**1) Develop a user registration form and store its data in any database using Express.  Form should also contain file upload (single, multiple) with validations.**

**Model :-**

const mongoose = require('mongoose')

const Q1Schema = new mongoose.Schema({

name: {

type:String

},

pass: {

type:String

},

email: {

type:String

},

phone: {

type:Number

},

avatar: {

type:String

}

})

module.exports = mongoose.model('Q1', Q1Schema)

**upload.js :-**

const express = require('express')

const mongoose = require('mongoose')

const multer = require('multer')

const path = require('path');

const app = express();

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

const Q1 = require('./model/q1')

mongoose.connect('mongodb://0.0.0.0:27017/Q1DBex')

const db = mongoose.connection

db.on('open', ()=>{

console.log('Connected to database!!')

})

var options = multer.diskStorage({

destination : function (req, file, cb) {

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

{

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

}

cb(null, './uploads');

} ,

filename: function (req, file, cb) {

cb(null, (Math.random().toString(30)).

slice(5, 10) + Date.now()

+ path.extname(file.originalname));

}

});

var upload= multer({ storage: options });

app.post("/url",()=>{})

app.post('/file\_upload', upload.single("myfile"), function (req, res, next) {

res.write(req.body.fname+" "+req.body.lname+"\n");

res.write("file uploaded");

res.end();

})

app.post('/photos\_upload',upload.array('photos',2), (req, res) => {

// req.files is array of `photos` files

console.log(req.files)

const add = new Q1({

name: req.body.name,

pass: req.body.pass,

email: req.body.email,

phone: req.body.phone,

avatar: req.body.photos

})

try {

const newq1 = add.save()

res.redirect('/data')

} catch (error) {

console.log('Error'+error)

}

})

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

try {

const data = await Q1.find()

res.json(data)

} catch (error) {

error

}

})

app.use(function (err, req, res, next) {

if (err instanceof multer.MulterError)

{

console.log("ERRRR");

res.status(500).send("file upload err "+err.message);

}

else

next(err);

});

app.listen(8000);

i**ndex.html :-**

<html>

<head>

<title>File Uploading Form</title>

</head>

<body>

<h3>Insert data & File Upload:</h3>

<br />

<form action = "/photos\_upload" method = "POST"

enctype = "multipart/form-data">

Name: <input type="text" name="name" size="50" /><br>

Password: <input type="password" name="pass" size="50" /><br>

Email: <input type="text" name="email"/><br>

Phone: <input type="number" name="phone" /><br>

Select a file : <input type="file" name="photos" size="500" multiple="multiple" />

<br />

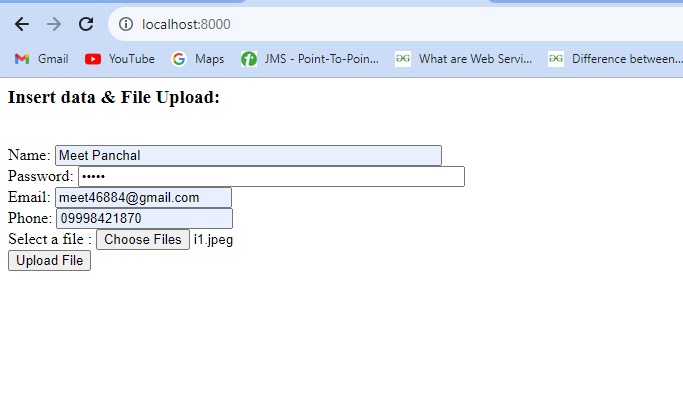
<input type = "submit" value = "Upload File" />

