VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

Fall Semester 2022-2023
E1 + TE1 slot

Course Code: CSE 3013

Course Title: Artificial Intelligence

Project Report
Assisted Vision – A Smart Eye

Submitted by:
Mudit Jain 20BCE0362
Nikhil Garg-20BCE2389
Ish Gupta-20BCE2394
Akshat Bisht-20BCE0629
Chinmay Bansal-20BCE0445

Under the guidance of

Dilipkumar S.

VIT , Vellore.

Introduction:

Our project aims to facilitate effortless learning for the visually impaired students by making interaction with real world objects easier. By using our technology, users can identify, perceive and detect real world objects via object detection. We also aim to enhance a number of cognitive skills, including counting and spatial skills simultaneously fostering social interaction by introducing games, as well as facilitate easy learning by features developed for academics.

Motivation:

Blind and visually impaired students have used a variety of methods to learn to read, write, and acquire other skills, both academic and non-academic.The challenge for educators of blind and visually impaired children, including those with other disabilities, is how to teach skills that sighted children typically acquire through vision. It also is very important for blind and visually impaired children, including those with other disabilities, who need orientation and mobility services, to receive appropriate instruction in orientation and mobility as early as possible. Students who are blind or visually impaired need specialized instruction in order to understand concepts in a highly visual world. This unique programming includes teaching through concrete and unifying experiences and learning by doing.

Objective:

- ➢ To implement object detection on live motion as well as images and videos
- ➢ To convert object names into speech
- ➢ To build an examination system on the basis of audio commands
- ➢ To build games that develop strong visualization skills

Literature Survey:

1. **Title: Assisting visually impaired people by Computer Vision- A Smart Eye**

   Year: 2021

   Author: Arun Kumar Ravula, Pallepati Vasavi, K.Ram Mohan Rao

   Inferences:

   Distance estimation was the most important feature as it is most useful for the user. When user uses the device, the object which is placed in front of him will be detected and spoken out to the user along with the distance of the object from him.

   Limitations:

   It can only detect the things but it can't allow the user to play games like tic-tac-toe using voice commands.

2. **Title: Efficient Multi- Object Detection and Smart Navigation Using Artificial Intelligence for Visually Impaired People**

   Year: 2020

   Author: Rakesh Chandra Joshi, Saumya Yadav, Malay Kishore Dutta, and Carlos M. Travieso-Gonzalez

   Inferences:

   The developed technology is found to be highly useful, with which users can also understand the surrounding scenario easily while navigation, without putting in too much effort. The proposed aid for visually impaired seems good in the sense that it does not need any prior knowledge about the position, shape and size of object and obstacles.

   Limitations:

   Use of in various distance sensors makes it expensive and not affordable. More sensors will be associated with it to detect, for example, downstairs and other trajectories, giving a wider range of assistance to the visually impaired.

3. **Title: Vision-Based System for Assisting Blind People to Wander**
   Unknown Environments in a Safe Way
   Year: 2021
   Author: Andrés A. Díaz-Toro , Sixto E. Campaña-Bastidas , Eduardo F. Caicedo-Bravo
   Inferences:
   Transfer of technology from autonomous cars to the field of assistive tools for blind people is feasible due to both have similar requirements such as real-time performance, work in unknown environments, robust to changing environments, and safe and the increase in accuracy and portability of 3D vision sensors and in the computing power and portability of embedded processors.
   Limitations:
   The general disadvantage is related with overloading the sense of hearing, especially in dynamic environments, like in.For tactile feedback, Braille displays haptic belts,vests, and gloves are used. Braille displays are difficult to interpret in dynamic environments, and the other ones need a period of training for interpreting the commands adequately.

4. **Title: A Survey on Recent Advances in AI and Vision-Based Methods for Helping and Guiding Visually Impaired People**
   Year: 2022
   Author: Helene Walle, Cyril De, Barthelem y erres, Gilles Venturini
   Inferences:
   The recent innovation of AI has increased robustness and efficiency. Future inventions will make it possible to propose safer wayfinding systems for BVIPs.
   Limitations:
   Acquisition devices are high cost
   The adaptation of such models to specific conditions is difficult.

5. **Title: Face detection and Recognition: A review**

   Year: 2018

   Author: Jashanpreet Kaur, Akansha, Harjeet Singh

   Inferences:

   The physiological characteristics of the human face with relevance to various expressions are associated with geometrical structures which are restored as base matching templates for the recognition system.

   Limitations:

   1:N problem where N is data in database

   Data privacy of stored data should be maintained

6. **Title: Object Detection Based on the Improved Single Shot MultiBox Detector**

   Year: 2019

   Author: Songmin Jia, Chentao Diao, Guoliang Zhang, Ao Dun

   Inferences:

   SSD object detection algorithms performs better than SSD and faster RCNN algorithms.

   Limitations:

   Choosing the correct layers among different neural network layers

   The increase of dimension proportional to increase of noise

7. **Title: Object detection in real time based on improved single shot multibox detector algorithm**

   Year: 2020

   Author: Ashwani Kumar, Justin Zhang, Hongbo Lyu

   Inferences:

   The accuracy of the proposed model is supposed to be more than 79.8%.

   Future research can extend the proposed algorithms.

   Limitations:

   Problem can occur when anonymous object is present in the image.

   Blockage, deformable objects

8. **Title: Face Recognition for the Visually Impaired**

   Year: 2020

   Author: Rabia Jafri, Syed Abid Ali, Hamid Arabnia

   Inferences:

   An overview of several systems being developed to aid this population in the face recognition task was presented in this paper.

   Limitations:

   The system should be portable allowing the user to carry it to different venues

   It should be as inconspicuous as possible

9. **Title: An Integrated Expert System with a Supervised Machine Learning based Probabilistic Approach to Play Tic-Tac-Toe**

   Year: 2021

   Author: Muhammad Sakib Khan Inan, Rizwan Hasan, Tabia Tanzin Prama

   Inferences:

   In this study, we studied a unique approach for generating a strategic Tic-Tac-Toc artificial agent in a probabilistic way by merging supervised machine learning with an integrated rule-based framework, with the objective of developing a fast and scalable AI Agent. This study extends on the use of supervised machine learning in games with probabilistic predictions.

   Limitations:

   If the AI agent initiates the game by making the initial move, it is only the Extreme Gradient Boosting (XGBoost) algorithm that can make a draw against the never losing Minimax algorithm.

   If the Adaboost algorithm loses every game against Minimax in these scenarios.

10. **Title: FaceNet: A Unified Embedding for Face Recognition and Clustering**

Year: 2020

Author: Florian Schroff, Dmitry Kalenichenko, James Philbin

Inferences:

In all our experiments we train the CNN using Stochastic Gradient Descent (SGD) with standard backprop and AdaGrad.

Limitations:

These are very interesting findings and it is somewhat surprising that it works so well. Future work can explore how far this idea can be extended. Presumably there is a limit as to how much the v2 embedding can improve over v1, while still being compatible. Additionally it would be interesting to train small networks that can run on a mobile phone and are compatible to a larger server side model.

