**Speed Reading App**

**(Using Web Development)**

A seminar Report Submitted by

Mansi Bajaj

20MA60R01

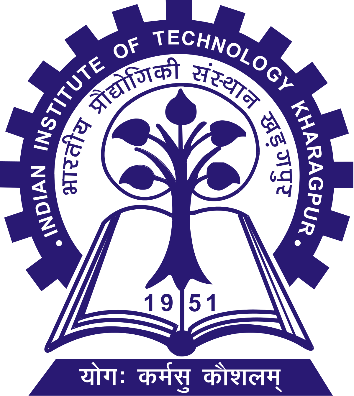
In partial fulfillment for the degree of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND DATA PROCESSING

AT



Department of Mathematics

Indian Institute of Technology, Kharagpur

West Bengal-721302, India

**UNDERTAKING**

I hereby declare that the work presented in this project report entitled **“Speed Reading App using Web Development”** issubmitted to the Department of Mathematics, Indian Institute of Technology, Kharagpur, for the partial fulfillment of the requirement of the Master of Technology degree in **“Computer Science and Data Processing”.**

I further declare that this work has not been the basis for the award of any other degree diploma or any other title elsewhere.

**Mansi Bajaj**

**20MA60R01**

M.Tech.(CSDP)

Department of Mathematics

IIT Kharagpur

**CERTIFICATE**

This is to certify that the work contained in this report entitled **Speed Reading App Using Web development** issubmitted by **Mansi Bajaj(20MA60R01)** in partial fulfillment for the award of **Master of Technology** in **Computer Science and Data Processing** of the **Indian Institute of Technology, Kharagpur.** The project report has been approved as it satisfies the academic requirements concerning the project work.

**Prof. Somesh Kumar**

**Department of Mathematics**

**Indian Institute of Technology, Kharagpur**

**INTRODUCTION**

Many instances, human beings sense crushed through the quantity of studying they must do and additionally conflict to pay attention and choose out critical points. One surprisingly powerful way to this is, Speed Reading. Speed Reading is one ofthe strategies claiming to enhance one’s capacity to examine quickly. The idea of velocity studying become notion to have shaped in past due 1950s, whilst Evelyn Wood, a faculty trainer coined the term. She believed studying may be achieved on the charge of 2700 wpm that is, 10 instances quicker than the common knowledgeable reader and similarly that the strategies allowed her to achieve this may be taught and sold. It is away that isn't always pretty much studying speedy however approximately making the complete studying manner greater efficient.

**PROBLEM STATEMENT**

The goal of my task is to layout a user-friendly studying utility using theprinciples of Web development onthewayto decorate the studying pace and performance of the reader.

My app is primarilybasedtotally atthe concept that plenty of the time wasted in studying is spent withinthe fraction of seconds becausethe eye’s attention moves among phrases and throughout the page. So, the goal is to lessen that wasted time.

**MOTIVATION**

Following are anumberof the motives that why is velocity analyzing appropriate for our mind.

* **Improved** **Memory**: The mind issortofa muscle. If we teach our brains, it'll develop morepotent and itwilllikelybe capableof carryingout better. Speed Reading demandingsituations our brains to carryout at a better level. When we teach our mind withtheintentionto absorb facts quicker, different regions of our mind mayalso enhance whichincludes your memory. Memory, while we study, acts like a stabilizer muscle that receives labored while we Speed Read.
* **Better** **Focus**: Most humans have the potential to study asaminimum twohundred wpm that's the common analyzing velocity. But a few humans alsocan study asmuchas threehundred wpm. Why is there thistypeof gap? There are two primary reasons. Number one, the conventional analyzing fashion we're taught isn't very green. The second cause is lossof recognition. If we aren't centered on what we're analyzing, our thoughts will wander and it becomes inquisitiveabout different mind. Speed Reading facilitates construct recognition.
* **Confidence**: I believe this is the result of knowing that we can learn just about any aspect of life faster once we have the ability to read faster and comprehend more. When we enhance our potential to study and study quicker, wecan discover that increasingly doorways open up for us and we beginto get extra alternatives in lifestyles. This is duetothefact each e**-**book or article, whetherornot it'sfar fiction or nonfiction, facilitates us to shift our attention and we startto see extra intensity in our lives. This new located intensity, boosts our self-confidence.
* **Improved** **Logic**: Reading is an exercising for our mind. When we teach our mind to study quicker, something great occurs. Our mind turnsinto extra green at sorting facts and locating correlations to different bits of facts formerly stored. The more we improve our reading speed, the faster this process happens and we will automatically notice improvements with logic as we get used to respond quick to what before would have taken longer to process.
* **Emotional Well Being:** Readingmaybeveryenjoyableingeneral**.** It can assist lessen strain asit receives our thoughts off issues and different mind that aren't wholesome nor beneficial. When we study quicker, wecan be even extra absorbed into the material. This causes us to focus predominantly on the information we are reading. This islikewise referredtoas active-meditation. Active meditation is a meditative state achieved by doing an activity. This state can release tension and increase emotional well-being. Speed Reading to me isn't prettymuch analyzing quicker. We can thinkabout it as an exercising. As togetheralongwithour muscles, if we exercising them regularly, wecan get morepotent. If we are exercising our mind, it'll get morepotent as well.

**WEB DEVELOPMENT**

* It refers to building, creating and maintaining the websites.
* Language’s use is dependent on the type of tasks and the platforms used.
* “It is the work done that happens behind the scenes that makes a website look great, work fast and perform well.”
* World wide Web (WWW) was the first website developed using HTML by Tim Ber Lee in 1989. It was a common source where people can share information. It is basically a collection of interlinked documents which can be viewed using web browser. The architecture it uses is Client Server Architecture.
* If the content of the website to be served to the clients by the server is not user specific, then the website is called Static website. Else, if it is user specific, it is called dynamic website.
* There are two fields of web development: Front End Development and Backend Development.

Frontend and Backend are the two most popular terms used in web development. These terms are very crucial for web development but are quite different from each other. Each side needs to communicate and operate effectively with the other as a single unit to improve the website’s functionality.

**FRONT END WEB DEVELOPMENT**

* It is the part of the website that the user interacts directly .
* It is known as the Client side of the application.
* It takes care about the layout, design and interactivity of the website.
* It includes everything that users experience directly: text colors and styles, images, graphs and tables, buttons, colors, and navigation menu. HTML, CSS, and JavaScript are the languages used for Front End development.
* By front end, we mean the presentation part of the data and not the actual data.
* Following are some of the technologies that come under the front end development:
* The structure, design, behavior, and content of everything seen on browser screens when websites, web applications, or mobile apps are opened up, is implemented by front End developers.
* Responsiveness and performance are two main objectives of the Front End. The developer must ensure that the site is responsive that is, it appears correctly on devices of all sizes. No part of the website should behave abnormally irrespective of the size of the screen.

