**Roll No : 28**

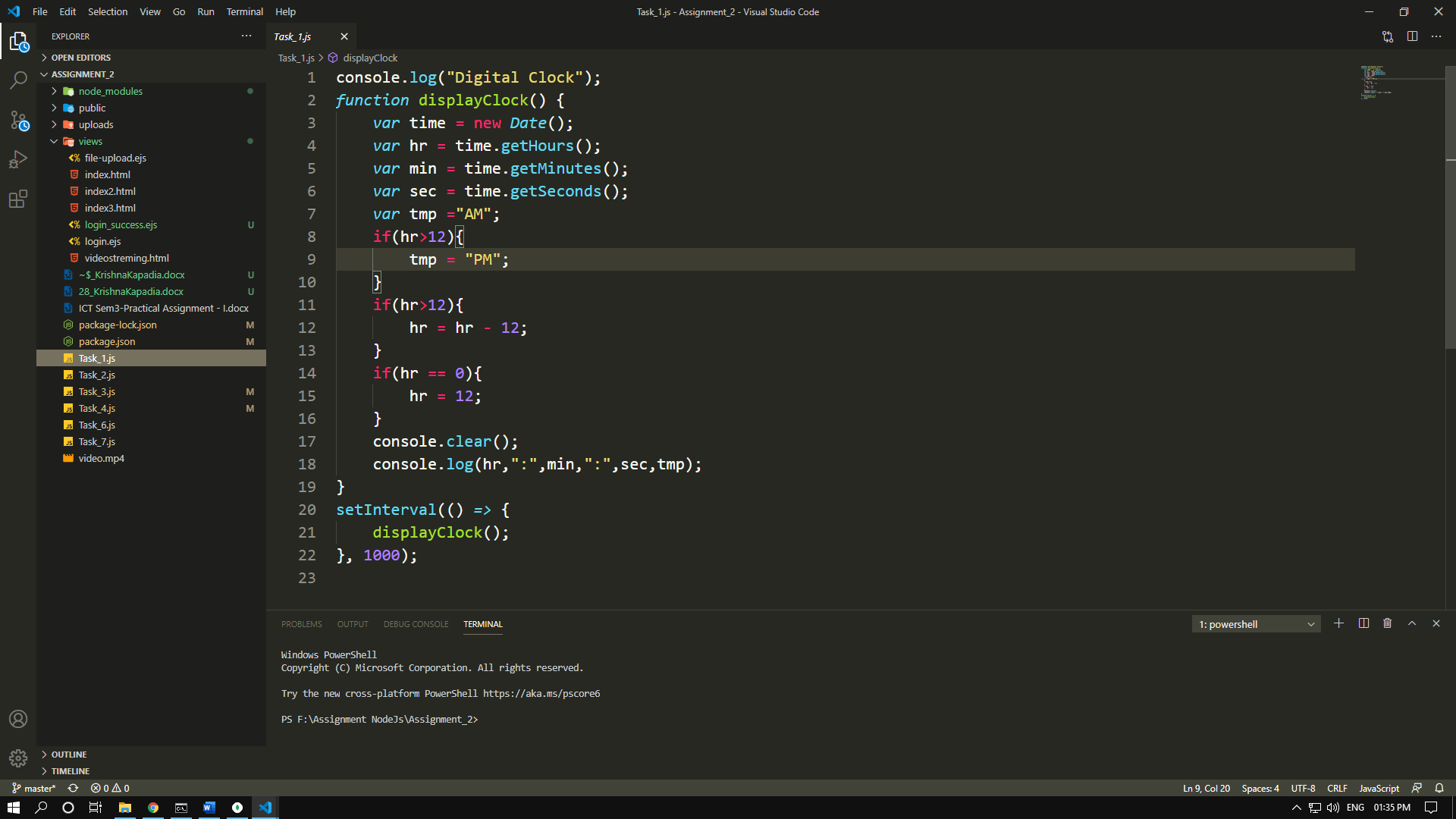
**Name : Krishna Kapadia**

**Subject : NodeJs**

**Assignment – 2**

**21/11/2020**

**// Directory Structure**



1. **Command line based digital clock with date and time.**

**Task\_1.js**

console.log("Digital Clock");

function displayClock() {

var time = new Date();

var hr = time.getHours();

var min = time.getMinutes();

var sec = time.getSeconds();

var tmp ="AM";

if(hr>12){

tmp = "PM";

}

if(hr>12){

hr = hr - 12;

}

if(hr == 0){

hr = 12;

}

console.clear();

console.log(hr,":",min,":",sec,tmp);