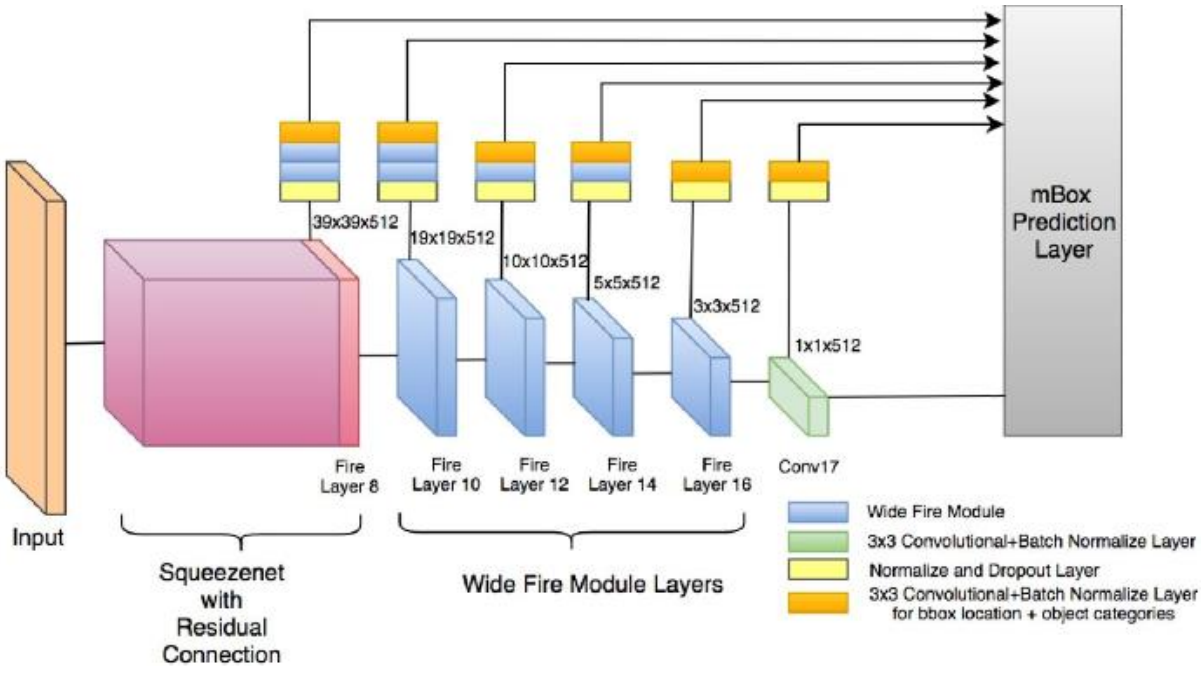
Algorithms Used:

For Smart Vision we have used the Single Shot Detection(SSD) algorithm.

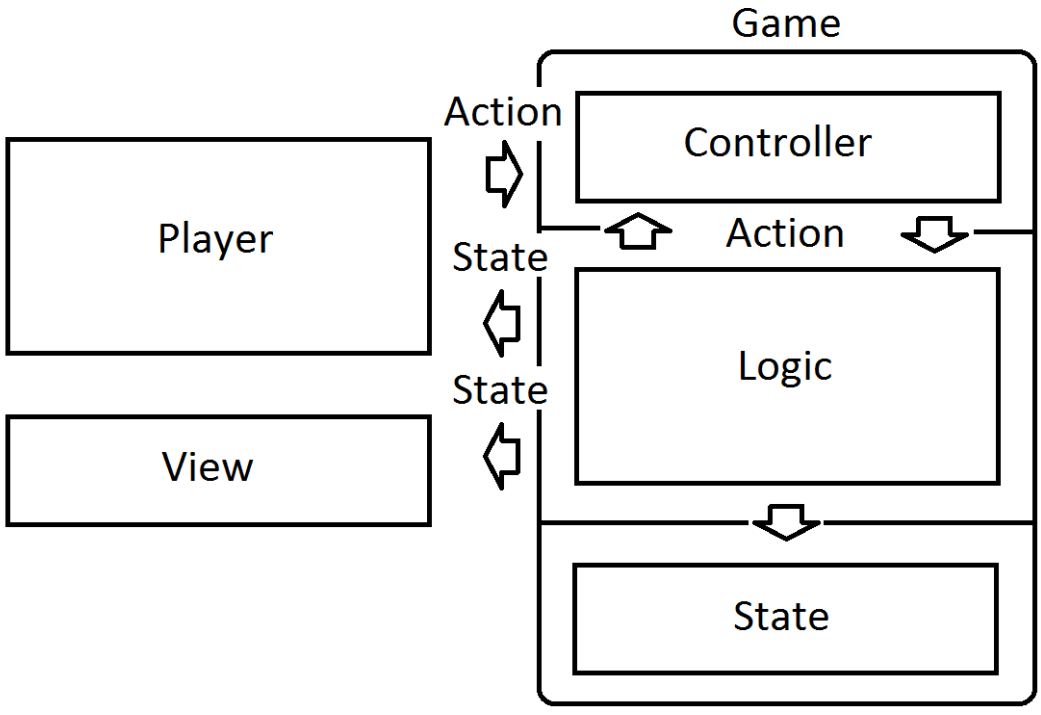
For Tic-Tac-Toe we have used the Recurrent Neural Network(RNN) algorithm.

Architectural Diagram:

For Smart Vision:



For Tic-Tac-Toe:



Parameters Used:

The algorithm we are using for the detection of objects is single shot detection(SSD) algorithm as it achieves a higher percentage of accuracy when compared to the next best algorithm which is R-CNN.

Recurrent Neural Network(RNN), these algorithms are ideal for sequential data like speech because they're able to "remember" what came before and use their previous output as input for their next move. Since words generally appear in the context of a sentence, knowing what came before and recycling that information into the next prediction goes a long way towards accurate speech recognition.

Experimental Setup and Softwares:

The experimental setup are as follows:-

Hardware

Intel core processors i5 and above

Modern Operating System: Windows 7 or 10
x86 64-bit CPU (Intel / AMD architecture)
4 GB RAM
5 GB free disk space

Softwares and Modules used

Python
Open CV
Numpy
Tensor Flow
React Modules
Node Modules
Yarn server

Methodology and Implementation

- Object Detection: This project aims to do real-time object detection through a laptop camera or webcam using OpenCV and MobileNetSSD.
- Voice Based Examination: We will implement this feature using Google's Text to Speech API.
- Voice Input Audio: React Speech Recognition.
- Tic Tac Toe Game: Javascript or Python react.
- Web Development
  Design: Figma for wireframing, design and prototype.
  Frontend: ReactJs
  Backend: NodeJs or Django
  Server: yarn
- The training of both the models is done through COCOMO dataset required for our project i.e. single shot detection(SSD) and Recurrent Neural Network(RNN).

Implementation

Code

Index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"
      content="Web site created using create-react-app"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <!--
      manifest.json provides metadata used when your web app is installed on a
      user's mobile device or desktop. See
https://developers.google.com/web/fundamentals/web-app-manifest/
    -->
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <!--
      Notice the use of %PUBLIC_URL% in the tags above.
```

```
    It will be replaced with the URL of the `public` folder during the build.
    Only files inside the `public` folder can be referenced from the HTML.

    Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
    work correctly both with client-side routing and a non-root public URL.
    Learn how to configure a non-root public URL by running `npm run build`.
  -->
  <title>React App</title>
 </head>
 <body>
  <noscript>You need to enable JavaScript to run this app.</noscript>
  <div id="root"></div>
  <!--
    This HTML file is a template.
    If you open it directly in the browser, you will see an empty page.

    You can add webfonts, meta tags, or analytics to this file.
    The build step will place the bundled scripts into the <body> tag.

