Department of Computer Science & Engineering
The University of Texas at Arlington

System Requirements Specification
CSE 4316: Senior Design I
Fall 2020

ALS Eye Tracking Group
ALS Eye Tracking APP

Anthony Vardaro
Michael Kosta
Zixiu Su
Thanhthao Le

Revision History

Revision	Date	Author(s)	Description
0.1	10.25.2020	AV, MK, KL, ZS	document creation

CONTENTS

List of Figures

1 Product Concept

In this section, the purpose, use, and intended user audience for the ALS Eye Tracking Application, will be discussed. The ALS Eye Tracking APP is an application used to allow someone to type out a sentence with only their eyes. The users for this application will be people who are unable to speak or type normally.

1.1 Purpose and Use

This application will be used to improve communication with anyone who is only able to move with their eyes. They will use their eyes to spell out words on a screen, forming sentences.

1.2 Intended Audience

The intended audience is for the medical field, to help someone who has a disease like ALS, or someone who does not have any arms, type only using their eyes. However, this product can be used by anyone.

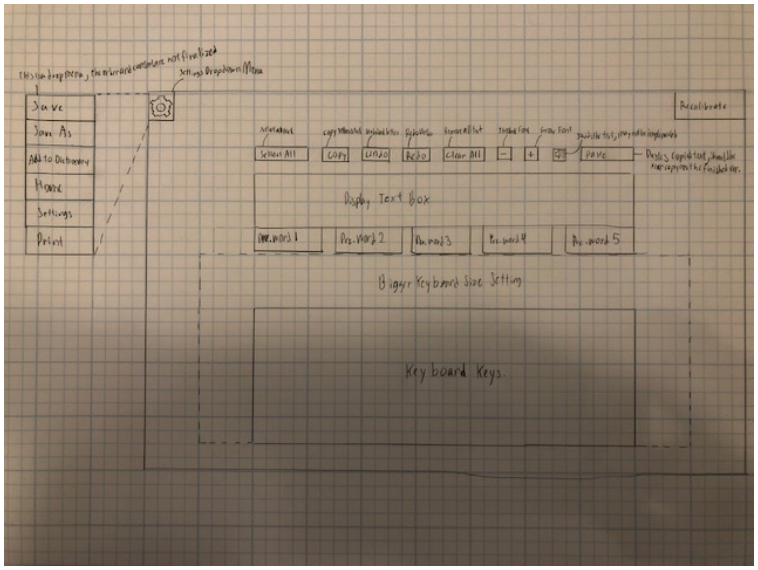

Figure 1: ALS Eye Tracker APP conceptual drawing

2 Product Description

This section will give an overview of the ALS Eye Tracking APP. This section will talk about the primary operational aspects of the product. We will talk about the features and functions that are found in the product, the external inputs and outputs, as well as, the product interfaces.

2.1 Features & Functions

This product is a software that will be used by an eye-tracker hardware. While using the eye-tracking hardware, the user will be able to create words or phrases with just their eyes. There will be a Keyboard, where the user will drag their cursor and hover over letters. Predicted words 1 through 5, which are slots used to display the words that are being predicted. A Display Text Box, which is where the letters being typed are displayed. Different buttons making it easier to edit the text. And a menu button, used to display a menu with different options.

2.2 External Inputs & Outputs

An eye tracking hardware is required to use the product. The eye tracker will track the users eyes movement and locate where the user is currently looking at. This will help create some words for the user to spell by just looking at letters on the screen. So, the user will be inputting different settings that are to their preference, then the user will use their gaze to type out a word. The application should be able to output the proper letter and word input and should be able to help the user type a full sentence.