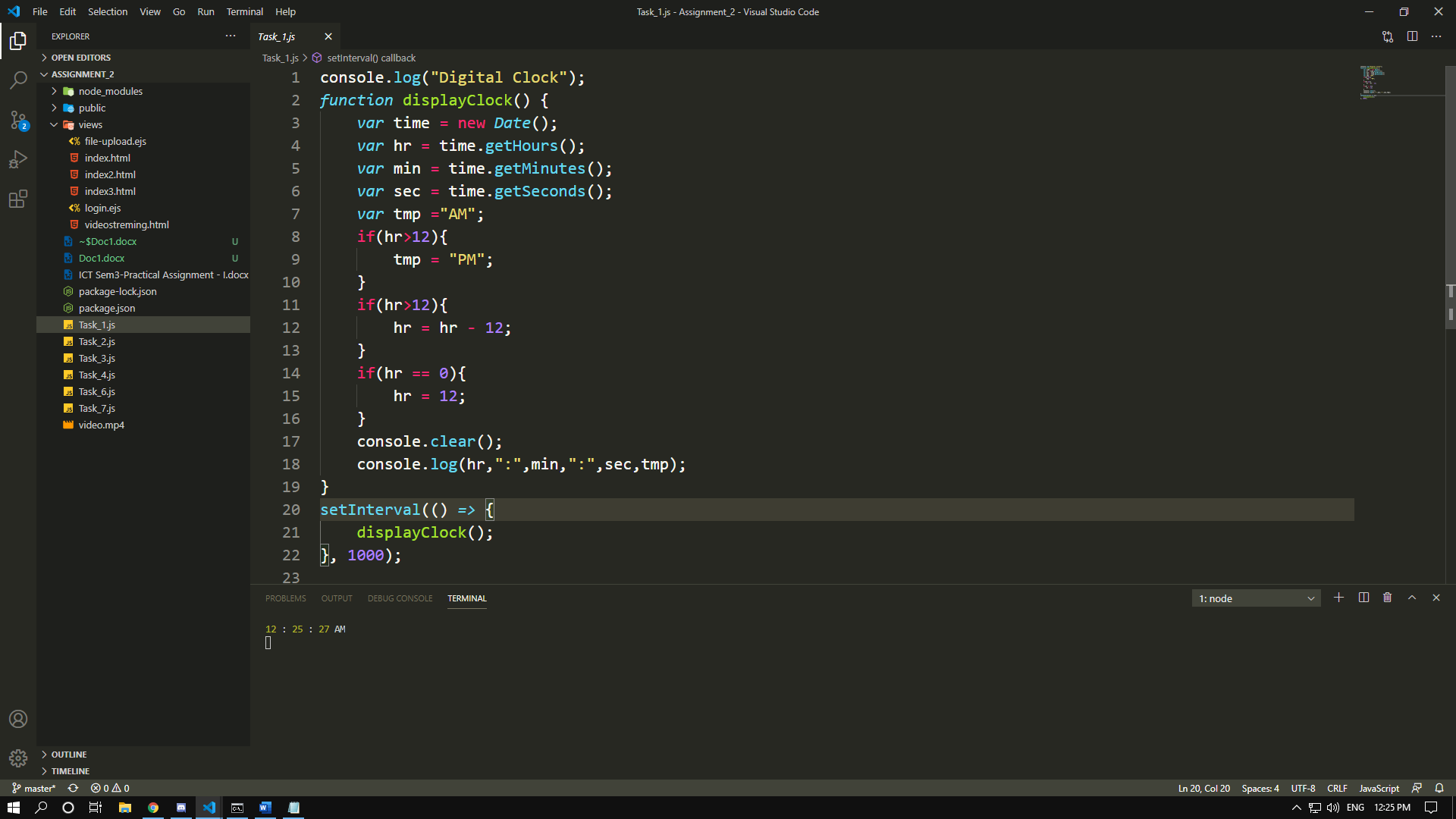
}

setInterval(() => {

displayClock();

}, 1000);

// Output



1. **Express File upload (single, multiple) with validations.**

**Task\_2.js**

// Image Upload With Validation

const express = require('express');

const multer = require('multer');

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

const path = require('path');

var upload = multer({ dest: 'uploads/' })

const app = express();

const port = 8080;

var router = express.Router();

router.use(express.static(\_\_dirname+'/public'));

var Storage = multer.diskStorage({

destination: function (req, file, cb) {

if (file.mimetype !== 'image/jpeg') {

return cb('Invalid file format'); //cb(err)

}

cb(null, './public/uploads');

},

filename:(req,file,cb)=>{

cb(null,file.fieldname+"\_"+Date.now()+path.extname(file.originalname));

}

});

var upload = multer({

storage:Storage

}).single('file');

app.set('view engine','ejs');

app.set('views', \_\_dirname + '/views');

app.use(express.urlencoded({ extended: true }));

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

res.render("file-upload");

});

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

console.log(req.file.filename);

res.render("file-upload");

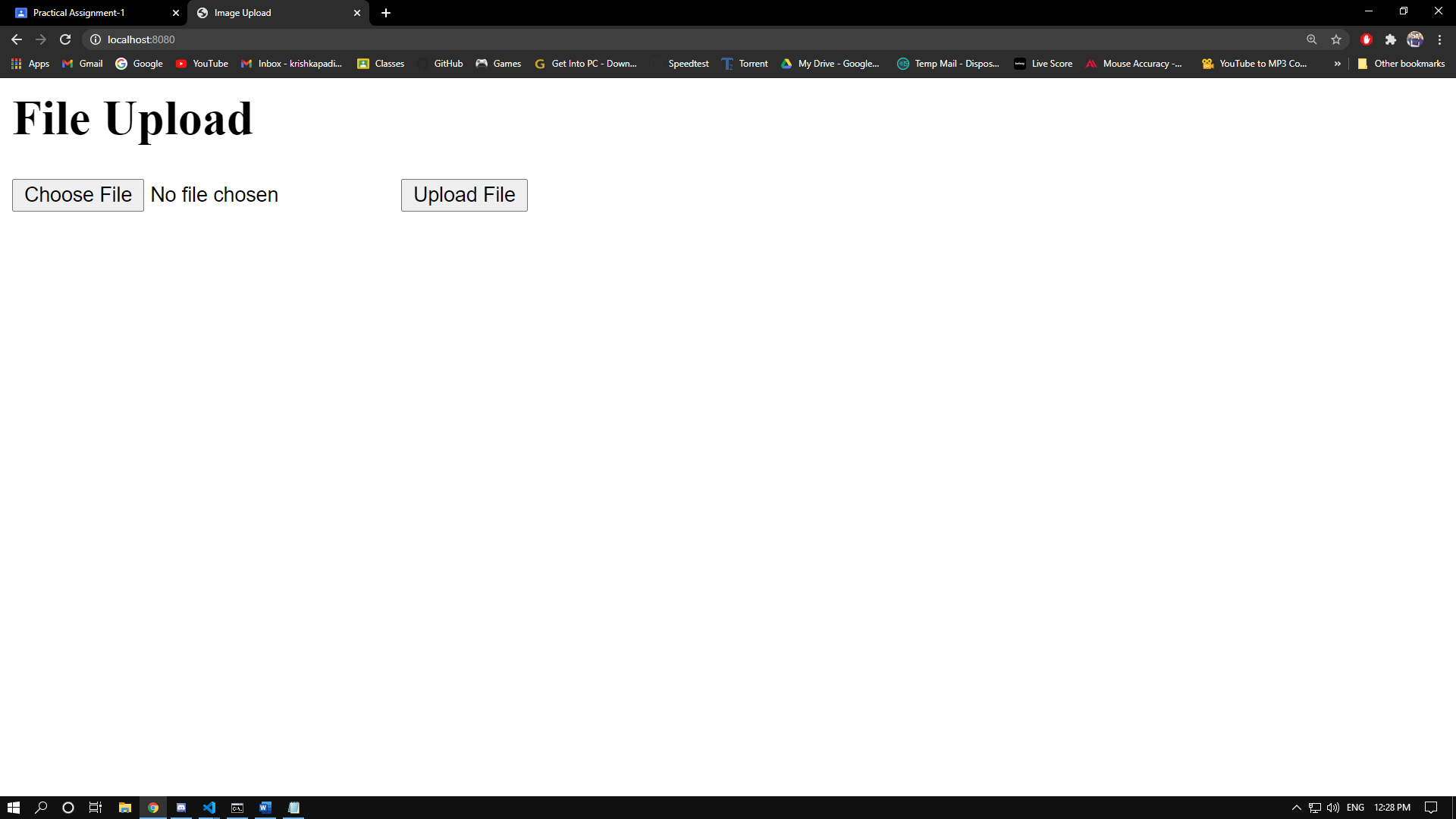
})

app.listen(port,()=>{

console.log("Server is runningon port number:",port);

})

//Output



**3) Express Login application with file session store**

**Task\_3.js**

// Session Login

const express = require('express');

var session = require('express-session');

var FileStore = require('session-file-store')(session);

const app = express();

const port = 8080;

app.set("view engine", "ejs");

app.use(express.urlencoded({ extended: true }));

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

res.render("login", { success: null, color: null });

});

app.use(session({

resave: true,

saveUninitialized: true,

secret: 'keyboard cat'

}));

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

var username = req.body.username;

var password = req.body.password;

if (username == "admin" && password == "admin") {

req.session.uname = username;

res.render("login", { success: "Login Successfull...", color: "Green" });

