COSC 6355: UBIQUITIOUS COMPUTING

Project Name: Sit-Stand-Walk at Heartbeat

Team 8:

Keshav Kasichainula, Mohammed Emtiaz Ahmed, Shaila Zaman, Taqees Duka, Bharat Surimalla

INTRODUCTION:

This is a health-related app which collects data from health kit, process and upload the data into the Amazon Web Service(AWS).

DESCRIPTION:

Our project has 2 parts: iPhone app and it's watch extension.

iPhone App:

On the iPhone app part, it collects the Step count and heart rate data from HealthKit and saves them in CSV format. At the home page user have to login by name, which is required. Because the name is used for creating the data file, to identify which user accounts for that data. After providing the name, the user can get a combined graph view for both heart rate and step counts.

There are two other views on heart rate and step separately. In each of the view, the graph for hourly data and raw data from the watch is showed in a table.

There is an option to upload those data to Amazon Web Services(AWS) for further analysis. When the user taps the "Upload" button, the CSV files will be saved in AWS and will show an alert message after finishing uploading.

Watch extension:

The watch part is an extension of the built-in system, which delegate users to control the health monitoring time duration. When user press the "Start Tracking" button, it starts to collect data regarding heart rate and step count. The user can finish the task by pressing "End Tracking" button. From here, the apps stop collecting data and requests Healthkit to save all the data. The watch app extension not only runs in the foreground but also in the background.

Data collection time frame:

(1) Every minute our app collects the step count.
(2) Every 5 second our app collects the heart rate data.

DESIGN

The following section will explore the design patterns adopted and the main motivations that led to the result.

Software Methodology

Before anything can be started on the application or the watch itself, there must be undoubtedly an authorization from the end user giving us access to the information of their personal statistics.

Apple API handles this so a simple call to the API provides us the ability to get their permission. Once permission is given, our app can collect data from Healthkit.

The actual step data comes from a query we can make to healthkit. Here we used HKStatisticsCollectionQuery; it gives us step count data based on our timeframe and data unit. For heart rate data we used HKSampleQuery; it gives us heart rate data based on our timeframe. After collecting all the data, we save them into csv files for further use.

Different graphs have been shown in this app for the heart rate and step number. It is not a user-friendly approach to show all the data only in the tabular format. That's why we used graph and table both. We used Charts framework to visualize the per hour data. Two kinds of charts have been displayed here, the bar chart for the steps and the line chart for the heart rate. The data have been formatted from minutes to hour to show the data pattern more clearly in the graph. For steps, it shows the total steps in an hour and for heart rate, it shows the average rate in an hour. For better comparison between step count and heart rate data we also generated a combined graph.

Perhaps the most interesting function and design option that we worked on was the ability to send the csv to the cloud. We decided Amazon Web Service (AWS) would provide the best flexibility on rendering our data. Not to mention AWS itself is so integrated into many different applications that the practicality of using the data grew immensely and hence why we decided to use that service to transmit our data.

Finally, for watch extension, we created session to communicate between watch and iPhone app. We used HKWorkout option for collecting data even the app is in background.

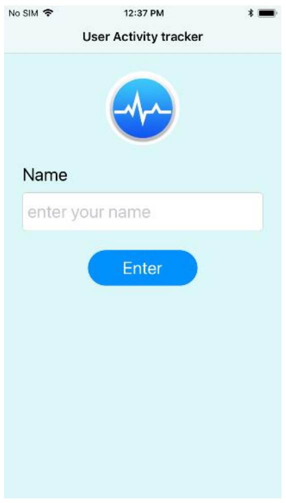
UI Design

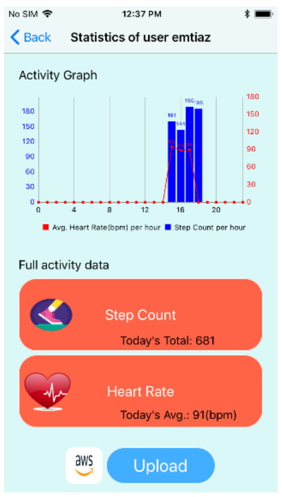
The UI is of paramount importance. If an aesthetic and easily viewable experience is not offered, then the rest of the contents is rendered mute. UI design took a very important precedence for us from the start. The key in our minds is simplicity. When an app becomes cluttered or inundated with too much information then the detraction has already begun. Our app spans two applications, one on the watch itself and the other being the iPhone device.

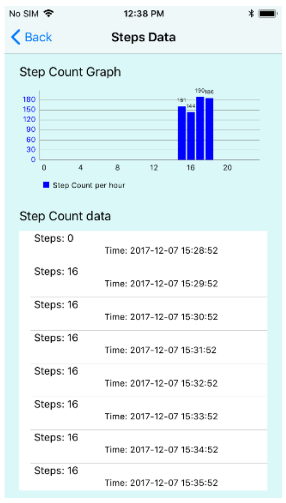
On the watch itself there isn't a whole commodity of action happening. There is a simple Start Tracking button that is more than enough to collect the data in the background and produce the results we need.