</form>

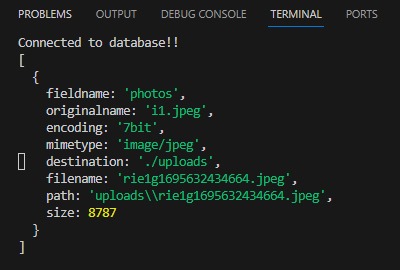
</body>

</html>

**Output :-**







**2) Express Login application with file session store.**

**Index.js :-**

const express = require('express')

const app = express()

const session = require('express-session')

const path = require('path')

const FileStore = require('session-file-store')(session)

const PORT = 3000

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

app.use(express.urlencoded({ extended: false }))

app.use(session({

secret: 'secret',

resave: false,

saveUninitialized: false,

store: new FileStore({ path: './session-data' })

}))

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

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

})

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

var username = req.body.username;

var password = req.body.password;

if (username && password) {

//credential validation logic here, if credential matches than we store data in session

req.session.loggedIn = true;

req.session.username = username

console.log(req.body)

return res.redirect('/home')

} else {

return res.send('Please Enter Username And Password!')

}

})

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

if (req.session.loggedIn) {

console.log('logged in')

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

} else {

console.log('not logged in')

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

}

})

app.listen(PORT, () => {

console.log(`server listening on http://localhost:${PORT}`)

})

**Home.html :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Home Page</title>

</head>

<body>

<div class="container">

<h2>Home Page, You Are Authentic User</h2>

</div>

</body>

</html>

**Login.html :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Log In Page</title>

<style>

.mb-2 {

margin-bottom: 2rem;

}

.mt-2 {

margin-top: 2rem;

}

.flex-center {

display: flex;

justify-content: center;

flex-direction: column;

align-items: center;

}

\* {

font-size: 1.2rem;

}

</style>

</head>

<body>

<div class="container flex-center">

<div class="header">

<h3>Log In</h3>

</div>

<div class="form-container mt-2 ">

<form class="flex-center" action="/login" method="post">

<div class="username mb-2">

<input type="text" placeholder="enter username" name="username" required>

</div>

<div class="password mb-2">

<input type="password" name="password" placeholder="enter password" id="" required>

</div>

<div class="submit">

<input type="submit" value="Log In">

</div>

</form>

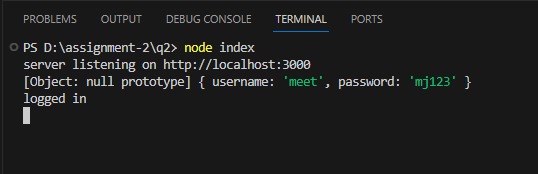
</div>

</div>

</body>

</html>

**Output :-**





**3) Express Login application with redis session store.**

**Output :-**

**4) Login with JWT, CRUD operations for students table with mongoose, express and any one template engine, Logout.**

**Database :-**

var mongoose = require('mongoose');

//database config

const server = "127.0.0.1:27017"

const Database = "studentDB"

class database {

constructor() {

this.\_connect()

}

async \_connect() {

try {

await mongoose.connect(`mongodb://${server}/${Database}`)

console.log(`database connection successfull`)

} catch (error) {

console.error('Database connection error: ' + error);

}

}

}

module.exports = new database()

**Middleware :-**

const jwt = require('jsonwebtoken')

const signToken = async (req, res) => {

//TODO: implement sign token logic here

}

const verifyToken = async (req, res, next) => {

const token = req?.cookies?.token

if (token) {

//verify token

if (jwt.verify(token, process.env.TOKEN\_SECRET)) {

// return res.send('token found and verified');

next()

} else {

return res.status(401).end('Unauthorized access');

}

}

else {

//redirect to login

return res.status(401).send('Token Missing')

}

}

module.exports = { signToken, verifyToken }

**Models :-**

const mongoose = require('mongoose')

const studentSchema = new mongoose.Schema({

name: String,

email: String,

password: String,

age: Number,

gender: String,

city: String

})

module.exports = new mongoose.model('student', studentSchema, 'students')

**Routes :-**

const express = require('express');

const { verifyToken } = require('../middleware/token');

const student = express.Router();

const studentModel = require('../models/student')

student.use(verifyToken)

//create

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

res.render('add');