    To begin the development, run `npm start` or `yarn start`.
    To create a production bundle, use `npm run build` or `yarn build`.
  -->
 </body>
</html>
```

Manifest.json

```
{
 "short_name": "React App",
 "name": "Create React App Sample",
 "icons": [
   {
    "src": "favicon.ico",
    "sizes": "64x64 32x32 24x24 16x16",
    "type": "image/x-icon"
   },
   {
    "src": "logo192.png",
    "type": "image/png",
    "sizes": "192x192"
   },
   {
    "src": "logo512.png",
    "type": "image/png",
    "sizes": "512x512"
   }
 ],
 "start_url": ".",
 "display": "standalone",
 "theme_color": "#000000",
 "background_color": "#ffffff"
```

```
}
```

Header.js

```
import logo from '../assets/logo.png';

import { Link } from 'react-router-dom';
import './Header.scss';

const Header = () => {
   return (
      <header>
         <Link to="/">
            <img src={logo} alt="logo" />
         </Link>

         <nav>
            <Link to="/">Home</Link>
            <Link to="/detection">Detect Object</Link>
            <Link to="/quiz">Exam</Link>
            <Link to="/board">Game</Link>
         </nav>
      </header>
   );
}

export default Header;
```

Board.js

```
import "./Board.scss";
import { useSpeechSynthesis, useSpeechRecognition } from "react-speech-kit";
import { useState, useEffect, useRef } from "react";

const Board = () => {
   const { speak, speaking, cancel } = useSpeechSynthesis();

   const [value, setValue] = useState("");

   const [reset, setReset] = useState(false);

   const [winner, setWinner] = useState('');

   const { listen, stop } = useSpeechRecognition({
      onResult: (result) => {
         console.log(result)
         setValue(result);
      },
   });
   useEffect(() => {
      switch (value) {
         case "1":
```

```
            case "top left":
            draw(1);
            break;
        case "2":
            case "top":
            draw(2);
            break;
        case "3":
            case "top right":
            draw(3)
            break;
        case "4":
            case "left":
            draw(4)
            break;
        case "5":
            case "centre":
            draw(5)
            break;
        case "6":
            case "right":
            draw(6)
            break;
        case "7":
            case "bottom left":
            draw(7)
            break;
        case "8":
            case "bottom":
            draw(8)
            break;
        case "9":
            case "bottom right":
            draw(9)
            break;
        case "stop":
            stop();
            speak({text:"Stopping"});
            break;
        case "reset":
            setReset(true);
            break;
        default:
            break;
    }
}, [value])

// Creating a turn state, which indicates the current turn
const [turn, setTurn] = useState(0);
```

```
// Creating a data state, which contains the
// current picture of the board
const [data, setData] = useState(["", "", "", "", "", "", "", "", ""]);

// Creating a reference for the board
const boardRef = useRef(null);

useEffect(async () => {
   if (winner === "") {
      await speak({ text: "Player " + (turn + 1) + "'s Turn" });
   } else {
      await speak({ text: winner });
   }
}, [turn, winner, setWinner]);

// Function to draw on the board
const draw = (index) => {
   if(speaking){
      cancel();
   }
   // Draws only if the position is not taken
   // and winner is not decided yet
   if (data[index - 1] === "" && winner === "") {
      // Draws X if it's player 1's turn else draws O
      const current = turn === 0 ? "X" : "O";

      // Updating the data state
      data[index - 1] = current;

      //Drawing on the board
      boardRef.current.querySelectorAll(".input")[index - 1].innerText = current;

      // Setting the winner in case of a win
      if (checkWin()) {
         setWinner(turn === 0 ? "Player 1 Wins!" : "Player 2 Wins!");
      } else if (checkTie()) {
         // Setting the winner to tie in case of a tie
         setWinner("It's a Tie!");
      } else {
         // Switching the turn
         setTurn(turn == 0 ? 1 : 0);
      }
   } else {
      let player = data[index - 1] === "X" ? "Player 1" : "Player 2";
      speak({ text: "Invalid Move, box occupied by " + player });
   }
};

// Checks for the win condition in rows
const checkRow = () => {
```

```
    let ans = false;
    for (let i = 0; i < 9; i += 3) {
        ans |=
            data[i] === data[i + 1] && data[i] === data[i + 2] && data[i] !== "";
    }
    return ans;
};

// Checks for the win condition in cols
const checkCol = () => {
    let ans = false;
    for (let i = 0; i < 3; i++) {
        ans |=
            data[i] === data[i + 3] && data[i] === data[i + 6] && data[i] !== "";
    }
    return ans;
};

// Checks for the win condition in diagonals
const checkDiagonal = () => {
    return (
        (data[0] === data[4] && data[0] === data[8] && data[0] !== "") ||
        (data[2] === data[4] && data[2] === data[6] && data[2] !== "")
    );
};

// Checks if at all a win condition is present
const checkWin = () => {
    return checkRow() || checkCol() || checkDiagonal();
};

// Checks for a tie
const checkTie = () => {
    let count = 0;
    data.forEach((cell) => {
        if (cell !== "") {
            count++;
        }
    });
    return count === 9;
};

// UseEffect hook used to reset the board whenever
// a winner is decided
useEffect(async () => {
    // Clearing the data state
    setData(["", "", "", "", "", "", "", "", ""]);

    // Getting all the children(cells) of the board
    const cells = boardRef.current.children;
```

```
      // Clearing out the board
      for (let i = 0; i < 9; i++) {
        cells[i].innerText = "";
      }

      // Resetting the turn to player 0
      setTurn(0);

      // Resetting the winner
      setWinner("");
      setReset(false);
    }, [reset, setReset, setWinner]);

    const initGame = async () => {
      await speak({
        text: "Welcome to Tic Tac Toe, you can either click on the boxes or say the number of the box you want to click on. The boxes range from 1 to 9. Player 1 will go first",
      });
      await listen({ interimResults: true });
    };

    return (
      <div className="board-container">
        <div className={`winner ${winner !== '' ? '' : 'shrink'}`}>
          {/* Display the current winner */}
          <div className='winner-text'>{winner}</div>
          {/* Button used to reset the board */}
          <button onClick={() => setReset(true)}>
            Reset Board
          </button>
        </div>
        <div ref={boardRef} className="board">
          <div className="input input-1" onClick={() => draw(1)}></div>
          <div className="input input-2" onClick={() => draw(2)}></div>
          <div className="input input-3" onClick={() => draw(3)}></div>
          <div className="input input-4" onClick={() => draw(4)}></div>
          <div className="input input-5" onClick={() => draw(5)}></div>
          <div className="input input-6" onClick={() => draw(6)}></div>
          <div className="input input-7" onClick={() => draw(7)}></div>
          <div className="input input-8" onClick={() => draw(8)}></div>
          <div className="input input-9" onClick={() => draw(9)}></div>
        </div>
        <div
          className="speech-button"
          onClick={() => {
            initGame();
          }}
        >
          Start Game
```

```
            </div>
        </div>
    );
};

export default Board;
```

Landing.js

```
import './Landing.scss';
import { useSpeechSynthesis, useSpeechRecognition } from "react-speech-kit";
import { useState, useEffect } from "react";
import { useNavigate } from "react-router-dom";

const Landing = () => {

    let navigate = useNavigate();

    const [value, setValue] = useState("");

    const { speak, speaking, cancel } = useSpeechSynthesis();

    const { listen, stop } = useSpeechRecognition({
        onResult: (result) => {
            console.log(result)
            setValue(result);
        },
    });
    useEffect(() => {
        switch (value) {
            case "1":
                case "detect object":
                    navigate('/detection');
                    stop();
                break;
            case "give exam":
                    navigate('/quiz');
                    stop();
                break;
            case "play game":
                    navigate('/board');
                    stop();
                break;
            case "stop":
                stop();
                break;
            default:
                break;
        }
    }, [value])
```

```
  const initListening = async () => {
    await speak({
      text: "Welcome to assisted vision, we are listening to you",
    })

    listen({ interimResults: true });
  };

  return (
    <div className="Landing">
      <div className="Landing-container">
        <div className='left-container'>
          <h1>Assisted Vision</h1>
          <h4>
            One stop solution to facilitate effortless learning for our sight impaired students. Enhancing a number of cognitive skills, Our technology includes teaching through unifying experiences.
          </h4>
          <button onClick={()=>{initListening()}} className="Landing-button">Give Command</button>
          <h6>like detect object, give exam, play game</h6>
        </div>
        <div className='right-container'>
          <img src={require('../assets/landing.png')} alt="landing" />
        </div>
      </div>
    </div>
  );
}

export default Landing;
```

Objectdetection.js

```
// Import dependencies
import React, { useRef, useState, useEffect } from "react";
import * as cocossd from "@tensorflow-models/coco-ssd";
import Webcam from "react-webcam";
import "./ObjectDetection.scss"
import { drawRect } from "../utilities/detection";
import { useSpeechSynthesis } from "react-speech-kit";

const ObjectDetection = () => {
  const webcamRef = useRef(null);
  const canvasRef = useRef(null);
  const { speak, speaking, cancel } = useSpeechSynthesis();
  const [loading, setLoading] = useState(true);

