# EyeSpy: A Mobile Vision Acuity Testing Application

Monishka Gautam[301470355], Bea MacDonald[301576522], Krish Mann[301565069], Leonardo Mormontoy[301573101], and Towhid Murad[301449367]

{mga120, bma89, kma133, lem6, tma66}@sfu.ca
CMPT340 Fall 2024, Prof. Hamarneh

**Abstract.** In this report, we will discuss the creation of a new mobile phone application capable of performing a simple vision acuity test to measure the severity of the condition Myopia, or rather, the ability to measure one's nearsightedness. Utilizing React Native, Apple's TrueDepth camera array, and onboard speech recognition APIs, we created a self-service eye test based on the Snellen chart that accurately scales down to a handheld format and adjusts the sizes of characters displayed based on signals from captured image depth data.

Overall, we found our application was accurate to some extent, though we struggled to accurately assess individuals with more severe myopia, due to compounding factors such as insufficient screen size provided by the base-model iPhone we were using to test the application, as well as lags in receiving updated depth data, particularly when the user instinctively leans forward during harder parts of the eye test. Individuals with more mild myopia (either naturally, or simulated through wearing glasses with an insufficient prescription), were given expected test results in our limited sample size. In the future, more research could be put into increase the application accuracy through by employing the use of other eye chart standards, expanding the usability of the application to those with other phone operating systems, and increasing coverage of eye conditions we can test for by introducing a new test flow for detecting hyperopia (farsightedness).

**Keywords:** Smartphone, Computer Vision, Speech Recognition, Visual Acuity, Snellen Chart, Depth Mapping, Myopia Test

## 1 Introduction

### 1.1 Problem Description

Vision impairment is a highly prevalent condition affecting at least 2.2 billion people globally, and most people will experience some form of vision-related condition in their lifetime which would require appropriate treatment and benefit from early identification [1]. Many individuals lack access to timely and affordable eye care, leading to delayed diagnosis of common conditions such as Myopia. There is an urgent need for a cost-effective solution to help identify eyeglass prescriptions.

### 1.2 Motivation

Improve access and affordability to eye care by providing an at-home solution. Estimate eyeglass prescription to identify Myopia and enable early intervention.

### 1.3 Background

The EyeSpy app leverages technologies like React Native, Apple's TrueDepth camera, and Expo to address a significant healthcare gap. TrueDepth technology allows the measurement of the user's distance from the phone with great accuracy, while React Native enables cross-platform development for iOS and Android. We opted to use implement a custom variation on the Snellen chart created by Herman Snellen, adapted for use on digital displays (particularly those with high DPI - dots per inch - counts). The incorporation of speech recognition allows for hands-free input by the user, thereby further enhancing usability. The goal of these technologies is to combine usefulness and usability so that at-home vision screening that work well.

### 1.4 Related Work

There has been numerous research conducted in this area of medicine as researchers have explored topics such as whether visual acuity assessment through mobile applications can improve the field of telehealthcare. Particularly, a study found that the use of mobile applications by non-ophthalmic emergency department staff was more accurate than with the use of a traditional Snellen chart as the app provides [2]. Furthermore, another study created criteria to validate whether the available apps could demonstrate evidence for clinical validation, and they recommended three apps that could be integrated into Teleconsultation Services in ophthalmic and non-ophthalmic settings [3].

### 1.5 Map of the Report

The rest of the report is structured as follows: Section 2 outlines the materials used, including the phone's depth mapping and speech recognition capabilities. Section 3 delves into the methods and algorithms implemented to replicate a Snellen chart[4] in a mobile application. Section 4 presents the results of our testing, including observations and metrics on the app's performance. Section 5 highlights the accomplishments achieved during development, while Section 6 discusses challenges encountered and solutions implemented. Section 7 explicitly outlines the contributions of each team member, and Section 8 discusses future improvements to the application. The report concludes with references and acknowledgments.

