Assignment 5

Morse Code Interpreter app

Final Report

Pengcheng Xia

24824887

**App Features:**

This application uses an arduino motion sensor to collect motions then generate Morse code and send it to client through Firebase. On Client side, It will decode the Morse code by using Morse code table and print out the the message on a local webpage. Users will be able to start or terminate the app through client side by clicking the checkbox ‘Accept signals’ and read the Morse code which detected from the arduino motion sensor as well as the decoded English message.

Functionality for iteration 1:

* Motion sensor collects all motion data
* Server can classify motion data into long motion and short motion
* Server can generate Morse code based on the motion data collected
* Morse code will be uploaded to Firebase
* Client can collect Morse code from Firebase and decode it into English
* App will display decoded English message on the webpage

New added functionality for iteration 2:

* App supports for numbers:[ 0 - 9 ]
* App supports for punctuation: '.', ',', ':', '?', "'", '-', '/', '(', ')', '@', '='
* App is able to stop communication from the server side
* App is able to start or pause communication from client side

**Server Side:**

The server will use timestamp to calculate last time of a motion. For instance,

when a motion is made, timestamp is taken. After end of motion, another timestamp is taken and their difference calculated. If the difference is more than 5 seconds, the motion will be count as a long motion. Otherwise, it will be a short motion. Server will upload "L" for long motion and "S" for short motion to firebase.

Moreover, if the time between the motion.on(‘motionend’) and the next motion and next motion(‘motionstart’) is more than 7 seconds, server will know there is a new word and upload string ‘new\_word’ into Firebase. On the other hand, If the gap time between two motions is more than 3 seconds and less than 7 seconds, it will be a new letter. Server will upload "new\_letter" to firebase.

**Client side:**

Each time the client detected a child is added, it will do as follow:  
Firstly, It will print out the Morse code it received on the webpage then:

* If added child on firebase is a "new\_letter",A string called "letter\_to\_decode" will be sent as a parameter to the decoding function.The output will be incremented to the string variable "output", which will be displayed on the webpage.
* If added child on firebase is a "new\_word",Client will be the same as when it receive “new\_letter” except a space will be added to the output.
* If added child on firebase is a "S” or “L”, client will incremented into "letter\_to\_decode" string variable.
* If the child added to firebase is not "S", "L", "new\_word" or "new\_letter", the client will changes output to "WE HAVE BEEN HACKED" to let the user know that something is wrong and deletes everything in the firebase database. Since these 4 are the only inputs that we upload from the server side.There is no other way anything different can make to the client side, unless somebody changed something in the database or due to some type of data-loss, giving an incorrect value.

**Test Plans:**

The tests we have runned for this application are mainly Black box testing. Firstly we decide what we were going test based on the user stories. Then we prioritise them in class High, Medium and low. For each test plan , we made two test cases. There are pre-conditions for each test, for example, the motion sensor should be in working condition. To make sure of that, we would check the sensor is correctly connected to the board before the test. The sample of our test plan is included in folder “Iteration2\_Docs”.

**Bug fixing:**

Algorithmically speaking, the program is quite reliable as it is simple in design.  
Only real way any faults can occur is if something happens with the firebase server  
during the storage of the data or while it is been sent or received. For instance, firebase may go offline for maintenance. There is one limitation to the program that is non-obvious. In the user interface, when the checkbox is unticked, the client side just ignores signals sent from the server. The server will still keep accepting motions and upload them to firebase but those data will just be ignored from the client side. once the box is ticked again, new children uploaded to firebase will be accepted by the client side.

**Time tracking:**

13 May: Group meeting (Working on the User Stories)

16 May: Making test plan for iteration 1

17 May: Writing Backlog and Coding for new features for iteration 2

20 May: Coding for index.html and updating user stories

21 May: Run testing for Client and server, Bug fixing

22 May: More testing and finishing up for the iteration 2 submission