Somno Health

Luke Valentine, Michael Ruiz, Jonathon Kastner, Samson Mammarappallil, Thomas Kuzis

6/8/20

Table of Contents

Introduction

Somno Health has brought over 25 FDA approved medical devices to the market. Their latest device, Eversleep, can collect clinical quality sleep data while allowing the user to stay in the comfort of their home. It is the only at-home sleep monitoring device that collects blood oxygen saturation data. It also tracks motion, snoring, pulse rate, and through a mobile app provides coaching tips.

Somno Health is trying to update a Corporate Wellness Portal for their sleep data. This would provide a space for physicians and businesses to evaluate patient/employee data.

Our task was to first decide between two different data portals, created by two separate CSM field session groups in Summer 2018 and Summer 2019. The portal from 2018 was created using JavaScript, and has been used by Somno Health since its development. The 2019 portal was created using Tornado, a Python based web framework. However, the 2019 portal was lacking some of the key functionality that the 2018 portal contained. Once we picked a data portal to move forward with, we had to add functionality to the existing portal. This required us to set up a development environment and then refactor and extend code to match Somno Health's vision.

Requirements

High-level Description/Vision

This project involved building an improved Corporate Wellness Portal for SomnoHealth Incorporated.

Functional Requirements

Decide between two existing portals. The new portal developed in 2019 was missing functionality for certain operations that were present in the older 2018 portal. Part of transitioning to the new portal needed to include translating this missing functionality and thoroughly testing for errors.

A list of functional requirements for this project is as follows:

- Current and accurate data should be shown on the active portal
    - Data entries should populate the portal's Reports/Results sections
    - Graphs for data entries should both generate and function correctly in the portal's Trends section. These graphs include bar charts for the quality sleep data and line charts to display measurement spikes.
- The portal automatically updates data on a regular basis
- Users should be able to search for specific data entries

  - This includes incorporating filters for data tracking seen in the 2018 portal
  - New filters, specifically the serial number corresponding to the user's tracker
- The active portal should have PDF generation for one or more recorded data entries, as seen in the 2018 portal
- The active portal should have coaching information for each nightly entry
- The active portal should have group functionality, where a user can monitor a subset of individuals and their data entries (a nurse monitoring groups of patients).
- Portal needs to have working and correctly tested "Delete" functionality
  - Allows the user to delete one or more nights of data on a specific device
  - This is currently implemented but at most has only seen light testing (this function's behavior is currently unknown)

Other wants / Stretch Goals:

- CSV output functionality for nightly data entries
- Automatically generated emails (sent once per week to physicians to review patient data)

Non-Functional Requirements

1. Sort, PDF, output, and search features must all be implemented with the Tornado web application framework (Python) if using the new portal.
2. Sort, PDF output, and search features must all be implemented in JavaScript if using the 2018 portal.
3. The portal must function efficiently when having 30,000 days of data.
4. Must make sure that no data is ever lost when implementing features.
5. Final product will have to be more aesthetically similar to the new 2019 portal.

System Architecture

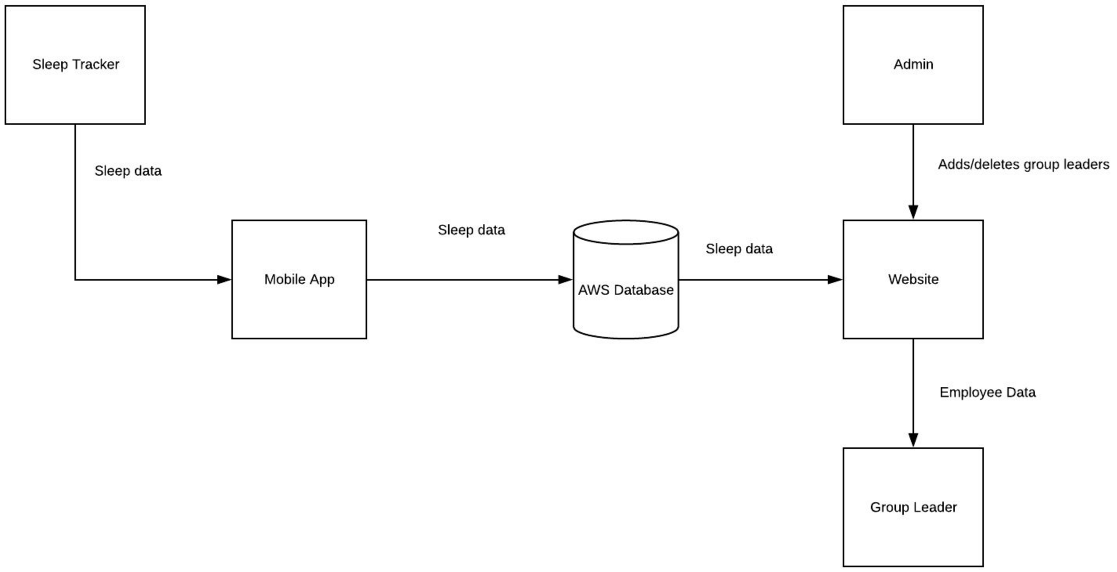
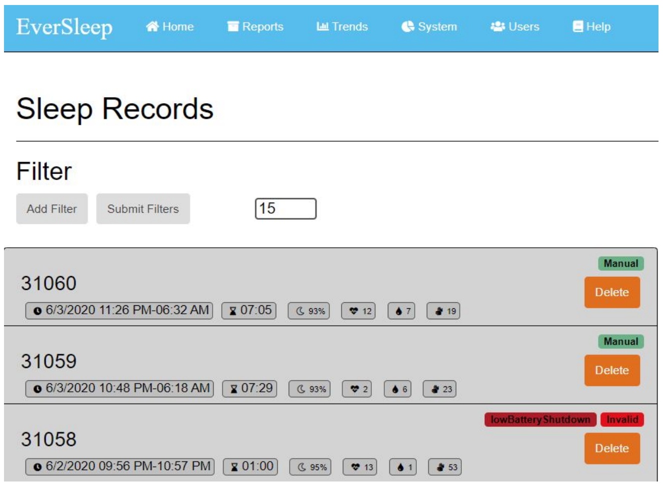


