

Patel Anjali Jayeshbhai

7th Semester

Roll no: 17

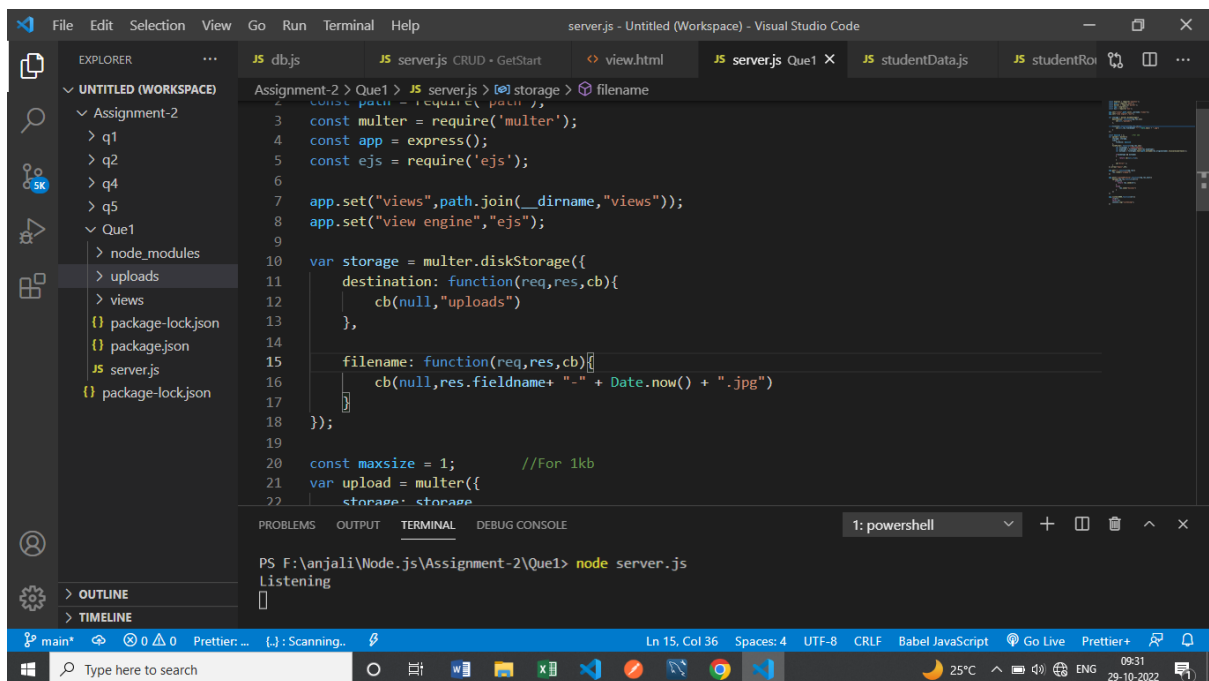
Node.JS Practical (705)

Practical Assignment-2

Que-1

→Code:

Server.js



```
Assignment-2 > Que1 > JS server.js > storage > filename
1  const path = require('path');
2  const multer = require('multer');
3  const app = express();
4  const ejs = require('ejs');
5
6
7  app.set("views",path.join(__dirname,"views"));
8  app.set("view engine","ejs");
9
10 var storage = multer.diskStorage({
11   destination: function(req,res,cb){
12     cb(null,"uploads")
13   },
14   filename: function(req,res,cb){
15     cb(null,res.fieldname+ "-" + Date.now() + ".jpg")
16   }
17 });
18
19
20 const maxsize = 1; //For 1kb
21 var upload = multer({
22   storage: storage
```

1: powershell

PS F:\anjali\Node.js\Assignment-2\Que1> node server.js
Listening

```
const express = require('express');
const path = require('path');
const multer = require('multer');
const app = express();
const ejs = require('ejs');

app.set("views",path.join(__dirname,"views"));
app.set("view engine","ejs");

var storage = multer.diskStorage({
  destination: function(req,res,cb){
```

```

        cb(null,"uploads")
    },

    filename: function(req,res,cb){
        cb(null,res.fieldname+ "-" + Date.now() + ".jpg")
    }
});

const maxsize = 1;    //For 1kb
var upload = multer({
    storage: storage,
    limits: {
        fileSize: maxsize
    },
    fileFilter: function(req,res,cb){
        var filetypes = /jpeg|jpg|png/;
        var mimetype = filetypes.test(res.mimetype);
        var extname = filetypes.test(path.extname(res.originalname).toLocaleLower
rCase());

        if(mimetype && extname)
        {
            return cb(null,true);
        }

        cb("Error" );
    }
}).array("mypic",10);

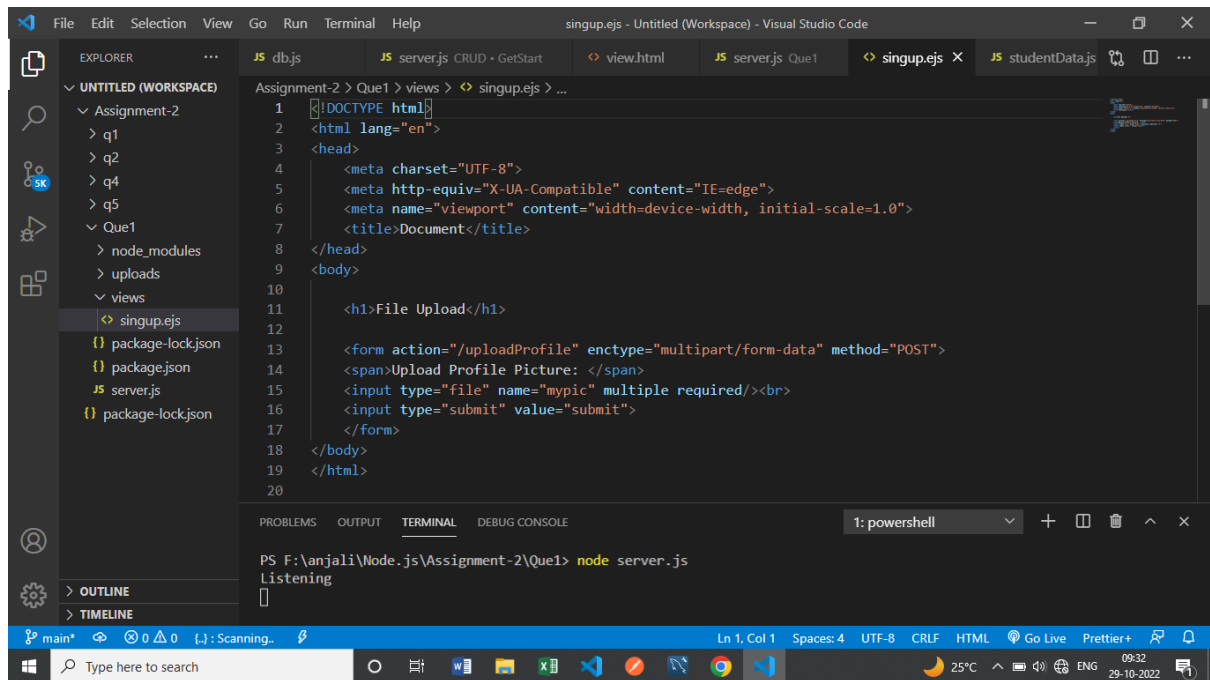
app.get("/",function(req,res){
    res.render("singup");
})

app.post("/uploadProfile",function(req,res,next){
    upload(req,res,function(err){
        if(err){
            return res.send(err);
        }
        else{
            res.send("Success")
        }
    })
})
})

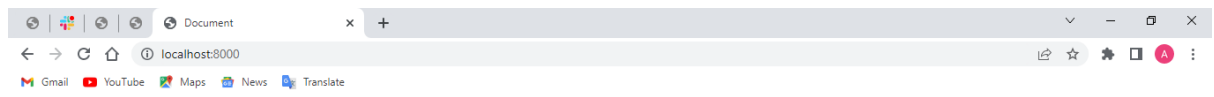
```

```
app.listen(8000,function(err){
  if(err)
    throw err
  console.log("Listening");
})
```

Signup.ejs



➔ Output:



File Upload

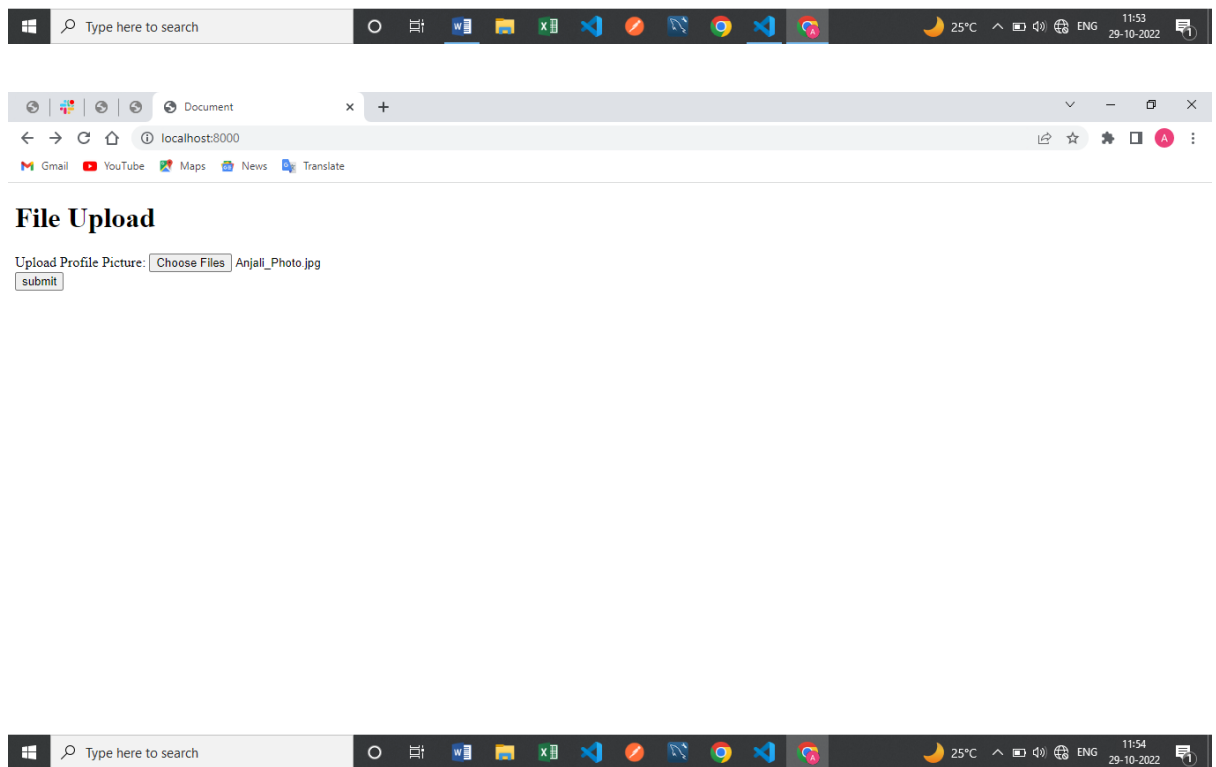
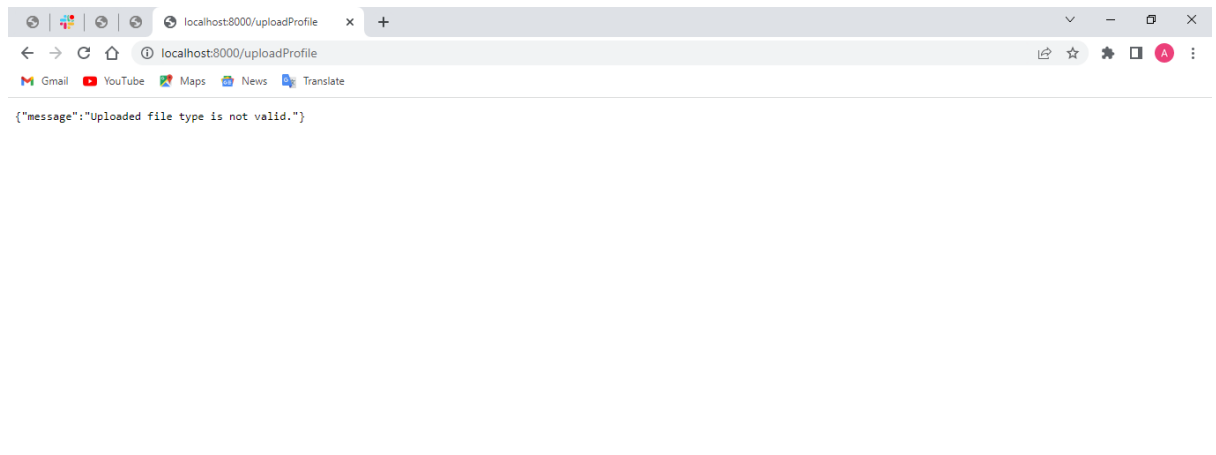
Upload Profile Picture: No file chosen

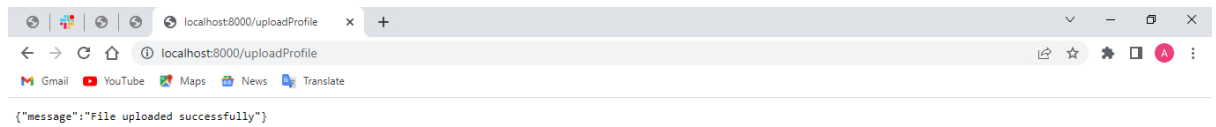


File Upload

Upload Profile Picture: Anjai_Schol...svi2022.pdf



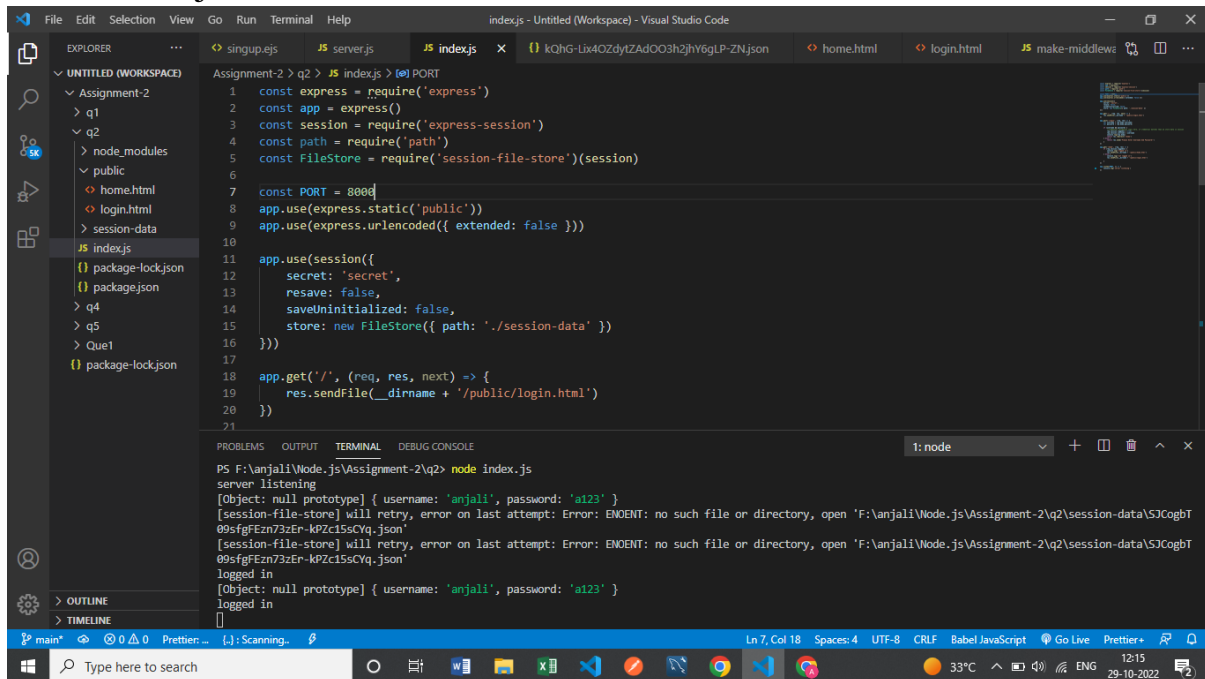




Que-2

➔ Code:

Index.js



The screenshot shows a Visual Studio Code editor with a workspace named 'index.js - Untitled (Workspace)'. The Explorer sidebar on the left shows a project structure with folders 'Assignment-2', 'q1', 'q2', 'node_modules', 'public', and 'session-data'. The 'index.js' file is selected in the Explorer. The main editor area displays the code for 'index.js'. The code defines an Express application, sets a session store using 'FileStore', and serves a static file 'login.html' from the 'public' directory. The terminal at the bottom shows the command 'node index.js' being executed, and the output indicates that the server is listening on port 8000 and that a login attempt was successful.

```
1 const express = require('express')
2 const app = express()
3 const session = require('express-session')
4 const path = require('path')
5 const FileStore = require('session-file-store')(session)
6
7 const PORT = 8000
8 app.use(express.static('public'))
9 app.use(express.urlencoded({ extended: false }))
10
11 app.use(session({
12   secret: 'secret',
13   resave: false,
14   saveUninitialized: false,
15   store: new FileStore({ path: './session-data' })
16 }))
17
18 app.get('/', (req, res, next) => {
19   res.sendFile(__dirname + '/public/login.html')
20 })
21
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

1: node

PS F:\anjalil\Node.js\Assignment-2\q2> node index.js

server listening

[Object: null prototype] { username: 'anjali', password: 'a123' }

[session-file-store] will retry, error on last attempt: Error: ENOENT: no such file or directory, open 'F:\anjalil\Node.js\Assignment-2\q2\session-data\5JCogbT09sfgFEzn73zEr-kPZc15sCvQ.json'

[session-file-store] will retry, error on last attempt: Error: ENOENT: no such file or directory, open 'F:\anjalil\Node.js\Assignment-2\q2\session-data\5JCogbT09sfgFEzn73zEr-kPZc15sCvQ.json'

logged in

[Object: null prototype] { username: 'anjali', password: 'a123' }

logged in

```
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 = 8000
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;