**Front end Languages:** The front end portion is built by using some languages which are discussed below:

**HTML**

* HTML stands for Hyper Text Markup Language.
* By Hyper Text we mean, the documents that we are able to design using HTML can have links that can redirect users to some other documents or some section of the same document.
* By markup we mean, it uses markers (tags) to structure the page.

**CSS**

* CSS stands for Cascading Style Sheets.
* It is intended to simplify the process of making web pages look presentable.
* CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page.

**JAVASCRIPT**

* It is a scripting language to provide a dynamic behaviour to the website.
* It basically adds interaction to the website.
* For example: There are like, dislike, upvote, downvote buttons on any social sites. Enabling those buttons or any such kind of activity that makes the user to interact with the page is all handled by Javascript.
* It is used to enhancing the functionality of a website to running cool games and web-based software.

**Front End Frameworks and Libraries:** 

* **AngularJS:** AngularJs is a JavaScript open-source front-end framework that is mainly used to develop single-page web applications(SPAs). It is a continuously growing and expanding framework which provides better ways for developing web applications. It changes the static HTML to dynamic HTML.
* **React.js:** React is a declarative, efficient, and flexible JavaScript library for building user interfaces. ReactJS is an open-source, component-based front-end library responsible only for the view layer of the application. It is maintained by Facebook.   
  Bootstrap: Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.
* **jQuery:** jQuery is an open-source JavaScript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object Model (DOM), and JavaScript. Elaborating the terms, jQuery simplifies HTML document traversing and manipulation, browser event handling, DOM animations, Ajax interactions, and cross-browser JavaScript development.  
  When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.

The **HTML DOM** model is constructed as a tree of **Objects**:



With the object model, JavaScript gets all the power it needs to create dynamic HTML:

* JavaScript can change all the HTML elements in the page
* JavaScript can change all the HTML attributes in the page
* JavaScript can change all the CSS styles in the page
* JavaScript can remove existing HTML elements and attributes
* JavaScript can add new HTML elements and attributes
* JavaScript can react to all existing HTML events in the page
* JavaScript can create new HTML events in the page

With AJAX, we can:

* Update a web page without reloading the page
* Request data from a server - after the page has loaded
* Receive data from a server - after the page has loaded
* Send data to a server - in the background

**BACK END DEVELOPMENT**

* It is the server side of the website.
* It is the part of the website that users cannot see and interact.
* It is used to store and arrange data.
* It is the portion of software that does not come in direct contact with the users.
* The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application.
* Server side languages used are: PHP, JAVA, Python, Node.js
* For example: While logging in to some website, servers’s work is to authenticate or validate the user first and then decide what data can be shown to the user.
* Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

**WORK DONE PREVIOUSLY**

* The app was structured using HTML and the javascript code was included to enable the following functions using buttons.
* **“Process the text**” **button:** This button processes the text entered by user in the textbox to read it further. Some space is created using ‘div’ in ‘HTML’, below the buttons where the processing of the text takes place.
* **“Start Reading” button:** This button enables the reading process of the whole text entered by the user word by word by the help of highlighter.
* **“Stop Reading” button:** This button disables reading process by unhighlighting everything.
* **“Delay variable”:** This variable is set to a particular value of time say 500 milliseconds. According to this time, the highlighter is moved to the next word.
* **“Decrease Speed” button:** This button increases the value of delay variable which makes the reading process slower.
* **“Increase Speed” button:** This button decreases the value of delay variable which makes the reading process faster.
* Apart from these, some styles were added using the concept of Cascading Style Sheets.
* Following are the HTML, CSS and Javascript codes for the same:

**HTML Code** is as follows:

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="sr.css">

</head>

<body>

<h1 style="text-align: center; font-family:serif; color:blueviolet;" > Speed Reader App</h1>

<p style="font-family: cursive; font-size: 24px; color:orangered; text-align: center;">Type or paste the text to speed read.</p>

<div class='action-point' >

<textarea style="width:300px;height:50px;" id="inputText"></textarea><br>

<button type="button" id="processText" onclick="processText()">Process Text</button>

<button type="button" id="StartReading" onclick="startReading()">Start Reading</button>

<button type="button" id="StopReading" onclick="stopReading()">StopReading</button></br>

Reading Delay:&nbsp <span id="readingDelay"></span> &nbsp milliseconds<br>

<button type="button" id="decreaseSpeed"

onclick="decreaseSpeed()">Decrease Speed</button>

<button type="button" id="increaseSpeed" onclick="increaseSpeed()">Increase Speed</button></br></br>

READ YOUR TEXT:</br>

<div id="textDiv" style="width:400px; height:600; border:double; "></div>

</div>

<script type="text/javascript" src="sr.js"></script>

</body>

</html>

**CSS Code** is as follows:

.action-point{

margin:auto;

margin: auto;

width: 30%;

padding: 10px;

text-align: center;

}

body{

background-color: lightgreen;

}

**Javacsript Code** is as follows:

var delay=500;

document.getElementById("readingDelay").innerHTML=delay;

var stopped=true;

function decreaseSpeed() {

delay+=50;

document.getElementById("readingDelay").innerHTML=delay;

stopReading();

}

function increaseSpeed() {

if (delay>50) {

delay-=50;

document.getElementById("readingDelay").innerHTML=delay;

stopReading();

}

}

function stopReading() {

stopped=true;

}

function processText(){

var text= document.getElementById("inputText").value;

for(var i=0;i<text.length;i++){

text=text.replace("\n"," </br> ");

}

var textDiv=document.getElementById("textDiv");

textDiv.innerHTML="";

var wordArray=text.split(" ");

for(var i=wordArray.length-1; i>-1; i--){

if (wordArray[i]=="") {

wordArray.splice(i,1);

}

}

for(var i=0; i<wordArray.length;i++){

var span=document.createElement("span");

span.setAttribute("class", "word");

span.setAttribute("id","word"+i);

span.innerHTML=wordArray[i]+" ";

textDiv.appendChild(span);

}

document.getElementById("inputText").value="";

}

function startReading() {

stopped=false;

var textDiv=document.getElementById("textDiv");

var i=0;

var timer = setInterval(readNextWord, delay);

function readNextWord() {

if (i>0) {

unHighlightFunction(textDiv.children[i-1]);

}

if (i<textDiv.children.length && !stopPressed) {

highlightFunction(textDiv.children[i]);

i++;

}

else

{

clearInterval(timer);

}

}

}

function highlightFunction(elt) {

elt.style.backgroundColor="yellow";

elt.style.fontSize="17px"

}

function unHighlightFunction(elt) {

elt.style.backgroundColor="white";

elt.style.fontSize="16px"

