/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**NAME :-** Shriram Hemant Salunke

**GR.No.:-** 21910486

**Roll.No:-**233054

**Topic:-** Skill Competency Exam

**Under the guidance :-** **Suvarna Pawar Maam**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**Aim:- To make more efficient and easy online examination System during this pandemic of covid19.**

**Objectives:-**

The main **objective** of the  **Project** on **Examination Management System** is to manage the details of **Examinations**, Papers, Courses, Marks, Faculties. It manages all the information about **Examinations**, Students, Faculties, **Examinations**.

**Part 1:- Online Signup, Login, Exam(Text), Database**

**Technolgy used:**

HTML

CSS

JavaScript

ReactJS

ExpressJS

mongoDB

**Code:**

**Home.js**

**import React from 'react'**

**import {useState, useEffect} from 'react'**

**import {useHistory} from 'react-router-dom'**

**import axios from 'axios'**

**import {Jumbotron, Button} from 'react-bootstrap'**

**import './../../bootstrap.min.css'**

**import './../../App.css'**

**import BackgroundImage from './../images/bg4.jpg'**

**const Home = () => {**

**const [user, setUser] = useState({name : "", test1Attempted : false, test2Attempted : false});**

**const history = useHistory();**

**const loadData = async () =>**

**{**

**try**

**{**

**const res = await axios.get("http://localhost:7000/getcurrentuser");**

**console.log(res.data);**

**setUser(res.data);**

**}catch(err)**

**{**

**console.log(err);**

**}**

**}**

**useEffect( () =>**

**{**

**loadData()**

**}, []);**

**const [showTest1Score, setShowTest1Score] = useState(false);**

**const [showTest2Score, setShowTest2Score] = useState(false);**

**const ToogleShowTest1Score = () =>**

**{**

**if(showTest1Score === false)**

**{**

**setShowTest1Score(true);**

**}**

**else**

**{**

**setShowTest1Score(false);**

**}**

**console.log(user.test1Score);**

**}**

**const ToogleShowTest2Score = () =>**

**{**

**if(showTest2Score === false)**

**{**

**setShowTest2Score(true);**

**}**

**else**

**{**

**setShowTest2Score(false);**

**}**

**console.log(user.test2Score);**

**}**

**const onLogoutClick = () =>**

**{**

**history.push("/");**

**}**

**const onTakeTest1 = () =>**

**{**

**history.push("/test1");**

**}**

**const onTakeTest2 = () =>**

**{**

**history.push("/test2");**

**}**

**return (**

**<div style={{backgroundImage:`url(${BackgroundImage})`, height:"1080px"}}>**

**<div>**

**<nav className="navbar navbar-expand-lg navbar-dark bg-primary">**

**<a className="navbar-brand" href="/home">Online Exam Project</a>**

**<button className="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">**

**<span className="navbar-toggler-icon"></span>**

**</button>**

**<div className="collapse navbar-collapse" id="navbarSupportedContent">**

**<ul className="navbar-nav mr-auto">**

**<li className="nav-item active">**

**<a className="nav-link" href="/home">Home <span className="sr-only">(current)</span></a>**

**</li>**

**<li className="nav-item">**

**<a className="nav-link" href="/home">Create New Exam</a>**

**</li>**

**</ul>**

**<button className="btn btn-danger my-2 my-sm-0" type="submit" onClick={onLogoutClick}>Logout</button>**

**</div>**

**</nav>**

**<h1 className="heading App">Welcome {user.name}</h1>**

**<Jumbotron className="py-5" >**

**<h1>General Knowledge Test 1</h1>**

**<p>**

**This quiz is made to test out your general understanding of things around you and the world that we live in...**

**</p>**

**<p>**

**{user.test1Attempted ? (<Button variant="success" onClick={ToogleShowTest1Score}>Check Score</Button>) : (<Button variant="primary" onClick={onTakeTest1}>Take test</Button>)}**

**{showTest1Score ? (<h3 style={{margin:"20px"}}>{user.test1Score} / 5</h3>) : (<p></p>)}**

**</p>**

**</Jumbotron>**

**<Jumbotron className="py-5">**

**<h1>DSF CE 4 on Unit 5 : Binary Search Trees</h1>**

**<p>**

**Continous Evaluation for Binary Search Trees**

**</p>**

**<p>**

**{user.test2Attempted ? (<Button variant="success" onClick={ToogleShowTest2Score}>Check Score</Button>) : (<Button variant="primary" onClick={onTakeTest2}>Take test</Button>)}**