})

student.post('/', async (req, res) => {

try {

const newStudent = req.body;

const student = new studentModel({

...newStudent

})

await student.save()

res.redirect('/')

} catch (error) {

res.send('There was problem creating students')

}

})

//read

student.get('/:id', async (req, res) => {

try {

const student = await studentModel.findById(req.params.id)

student ? res.render('view', { data: student }) : res.render('index', { noDataError: 'No students found' })

} catch (error) {

res.send('There was problem getting student')

}

})

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

try {

const students = await studentModel.find()

students ? res.render('index', { data: students }) : res.render('index', { noDataError: 'No students found' })

} catch (error) {

res.send('There was problem getting students')

}

})

//update

student.get('/edit/:id', async (req, res) => {

try {

const student = await studentModel.findById(req.params.id)

student ? res.render('update', { data: student }) : res.render('index', { noDataError: 'student not found' })

} catch (error) {

res.send('There was problem getting student')

}

})

student.post('/edit/:id', async (req, res) => {

console.log('student post /:id')

try {

const updatedStudent = req.body;

const id = req.params.id;

console.log('updated student: ', updatedStudent)

delete updatedStudent['\_id']

await studentModel.findOneAndUpdate({ \_id: id }, updatedStudent)

return res.redirect('/')

} catch (error) {

return res.send('There was problem updating students')

}

})

//delete

student.get('/delete/:id', async (req, res) => {

try {

const student = await studentModel.findById(req.params.id)

student ? res.render('delete', { data: student }) : res.render('index', { noDataError: 'student not found' })

} catch (error) {

res.send('There was problem getting student')

}

})

student.post('/delete', async (req, res) => {

console.log('delete student request')

try {

const id = req.body.\_id

console.log("id: ", id)

await studentModel.findOneAndDelete({ \_id: id })

res.redirect('/')

} catch (error) {

console.log('Error: ', error)

res.send('There was problem deleting students')

}

})

module.exports = student

**Add.ejs :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

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

<title>Add Student</title>

</head>

<body>

<div class="container flex-center">

<div class="header">

<h3>Add Student</h3>

</div>

<div class="err">

<% if(typeof errors!=='undefined' ){ %>

<p>

<%= errors.validationError %>

</p>

<% } %>

</div>

<div class="form-container mt-2 ">

<form class="flex-center" action="/student/" method="post">

<div class="username mb-2">

<input type="text" placeholder="enter your name" name="name" required>

</div>

<div class="password mb-2">

<input type="password" name="password" placeholder="enter password" id="" required>

</div>

<div class="email mb-2">

<input type="email" name="email" id="" placeholder="enter your email" required>

</div>

<div class="city mb-2">

<input type="text" name="city" id="" placeholder="enter your city" required>

</div>

<div class="age mb-2">

<input type="number" name="age" id="" min="10" max="80" placeholder="Age here" required>

</div>

<div class="gender mb-2">

<input type="radio" name="gender" value="Male" id="" checked>Male

<input type="radio" name="gender" value="Female" id="">Female

</div>

<div class="submit">

<input style="padding:0.4rem" type="submit" value="Add">

</div>

</form>

</div>

</div>

<div class="links mt-2">

<a href="/logout">Log Out</a>

</div>

</body>

</html>

**Delete.ejs :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Student Detail</title>

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

</head>

<body>

<% if(typeof(data)!=='undefined' ) { %>

<div class="student-container">

<form class="flex-center" action="/student/delete" method="post">

<input type="hidden" name="\_id" value="<%= data?.\_id %>">

<div class="username mb-2">

<input type="text" placeholder="enter your name" name="name" value="<%= data?.name %>" disabled>

