph.py`: The script that implements the graph-based system for analyzing sleep metrics.
- `sleepPlotRender.py`: The visualization script to generate plots of sleep performance metrics.