**{showTest2Score ? (<h3 style={{margin:"20px"}}>{user.test2Score} / 8</h3>) : (<p></p>)}**

**</p>**

**</Jumbotron>**

**</div>**

**</div>**

**)**

**}**

**export default Home**

**-------------------------------------------------------------------------------------------------------------------------------------**

**Login.js**

**import React from 'react'**

**import axios from 'axios'**

**import {useState, useContext} from 'react'**

**import {useHistory} from 'react-router-dom'**

**import './../../bootstrap.min.css'**

**import '../../App.css'**

**import BackgroundImage from './../images/bg2.jpg'**

**const Login = () => {**

**let history = useHistory();**

**const [grNo, setGrNo] = useState("");**

**const [password, setPassword] = useState("")**

**const onSubmitClick = async (e) =>**

**{**

**e.preventDefault();**

**const loginCredentials =**

**{**

**"grNo" : grNo,**

**"password" : password**

**}**

**try**

**{**

**const res = await axios.post("http://localhost:7000/login", loginCredentials);**

**alert(res.data);**

**if(res.data === "Login successfull")**

**{**

**history.push("/home");**

**}**

**else**

**{**

**}**

**}**

**catch(err)**

**{**

**console.log(err);**

**}**

**}**

**const onCreateNewAccountClick = (e) =>**

**{**

**e.preventDefault();**

**history.push("/signup");**

**}**

**return (**

**<div style={{backgroundImage:`url(${BackgroundImage})`, height:"1080px"}}>**

**<div className="container" style={{maxWidth : "500px", background:"white", color:"black"}} >**

**<label className="Heading">Login</label>**

**<form>**

**<div className="form-group">**

**<label for="grInput">Gr No.</label>**

**<input type="number" className="form-control" id="grInput" placeholder="Enter Gr. No." onChange={(e) => setGrNo(e.target.value)}></input>**

**</div>**

**<div className="form-group">**

**<label for="passwordInput">Password</label>**

**<input type="password" className="form-control" id="passwordInput" placeholder="Password" onChange={(e) => setPassword(e.target.value)}></input>**

**</div>**

**<button type="submit" className="btn btn-primary" onClick={onSubmitClick}>Submit</button>**

**</form>**

**<br></br>**

**<button className="btn btn-success" onClick={onCreateNewAccountClick}>Create New Account</button>**

**</div>**

**</div>**

**)**

**}**

**export default Login**

**-------------------------------------------------------------------------------------------------------------------------------------**

**SignUp.js**

**import React from 'react'**

**import {useState} from 'react'**

**import {useHistory} from 'react-router-dom'**

**import './../../bootstrap.min.css'**

**import'./../../App.css'**

**import axios from 'axios'**

**import BackgroundImage from './../images/bg3.jpg'**

**const SignUp = () => {**

**let history = useHistory();**

**const [name, setName] = useState("");**

**const [grNo, setGrNo] = useState("");**

**const [email, setEmail] = useState("");**

**const [password, setPassword] = useState("")**

**const onSubmitClick = async (e) =>**

**{**

**const user = {**

**"name" : name,**

**"grNo" : grNo,**

**"email" : email,**

**"password" : password,**

**}**

**try**

**{**

**const res = await axios.post("http://localhost:7000/signup", user);**

**alert(res.data);**

**}**

**catch(err)**

**{**

**console.log(err);**

**}**

**}**

**const onLoginClick = (e) =>**

**{**

**e.preventDefault();**

**history.push("/login");**

**}**

**return (**

**<div style={{backgroundImage:`url(${BackgroundImage})`, height:"1080px"}}>**

**<div className="container" style={{maxWidth : "500px", background:"white", color:"black"}}>**

**<label className="Heading">Sign-Up</label>**

**<form>**

**<div className="form-group">**

**<label for="nameInput">Name</label>**

**<input type="text" className="form-control" id="nameInput" placeholder="Enter Name" onChange={(e) => setName(e.target.value)}></input>**