  // Main function
  const runCoco = async () => {
```

```
    const net = await cocossd.load();
    console.log("Model loaded.");
    setInterval(() => {
      detect(net);
    }, 10);
  };

  const detect = async (net) => {
    if (
      typeof webcamRef.current !== "undefined" &&
      webcamRef.current !== null &&
      webcamRef.current.video.readyState === 4
    ) {
      setLoading(false);
      const video = webcamRef.current.video;
      const videoWidth = webcamRef.current.video.videoWidth;
      const videoHeight = webcamRef.current.video.videoHeight;

      webcamRef.current.video.width = videoWidth;
      webcamRef.current.video.height = videoHeight;

      canvasRef.current.width = videoWidth;
      canvasRef.current.height = videoHeight;

      const obj = await net.detect(video);

      const ctx = canvasRef.current.getContext("2d");
      drawRect(obj, ctx, speak, speaking, cancel);
    }
  };

  useEffect(() => { runCoco() }, []);

  return (
    <div className="ObjectDetection">
      <div className="webcam-div">
        <Webcam
          ref={webcamRef}
          muted={true}
          style={{
            position: "absolute",
            marginLeft: "auto",
            marginRight: "auto",
            left: 0,
            right: 0,
            textAlign: "center",
            zindex: 9,
            width: 640,
            height: 480,
          }}
```

```
            />

            <canvas
               ref={canvasRef}
               style={{
                  position: "absolute",
                  marginLeft: "auto",
                  marginRight: "auto",
                  left: 0,
                  right: 0,
                  textAlign: "center",
                  zindex: 8,
                  width: 640,
                  height: 480,
               }}
            />
         </div>
         {loading &&
            <h3>Loading</h3>
         }
      </div>
   );
}

export default ObjectDetection;
```

Question.js

```
export default [
   {
      question: "Sample Question 1",
      options: ["Option 1", "Option 2", "Option 3", "Option 4"],
      answer: "Option 1"
   },
   {
      question: "Sample Question 2",
      options: ["A veryyyyyyyyyyyy long option", "Superrrrrrrrrrrrrrrrrrrrrrr longgggggggggggggggggggggggggggggggggggg option which has a lot of words for testing purposes", "Option 3", "Option 4"],
      answer: "Superrrrrrrrrrrrrrrrrrrrrrr longgggggggggggggggggggggggggggggggggggg option which has a lot of words for testing purposes"
   },
   {
      question: "Sample Question 3",
      options: ["Option 1", "Option 2", "Option 3", "Option 4"],
      answer: "Option 3"
   },
   {
      question: "Sample Question 4",
      options: ["Option 1", "Option 2", "Option 3", "Option 4"],
      answer: "Option 4"
```

```
    }
]
```

Quiz.js

```
import { useEffect, useState } from 'react';
import { useSpeechSynthesis, useSpeechRecognition } from "react-speech-kit";
import questions from './questions.js';
import './Quiz.scss'

const Quiz = () =>{
    const [question, setQuestion] = useState(0);
    const [value, setValue] = useState("");
    const [selected, setSelected] = useState([]);

    const { speak, speaking, cancel } = useSpeechSynthesis();

    const { listen, stop } = useSpeechRecognition({
        onResult: (result) => {
            console.log(result)
            setValue(result);
        },
    });

    const selectAnswer = async (i) => {
        let arr = [...selected]
        let option = questions[question].options[i];
        arr[question] = option;
        await speak({text: "Selected " + option, queue: false});
        setSelected(arr);
    }

    const startListening = async () => {
        await speak({text: "We are listening to you", queue: false});
        await readQuestionAndOptions();
        await speak({text: "Please select an option", queue: false});
        listen();
    }

    const readQuestionAndOptions = async () => {
        await speak({text: "current question: ", queue: true});
        await speak({text: questions[question].question, queue: true});
        return;
    }

    useEffect(() => {
        let arr = []
        for(let i=0; i<questions.length; i++){
            arr.push("")
        }
        setSelected(arr)
```

```
}, []);

useEffect(()=>{
   switch (value) {
      case "read question":
         speak({text: questions[question].question, queue: false});
         break;

      case "read options":
         speak({text: questions[question].options, queue: false});
         break;

      case "next question":
         changeQuestion("next")
         break;

      case "previous question":
         changeQuestion("prev")
         break;

      case "read option 1":
         speak({text: questions[question].options[0], queue: false});
         break;

      case "read option 2":
         speak({text: questions[question].options[1], queue: false});
         break;

      case "read option 3":
         speak({text: questions[question].options[2], queue: false});
         break;

      case "read option 4":
         speak({text: questions[question].options[3], queue: false});
         break;

      case "select option 1":
         selectAnswer(0);
         break;

      case "select option 2":
         selectAnswer(1);
         break;

      case "select option 3":
         selectAnswer(2);
         break;

      case "select option 4":
         selectAnswer(3);
```

```
            break;

        case "read selected option":
            const option = selected[question]
            speak({text: option!="" ? option:"No option selected", queue: false});
            break;

        case "read all selected options":
            for(let i=0; i<selected.length; i++){
                const option = selected[i]
                speak({text: "Question: " + questions[i].question, queue: true});
                speak({text: option!="" ? "Option selected: " + option:"No option selected", queue: true});
            }
            break;

        case "stop":
            stop();
            speak({text: "Stopped", queue: false});
            break;

        default:
            break;
    }
  },[value])

  useEffect(()=>{
    readQuestionAndOptions();
  },[question])

  const changeQuestion = (where) =>{
    if(where === 'next' && question < questions.length-1){
      setQuestion(question+1);
    }else if(where === 'prev' && question > 0){
      setQuestion(question-1);
    }
  }

  return(
    <div className="Quiz">
      <h1>Quiz</h1>
      <div className="question-section">
        <p className="question"> {questions[question].question}</p>
        <div className="options">
          {questions[question].options.map((option, i)=>{
            return(
              <button className={`option ${selected[question] === option ? "selected" : ""}`} key={i} onClick={()=>{selectAnswer(i)}}>
                {option}
              </button>
```

```
                )
              })}
            </div>
          </div>
          <div className="bottom-buttons">
            <div onClick={()=>{changeQuestion("prev")}} className="bottom-button">Prev Q</div>
            <div onClick={()=>{startListening()}} className="bottom-button voice">Start Voice Recognition</div>
            <div onClick={()=>{changeQuestion("next")}} className="bottom-button">Next Q </div>
          </div>
        </div>
    )
}

export default Quiz;
```

Index.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);
```

App.js

```
import {
    BrowserRouter as Router,
    Routes,
    Route,
    Link
  } from "react-router-dom";

// Importing the required components
import Board from './containers/Board';
import Quiz from './containers/Quiz';
import Header from "./components/Header";
import Landing from "./containers/Landing";
import ObjectDetection from "./containers/ObjectDetection";

// Importing the CSS File
import "./App.css";

function App() {
    return (
```