Transitioning over to the application on the device now. Once the application is opened the main screen presents a simple login type in which the user can type his/her name. We used scroll view so as to not obscure the actual field when the keyboard appears. From the onset there is an established theme of light blue that is followed on all the pages. The reason this light blue was chosen is because it invokes a calm and collected mood. It also represents a sound mind and healthy body.

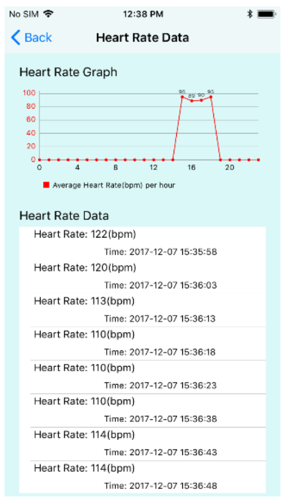
Next there is the main information of Step Count and Heart Rate highlighted by UIView which immediately draws the attention of the end user as it stands in contrast to everything else. Lastly, there is an upload button which is further illuminated by the AWS Amazon logo to showcase to the end user that their data will be traversed to the clouds. The remaining pages of the application showcases graphs relating to the steps and heart rate respectively as well presenting a table view of the data itself.







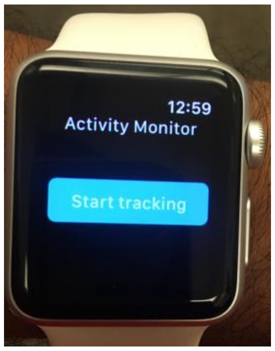
Steps: 0	Time: 2017-12-07 15:28:52
Steps: 16	Time: 2017-12-07 15:29:52
Steps: 16	Time: 2017-12-07 15:30:52
Steps: 16	Time: 2017-12-07 15:31:52
Steps: 16	Time: 2017-12-07 15:32:52
Steps: 16	Time: 2017-12-07 15:33:52
Steps: 16	Time: 2017-12-07 15:34:52
Steps: 16	Time: 2017-12-07 15:35:52

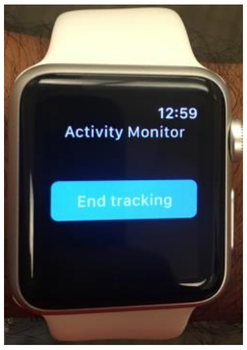


Heart Rate: 122(bpm)	Time: 2017-12-07 15:35:58
Heart Rate: 120(bpm)	Time: 2017-12-07 15:36:03
Heart Rate: 113(bpm)	Time: 2017-12-07 15:36:13
Heart Rate: 110(bpm)	Time: 2017-12-07 15:36:18
Heart Rate: 110(bpm)	Time: 2017-12-07 15:36:23
Heart Rate: 110(bpm)	Time: 2017-12-07 15:36:38
Heart Rate: 114(bpm)	Time: 2017-12-07 15:36:43
Heart Rate: 114(bpm)	Time: 2017-12-07 15:36:48

(a) (b) (c) (d)

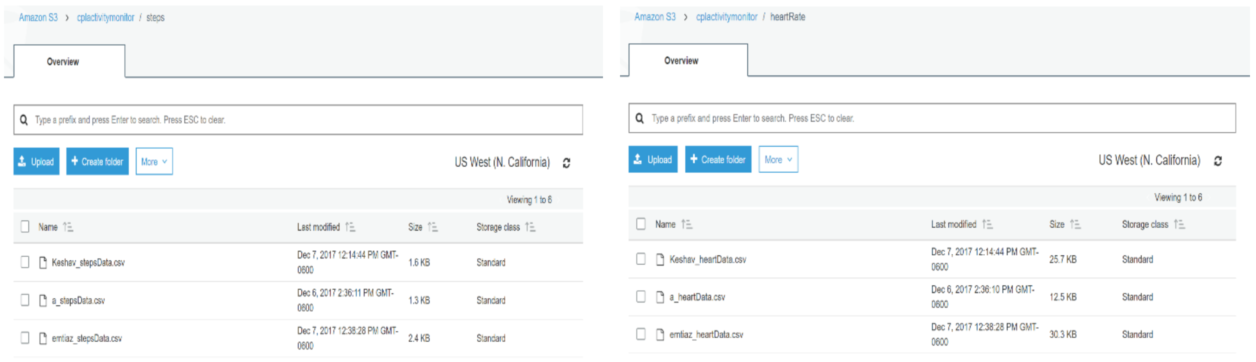
Figure 1: (a) Home page (b) Statistics page; it has combined graph of heart rate and step count based on per hour data, Step count and Heart rate view with today's total step count and average heart rate, AWS upload button (c) Steps data page; it has step count hourly graph with all raw data in a table view (d) Heart Rate data page; it has average heart rate hourly graph with all raw data in a table view.





