UBraille

Braille Learning Application

Team Members
Edgar Pineda (Computer Science)
Hyunmin Cho (Computer Science)

Advisors
Cecilia Feeley
Hao Tang
Katherine Olives
Zhigang Zhu

The City College of New York

Lighthouse Guild

Goodwill NYNJ

Table of Contents

Abstract

According to the World Health Organization there are currently estimated to be about 285 million people that are visually impaired. People that are considered to be low vision have a difficult time doing everyday activities. The everyday activity that UBraille wants to assist on for low vision users is reading. There are multiple different types of low vision that exist and UBraille would be most effective helping people that are impacted by central vision loss, peripheral vision loss and blurry or hazy vision. UBraille would also be beneficial to the people that would eventually become blind due to a disease such as glaucoma which is detectable years in advance.

UBraille is a braille learning mobile application with a design for individuals that are considered to be low vision. Our application is meant to be used as a tool to learn braille to prepare individuals that would need braille in their future due to incoming blindness. With UBraille's flashcard, trial, dictation, and translation feature we hope individuals would learn enough braille to prepare them for potential future blindness.The flashcard feature is meant for users to start getting familiar with the braille grid and markings. Using our trial feature users will be able to interact with the braille grid and place their own markings to see what English letter it would form. Our dictation feature is for users to start forming full words and eventually sentences with enough practice in braille. The final feature UBraille offers is the ability to translate an image of braille to English and would be read to the user using text-to-speech. With all our features we hope to be used as a tool to learn braille to prepare low vision users for possible future blindness. Also, any other user that would want to learn braille just for their personal use.

Background

According to the World Health Organization there are currently estimated to be about 285 million people that are visually impaired. Visually impaired or low vision complicates everyday activities that one would need to do such as reading, driving, recognizing people's faces, unable to distinguish between different colors, and lack of ability to see television or computer screens. Reading is an important activity that everyone needs, including low vision people to understand important documents. There are different types of low vision central vision loss, peripheral vision loss, night blindness, and blurry or hazy vision. The types of low vision that would reduce the ability to read would be central vision loss, peripheral vision loss, and blurry or hazy vision. Glaucoma and age-related macular degeneration would be the causes of low vision that would reduce the ability to read.

Statement of the Problem

Low vision eventually can lead to blindness which is only a growing problem. According to the National Institutes of Health, the amount of visual impairment and blindness are only going to double in 2050. It is projected that around 8 million people in the United States alone are visually impaired or blind. Having the ability to read is important for everyone and it would be difficult for low vision people.

Learning braille would regain the ability for low vision people to read important documents or just read in general. In a report by the National Federation of Blindness, they explain that there is a teacher crisis. In 2003 there were only about 6,700 full-time teachers for blind students to teach them braille, and these teachers were only for blind students. It would be

difficult for a low vision person to find a braille teacher. While there are applications in the market to learn braille they do not have the needs of low vision users.

Rationale

Low vision people require certain needs for them to effectively use UBraille as a tool to learn braille. With the features, we hope that users will be able to learn some braille to prepare them for future blindness or help low vision people with understanding braille for ease of mind for reading those important documents using braille. Our design is meant to be used by low vision users that may currently be struggling with reading.

For users that would have some sort of field of vision lost either central field loss or peripheral field loss, we would include text-to-speech for these users. In central field loss, people are unable to see from the center of their eye by having a dark spot on it. Every person that faces this issue has different sizes of the black spot. For users that have peripheral field loss, they can only see from the center of their eye and everything else around it is dark which would not allow them to read properly. To accommodate both these types of users we would use text-to-speech where possible in our application to limit the amount of reading that might be needed in our application.

Low vision users also require the user interface to have certain font, coloring, and layout. With UBraille we focused our user interface with these needs that low vision users require. Some low vision users require a high contrast level which we tried to obtain with our black and orange colors. Font size is also important for low vision users some require a bigger font size while others smaller font size. Our application displays a bigger font size while using text-to-speech to accommodate the ones that might need a smaller font size.

Design and Development

Technology Stack

Figure 1. UBraille Tech Stack

We choose to use React Native through expo cli to build our user interface since it creates native applications for both iOS and Android platforms. Having a single code base benefits us so our application can be accessed on both platforms. Since UBraille needs a login system to keep track of the level the user is on for our dictation feature we went with Firebase for authentication instead of building out a custom login backend using nodeJS. All the braille information is gathered from our RESTful API that was written in javascript using nodeJS which is currently hosted on google cloud run. When a request gets sent to the nodeJS for braille information the server fetches the data from our PostgreSQL database which is being hosted on google cloud SQL server.