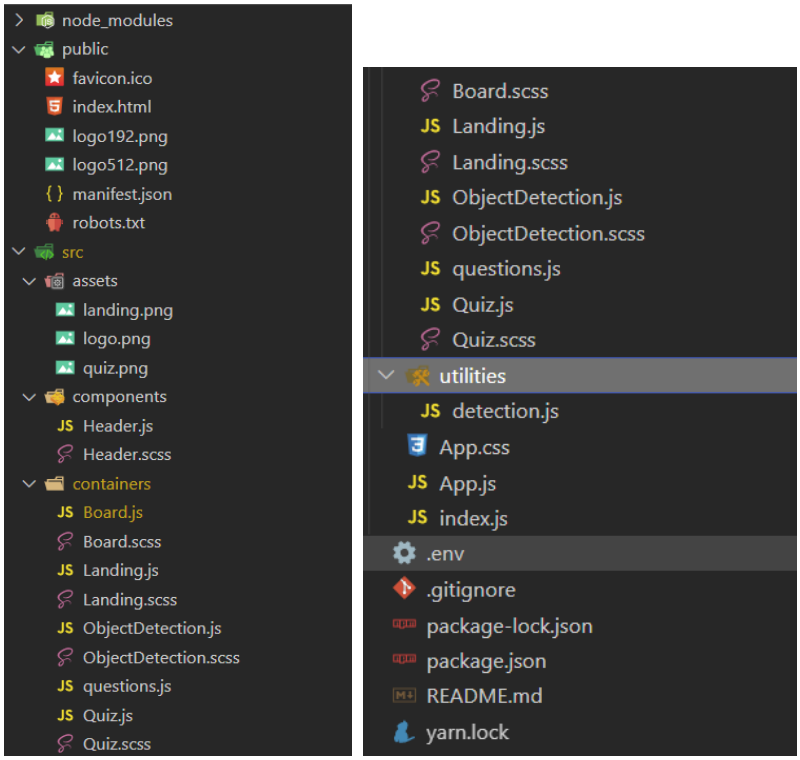
```
      <div className="App">
        <Router>
          <Header/>
          <Routes>
            <Route path="/detection" element={<ObjectDetection/>} />
            <Route path="/board" element={<Board />}/>
            <Route path="/quiz" element={<Quiz />}/>
            <Route path="/" element={<Landing />}/>
          </Routes>
        </Router>
      </div>
    );
}

export default App;
```

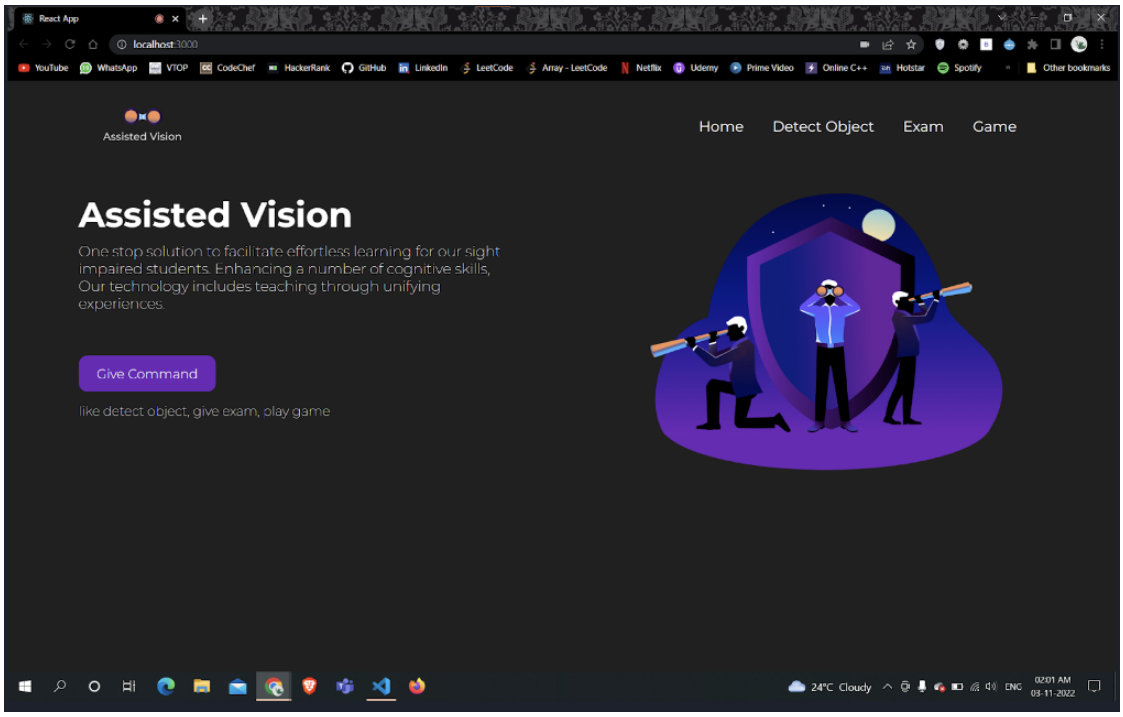
Code Link:

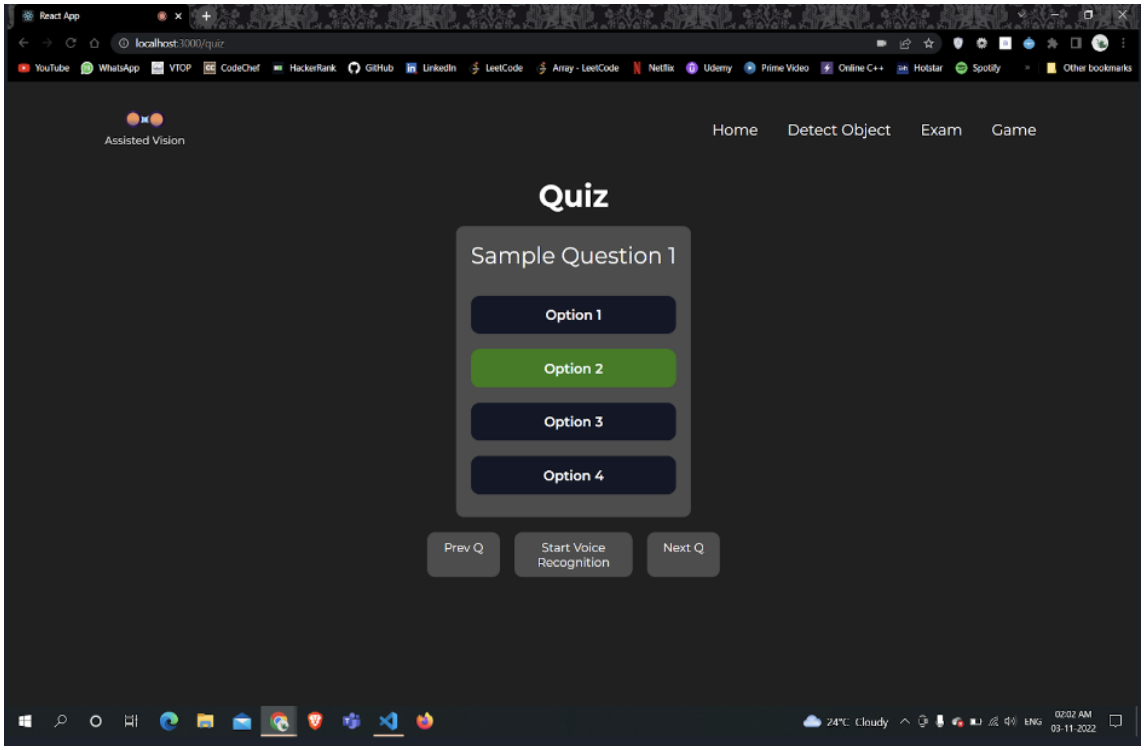
https://github.com/themuditjain/Assisted-Vision---A-Smart-Eye