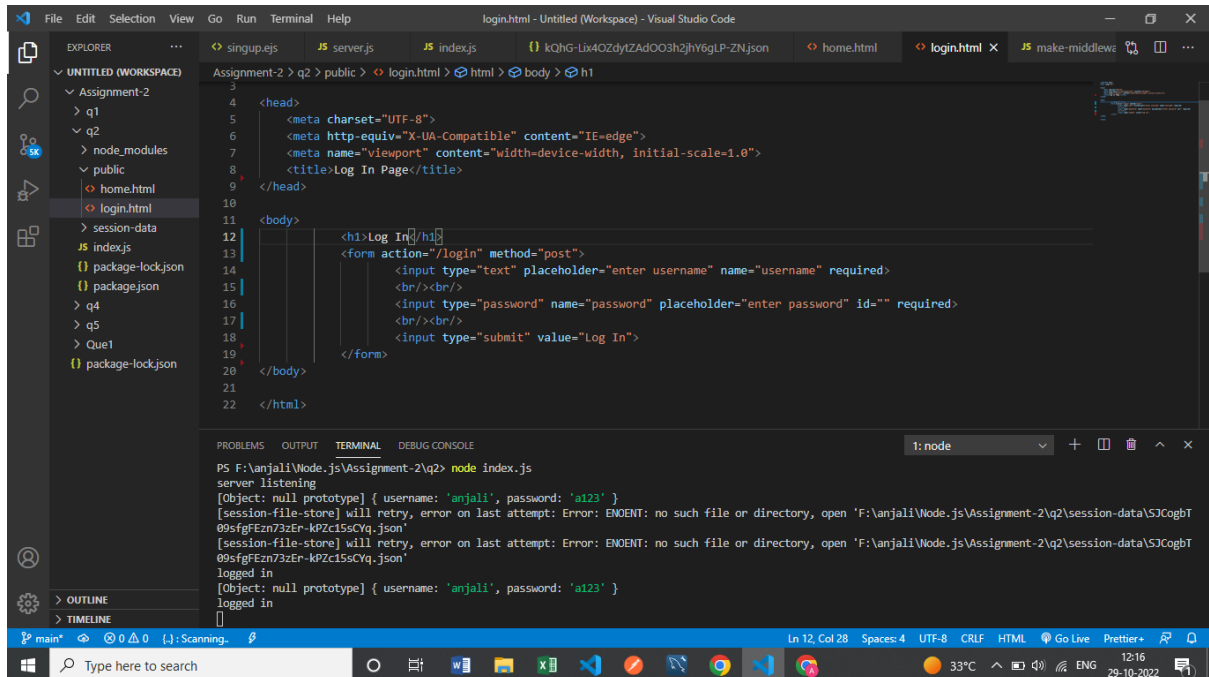
  if (username && password) {
    //credential validation logic here, if credential matches than we store data i
n session
    req.session.loggedIn = true;
    req.session.username = username
    console.log(req.body)
    return res.redirect('/home')
  } else {
    return res.send('Please Enter Username 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`)
})

```


Login.html



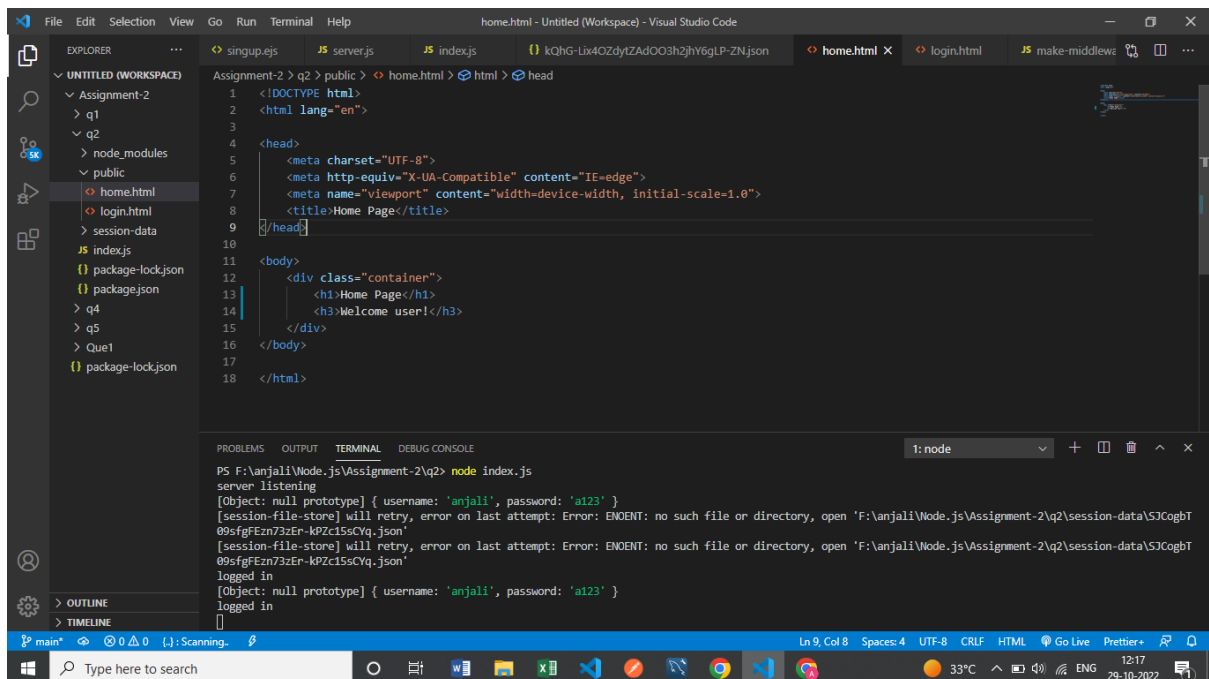
The screenshot shows the Visual Studio Code editor with the 'login.html' file open. The file content is as follows:

```
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Log In Page</title>
9 </head>
10
11 <body>
12   <h1>Log In</h1>
13   <form action="/login" method="post">
14     <input type="text" placeholder="enter username" name="username" required>
15     <br/><br/>
16     <input type="password" name="password" placeholder="enter password" id="" required>
17     <br/><br/>
18     <input type="submit" value="Log In">
19   </form>
20 </body>
21
22 </html>
```

The terminal output shows the following:

```
PS F:\anjali\Node.js\Assignment-2\q2> node index.js
server listening
[Object: null prototype] { username: 'anjali', password: 'a123' }
[session-file-store] will retry, error on last attempt: Error: ENOENT: no such file or directory, open 'F:\anjali\Node.js\Assignment-2\q2\session-data\SJCogbT09sfgFEzn73zEr-kPZc15sCYq.json'
[session-file-store] will retry, error on last attempt: Error: ENOENT: no such file or directory, open 'F:\anjali\Node.js\Assignment-2\q2\session-data\SJCogbT09sfgFEzn73zEr-kPZc15sCYq.json'
logged in
[Object: null prototype] { username: 'anjali', password: 'a123' }
logged in
```

Home.html



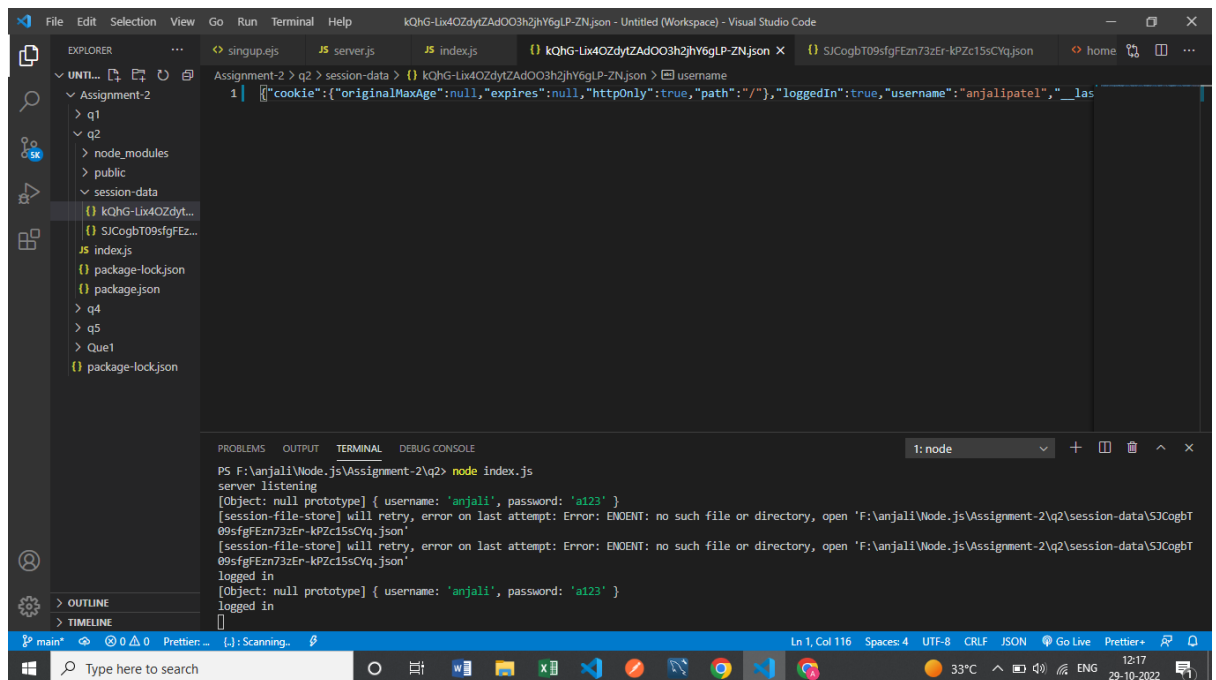
The screenshot shows the Visual Studio Code editor with the 'home.html' file open. The file content is as follows:

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta http-equiv="X-UA-Compatible" content="IE=edge">
7   <meta name="viewport" content="width=device-width, initial-scale=1.0">
8   <title>Home Page</title>
9 </head>
10
11 <body>
12   <div class="container">
13     <h1>Home Page</h1>
14     <h3>Welcome user!</h3>
15   </div>
16 </body>
17
18 </html>
```

The terminal output shows the following:

```
PS F:\anjali\Node.js\Assignment-2\q2> node index.js
server listening
[Object: null prototype] { username: 'anjali', password: 'a123' }
[session-file-store] will retry, error on last attempt: Error: ENOENT: no such file or directory, open 'F:\anjali\Node.js\Assignment-2\q2\session-data\SJCogbT09sfgFEzn73zEr-kPZc15sCYq.json'
[session-file-store] will retry, error on last attempt: Error: ENOENT: no such file or directory, open 'F:\anjali\Node.js\Assignment-2\q2\session-data\SJCogbT09sfgFEzn73zEr-kPZc15sCYq.json'
logged in
[Object: null prototype] { username: 'anjali', password: 'a123' }
logged in
```

➔ Session Data:

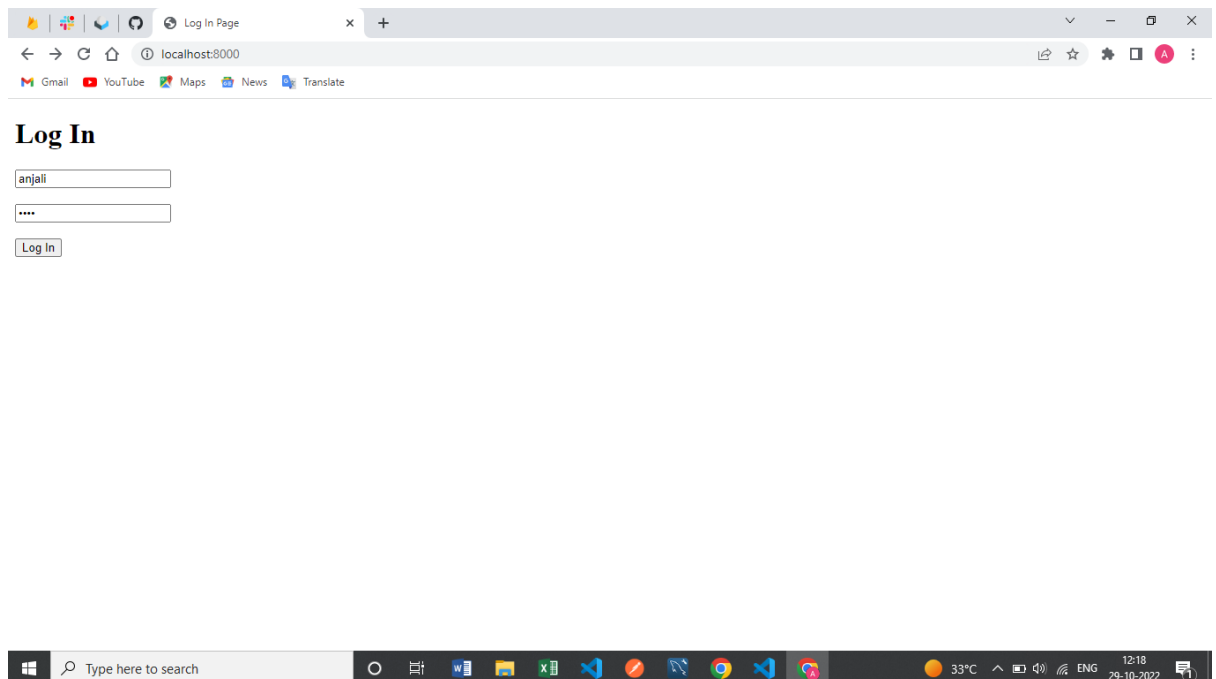


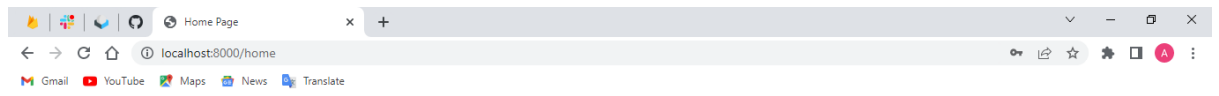
The screenshot shows the Visual Studio Code interface. The Explorer panel on the left displays the project structure, including a 'session-data' directory. The main editor shows a file named 'KQhG-Lix4OZdytZAdOO3h2jY6gLP-ZNjson' with the following content:

```
Assignment-2 > q2 > session-data > {} KQhG-Lix4OZdytZAdOO3h2jY6gLP-ZNjson > username
1 | [{"cookie":{"originalMaxAge":null,"expires":null,"httpOnly":true,"path":"/"},"loggedIn":true,"username":"anjalipatel","__las
```

The terminal at the bottom shows the output of running 'node index.js' in the 'session-data' directory. The output indicates that the server is listening and that the session file store is retrying due to a file not found error. The session data is logged in, and the user 'anjalipatel' is logged in.

➔ Output:





Home Page

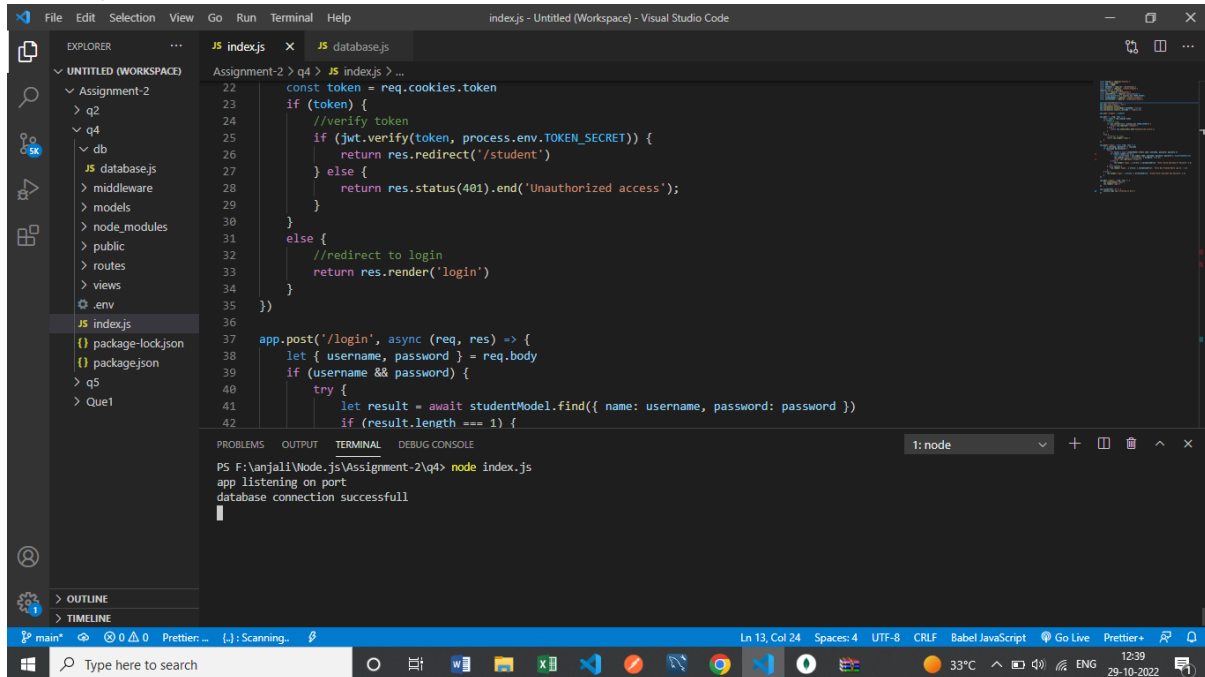
Welcome user!



Que-4

➔ Code:

Index.js



The screenshot shows the Visual Studio Code editor with a workspace named 'index.js - Untitled (Workspace)'. The Explorer sidebar on the left shows a project structure with folders like 'Assignment-2', 'q2', 'q4', 'db', 'database.js', 'middleware', 'models', 'node_modules', 'public', 'routes', 'views', and files like '.env', 'index.js', 'package-lock.json', and 'package.json'. The main editor area displays the 'index.js' file with the following code:

```
22 const token = req.cookies.token
23 if (token) {
24   //verify token
25   if (jwt.verify(token, process.env.TOKEN_SECRET)) {
26     return res.redirect('/student')
27   } else {
28     return res.status(401).end('Unauthorized access');
29   }
30 }
31 else {
32   //redirect to login
33   return res.render('login')
34 }
35 })
36
37 app.post('/login', async (req, res) => {
38   let { username, password } = req.body
39   if (username && password) {
40     try {
41       let result = await studentModel.find({ name: username, password: password })
42       if (result.length === 1) {
```

The TERMINAL panel at the bottom shows the command 'node index.js' being executed, with the output: 'app listening on port' and 'database connection successful'.

```
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) {
        const accessToken = jwt.sign({ name: username, password: password }, accessTokenSecret)
        res.cookie('token', accessToken, { httpOnly: true });
        return res.redirect('/student')
      } 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) => {

