

**Project Report**  
**On**  
**ALL IN ONE WEBSITE**

---

**Submitted by**

**P Aravind-R170580**

**Under the guidance of**

**S Rajeshwari**

**Department of Computer Science and Engineering**



**Rajiv Gandhi University of Knowledge and Technologies(RGUKT),  
R.K.Valley, Kadapa, Andra Pradesh.**



**Rajiv Gandhi University of Knowledge Technologies**  
**RK Valley, Kadapa (Dist), Andhra Pradesh, 516330**

---

## **CERTIFICATE**

This is to certify that the project work titled “**ALL IN ONE WEBSITE**” is a bonafied project work submitted by **P. Aravind** in the department of COMPUTER SCIENCE AND ENGINEERING in partial fulfillment of requirements for the award of degree of Bachelor of Technology in Computer science and engineering for the year 2022-2023 carried out the work under the supervision

GUIDE  
**S Rajeshwari**

HEAD OF THE DEPARTMENT  
**N Sathyanandram**

## **ACKNOWLEDGEMENT**

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am extremely grateful to our respected Director, Prof. K. SANDHYA RANI for fostering an excellent academic climate in our institution.

I also express my sincere gratitude to our respected Head of the Department Mr N Sathyanandram for his encouragement, overall guidance in viewing this project a good asset and effort in bringing out this project.

I would like to convey thanks to our guide at college S Rajeshwari for his guidance, encouragement, co-operation and kindness during the entire duration of the course and academics.

My sincere thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all our friends and family members for their encouragement.

## **INDEX**

<b>S.NO</b>	<b>INDEX</b>	<b>PAGE NUMBER</b>
1	Abstract	5
2	Introduction	6
3	purpose	7
4	Scope	7
5	Requirement Specification	8-9
6	Analysis and design	10-11
6.1	Usecase	12-13
6.2	ER Diagram	14-15
7	Project code	16-20
8	Project Output	21-23
9	Test Case report	24
10	Conclusion	25

## **Abstract**

This project is about performing multiple operations in a single website where we can convert text to speech, one country currency value to another country currency value, and we can have Text editor in it. This project also consists conversion of long URL to short URL.

## **Introduction**

We made an interactive website based upon the Text to Speech converter, Text Editor, Currency converter, URL shortener and Password generator. The object of this website is to place all the process converters at one place.

### **Text to Speech:**

A Text to Speech is a software program that takes text as an input and converts it into speech automatically.

### **Currency Converter:**

A currency Converter allows traders, investors and speculators to compare the value one currency against each other.

A currency converter uses exchange rates to show users how the values of two currencies related to each other.

### **Text Editor:**

A text editor is a computer program that lets a user enter, change, store, and usually print text.

Typically a Text Editor provides an empty display screen with a fixed line length visible line numbers. You can then fill the lines in with text, line by line.

### **Password generator:**

A random password generator is a software program that takes input from a random or pseudo random number generator and automatically generates a password.

### **URL shortner:**

URL shortner is a software program that converts lengthy URLs into Short URL.

It is used for transforming long, ugly links(url) into nice, memorable and trackable short URLs.

## **Purpose**

The main purpose of this “ALL IN ONE” website is about to perform multiple tasks in a single website. Without using separate apps for simple tasks, we have designed this website for performing all these simple tasks at one place. This website will be very useful for students, officials, investors for performing text-to-speech, currency Exchange and Random password generator.






## **Scope**

The scope of this project is about performing multiple operations in a single website where we can convert text to speech, one country currency value to another country currency value, and we can have Text editor in it. This project also consists conversion of long URL to short URL.

In this project we have developed all these functionalities by using simple reactjs code instead of using machine learning or other complicated algorithms.

### **Advantages:**

The major advantage of this project is performing multiple actions at a single website.

-  By using text-to-speech we can simply convert text into audio which helps to the people who are not able to read.
-  TextEditor provides a lot of advantages to the user to perform different kinds of operations on the text like write, edit, add, delete, bold, italic...etc.
-  The main advantage of Currency exchange is simply converting currency value from one form to another ex: INR – USD.
-  A strong password generated online can help you to protect the security of your personal and professional email accounts, social network accounts, wifi encryption, banking and financial and savings accounts.
-  URL shortner helps in reducing the size of the huge and lengthy url into a small and understandable url with a QR code which indicates the URL.

## **Requirement Specification**

### **Hardware Configuration:**

#### **Client Side:**

<b>Ram</b>	4 GB
<b>Hard disk</b>	10 GB
<b>Processor</b>	1.0 GHz

### **Software Requirement:**

<b>Client side Language</b>	ReactJs
<b>Database Server</b>	NO DATABASE USED
<b>Operating System</b>	Ubuntu, Windows or any equivalent OS



## **REACT.JS**

- 🟢 React is a free and open source front-end javascript library for building user interfaces based on UI components.
- 🟢 It is declarative, efficient and flexible open source javascript library for building simple, fast, and scalable front-ends of web applications.

