**Practical 15**

**AIM:**

**Create Secure Server using Nodejs and ExpressJs.**

**Note: Explore nodemon package to handle server automatically**

**CODE:**

const express = require("express");

const https = require("https");

const helmet = require("helmet");

const path = require("path");

const fs = require("fs");

const app = express();

app.use(helmet());

const port = 3001;

app.use(

  express.urlencoded({

    extended: true,

  })

);

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

  res.send("Welcome to ssl server");

});

const sslServer = https.createServer(

  {

    key: fs.readFileSync(path.join(\_\_dirname, "certi", "key.pem")),

    cert: fs.readFileSync(path.join(\_\_dirname, "certi", "cret.pem")),

  },

  app

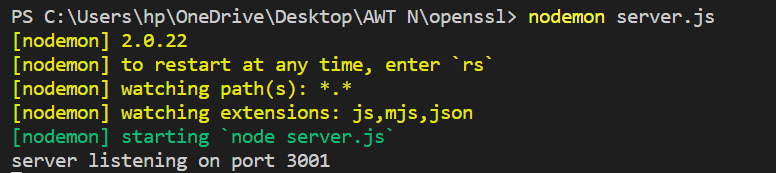
);

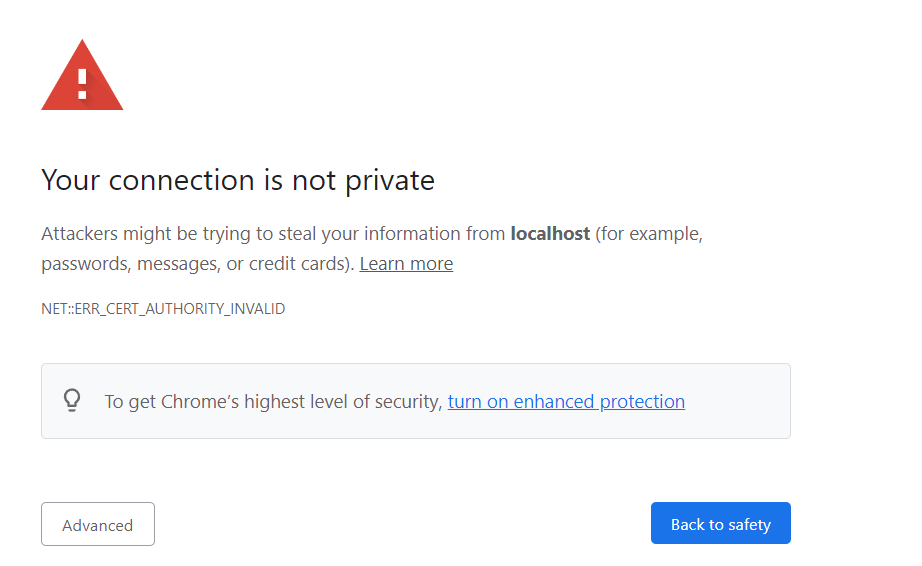
sslServer.listen(port, () => {

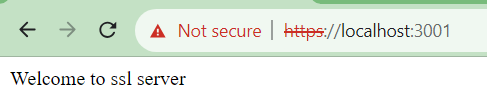
  console.log(`server listening on port ${port}`);

});

**OUTPUT:**







**Practical 16**

**AIM:**

**Create Login and Registration using ExpressJs.**

**Note:**

* **Apply Session, Cookies and JWT Token Concept.**
* **Apply Passport.js for Authentication middleware**

**CODE:**

**App.js**

const express = require("express");

const app = express();

const port = 3000;

const bodyParser = require("body-parser");

const session = require("express-session");

const cookieParser = require("cookie-parser");

const jwt = require("jsonwebtoken");

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

app.use(cookieParser());

app.use(

  session({

    secret: "secret",

    resave: true,

    saveUninitialized: true,

  })

);

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

  res.send("Hello World!");

});

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

  res.sendFile(\_\_dirname + "/Views/login.html");

});