}

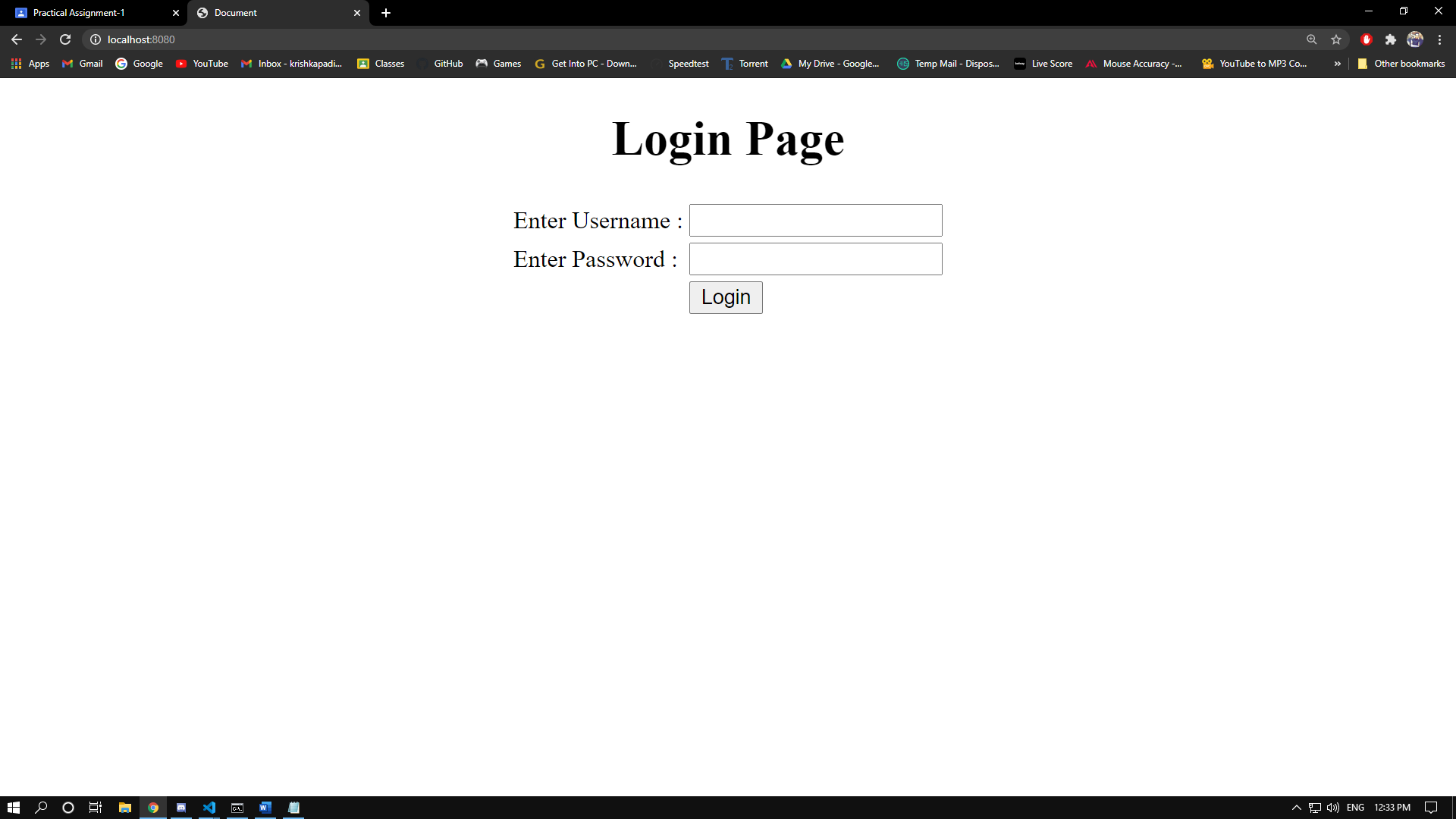
else {

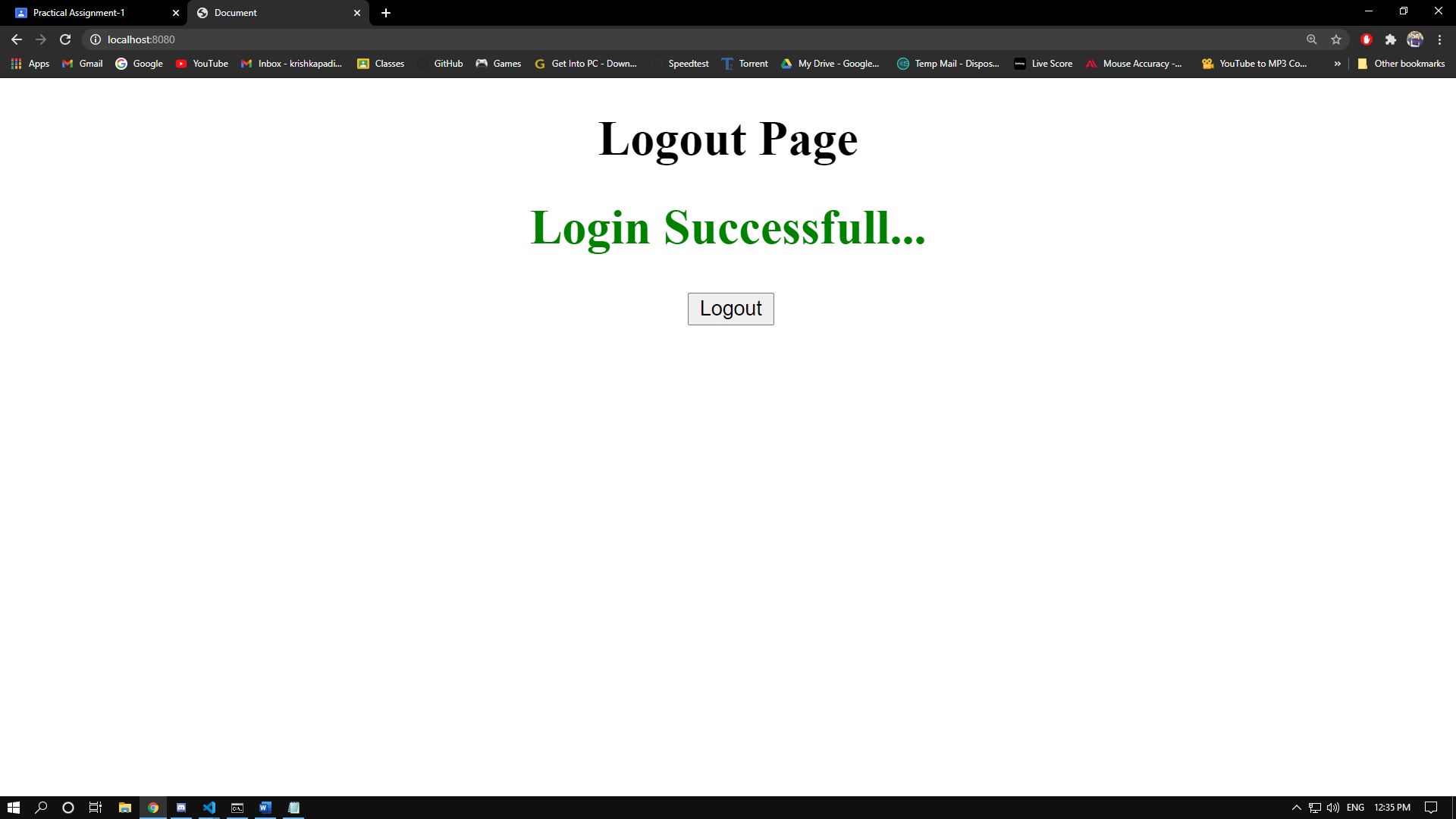
res.render("login", { success: "Invalid Username or Password", color: "Red" });

}

});

app.listen(port);