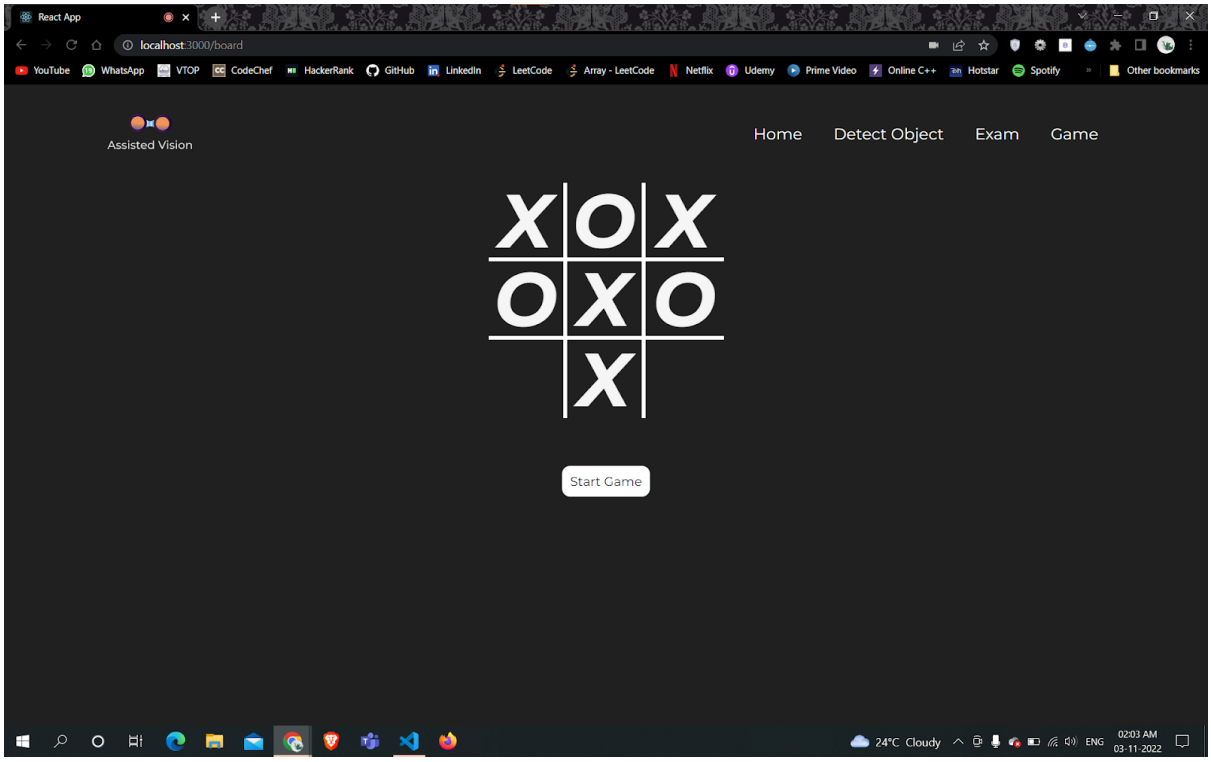
List of All files used in the code:

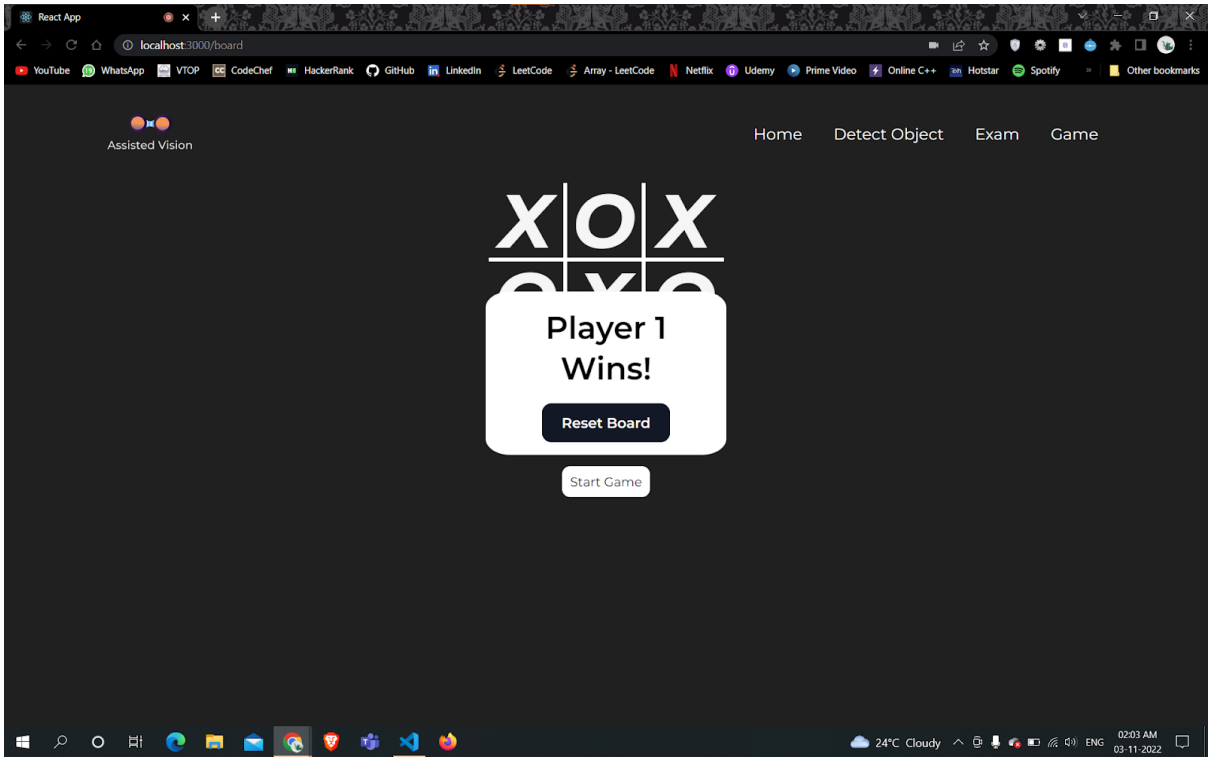


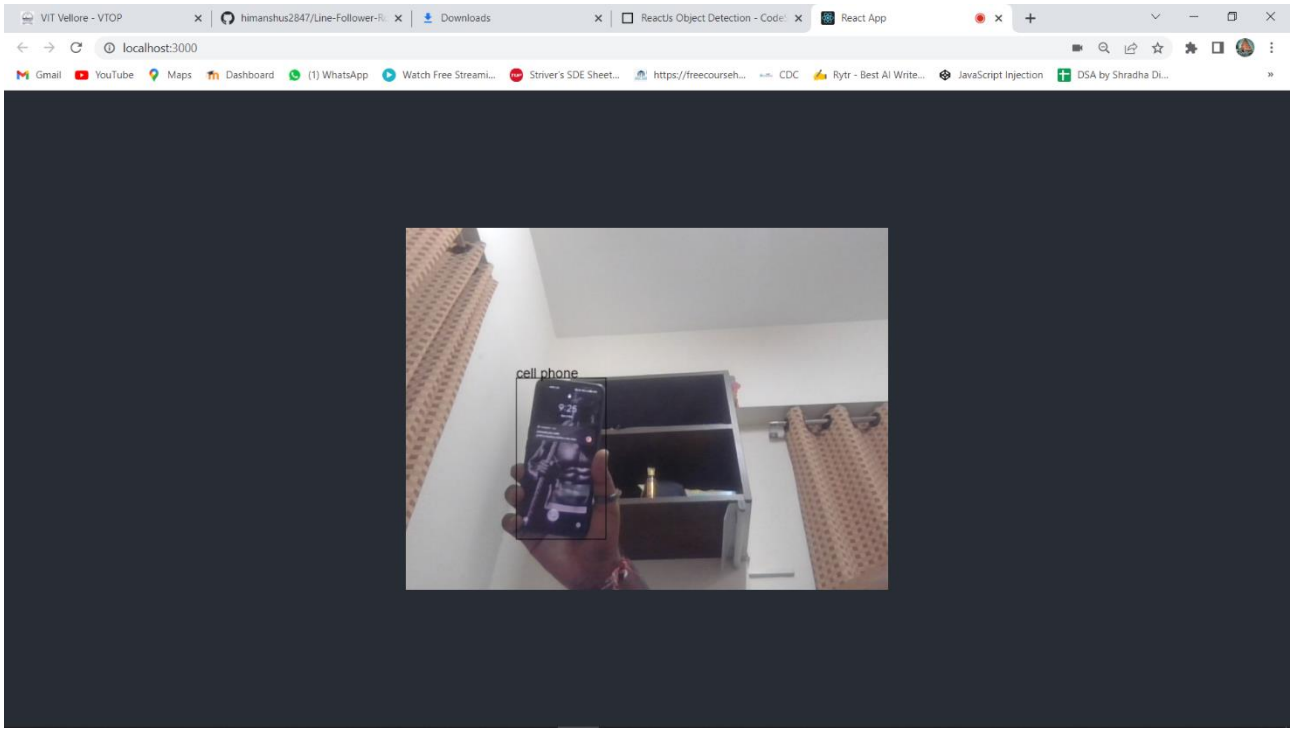
Results:

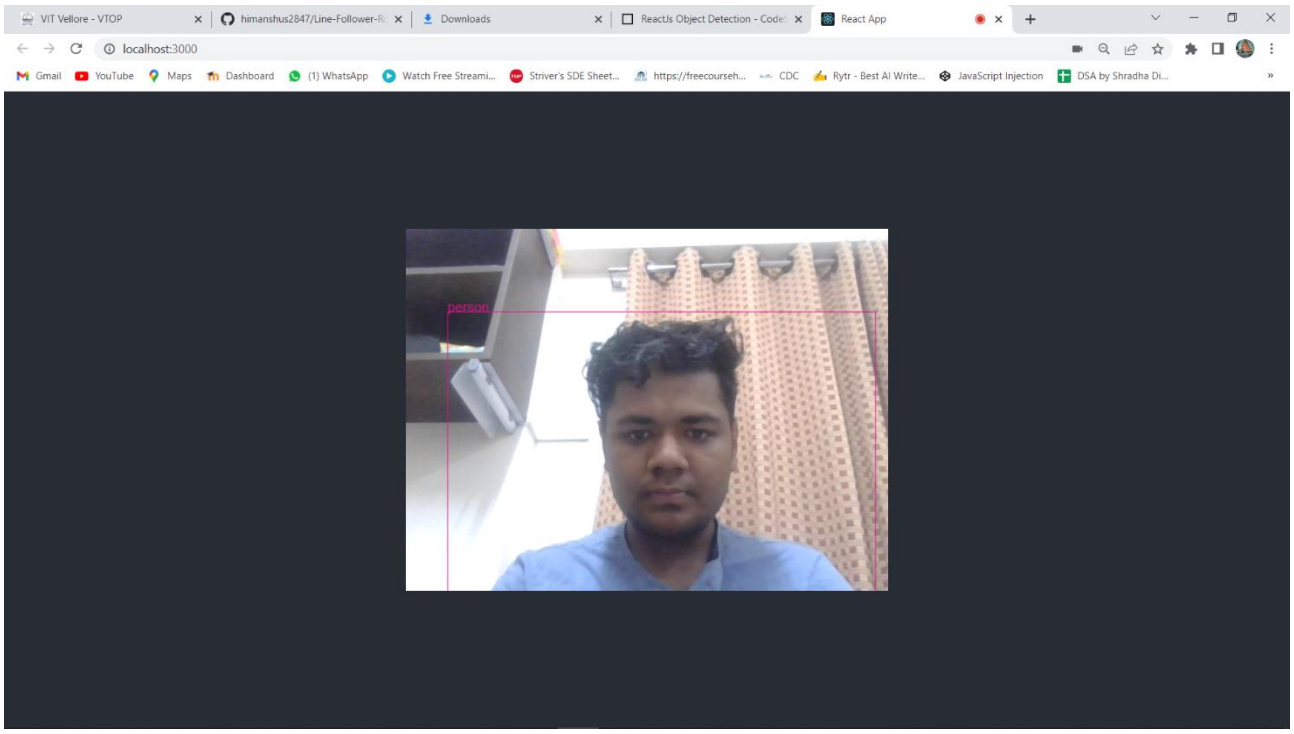

React App
localhost:3000
Assisted Vision
Home
Detect Object
Exam
Game
Assisted Vision
One stop solution to facilitate effortless learning for our sight impaired students. Enhancing a number of cognitive skills, Our technology includes teaching through unifying experiences.
Give Command
like detect object, give exam, play game

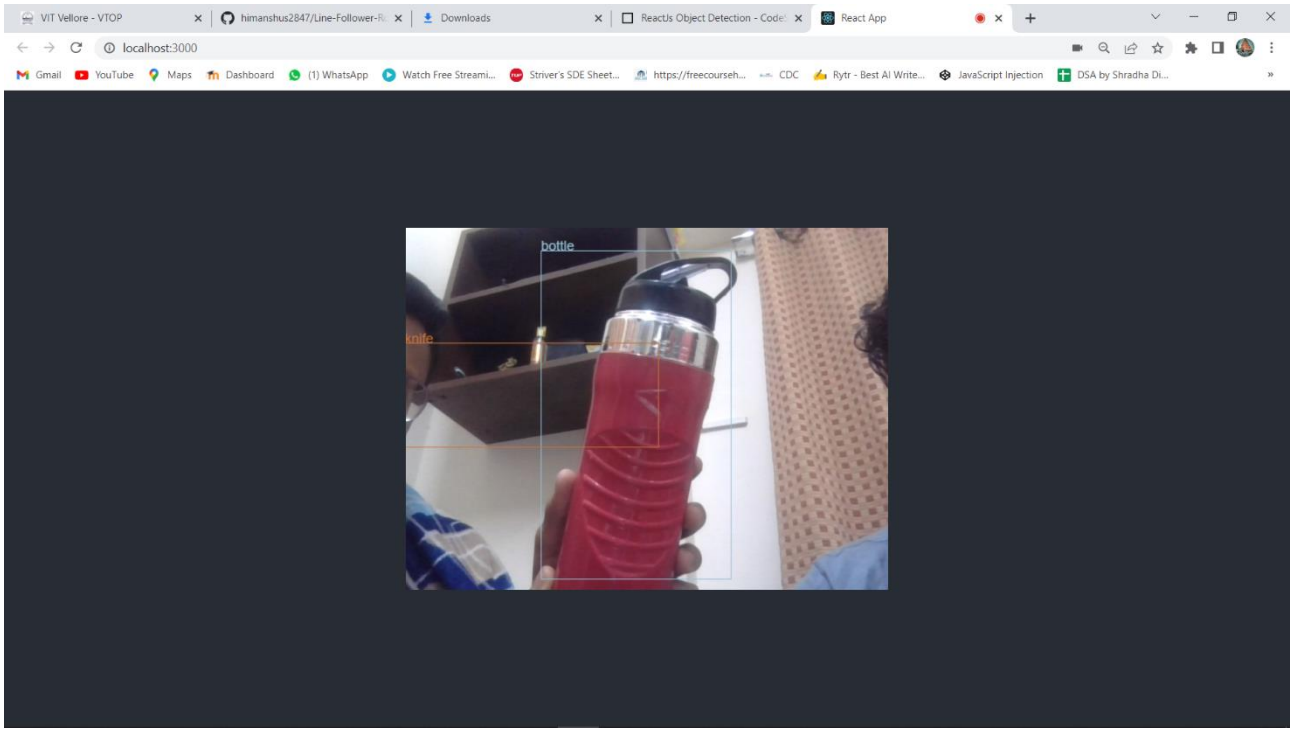

React App
localhost:3000/quiz
Assisted Vision
Home
Detect Object
Exam
Game
Quiz
Sample Question 1
Option 1
Option 2
Option 3
Option 4
Prev Q
Start Voice Recognition
Next Q