### **It is Used for:**

- 🟡 Mobile Applications
- 🟡 Desktop Applications
- 🟡 Web Applications
- 🟡 Application servers
- 🟡 Data Base Connection

- ➔ The main objective of ReactJS is to develop user interfaces that improves the speed of the apps.
- ➔ It uses virtual DOM(javascript object), which improves the performance of the app. The javascript virtual DOM is faster than the regular DOM.
- ➔ We can use ReactJS on the client and server side as well as with other frameworks. It uses component and data patterns that improve readability and helps to maintain larger apps.
- ➔ ReactJS is all about components and its application is made up of multiple components, and each component has its own logic and controls these components can be reusable which help you to maintain the code when working on larger scale.

## **Analysis and Design**

### **Analysis:**

ALL IN ONE website is build for performing mutiple tasks and expand my programming knowledge.

And I am good at JavaScript,Html,Css and ReactJs so i came into thought creating simple game Brick Breaker Game.

### **Disadvantage of present system:**

- **Manual Control**
- **Time consuming**

## **Design Introduction:**

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization. Once the software requirements have been analyzed and specified the software design involves three technical activities - design, coding, implementation and testing that are required to build and verify the software.

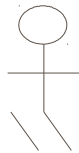
The design activities are of main importance in this phase, because in this activity, decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decisions have the final bearing upon reliability and maintainability of the system. Design is the only way to accurately translate the customer's requirements into finished software or a system.

Design is the place where quality is fostered in development. Software design is a process through which requirements are translated into a representation of software. Software design is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data

## **UML Diagrams:**

Actor:

A coherent set of roles that users of use cases play when interacting with the use cases. an observable result of value of an actor.



Use case: A description of sequence of actions, including variants, that a system performs yields an observable result of value of an actor. actor diagram is drawn in a eclipse shape



UML stands for Unified Modeling Language. UML is a language for specifying, visualizing and documenting the system. This is the step while developing any product after analysis. The goal from this is to produce a model of the entities involved in the project which later need to be built. The representation of the entities that are to be used in the product being developed need to be designed.

### **USECASE DIAGRAMS:**

Use case diagrams model behavior within a system and helps the developers understand of what the user require. The stick man represents what's called an actor.

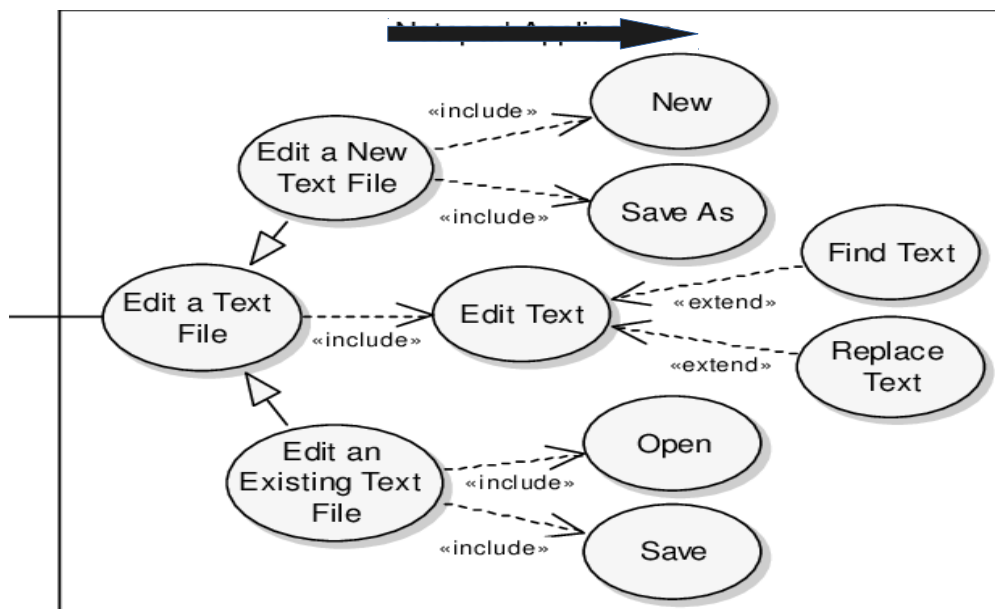
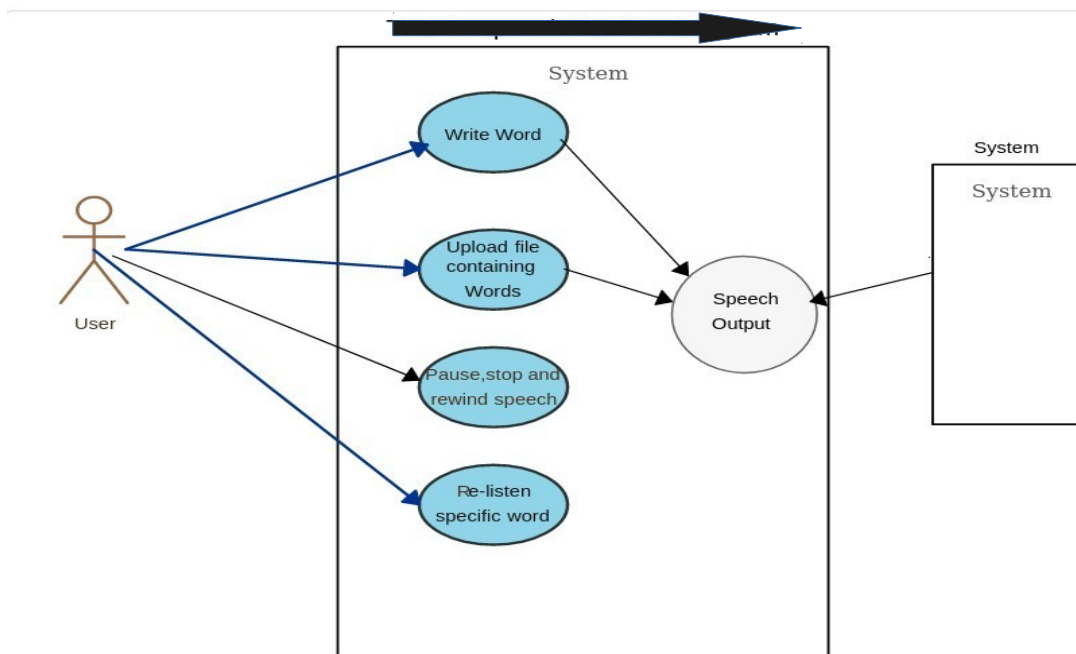
Use case diagram can be useful for getting an overall view of the system and clarifying that can do and more importantly what they can't do.

