Test Plan

To: Professor Pisano

From: Arley Trujillo, Shivani Bhatia, Laura Salinas, Priya Kapadia, Steven Graham

Team: 14

Date: 3/29/18

Subject: Functional Deliverable Test Plan

1.0 Summary of Customer Requirements

Our customer requirements from last semester can be broken down into two modules, a software and hardware package that come together as a speech therapy system to help non-vocal autistic children take the first steps toward vocalizing and saying words.

The hardware module included a robot that would employ expressive language training by rewarding a child's correct vocalizations with positive actions like dancing and singing. Along with this robot we needed a way to capture the children's vocalizations and also relay examples of what words sound like.

On the software end we needed to find a way to compare the spoken word to the model word and in return get a score. This score would be used to devise a leveling system that would be used to scale reward behavior based on how a child is progressing throughout the course. To bridge the gap between software and hardware we would need to create either a web or phone application that adult users would use to interact with our product. Together the whole system had to have a latency lower than three seconds from beginning to end and would have to store information by child for at least two to four years.

2.0 Total Deliverable Significance

2.1 CHiP Robot

This deliverable satisfies our requirement to have a robot that is capable of interacting with children and motivating them to begin speaking. From the beginning of the project up to this year we went through two iterations of robots and settled on our second choice as it provided a wider range of expressions that could be used as behavioral rewards for our users. In addition a wider range in actions, the CHiP robot, which looks and acts like a dog, has a more friendly appeal and is more likely to stimulate children to adapt to the system. Given that the robot is our way of bridging the gap between learning and playing, we wanted to make sure the robot was robust enough to satisfy a child's curiosity while not sacrificing the depth in skills it needed to perform.

2.2 Android Application

With the phone application we set out to create a user friendly way for parents or speech language therapists using our product to interact with and control the lessons the child is going through. Our Android application both meets and **exceeds** the requirements initially stated by our customer. While the application just needed to satisfy control over the robot, we wanted to go a few steps beyond this and create a platform from which adult users could also monitor the child's progression through dynamic graphs and charts. During regular use the application is able to cycle through hundreds of words and begin lessons with the click of a button. As vocalizations are captured and processed, the application displays the results on screen for a clear understanding of where children need additional help. In addition to the default reward action set for the robot, we also included an additional feature where every other skill that is built into the robot's SDK is accessible and can be changed to create a more unique customization according to a child's particular interests.

2.3 Similarity Score Algorithm

Arguably the most important part of our software package requirements was the algorithm that would take user input and compare it to correct pronunciations of that word returning a score. In order to accomplish this task we contacted a company whose product helped users perfect their pronunciation and fluency across various languages. Using their API in our application, this deliverable successfully meets and **exceeds** our customers requirements for speech recognition and comparison. The importance of this deliverable stems from its use as the foundation for the rest of our program to function. Without a similarity score to base ourselves off of, there would be no tangible way to measure how well children are saying words and if they are indeed progressing or not. This functionality also triggers a response from the robot so without a similarity score we would essentially not have a functioning system.

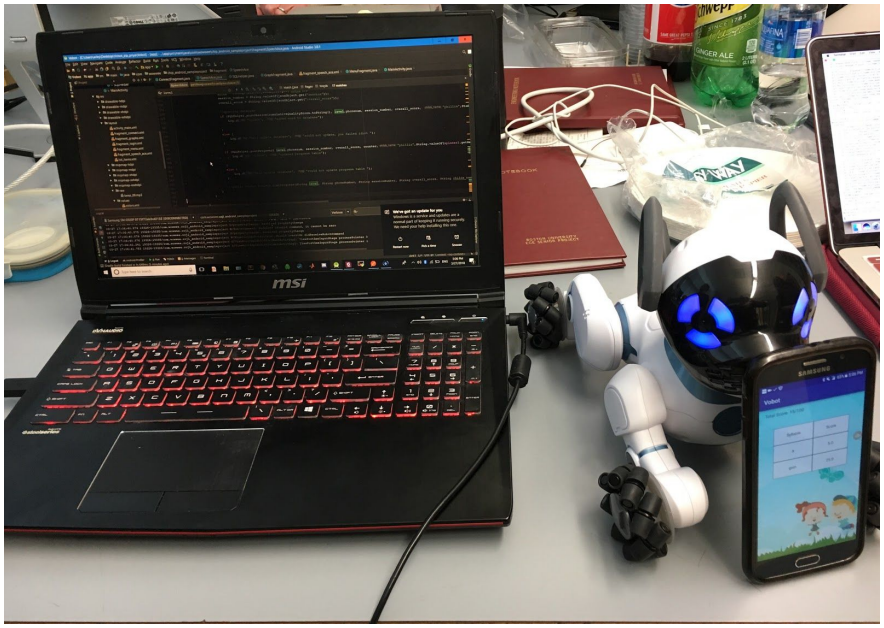
In addition to the gross score provided from this API, Speechace is also able to break down the input vocalizations into phonemes and syllables which we use to display to those monitoring a child's progress. We are not only providing a general idea of what words need to be improved, but an in depth breakdown of what syllables and phonemes making up those words need to be worked on.