2.3 Product Interfaces

The interface will have a text box on the top while having a keyboard under the text box. The keyboard will be divided in different sections to where all the letters are shown. After a letter is selected, sections will change their size. Some sections will shrink while other sections will grow based on the selected letter. There will also be predicted words displayed, based on what the user has typed so far. There will be buttons for selecting all, copying, pasting, undoing, redoing, clearing all, recalibrating, (increasing text size, decreasing text size, and sounding out the words; everything here is optional). There will also be a Menu button where the user can Save, Save As, Add a word to the stored dictionary, go to the home screen, access the settings, and print the type words; some of these are optional and may not be included in the final version.

3 Customer Requirements

For the ALS Eye Tracker APP to work, the customer has asked us to use a Tobii Eye Tracker, the customer has asked for the letters to change in size, and the customer has asked to be able to capture the letters that are being focused on.

3.1 Eye Tracker (Tobii 5)

3.1.1 Description

The eye tracker is required in order to track what the user is looking at according to the screen. It was required that we get a Tobii Eye Tracker, we decided to get the latest version, the Tobii 5.

3.1.2 Source

CSE Design Specification and Sponsor

3.1.3 Constraints

The eye-tracking hardware may be malfunctioning, they may not possess the funds or time required to get the eye tracker, their eyes could get tired while using it, and they may not have a computer that can sustain this technology.

3.1.4 Standards

Pupil Centre Corneal Reflection (PCCR), standard for gathering eye data in all screen based eye trackers. [3]

3.1.5 Priority

Critical

3.2 Changing Size of Letters

3.2.1 Description

After selecting certain letters, the letters size will change based on the selected letter. This feature will create an easier time for the user to create words that are commonly used, due to there being an easier time looking at the letters they want to type.

3.2.2 Source

CSE Design Specification and Sponsor

3.2.3 Constraints

The eye-tracking hardware may be malfunctioning and it could be difficult trying to balance the sizes of the letters for all 26 letters of the alphabet.

3.2.4 Standards

Standard text size is between 8 and 12 points, meaning, all letters that may not be used should be smaller than 8 points and all letters that will probably be used should be larger than 12 points. [2]

3.2.5 Priority

High

3.3 Focus Capturing the Letters

3.3.1 Description

Capturing letters will ensure that the letter is being registered by the user based on the tracking location. There will be multiple ways of trying to accomplish this, one being having a timer keep track of the amount of time looking at a letter and the other being where we use blinks as clicks.

3.3.2 SOURCE

CSE Design Specification and Sponsor

3.3.3 CONSTRAINTS

The eye-tracking hardware may be malfunctioning, it can be difficult trying to add multiple different ways of clicking, and it could be difficult adding in blink time; due to people having dry eyes.

3.3.4 STANDARDS

An average person can go 1-3 minutes without blinking. So, setting the stair duration to around 5-10 seconds could allow someone to get 18-36 letters in before needing to blink. [1]

3.3.5 PRIORITY

Critical

4 Packaging Requirements

Our final package will include: a precompiled executable for popular operating systems or script files that run on its runtime environment; and source code with a make file or equivalent, and instructions on how to compile and run. It may also include a tutorial if it is needed to explain how to use the software.

4.1 Executable file and source code

4.1.1 Description

The pre-compiled executable that directly runs our main program on Windows 10 and/or Linux 4. The format will be .exe on Windows and ELF on Linux. Source code will come with a make file or equivalent and a readme file for usage instructions.

4.1.2 Source

Team

4.1.3 Constraints

Our program may only support limited versions of operating systems and require a basic compiler or runtime environment.

4.1.4 Standards

Not Necessary

4.1.5 Priority

Critical

4.2 JavaScript and support files

4.2.1 Description

Our program may run on Node.js with Electron framework. The program will be in a folder with all files needed.

4.2.2 Source

Team

4.2.3 Constraints

Requires runtime environment npm.

4.2.4 Standards

Not Necessary

4.2.5 Priority

Critical

4.3 Tutorial

4.3.1 Description

A tutorial document explaining how to install and use our program. It will be uploaded to our website or a GitHub repo.

4.3.2 Source

Team

4.3.3 CONSTRAINTS

Requires download from the Internet.