Introduction Section

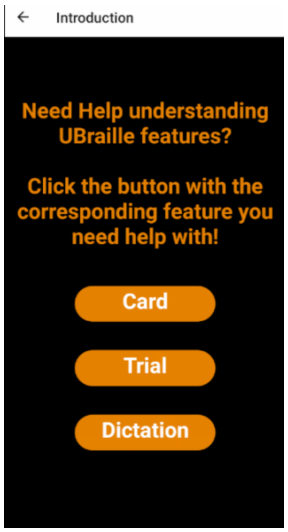

Figure 2: Introduction

The introduction section is for users that might find our other features confusing or want to get a better understanding of the features we offer on UBraille. The users can come to the introduction section to get a better understanding of any feature they might need a better understanding. When a user clicks on the button displayed in figure 2, the introduction section, a popup would appear giving instructions on how to use the feature. The text is displayed in large font for our low vision users. Also, when a user clicks a button the pop-up would use text-to-speech to read the pop-up to the user. Using text-to-speech for this section allows our low vision users that would have a difficult time reading the pop-up to get the necessary help that they might need to use our features. Implementation of this section was recommended by users that were having difficulties understanding the other features UBraille offers. As long as the user is logged in the introduction section would be available to them through our menu which can only be seen when a user is logged in.

Flashcards Feature

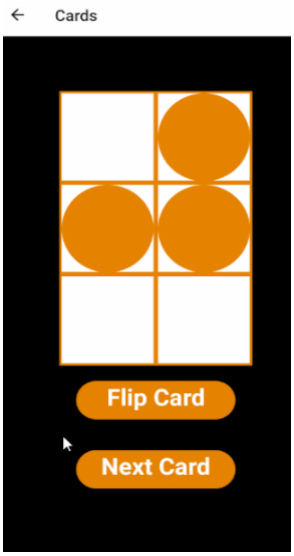

Figure 3.1: FlashCard feature(1)

UBraille has a flashcard feature. The goal of this feature is for users to start getting familiar with the braille grid and markings. The interface is designed with our low vision users in mind trying to obtain a high contrast level that would be needed for those users. To achieve a high contrast level we choose a black background. With the background being black we needed our grid to stand out for users so we chose a white background for the braille grid as seen in figure 3.1. The markings and buttons are orange to help achieve this high contrast level that is needed for the users.

In this feature, a user will see a random braille grid with its markings that form an English letter. This data is obtained from our nodeJS API fetching it from the PostgreSQL server where our braille information is stored. As seen in figure 3.1 there are two buttons at the bottom of the braille grid “Flip Card” and “Next Card”. These buttons allow users to go at their own pace and will be able to click these buttons when they feel ready.

Figure 3.2: FlashCard feature(2)

The “Flip Card” button seen in figure 3.1 is meant to be clicked by a user when they are ready to see the English corresponding letter of the given braille grid. As seen in figure 3.2 the English letter font size is close to the full size of the card for our low vision users. Making the letter visible for the low vision users would help them see the letter. Also, when the button is flipped the letter is being read to them using text-to-speech to ensure that the user can determine what letter the given braille grid forms. Using text-to-speech and big font size the goal is to start getting users familiar with the braille grid and braille markings.

The “Next Card” button seen in figure 3.1 is meant to be clicked when a user is ready to move onto the next braille grid. We want the user to cycle through different letters at their own pace eventually going through all the letters in the English alphabet. The selection of the next grid is randomly given to the user. The application saves the letter the user was currently seeing before clicking the button so that the user does not get a repeated card. Users will always have access to the flashcard feature while they are logged in.

Trial Feature

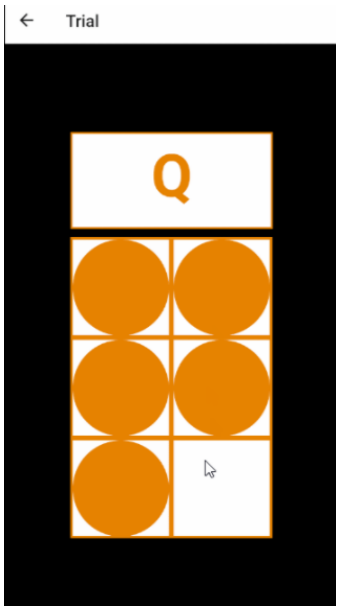

Figure 4: Trial feature

UBraille also offers a trial feature that is meant to be used as a practice to start getting familiar with forming English letters with the grid. A similar grid is shown in the trial feature as seen in figure 4 to the grid shown in figure 3.1. We want users to see a constant braille grid and not have it change with each feature we offer. The only difference between figure 4 and 3.1 is that in figure 4 the grid is touchable. Meaning that when one of the squares is touched by the user it would either place a braille marking if there is not a mark and if a mark is already placed there it would remove that mark.