Use case diagram consists of use cases and actors and shows the interaction between the use case and actors.

- The purpose is to show the interactions between the use case and actor.
- To represent the system requirements from user's perspective.
- An actor could be the end-user of the system or an external system.

**USECASE DIAGRAM:** A Use case is a description of set of sequence of actions. Graphically it is rendered as an ellipse with solid line including only its name. Use case diagram is a behavioral diagram that shows a set of use cases and actors and their relationship. It is an association between the use cases and actors. An actor represents a real-world object. Primary Actor – Sender, Secondary Actor Receiver.

## Use Case Diagrams:



## **ER Diagram:**

The Entity-Relationship (ER) model was originally proposed by Peter in 1976 [Chen76] as a way to unify the network and relational database views. Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database design for the database designer, the utility of the ER model is:

- It maps well to the relational model. The constructs used in the ER model can easily be transformed into relational tables.
- It is simple and easy to understand with a minimum of training. Therefore, the model can be used by the database designer to communicate the design to the end user.
- In addition, the model can be used as a design plan by the database developer to implement a data model in specific database management software.

## **ER Notation**

There is no standard for representing data objects in ER diagrams. Each modeling methodology uses its own notation. The original notation used by Chen is widely used in academics texts and journals but rarely seen in either CASE tools or publications by non-academics. Today, there are a number of notations used; among the more common are Bachman, crow's foot, and IDEFIX.