Figure 1: Architecture Diagram

Somnohealth's EverSleep trackers work by interacting with a user's smartphone, which then passes their nightly sleep data on to an AWS database. This database is then accessed by the Eversleep web portal. On this portal the admins should have the ability to create groups of users, and then assign a group leader who will be able to see all of their groups' user data. This general architecture is shown in Figure 1 above. For the project we were working on this interaction between an admin, the web portal, and the group leaders. Figure 2 below depicts the records screen in which a group leader or admin is capable of expanding individual nights of data to view everything from physical fitness to sleep quality of a patient.



Sleep Records

Filter

15

31060 — Manual
6/3/2020 11:26 PM-06:32 AM | 07:05 | 93% | 12 | 7 | 19

31059 — Manual
6/3/2020 10:48 PM-06:18 AM | 07:29 | 93% | 2 | 6 | 23

31058 — lowBatteryShutdown Invalid
6/2/2020 09:56 PM-10:57 PM | 01:00 | 95% | 13 | 1 | 53

Figure 2

Technical Design

Patient Group Functionality

One of the most interesting elements of our final design was the addition of group functionality into the portal. This enables an admin user to designate either a new or existing user as a group leader by entering the name of the group in the "Group Name" field. If the group already exists in the database, the user is added to the existing group. Otherwise, a new group is created with the specified name and the user is placed into the group as the first member. Using this, medical professionals such as nurses and physicians can be added to the portal as group leaders, and have groups for their patients. Those group leaders can then add individual serial numbers to their group as they prescribe EverSleep trackers to their patients. When viewing the Records section of the portal, the group leader would see all of the data for only the users in their group (or only the data coming from their added serial numbers). This works for private use as well, because an individual can request to become a "group leader" where they are the only user in their group. Therefore, they can view and download their sleep data for external use in CSV, JSON, and PDF format. An example of the updated "Add User" process with groups is shown in Figure 3 below:

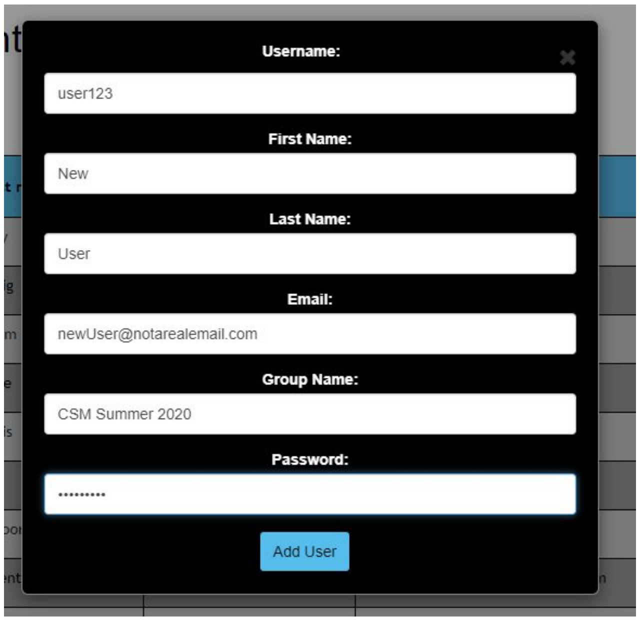
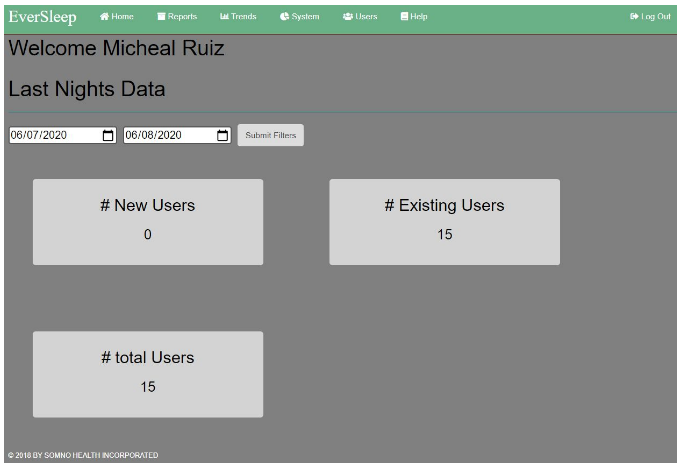

