Stayin' Alive CPR Training App

CPR is an important tool that can help save a person's life by pumping blood and delivering oxygen to the brain until advanced treatment can stimulate the heart to start working independently. Unfortunately, CPR has a low survival rate, especially among children, with only 11% of the 16000 children who require CPR annually surviving to discharge. While there are existing CPR feedback devices, these tend to suffer from limitations that greatly limit their usefulness, such as high price, inaccurate calibration and incomplete feedback. To help bridge this gap, the Stayin' Alive CPR Training App is designed to help provide immediate and longitudinal feedback about the users CPR technique in an easy to use, cost effective manner.

The Stayin' Alive CPR app is an iOS app developed using Xamarin in Microsoft's Visual Studio. This specific app platform was specifically chosen as it allows for a high amount of code reuse between iOS and Android apps, so despite the current app being specific to iOS, future work on an Android app can be quickly developed using much of the same code.

The App receives data from a sensor in order to calculate and provide immediate feedback to the user. The app communicates with the sensor over MQTT, which is a communication protocol that is designed for communication with embedded devices. The information that is sent from the sensor to the app includes acceleration data with corresponding timestamps. From this information, the app derives the depth and rate of compressions as well as making a determination if the trainee has removed their hands from the mannequin. While this information is important, it can provide incomplete information about the training session as there are events that cannot be detected by the sensor. For this reason, the app provides an easy way for the user to mark when the trainee stops compressions to use an AED, administers epinephrine, provides ventillations or transfers the victim to a backboard.

After the training session the user is presented a summary of the events of the training session. This includes graphs that display the depth and rate of compressions, as well as the user average and target range for both metrics, as well as a timeline displaying user entered events, such as applying an AED or administering ventilations. The user is also given the option to export this data in order to track trainee performance over time.

While we were able to perform two rounds of testing, we ran into issues that affected our ability to perform in depth testing. While the app is one part of a two part system, the sensor was developed by another team and was subject to its own setbacks and constraints. Delivery delays and equipment that didn't meet expectations made it necessary to develop a method for testing the features of the app independently. To do this, we developed a script that creates data within the expected range then sends information to the app. Because of this issue, the majority of the feedback we received during testing was concerned with the ease of use of the app and the feedback provided to the user. Whille the users generally found the interface easily understandable, they provided ideas to improve the feedback such as smoothing the output data as it was previously moving quickly and may be hard to read at a glance as well as allowing the output graphs to be scrollable, as they compress to the point of illegibility in longer training sessions.

While the app is well on its way to becoming a useful product there are features and improvements that may be made in future development. As the app is currently only for iOS, an Android implementation may make the app more accessible to a larger number of users. Future developers may want to implement an account system in order to help facilitate tracking users over time. For the same purpose future development may include a way to directly upload training data to a database as organizations may wish to track a large number of users.