</div>

<div class="password mb-2">

<input type="password" name="password" placeholder="enter password" value="<%= data?.password %>"

id="" disabled>

</div>

<div class="email mb-2">

<input type="email" name="email" id="" placeholder="enter your email" value="<%= data?.email %>"

disabled>

</div>

<div class="city mb-2">

<input type="text" name="city" id="" placeholder="enter your city" value="<%= data?.city %>"

disabled>

</div>

<div class="age mb-2">

<input type="number" name="age" id="" min="10" max="80" placeholder="Age here"

value="<%= data?.age %>" disabled>

</div>

<div class="gender mb-2">

<input disabled type="radio" name="gender" value="Male" id="" <% if(data?.gender==='Male' )

{%>checked<%} %>

>Male

<input disabled type="radio" name="gender" value="Female" id="" <% if(data?.gender==='Female'

){%>

checked<% }%>>Female

</div>

<div class="submit">

<input style="padding:0.4rem" type="submit" value="Confirm Delete">

</div>

</form>

</div>

<div class="links">

<a href="/student/">Go Back</a>

</div>

<div class="links mt-2">

<a href="/logout">Log Out</a>

</div>

<% } else{ %>

Student Not Available!

<%}%>

</body>

</html>

**Index.ejs :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Home Page</title>

<style>

.err {

color: rebeccapurple;

}

.mr-1 {

margin-right: 0.3rem

}

.mt-2 {

margin-top: 2rem;

}

</style>

</head>

<body>

<div class="container">

<div class="error">

<% if(typeof(noDataError)!=='undefined' ) { %>

<p class="err">

<%= {noDataError} %>

</p>

<%}%>

<% if(typeof(accessToken)!=='undefined' ) { %>

<script>

localStorage.setItem('token', "<%= JSON.stringify(accessToken) %>"); // <- setup token into localStorage, (but i think it's not good place for that, and would be better get token with another authorization request)

</script>

<%}%>

</div>

<div class="error">

<% if(typeof(data)!=='undefined' ) { %>

<table border="1">

<thead>

<tr>

<th>name</th>

<th>email</th>

<th>age</th>

<th>city</th>

<th>gender</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

<% data.forEach(student=>{

%>

<tr>

<td>

<%= student.name %>

</td>

<td>

<%= student.email %>

</td>

<td>

<%= student.age %>

</td>

<td>

<%= student.city %>

</td>

<td>

<%= student.gender %>

</td>

<td>

<div class="mr-1">

<a href="/student/edit/<%=student.\_id%>" class="mr-1">edit</a>

<a href="/student/delete/<%=student.\_id%>" class="mr-1">delete</a>

<a href="/student/<%=student.\_id%>" class="mr-1">view</a>

</div>

</td>

</tr>

<% }) %>

</tbody>

</table>

<%}%>

<div class="links mt-2">

<a href="/student/add">Add Student</a>

</div>

<div class="links mt-2">

<a href="/logout">Log Out</a>

</div>

</div>

</div>

</body>

</html>

**Login.ejs :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Log In Page</title>

<style>

.mb-2 {

margin-bottom: 2rem;

}

.mt-2 {

margin-top: 2rem;

}

.flex-center {

display: flex;

justify-content: center;

flex-direction: column;

align-items: center;

}

\* {

font-size: 1.2rem;

}

.err {

color: red;

}

</style>

</head>

<body>

<div class="container flex-center">

<div class="header">

<h3>Log In</h3>

</div>

<div class="err">

<% if(typeof errors!=='undefined' ){ %>

<p>

<%= errors.validationError %>

</p>

<% } %>

</div>

<div class="form-container mt-2 ">

<form class="flex-center" action="/login" method="post">

<div class="username mb-2">

<input type="text" placeholder="enter username" name="username" required>

</div>

<div class="password mb-2">

<input type="password" name="password" placeholder="enter password" id="" required>

</div>

<div class="submit">

<input type="submit" value="Log In">

</div>

</form>

</div>

</div>

</body>

</html>

**Update.ejs :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Edit Student</title>

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

</head>

<body>

<div class="container flex-center">

<div class="header">

<h3>Edit Student</h3>

</div>

<div class="err">

<% if(typeof errors!=='undefined' ){ %>

<p>

<%= errors.validationError %>

</p>

<% } %>

</div>

<div class="form-container mt-2 ">

<form class="flex-center" action="/student/edit/<%= data?.\_id %>" method="post">

