STUDIA UNIVERSITATIS BABEŞ-BOLYAI
Engineering 67(1) 2022 DOI: 10.24193/subbeng.2022.1.23

Application for speech assistance of people with hearing disability

Mihaela Dorica Stroia, Cornel Hațiegan*, Bogdan Daniel Borcilă

Abstract. *Worldwide, there are many people with hearing impairments. Depending on its severity, hearing impairment can be solved using a more or less performant hearing device, according to one's needs, but in the case of a total loss of hearing, these devices are rather expensive and not affordable for everyone. With these situations in mind, we developed a simple application, named Talky, which can be used to assist people with hearing disabilities, mostly total hearing loss, in order to facilitate the communication process with others. Talky is simple to use and may be useful for people who have difficulty speaking.*

Keywords*: speech, disability, hearing, framework, React*

1. Introduction

Statistics show that worldwide, 5% of the global population, representing 360 million of people, suffer from a variable degree of hearing impairment [1]. In Romania, in 2018, more than 23 thousand of people were declared with total hearing loss, of which almost 2000 were children. And we are talking only about people registered with the Romanian National Association of the Deaf. Whatever the cause of the hearing loss, not being able to communicate as a human is not only frustrating but also causes a great deal of suffering [2].

Hearing device technology has undergone a significant development in the past several years. With a small device implanted in the auditory canal, a person with a total hearing loss would be able to hear [3]. However, one issue arises. This type of technology, though performant, is quite expensive and not affordable for the average person.

With the above situations in mind, we thought of finding a solution that might come in handy of people with hearing loss and not only, with speech impairment as well, a solution that can be affordable and come in handy to everyone. Considering that, nowadays, everyone has a smartphone or a computer device that supports writing and audio software, we developed an application we named Talky, one that is simple, easy to use, has portability and has a friendly interface for the user with hearing or speech impairment [4].

2. Developing Talky application using specific programming languages

Talky is thought to function on a desktop, tablet and smartphone, all of these devices allowing for writing and audio [5,7]. From the user's point of view, the application has two main parts:

- when he wants to communicate, to "talk", he uses the "text to speech" component of the application, which converts the written text to audio sound;
- when he wants to "hear" what the other person is saying, he uses the "speech to text" component, which converts the audio sound into text that the user can read.

Both components of the application were built using the React framework, which allows dividing the code in fragments that can be written and debugged independently from one another [5,6]. For the design of the application's interface, we used Bootstrap's facilities.

```
import React,{Fragment} from 'react';
import logo from "../src/components/images/Logo@2x.png";
import microphone from "../src/components/images/Microphone@2x.png";
import "./App.css";
import SpeechToText from './components/SpeechToText/SpeechToText';
import Speech from "./components/TextToSpeech/TextToSpeech"

const App=()=>{
  return(
    <Fragment>
      <div className='container-fluid mt-3 mt-sm-5'>
        <header>
          <img className='img-fluid logo' src={logo} />
          <img className='img-fluid justify-content-end micro' src={microphone} />
        </header>
        <div className='container text-center'>
          <h1 className='title '>Comunication is the key</h1>
        </div>
        <div className='container-fluid d-block d-sm-flex mt-5'>
          <SpeechToText className="justify-content-center"/>
          <Speech className="justify-content-center"/>
        </div>
      </div>
    </Fragment>
```

Figure 1. Main code of Talky application.

For programming code, we used the Visual Studio Code editor, which comes with IntelliSense for JavaScript, TypeScript, CSS and HTML, as well as debugging support for Node.js [4,7,9], languages we needed for developing Talky. The application supports the English language, but the support for additional languages is provided by extensions, available with no charge on the VS Code Marketplace.

The code for the main component of the application Talky, as depicted in figure 1, is used to import the "Text-to-speech" and "Speech-to text" code parts and the buttons the user will see on the interface. Both components will return the results to the main application.

2.1. Talky "Speech-to-text" component

For building the "Speech-to-text" component of the Talky application, we have used both JavaScript and HTML languages. Part of the JavaScript programming code for this functionality is shown in figure 2.

```
function SpeechToText() {
  const [isListening, setIsListening] = useState(false)
  const [note, setNote] = useState(null)

  useEffect(() => {
    handleListen()
  }, [isListening])

  const handleListen = () => {
    if (isListening) {
      mic.start()
      mic.onend = () => {
        console.log('continue..')
        mic.start()
      }
    } else {
      mic.stop()
      mic.onend = () => {
        console.log('Stopped Mic on Click')
      }
```

Figure 2. "Speech-to-text" JavaScript code.

The main function of this component is connected to the "Start/Stop" button function. The HTML code, as presented in figure 3, of "Speech-to-text", will make visible the button "Start/Stop" on the user's interface. When the user presses the "Start/Stop" button on the screen, the application will check whether the device's microphone is activated or not, and if not, the application will take action in that direction, preparing the microphone. After this step, the application will start recording the audio, exactly what the other person is saying, until the user presses the "Start/Stop" again.