## 2 Materials

The key materials used for the application was enabling real-time acquisition of depth data from the user's camera, as well as real-time speech recognition for hands-free interaction and evaluation. Depth data was collected from Apples proprietary TrueDepth camera API. By using this information to determine the user's distance from the phone, the Snellen chart letters are shown at the appropriate size according to the user's distance. Accurate vision testing requires high-resolution depth data, which the TrueDepth camera provides. Speech data was collected through native Speech Recognition APIs for IOS and Android respectively, accessed through expo-speech-recognition[5], open-source wrapper for these native endpoints.

## 3 Methods

### 3.1 Visual Acuity and Depth Integration

The visual acuity formula for a Snellen Chart depends on two variables: the distance at which the test is taken — which is conventionally 6 meters/20 feet — and the smallest row an individual can read. Conventionally, the letters on the smallest row, known as optotypes, are sized such that it subtends 5 minutes of arc

$$VisualAcuity = \frac{Readable\ distance}{Readable\ distance\ with\ normal\ acuity} \tag{1}$$

**Meaning:**

- 20/20 vision translates to the ability to clearly see at a 20ft distance what a person with normal vision can see at the same distance.
- 20/40 vision translates to the ability to clearly see at a 20ft distance what a person with normal vision can see at a 40ft distance.

Hence, the largest letter measures 20/200 in visual acuity. In our case, it varies slightly due to the nature of our project. Instead of targeting a length of 20 feet - something which would be impossible to do handheld. We automatically scale the sizes of the Optotypes to account for the user's current distance away from their phone. Because this is a linear operation, we retain the effectiveness of a longer range Snellen Chart.

To calculate the dynamic height of the Optotype in our application, we consider the current row's Snellen Fraction, defined as the distance of the test, over the length required such that the optotypes on that row match the length of the smallest row. This equation is standardized in arcminutes, defined as:

> "A unit for measuring small angles, used in geometry, surveying, mapmaking, and astronomy. An arcminute is one-sixtieth of a degree and is divided into 60 arc seconds (symbol ")" [6].

In optometry, the arcminutes work as a constant reference, where the smallest optotype will subtend 5 arcminutes.
**Examples:**

- 20/20 vision: At 20ft, the letter subtends 5 arcminutes.
- 20/40 vision: At 20ft, the letter subtends 10 arcminutes or 5 arcminutes at 40ft.

Given a fractional visual acuity for a row in the Snellen Chart, arcminutes are defined as:

$$Arcminutes = 5 \times \frac{Visual\ Acuity\ Denominator}{Visual\ Acuity\ Numerator} \tag{2}$$

**Table 1.** Snellen Chart Visual Acuity and Arcminutes

| Row in Tests | Visual Acuity | Arcminutes |
|---|---|---|
| 1 | 20/200 | 50 |
| 2 | 20/100 | 25 |
| 3 | 20/70 | 17 |
| 4 | 20/50 | 12 |
| 5 | 20/40 | 10 |
| 6 | 20/30 | 7 |
| 7 | 20/25 | 6 |
| 8 | 20/20 | 5 |

Given the current distance in meters and the arcminutes based on the test row, the height of a letter is calculated as [7]:

$$Letter\ Height\ (m) = 2 \times \texttt{currentDistance} \times \tan\left(\frac{ArcminutesInRadians}{2}\right) \tag{3}$$

In order to convert the height from meters to pixels:

```
MetersToPixels()
{
    inches = meters * 39.3701
    Return (inches * screenDPI)
}
```

DPI, or Dots Per Inch is a constant provided by the phone. This equation is calculated in real-time as the distance from the camera to the user changes (currentDistance).