In figure 4 the trial feature you could see a white box with the letter “Q”. The letter that is displayed there is from the placed markings in figure 4. As long as the grid forms an English letter that letter would be displayed in the white box. However, if the user makes places markings that do not make an English letter the word “invalid” would appear on the white box. With our low vision users in mind, we wanted the letter or the word “invalid” to appear visible to those users with the font being large enough to not take away the attention of the grid. Also, we

plan to add text-to-speech to read what is being displayed on the white box. Adding text-to-speech would benefit users that are considered low vision.

Dictation Feature

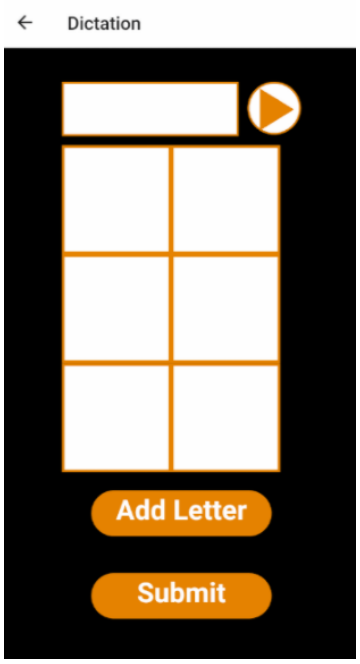

Figure 5: Dictation feature

UBraille's dictation feature is for users to start forming full words with the braille knowledge they have gained from the previous two features flashcard and trial. In this feature, the user first needs to click on the play button as seen in figure 5. When the button is clicked the user will hear a word being read to them using text-to-speech. Having this option instead of displaying the word they need to type would benefit low vision users due to the feature already having so much going on.

After listening to the word the user will use the touchable braille grid to spell out the word that they were given. The touchable braille grid can be seen in figure 5 which is extremely similar to the grid in figure 4. One difference in this feature is that the user will not be able to see the letter the grid forms until they tapped on the "Add letter" button. Since there is only one

braille grid users are only able to add one letter at a time to the input box which can be seen in figure 5 right above the braille grid. The user is also unable to remove letters once added to the input box.

There is also another button at the bottom of the braille grid which is the “Submit” button. Once a user clicks this button it would compare the user input with the desired input a pop-up would appear. Two different pop-ups could appear for the user both of the pop-ups are read to the user using text-to-speech. Using text-to-speech would benefit low vision users in case they have a difficult time reading the pop-up. One of the possible pop-ups that could be displayed would congratulate the user on obtaining the correct answer. With this pop up the user will move on to the next level. The other pop-up that could be displayed to the user would notify them that the input is incorrect and they would have to try again. In the case of this pop up the user will remain in the current level they are on.

As mentioned before there are multiple different levels UBraille offers in our dictation feature. With each successfully completed level the following level would ask to input a more difficult word. Successful completion of all the levels that UBraille offers in this feature would benefit users in being able to write any word in braille.

Translation Feature

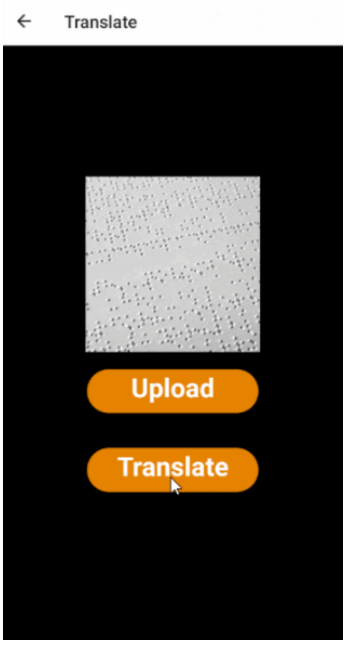

Figure 6: Translation feature

UBraille's final feature is the translation feature that came after we gathered feedback from users saying they would like a feature to incorporate some braille touching since braille is a touch language. With this feature, we hope that users get reliable translations from braille to English while being able to touch the braille. In this feature, users will first see the “Upload” button a user will be able to upload an image of braille from their camera roll. After selecting the image that image would appear as a preview above the “Upload” button as seen in figure 6. Also, another button will appear which is the “Translate” button. This “Translate” button the user would need to tap to receive their translation. The result from the translation would be read to them using text-to-speech. Throughout all the features that UBraille offers text-to-speech is an important tool for our low vision users.