**4) Develop a Video streaming server using nodejs.**

**Task\_4.js**

const express = require('express');

const app = express();

const fs = require('fs');

const port = 8080;

router.use(express.static(\_\_dirname + '/public'));

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

res.sendFile(\_\_dirname + "/views/videostreming.html");

});

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

const range = req.headers.range;

if (!range) res.status(400).send("Requires Range Header");

const videoPath = "video.mp4";

const videoSize = fs.statSync("video.mp4").size;

const CHUNK\_SIZE = 10 \*\* 6;

const start = Number(range.replace(/\D/g, ""));

const end = Math.min(start + CHUNK\_SIZE, videoSize - 1);

const contentLength = end - start + 1

const headers = {

"Content-Range": `bytes ${start}-${end}/${videoSize}`,

"Accept-Ranges": "bytes",

"Content-Length": contentLength,

"Content-Type": "video/mp4",

};

res.writeHead(206, headers);

const videoStream = fs.createReadStream(videoPath, { start, end });

videoStream.pipe(res);

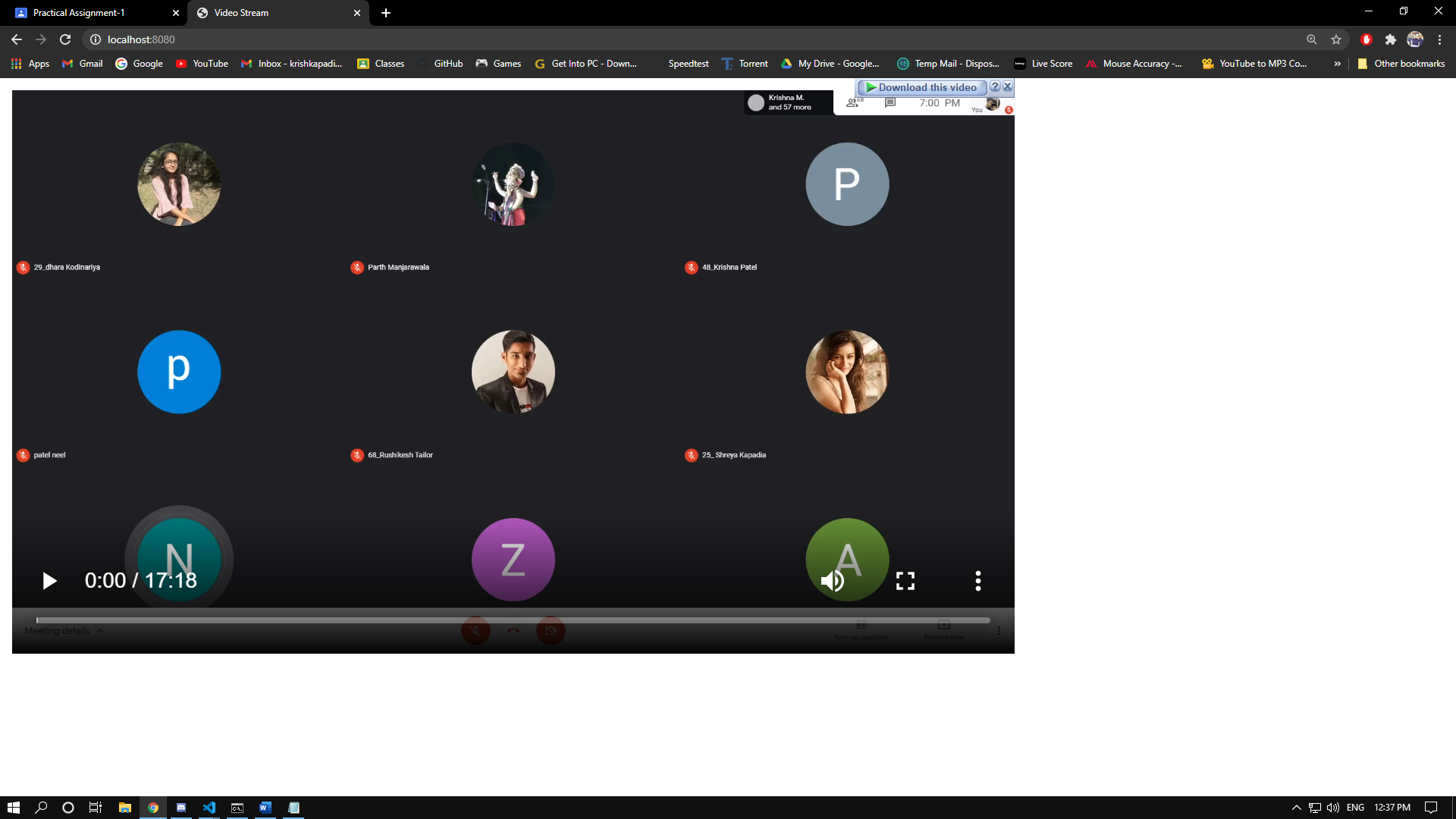
});

app.listen(port, () => {

console.log("Server is running on port number : ", port);

});