}

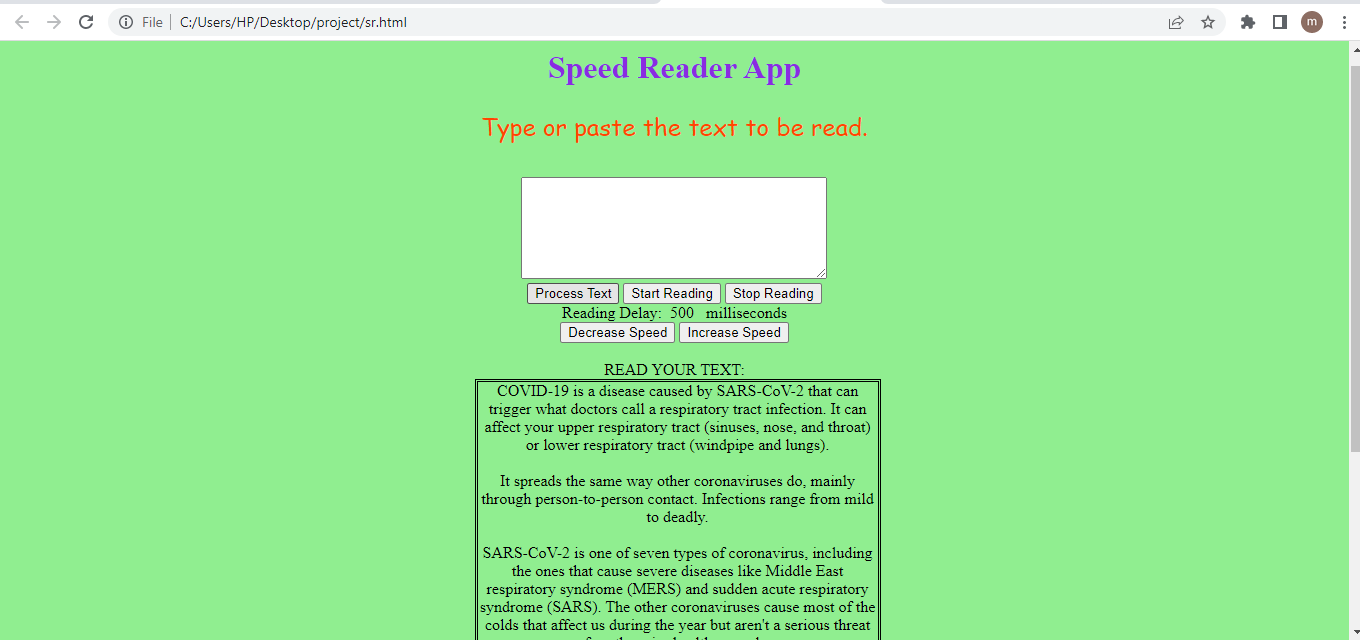
===================================================================

**OUTPUT**

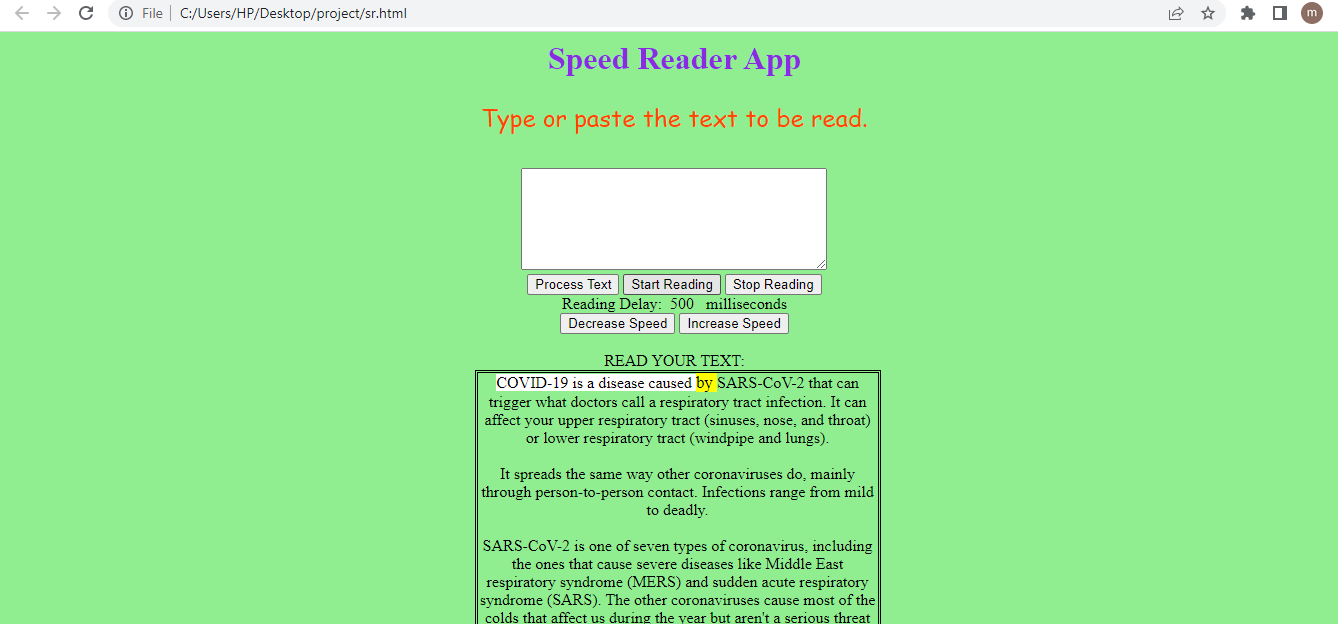
Following is the app created. It consists of a textbox for the user to enter the text to be read along with the buttons described above.