2.4 Database

While not in our original requirements for this project, we decided to implement a database that would handle simple user authentication as well as a backend where children's progressions across words would be pushed to and pulled from during learning sessions. This deliverables significance lies in that it adds a layer of simplicity for the application user, and also serves to slim down the application size so that all these years of data are not stored locally on the device. The ability to pull and push information from the database allows Vobot to create profiles for each individual user and designate which learning data corresponds to whom.

3.0 Equipment and Setup

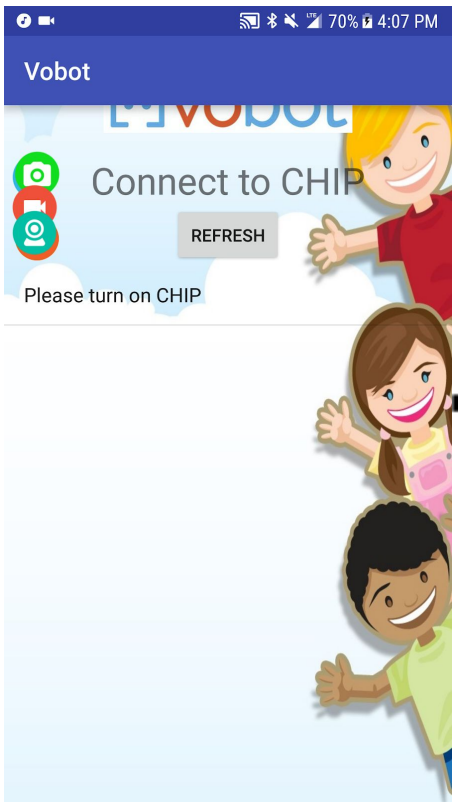
The equipment for this test consists of the WowWee CHiP robot, an Android cellphone and a laptop used to run the instance of the database on Amazon EC2.



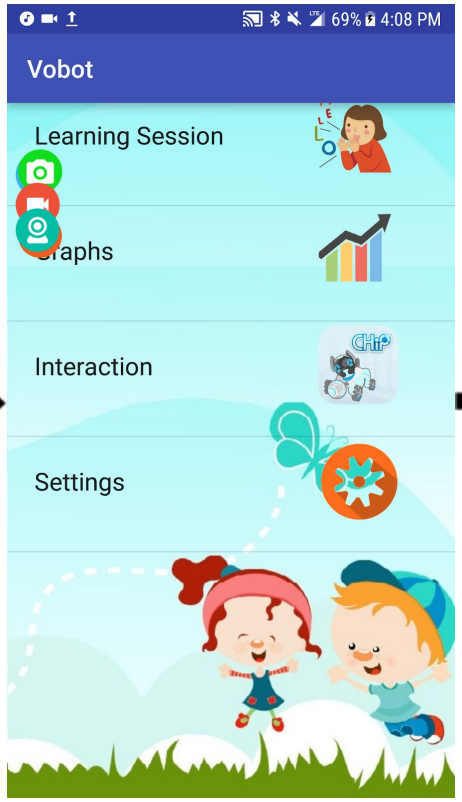
One complete loop of the test proceeds as such:

1. The application prompts user for input by playing a sample audio file of the chosen word to practice.
2. The phone records what the user says for 3 seconds and saves it as an audio file.
3. The audio file is sent to the Speechace API for analysis with relevant parameters.
4. The response similarity score is received by the application in the form of a JSON object.
5. The JSON response object is parsed and displayed on the screen as a graph.
6. If the similarity score is greater than the threshold (30% at this time), the robot displays a reward behavior in the form of a dance.

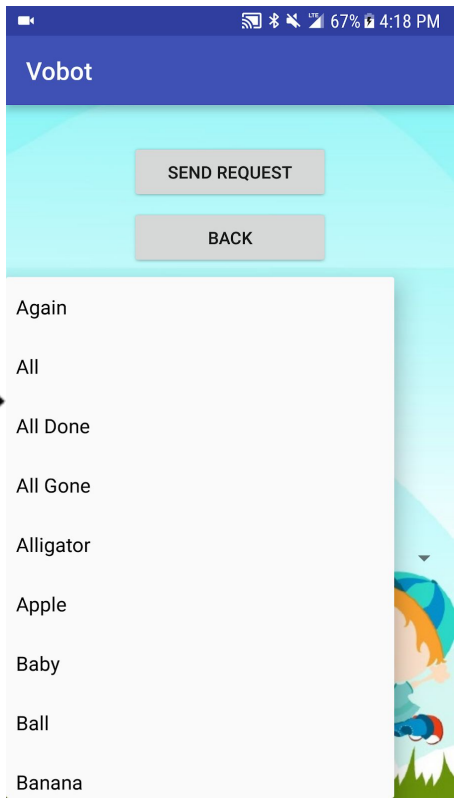
Application Cycle Screenshots (Left to Right)