### 3.2 Letter Selection

Using a constant set of letters reduces re-usability of the test due to memorizing the pattern. Because of that, we randomize the letters shown each time. The specific letters we chose are picked based on the British Standards Institution's guidance that letters like "C, D, E, F, H, K, N, P, R, U, V, Z" are all equally legible in this context.[8]

### 3.3 How a Prescription/Result is Assigned

The result is based on the user's performance in the test. The app follows these rules:

- The left eye is tested first, with the user attempting to identify 5 letters in a row. If at least 3/5 letters are correct, the user moves to the next row with smaller letters.
- If fewer than 3/5 letters are correct, the test stops, and the user's result is recorded as the previous successful row's visual acuity. For example, failing at row 3 indicates a visual acuity of 20/100.
- The same procedure applies to the right eye test.

The results are then stored and displayed on the results page, where the user is informed of their approximate prescription.

## 4 Results

Regarding the performance of our application, it performs as expected and no delays are present while calculating the distance from the end-user to their device which as the core function of this project enables the rest of the features as smooth as intended. The results of our project are better shown in the demo video due to the nature of our features; the use of camera depth( letter size adjustment done in real-time), and speech recognition are more impressive to be seen in the video, but we can briefly show images of how our program works and how these features work.

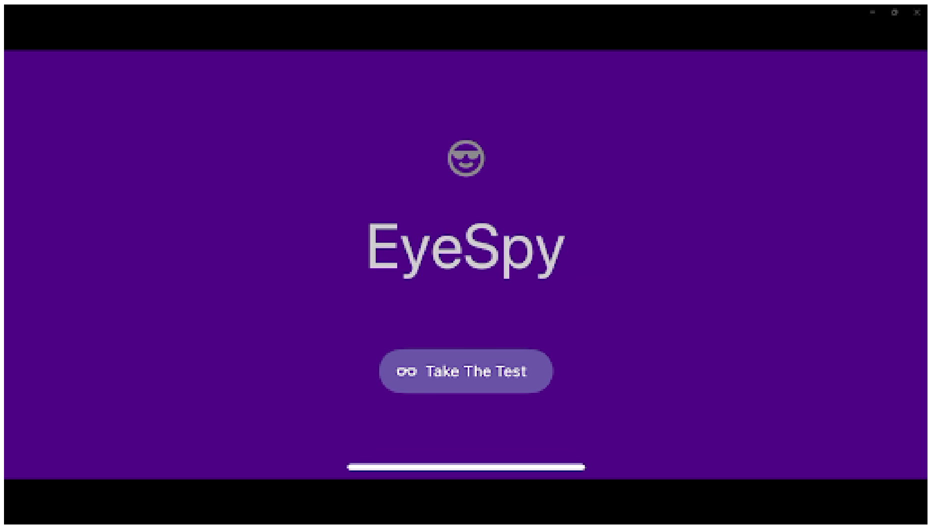
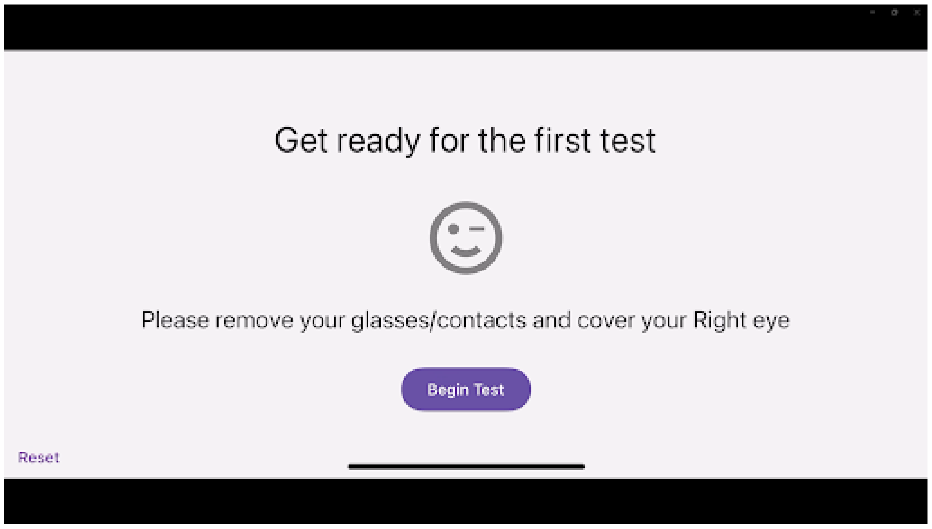