4.3.4 STANDARDS

Not Necessary

4.3.5 PRIORITY

Future

5 Performance Requirements

This section gives an overview on the performance between the equipment and the software. The equipment needs to work together to help create and improve the product. We will discuss how the Eye Tracker will be collaborating with the users machine.

5.1 Collaborating Eye-Tracker

5.1.1 Description

The eye-tracker must be connected to the computer that needs the product. Collaborating the eye-tracker will ensure the eye-tracker is connected to the users machine.

5.1.2 Source

CSE Design Specifications and Team

5.1.3 Constraints

Eye-Tracker may be malfunctioning, or the users machine could be malfunctioning.

5.1.4 Standards

Not Necessary

5.1.5 Priority

Critical

6 Safety Requirements

It is crucial that our product can be operated without risking the users safety. This involves not requiring any motion input from the user while using the app. The app must also be able to safely take the users gaze info without causing them distress or discomfort in their eyes.

6.1 Gaze-driven user interface

6.1.1 Description

The eye-tracking must be functional in the absence of mechanical input from the user. This mitigates the risk of the user injuring themselves while attempting to maneuver the app.

6.1.2 Source

CSE Design Specifications and Team

6.1.3 Constraints

The software may require non-gaze input to configure initial settings. However these configuration and installment procedures are expected to be done once, during installation of the software.

6.1.4 Standards

Pupil Centre Corneal Reflection (PCCR), standard for gathering eye data in all screen based eye trackers.

6.1.5 Priority

Critical

7 Maintenance & Support Requirements

This section gives an overview on errors that may have occurred. We will be discussing inaccurate capture and bug fixes, which are the events that could cause us to perform maintenance on our application.

7.1 Inaccurate Capture

7.1.1 Description

The eye-tracker may not capture the correct letters or symbols. Re-collaborating the eye-tracker to the machine may be required.

7.1.2 Source

Team

7.1.3 Constraints

Machine may be malfunctioning (Restart may be required) FIles may be corrupted

7.1.4 Standards

Not Necessary

7.1.5 Priority

High

7.2 Bugs

7.2.1 Description

Every code will have bugs, no matter how well you can program. Most of the time, bugs are minor issues that you notice during testing, things like a button not working or a value not being correctly changed.

7.2.2 Source

Team

7.2.3 Constraints

Button may not be working correctly Value may not be correctly changed

7.2.4 Standards

Not Necessary

7.2.5 Priority

Moderate

8 Other Requirements

Beyond the minimum requirements listed above, our product will also require cloud infrastructure to facilitate CRUD operations.

8.1 Cloud Infrastructure

8.1.1 Description

The app will require cloud infrastructure, in the form of Amazon Web Services or Heroku, to host our web API, database, and any other REST related procedures associated with the app.

8.1.2 Source

CSE Design Specifications and Team

8.1.3 Constraints

This requirement is open-ended. We may decide to opt for a serverless infrastructure in the future, for the sake of productivity and limit scope in this project.

8.1.4 Standards

The industry standard is running scalable applications on AWS. For the purpose of this project, we may be able to host our API routes via AWS Lambda as opposed to renting an entire server instance.

8.1.5 Priority

Low

9 Future Items

The only requirement that is not necessary, but will help if implemented in the future would be the Tutorial. Using a tutorial would be a great help to whoever uses the eye tracking app, but is not necessary during this portion of the project.

9.1 Tutorial

9.1.1 Description

A tutorial document explaining how to install and use our program. It will be uploaded to our website or a GitHub repo.

9.1.2 Source

Team

9.1.3 Constraints

Requires download from the Internet.

9.1.4 Standards

Not Necessary

9.1.5 Priority

Future

REFERENCES

[1] How long can a person go without blinking?

[2] How do tobii eye trackers work? - learn more with tobii pro, Aug 2015.

[3] Claudia, By, and Claudia. The basics of font size, Sep 2019.