****

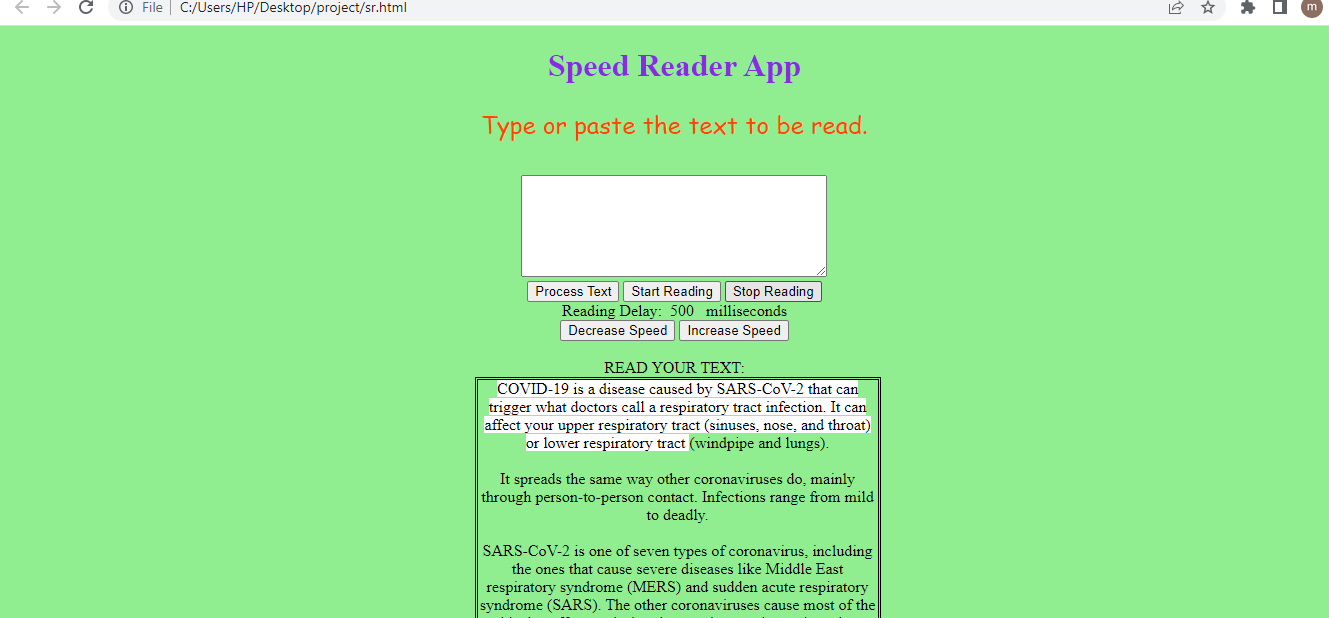
Following is the functioning of the “process text” button. It processes the entered text to the area below the buttons so that the text can be read word by word.

****

Following is the functioning of the start reading button. It starts reading the text by highlighting word by word according to the delay variable that is, after every 500 ms.

****

Following is the functioning of the “stop reading” button. It stops reading by unhighlighting the text.

****

**=====================================**

**WORK DONE NOW**

* The speed reader app is reconstructed using HTML, CSS and jQuery and jQuery Mobile.
* jQuery UI and jQuery mobile are JavaScript libraries built on top of jQuery.

To be more specific:

* jQuery is a JavaScript library meant for DOM operations.
* jQuery UI is a JavaScript library built on top of jQuery which is meant for UI operations. jQuery UI contains a rich set of UI widgets such as date picker, dialog etc.
* jQuery mobile is a JavaScript framework built on top of jQuery. Using jQuery mobile, one can build a mobile website or applications very quickly.

**UTILITIES IN THE SPEED READER APP**

**Using above, Following Features** are incorporated in the app:

The app consists of a home page that contains three options for the user:

First option allows the user to enter text in the textbox and read it word by word.

Second option allows the user to read any txt or pdf file using the url of the file.

Third option allows the user to listen the text or content of the file entered by user.

The speed reader app has been incorporated by the following features:

* **A speed controlling slider** **:** to switch between speed reading mode and the regular reading mode. It helps the user to set the speed of the words flashing in the div area according to his/her convenience.
* **A font controlling slider:** to manage the font size according to the user’s convenience. It helps the user to increase or decrease the size of the words flashing in the div area.
* **A start reading button:** This button enables the reading of the text or content of the file whose url is entered by the user.
* **A pause and a resume button:** This button allows the user to pause the reading anytime and resume it again.
* **A progress slider:** It shows the progress of the reading. Using this slider, we can go forward or backward in our reading.
* **Interface** of the app is **Minimalist**.
* **Upload file feature**: This feature is added to eliminate the constraint of limited resources. Using this feature, user is allowed to read any file in txt or pdf format using the url of the file.
* **Audio for visually impaired people:** If the user cannot read the content, this feature allows the user to listen to the content in any of the accent he/she chooses.
* In the listening section, there are two extra sliders for rate and pitch by which the user can increase or decrease the audio speed and pitch.
* In addition to this, there is an option to select an accent from various accents in which the user wants to listen the content.
* User can either enter the text in the textbox and listen to it or can enter the url of the file he/she wants to listen.

**THEORY**

**HTML functions used:**

* **Input Attributes:**
* **id:** The id attribute specifies a unique id for an HTML element. The id attribute is mostly used to point to a style in a style sheet, and by JavaScript to manipulate the id with the specified id.
* **type:** defines a single-line text input field.
* **value:** The value attribute specifies the value of an <input> element. The value attribute is used differently for different input types.
* **name:** This attribute specifies the name of an <input> element.
* **size:** The size attribute specifies the visible width in characters of an <input> element.
* **class:** The <input> tag specifies an input field where the user can enter data.

**CSS functions used:**

* **background-color:** It sets the background colour for an element.
* **background:** It sets the background for the element.
* **margin:** The margin property sets the margin for an element.
* **text-align:** This property specifies the horizontal alignment of text in an element.
* **padding:** It is the space between the content and its border.
* **margin-left:** It sets the left margin for an element.
* **width:** It sets the width of an element. The width of an element does not include padding, borders, or margins!
* **font-size:** This property sets the size of the font.
* **font-family:** It decides the font of the text.
* **font-weight:** It mentions the thickness of the text. For example: bold
* **color:** This property specifies the color of the text.

**jQuery functions used:**

* **click():** This function is an in-built function in jQuery that starts a click event. Whenever a relevant element is clicked upon, an event is fired by a click() method that is, the click() method attaches a function to run when a click event occurs.
* **hide():** This method hides the selected elements. This is similar to the CSS property display:none.
* **show():** To show hidden elements, this method is used.
* **text():** This method sets or returns the text content of the selected elements. When this method is used to return content, it returns the text content of all matched elements. When this method is used to set content, it overwrites the content of all matched elements.
* **setInterval():** This method calls a function at specified intervals (in milliseconds).
* **clearInterval():** The setInterval() method continues calling the function until clearInterval() is called.
* **on():** This method attaches one or more event handlers for the selected elements and child elements. This method simplifies jQuery code base. Event handlers attached using the on() method will work for both current and future elements. To remove event handlers, we can use the off() method.
* **Promise:** “Producing code” is code that can take some time. “Consuming code” is code that must wait for the result. A promise is a javascript object that links producing code and consuming code. A Javascript Promise object contains both the producing code and calls to the consuming code. When the producing code obtains the result, it should call one of the two callbacks: For success the call is myResolve(result value) and for error the call is myReject(error object). A Javascript Promise object can be pending, fulfilled or rejected.
* **SpeechSynthesis:** It is the interface of the web speech API which is the controller interface for the speech service. This can be used to retrieve information bout the synthesis voices available on the device.
* **speechSynthesisUtterance:** This interface of the Web speech API represents a speech request. It contains the content the speech service should read and information about how to read it. For example: language, pitch and volume). The method speechSynthesisUtterance() returns a new speechSynthesisUtterance object instance.
* **speechSynthesis.getvoices():** It returns a list of SpeechSynthesisVoice objects representing all the available voices on the current device.
* **speechSynthesis.speak():** Adds an utterance queue that is, it will be spoken when any other utterances queued before it have been spoken.