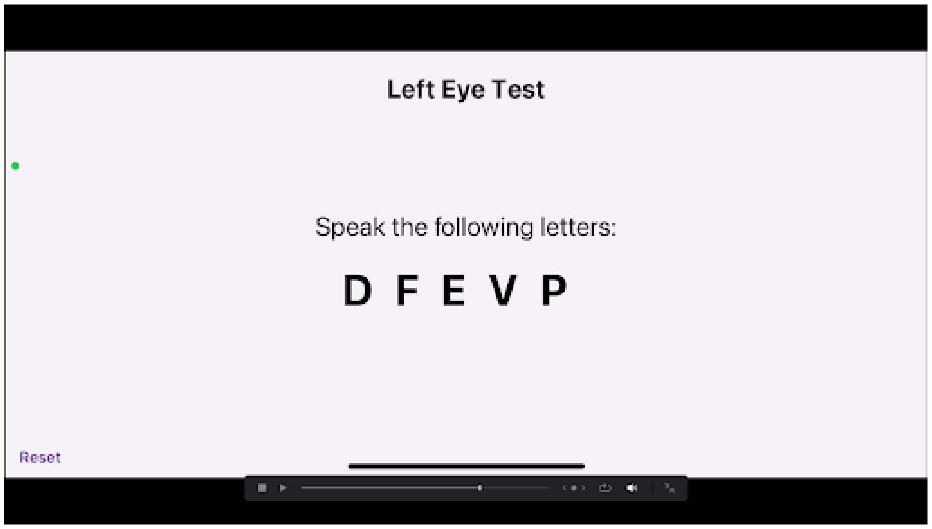
**Fig. 1.** Starting Page



Get ready for the first test

Please remove your glasses/contacts and cover your Right eye

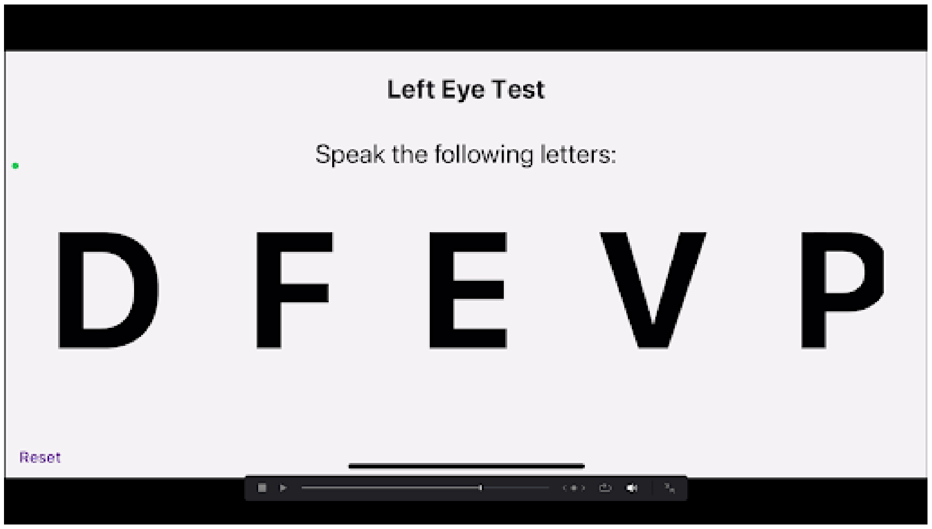
**Fig. 2.** Test Welcome page


Left Eye Test

Speak the following letters:

D F E V P

**Fig. 3.** Depth test comparison (Far)


Left Eye Test

Speak the following letters:

D F E V P

**Fig. 4.** Depth test comparison (Near)

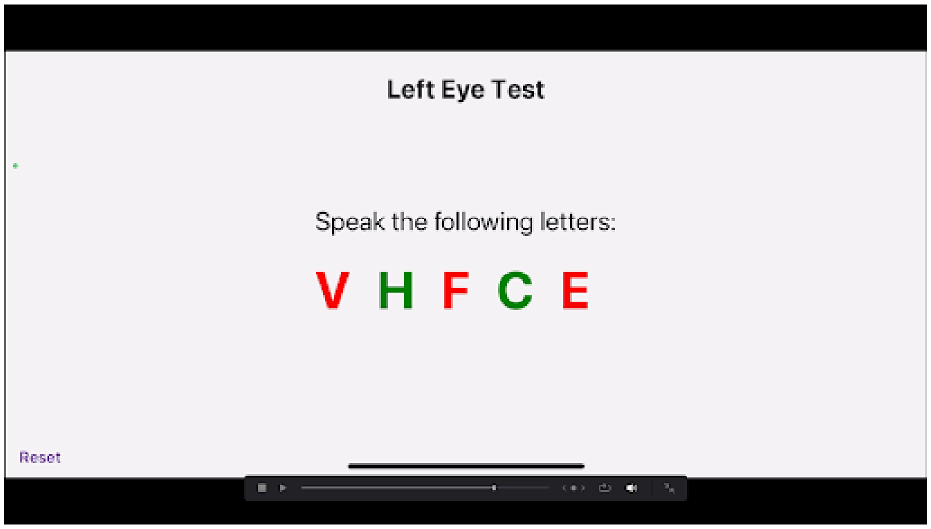


**Fig. 5.** Example of Partially Wrong 2/5 row

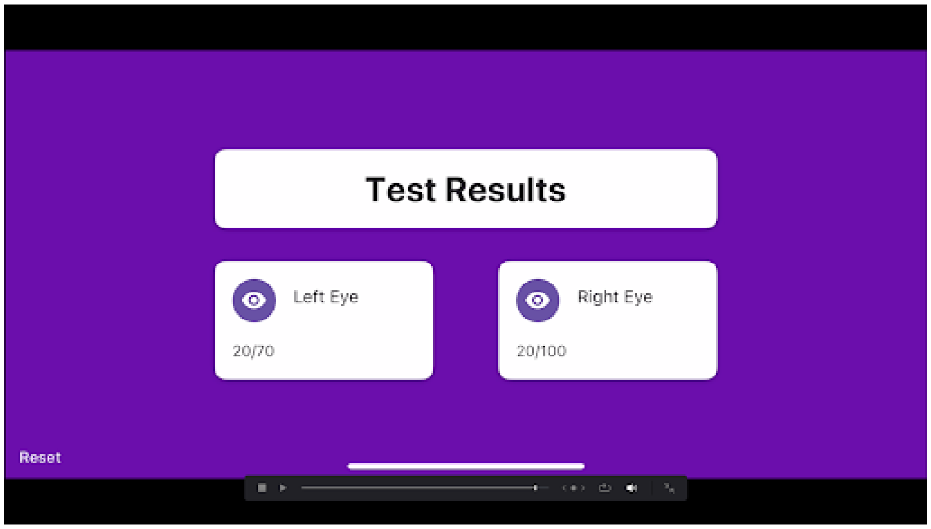


**Fig. 6.** Output of test of Left and Right test respectively

## 5 Accomplishments

### 5.1 Learnings

- Working with React Native and Expo
- Swift and mobile app development
- Writing custom plugins
- Speech recognition
- Understanding the condition and diagnosis/background

For many of the group members, this was the first time working with React Native, and we are very proud of having built this project while gaining a better understanding of the software framework. Furthermore, working with the Swift programming language and iOS development environment was a highly challenging yet rewarding experience.

In addition, familiarizing ourselves with the Expo framework to run natively on Android and IOS enabled us to build and scale our application. The topic of ophthalmology, in particular Myopia, was of interest to our team, and we have a much greater grasp of the condition and its discovery.