Evaluation

Throughout the development process of UBraille, we send out multiple surveys and let users test our application. The feedback that was gathered helped us with the user interface being more low vision friendly. There were multiple updates we did to the user interface due to the feedback we got. The first major user interface update would be in the flashcard feature; the displayed English letter when the card was flipped was too slow for some low vision users. We updated this to make the font size fit the whole card and also made the letter bold so it could be more visible for low vision users. Another major update on our user interface would be the trial feature where the feedback is given was that the English letter was not that visible for all users. The solution for this is to add a white box background that the letter would go inside for the letter can be more visible. Also, changing the font size and making the font bold would benefit the low vision users.

Also, we got great feedback from the surveys mentioning another feature we could have added which is the translation feature. Since braille is a touch language rather than an actual verbal language this translation feature would incorporate some touch to our application. At first, the resulting translation would appear in a pop-up and the user would have to read it. We gathered more feedback from this implementation and got negative feedback regarding it. Most of the feedback was that the font was too small for all users. The solution of this would be to use text-to-speech instead of having a pop-up where the users would have to read. All user feedback from testers of our application and surveys that were conducted was very helpful to understand the needs and wants of our low vision users.

Marketing

UBraille will be made available both on the iOS app store and Google play store. Both the iOS app store and Google play store require a fee to publish an application on their platform. The fee for the iOS app store is $99 which is an annual fee and for the google play store is a one-time payment of $25. Currently, both the backend API and PostgreSQL server are on Google Cloud Platform. The pricing for the PostgreSQL SQL cloud server hosted on Google cloud platform is currently at $0.25 a day which is about a $100 yearly fee for this server. This SQL cloud server is the cheapest option on Google cloud platform and will have to be scaled up when more users use our application. The backend API it is being hosted on google cloud run and the cost of the server is estimated to be about $1,500 yearly fee. The final cost factor of our application would be authentication through firebase. UBraille is currently using the free 10k authentication request we have per month, however, when more users begin to use the application it would be upgraded to the pay-as-you-go plan offered by firebase. The total yearly cost of UBraille is estimated to be about $1,700.

UBraille's revenue plan is to do the following: accept donations, pay per translation, one-time payment for more dictation levels, and advertising done through our application. Translation of an image would have a high price cost if we choose to go with an external API for this feature which is the reason for the pay per translation. Each translation was estimated to cost the user about $1.00 for four translations. We could offer a user a few translations to try out our translation feature, however, we would only offer around 10-20 free translations per account. A user will be able to purchase more dictation levels for a one-time payment of $1.99 to continue learning braille. UBraille will also accept donations for the development of the translation feature. Eventually, once we gathered enough donations we could offer unlimited free

translations for all users. The final revenue stream could be displaying different advertising on our application and this would go for keeping the servers up and running.

Branding

Figure 7: UBraille Logo

For our branding, we wanted to try to obtain a simple look with a high contrast level for our low vision users. To obtain this high contrast level we first choose a background color that would help us with black contrast. Afterward, we needed the contrast level to be high orange helped us obtain that high level while making the text pop for the users. Having the word braille in "UBraille" would help the user distinguish that we are a braille application. On the lower right-hand side of the text, there are braille markings that spell out the word "braille". The braille markings are located at the bottom right side of the logo to let users see an example of braille.

References

1. "Low Vision." *National Eye Institute*, U.S. Department of Health and Human Services, www.nei.nih.gov/learn-about-eye-health/eye-conditions-and-diseases/low-vision.
2. "Global Data on Visual Impairment." *World Health Organization*, World Health Organization, 8 Dec. 2017, www.who.int/blindness/publications/globaldata/en/.

3. *Accessibility Requirements for People with Low Vision*, www.w3.org/TR/low-vision-needs/.
4. "Fast Facts of Common Eye Disorders." *Centers for Disease Control and Prevention*, Centers for Disease Control and Prevention, 9 June 2020, www.cdc.gov/visionhealth/basics/ced/fastfacts.htm.
5. "Visual Impairment, Blindness Cases in U.S. Expected to Double by 2050." *National Institutes of Health*, U.S. Department of Health and Human Services, 19 May 2016, www.nih.gov/news-events/news-releases/visual-impairment-blindness-cases-us-expected-double-2050.
6. *The Braille Literacy Crisis in America Facing the Truth, Reversing the Trend, Empowering the Blind*. The National Federation of the Blind Jernigan Institute, 29 Mar. 2009, nfb.org/images/nfb/documents/pdf/braille_literacy_report_web.pdf.

Acknowledgments

Professor Zhigang Zhu, Department of Computer Science

Gerardo A Blumenkrantz, Associate Professor, Branding + Integrated Communications (BIC) Creative Track Director

Katherine Olives, Program Manager, Zahn Innovation Center

Contributions

Edgar Pineda

User Interface (Frontend)

API development (Backend)

Weekly logs

Presentations

Surveys

Report

Video

Hyunmin Cho