INT 222

Name:ravindra Yadav

Roll.no: 11708826

Q.1 Create express based application to store the information of student (id, name, course, gender). Also create one field for uploading the resume of student and that resume should be submit in new file also display resume on the server when user submit the form. (Apply some css properties to form and relevant background-image)

ANS:

Code:

Index.js

var express=require('express');

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

const path=require('path')

var morgan=require('morgan')

var multer=require('multer')

const app=express();

const Port=process.env.PORT||3001

app.use(bodyParser.json());

app.use(bodyParser.urlencoded({extended:true}))

app.use(express.static('public'))

// app.use(morgan());

app.get('/',(req,res)=>{

    res.sendFile(\_\_dirname+'/index.html')

    // next()

})

const storage=multer.diskStorage({

    destination:'./public/uploads',

    filename: function(req,file,cb)

    {

        //cb is the call back function

        cb(null,file.fieldname+"-"+Date.now()+path.extname(file.originalname))

    }

})

const upload=multer({

    storage:storage//store at the above defined storage

}).single('resume')

app.post('/upload',(req,res)=>{

    upload(req,res,(err)=>{

        if(err)

        {

            // let path=\_\_dirname+'/index.html'

            // res.sendFile(path);

            console.log("error is being logged",err)

        }

        else

        {

            // console.log(req.file);

            if(req.file==undefined)

            {

                console.log("error file type undefined")

                // let path=\_\_dirname+'/index.html'

                // res.sendFile(path)

            }

            else

            {

                //parsing the value of the name id course and gender

                console.log(req.body.name);

                console.log(req.body.id);

                console.log(req.body.course);

                console.log(req.body.gender);

                // alert("file uploaded succesfully")

                console.log("req-file",req.file)

                res.sendFile(\_\_dirname+"/"+req.file.path);

            }

        }

    })

})

app.listen(Port,()=>{

    console.log(`the server is running at localhost:${Port}`)

})

Index.html

<!doctype html>

<html lang="en">

  <head>

    <!-- Required meta tags -->

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->

    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/css/bootstrap.min.css" integrity="sha384-TX8t27EcRE3e/ihU7zmQxVncDAy5uIKz4rEkgIXeMed4M0jlfIDPvg6uqKI2xXr2" crossorigin="anonymous">

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

    <title>Hello, world!</title>

  </head>

  <body>

    <div class="form-div container">

      <h1 style="text-align:center;">

        FORM DETAILS

      </h1>

      <form action="/upload" method="post" enctype="multipart/form-data">

        <div class="form-group">

            <label for="id">ID</label>

            <input type="number" class="form-control" name="id" id="id">

        </div>

        <div class="form-group">

            <label for="name">Name</label>

            <input type="text" class="form-control"  name="name" id="name">

        </div>

        <div class="form-group">

            <label for="course">Course</label>

            <input type="text" class="form-control"  name="course" id="course">

        </div>

        <div class="form-group">

            <label for="gender">Gender</label>

            <select name="gender" class="form-control"  id="gender">

                <option value="Male">Male</option>

                <option value="Female">Female</option>

                <option value="Any Other">Any other</option>

            </select>

        </div>

        <div class="form-group">

          <label for="resume">Upload Resume</label>

          <input type="file" class="form-control-file" name="resume" id="resume">

        </div>

        <button class="btn btn-primary" type="submit">Submit</button>

    </form>

    </div>

    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: jQuery and Bootstrap Bundle (includes Popper) -->

    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.bundle.min.js" integrity="sha384-ho+j7jyWK8fNQe+A12Hb8AhRq26LrZ/JpcUGGOn+Y7RsweNrtN/tE3MoK7ZeZDyx" crossorigin="anonymous"></script>

    <!-- Option 2: jQuery, Popper.js, and Bootstrap JS

    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js" integrity="sha384-9/reFTGAW83EW2RDu2S0VKaIzap3H66lZH81PoYlFhbGU+6BZp6G7niu735Sk7lN" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@4.5.3/dist/js/bootstrap.min.js" integrity="sha384-w1Q4orYjBQndcko6MimVbzY0tgp4pWB4lZ7lr30WKz0vr/aWKhXdBNmNb5D92v7s" crossorigin="anonymous"></script>

    -->

  </body>

</html>

Style.css

body

{

    background-color: beige;

    padding-top: 10%;

}

/\* for hiding the number spin \*/

input[type='number'] {

    -moz-appearance:textfield;

}

input::-webkit-outer-spin-button,

input::-webkit-inner-spin-button {

    -webkit-appearance: none;

}

.btn

{

    margin-left: 50%;