```

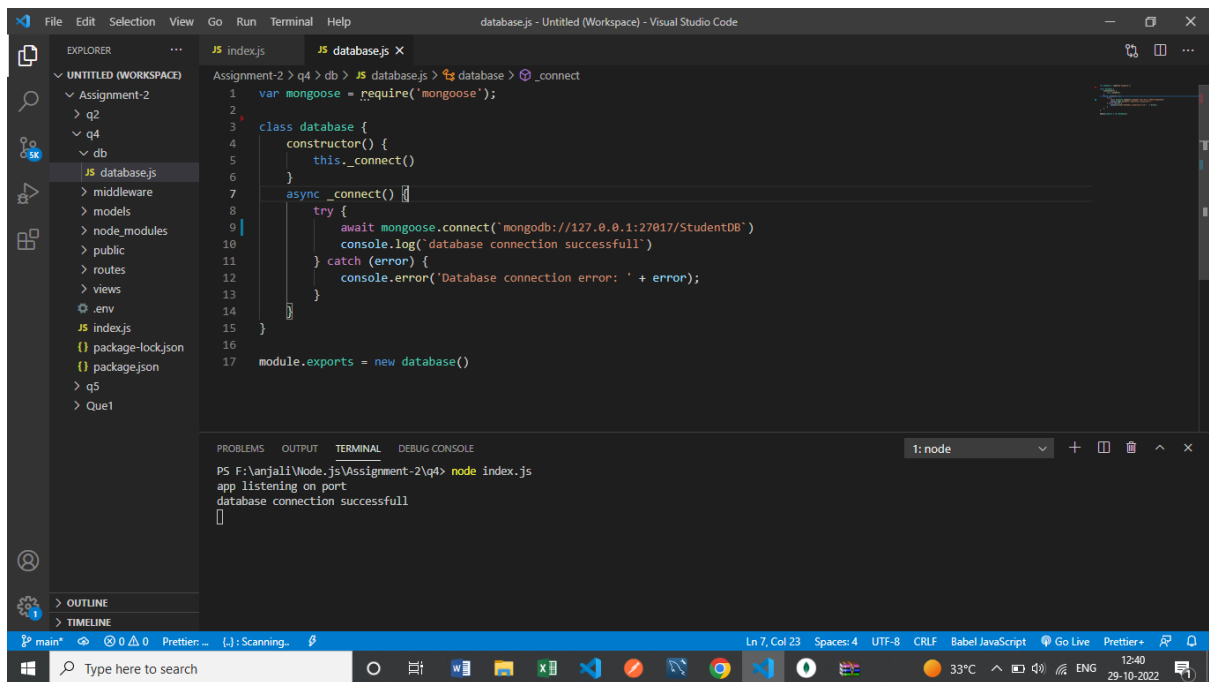
```

    res.clearCookie('token')
    res.render('login')
  })

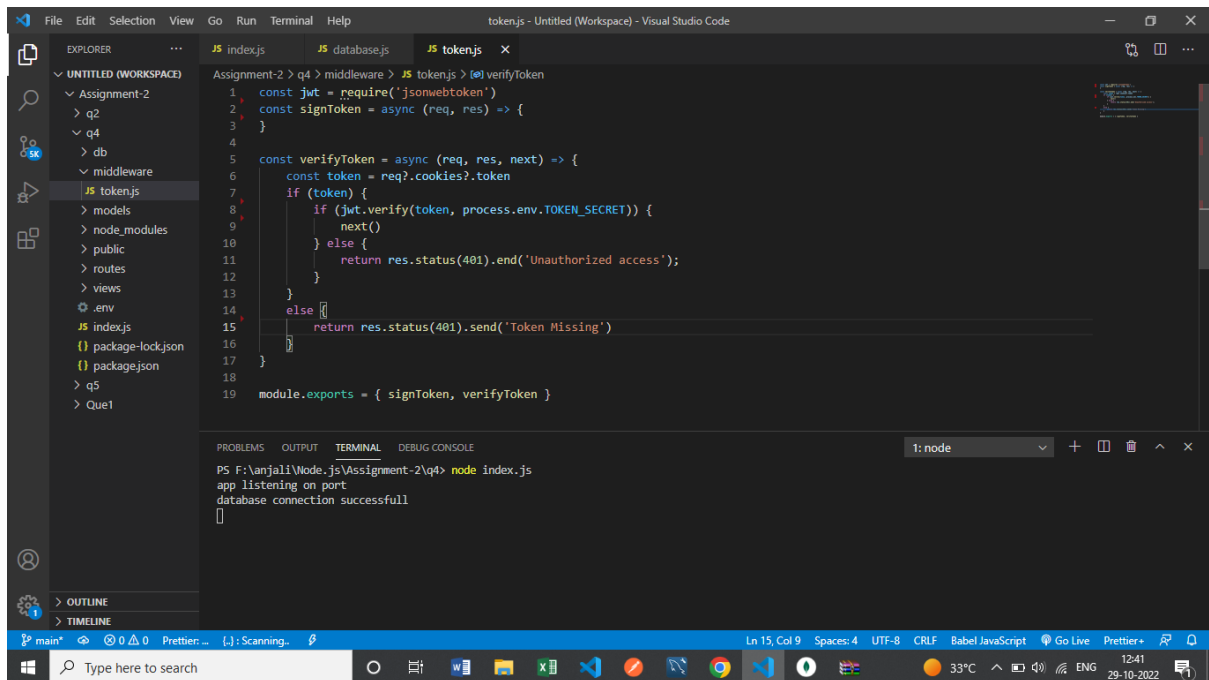
  app.listen(PORT, () => {
    console.log(`app listening on port`)
  })

```

Database.js



Token.js



The screenshot shows a Visual Studio Code editor with a workspace named 'tokenjs'. The Explorer sidebar on the left shows a project structure with folders like 'Assignment-2', 'q2', 'q4', 'db', 'middleware', and 'models'. The 'models' folder is expanded, showing 'tokenjs.js'. The main editor displays the code for 'tokenjs.js', which includes a 'verifyToken' function that checks a JWT token and returns a 401 status if it's missing or unauthorized. The terminal at the bottom shows the command 'node index.js' being executed, with output indicating the app is listening on port and the database connection is successful.

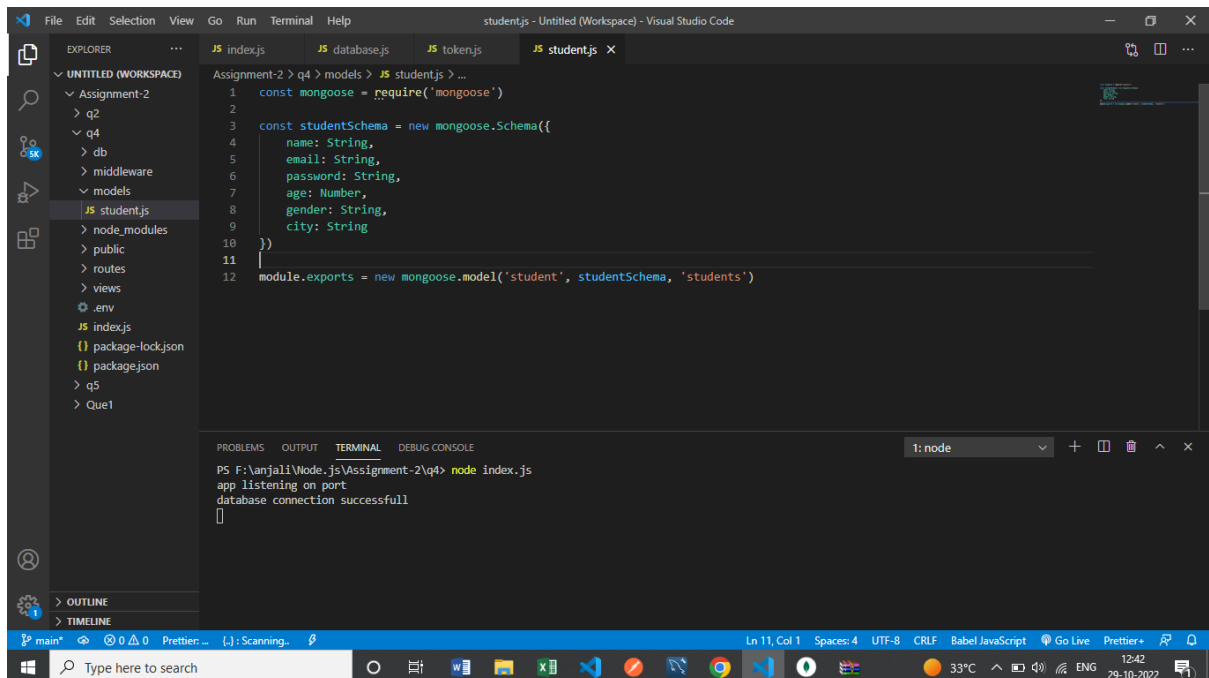
```
1 const jwt = require('jsonwebtoken')
2 const signToken = async (req, res) => {
3 }
4
5 const verifyToken = async (req, res, next) => {
6   const token = req?.cookies?.token
7   if (token) {
8     if (jwt.verify(token, process.env.TOKEN_SECRET)) {
9       next()
10    } else {
11      return res.status(401).end('Unauthorized access');
12    }
13  } else {
14    return res.status(401).send('Token Missing')
15  }
16 }
17
18 module.exports = { signToken, verifyToken }
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

1: node

PS F:\anjalil\Node.js\Assignment-2\q4> node index.js
app listening on port
database connection successful

Student.js (Model)



The screenshot shows a Visual Studio Code editor with a workspace named 'studentjs'. The Explorer sidebar on the left shows a project structure with folders like 'Assignment-2', 'q2', 'q4', 'db', 'middleware', and 'models'. The 'models' folder is expanded, showing 'student.js'. The main editor displays the code for 'student.js', which defines a Mongoose schema for a student with fields like name, email, password, age, gender, and city. The terminal at the bottom shows the command 'node index.js' being executed, with output indicating the app is listening on port and the database connection is successful.

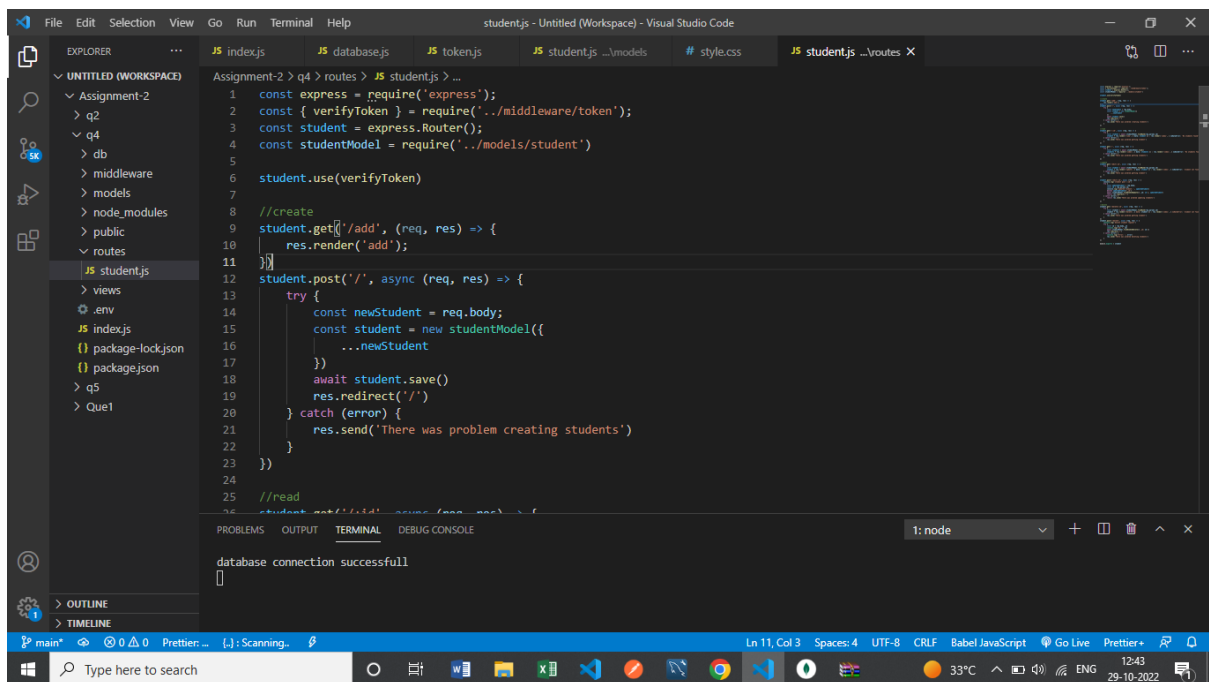
```
1 const mongoose = require('mongoose')
2
3 const studentSchema = new mongoose.Schema({
4   name: String,
5   email: String,
6   password: String,
7   age: Number,
8   gender: String,
9   city: String
10 })
11
12 module.exports = new mongoose.model('student', studentSchema, 'students')
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