**</div>**

**<div className="form-group">**

**<label for="grInput">Gr No.</label>**

**<input type="number" className="form-control" id="grInput" placeholder="Enter Gr. No." onChange={(e) => setGrNo(e.target.value)}></input>**

**</div>**

**<div className="form-group">**

**<label for="emailInput">Email address</label>**

**<input type="email" className="form-control" id="emailInput" aria-describedby="emailHelp" placeholder="Enter email" onChange={(e) => setEmail(e.target.value)}></input>**

**<small id="emailHelp" className="form-text text-muted">We'll never share your email with anyone else.</small>**

**</div>**

**<div className="form-group">**

**<label for="passwordInput">Password</label>**

**<input type="password" className="form-control" id="passwordInput" placeholder="Password" onChange={(e) => setPassword(e.target.value)}></input>**

**</div>**

**<button type="submit" className="btn btn-primary" onClick={onSubmitClick}>Submit</button>**

**</form>**

**<br></br>**

**<button className="btn btn-success" onClick={onLoginClick}>Login for Existing User</button>**

**</div>**

**</div>**

**)**

**}**

**export default SignUp**

**------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

**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>Online Exam Project</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>**

**---------------------------------------------------------------------------------------------------------------------------------------------**

**App.css**

**.App {**

**text-align: center;**

**min-width: 100%;**

**min-height: 100%;**

**background-position: center;**

**background-size: cover;**

**}**

**.Heading**

**{**

**font-size: 40px;**

**font-weight: bold;**

**}**

**.backgroundImage**

**{**

**background-image: url('https://www.sss-solutions.org/wp-content/uploads/2018/01/1116146294-login-page-background-image-112.jpg');**