**APPENDIX**

**HTML Code:**

<!DOCTYPE html>

<html>

<head>

<title>

Speed reader App

</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1">

<link rel="stylesheet" href="https://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.css" />

<link rel="stylesheet"href="//cdnjs.cloudflare.com/ajax/libs/materialize/0.95.1/css/materialize.min.css">

<script src="https://code.jquery.com/jquery-1.11.1.min.js"></script>

<script src="https://code.jquery.com/mobile/1.4.5/jquery.mobile-1.4.5.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/pdf.js/2.13.216/pdf.min.js" integrity="sha512-IM60GPudO4jk+ZQm3UlJgKHhXQi5pNDM6mP+pLKL968YgkHMc7He3aGJOVHEZ9rJ4vAaEtJ8W6SKa7Qq4inzBA==" crossorigin="anonymous" referrerpolicy="no-referrer"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/materialize/0.95.1/js/materialize.min.js"></script>

<link rel="stylesheet" href="app.css">

</head>

<body>

<div data-role="page" id="page1">

<div data-role="header">

</div>

<div role="main" class="ui-content">

<div class="header">

<h1> Reading Expert </h1>

<p id="tagline"> To experience faster reading</p>

</div>

<div id="userchoice">

<button id="text"><a href="#page2"> <span id="entertext">Enter Text</span></a></button>

<p style="color:black;" "font-size:20px"> OR </p>

<button id="upload"><a href="#page3"><span id="uploadfile">Upload a file</span></a></button>

<p style="color:black;" "font-size:20px"> OR </p>

<button id="listen"><a href="#page4"><span id="listentext">Listen the text</span></a></button>

</div>

</div>

</div>

<div data-role="page" id="page2">

<div role="main" class="ui-content">

<div id="controllers">

<div style="color:black; font-weight:bold; font-size:20px; font-family:serif;"> Font-size:<span id="fontsize">50</span>px

</div>

<form class="full-width-slider" id="fontsizeslider-wrap">

<input type="range" name="fontsizeslider" id="fontsizeslider" min="20" max="100" value="50" step="5" data-highlight="true"></form><br><br>

<div style="color:black; font-weight:bold; font-size:20px; font-family:serif;">Speed:<span id="speed">300</span> words per minute</div>

<form class="full-width-slider" id="speedslider-wrap">

<input type="range" name="speedslider" id="speedslider" min="50" max="600" value="300" step="50" data-highlight="true">

</form><br><br>

<form class="full-width-slider">

<label for="progressslider" style="color:black; font-weight:bold; font-size:20px; font-family:serif;">Progress: <span id="percentage">0</span>%</label>

<input type="range" name="progressslider" id="progressslider" min="0" max="100" value="0" data-highlight="true">

</form>

</div>

<div id="result"> word </div>

<div id="error">

You have not entered enough text!

</div>

<p id="pastetext" style="color:black; font-family: serif; font-size:30px; font-weight: bold;"> Paste the text to be read :</p>

<div id="inputs" style="text-align:center">

<textarea id="userInput" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold;"></textarea>

<button id="start" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;" > Start Reading</button>

<button id="refresh" style=" width: 50%; margin:20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Refresh </button>

<button id="pause" style=" width: 50%; margin:20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Pause</button>

<button id="resume" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Resume</button>

</div>

</div>

</div>

<div data-role="page" id="page3">

<div data-role="header" >

</div>

<div role="main" class="ui-content">

<p id="url" style="color:black; font-family: serif; font-size:20px; font-weight: bold;"> Enter url of the file to be read:</p>

<input type="url" id="inputurl" size="10">

<form action="/action\_page.php">

<label for="pagenum" style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Enter pdf page no.:</label>

<input type="number" id="pageno" value="1" size="4">

</form>

<button id="submit">Submit</button>

<div id="controllers">

<div style="color:black; font-family: serif; font-size:20px; font-weight: bold;"> Font-size:<span id="fontsize2" >50</span>px

</div>

<form class="full-width-slider" id="fontsizeslider2-wrap">

<input type="range" name="fontsizeslider2" id="fontsizeslider2" min="20" max="100" value="50" step="5" data-highlight="true">

</form>

<br><br>

<div style="color:black; font-family: serif; font-size:20px; font-weight: bold;"><span id="speed2">300</span> words per minute</div>

<form class="full-width-slider" id="speedslider2-wrap">

<input type="range" name="speedslider" id="speedslider2" min="50" max="600" value="300" step="50" data-highlight="true">

</form><br><br>

<form class="full-width-slider">

<label for="progressslider" style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Progress: <span id="percentage2">0</span>%</label>

<input type="range" name="progressslider2" id="progressslider2" min="0" max="100" value="0" data-highlight="true">

</form>

</div>

<div id="result2"> word </div>

<div id="error2">

You have not entered enough text!

</div>

<div id="inputs2" style="text-align:center">

<div id="userInput2" style="border: 2px black; width: 50%; margin: 20px auto; font-family:serif; font-weight:bold;"></div>

<button id="start2" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Start Reading</button>

<button id="refresh2" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Refresh </button>

<button id="pause2" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Pause</button>

<button id="resume2" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Resume</button>

</div>

</div>

</div>

<div data-role="page" id="page4">

<div data-role="header" >

</div>

<div role="main" class="ui-content">

<div class="container">

<div class="row">

<nav>

<div class="nav-wrapper">

<div class="col s12">

</div>

</div>

</nav>

</div>

<form class="col s8 offset-s2">

<div class="row">

<label style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Choose voice</label>

<select id="voices"></select>

</div>

<div class="row">

<div class="col s6">

<label style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Rate</label>

<p class="range-field">

<input type="range" id="rate" min="1" max="100" value="10">

</p>

</div>

<div class="col s6">

<label style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Pitch</label>

<p class="range-field">

<input type="range" id="pitch" min="0" max="2" value="1" />

</p>

</div>

</div>

<div class="row">

<div class="input-field col s12">

<textarea id="message" class="materialize-textarea"></textarea>

<label style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Write message</label></div>

<p style="color:black;" "font-size:20px"> OR </p>

<p id="url2" style="color:black; font-family: serif; font-size:20px; font-weight: bold;"> Enter url of the file to be read:</p>

<input type="url" id="inputurl2" size="10">

<form action="/action\_page.php">

<label for="pagenum" style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Enter pdf page no.:</label>

<input type="number" id="pageno2" value="1" size="4"></form>

<div id="userInput3" style="border: 2px black; width: 50%;

margin: 20px auto; font-family:serif; font-weight:bold;"></div></div>

<a href="#" id="speak" class="waves-effect waves-light btn" style="color:black; font-family: serif; font-size:20px; font-weight: bold;">Speak</a></form> </div>

<div id="error3">

You have not entered enough text!</div>

<div id="inputs3" style="text-align:center">

<button id="refresh3" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Refresh </button>