<input type="hidden" name="\_id" value="<%= data?.\_id %>">

<div class="username mb-2">

<input type="text" placeholder="enter your name" name="name" value="<%= data?.name %>" required>

</div>

<div class="password mb-2">

<input type="password" name="password" placeholder="enter password" value="<%= data?.password %>"

id="" required>

</div>

<div class="email mb-2">

<input type="email" name="email" id="" placeholder="enter your email" value="<%= data?.email %>"

required>

</div>

<div class="city mb-2">

<input type="text" name="city" id="" placeholder="enter your city" value="<%= data?.city %>"

required>

</div>

<div class="age mb-2">

<input type="number" name="age" id="" min="10" max="80" placeholder="Age here"

value="<%= data?.age %>" required>

</div>

<div class="gender mb-2">

<input type="radio" name="gender" value="Male" id="" <% if(data?.gender==='Male' ) {%>checked<%} %>

>Male

<input type="radio" name="gender" value="Female" id="" <% if(data?.gender==='Female' ){%>

checked<% }%>>Female

</div>

<div class="submit">

<input style="padding:0.4rem" type="submit" value="Update">

</div>

</form>

</div>

</div>

<div class="links mt-2">

<a href="/logout">Log Out</a>

</div>

</body>

</html>

**View.ejs :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Student Detail</title>

</head>

<body>

<% if(typeof(data)!=='undefined' ) { %>

<div class="student-container">

<p>name: <%= data.name %>

</p>

<p>email: <%= data.email %>

</p>

<p>age: <%= data.age %>

</p>

<p>city: <%= data.city %>

</p>

<p>gender: <%= data.gender %>

</p>

</div>

<div class="links">

<a href="/student/">Go Back</a>

</div>

<div class="links mt-2">

<a href="/logout">Log Out</a>

</div>

<% } else{ %>

Student Not Available!

<%}%>

</body>

</html>

**Index.js :-**

const express = require('express')

const app = express()

const PORT = 8000

const database = require('./db/database')

const student = require('./routes/student')

require('dotenv').config()

const jwt = require('jsonwebtoken')

const cookieParser = require('cookie-parser')

const accessTokenSecret = process.env.TOKEN\_SECRET;

const studentModel = require('./models/student')

const verifyToken = require('./middleware/token')

app.use(cookieParser())

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

app.use(express.json())

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

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

app.use('/student', student)

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

const token = req.cookies.token

if (token) {

//verify token

if (jwt.verify(token, process.env.TOKEN\_SECRET)) {

return res.redirect('/student')

} else {

return res.status(401).end('Unauthorized access');

}

}

else {

//redirect to login

return res.render('login')

}

})

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

let { username, password } = req.body

if (username && password) {

try {

let result = await studentModel.find({ name: username, password: password })

if (result.length === 1) {

//Generate Token

const accessToken = jwt.sign({ name: username, password: password }, accessTokenSecret)

res.cookie('token', accessToken, { httpOnly: true });

return res.redirect('/student')

// return res.render('index', { session: result[0], accessToken: accessToken })

} else {

res.render('login', { errors: { validationError: 'Enter Valid Username Or Password' } })

}

} catch (error) {

res.render('login', { errors: { validationError: 'There Was Problem While Log In.' } })

}

} else {

res.render('login', { errors: { AllFieldsError: 'Please Enter Username And Password' } })

}

})

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

res.clearCookie('token')

res.render('login')