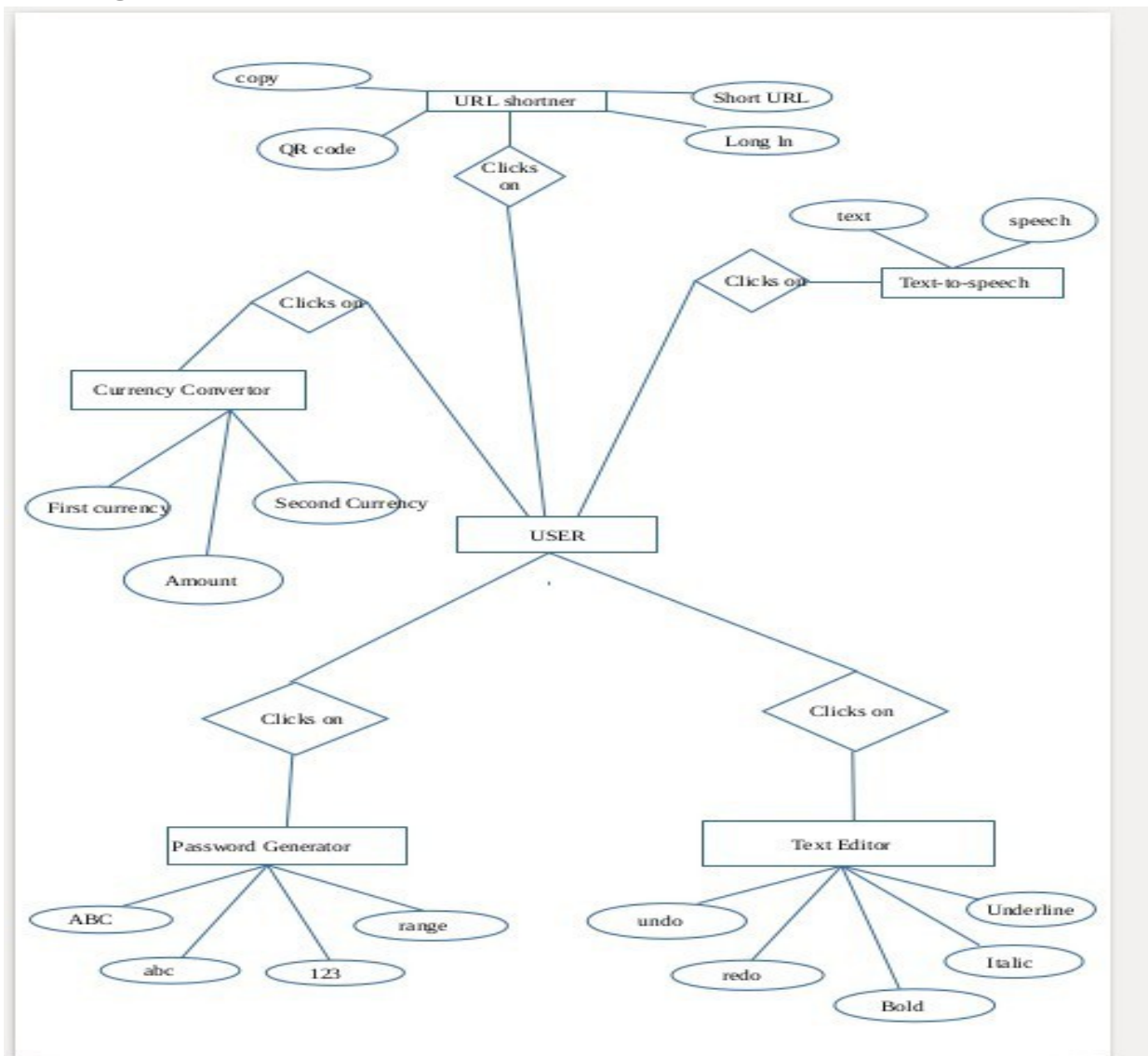
All notational styles represent entities as rectangular boxes and relationships as lines connecting boxes. Each style uses a special set of symbols to represent the cardinality of a connection. The notation used in this document is from Martin. The symbols used for the basic ER constructs are:

- **Entities** are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.
- **Relationships** are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs

- **Attributes**, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
- **Cardinality** of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.