<button id="pause3" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Pause</button>

<button id="resume3" style=" width: 50%; margin: 20px auto; font-family:serif; font-weight:bold; font-size:20px; background-color:#ffe6f9;"> Resume</button>

</div></div></div>

<script src="app2.js">

</script>

</body>

</html>

**CSS Code:**

#page1{

background: url(https://blog.hubspot.com/hubfs/books-every-aspiring-ceo-should-read.jpg);

background-position: center center;

background-size: 100% 100%;

/\*color: #00001a;\*/

/\*text-align: center;\*/

/\*font-family: Times,serif;\*/

/\*font-weight: bold;\*/

}

#page2,#page3,#page4{

background: url(https://img.freepik.com/free-photo/turquoise-matt-suede-fabric-background-velvet-texture\_113767-1486.jpg?w=740);

background-position: center center;

background-size: 100% 100%;

text-align: center;

}

h1{

font-family: Georgia,serif;

font-size: 60px;

color: #00004d;

text-align: center;

}

#tagline{

font-family: cursive,sans-serif;

font-size: 30px;

color: #cc0099;

text-align: center;

}

#userchoice{

width: 30%;

text-align: center;

margin: 20px auto;

font-weight: bold;

font-family: Georgia,serif;

font-size: 20px;

}

#text, #upload, #listen{

font-family: Georgia,serif;

font-weight: bold;

font-size: 20px;

background-color: #b3b3cc;

text-decoration-color: red;

border: 2px solid black;

}

#entertext,#uploadfile,#listentext{

color: black;

font-family: Georgia,serif;

font-size: 27px;

}

#error, #error2{

background-color: #D87777;

width: 50%;

margin: 20px auto;

font-size: 20px;

}

#result, #result2{

font-size: 50px;

background-color: rgba(255,255,255,0.5);

padding: 5% 0;

}

.full-width-slider input{

display: none;

}

.full-width-slider .ui-slider-track{

margin-left: 15px;

}

#userInput{

width: 50%;

margin: 20px auto;

}

#controllers{

text-align: center;

width: 50%;

margin: 20px auto;

}

**JS Code:**

$(function(){

var myArray;

var inputLength;

var reading = false;

var action;

var frequency = 200;

function enter\_text(refreshID,resumeID,pauseID,controllersID,resultID,errorID,startID,userInputID,progresssliderID,fontsizesliderID,speedsliderID,fontID,speedID,percentageID){

$(refreshID).hide();

$(resumeID).hide();

$(pauseID).hide();

$(controllersID).hide();

$(resultID).hide();

$(errorID).hide();

$(startID).click(function(){

myArray = $(userInputID).val().split(/\s+/);

console.log(myArray);

inputLength = myArray.length;

if(inputLength>1){

reading = true;

$(startID).hide();

$(errorID).hide();

$(userInputID).hide();

$(refreshID).show();

$(pauseID).show();

$(controllersID).show();

$(progresssliderID).attr("max", inputLength-1);

$(resultID).show();

$(resultID).text(myArray[0]);

action = setInterval(read, frequency);

}else{

$(errorID).show();

}

});

$(refreshID).click(function(){

location.reload();

});

$(pauseID).click(function(){

clearInterval(action);

reading = false;

$(pauseID).hide();

$(resumeID).show();

});

$(resumeID).click(function(){

action = setInterval(read, frequency);

reading = true;

$(resumeID).hide();

$(pauseID).show();

});

$(fontsizesliderID+"-wrap").on("slidestop",

function(event,ui){

$(fontsizesliderID).slider("refresh");

var slidervalue =

parseInt($(fontsizesliderID).val());

$(resultID).css("fontSize", slidervalue);

$(fontID).text(slidervalue);

});

$(speedsliderID+"-wrap").on("slidestop", function(event,ui){

$(speedsliderID).slider("refresh");

var slidervalue =

parseInt($(speedsliderID).val());

$(speedID).text(slidervalue);

clearInterval(action);

frequency = 60000/slidervalue;

if(reading){

action = setInterval(read, frequency);

}

});

$(progresssliderID).on("slidestop", function(event,ui){

$(progresssliderID).slider("refresh");

var slidervalue =

parseInt($(progresssliderID).val());

clearInterval(action);

$(resultID).text(myArray[slidervalue]);

$(percentageID).text(Math.floor(slidervalue/(inputLength-1)\*100));

if(reading){

action = setInterval(read, frequency);

}

});

function read(){

var value=$(progresssliderID).val();

if(value== inputLength-1){//last word

clearInterval(action);

reading = false; //move to none reading mode

$(pauseID).hide();

}else{

$(resultID).text(myArray[value]);

value++;

$(progresssliderID).val(value).slider('refresh');

$(percentageID).text(Math.floor(value/(inputLength-1)\*100));

}

}

}

$("#text").click(enter\_text("#refresh","#resume","#pause","#controllers","#result","#error","#start","#userInput","#progressslider","#fontsizeslider","#speedslider","#fontsize","#speed","#percentage"));

$("#upload").click(enter\_text("#refresh2","#resume2","#pause2","#controllers2","#result2","#error2","#start2","#userInput2","#progressslider2","#fontsizeslider2","#speedslider2","#fontsize2","#speed2","#percentage2")); $("#submit").on("click",function(event){

$("#error2").hide();

if($("#inputurl").val().slice(-3)=="pdf"){

pdfjsLib.getDocument($("#inputurl").val()).promise.then(function (PDFDocumentInstance) {

var totalPages = PDFDocumentInstance.numPages;

var pageNumber = +$("#pageno").val();

console.log(pageNumber);

getPageText(pageNumber , PDFDocumentInstance).then(function(textPage){

$("#userInput2").val(textPage);

$("#userInput2").text(textPage);

console.log(textPage);

});

});

}

else if($("#inputurl").val().slice(-3)=="txt"){

$.get($("#inputurl").val(), function(data){

$("#userInput2").val(data);

console.log(data);

});

}

else{

$("#error2").show();

}

});

function getPageText(pageNum, PDFDocumentInstance)

return new Promise(function (resolve, reject) {

PDFDocumentInstance.getPage(pageNum).then(function (pdfPage) {

pdfPage.getTextContent().then(function (textContent) {

var textItems = textContent.items;

var finalString = "";

for (var i = 0; i < textItems.length; i++) {

var item = textItems[i];

finalString += item.str + " ";

}

resolve(finalString);});

});

});

}

speechSynthesis.onvoiceschanged = function() {

voicesID = "#voices"

var $voicelist = $(voicesID);

if($voicelist.find('option').length == 0) {

speechSynthesis.getVoices().forEach(function(voice, index) {

var $option = $('<option>')

.val(index)

.html(voice.name + (voice.default ? ' (default)' :''));

$voicelist.append($option);

});