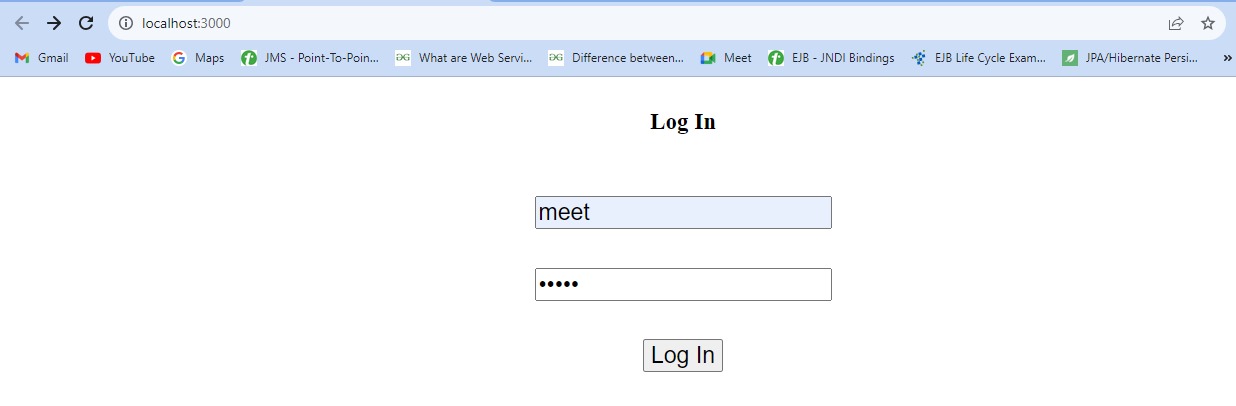
})

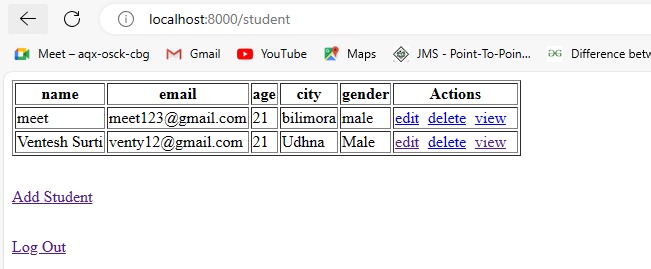
app.listen(PORT, () => {

console.log(`app listening on http://localhost:${PORT}`)

})

**Output :-**



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

**Auth :-**

const jwt = require('jsonwebtoken')

const sign = (data) => {

const token = jwt.sign({ ...data }, process.env.TOKEN\_SECRET, { algorithm: 'HS256', expiresIn: process.env.EXPIRY })

return token

}

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

let token = req.headers["authorization"] || req.headers["x-access-token"];

token = token?.split(" ")[1]

if (token) {

try {

if (jwt.verify(token, process.env.TOKEN\_SECRET)) {

return next()

} else {

return res.status(401).send('Unauthorized Access')

}

} catch (JsonWebTokenError) {

return res.status(401).send('Something went wrong!');

}

} else {

return res.status(401).send('Unauthorized Access')

}

}

module.exports = { sign, verify }

**Database :-**

var mongoose = require('mongoose');

const DB\_URL = process.env.DB\_URL

class database {

constructor() {

this.\_connect()

}

async \_connect() {

try {

await mongoose.connect(DB\_URL)

console.log(`database connection successfull`)

} catch (error) {

console.error('Database connection error: ' + error);

}

}

}

module.exports = new database()

**Model :-**

const mongoose = require('mongoose')

const studentSchema = new mongoose.Schema({

name: String,

email: String,

password: String,

age: Number,

gender: String,

city: String

})

module.exports = mongoose.model('student', studentSchema, 'students')

**Routes :-**

const express = require('express')

const app = express.Router()

//model

const student = require('../models/studentModel')

//getting all students

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

try {

const data = await student.find()

if (data) {

return res.send(data)

} else {

return res.status(404).send('No Data Found')

}

} catch (error) {

console.error(error);

return res.status(500).send('Something went wrong')

}

})

//getting single student

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

const id = req.params.id

try {

const data = await student.findById(id)

if (data) {

return res.send(data)

} else {

return res.status(404).send('No Data Found')

}

} catch (error) {

console.log(error)

res.status(500).send('Something went wrong')

}

})