The application will proceed to convert the sound into text. The resulted text will be shown on the device's screen to the user. The JavaScript function setIsListening will convert the recorded audio into a text array (list), and this array will be available to the user in a paragraph format [7]. The paragraph in the HTML code will not appear until the recording event is closed.

```
return (
  <Fragment>
      <div className='container-fluid text-center'>
    <div className="container mt-4">
      <div className="box col-lg-5">
        <h2 className='subtitle'>Speech-Text</h2>
        {isListening ? <span>🎙️</span> : <span>🔴🎙️</span>}
        <button onClick={handleDeleteNotes} disabled={!note}>
          Delete Note
        </button>
        <button className='start_stop' onClick={() => setIsListening(prevState => !prevState)}>
          Start/Stop
        </button>
        <p className='note'>{note}</p>
      </div>
    </div>
    </div>
    </Fragment>
)
```

Figure 3. "Speech-to-text" HTML code.

We also implemented a "Delete" button that is designed for deleting previous "conversations". While the processes of recording or converting are ongoing, the "Delete" button will be inactive.

2.2. Talky "Text-to- speech" component

For building the "Speech-to-text" component of the Talky application, we have used JavaScript and HTML languages, but both in a single function, as can be seen in figure 4. The logic for this component is the same as for the previous one.

We are using a JavaScript function that aims to play the audio of whatever word the user enters into the input HTML element [7,9]. The audio playback function of the input data is assigned to the "Speech" button. After the user writes a text, he will press the "Speech" button. We have used a value type command with the purpose of listening to any event as input the user will give and trigger the playback audio function.

This component has available, as the previous one, the “Delete” option that works similarly. When the user writes his message, the text converted in audio, he is able to delete it or correct it.

```
const Speech = () => {
  const [value, setValue] = React.useState("");
  const { speak } = useSpeechSynthesis();
  const handleDeleteNotes = () => {
    setValue(" ")
  }

  return (
    <div className="container text-center mt-4 ">
      <div className="group d-block text-center align-middle">
      <h2 className="txt-title ">Text-Speech</h2>
      <input className="input"
       type="text"
      value={value}
      placeholder="Write something"
      onChange={(e) => setValue(e.target.value)}
      />
        <button className="text-center btn" onClick={() => speak({ text: value }, )}>Speech</button>
        <button onClick={handleDeleteNotes} disabled={!value}>
            Delete Note
          </button>
      </div>
```

Figure 4. “Text-to-speech” code.

3. Talky interface and functioning

The first impact of the user with the Talky application is the one presented in figure 5. In the upper left corner, we can notice the application’s logo, designed using Adobe XD software [8].

Both “Speech-to-text” and “Text-to-speech” components appear on the user’s device, but only one can be used at a time, according to the user’s needs. If the user wants to “listen”, he will work on the left side of the application. In a similar manner, if the user wants to “talk”, he will work with the right side of the application.

As we can observe, to use Talky, one does not need special skills or knowledge, because the application is simple, easy to install on a laptop, desktop, tablet or smartphone, and is user friendly. Simply by touching a button, a person with hearing loss can be able to “communicate” with others, whether he chooses to “talk” or to “listen”.

In the following, we will explain in detail how the two components contained within the Talky application can be used.

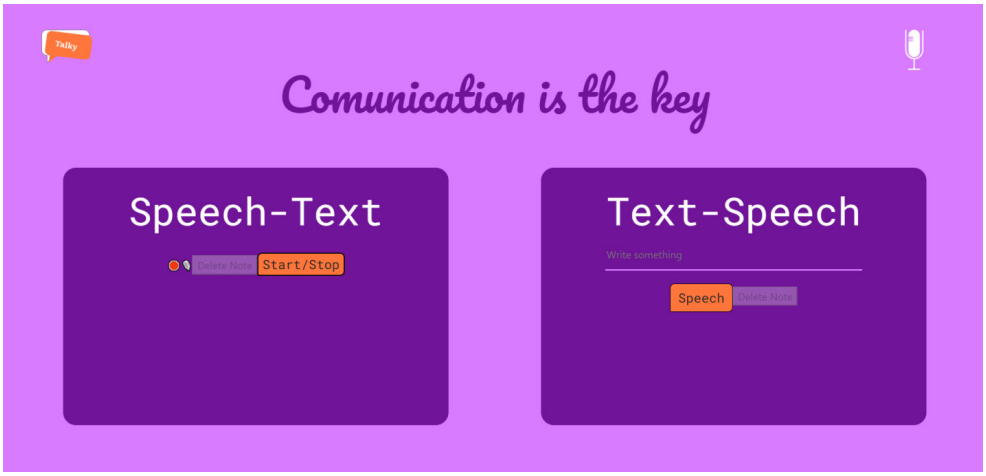

Figure 5. Talky user interface.

In figure 6, we have a better view of the "Speech-to-text" component while it is being used. When the user presses the "Start/Stop" button, the application will start recording the speaker's voice. We have added a red icon on the left side of the "Speech-to-text" that has the role of indicating the user if the component is activated or not. If the red icon is not present on the screen, it indicates the user that the recording has started and the audio conversion is in progress.