**}**

**.PageNotFound**

**{**

**text-align: center;**

**font-size: 100px;**

**font-weight: bold;**

**}**

**.McqContainer**

**{**

**background-color: #555;**

**color: #ddd;**

**border-radius: 10px;**

**padding: 20px;**

**font-family: 'Montserrat', sans-serif;**

**max-width: 700px;**

**margin: 40px;**

**}**

**.McqOptionsGroup**

**{**

**font-size: large;**

**}**

**.GoToLogin**

**{**

**margin: 0;**

**position: absolute;**

**top: 50%;**

**left: 50%;**

**-ms-transform: translate(-50%, -50%);**

**transform: translate(-50%, -50%);**

**}**

**.App-logo {**

**height: 40vmin;**

**pointer-events: none;**

**}**

**@media (prefers-reduced-motion: no-preference) {**

**.App-logo {**

**animation: App-logo-spin infinite 20s linear;**

**}**

**}**

**.App-header {**

**background-color: #282c34;**

**min-height: 100vh;**

**display: flex;**

**flex-direction: column;**

**align-items: center;**

**justify-content: center;**

**font-size: calc(10px + 2vmin);**

**color: white;**

**}**

**.App-link {**

**color: #61dafb;**

**}**

**@keyframes App-logo-spin {**

**from {**

**transform: rotate(0deg);**

**}**

**to {**

**transform: rotate(360deg);**

**}**

**}**

**\* {**

**margin: 0;**

**padding: 0;**

**box-sizing: border-box**

**}**

**body {**

**background-color: #333**

**}**

**.container {**

**background-color: #555;**

**color: #ddd;**

**border-radius: 10px;**

**padding: 20px;**

**font-family: 'Montserrat', sans-serif;**

**max-width: 700px**

**}**

**.container>p {**

**font-size: 32px**

**}**

**.question {**

**width: 75%**

**}**

**.options {**

**position: relative;**

**padding-left: 40px**

**}**

**#options label {**

**display: block;**

**margin-bottom: 15px;**

**font-size: 14px;**

**cursor: pointer**

**}**

**.options input {**

**opacity: 0**

**}**

**.checkmark {**

**position: absolute;**

**top: -1px;**

**left: 0;**

**height: 25px;**

**width: 25px;**

**background-color: #555;**

**border: 1px solid #ddd;**

**border-radius: 50%**

**}**

**.options input:checked~.checkmark:after {**

**display: block**

**}**

**.options .checkmark:after {**

**content: "";**

**width: 10px;**

**height: 10px;**

**display: block;**

**background: white;**

**position: absolute;**

**top: 50%;**

**left: 50%;**

**border-radius: 50%;**

**transform: translate(-50%, -50%) scale(0);**

**transition: 300ms ease-in-out 0s**

**}**

**.options input[type="radio"]:checked~.checkmark {**

**background: #21bf73;**

**transition: 300ms ease-in-out 0s**

**}**

**.options input[type="radio"]:checked~.checkmark:after {**

**transform: translate(-50%, -50%) scale(1)**

**}**

**.btn-primary {**

**background-color: #555;**

**color: #ddd;**

**border: 1px solid #ddd**

**}**

**.btn-primary:hover {**

**background-color: #21bf73;**

**border: 1px solid #21bf73**

**}**

**.btn-success {**

**padding: 5px 25px;**

**background-color: #21bf73**

**}**

**@media(max-width:576px) {**

**.question {**

**width: 100%;**

**word-spacing: 2px**

**}**

**}**

**--------------------------------------------------------------------------------------------------------------------------------------------------------**