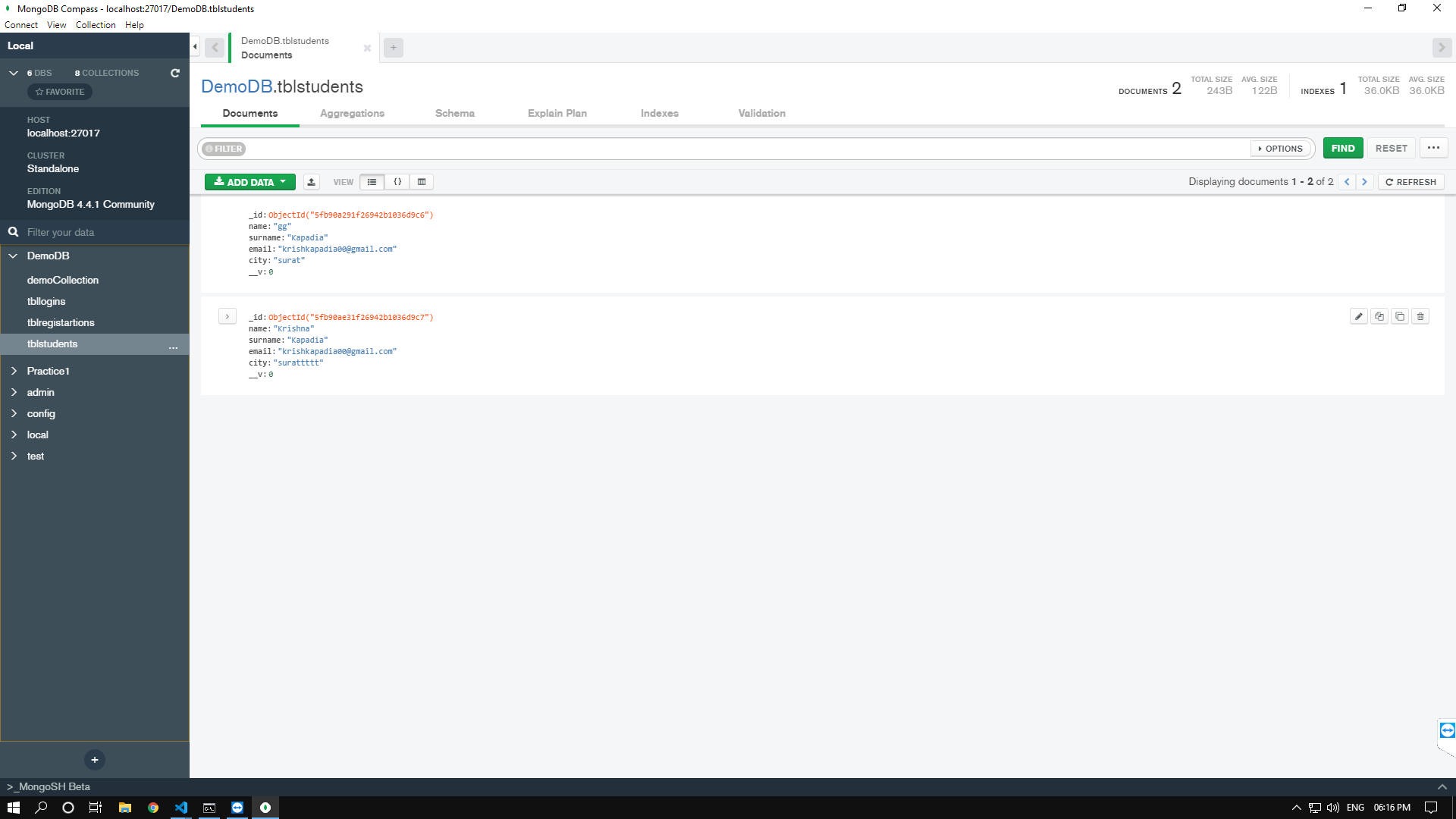
//Output



**5) Live cricket score application with websocket in nodejs.**

// Cant find API for cricket score

**6) Login, CRUD operations for students table with mongodb, express and any one template engine, Logout.**



// Username : admin

// Password : admin

**Task\_6.js**

// 6) Login, CRUD operations for students table with mongodb, express and any one template engine, Logout.

// Username : admin

// Password : admin

const express = require('express');

const mongoose = require('mongoose');

const app = express();

const port = 8080;

// MongoDb Connection

mongoose.connect('mongodb://localhost/DemoDB', { useNewUrlParser: true, useUnifiedTopology: true });

// Connect With Database

const db = mongoose.connection;

db.on('error', console.error.bind(console, 'connection error:'));

db.once('open', function () {

console.log("Database Connection Successful.");

});

// Creating Schema

const tblLoginSchema = new mongoose.Schema({

username: String,

password: String

});

const tblStudentSchema = new mongoose.Schema({

name: String,

surname: String,

email: String,

city: String

})

// Referencing Schema

const tblLogin = mongoose.model('tblLogin', tblLoginSchema);

const tblStudent = mongoose.model('tblStudent',tblStudentSchema);

// Set View Engine

app.set("view engine", "ejs");

// Url Encoder

app.use(express.urlencoded({ extended: true }));

// Get Method

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

res.render("login", { success: null, color: null });

});

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

var msg = null;

var username = req.body.username;

var password = req.body.password;

tblLogin.findOne({ username: username, password: password }, (err, data) => {

if (data == null)

res.render("login", { success: "Invalid Username or Password", color: "Red" });

else

res.render("student", { success: "Successfull Login", color: "Green" });

});

});

app.post("/student", (req, res) => {

var msg = null;

var name = req.body.name;

var surname = req.body.surname;

var email = req.body.email;

var city = req.body.city;

var demoObj = new tblStudent({

name:name,

surname:surname,

email:email,

city:city

});

demoObj.save((err, data) => {

console.log(data);

})

res.redirect("/studentdata");

});

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

tblStudent.find((err, data) => {

res.render("student", { data: data });

});

})

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

tblStudent.find((err, data) => {

res.render("student", { data: data });

});

})

app.post("/delete", (req, res) => {

var id = req.query.id;

var del = tblStudent.findByIdAndDelete(id)

del.exec((err,data)=>{

res.redirect("/studentdata");

})

});

app.post("/updateData", (req, res) => {

var id = req.query.id;

tblStudent.find({ \_id: id }, (err, data1) => {

if (!err) {

tblStudent.find((err, data) => {

if (!err) {

res.render("updateView.ejs", { data: data, params: data1 });

}

})

}

});

});

app.post("/update", (req, res) => {

var id = req.query.id;

tblStudent.findOneAndUpdate({\_id:id}, { name: req.body.name, surname: req.body.surname, city: req.body.city, email: req.body.email }, (err, data1) => {

if (!err) {

res.redirect("/studentdata");