Importantly, our team was able to develop our programming skills and the fundamental skills of utilizing documentation and discussion forums. Moreover, we learned to manage our time and meet the deadlines. Every member in the group was able to manage and contribute effectively.

## 6 Obstacles

While working on Depth Detection there were several obstacles that were to be overcome.

- **Unavailable Depth Data:** When starting the project, no packages existed for React Native that exposed depth data from the TrueDepth camera, requiring us implement our own endpoints.
- **Issues with React Native's Frame Processor Plugin:** Incorporating Objective-C/Swift code with native React Native functionality proved challenging. We lacked clarity on how to bridge the two seamlessly.
- **Bridging Resources:** After researching, we found resources on how to bridge Objective-C/Swift code with React Native, which provided guidance for integration.
- **Adapting Pre-Existing Code:** We discovered outdated code from a previous implementation that almost worked but required significant reimplementation to extract the depth property correctly from video frames. Our eventual implementation for exposing depth data was based on this work.

While working on the voice recognition there were also several obstacles that with the support of the team were overcome.

- **Error using Library: React-native-voice** We discovered our current build tooling didn't support the modules we were utilizing and had to change to a different method.
- **IOS Error using Library: React-native-voice** After switching from our previous build tools to native builds, we found that on our Android device the voice recognition was working perfectly but after trying it in IOS it was not detecting the microphone nor asking for permissions. We decided to switch to a different speech recognition module in response to this.
- **Trouble using Library: expo-speech-recognition** We were having some trouble with the switch to the new module, but with Bea's assistance we made the Voice recognition feature work as intended.

## 7 Contributions

The idea of the project was provided by Bea MacDonald and the scope was refined as a collective by the whole team. Specific sections of the application were delegated as follows:

### 7.1 App Scaffolding

Scaffolding of the application to provide structure for the other team members to build on was created by Bea MacDonald. To simplify mobile application development as much as possible, while playing into the skillset the team had, React Native [9] powered by Typescript and Node.JS, was chosen for the application's framework. Additionally, Expo was used to simplify the process of building applications onto native hardware, something our team had no prior experience with.

### 7.2 Depth Detection and Output Capture

Depth Detection was completed by Bea MacDonald and Monishka Gautam. Using a custom fork [10] of Marc Rousavy's react-native-vision-camera module[11]. Adapted from now-outdated work Thomas Coldwell published in 2022 [12], we were able to append the video's depth map to each frame provided by the plugin, exposing that previously unavailable data in a now consumable format.

Logic for extracting the relevant segment of depth data from the Frame for purposes of determining the end-user's distance from their device's camera was completed by Bea MacDonald.

### 7.3 Snellen/Acuity Chart Display & Speech Recognition

Work on Displaying the Snellen/Acuity Chart, as well as speech recognition was completed by Krish Mann, Leo Mormontoy, and Towhid Murad. Additional technical advice and debugging support was provided by Bea MacDonald.

Speech recognition was enabled through the use of the expo-speech-recognition plugin created by GitHub user jamsch [11], which leverages native speech recognition APIs provided by Apple and Google respectively (Google/Android APIs were used for test builds due to Android's significantly more relaxed build requirements).

### 7.4 UI and UX Design

After conducting research on similar apps that are available, UI and UX flow for the application was designed by Monishka Gautam. This included creating a clear flow of the application, choosing a complementary color scheme, and designing the application for ease of use and providing clear instructions to operate the app correctly.

To simplify the creation of the user interface in React Native's rather restrictive ecosystem, we utilized the React Native Paper UI component library[8] to speed up development and create functional UIs faster.

Additional work on colours and non-functional styling was handled by Krish Mann, Leo Mormontoy, and Towhid Murad during the later stages of development. Occasional contributions were also provided by Bea MacDonald.

### 7.5 Report

Contributions to the report were made by all members, though scaffolding of the report's content was spearheaded by Monishka Guatam.