**Existence** is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.

## ER Diagram



# CODE

## Password.js

```
import React, { useState } from 'react'
import './passwordGenerator.css';
import { ToastContainer, toast } from 'react-toastify';
import 'react-toastify/dist/ReactToastify.css';
import copy from "copy-to-clipboard";
import passwordhome from './passwordhome.svg';
import { Barcode } from './Barcode';
import { motion } from 'framer-motion';

const PasswordGenerator = () => {
  const generatedNotify = () => toast.success("Generated Password");
  const copyPassword = () => toast.warn("Copied to Clipboard")
  const [password, setPassword] = useState("")
  const [passwordLength, setPasswordLength] = useState(15)
  const [upperCase, setupperCase] = useState(false)
  const [lowerCase, setlowerCase] = useState(false)
  const [includeNumbers, setIncludeNumbers] = useState(false)
  const [symbols, setsymbols] = useState(false)
  const numbers = "0123456789";
  const upperCaseLetters = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  const lowerCaseLetters = "abcdefghijklmnopqrstuvwxyz";
  const specialCharacters = "!^+%&/'=?_#$%&{[]};:;>÷`<.*-@é";
  const handleGeneratePassword = () => {
    if (!upperCase && !lowerCase && !includeNumbers && !symbols) {
    }
    else {
      let characterList = ""
      if (includeNumbers) {
        characterList = characterList + numbers
      }
      if (upperCase) {
        characterList = characterList + upperCaseLetters
      }
      if (lowerCase) {
        characterList = characterList + lowerCaseLetters
      }
      if (symbols) {
        characterList = characterList + specialCharacters
      }
      setPassword(createPassword(characterList))
      generatedNotify()
    }
  }

  const createPassword = (characterList) => {
    let password = ""
    const characterListLength = characterList.length
    for (let i = 0; i < passwordLength; i++) {
      const characterIndex = Math.round(Math.random() * characterListLength)
      password = password + characterList.charAt(characterIndex)
    }
    return password
  }

  const handleCopyPassword = (e) => {
    if (password === "") {
    }
    else {
      copy(password)
      copyPassword()
    }
  }
}
```

```
return (
  <>

    <motion.div
      initial={{ opacity: 0, x: "100%" }}
      animate={{ opacity: 1, x: 0 }}
      exit={{ opacity: 0, y: "-100%" }}
      transition={{ duration: 0.5 }}
      className='passwordmain'>

      <div class="container">
        <div class="row passwordrow">
          <h1 className='passwordheading'>Random Password Generator</h1>
          <p className='passwordheading'>Create strong and secure passwords to
            keep your account safe online</p>
          <div class="col-md-1"></div>
          <div class="col-md-4"><img src={passwordhome}/>
          </div>
          <div class="col-md-1"></div>
          <div class="col-md-6">
            <div class="row">
              <div class="col-md-10 generatedPassword d-flex justify-
                content-around">
                <input className='generatedPasswordInput'
                  value={password}/>
                <span className='generatedPasswordInputs span btn btn-
                  primary'>{passwordLength<10?"Weak":passwordLength>10&&passwordLength
                    <35?"Good":"Very Secure"}</span>
                </div>
                <div class="col-md-2">
                  <button className='btn btn-primary mt-2 '
                    onClick={handleCopyPassword}>Copy</button>
                </div>
              <div class="row mt-4">
                <div class="col-md-5">
                  <h5 className='passwordLength'>Password length :
                    {passwordLength}</h5>
                </div>
                <div class="col-md-7">
                  <input type="range" value={passwordLength} onChange={(e)
                    => setPasswordLength(e.target.value)} />
                </div>
              </div>

              <div class="row mt-5">
                <div class="col-md-4">
                  <h5 className='passwordLength'>Characters used:</h5>
                </div>
                <div class="col-md-8">
                  <div style={{ display:"inline-
                    block",marginRight:"20px"}}>
                    <input className='custom-control-input'
                      checked={upperCase} onChange={(e) => setupperCase(e.target.checked)}
                      type="checkbox" id="uppercase-letters" name="uppercase-letters" />
                    <h5 style={{ display:"inline-
                      block",marginRight:"20px"}}>htmlFor="passwordLength">ABC</h5>
                  </div>
                  <div style={{ display:"inline-
                    block",marginRight:"20px"}}>
                    <input className='custom-control-input'
                      checked={lowerCase} onChange={(e) => setlowerCase(e.target.checked)}
                      type="checkbox" id="lowercase-letters" name="lowercase-letters" />
                    <h5 style={{ display:"inline-
                      block",marginRight:"20px"}}>htmlFor="passwordLength">abc</h5>
                  </div>
                  <div style={{ display:"inline-
                    block",marginRight:"20px"}}>
                    <input className='custom-control-input'
                      checked={includeNumbers} onChange={(e) =>
                      setIncludeNumbers(e.target.checked)} type="checkbox" id="include-numbers"
                      name="include-numbers"/>
                    <h5 style={{ display:"inline-
                      block",marginRight:"20px"}}>htmlFor="passwordLength">123</h5>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  );
```



# Texttospeech.js

```
import React,{useState} from 'react';
import { useSpeechSynthesis } from 'react-speech-kit';
import { useSpeechRecognition } from 'react-speech-kit';
import { motion } from 'framer-motion';
import './TexttoSpeech.css'
const TexttoSpeech = () => {
  const { speak } = useSpeechSynthesis();
  const [value, setValue] = useState("");
  const [text, setText] = useState("");
  const { listen, listening, stop } = useSpeechRecognition({
    onResult: (result) => {
      setText(result);
    },
  });
  const handleSpeech = () => {
    listen()
  }
  const handleSpeechstop = () => {
    stop()
    setText("")
  }
  return (
    <>
    { /*Text to speech converter */}
    <motion.div className="TextToSpeechMain"
    initial={{ opacity: 0, x: "-100%" }}
    animate={{ opacity: 1, x: 0 }}
    exit={{ opacity: 0, y: "-100%" }}
    transition={{ duration: 0.5 }}
    >
      <div className='row'>
        <div className='col-md-6 m-5'>
          <h1>TexttoSpeech Converter</h1>
          <textarea
            rows="15"
            className='form-control textareaspeech'
            value={value}
            onChange={(event) => setValue(event.target.value)}
          />
          <div className='text-center d-flex justify-content-around mt-3'>
            <button className='btn btn-primary' onClick={() => speak({ text: value })}>Speak</button>
            <button className='btn btn-danger' onClick={() => setValue("")}>clear</button>
          </div>
        </div>
      </div>
    </>
  )
}
```