Figure 6. Speech-to-text user interface.

When the user presses the “Start/Stop” button once again, the text of the converted audio will display on the screen. In the process of recording and audio conversion, the “Delete” button is grey-colored, thus inactive. When the process stops, the “Delete” button is white-colored, thus active and the user can delete data.

As we show in figure 6, we spoke the word “Hello”, which appeared on the screen as text.

For the “Text-to-speech” component from figure 7, we have the text area where initially appears “write something”, the “Speech” button and the “Delete” button. The user writes his message in the text area and presses the “Speech” button for the text to sound conversion and audio play events to take place. The other person will be able to hear what the user has written. We can notice that the “Delete” button is active while writing a text.

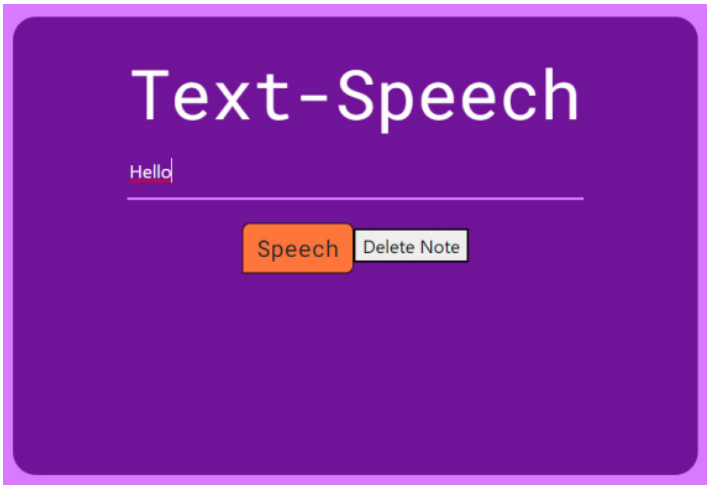

Figure 7. Speech-to-text user interface.

4. Conclusions

Hearing loss can cause plenty of suffering to a person, and there are cases where this issue cannot be solved by medical intervention. Although the technology of hearing devices has evolved so much in the past years that with a small device implanted in the ear canal, one can hear again, it is not yet affordable for the average person. We thought and developed the Talky application to come to the aid of the average person with loss of hearing. Talky is not a healing solution but a mean to help people integrate in everyday life. We have used simple technology and our programming skills to develop an affordable product.

References

[1] Lin F.R., Niparko J.K., Ferrucci L, Hearing Loss Prevalence in the United States, *Arch Intern Med.*, 20.11, 171(20): 1851–1853, doi: 10.1001/archinternmed.2011.506

[2] Newton VE, Shah SR. Improving communication with patients with a hearing impairment, *Journal of Community Eye Health*, 10(3), 2013, 26(81): 6-7. PMID: 23840079; PMCID: PMC3678307

[3] Cox R.M., Johnson J.A., Xu J., Impact of Hearing Aid Technology on Outcomes in Daily Life I: the Patients' Perspective, *Ear and Hearing-The official journal of the American Auditory Society*, 37(4), 2016, pp. 224-237, doi: 10.1097/AUD.0000000000000277.

[4] Stroia M.D., Hațiegan C., Popescu C., Bugar S.V., System design for improving our home comfort, *Annals of the „Constantin Brancusi" University of Targu Jiu, Engineering Series*, issue 2, 2021, pp. 19-23.

[5] Accomazzo A., Murray N., Lerner A., *Fullstack React: The Complete Guide to ReactJS and Friends*, Publishing House FullStack.IO, 2017.

[6] Vipul A., *ReactJS by Example - Building Modern Web Applications with React*, Publishing House Packt Publishing Limited, 2016.

[7] Lopez L., *React: QuickStart Step-By-Step Guide to Learning React JavaScript Library*, Publishing House Createspace Independent, 2017.

[8] Schwarz D., *Jump Start Adobe XD*, Publishing House Site Point, 2017.

[9] Duckett J., *HTML and CSS: Design and Build Websites*, Publishing House Wiley, 2011.

Addresses:

- Lect. Dr. Eng. Mihaela-Dorica Stroia, Babeș-Bolyai University, Faculty of Engineering, Piața Traian Vuia, nr. 1-4, 320085, Reşiţa, Romania
  mihaela.stroia@ubbcluj.ro
- Lect. Dr. Fiz. Cornel Hațiegan, Babeș-Bolyai University, Faculty of Engineering, Piața Traian Vuia, nr. 1-4, 320085, Reşiţa, Romania
  cornel.hatiegan@ubbcluj.ro
  *(*corresponding author)*
- Stud. Bogdan Daniel Borcilă, Babeș-Bolyai University, Faculty of Engineering, Piața Traian Vuia, nr. 1-4, 320085, Reşiţa, Romania
  bogdan.daniel.borcila@stud.ubbcluj.ro