**Description of Reconnect:**

An interactive platform that assists individuals with impaired hearing to reconnect with communities and reintegrate into the society

**Inspiration**

As medical treatment for impaired hearing continues to improve, there is an increase in demand for speech therapy programs. As of now, there is an existing gap in the market for innovative platforms that facilitate self-initiated learning or relearning of communication skills. *Reconnect* assists individuals with speech issues with the recovery process through its progressive framework. Users would relearn simple sentences that include self-introduction and request for help before moving onto complex sentences that pertain to buying groceries, etc. The entire process will be supported by real-time feedback on their pronunciation, the relative speed of their vowel enunciation, and the duration of the breaks they have between clauses and sentences.

**What it does**

*Reconnect* first allows the user to listen to a sound file that contains a sentence which the user should learn as part of its progressive relearning process. The user would repeat after the sound file before *Reconnect* begins to evaluate the user’s speech.

*Reconnect* uses Microsoft Azure’s Speech-to-Text function to convert the user’s speech input into text. By comparing the text against the sentence provided to the user, *Reconnect* is able to determine if the user’s pronunciation is adequately correct. Following which, Azure’s Text-to-Speech function is used to generate a separate speech output from the same sentence. These two .wav files will then be processed by *Reconnect*.

*Reconnect* uses the SciPy library to convert the sound files into audio data chunks. By using our self-developed algorithms to process the audio data’s amplitude, frequency, and breaks, *Reconnect* is able to determine the relative speed of vowel enunciation, and the presence of unnaturally long or short breaks between words and sentences.

Finally, *Reconnect* will compile all of these feedback before presenting them to the user. The user will then be given the opportunity to try again. The user can also type a sentence which he or she hopes to practice, and *Reconnect* will generate a sound file to facilitate the same learning process as mentioned above.

**How we built it**

We used React to develop the front-end user interface. We integrated our web-scraper and Microsoft Azure’s CV library together, with Node.js.

**Challenges we ran into**

Our team is new to JavaScript development using React.js and Node.js, and we spent a lot of time familiarizing ourselves with the different aspects of it, as we ran into various nasty bugs along the way.

We also faced a lot of difficulty in trying to make the three difficult components of our program talk to eachother. We build a front-end which took form data, a web scraper which formatted data from a URL, and a way of getting suggested alt-text for an image, but taking them all together took the support of Build Day's mentors. We finally got it working in the end and it was quite rewarding.

Accomplishments that we are proud of

We gained solid progress on the idea we brainstormed. Each member worked productively on their respective tasks, and despite this being the first hackathon for most of us, our project came together slowly, with a functional front-end and a back-end that yields satisfactory results.

What we learned

We familiarized ourselves with the general workflow of hackathons along with tools and software required in these kinds of competitions.

We definitely learned a lot regarding web development using Node and React. More importantly, we grasped how to plan the development process effectively so that each module can come together timely.

What's next for Alt-access

-Implementing new features for re-rendering the webpage, such as correcting text for dyslexia.

-Research more into specific types of color-blindness, and develop relevant contrasting options.

-Integrated screen-reader

-Chrome extension