UKE M-Lab Project 2021/2022: Retrospective

This document contains a review of the completed UKE project of the 2021/2022 university course of the UHH M-Lab. It will shortly mention aspects, issues and experiences we as a project group went through during the development.

The UKE M-Lab Team 2021/2022:

Corvin Biebach – Bachelor Computer Science
corvin.biebach@studium.uni-hamburg.de

Maximilian Brosius – Master Computer Science
mail@maxbrosius.de

Fynn-Ole Menk – Bachelor Software-System Development
fynn.menk@gmail.com

Anni Reinert – Bachelor Business Computer Science
anni.reinert@rb-reinert.de

Noah Scheld – Bachelor Computer Science
nickels12er@gmail.com

Arne Struck – Master Computer Science
arne.struck@studium.uni-hamburg.de

Mudassar Zahid – Bachelor Computer Science
mudassar.zahid@studium.uni-hamburg.de

Table of contents

1 Design and Screen

During the development and while simulating the app we found out, that we don't own a suitable reference device. This is why we decided on the tablet devices of Pixel C and iPad Pro.

Due to the screen size constraints, we concluded that the screen size and the readability compete with each other. Our solution with the current interface displays the graphs rather big. Adding more graphs to the screen results in the necessity to scroll the graphs. For emergencies probably not the best solution.

Adding to that, buttons need to be displayed large as well so that interacting with them still feels intuitive and easy, especially with a medical emergency situation in mind. All displayed widgets could be scaled down, but for the context of M-Lab, we decided against it and stuck to our design decisions.

2 Alarm Management

We learned during the requirement elicitation that alarms of different parameters don't generally exclude each other. The only special case we singled out is that in Defibrillator-Mode alarms can be generally muted. Reanimation and chest compression leads to a behaviour in which most sensors values aren't useful, so alarms need to be muted for a short amount of time.

Because the device only plays one alarm at a time, alarms must be sorted according to their importance to make prioritisation decisions. For example, when a SpO2 and NIBP alarm occur at the same time, the device can only play one alarm sound. The device needs to decide which sound to play. The prioritisation for the alarms is handled by evaluating the triggered boundary deviation.

For example:

Sensor: Heart Frequency

Upper Boundary: 120

Lower Boundary: 50

Sensor Deviation: 10%

Middle Alerts for upper Boundary: 121 – 131

High Alerts for upper Boundary: > 132

Middle Alerts for lower Boundary: 49-45

High Alerts for upper Boundary: <45

In addition, two things are partly mutually exclusive: The screen should always show the current status, but the user should still be informed about what just happened, even if it is no longer current. For example, the SpO2 Sensor is displaying about 50% but the doctor sees, that the patient is in not so a bad condition. He fixes the sensor and places it on another finger. Why does the user need to confirm the alarm on the device, when the sensor is at 99% already? On the other hand, when a sensor hits a boundary very shortly and goes back into its boundaries, our application would not inform the user fast enough or switches too fast back to "no alarm". This could be handled with a timer delay. Before an alarm switches back to a lower priority it should be displayed between 7-10 seconds. Since flutter has issues with timers and their accuracy, this could not be implemented without running into major issues. This issue with timers also leads to the problem of the sounds not starting directly when an alarm occurs. Fixing this issue would lead to overlaying sounds several times. This is also a problem of our mocked data which is displayed in fast jumps and in probably bigger steps than a real device would update this data in.

A user should always be informed when the state of the patient changes for the worse, so a smart boundary adjustment was only implemented for middle alerts. Is a middle alert activated 3 times in a row without becoming a higher alert or different alert (for example from higher to lower boundary alarm): the user gets the opportunity to adjust the boundary with a button. By tapping the button the application lowers or raises the boundary.

3 Data

Mr. Neuhaus from Weinmann provided us with data twice, once actual recorded ventilation and ECG data and the second time generated simulation data.

As the patient scenarios are very complex, finding data that fits our needs is near impossible. Even just finding a healthy ECG wave is a very hard task. That's why we are building our datasets ourselves.

Since we immediately wanted to visualize the provided data within our application, we did not bother constructing a complex JSON file. Thus we built a simple list holding dictionaries for each graph (ECG, paw, pleth, etc.). Since the data is raw and unfiltered, a lot of noise was present. To filter out the noise Mr. Neuhaus recommended using an algorithm called simple moving average. We were now able to display an ECG wave as a graph. However, for a

realistic representation of the associated heart frequency, we would have to analyze the number of pqrst complexes in one minute. Which would have taken too much time to implement correctly, we decided against analyzing our graphs. We still lacked a data source to realistically display a heart frequency.

Our most basic approach was generating semi-random numbers and just displaying those. The problem that arises with randomly generated numbers is that a random data point is not dependent on its predecessor. In the realm of medical data, this leads to serious inconsistencies. Since the heart frequency might be 120 at time x and 75 the next timestep this does not mimic a real trend for a heart frequency (even in the most dramatic cases hf would not drop by that much).

To tackle this issue, we decided to generate our heart frequency with an algorithm called random walk. A random walk generates a list of values each based on the last value to allow for a more realistic trend. Our implementation of a random walk can be found under the section "Random Walk" in the notebook. The relationship between ECG and heart frequency is not unique and almost every graph brings its associated value (i.e. pleth → spo2, co2 → co2Absolute). With our strategy, we were able to generate all the necessary data streams.

With a combination of the random walk and the clean, easily loopable data we received, we had the tools to construct any scenario we wanted. Now that we had all the basic components needed, we tried to replicate the given patient scenarios as close as possible.

4 Flutter Project

Using Flutter as a Framework offered a lot of possibilities for implementing the design with widgets. For the graph visualization, we used Syncfusion. Because of the amount of data we experienced performance issues during simulating the app. At the end, when the application was running on a real iPad as a built app, the performance issues were not so dominant anymore.

Another issue we had to face was integrating the self-made sounds into Flutter. We used the flutter package Audioplayer and especially the Audiocache. The audio player has known issues we experienced during the development (e.g. timer and sound overlaying).

All in all, we as a team came to the conclusion that Flutter is not suitable to develop such an application for a real future 3-in-1 device.

5 Project Management

At the beginning of the semester, we followed the suggestion and decided on Git-Lab as our project management tool. Looking back, we would not make the same decision again. Switching the tool during the development was not an option.

Due to the time schedules of 7 students, we didn't meet daily or regularly aside from the mandatory course meetings. A lot of organisation and management was made via text messages in slack. Meetings took place more spontaneous and lasted longer than normal meetings in project management would have. Furthermore, we underestimated the workload for M-Lab and some team members will have to neglect their exams or other deliverables for the winter semester.

In the beginning, the aim of the project was to find new approaches for the alarm management of a 3-in-1 device. For that task, we had to build a whole interface of monitoring, ventilation and defibrillation with their functionalities which lead to developing two projects at the same time. Therefore, the workload was larger than expected especially with the data processing.

6 Summary

For the M-Lab project, we are very proud of the final application we developed over the last month. Everyone learned a lot about app development and could gain knowledge in new areas. We are aware, that there is room for improvement and there is always a part of the project, that could be done better or completely different. But in the end, we are happy to deliver an application for the UKE where they can experience our approaches of a prototype and ideas of a 3-in-1 device.