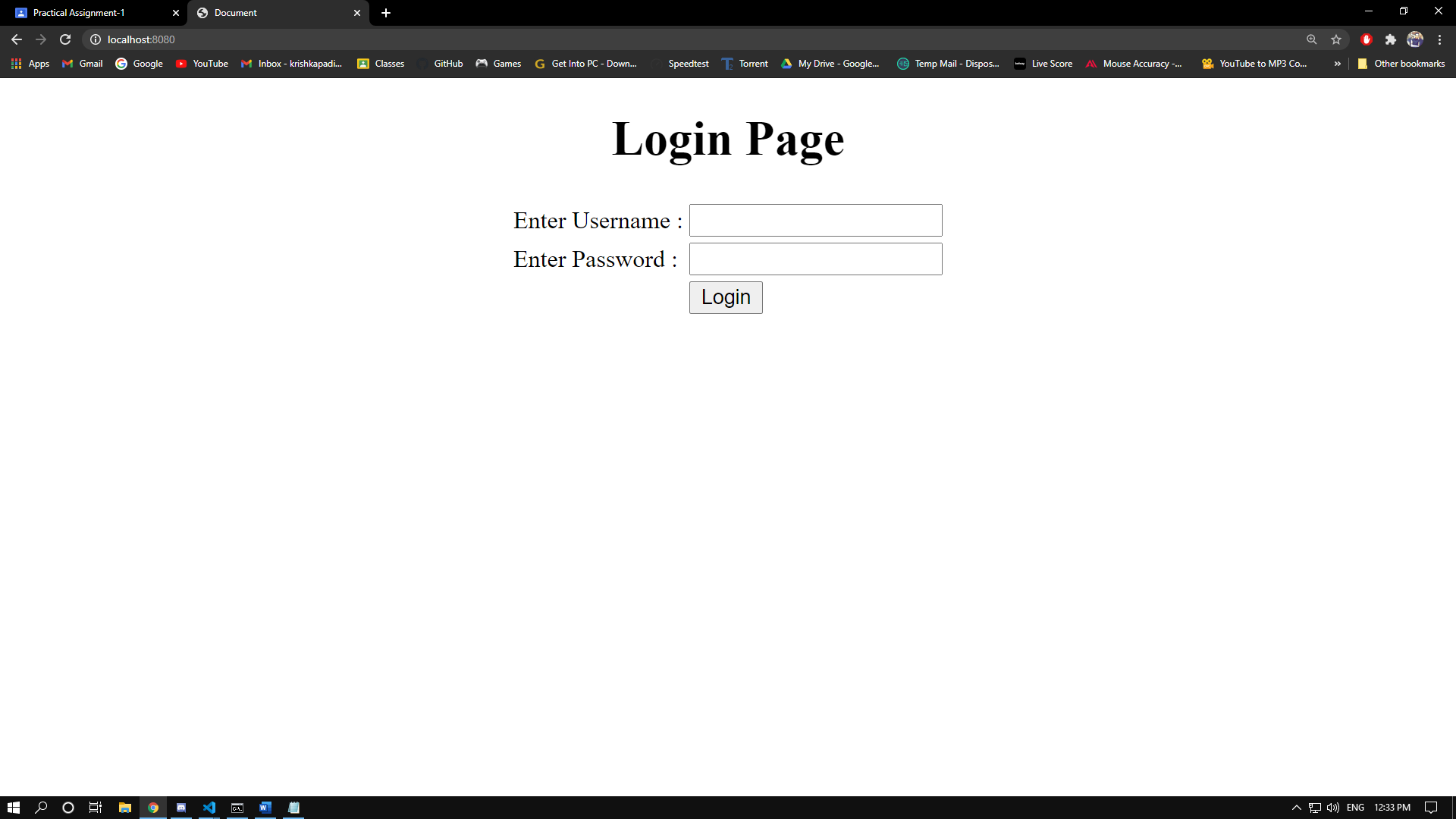
}

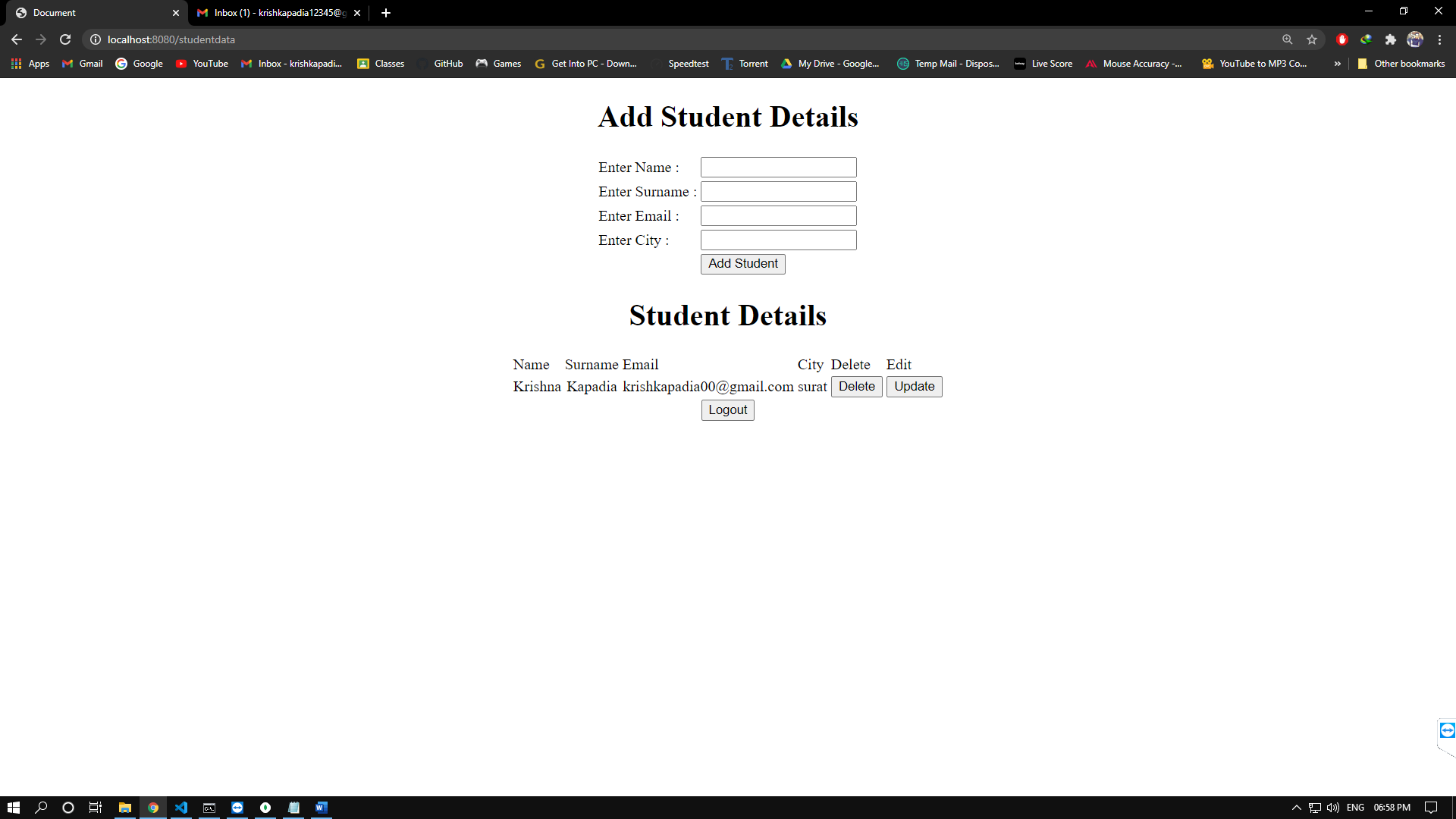
});

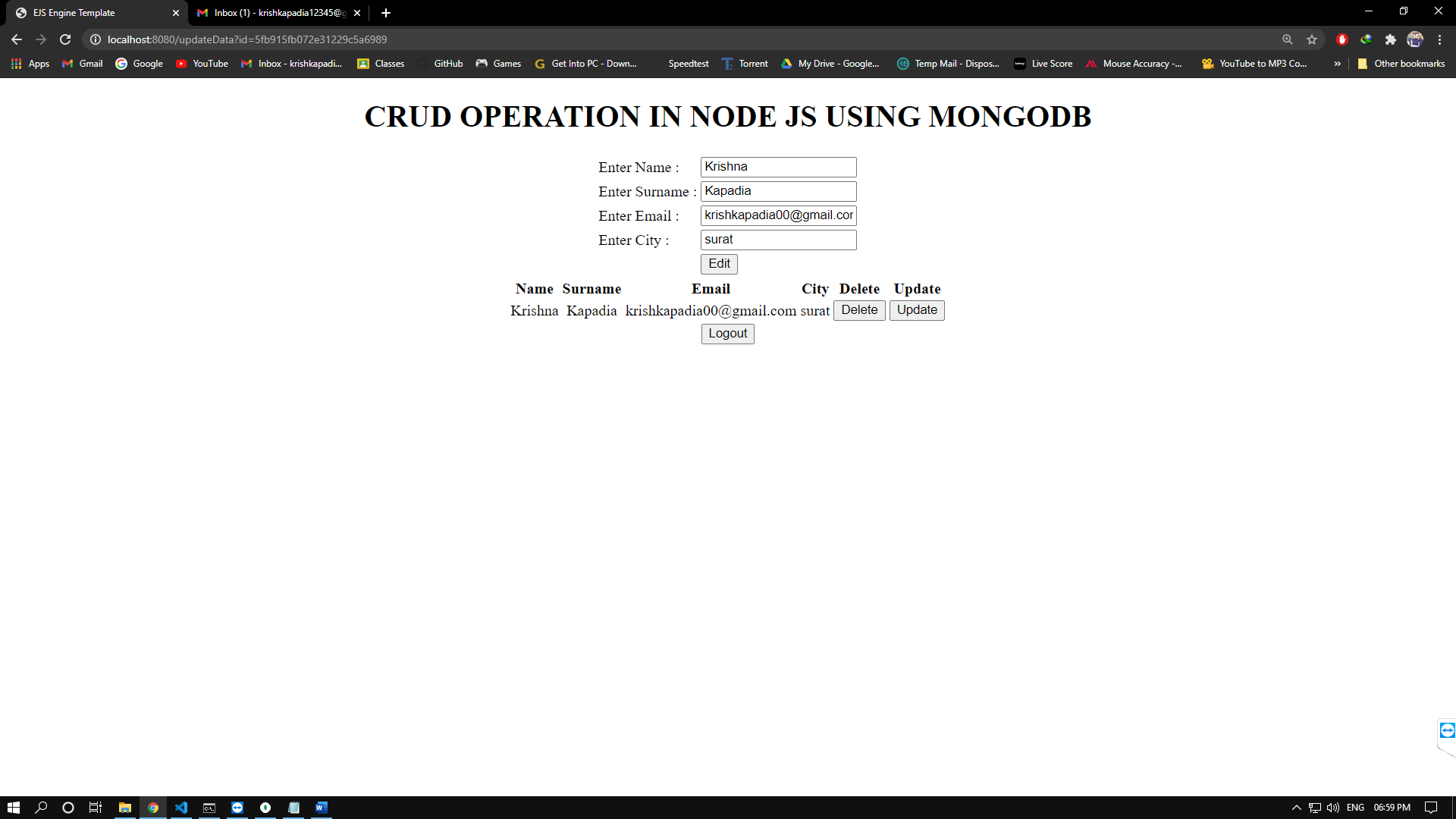
})