}

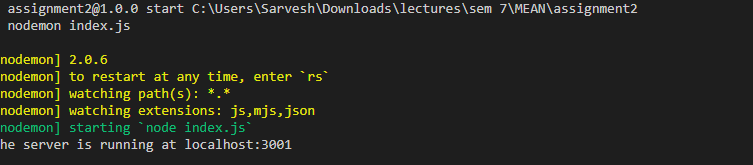
label

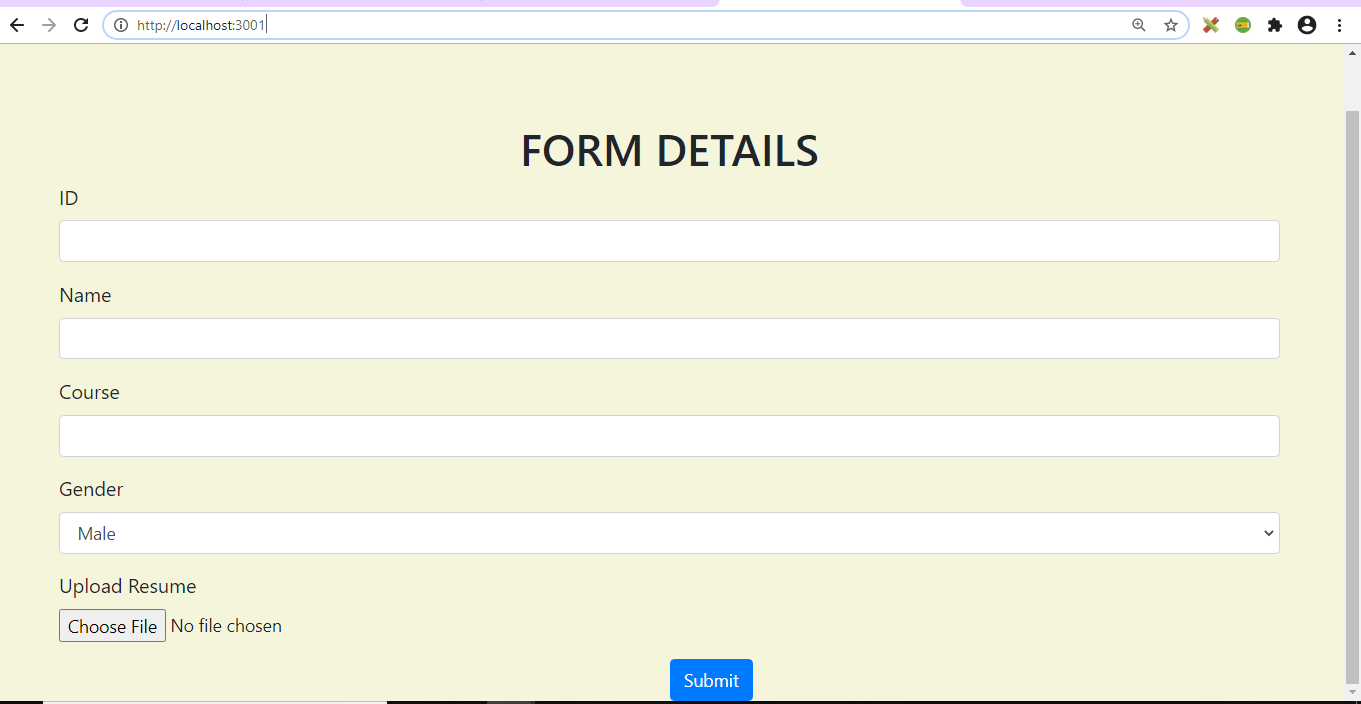
{

    font-size: large;