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

  res.sendFile(\_\_dirname + "/Views/register.html");

});

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

  const username = req.body.username;

  const password = req.body.password;

  if (username && password) {

    const user = {

      username: username,

      password: password,

    };

    const token = jwt.sign(user, "secretkey", (err, token) => {

      res.cookie("token", token);

      res.redirect("/dashboard");

    });

  } else {

    res.send("Please enter Username and Password!");

  }

});

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

  const username = req.body.username;

  const password = req.body.password;

  if (username && password) {

    const user = {

      username: username,

      password: password,

    };

    const token = jwt.sign(user, "secretkey", (err, token) => {

      res.cookie("token", token);

      res.redirect("/dashboard");

    });

  } else {

    res.send("Please enter Username and Password!");

  }

});

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

  if (req.cookies.token) {

    const token = req.cookies.token;

    const decoded = jwt.verify(token, "secretkey");

    res.send(`Welcome ${decoded.username}`);

  } else {

    res.send("Please login to view this page!");

  }

});

app.listen(port, () => {

  console.log(`Example app listening at ${port}`);

});

**Practical 17**

**AIM:**

**Create Coure page which contains University Name, Institute Name, Department Name, CoursName, CourseCode, Semester. Handle.**

**Note:**

* **Create Middleware to Sanitize and validate All fileds**

**CODE:**

**Course.js**

const express = require("express");

const { sanitize, validate } = require("../middleware");

const router = express.Router();

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

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

});

router.post("/course", sanitize, validate, (req, res) => {

  const courseInfo = req.body;

  res.json({

    message: "Course information submitted successfully",

    data: courseInfo,

  });

});

module.exports = router;

**app.js**

const express = require("express");

const bodyParser = require("body-parser");

const { sanitize, validate } = require("./middleware");

const app = express();

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

app.use(bodyParser.json());

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

app.use("/course", sanitize, validate);

app.use(express.static("views"));

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

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

});

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

  const courseInfo = req.body;

  res.json({

    message: "Course information submitted successfully",

    data: courseInfo,

  });

});

app.use((req, res) => {

  res.status(404).json({ error: "Not Found" });

});

app.listen(port, () => {

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

});

**Middleware.js**

const { body, validationResult } = require("express-validator");

const sanitize = (req, res, next) => {

  Object.keys(req.body).forEach((key) => {

    req.body[key] = sanitizeInput(req.body[key]);

  });

  next();

};

const validate = [

  body("university").notEmpty().withMessage("University Name is required"),

  body("institute").notEmpty().withMessage("Institute Name is required"),

  body("department").notEmpty().withMessage("Department Name is required"),

  body("courseName").notEmpty().withMessage("Course Name is required"),

  body("courseCode").notEmpty().withMessage("Course Code is required"),

  body("semester").notEmpty().withMessage("Semester is required"),

  (req, res, next) => {

    const errors = validationResult(req);

    if (!errors.isEmpty()) {

      return res.status(400).json({ errors: errors.array() });

    }

    next();

  },

];

const sanitizeInput = (input) => {

  return input.trim();

};

module.exports = { sanitize, validate };

**coursepage,html**

<!-- views/coursePage.html -->

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

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

    <title>Course Page</title>

</head>

<body>

    <h1>Course Information</h1>

        <form action="/course" method="post" enctype="application/x-www-form-urlencoded">

        <label for="university">University Name:</label>

        <input type="text" name="university" required>

        <br>

        <label for="institute">Institute Name:</label>

        <input type="text" name="institute" required>

        <br>

        <label for="department">Department Name:</label>

        <input type="text" name="department" required>

        <br>

        <label for="courseName">Course Name:</label>

        <input type="text" name="courseName" required>

        <br>

        <label for="courseCode">Course Code:</label>

        <input type="text" name="courseCode" required>

        <br>

        <label for="semester">Semester:</label>

        <input type="text" name="semester" required>

        <br>

        <button type="submit">Submit</button>

    </form>

</body>

</html>

**OUTPUT:**