(a) (b)

Figure 2: Watch extension part (a) Initial view after authorization (b) View after tapping the start tracking button.


Amazon S3 › cplactivitymonitor / steps
Overview
Type a prefix and press Enter to search. Press ESC to clear.
Upload
Create folder
More
US West (N. California)
Viewing 1 to 6

Name	Last modified	Size	Storage class
Keshav_stepsData.csv	Dec 7, 2017 12:14:44 PM GMT-0600	1.6 KB	Standard
a_stepsData.csv	Dec 6, 2017 2:36:11 PM GMT-0600	1.3 KB	Standard
emtiaz_stepsData.csv	Dec 7, 2017 12:38:28 PM GMT-0600	2.4 KB	Standard

(a)

Amazon S3 › cplactivitymonitor / heartRate
Overview
Type a prefix and press Enter to search. Press ESC to clear.
Upload
Create folder
More
US West (N. California)
Viewing 1 to 6

Name	Last modified	Size	Storage class
Keshav_heartData.csv	Dec 7, 2017 12:14:44 PM GMT-0600	25.7 KB	Standard
a_heartData.csv	Dec 6, 2017 2:36:10 PM GMT-0600	12.5 KB	Standard
emtiaz_heartData.csv	Dec 7, 2017 12:38:28 PM GMT-0600	30.3 KB	Standard

(b)

Figure 3: AWS csv file upload (a) step file saved under step folder based on user name (b) heart rate data file saved under heart rate folder based on user name.

<u>TESTING</u>

Testing plays a crucial role before deployment of the app into real world. Following are the testing techniques we have followed.

Each module is unit tested to check its functionality. The Home screen where the user enters his/her name to get the statistics validates the name and gives access to the statistics only if the authorization is valid.

Next as a test, ensured the data was populating from the resulting query. If this was not the case, then no data would be displayed, and we would know there is some issue in the particular functions.

We were also able to validate the data, meaning whether the data was accurate or not. This could easily be checked in one of two ways either A) Check the step rate or heart rate directly from the Apple Watch or B) Check from the native Health App that is on every latest iPhone. We opted to use the latter as the Healthkit API also includes data collected from the iPhone which many users do not know. If it is enabled, the iPhone also collects from its side based off its GPS. We wanted to get the culmination of everything.

Once data was being pulled in and we validated it for accuracy, the hard part of the testing was now finished as the data is the most important thing. The rest we were able to surmise whether something was working or not.

Finally, for functional testing the functional part, we had to ensure the cloud services was connected and appropriately sending the data. This we were able to do by logging into the AWS itself and validating whether the file was sent over or not. Again, this was an outsource service we were using so testing was slightly more difficult as we did not readily have information if it failed, however upon success the result would clearly be on their web server.

Lastly for the testing the application once completed, we conducted a series of exercises to simulate the real world. The app related to a watch and the person wearing the watch is made to walk on treadmill for 5 minutes at 5mph speed and then the statistics given by the watch are compared with the statistics given by the treadmill. This is repeated with different persons walking with different speed for different time. In each case the time and calories burned is compared with the ones given by treadmill. Similar testing is followed for other functionalities as well.

COMMUNICATION:

As a group project shared by 5 people we wanted to have an effective communication among ourselves to make the project work going smoothly without any issues. We have decided to have group meeting once in a week either before or after the class as it is easy for everyone to attend. We have created a repository in github for the code and any updates to the code are committed over there. It acted as a version controller and made easy for everyone in the group to know the modifications and added features over the development time period. We have also used slack to exchange ideas and code snippets. We have created a separate WhatsApp group to be in touch with progress of other members of the team. With all the efforts taken to communicate well we have completed each task of it on time.

WORK DISTRIBUTION:

- Keshav Kasichainula
    - (i) AWS part
    - (ii) Read step count data from health kit
    - (iii) UI
- Shaila Zaman , Taqees Duka & Bharat Surimalla
    - (i) Heart rate graph
    - (ii) Step count graph
    - (iii) UI design, Table view
    - (v) Testing
- Mohammed Emtiaz Ahmed
    - (i) Read and save heart rate data from health kit
    - (ii) Watch extension part
    - (iii) Combined graph
    - (iv) UI

CONCLUSION

In conclusion at the end, our app demonstrates the ability to capture analytics and present it in a meaningful way. That is only the first step to a broader range of applications in which the data can be synthesized and used appropriately. This app works as a personal monitor on to your way to a better, stronger, healthier life.