# Currencyexchange.js

```
import React, { useState } from 'react'
import axios from 'axios';
import Dropdown from './Dropdown';
import { motion } from 'framer-motion';
const CurrencyExchange = () => {
  const [firstCurrency,setFirstCurrency]=useState("")
  const[secondCurrency,setSecondCurrency] = useState("")
  const [amount,setAmount] = useState(0)
  const handleClick = ()=>{
    const options = {
      method: 'GET',
      url: 'https://currency-converter-by-api-ninjas.p.rapidapi.com/v1/convertcurrency',
      params: {have: `${firstCurrency}`, want: `${secondCurrency}`, amount: `${amount}`},
      headers: {
        'X-RapidAPI-Key': '9c2bb9bb51msha6379474ab7af30p14325cjsnacc168d28352',
        'X-RapidAPI-Host': 'currency-converter-by-api-ninjas.p.rapidapi.com'
      }
    };
    axios.request(options).then(function (response) {
      setAmount(response.data.new_amount)
    }).catch(function (error) {
      console.error(error);
    });
  }
  const handleChange = ()=>{
    const temp = firstCurrency;
    setFirstCurrency(secondCurrency);
    setSecondCurrency(temp);
  }

  return (
    <div className='currencymain'>
      <motion.div className='currencybody text-center'
        initial={{ opacity: 0, x: "-100%" }}
        animate={{ opacity: 1, x: 0 }}
        exit={{ opacity: 0, y: "-100%" }}
        transition={{ duration: 0.5 }} >
        <h1 className='text-center'>Real Time CurrencyExchange</h1>
        <Dropdown value={firstCurrency} setCurrency={setFirstCurrency} />
        <Dropdown value={secondCurrency} setCurrency={setSecondCurrency}/>
        <button className='btn btn-primary exchange' onClick={handleClick}>Exchange</button>
        <div className='exvchangebutton'>
          <button onClick={handleChange}><span class="MuiIconButton-label"><svg class="MuiSvgIcon-root cc49" focusable="false" viewBox="0 0 24 24" aria-hidden="true"><path d="M6.99 11L3 15l3.99 4v-3H14v-2H6.99v-3zM21 9l-3.99-4v3H10v2h7.01v3L21 9z"></path></svg></span></button>
        </div>
        <div className='inputoutputfields'>
          <input type="number" onChange={(e)=>setAmount(e.target.value)} />
          <input value={amount} />
        </div>
      </motion.div>
    </div>
  )
}
```

export default CurrencyExchange

# TextEditor.js

```
import React from 'react';
import 'suneditor/dist/css/suneditor.min.css'
import SunEditor from 'suneditor-react';
import './texteditor.css';
import {motion} from 'framer-motion';

const TextEditor = () => {
  const handleChange = ()=>{
    console.log()
  }
  return (
    <>
      <motion.div
        initial={{ opacity: 0, x: "-100%" }}
        animate={{ opacity: 1, x: 0 }}
        exit={{ opacity: 0, y: "-100%" }}
        transition={{ duration: 0.5 }}
        className='texteditormain'>
        <h1 className='text-center p-3'>Text-Editor</h1>
        <SunEditor
          autoFocus={true}
          placeholder="Enter something ...."
          height='430px'
          onChange={handleChange}
          setDefaultStyle="font-family:Roboto Mono;font-size:30px;"
          setOptions={{
            buttonList: [
              ['undo', 'redo'],
              ['font', 'fontSize', 'formatBlock'],
              ['paragraphStyle', 'blockquote'],
              ['bold', 'underline', 'italic', 'strike', 'subscript', 'superscript'],
              ['fontColor', 'hiliteColor', 'textStyle'],
              ['removeFormat'],
              '/',
              ['outdent', 'indent'],
              ['align', 'horizontalRule', 'list', 'lineHeight'],
              ['table', 'link', 'image', 'video', 'audio' /**, 'math' */],
              ['fullScreen', 'showBlocks', 'codeView'],
              ['preview', 'print'],
              ['save', ],
            ]
          }}
        />
      </motion.div>
    </>
  )
}
```

export default TextEditor

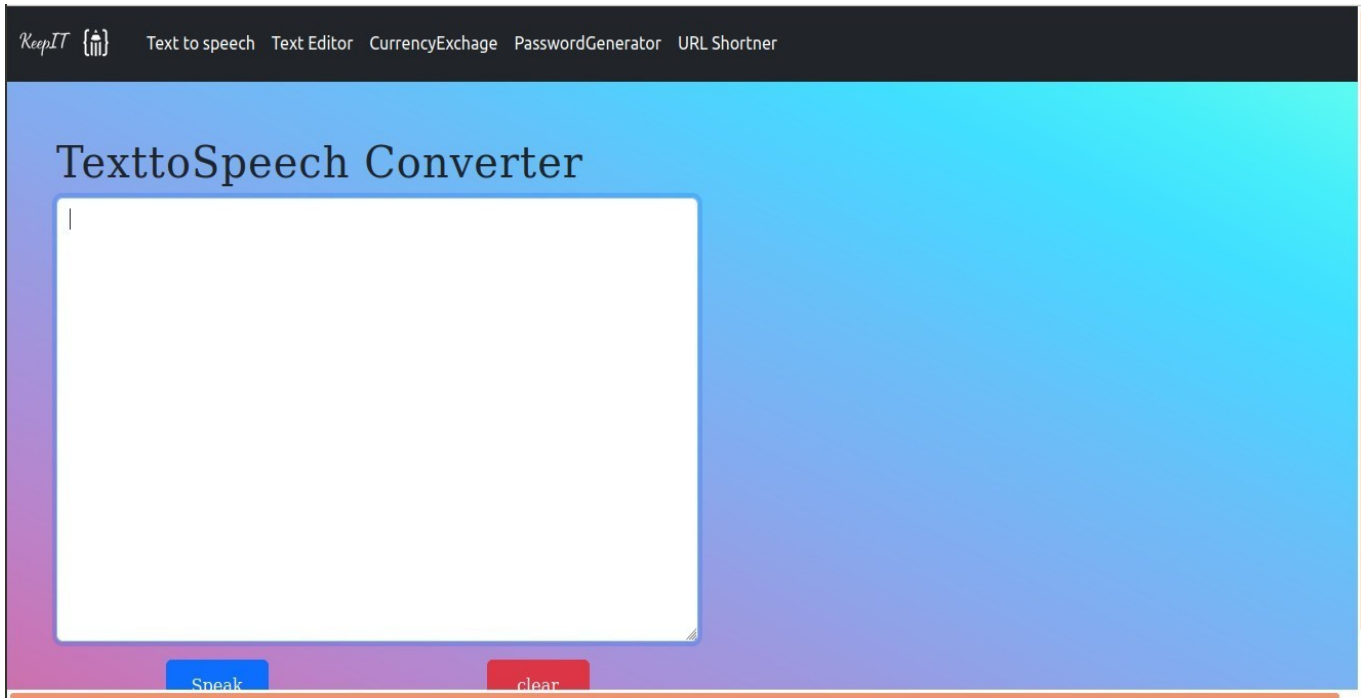
# URLshortner.js

```
import React,{useState} from 'react';
import axios from 'axios';
import './urlshortner.css';
import urlshortner from './urlshortener.jpg';
import copy from "copy-to-clipboard";
import { UrlShortnerBarcode } from './UrlShortnerBarcode';
import { Checkmark } from 'react-checkmark';
import { motion } from "framer-motion";
```