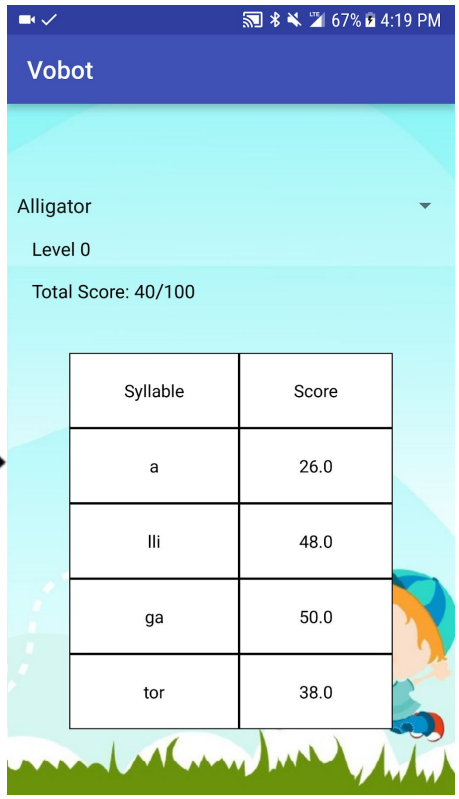
1



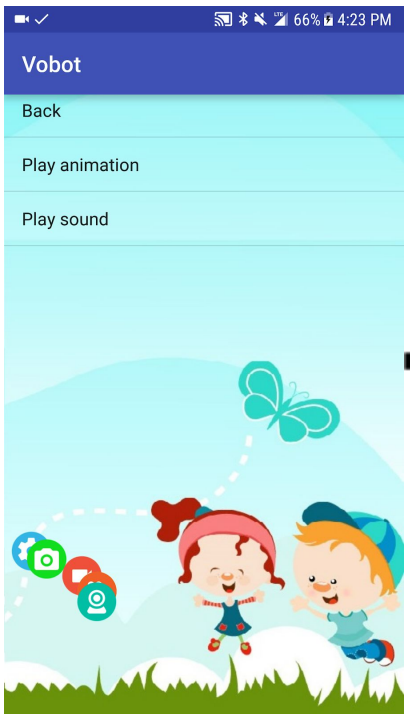
2



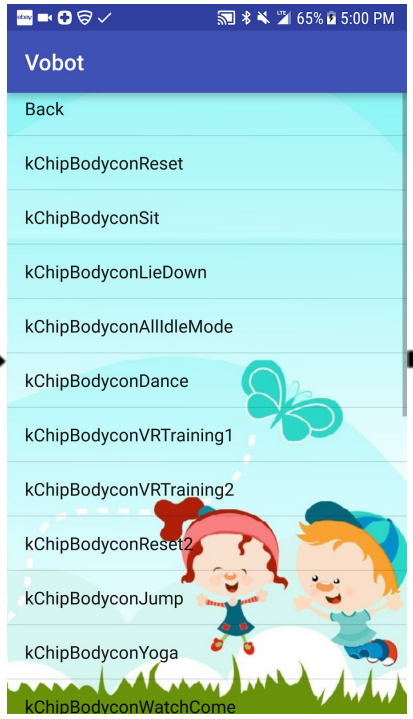
3



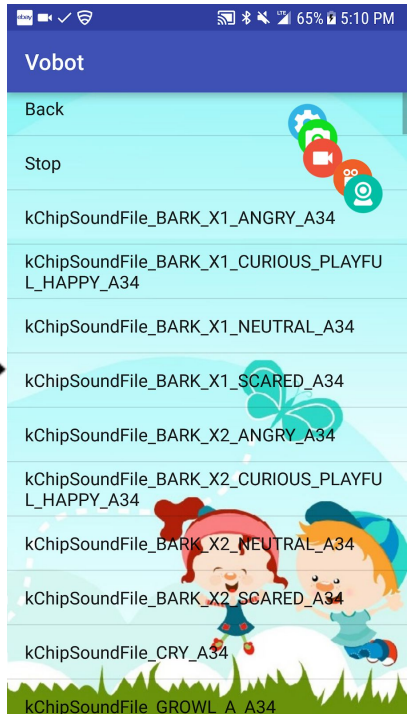
Additional Functionality



Customization Menu



Customize Reward Action



Reward Sounds

4.0 Data Collection

Data collection prior to this deliverable test needed to be conducted in order to find vulnerabilities in design and to correct timing for accurate audio processing.

One of the first problems we found was that as users we were unaware when the application started recording after the model word was heard on the speaker. We collected multiple .wav files of ourselves speaking after the model word was heard in order to figure out what kind of time gap we were working with. Once enough samples were gathered we added an audible *beep* after the model word that would indicate to the user that they could start speaking.

On the database side we needed to configure it to permanently save the phone number we inputted at the start of the application, this method allows us to only pull and push session information for the user currently logged in. To make sure that the application and database were communicating correctly we would post the cell phone number from the application and confirm the contents in the database were matching accordingly. This was done for several phone numbers to confirm functionality was consistent.

5.0 Success Criteria

This test will be successful if we are able to go through our outlined system pipeline without faults or latency delays greater than three seconds. In addition, the robot should only perform the reward behavior if the score is greater than the set threshold we have programmed it for.