1: node

PS F:\anjalil\Node.js\Assignment-2\q4> node index.js
app listening on port
database connection successful

Student.js (Router)



```
1 const express = require('express');
2 const { verifyToken } = require('../middleware/token');
3 const student = express.Router();
4 const studentModel = require('../models/student')
5
6 student.use(verifyToken)
7
8 //create
9 student.get('/add', (req, res) => {
10   res.render('add');
11 })
12 student.post('/', async (req, res) => {
13   try {
14     const newStudent = req.body;
15     const student = new studentModel({
16       ...newStudent
17     })
18     await student.save()
19     res.redirect('/')
20   } catch (error) {
21     res.send('There was problem creating students')
22   }
23 })
24
25 //read
26 student.get('/:id', async (req, res) => {
```

database connection successfull

```
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('/')
  }
})

```

```

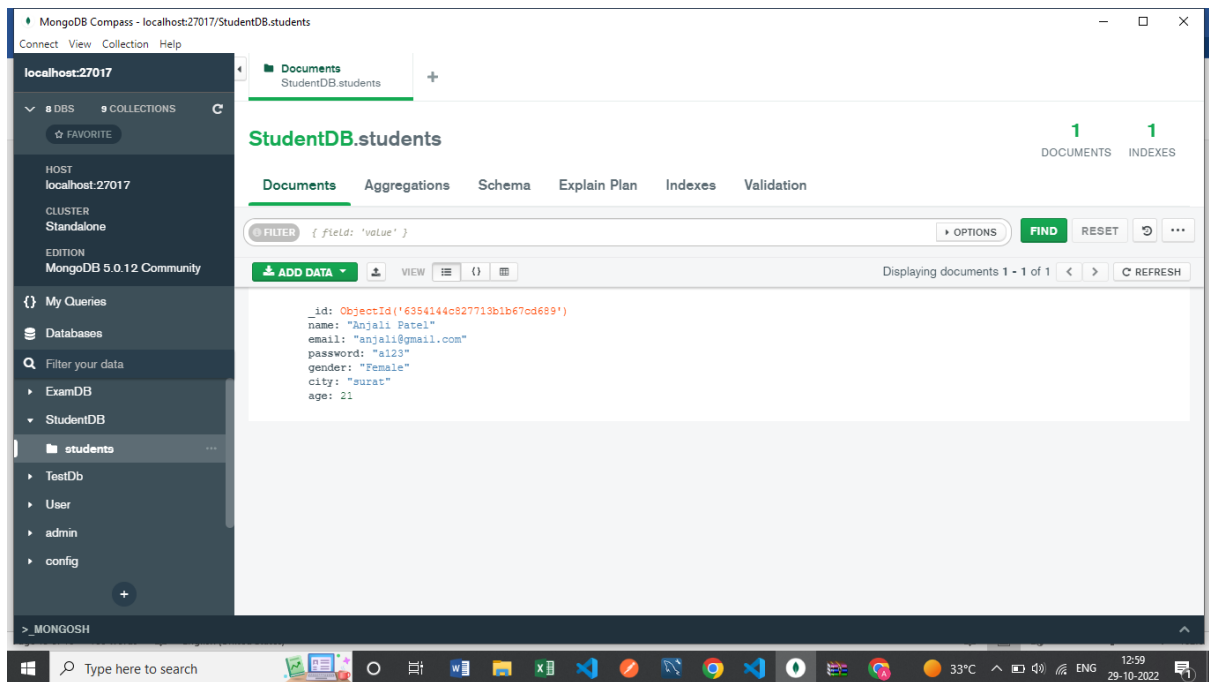
    } catch (error) {
      return res.send('There was problem updating students')
    }
  })

//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', { noData:
Error: 'student not found' })
  } catch (error) {
    res.send('There was problem getting student')
  }
})

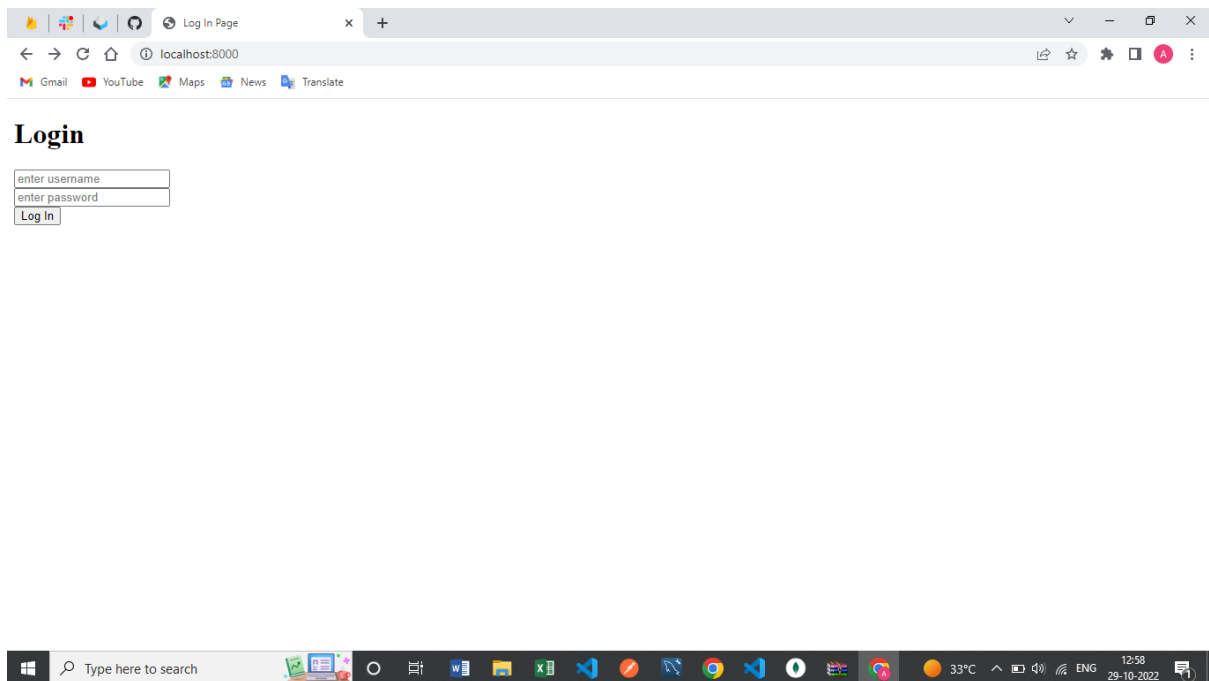
student.post('/delete', async (req, res) => {
  console.log('delete student request')
  try {
    const id = req.body._id
    console.log("id: ", id)
    await studentModel.findOneAndDelete({ _id: id })
    res.redirect('/')
  } catch (error) {
    console.log('Error: ', error)
    res.send('There was problem deleting students')
  }
})

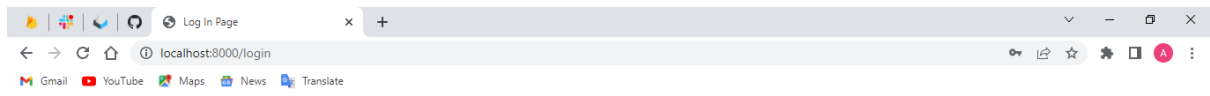
module.exports = student

```



➔ Output:



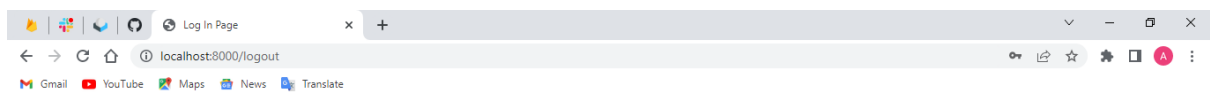


Login

Enter Valid Username Or Password

anjali

Log In

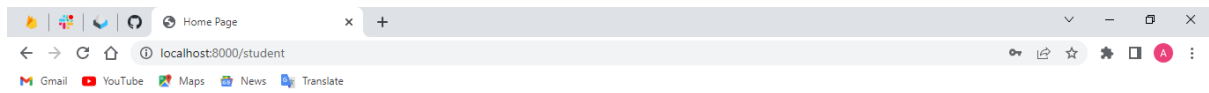


Login

Anjali Patel

Log In

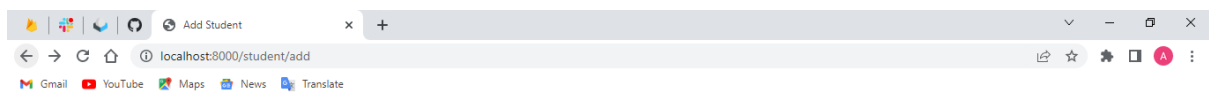




name	email	age	city	gender	Actions
Anjali Patel	anjali@gmail.com	21	surat	Female	edit delete view