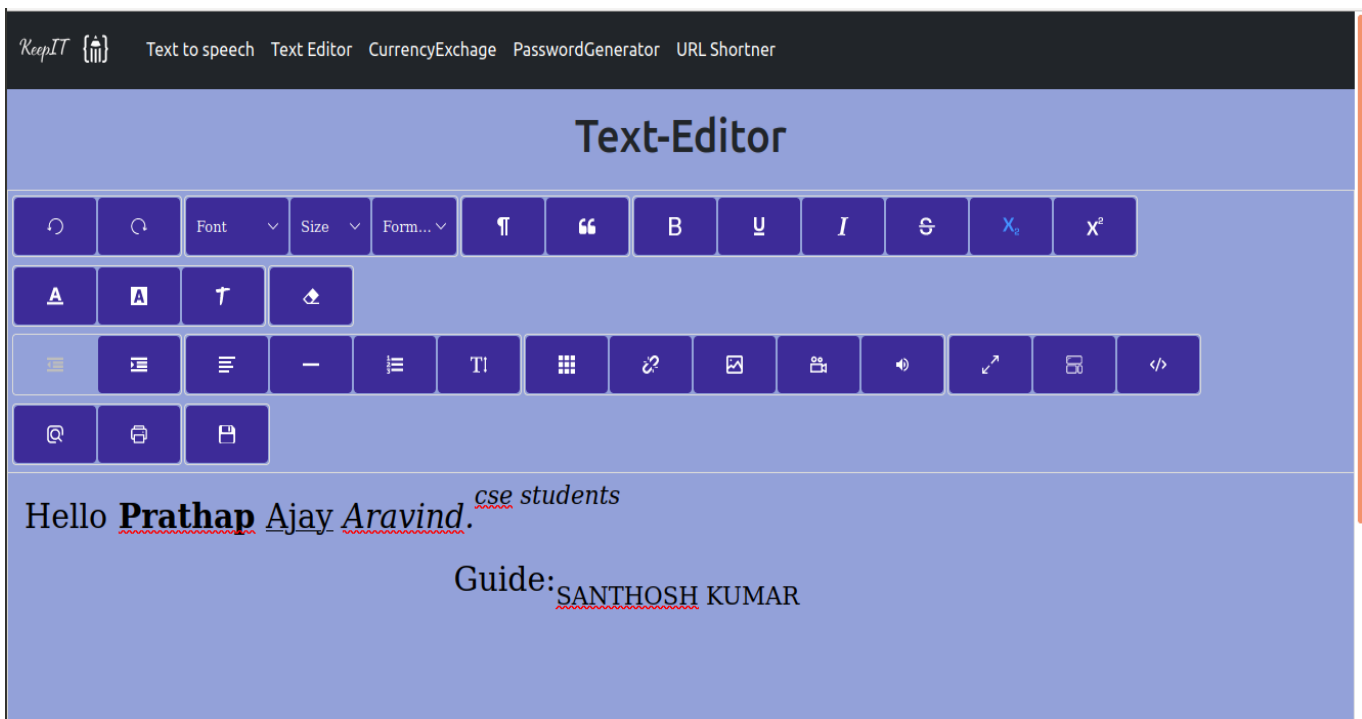
```
const UrlShortner = () => {
  const [userInput, setUserInput] = useState("");
  const [shortenedLink, setShortenedLink] = useState("")
  const [error, setError] = useState("")
  const fetchData = async () => {
    try {
      const response = await axios( `https://api.shrtco.de/v2/shorten?url=${userInput}` );
      setShortenedLink(response.data.result.full_short_link);
    } catch (e) {
      console.log(e);
      setError(e.message);
    }
  }
  const handleCopyPassword = ()=>{
    copy(shortenedLink)
    setShortenedLink("")
  }
  return (
    <>
    <motion.div
      initial={{ opacity: 0, x: "-100%" }}
      animate={{ opacity: 1, x: 0 }}
      exit={{ opacity: 0, y: "-100%" }}
      transition={{ duration: 0.5 }}
      className='row urlShortner p-5'>
      <h1 className='text-center text-white urlShortnerHeading'>URL Shortener</h1>
      <div className='col-md-1'></div>
      <div className='col-md-5'>
        <input value={userInput} className='urlShortnerInput' placeholder='Paste the Url' onChange={(e)=>setUserInput(e.target.value)}>/>
        <span><button className=" btn btn-info urlShortnerInputButton" onClick={() => {
          fetchData()
          setUserInput("")
        }}>Shorten Url</button> </span>
      </div>
      {
        shortenedLink.length>0 ?
        <>
        <input className='urlShortnerInput mt-5' value={shortenedLink}/>
        <span className='urlShortnerInputButton btn btn-warning ' onClick={handleCopyPassword}>Copy</span>
        <div className='text-center mt-5'> <UrlShortnerBarcode UrlData={shortenedLink} />
        </div>
        </>
        : error.length > 0 &&
        <>
        <div>
          <h3 className='urlShortnerPara mt-5'>An error occurred to generate the URLThe URL is not valid.</h3>
          <h5 className='urlShortnerPara possibleError'>Possible Errors : </h5>
          <p className='urlShortnerPara'> <Checkmark size='26px' />Check if the domain is correct</p>
          <p className='urlShortnerPara'> <Checkmark size='26px' />Check if the site is online</p>
          <p className='urlShortnerPara'> <Checkmark size='26px' />Check the length of the url is not too small</p>
          <p className='urlShortnerPara'> <Checkmark size='26px' />Check the address bars and punctuation</p>
          <p className='urlShortnerPara'> <Checkmark size='26px' />The URL may have been blocked</p>
          <p className='urlShortnerPara'> <Checkmark size='26px' />The url may have been reported</p>
          <p className='urlShortnerPara'> <Checkmark size='26px' />Make sure the url does not contain spam</p>
        </div>
        <span className='urlShortnerInputButton btn btn-danger '>Enter valid URL</span>
        </>
      </div>
      <div className='col-md-6'>
        <img src={urlshortner} width="500px" className='urlshortnerimage mt-5'>
      </div>
    </motion.div>
    </>
  )
}
```