$voicelist.material\_select(); }}

function speak(text, voice, rate, pitch) {

var msg = new SpeechSynthesisUtterance();

msg.voice = voice

msg.rate = rate / 10;

msg.pitch = pitch;

msg.text = text;

msg.onend = function(e) {

console.log('Finished in ' + e.elapsedTime + ' seconds.');

};

speechSynthesis.speak(msg);

}

function speakWrap(messageID,voicesID,rateID,pitchID){

voices = window.speechSynthesis.getVoices()

speak($(messageID).val(), voices[$(voicesID).val()], $(rateID).val(), $(pitchID).val());

}

$("#error3").hide();

$("#speak").click(function(){

var txt=$("#message");

if(txt.val()!=null && txt.val() != "") {

speakWrap("#message","#voices","#rate","#pitch");

}

else if($("#inputurl2").val().slice(-3)=="txt"){

$.get($("#inputurl2").val(), function(data){

$("#userInput3").val(data);

console.log(data);

speakWrap("#userInput3","#voices","#rate","#pitch");});

}

else if($("#inputurl2").val().slice(-3)=="pdf"){

pdfjsLib.getDocument($("#inputurl2").val()).promise.then(function (PDFDocumentInstance) {

var totalPages = PDFDocumentInstance.numPages;

var pageNumber = +$("#pageno2").val();

console.log(pageNumber);

getPageText(pageNumber , PDFDocumentInstance).then(function(textPage){

$("#userInput3").val(textPage);

speakWrap("#userInput3","#voices","#rate","#pitch"); }) });

}

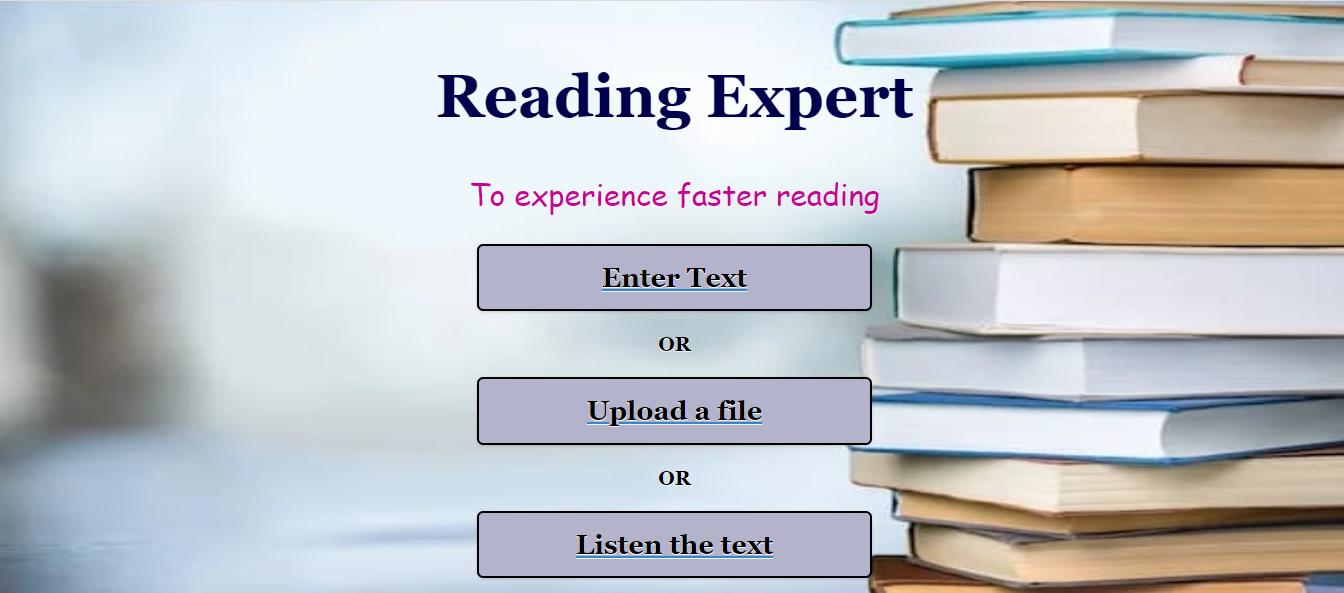
else{

$("#error3").show();}

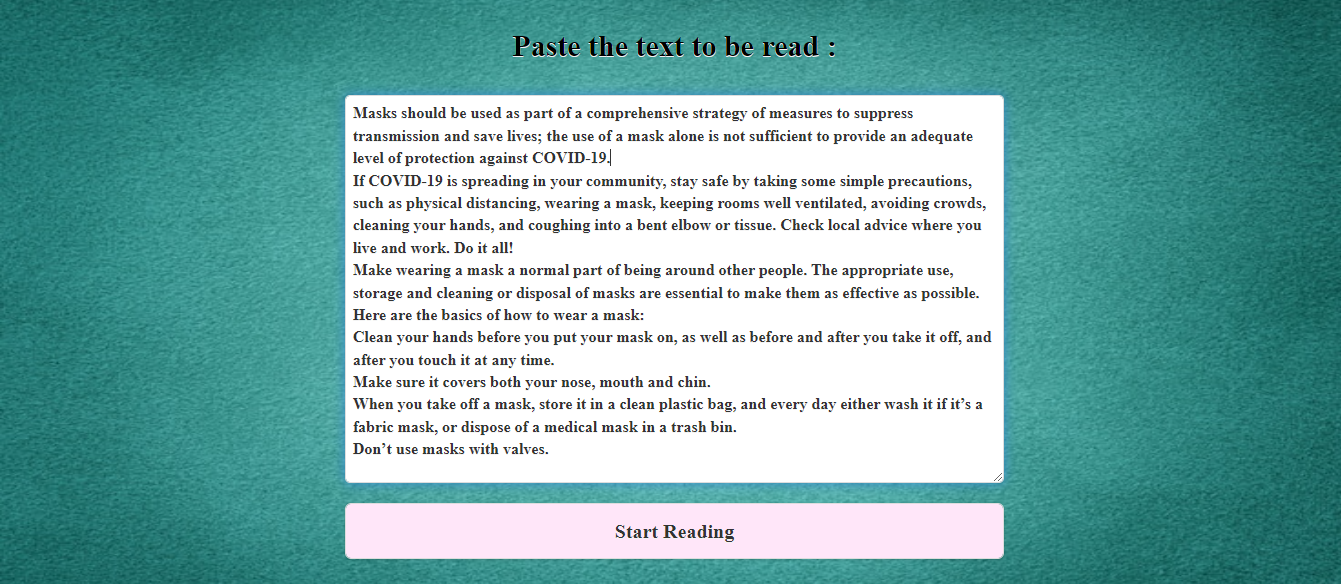
});});

**OUTPUT**

* Following is the home page of the app having buttons to give user a choice.