app.listen(port, () => {

console.log("Server is running on port number : ", port);

});





**7) Login, CRUD operations for students table with mongodb, express and frontend(html,css,javascript/jquery/angularjs), Logout.**

**Task\_7.js**

// Login, CRUD operations for students table with mongodb, express and frontend(html, css, javascript / jquery / angularjs), Logout.

// Username : admin

// Password : admin

const { urlencoded } = require('body-parser');

const express = require('express');

const mongoose = require('mongoose');

const app = express();

//var router = express.Router();

const port = 8080;

// MongoDb Connection

mongoose.connect('mongodb://localhost/DemoDB', { useNewUrlParser: true, useUnifiedTopology: true });

// Connect With Database

const db = mongoose.connection;

db.on('error', console.error.bind(console, 'connection error:'));

db.once('open', function () {

console.log("Database Connection Successful.");

});

// Creating Schema

const tblLoginSchema = new mongoose.Schema({

username: String,

password: String

});

const tblStudentSchema = new mongoose.Schema({

name: String,

surname: String,

email: String,

city: String

})

// Referencing Schema

const tblLogin = mongoose.model('tblLogin', tblLoginSchema);

const tblStudent = mongoose.model('tblStudent', tblStudentSchema);

app.use(express.urlencoded({ extended: true }));

//router.use(express.static(\_\_dirname + '/public'));

app.set("view engine", "ejs");

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

res.sendFile(\_\_dirname + "/views/index.html",{name : "Krishna"});

});

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

var username = req.body.username;

var password = req.body.password;

tblLogin.findOne({ username: username, password: password }, (err, data) => {

if (data == null)

{

//res.render("login", { success: "Invalid Username or Password", color: "Red" });

res.sendFile(\_\_dirname + "/views/index2.html");

}

else

{

//res.render("login", { success: "Successfull Login", color: "Green" });

res.redirect("/studentdata");

//res.sendFile(\_\_dirname + "/views/index3.html");

}

console.log(data);

})

});

app.post("/student", (req, res) => {

var msg = null;

var name = req.body.name;

var surname = req.body.surname;

var email = req.body.email;

var city = req.body.city;

var demoObj = new tblStudent({

name: name,

surname: surname,

email: email,

city: city

});

demoObj.save((err, data) => {

console.log(data);

})

res.redirect("/studentdata");

});

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

tblStudent.find((err, data) => {

//res.render("student", { data: data });

res.send(JSON.stringify(data));

//res.sendFile(\_\_dirname + "/views/index3.html",{data:"Hello"});

});

})

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

tblStudent.find((err, data) => {

res.sendFile(\_\_dirname + "/views/index3.html", { data: "Hello" });

});

})

app.post("/delete", (req, res) => {

var id = req.query.id;

var del = tblStudent.findByIdAndDelete(id)

del.exec((err, data) => {

res.redirect("/studentdata");

})

});

app.post("/updateData", (req, res) => {

var id = req.query.id;

tblStudent.find({ \_id: id }, (err, data1) => {

if (!err) {

tblStudent.find((err, data) => {

if (!err) {

res.render("updateView.ejs", { data: data, params: data1 });

}

})

}

});

});

app.post("/update", (req, res) => {

var id = req.query.id;

tblStudent.findOneAndUpdate({ \_id: id }, { name: req.body.name, surname: req.body.surname, city: req.body.city, email: req.body.email }, (err, data1) => {

if (!err) {

res.redirect("/studentdata");

}

});

})

app.listen(port,()=>{

console.log("Server is running on port number : ",port);

});

**// Index3.html**

<html >

<head>

<title>Students</title>

</head>

</body>

<center>

<h1>Add Student Details</h1>

<form action="/student" method="POST">

<table>

<tr>

<td>Enter Name : </td>

<td><input type="text" name="name"></td>

</tr>

<tr>

<td>Enter Surname : </td>

<td><input type="text" name="surname"></td>

</tr>

<tr>

<td>Enter Email : </td>

<td><input type="text" name="email"></td>

</tr>

<tr>

<td>Enter City : </td>

<td><input type="text" name="city"></td>

</tr>

<tr>

<td></td>

<td><input type="submit" value="Add Student"></td>

</tr>

</table>

</form>

<table >

<tbody id="div1">

<tr>

<th>Id</th>

<th>Name</th>

<th>Surname</th>

<th>Email</th>

<th>City</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</tbody>

</table>

<script src="https://code.jquery.com/jquery-3.5.1.js"></script>

<script>

$(document).ready(() => {

var table

$.ajax({

url: '/student',

method: 'get',

dataType: 'json',

success: function (result) {

console.log("Length: " + result.length)

console.log(result[0].\_id);

for (let index = 0; index < result.length; index++) {

var html = `

<tr>

<td style="text-align: center;"> ` + result[index].\_id + ` </td>

<td style="text-align: center;"> ` + result[index].name + ` </td>

<td style="text-align: center;"> ` + result[index].surname + ` </td>

<td style="text-align: center;"> ` + result[index].email + ` </td>

<td style="text-align: center;"> ` + result[index].city + ` </td>

<td style="text-align: center;">

<form action="/delete?id=`+ result[index].\_id + ` " method="post"><input type="submit" value="Delete">

</form>

</td>

<td style="text-align: center;">

<form action="/updateData?id=`+ result[index].\_id + `" method="post"><input type="submit" value="Update">

</form>

</td>

</tr>

`;

//console.log(result);

$("#div1").append(html);

}

},

});