# Evaluation


## Text to speech



## Text editor



## Currency exchange

KeepIT 

[Text to speech](#) [Text Editor](#) [CurrencyExchange](#) [PasswordGenerator](#) [URL Shortner](#)

### Real Time CurrencyExchange


USD

▼

INR

▼

Exchange




1

▼

79.65


## Random password generator

KeepIT 

[Text to speech](#) [Text Editor](#) [CurrencyExchange](#) [PasswordGenerator](#) [URL Shortner](#)


### Random Password Generator

Create strong and secure passwords to keep your account safe online



Good

Copy



GoodCopy

Password length : 22

Characters used:


☒ ABC

☒ abc


☒ 123

☒ # \$ &

Generate Password




## URL shortner

KeepIT  [Text to speech](#) [Text Editor](#) [CurrencyExchange](#) [PasswordGenerator](#) [URL Shortner](#)

URL Shortener

Shorten Url

Copy



## TEST CASE REPORT

S.no	Test Case	Input	Actual Output	Expected Output	Is Actual output is same as Expected output?
1	Type the text	Text	Speech Should come	Speech came	Yes
2	Data Modify	Data-Text, Undo, Redo, Font, etc.,	Data will modify	Data Modified	Yes
3	Currency Exchange	First Currency, Second Currency, Amount	Currency will change	Currency Changed	Yes
4	Password Generator	Range,ABC, abc,123,special characters	Password will generate with QR code	Password generated with QR code	Yes
5	URL shortner	Lengthy URL	Short and simple URL generated with QR code	Short URL generated	Yes



## **Conclusion**

By developing this website we can create simple and easy user interface which helps in performing multiple tasks in a single website. Apps will perform only a task based on the core objective of that app. Instead of apps by using these type of websites it is simple for accessing, Simple codes are used in developing.

”ALL IN ONE” website is a multifunctional performing website that we have created .

## **References**

**For React.Js**

<https://reactjs.org/>

<https://www.w3schools.com/REACT/DEFAULT.ASP>

\*\*\*\*\***THANKYOU**\*\*\*\*\*