//creating student

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

try {

const reqData = req.body

const newStudent = new student({ ...reqData })

const response = await newStudent.save()

res.json("Student added successfully")

} catch (error) {

console.log(error)

res.status(500).send('Something went wrong')

}

})

//updating student

app.put('/:id', async (req, res) => {

try {

const reqData = req.body

const id = req.params.id

const response = await student.findByIdAndUpdate(id, reqData)

if (response) {

return res.json("Student updated successfully")

} else {

return res.status(404).send('No Data Found For Given Id')

}

} catch (error) {

console.error(error);

res.status(500).send('Something went wrong')

}

})

//deleting student

app.delete('/:id', async (req, res) => {

try {

const id = req.params.id

const response = await student.findByIdAndDelete(id)

if (response) {

res.json('student deleted successfully')

} else {

res.status(500).json('Something went wrong')

}

} catch (error) {

console.log(error)

res.status(500).json('Something went wrong')

}

})

module.exports = app

**index.js :-**

const express = require('express')

const app = express()

const cors = require('cors')

const { sign, verify } = require('./auth/authenticate')

require('dotenv').config()

//database

require('./db/database')

//router

const studentRouter = require('./routes/studentRoute')

//model

const student = require('./models/studentModel')

//config

const PORT = process.env.PORT

//cors

app.use(cors())

//body-parser and json for sending json and reading body

app.use(express.json())

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

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

console.log('default response')

res.send('DEFAULT RESPONSE FROM SERVER')

})

//login request

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

let { username, password } = req.body

username = username

password = password

if (username && password) {

let response = await student.find({ name: username, password: password });

if (response.length === 1) {

return res.json(sign({ username }))

} else {

return res.status(401).json('Invalid username or password')

}

} else {

return res.status(401).json('Please provide username and password')

}

})

app.use('/get-token', sign)

app.use('/student', [verify], studentRouter)

app.listen(PORT, () => {

console.log(`app listening on http://localhost:${PORT}`);

})

**Index.html :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Home Page</title>

<style>

.err {

color: rebeccapurple;

}

.mr-1 {

margin-right: 0.3rem

}

.mt-2 {

margin-top: 2rem;

}

</style>

</head>

<body>

<div class="container">

<div class="error">

<table border="1">

<thead>

<tr>

<th>name</th>

<th>email</th>

<th>age</th>

<th>city</th>

<th>gender</th>

<th>Actions</th>

</tr>

</thead>

<tbody id="tableBody">

</tbody>

</table>

<div class="links mt-2">

<a href="./add.html">Add Student</a>

</div>

<div class="links mt-2">

<a href="./logout.html">Log Out</a>

</div>

</div>

</div>

</body>

<script>

const tableBody = document.getElementById('tableBody')

const fetchStudents = async () => {

const url = "http://localhost:8000/student"

const response = await fetch(url, {

headers: {

'Authorization': `Bearer ${localStorage.getItem('authToken')}`

}

})

return response.json()

}

fetchStudents().then(data => {

data.forEach(student => {

tableBody.innerHTML = tableBody.innerHTML + `

<tr>

<td>

${student.name}

</td>

<td>

${student.email}

</td>

<td>

${student.age}

</td>

<td>

${student.city}

</td>

<td>

${student.gender}

</td>

<td>

<div class="mr-1">

<a href="./update.html?id=${student.\_id}" class="mr-1">edit</a>

<a href="./delete.html?id=${student.\_id}" class="mr-1">delete</a>

<a href="./view.html?id=${student.\_id}" class="mr-1">view</a>

</div>

</td>

</tr>

`

});

})

</script>

</html>

**View.html :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Student Detail</title>

</head>

<body>

<div class="student-container" id="studentContainer">

</div>

<div class="links">

<a href="./student/">Go Back</a>

</div>

<div class="links mt-2">

<a href="./logout.html">Log Out</a>

</div>

</body>

<script>

const studentContainer = document.getElementById('studentContainer')

const fetchStudent = async (id) => {

console.log('fetchStudent')

const url = `http://localhost:8000/student/${id}`

const response = await fetch(url, {

headers: {

'Authorization': `Bearer ${localStorage.getItem('authToken')}`

}

})

return response.json()

