**Advanced Software Engineering**

**Lab-9**

**Team 1\_2**

Srichakradhar Reddy (32)  
Sathwik(5)

JWT tokens are used for authorization in HTTP environments.

**REST API for JWT authorization server**

const express = require('express')

const app = express()

const port = 3000

app.get('/', (req, res) => res.send('Hello World!'))

app.listen(port,

() => console.log(`Simple Express app listening on port ${port}!`))

### The routes to serve secret and readme are used to provide the resoponse

Secret – protected by JWT authorization

readme – public and unprotected

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

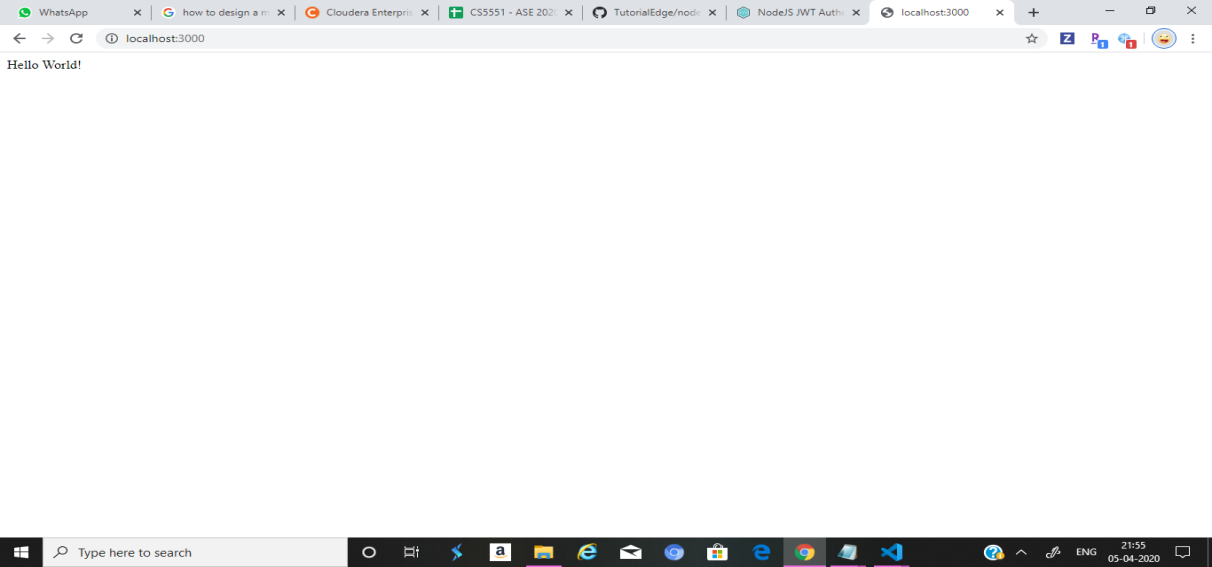
res.json({ "message" : "THIS IS SUPER SECRET, DO NOT SHARE!" })

})

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

res.json({ "message" : "This is open to the world!" })

})



**NodeJS app – default route**

**Route for authorization with JWT**

**const jwt = require('jsonwebtoken');**

const fs = require('fs')