Username: user123
First Name: New
Last Name: User
Email: newUser@notarealemail.com
Group Name: CSM Summer 2020
Password: •••••••••

Figure 3: An example of the Add-User popup.
A section for Group Name is used to specify which group the user will be assigned to.

Portal Graphical Changes

Another interesting element from a coding/learning standpoint was the revamping of the web portal from an aesthetic standpoint. Using the "inspect element" tool every early CS student uses to mess with their friends from a young age as an actual coding tool was a great experience. This basic Google Chrome tool allowed us to easily find where the style code for every element of the portal was located. Figures 4 and 5 below show the original look of the portal and the new look, which better matches the company's color scheme.



EverSleep Home Reports Trends System Users Help Log Out

Welcome Micheal Ruiz

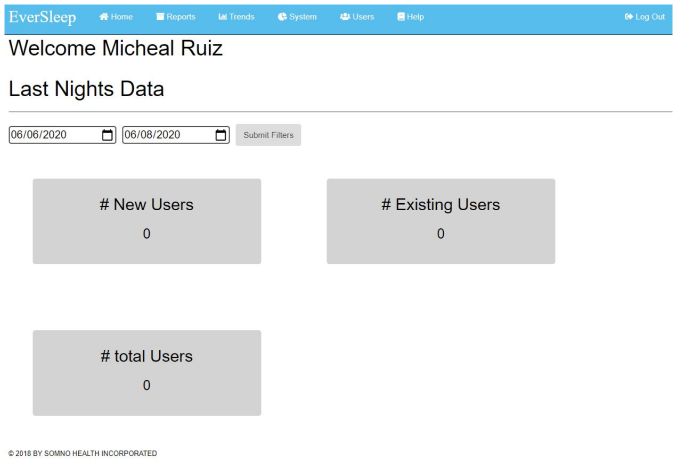
Last Nights Data

06/07/2020 06/08/2020 Submit Filters

# New Users	# Existing Users
0	15

total Users
15

© 2018 BY SOMNO HEALTH INCORPORATED



EverSleep Home Reports Trends System Users Help Log Out

Welcome Micheal Ruiz

Last Nights Data

06/06/2020 06/08/2020 Submit Filters

# New Users	# Existing Users
0	0

total Users
0

© 2018 BY SOMNO HEALTH INCORPORATED

Figures 4 & 5

Database

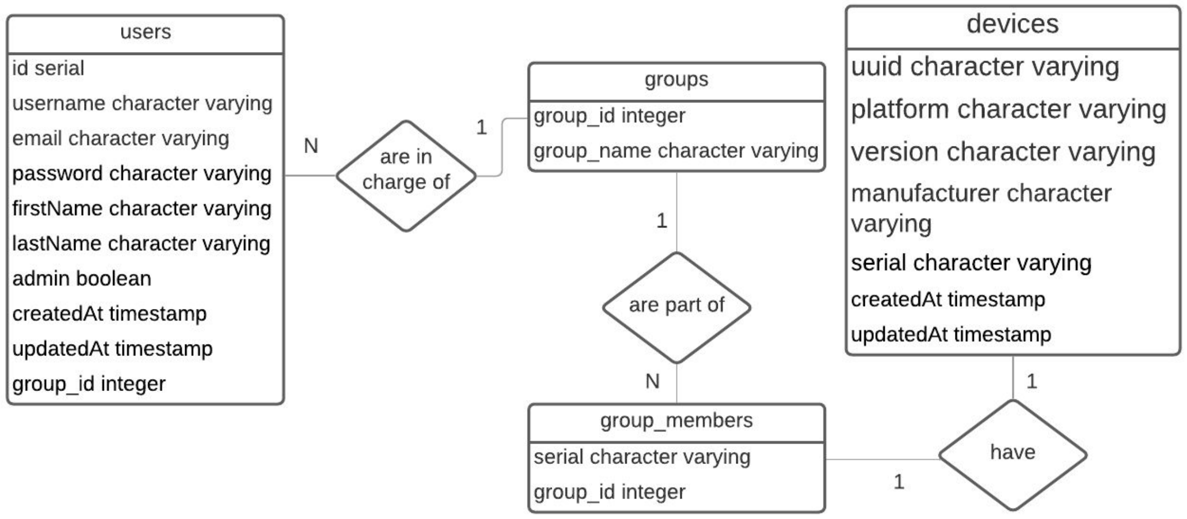


Figure 6: Database entity relationship diagram

The database stores a lot of data, so we focused on the part of the database we were using for the entity relationship diagram in Figure 6 above. Our implementation stores the website user's sign-in information, then allows them to see sleep data for members of their own group. A lot of the functionality for the database already existed when we started, we just needed to give the users table a group ID column, fill in the new column for each row, and add an admin group.

Quality Assurance

Quality Assurance was very meaningful for this project in particular since the client was looking for our finished project at the end of the field session to be "ready to go" for customers. They hope to be able to sell the technology to individuals as well as physicians. A bulleted list below details the process we followed to ensure that the product was ready to be delivered to our client's customers at the end of the field session.

- User Interface Testing
    - Manual testing the Data Portal's capabilities from client perspective
    - Groups functionality
        - Testing done as an Administrator and as a Group Leader.
    - Searching capability
        - Testing that all variables can be searched on properly.
    - Login
        - Checked for hashing, investigated use of passport node.js library
    - Graphs
    - PDF output
    - CSV output
- User Acceptance
    - Feedback from the potential group leaders
    - Continuous feedback from client
    - User interface feedback
- Code Reviews
    - Variable names
    - Commenting
    - No duplicate code
    - Efficiency (especially with querying from database)
    - Error Handling
- Documentation
    - Write good documentation for setting up AWS/Server and deploying code so that future field session groups can get a faster start
    - Document changes made
    - If time allows document code/process as a whole

- Security/Privacy Checklist
    - Check for SQL injection attacks
    - Research “passport” library weaknesses
    - Ensure that the database contains no user names, ex: Joe Smith to ensure patient privacy on website