}

const searchUrl = window.location.search

id = searchUrl?.split("?")[1]?.split("=")[1]

if (id) {

fetchStudent(id).then(data => {

studentContainer.innerHTML = `

<p>name: ${data.name}</p>

<p>email: ${data.email}</p>

<p>age: ${data.age}</p>

<p>city: ${data.city}</p>

<p>gender: ${data.gender}</p>

`

})

} else {

alert("Please provide student id")

}

</script>

</html>

**Login.html :-**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Log In Page</title>

<style>

.mb-2 {

margin-bottom: 2rem;

}

.mt-2 {

margin-top: 2rem;

}

.flex-center {

display: flex;

justify-content: center;

flex-direction: column;

align-items: center;

}

\* {

font-size: 1.2rem;

}

.err {

color: red;

}

</style>

</head>

<body>

<div class="container flex-center">

<div class="header">

<h3>Log In</h3>

</div>

<div class="err">

</div>

<div class="form-container mt-2 ">

<div class="flex-center">

<div class="username mb-2">

<input type="text" placeholder="enter username" name="username" id="txtusername" required>

</div>

<div class="password mb-2">

<input type="password" name="password" placeholder="enter password" id="txtpassword" required>

</div>

<div class="submit">

<input type="submit" value="Log In" onclick="handleLogin()">

</div>

</div>

</div>

</div>

</body>

<script>

const usernameField = document.getElementById("txtusername")

const passwordField = document.getElementById("txtpassword")

const handleLogin = async () => {

let username = usernameField.value

let password = passwordField.value

postData('http://localhost:8000/login', { username, password })

.then((data) => {

if (data) {

localStorage.setItem('authToken', data)

window.location.replace('./index.html')

}

}).catch(error => {

console.log(error)

});

}

async function postData(url = '', data = {}) {

// Default options are marked with \*

try {

const response = await fetch(url, {

method: 'POST', // \*GET, POST, PUT, DELETE, etc.

mode: 'cors', // no-cors, \*cors, same-origin

cache: 'no-cache', // \*default, no-cache, reload, force-cache, only-if-cached

credentials: 'same-origin', // include, \*same-origin, omit

headers: {

'Content-Type': 'application/json'

// 'Content-Type': 'application/x-www-form-urlencoded',

},

redirect: 'follow', // manual, \*follow, error

referrerPolicy: 'no-referrer', // no-referrer, \*no-referrer-when-downgrade, origin, origin-when-cross-origin, same-origin, strict-origin, strict-origin-when-cross-origin, unsafe-url

body: JSON.stringify(data) // body data type must match "Content-Type" header

});

if (response.status === 200) {

return response.json(); // parses JSON response into native JavaScript objects

} else {

let msg = await response.json()

alert(msg)

}

} catch (error) {

alert(error)

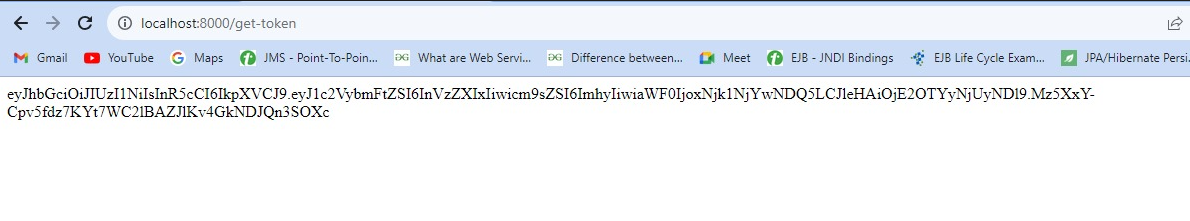
}

}

</script>

</html>

**Output :-**

****