Tracking of references was completed by Monishka Gautam and Bea MacDonald

### 7.6 Demo Video

Demo video was completed by Bea MacDonald

### 7.7 Other Repo metadata

Other metadata for the application, such as the build instructions, were completed by Bea MacDonald, with thanks to Monishka Gautam who validated their accuracy on MacOS and found workarounds to OS-specific bugs that occurred.

## 8 Conclusions and Discussions

Through this project, we successfully demonstrated the potential for using mobile technology to perform Visual Acuity tests, specifically directed at Myopia. By leveraging React Native, Swift, the Expo environment, Expo Speech Recognition, and Apple's TrueDepth Camera (and their respective APIs), we developed an application capable of approximating the Snellen chart test in a handheld format. This innovation highlights the growth opportunities in healthcare industries and remote diagnostics for telehealth, offering a cost-effective, accessible solution to individuals who may not have time to access eye care services.

However, the project is not without limitations, as the smartphone screens are small, constraining the display of the Snellen chart. This introduced inaccuracies and delays in-depth data processing, both in hardware and software. These limitations underline the further refinement required for the optimization of the application.

Critically analyzing the project, the integration of depth mapping with Snellen chart scaling provided a promising framework for accurate distance measurement, and the use of the speech recognition API improved the app's usability, particularly for hands-free operation. Additionally, the bridge between iOS and React Native components demonstrated technical adaptability.

On the other hand, certain areas need improvement, such as the app's overall accuracy, specifically for severe Myopia cases, as they could benefit from adapting to a modern visual acuity test standard such as the LogMAR chart, which may perform better at various test distances. If the app is used on an iPad, it could provide a solution for better test results. Finally, improving depth data precision and exploring cross-platform compatibility (e.g., on Android) would enhance the app's usability and accessibility to a wider audience.

In summary, this project represents a promising step toward combining eyecare tests and portable technology, while also highlighting key areas for improvement and future development. With further refinement, the application has the potential to make a significant impact in vision diagnostics.

## 9 Future Work

The first major item we see value in pursuing is addressing the inaccuracies we observed in EyeSpys measurements and implementations. To address our three largest identified shortcomings: Inadequate screen space, depth reading delays and precision, and methodology, one could investigate the following solutions:

- Redesigning the application to fit the screen of an iPad tablet, in particular, one equipped with Apple's TrueDepth/Face ID technology such as the iPad Pro[13]
- Using Face Tracking to limit depth readouts to only the region of the person's face, to reduce conflicting depth data read from nearby surfaces
- Replacing the current Snellen chart implementation with the more modern LogMAR chart, which builds upon the Snellen chart in order to realize higher measurement accuracy, particularly at non-standard depths.

Supporting cross-platform capabilities with Android is another interesting proposal, though unfortunately, methods to acquire depth data from Android devices (particularly the front-facing "Selfie" camera) are not currently established or supported to the extent that they are on iPhone, making this a rather challenging objective.

Lastly, we would like to increase the scope of the app by adding new accessibility features and resources to our users. Supporting other Acuity tests such as the "Tumbling E" chart, for example, would improve accessibility to non-English speakers and children who are not familiar with the English alphabet. Taking inspiration from other publicly available apps, we believe it would be useful if other helpful resources were included right in the app, such as a map of known optometrist clinics nearby, and better descriptions for what exactly a user's test result means in terms of their everyday life. Additionally, adding support for a hyperopia test would increase the number of people who could benefit from our application's services.