**server.js**

**const express = require('express');**

**const connection = require('./models/db.connect')**

**const authenticatonRouter = require('./routes/authentication');**

**const testRouter = require("./routes/Test")**

**var bodyParser = require('body-parser')**

**var cors = require('cors');**

**const path = require('path');**

**const server = express();**

**server.use(cors())**

**server.use(bodyParser.urlencoded({ extended: false }))**

**// parse application/json**

**server.use(bodyParser.json())**

**server.use('/', authenticatonRouter);**

**server.use('/', testRouter);**

**const PORT = process.env.PORT || 7000;**

**server.get("/name", async (req, res) =>**

**{**

**res.send("Thanks for visiting me!");**

**console.log("recieved a request at /name");**

**})**

**server.post("/checkAnswer", async (req, res) =>**

**{**

**console.log("request recieved for check answer");**

**const selectedOption = req.body.selectedOption;**

**if(selectedOption === "Option-3")**

**{**

**responseJson = {**

**"isCorrect" : "true"**

**}**

**res.json(responseJson);**

**}**

**else**

**{**

**responseJson = {**

**"isCorrect" : "false"**

**}**

**res.json(responseJson);**

**}**

**})**

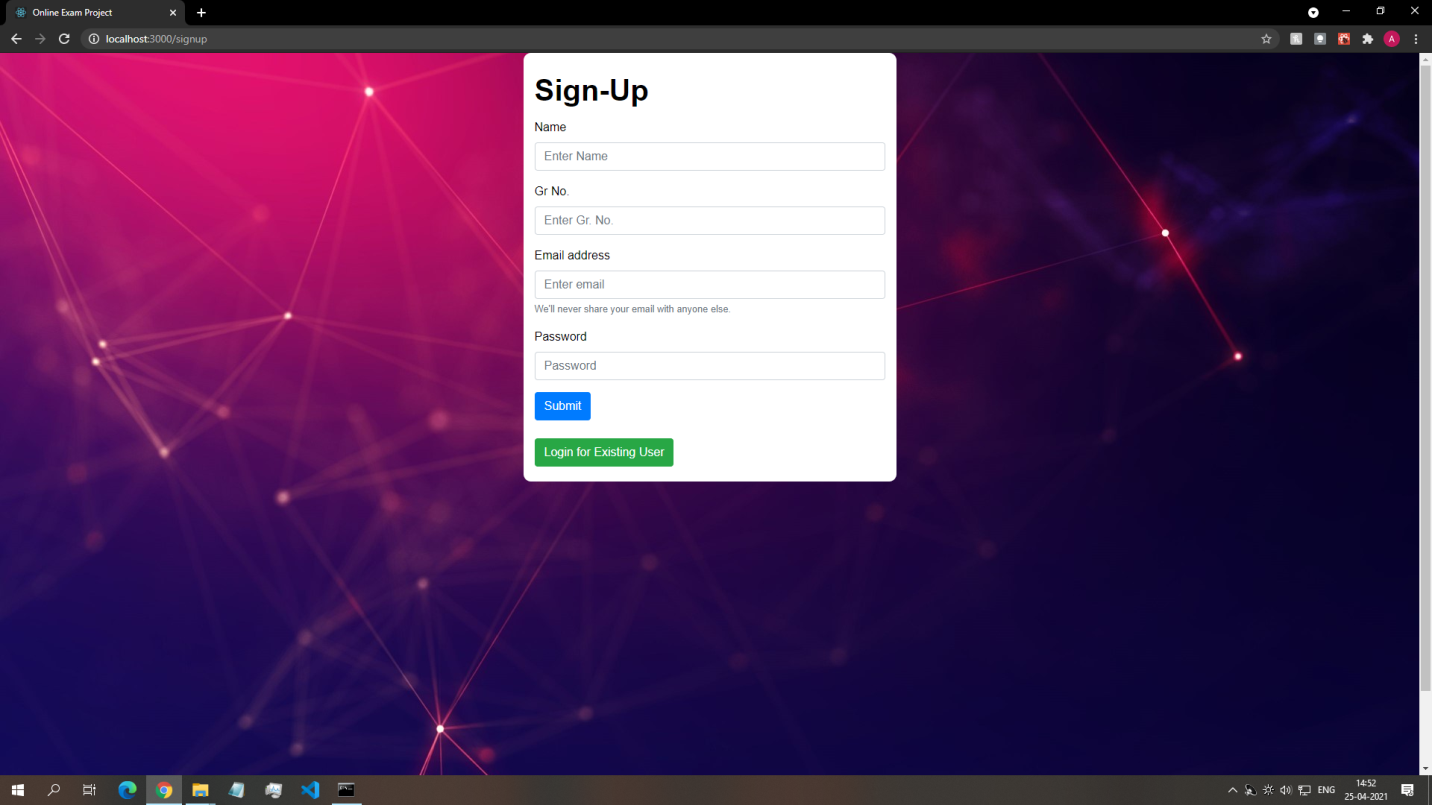
**server.listen(PORT, () =>**

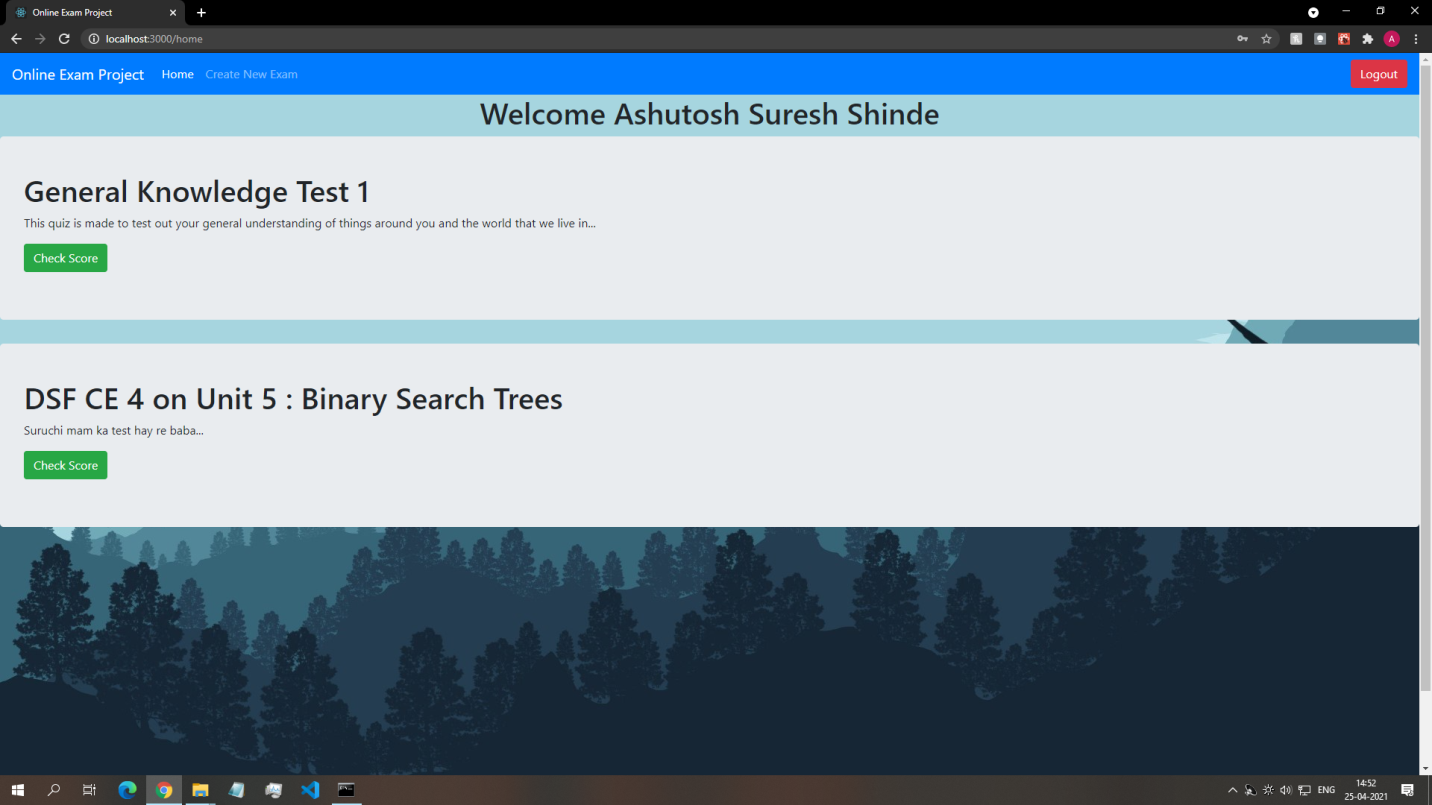
**{**

**console.log("Server listening at port " + PORT);**

**})**

**Output Screenshots**

****

****

**Part 2:- Online AI Proctor**

**AI PROCTOR**

To ***avoid online malpractices in  online*** exam. There are ***so many screens so it is  difficult*** for the proctor to ***keep watch on every student simonteanously.***

**How Ai proctor works**

Software  captures  the  image of user  in realtime . Then it is stored in AWS S3 further analynsied  by Amazon Rekognition service , which describes  all the details of user.