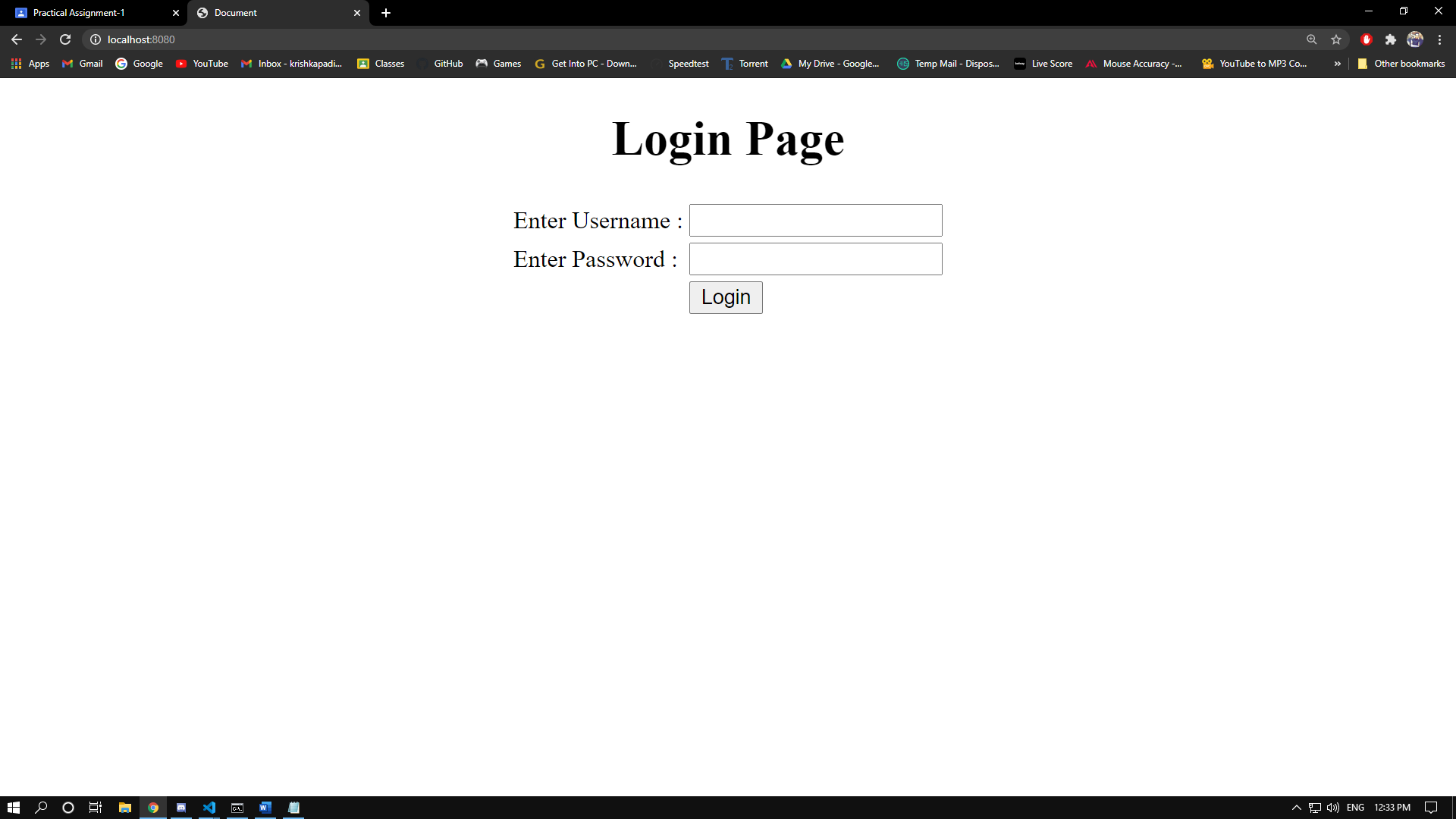
})

</script>

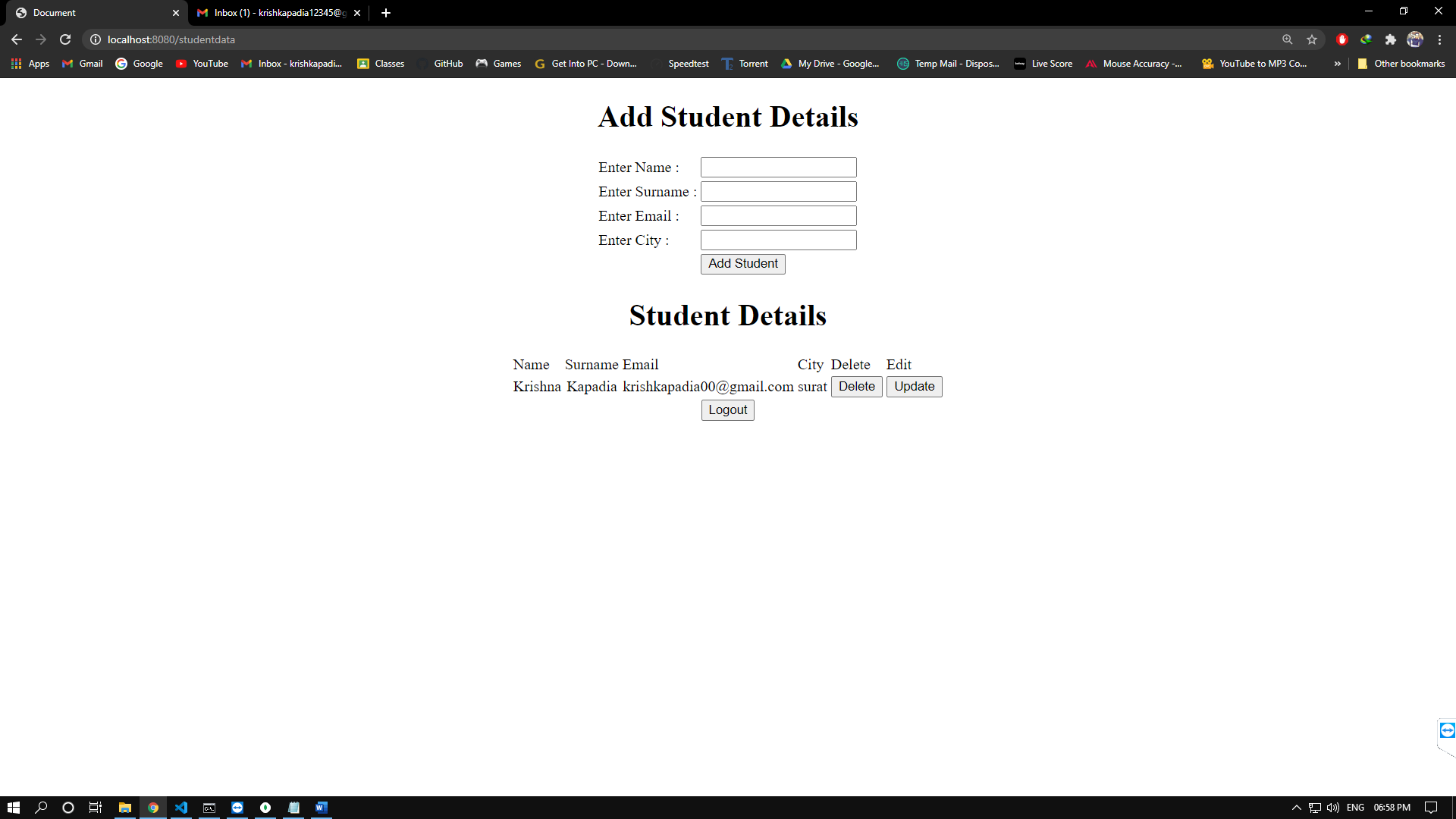
</center>

<body>

</html>

// Login Page

// Add Student Data



// Update Student Data