[Add Student](#)

[Log Out](#)

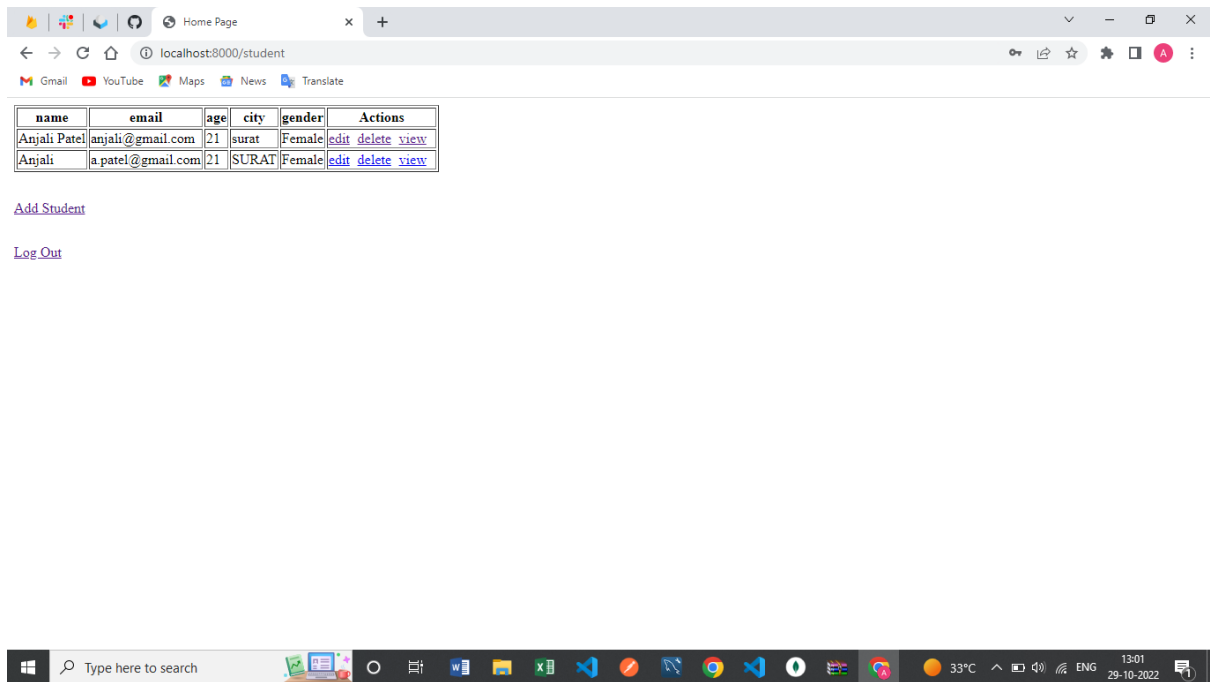


Add Student

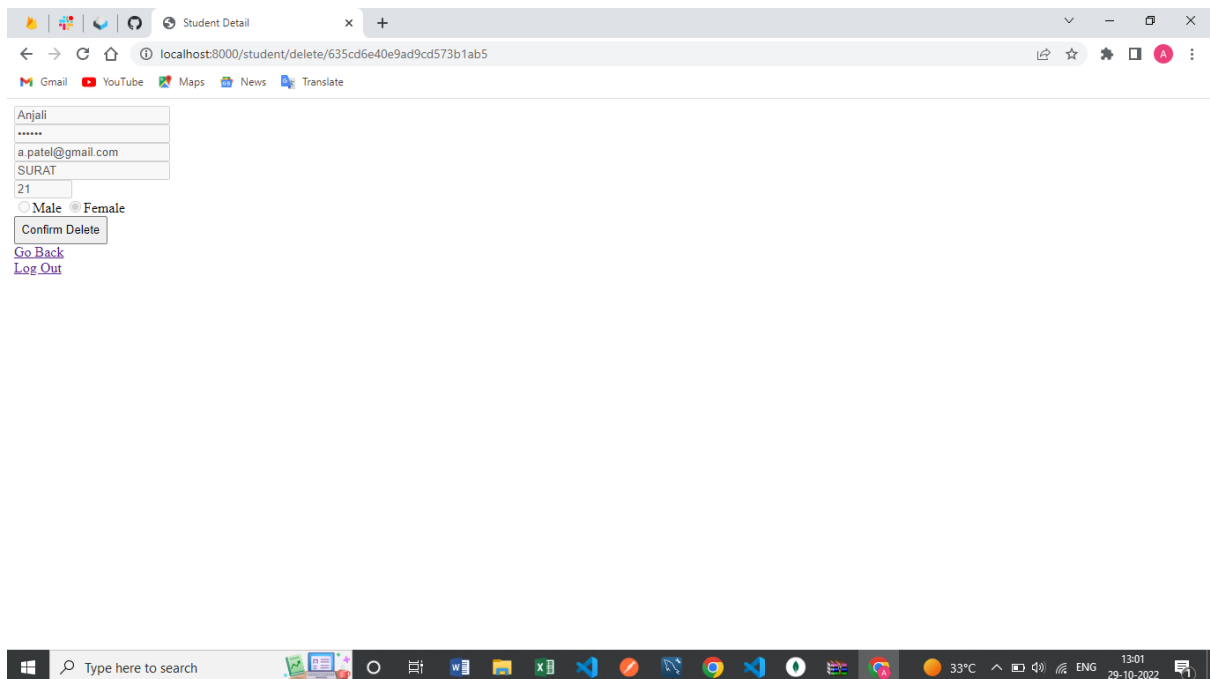
☐ Male ☒ Female

[Log Out](#)

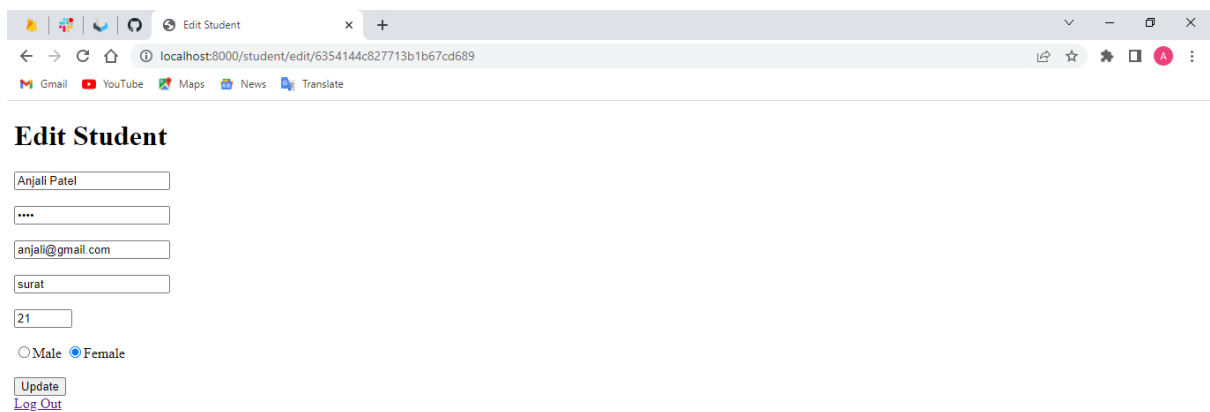
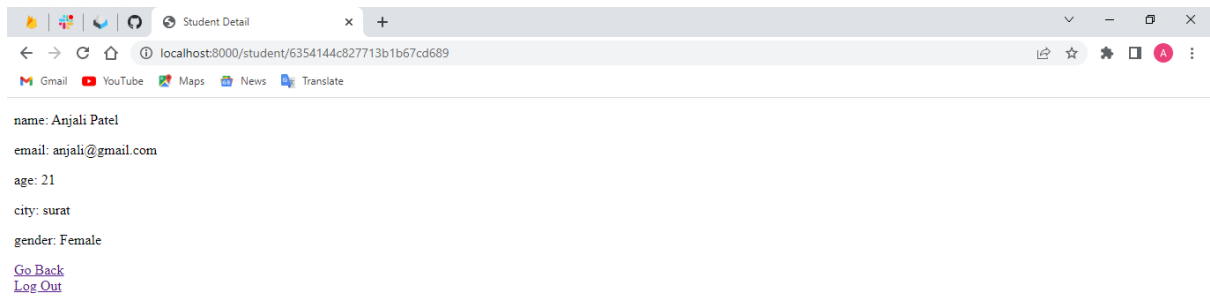




Delete student data



Display Student data



Home Page

localhost:8000/student

Gmail YouTube Maps News Translate

name	email	age	city	gender	Actions
Anjali Patel	anjali Patel@gmail.com	20	surat	Female	edit delete view

[Add Student](#)

[Log Out](#)

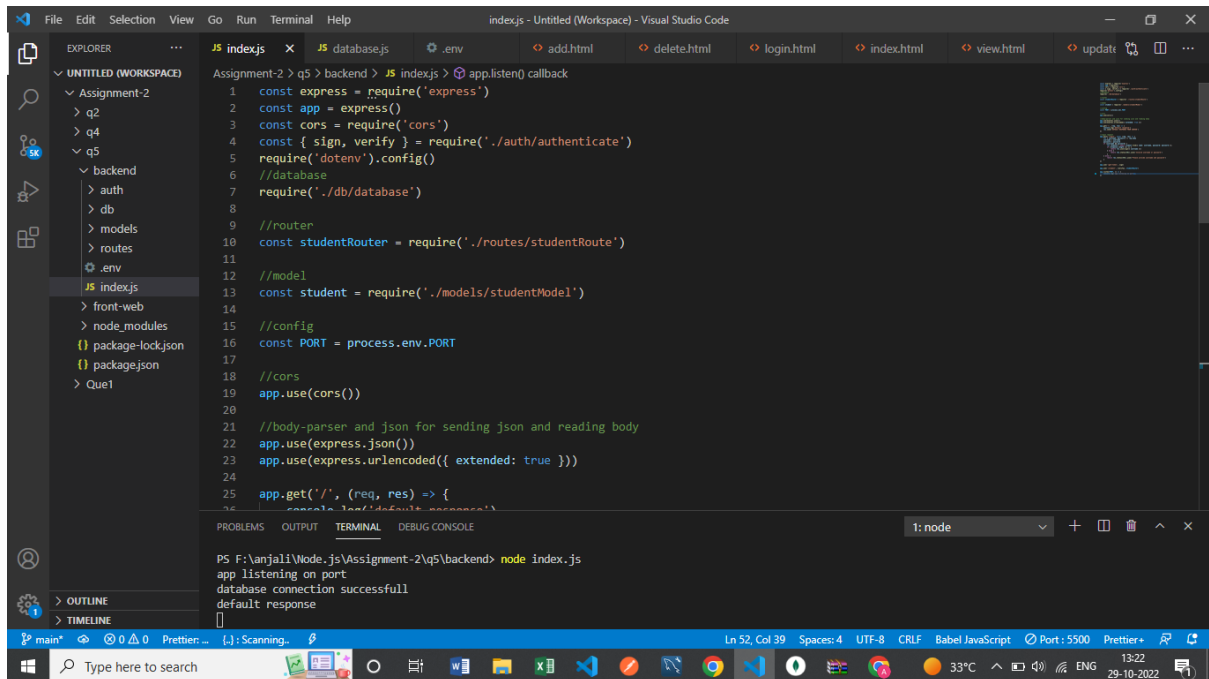
Type here to search

33°C 13:06 29-10-2022

Que-5

➔ Code:

Index.js



The screenshot shows the Visual Studio Code interface with the 'indexjs - Untitled (Workspace)' window. The Explorer sidebar on the left shows the project structure, including 'Assignment-2' and its subfolders 'q2', 'q4', 'q5', 'backend', 'auth', 'db', 'models', 'routes', '.env', 'indexjs', 'front-web', 'node_modules', 'package-lock.json', 'package.json', and 'Que1'. The 'indexjs' folder is selected, and the 'index.js' file is open in the editor. The code in the editor is as follows:

```
1 const express = require('express')
2 const app = express()
3 const cors = require('cors')
4 const { sign, verify } = require('./auth/authenticate')
5 require('dotenv').config()
6 //database
7 require('./db/database')
8
9 //router
10 const studentRouter = require('./routes/studentRoute')
11
12 //model
13 const student = require('./models/studentModel')
14
15 //config
16 const PORT = process.env.PORT
17
18 //cors
19 app.use(cors())
20
21 //body-parser and json for sending json and reading body
22 app.use(express.json())
23 app.use(express.urlencoded({ extended: true }))
24
25 app.get('/', (req, res) => {
26   //default response
27 })
```

The terminal at the bottom shows the command 'node index.js' being executed, and the output is:

```
PS F:\anjal\Mode.js\Assignment-2\q5\backend> node index.js
app listening on port
database connection successfull
default response
```

```
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')
  }
})

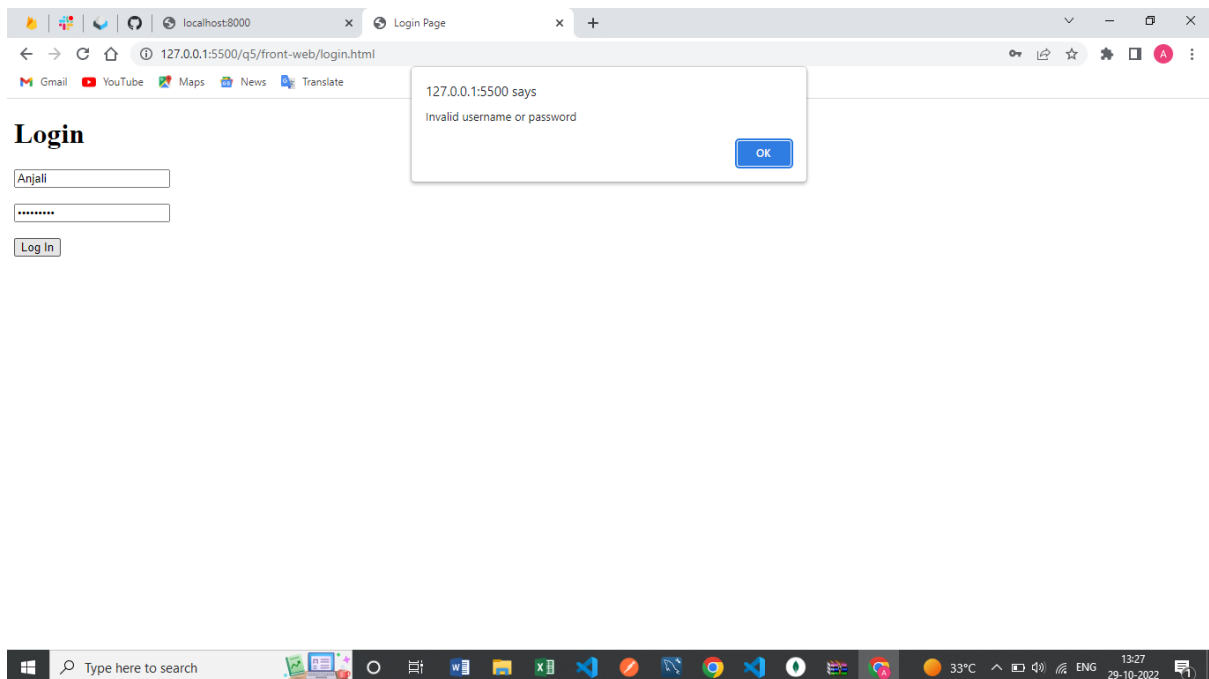
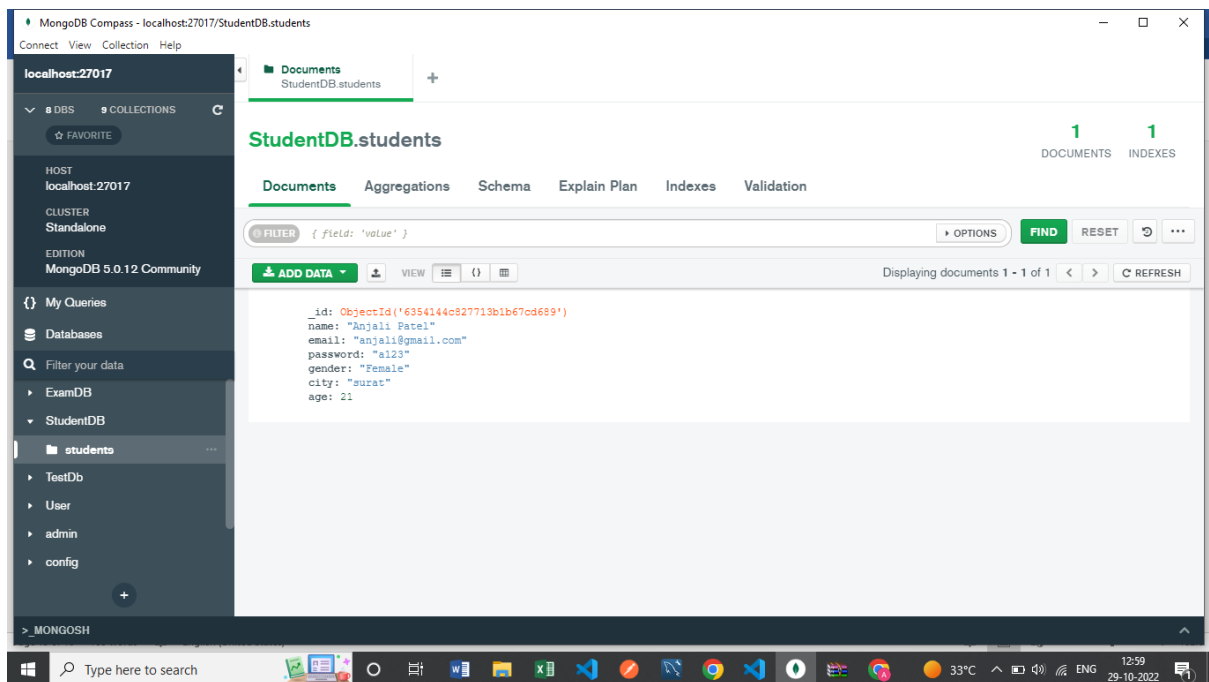
app.use('/get-token', sign)

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

app.listen(PORT, () => {
  console.log(`app listening on port`);
})

```

➔ Output:



localhost8000 x Home Page x +

127.0.0.1:5500/q5/front-web/index.html

Gmail YouTube Maps News Translate

name	email	age	city	gender	Actions
Anjali Patel	anjali Patel@gmail.com	20	surat	Female	edit delete view

[Add Student](#)
[Log Out](#)

Type here to search

33°C 13:29 29-10-2022

localhost8000 x Add Student x +

127.0.0.1:5500/q5/front-web/add.html

Gmail YouTube Maps News Translate

Add Student

enter your name
enter password
enter your email
enter your city
Age he
☒ Male ☐ Female
Add
[Log Out](#)

Type here to search

33°C 13:29 29-10-2022

localhost8000 Edit Student +

127.0.0.1:5500/q5/front-web/update.html?id=6354144c827713b1b67cd689

Gmail YouTube Maps News Translate

Edit Student

Anjali Patel
....
anjaliipatel@gmail.com
surat
20
<input type="radio"/> Male <input checked="" type="radio"/> Female
Update

[Log Out](#)



localhost8000 Delete Student +

127.0.0.1:5500/q5/front-web/delete.html?id=6354144c827713b1b67cd689

Gmail YouTube Maps News Translate

name: Anjali Patel
email: anjalipatel@gmail.com
age: 20
city: surat
gender: Female

[Go Back](#)
[Log Out](#)