****

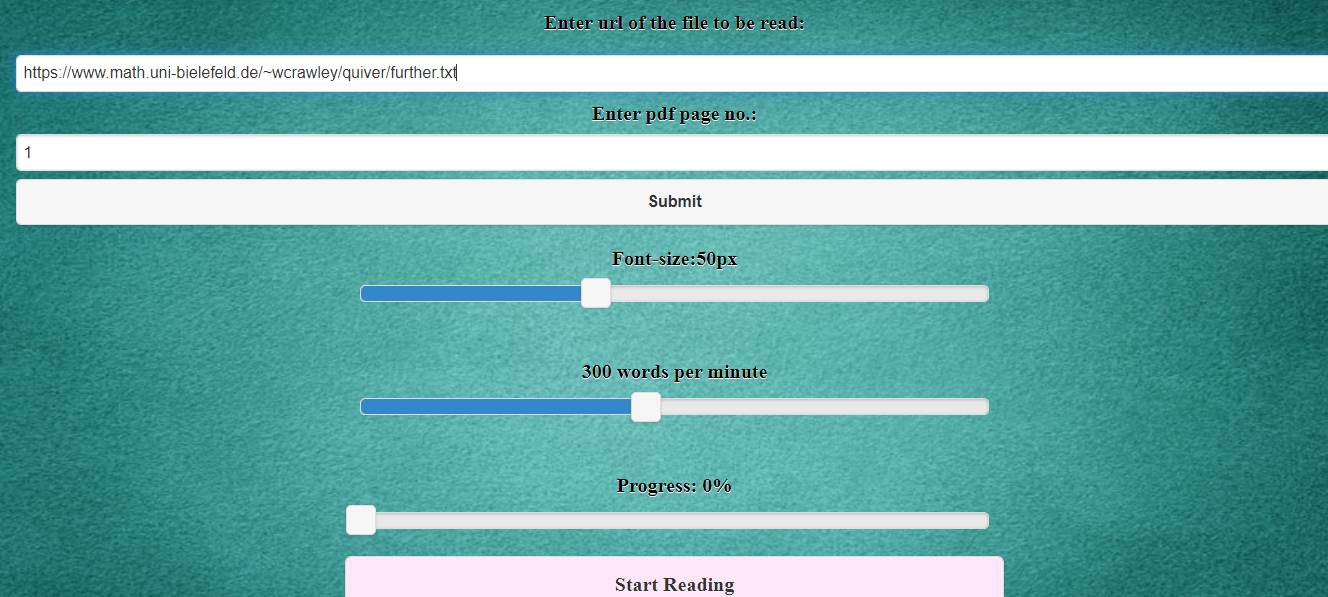
* Following is the view of the page when user chooses to read by entering the text.

****

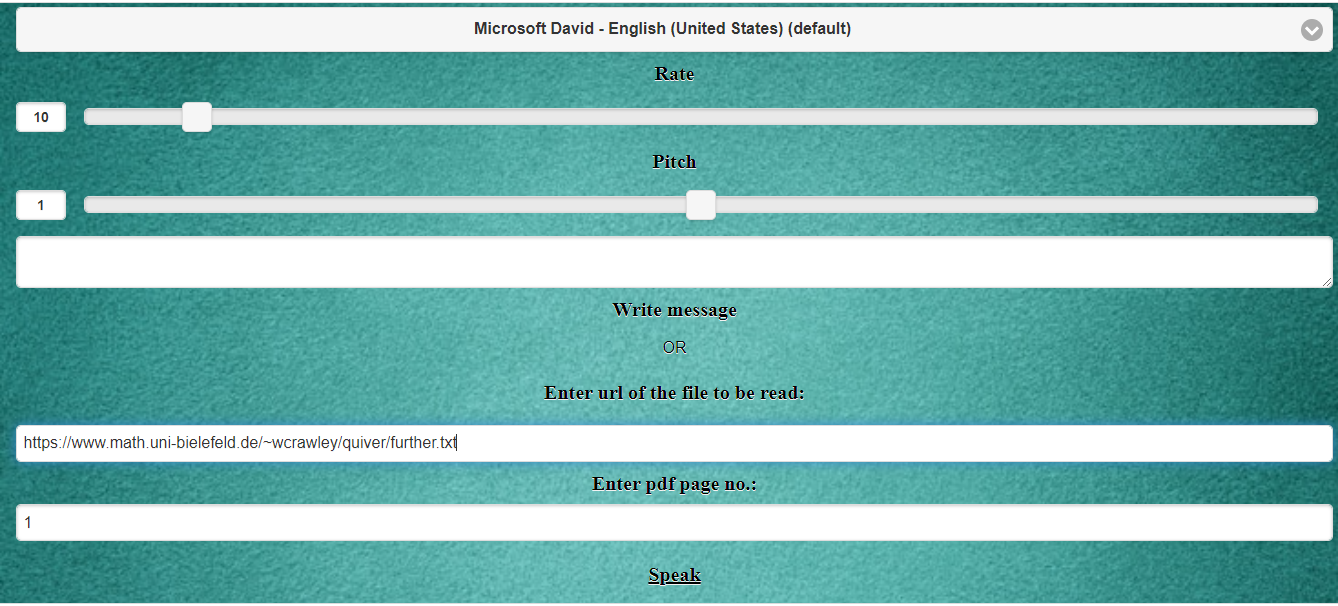
* Following is the view when the reading starts. Each word of the entered text is flashed one by one into a transparent box. The progress slider is showing the progress of the reading. Also, the user can refresh or pause the reading anytime using the buttons below.

****

* Following is the view of the page when user chooses to read by uploading file. User can enter url of the file in the space given and submit it. By submitting, the file gets load and then reading can be started using the start reading button. In case user enters url of the pdf file, he/she needs to enter the page number of the pdf he/she wants to read.

****

* Following is the view of the page when user chooses to listen the content of the file or the content of the text entered. User can manage rate and pitch using the sliders and speak button enables the audio for the user to listen the text.

****

**FUTURE SCOPE**

* The app can be incorporated with a feature such that the text can be extracted from a photograph also.
* Various options can be added for more convenience. For example: choice of reading word by word or sentence by sentence or paragraph by paragraph.
* The feature of saving and downloading the audio file in the device can be incorporated in the app so that the audio file can be transferred to any other device like cell phone and hence user can listen to it whenever he/she needs.
* Additional features such as book-marking, note-taking and a built-in dictionary.
* The app can be made available to android and iOS by uploading it to AppStore or google play store.

**REFERENCES**

* <https://en.wikipedia.org/wiki/Speed_reading>
* <https://irisreading.com/5-reasons-why-speed-reading-is-good-for-your-brain/>
* <https://en.wikipedia.org/wiki/Web_development>
* <https://developer.mozilla.org/en-US/docs/Learn>
* <https://jquery.com/>
* <https://jquerymobile.com/>
* <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
* <https://developer.mozilla.org/en-US/docs/Web/CSS>
* <https://developer.mozilla.org/en-US/docs/Web/API/SpeechSynthesis>