**CSCI321 Draft #1 Requirements**

**Non-Functional Requirements**

* The program should be compatible with users who are using the Windows platform.
* The program should be compatible with users who are on MacOS. (Desirable)
* Demographics?
* The program should be easy to use and learn for new users. No specific knowledge is required. No addition training is required so that within an hour of navigating the program, the user should have a good overview of the system.
* The program should have user documentation and manuals to instruct users on how to use the program.
* The program should have an interface and will be in English.
* The program should provide a “Help” page for users at the main page. The help page provides sample tips and also insight on existing functionalities to the user.
* The program should provide users with the ability to give feedback or report bugs.

**Functional Requirements**

* The program would be capable of running with and without the Oculus Rift
* The program is capable of highlighting the correct finger required to hit the key displayed on the screen. (1st stage)
* The program is capable of displaying alphabets on screen which is used for training or testing of the user. As difficulty increases, it is capable of display a combination of alphabets and then existing words. The words are picked from a dictionary. (1st stage)
* The program is capable of providing user statistics which may include “Keys entered correctly”, “Accuracy” and “Total Score”. (How to calculate score and accuracy?) This is only implemented for much later stages.
* At earlier stages of the game, the program should be able to give stars to the user which are based on the accuracy the user has achieved on the stage. (3 stars – 90% accuracy, 2 stars 50 – 90% accuracy, and 1 star for any below that)
* The program is capable of making different sounds (positive and negative) depending on the input by the user when playing the game. If the user has input the correct key which is displayed on the screen, a “positive” sound will play and vice versa.
* The program should allow the user to shoot at keys when prompted. This allows the user to have a general understanding of the location of keys on the keyboard. (2nd stage)
* The program should provide the users with a virtual keyboard. The keys on the virtual keyboard will be widely spread for easier detection by the leap motion. Keys on the virtual keyboard will lighten up to tell users which key to hit.
* The program should be able to provide a test stage for the user. In this test stage, a key is prompted on the screen and the user must react with the correct finger which will be detected by the leap motion. After 5 seconds of non-responsiveness, the program should provide hints to the user on which finger to use. The first hint will be on which hand is to be used, and after another 5 seconds, only telling the user which finger to use. (A penalty of sorts can be given to the user) (1st stage)
* To provide a much more intuitive learning, each keys prompted on the screen can be associated with for example, the sound of different animals or the alphabet song. \*\*
* The program should provide the users with a competitive mode, where 2 users can challenge each other to see who has typed more correct keys then the other (1st, 2nd and 3rd)
* The program should provide users with a cooperative mode where they work together to beat the stage. A user will be the left hand and the other will be the right hand. A timer will be used, the users must complete the stage within the given time and accurately.
* The program should allow the user to type normal words. As the user progresses, upper case letters, upper case and lower case, a combination of upper case and lower case words with punctuations will be added. Accuracy of the users typing will be checked. (Needs work on 3rd stage)
* The program should allow support for keyboards. Instead of using the leap motion to learn how to type, the users can use a keyboard to learn how to type instead. (probably only for the last stage, and since the process of learning would be the same on a keyboard too)
* The program should have a “Free play” mode. This mode allows user to play freely, where the users are prompted to type a word or a short sentence. This mode is only available for users who have completed the first two stages. The user is given unlimited time, and as the user progresses the difficulty increases. The scaling of difficulty can either depend on the time spent so far on the mode or the accuracy so far in the mode.
* The program has a “Daily Practice” mode. The practice mode, depending on the stage of the user is currently in, gives a short refresher course on that stage. For example in stage 3, the user can be given a few words to type.
* The program is capable of collecting gameplay data from users. This data collected is used to find out the words where most people are having difficulty in which can be included in the “Daily Practice”.
* The program is capable of having multiple save files. The multiple save files allow multiple users on the same computer to play the game without having their progress mixed up.
* The program should allow the user to navigate through the menu using hand gestures. The opening of a fist might as “Enter” and the hand slapping gesture to navigate through the menu.