Results

Features Not Completed

We did not have time to implement automated reporting, which would have sent emails to group leaders detailing some patient data. Our client mentioned that he was fine with sending out weekly emails showing statistics for now. While the creation and assignment of groups is available in the end product, the ability to add individual serial numbers to groups has not been implemented.

Performance Testing Results

Performance testing for our project consisted mostly of ensuring that the web portal can query the database in a reasonable amount of time. This involved making the individual queries as simple as possible while still returning the intended information. While the new queries were never formally timed for efficiency, the new search features on the site did not perform with any noticeable lag during user interface testing.

Summary of Testing

New or updated functionality was tested primarily through accessing the portal itself. When improving search functionality, new filters were tested with values on the portal to ensure that they return accurate information and work with the existing SQL operators. Repairing the existing filters was done in a similar fashion, where the corrected filter was tested in the portal by both our team and the client. When testing the framework for the group functionality, we used SQL queries both in the portal’s source code and directly in the database to validate that new groups were created as necessary.

Future Work

A future extension of our work would entail completing group functionality, as well as adding functionality for reporting group trends to the group leaders. Some optimization of the code might be necessary as well, since we were unable to perform this optimization ourselves. Finally, some work needs to be done to make the website more secure and safe for users. This would involve ensuring that the database is not vulnerable to SQL injection attacks and ensuring adequate safety of all users as they are accessing the portal.

<u>Lessons Learned</u>

Our team faced large initial hurdles in setting up a development environment, understanding a large code base, and learning the new technologies we needed to use. Ultimately we spent more time than we would have liked getting ready to actually refactor the existing code. This process could have been more efficient had we had access to better documentation. Hence, we created better documentation for the next group so they can set up a development environment quickly. This way they can get to editing the code and learning what it does in a shorter time period. The lesson learned here is that incomplete or non-existent documentation can delay progress significantly.

Additionally, we were given two different code bases at the start of field session, and we spent valuable time choosing which data portal to start off with. If we had shortened our time researching the two different data portals created by two separate field session groups, then we could have had more time adding additional features. Essentially, we spent the first week dedicated to understanding and researching the two different code bases/data portals, giving us less time to make changes. So we think if we had been more decisive in that first week we could have made more progress, the lesson here is to stay focused on the end game which was creating a functional data portal in the time given.