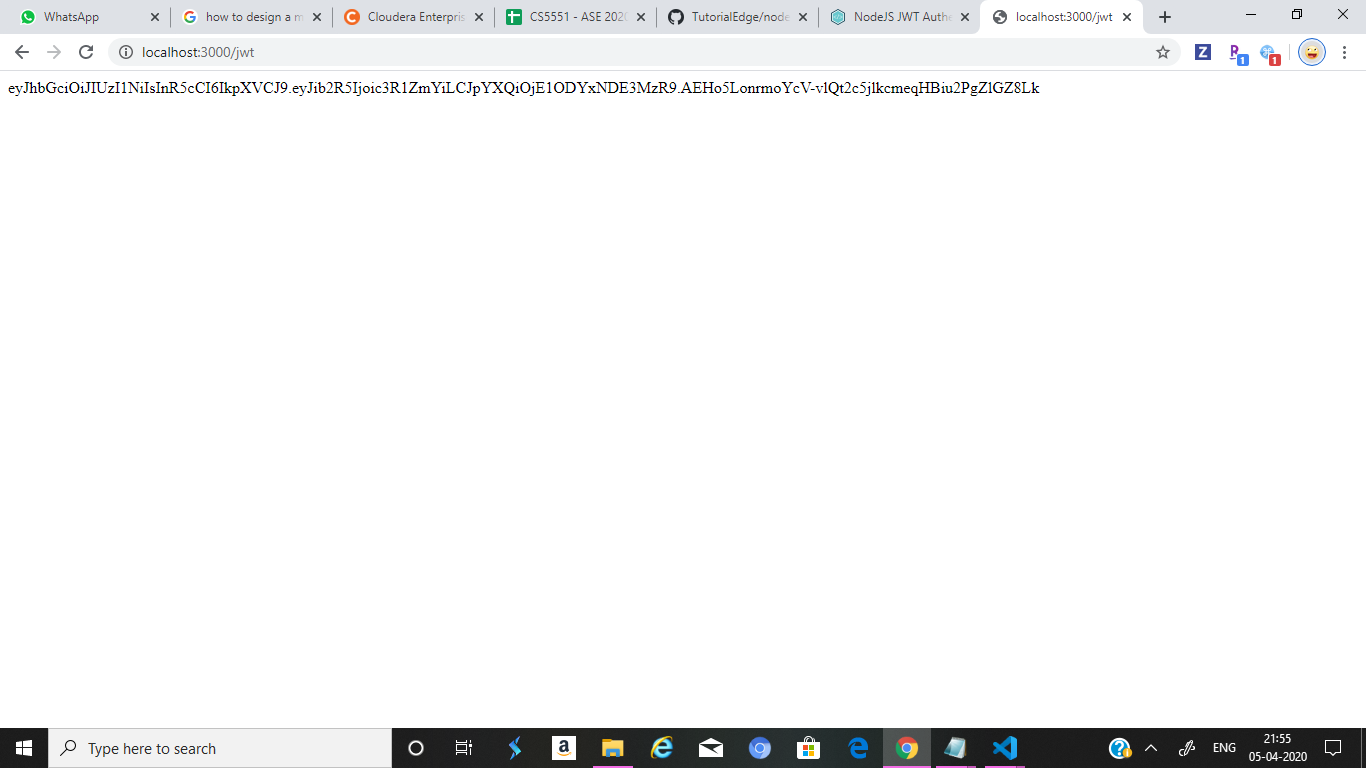
app.get('/jwt', (req, res) => {

let privateKey = fs.readFileSync('./private.pem', 'utf8');

let token = jwt.sign({ "body": "stuff" }, "MySuperSecretPassPhrase", { algorithm: 'HS256'});

res.send(token);

})



**NodeJS server - JWT token endpoint**

function isAuthenticated(req, res, next) {

if (typeof req.headers.authorization !== "undefined") {

let token = req.headers.authorization.split(" ")[1];

let privateKey = fs.readFileSync('./private.pem', 'utf8');

jwt.verify(token, privateKey, { algorithm: "HS256" }, (err, user) => {

if (err) {

res.status(500).json({ error: "Not Authorized" });

throw new Error("Not Authorized");

}

return next();

});

} else {

res.status(500).json({ error: "Not Authorized" });

throw new Error("Not Authorized");

}

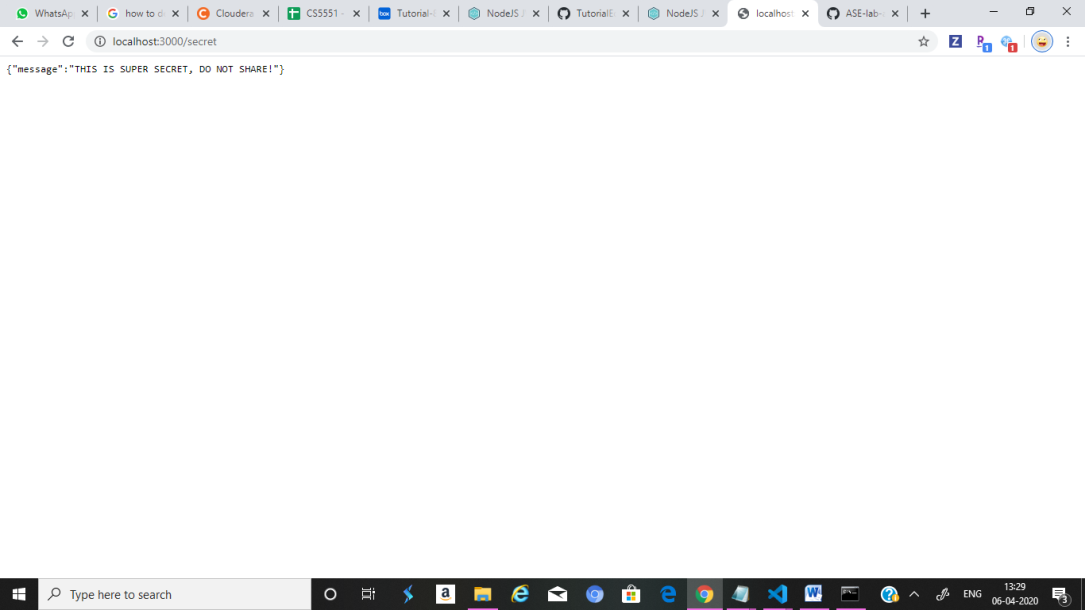
}

The isAuthorized middleware function is used to authorize requests

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

res.json({ "message" : "THIS IS SUPER SECRET, DO NOT SHARE!" })

})



**When authorized user access the app**