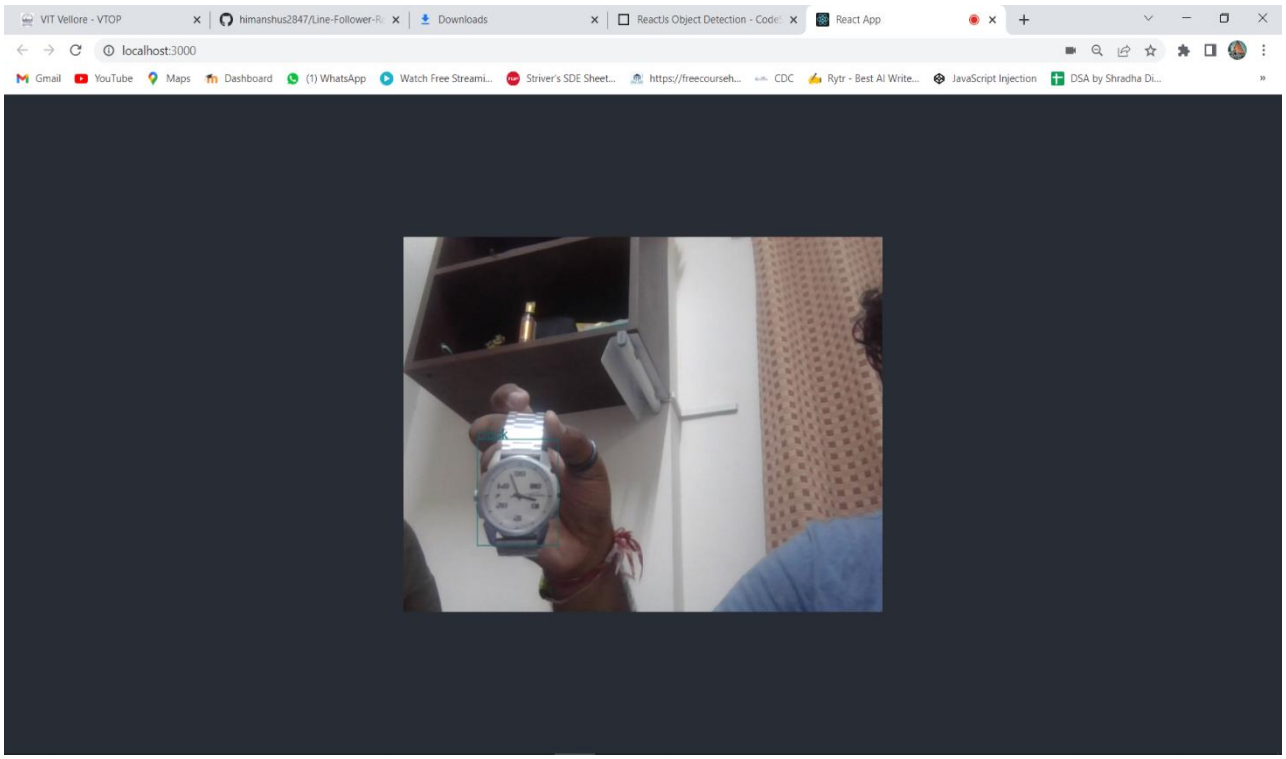

React App
localhost:3000/board
YouTube WhatsApp VTOP CodeChef HackerRank GitHub LinkedIn LeetCode Array - LeetCode Netflix Udemy Prime Video Online C++ Hotstar Spotify Other bookmarks
Assisted Vision
Home Detect Object Exam Game
X O X
O X O
X
Start Game
24°C Cloudy ENG 02:03 AM 03-11-2022


React App
localhost:3000/board
YouTube WhatsApp VTOP CodeChef HackerRank GitHub LinkedIn LeetCode Array - LeetCode Netflix Udemy Prime Video Online C++ Hotstar Spotify Other bookmarks
Assisted Vision
Home Detect Object Exam Game
X O X
Player 1 Wins!
Reset Board
Start Game
24°C Cloudy ENG 02:03 AM 03-11-2022


VIT Vellore - VTOP
himanshus2847/Line-Follower-R
Downloads
ReactJs Object Detection - Code
React App
localhost:3000
Gmail
YouTube
Maps
Dashboard
(1) WhatsApp
Watch Free Streami...
Striver's SDE Sheet...
https://freecourseh...
CDC
Rytr - Best AI Write...
JavaScript Injection
DSA by Shradha Di...
cell phone


VIT Vellore - VTOP
himanshus2847/Line-Follower-R
Downloads
ReactJs Object Detection - Code
React App
localhost:3000
Gmail
YouTube
Maps
Dashboard
(1) WhatsApp
Watch Free Streami...
Striver's SDE Sheet...
https://freecourseh...
CDC
Rytr - Best AI Write...
JavaScript Injection
DSA by Shradha Di...
person


VIT Vellore - VTOP
himanshus2847/Line-Follower-R
Downloads
ReactJs Object Detection - Code
React App
localhost:3000
Gmail
YouTube
Maps
Dashboard
(1) WhatsApp
Watch Free Streami...
Striver's SDE Sheet...
https://freecourseh...
CDC
Rytr - Best AI Write...
JavaScript Injection
DSA by Shradha Di...
bottle
knife


VIT Vellore - VTOP
himanshus2847/Line-Follower-R
Downloads
ReactJs Object Detection - Code
React App
localhost:3000
Gmail
YouTube
Maps
Dashboard
(1) WhatsApp
Watch Free Streami...
Striver's SDE Sheet...
https://freecourseh...
CDC
Rytr - Best AI Write...
JavaScript Injection
DSA by Shradha Di...

Discussion

For converting our input i.e. speech to the computer language so that it can understand it and respond according to it we have used the React modules across our project the main Module which is used for converting the speech to the react is

React-speech-kit

And the modules which are performing here and completing our purpose is:
useSpeechSynthesis and useSpeechRecognition

Real Time Applications

As visual impairment is a serious issue, it is obvious that there have been many attempts to help the affected people in the past like the development of Braille in 1819.
But, with the development of technologies like artificial intelligence and machine learning, we now have much more sophisticated systems like -

-Aira Headset
-Argus II
-OrCam MyEye
-BrainPort

Future Aspects

- During this project we won't be making any of the hardware device but in future we can incorporate IOT in this working in collaboration with someone from hardware company.
- We can incorporate many other games in it.
- In future we can also make an application where the camera application will auto detect the objects which is in front of it.

References

- https://www.mdpi.com/2076-3417/12/5/2308/htm
- https://www.researchgate.net/publication/323390774_Face_detection_and_Recognition_A_review#:~:text=Abstract%20and%20Figures,digital%20cameras%20and%20social%20tagging
- https://www.researchgate.net/publication/332942398_Object_Detection_Based_on_the_Improved_Single_Shot_MultiBox_Detector
- https://www.researchgate.net/publication/344775420_Object_detection_in_real_time_based_on_improved_single_shot_multi-box_detector_algorithm
- https://www.researchgate.net/publication/281870946_Face_recognition_for_the_visually_impaired
- https://www.who.int/en/news-room/fact-sheets/detail/blindness-and-visual-impairment
- Arun Kumar Ravula1* , Pallepati Vasavi2 , K.Ram Mohan Rao3 1*Department of Computer Science and Engineering, Osmania University College of Engineering, Osmania University, Hyderabad, Telangana, India, 2Department of Computer Science and Engineering, Sreenidhi Institute of Science and Technology, Hyderabad, Telangana, India 3Department of Computer Science and Engineering, Vasavi College of Engineering, Hyderabad, Telangana, India,
- Andrés A. Díaz-Toro , 1 Sixto E. Campaña-Bastidas , 1 and Eduardo F. Caicedo-Bravo 2 1 School of Basic Sciences, Technology and Engineering ECBTI, Universidad Nacional Abierta y a Distancia UNAD, Pasto 520001, Colombia 2 School of Electrical and Electronic Engineering EIEE, Universidad del Valle, Cali 76001, Colombia Correspondence should be addressed to Andrés A. Díaz-Toro; andres.diaz@unad.edu.co Received 1 December 2020; Revised 18 February 2021; Accepted 20 February 2021; Published 5 March 2021 Academic Editor: Eduard Llobet