## 10 References

1. World Health Organization, "Blindness and vision impairment," *World Health Organization*, Aug. 10, 2023. Available: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment
2. A. S. Pathipati, E. H. Wood, C. K. Lam, C. S. Sáles, and D. M. Moshfeghi, "Visual acuity measured with a smartphone app is more accurate than Snellen testing by emergency department providers," *Graefe's Archive for Clinical and Experimental Ophthalmology*, vol. 254, no. 6, pp. 1175–1180, Mar. 2016. doi: https://doi.org/10.1007/s00417-016-3291-4.
3. K. Kawamoto, N. Stanojcic, J.-P. O. Li, and P. B. M. Thomas, "Visual Acuity Apps for Rapid Integration in Teleconsultation Services in all Resource Settings," *Asia-Pacific Journal of Ophthalmology*, vol. Publish Ahead of Print, Feb. 2021. doi: https://doi.org/10.1097/apo.0000000000000384.
4. C. Vimont, "All about the eye chart," American Academy of Ophthalmology, https://www.aao.org/eye-health/tips-prevention/eye-chart-facts-history (accessed Nov. 30, 2024).
5. Jamsch, "Jamsch/Expo-speech-recognition: Speech recognition for REACT NATIVE EXPO projects," *GitHub*. Available: https://github.com/jamsch/expo-speech-recognition (accessed Nov. 29, 2024).
6. Oxford Reference, "Arc minute," *Oxford Reference*. Available: https://www.oxfordreference.com/display/10 9780192806314-e-106 (accessed Nov. 28, 2024).
7. G. L. Howett, "Wayback Machine," *Archive.org*, 2017. Available: https://web.archive.org/web/20171211161 C13-ff8dc22d75e66f29ebdb2bb2085ee683/pdf/GOVPUB-C13-ff8dc22d75e66f29ebdb2bb2085ee683.pdf (accessed Nov. 30, 2024).
8. D. Hirasuna, "International Eye Charts: The Better to See You," *Atissue-journal.com*, Jun. 09, 2019. Available: https://atissuejournal.com/2019/06/03/international-eye-charts-the-better-to-see-you/ (accessed Nov. 27, 2024).
9. Wynne.Tr, "Native Modules: Export Objective C Modules to React Native," *Medium*, Jun. 03, 2022. Available: https://betterprogramming.pub/native-modules-objective-c-to-react-native-for-beginners-9ec49ac5ca65 (accessed Nov. 30, 2024).
10. B. Donaldson, "Q-Bea/react-native-vision-camera at take2," GitHub, https://github.com/Q-Bea/react-native-vision-camera/tree/take2 (accessed Nov. 29, 2024).

11. M. Rousavy, "VisionCamera Documentation," *react-native-vision-camera.com.* Available: https://react-native-vision-camera.com/
12. T. Coldwell, "Thomas-Coldwell/react-native-Skia-video: Video encoding/decoding support for react native SKIA," *GitHub.* Available: https://github.com/thomas-coldwell/react-native-skia-video (accessed Nov. 28, 2024).
13. "AVFoundation — Apple Developer Documentation," *developer.apple.com.* Available: https://developer.apple.com/documentation/avfoundation
14. Zafar7645, "GitHub - Zafar7645/EyeSuggest: The B.E Major Project: Measure Visual Acuity Using a Mobile Application," *GitHub*, 2022. Available: https://github.com/Zafar7645/EyeSuggest (accessed Nov. 30, 2024).
15. M. Babu, A. Bhaskaran, B. Abhilash, N. Sudhakar, and V. Dixitha, "Comparison of smartphone application-based visual acuity with traditional visual acuity chart for use in tele-ophthalmology," *Taiwan Journal of Ophthalmology*, vol. 0, no. 0, p. 0, 2022. doi: https://doi.org/10.4103/tjo.tjo$_{72}$2.*L.Suoetal.*, \*TheUseofMobileAppsforV*
*ASystematicReviewwithAMeta*−*Analysis*(*Preprint*)," *JMIR mHealth and uHealth*, *Dec.*2020.*doi* : *https* : //*doi.org*/10.2196/26275.

16. A. S. Pathipati, E. H. Wood, C. K. Lam, C. S. Sáles, and D. M. Moshfeghi, "Visual acuity measured with a smartphone app is more accurate than Snellen testing by emergency department providers," *Graefe's Archive for Clinical and Experimental Ophthalmology*, vol. 254, no. 6, pp. 1175–1180, Mar. 2016. doi: https://doi.org/10.1007/s00417-016-3291-4.