If there is anysort of  electronics detected then software will provide  the details to the proctor.

**Amazon Rekognition**

 Amazon Rekognition makes it easy to add image and video analysis to your applications using proven, highly scalable, deep learning technology that requires no machine learning expertise to use

**Amazon S3**

Amazon S3 or Amazon Simple Storage Service is a service offered by Amazon Web Services that provides object storage through a web service interface.

Libraries used

Cv2

This library helps in to capture photo of user

**Boto3**

**Boto3** is the name of the **Python** SDK for AWS. It allows you to directly create, update, and delete AWS resources from your **Python** scripts.

**CODE:**

#pip install  opencv-python

import cv2

#for timepass

# web cam photo click

cap  = cv2.VideoCapture(0)

myphoto = "sairaj.jpg"

ret  ,  photo = cap.read()

cv2.imwrite( myphoto , photo)

cap.release()

# upload photo into cloud storage :  S3

region = "us-east-1"

bucket = "myawsaibucket1637"

# pip install boto3

# pip  install aws

import boto3

upimage = "file.jpg"

s3 = boto3.resource('s3')

s3.Bucket(bucket).upload\_file(myphoto , upimage)

rek = boto3.client('rekognition' , region )

response = rek.detect\_labels(

 Image={

          'S3Object': {

              'Bucket': bucket,

              'Name': upimage,

          }

      },

      MaxLabels=10,

      MinConfidence=60

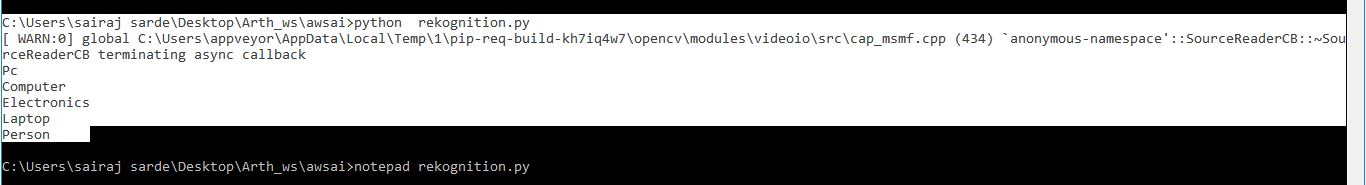
)

for i in range(5):

    print ( response['Labels'][i]['Name'] )

**Ouput of code(SS):-**

This output of code  to below  image.



Here you can see , **AWS Rekognition** gives all the details of the image.