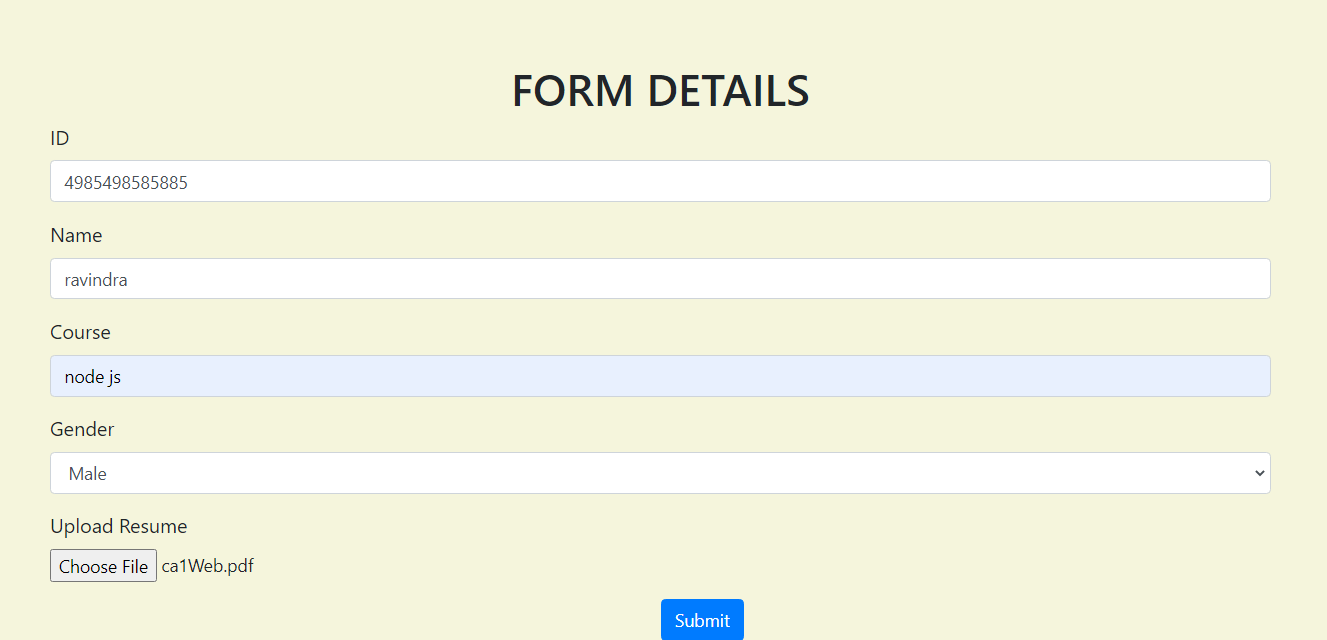
}

Output:

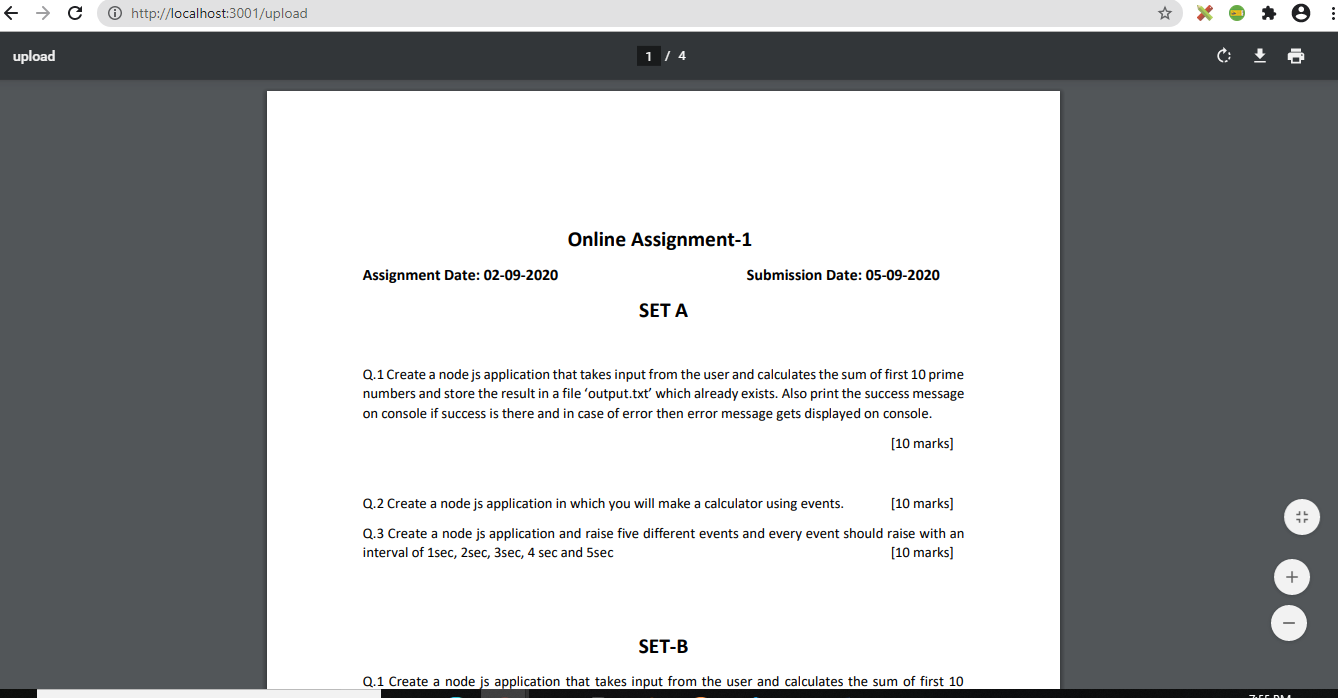


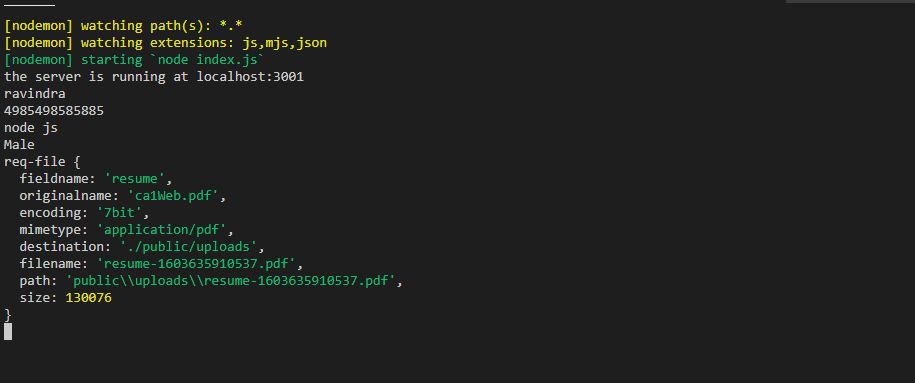


Filling the inputs



Output on /upload





Q: 2

Create a clone of zoom app using socket.io. In this application add some features like at what time user has joined the chat and once user joined the chat send the message to all the users that this user has joined the chat and also display message to all the users when any user left the chat along with their time.

ANS:

Index.js

const express = require('express')

const path=require('path')

const app = express()

const server = require('http').createServer(app)

const io =require('socket.io')(server)

const port = process.env.PORT || 3001

app.use(express.static('./public/'))

// just to test the server

app.get('/', (req, res) => {

    res.sendFile(\_\_dirname+'/viewpage.html')

})

io.on('connection',socket=>{

    // io.emit

    // {

    // }

    socket.on('chat', message => {

        // console.log(message.name,":", message.mess)

        io.emit('chat', message)

      })

})

server.listen(port, () => {

  console.log(`Server running on port: ${port}`)

})

Viewpage.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>ZOOM</title>

    <link rel="stylesheet" href="/style.css">

</head>

<body>

    <div class="container">

        <h1>ZOOM</h1>

        <div id="chatSection">

            <div class="chat-window"></div>

            <form class="chat-form">

              <label class="chat-label">

                Enter a message:

                <input type="text" class="chat-input" />

              </label>

              <input type="submit" class="chat-submit" value="enter" />

            </form>

          </div>

    </div>

    <!-- <style>

      body

      {

        margin-left: 30%;

      }

      #container {

        width: 700px;

        margin: 0 auto;

      }

      .chat-input {

        width: 300px;

        height: 35px;

        border: solid 1px #444;

      }

      .chat-submit {

        width: 50px;

        height: 35px;

        border: solid 1px #444;

      }

      .chat-window {

        height: 300px;

        width: 400px;

      }

      #chatSection {

        float: left;

        border: 1px grey solid;

        border-radius: 10px;

        padding: 10px;

      }

    </style> -->

    <script src="socket.io/socket.io.js"></script>

    <script src="/script.js"></script>

</body>

</html>

Script.js

 const socket=io()

var name;

const chatWindow = document.querySelector('.chat-window')

const chat = document.querySelector('.chat-form')

const Input = document.querySelector('.chat-input')

do

{

    name=prompt("enter your name")

}

while(!name);

//for rendering the message on the page

const renderMessage = message => {

    const div = document.createElement('div')

    div.classList.add('render-message')

    // let markup=`

    // <h4>${msg.user}</h4>

    // <p>${msg.message}</p>`

    div.innerHTML = `<h4>from:${message.name}</h4>

    <p>${message.mess}</>`

    chatWindow.appendChild(div)

  }

chat.addEventListener('submit', event => {

  event.preventDefault()

  tempObj={

      name:this.name,

      mess:Input.value

  }

  socket.emit('chat',tempObj)

  Input.value = ''

})

// socket.on('chat', message => {

//     console.log(message.name,":", message.mess)

//   })

socket.on('chat', message => {

    // make sure to modify this

    renderMessage(message)

  })

Public/style.css

#container {

    width: 700px;

    margin: 0 auto;

  }

  .chat-input {

    width: 300px;

    height: 35px;

    border: solid 1px #444;

  }

  .chat-submit {

    width: 50px;

    height: 35px;

    border: solid 1px #444;

  }

  .chat-window {

    height: 300px;

    width: 400px;

  }

  #chatSection {

    float: left;

    border: 1px grey solid;

    border-radius: 10px;

    padding: 10px;

  }

  /\* .render-message

  {

    background:rgb(189, 199, 189);

  } \*/

  .render-message p

  {

    color: coral;

  }

  .render-message h3

  {

    color: blueviolet;

  }

Output:

