

IT314: Software Engineering

Lab Session 8 - Unit Testing

Student Information System G23

→ Here we are Testing the login module of Student Information System And we are using **mocha** and **chai** Package for Testing in node.js.

So here we are test Only for Student Login Because For Faculty and Admin login details is same so Test Case For Student are Same for admin as well as Faculty

- **Test Cases For Student Login:**

- 1) Enter a Valid Credentials
- 2) Enter a Email which is not stored in our Database
- 3) Null Value in Email
- 4) Null Value in Password
- 5) Enter a Email which is store in our database with wrong password
- 6) Enter a Invalid Credentials in all fields
- 7) Enter a Invalid Email Format.

- **Here is Login Code which we are Testing:**

```
app.get('/studentLogin', checkNotAuthenticatedStudent, (req, res) => {
  res.status(200).render('studentLogin.ejs')
}))

app.post('/studentLogin', function(req, res, next) {
  passportStudent.authenticate('student', function(err, user, info) {
    if (err) {
      return next(err);
    }
    if (!user) {
      // Authentication failed, send a 401 status
      return res.status(401).redirect('/studentLogin');
    }
    // Authentication successful, send a 200 status and redirect to
    '/studentHome'
    req.login(user, function(err) {
      if (err) {
```

```

        return next(err);
    }
    return res.status(200).redirect('/studentHome');
  });
})(req, res, next);
});

```

- **Testing in Terminal:**

→ So Here is Testing Code for when **user send a get Request to Student Login Page** so if User successfully redirect to that page so this test case will pass and **second test case is for All Valid Credentials** When User enter valid Credentials this test case will pass.

```

const chai = require("chai");
const expect = chai.expect;
const chaiHttp = require("chai-http");
const server = require("../server"); // Your backend server file

chai.use(chaiHttp);

describe("http://localhost:3000", () => {
  describe("GET /studentLogin", () => {
    it("should return a 200 status code", async () => {
      const res = await chai.request(server).get("/studentLogin");
      expect(res).to.have.status(200);
    });
  });

  describe("GET /studentHome", () => {
    it("should return a 200 status code", async () => {
      const res = await chai
        .request(server)
        .post("/studentLogin")
        .send({ email: "nagrechadevarsh@gmail.com", password: "a" });
      expect(res).to.have.status(200);
    });
  });
});

```

→ **Output of Above Test Cases.**

```
PS D:\DAIICT\Sem 6\Software Engineering IT314\Project\Student_Information_System> npm test

> student-information-system@1.0.0 test
> mocha

http://localhost:3000
  GET /studentLogin
    ✓ should return a 200 status code (61ms)
  GET /studentHome
(node:6640) [DEP0066] DeprecationWarning: OutgoingMessage.prototype._headers is deprecated
(Use `node --trace-deprecation ...` to show where the warning was created)
    ✓ should return a 200 status code

2 passing (171ms)
```

→ This is the test code for **When User Enters Wrong Password** and **When All the Credentials is null** And **When Entered Email id is not registered** And when **Email is Wrong** or **password is wrong** and **All the credentials are invalid**.

```
const chai = require("chai");
const expect = chai.expect;
const chaiHttp = require("chai-http");
const server = require("../server"); // Your backend server file

chai.use(chaiHttp);

describe("http://localhost:3000", () => {

  describe("GET /studentHome", () => {

    it("should return a 401 status code", async () => {
      const res = await chai
        .request(server)
        .post("/studentLogin")
        .send({ email: "nagrechadevarsh@gmail.com", password: "1234" });
      expect(res).to.have.status(401);
    });
  });
});
```

```

it("should return a 401 status code", async () => {
  const res = await chai
    .request(server)
    .post("/studentLogin")
    .send({ email: "nagrechrrh@gmail.com", password: "a" });
  expect(res).to.have.status(401);
});

it("should return a 401 status code", async () => {
  const res = await chai
    .request(server)
    .post("/studentLogin")
    .send({ email: "", password: "" });
  expect(res).to.have.status(401);
});
});
});

```

→ This is Output for Above test Cases.

```

PS D:\DAIICT\Sem 6\Software Engineering IT314\Project\Student_Information_System> npm test

> student-information-system@1.0.0 test
> mocha

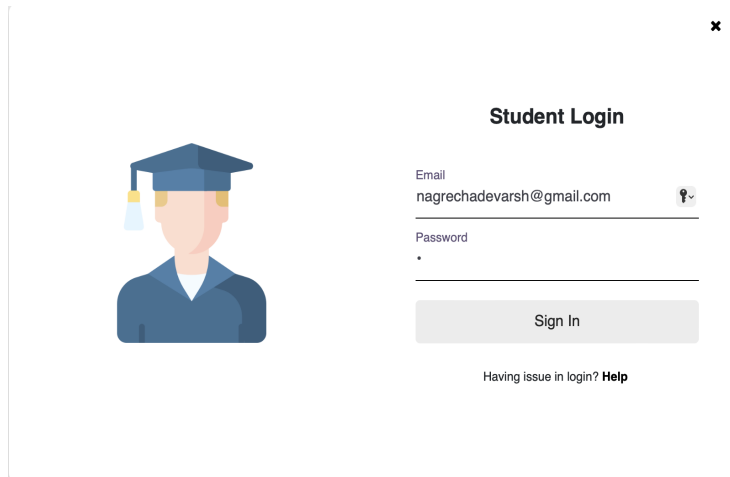
http://localhost:3000
GET /studentHome
(node:23988) [DEP0066] DeprecationWarning: OutgoingMessage.prototype._headers is deprecated
(Use `node --trace-deprecation ...` to show where the warning was created)
  ✓ should return a 401 status code (64ms)
  ✓ should return a 401 status code
  ✓ should return a 401 status code

  3 passing (202ms)

```

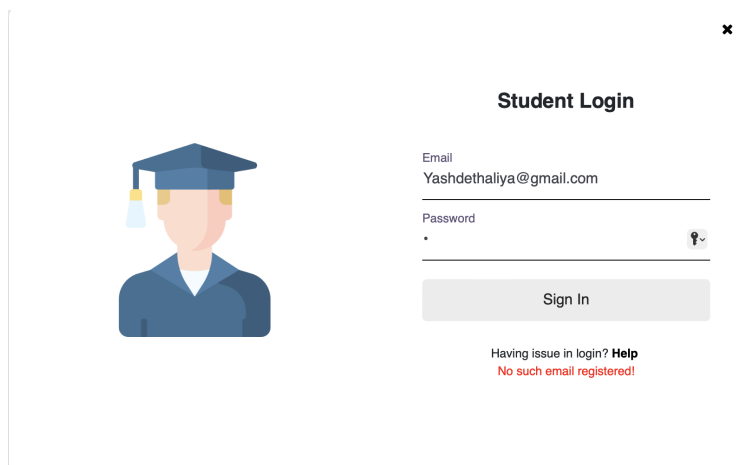
- **GUI Testing:**

- 1) With All Valid Credentials



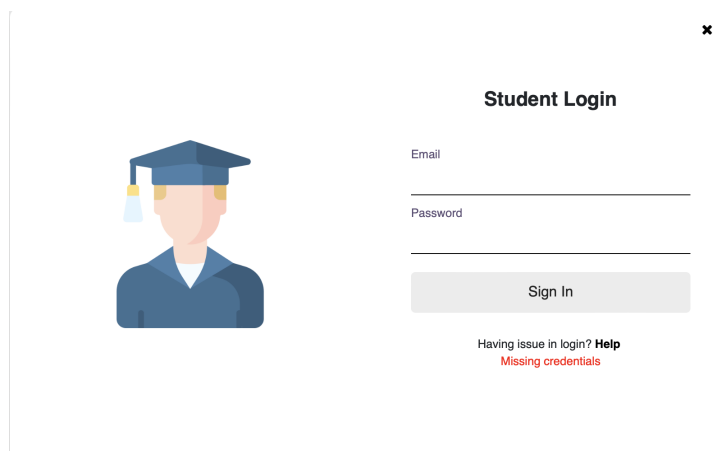
A screenshot of a web browser window displaying a 'Student Login' form. On the left is a blue graduation cap icon. The form has two input fields: 'Email' with the value 'nagrechadevarsh@gmail.com' and 'Password' with a single dot. A 'Sign In' button is below the fields. At the bottom, it says 'Having issue in login? [Help](#)'. The browser window has a close button (x) in the top right corner.

- 2) Enter a Email which is not stored in our Database



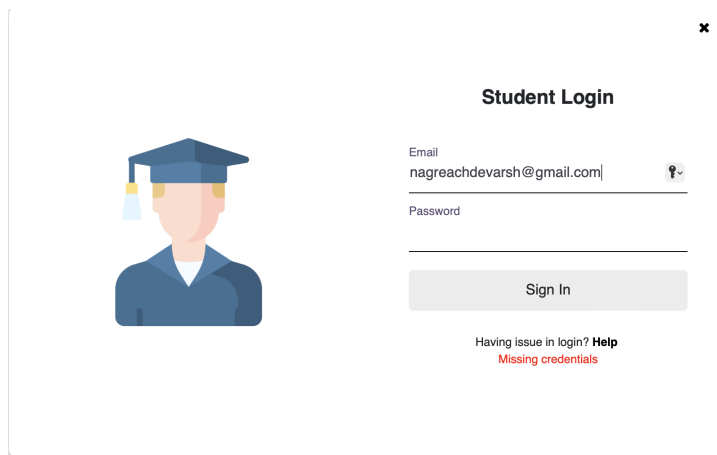
A screenshot of the 'Student Login' form with the email 'Yashdethaliya@gmail.com' entered. Below the 'Sign In' button, a red error message reads: 'Having issue in login? [Help](#)
No such email registered!'. The browser window has a close button (x) in the top right corner.

- 3) Null Value in Email



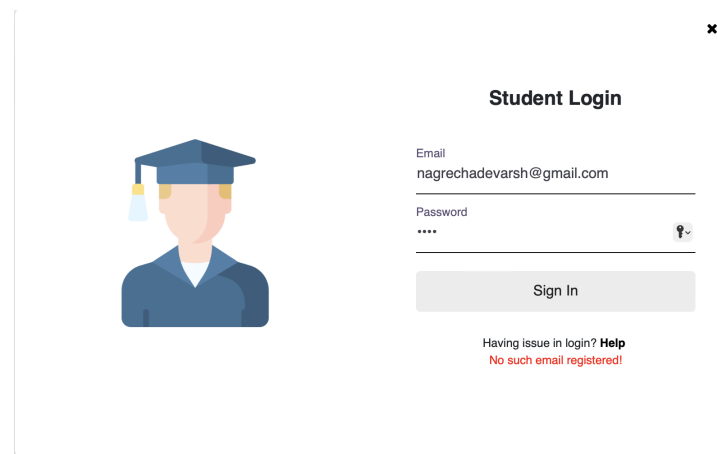
A screenshot of the 'Student Login' form with the email field empty. Below the 'Sign In' button, a red error message reads: 'Having issue in login? [Help](#)
Missing credentials'. The browser window has a close button (x) in the top right corner.

4) Null Value in Password



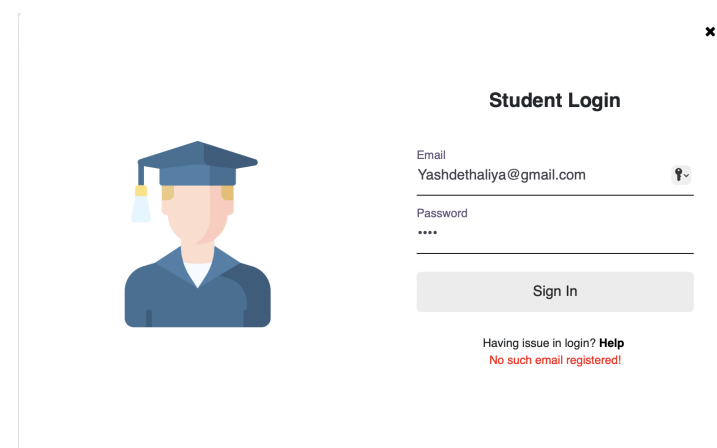
The screenshot shows a 'Student Login' form. On the left is an illustration of a student in a blue graduation cap and gown. The form has two input fields: 'Email' with the value 'nagrechdevvarsh@gmail.com' and 'Password' which is empty. Below the fields is a 'Sign In' button. At the bottom, a message reads: 'Having issue in login? [Help](#) Missing credentials'.

5) Enter a Email which is store in our database with wrong password




The screenshot shows the 'Student Login' form. The 'Email' field contains 'nagrechdevvarsh@gmail.com' and the 'Password' field contains four dots. The 'Sign In' button is visible. The error message at the bottom reads: 'Having issue in login? [Help](#) No such email registered!'.

6) Enter a Invalid Credentials in all fields



The screenshot shows the 'Student Login' form. The 'Email' field contains 'Yashdethaliya@gmail.com' and the 'Password' field contains four dots. The 'Sign In' button is visible. The error message at the bottom reads: 'Having issue in login? [Help](#) No such email registered!'.

7) Enter a Invalid Email Format.



Student Login

Email
Yashdethaliyagmail.com

Password
.....

Sign In

Having issue in login? [Help](#)
No such email registered!