IT314: Software Engineering
Lab 8

Group 17: A platform for creating and managing online polls and surveys

Mentor: Vaishnavi

Members:
202001134 Prajapati Hardik Pravinbhai
202001145 Prajapati Harshkumar Mukundbhai
202001143 Vadi Harsh Rajubhai
202001141 Kapasi Harshil Ileshkumar
202001208 Dishal Vala
202001125 Prashasti Srivastava
202001123 Patel Het Bhupendrabhai
202001166 Master Jinkal Hirenkumar
202001144 Sangani Hemil Sharadbhai

- Take a module that you have implemented in any programming language. (Java, Python, C, C++, etc.)

  We will be considering the Sign-up and Sign-in modules for the submission. The language used is JavaScript.

- Write the test cases.

Since in our project we have implemented signup module and we have checked that module among various testcases.

One of the first requirement for that module is to provide all the credentials and if any of them is missing will return “400” status code with “missing credentials”.

There are 40 diffrent testcases for this particular case and it is unnecessary to do so, If we even miss only one field then also this case should triggered.so only four testcase should cover almost case.

```
// when any credentials from email, password, name, username are missing
describe("when any credentials are missing", () => {
    it("should return 400", async () => {
        const bodyData = [
        { email: "chhotu123@gmail.com", password: "chhotu", name: "Chhotu"},
        { password: "chhotu", name: "Chhotu", username: "chhotu" },
        { email: "chhotu123@gmail.com", password: "chhotu", username: "chhotu"},
        { email: "chhotu123@gmail.com", username: "chhotu", name: "Chhotu"}
        ]
        for( const data of bodyData){
            const res = await supertest(app).post("/api/auth/register").send(data);
            expect(res.statusCode).toEqual(400);
        }
    });
});
```

This case triggered when the email is in wrong format. This case required only one testcase.

```
// invalid email format
describe("when email is invalid", () => {
    test("should return 400", async () => {
        const res = await supertest(app).post("/api/auth/register").send({
            email: "chhotu123gmail.com", password: "chhotu", name: "Chhotu",username: "chhotu"
        });
        expect(res.statusCode).toEqual(400);
    }, 30000);
});
```

If given email is already registered then this case triggered and this required one testcase to check.

```
// duplicate email
describe("when email is already registered", () => {
    test("should return 409", async () => {
        const res = await supertest(app).post("/api/auth/register").send({
            email: "chaman123@gmail.com", password: "chaman", name: "duplicate",username: "chaman_lite"
        });
        expect(res.statusCode).toEqual(409);
    }, 30000);
});
```

Similarly for username as email only one testcase require to check the case.

```
// duplicate username
describe("when username is already registered", () => {
    test("should return 409", async () => {
        const res = await supertest(app).post("/api/auth/register").send({
            email: "chamandup@gmail.com", password: "chaman", name: "duplicate",username: "chaman"
        });
        expect(res.statusCode).toEqual(409);
    }, 30000);
});
```

If all the field are added and also right credentials then the expected value for the testcase is 200 status code and message body as “User registered successfully”.

```
// on successfull registeration
describe("when all credentials are present", () => {
    test("should have success status code, message and json object", async () => {
        const res = await supertest(app).post("/api/auth/register").send({
            email: "first123@gmail.com", password: "first", name: "Pehla",username: "first"
        });
        expect(res.statusCode).toEqual(200);
        expect(res.body).toHaveProperty("message","User registered successfully.");
        expect(res.headers["content-type"]).toEqual(expect.stringContaining("json"));
    }, 30000);
});
```

Another module to test is sign-in. And in this module, expecting email and password.

Test1 : when either field is empty

```
84
85  // test for invalid registeration with insufficient credentials
86  describe("when email or password or both are missing", () => {
87      it("should return 400", async () => {
88          const bodyData = [
89              {email: "chaman123@gmail.com"},
90              {password: "chaman"},
91              {}
92          ]
93          for( const data of bodyData){
94              const res = await supertest(app).post("/api/auth/login").send(data);
95              expect(res.statusCode).toEqual(400);
96          }
97      });
98  });
99
```

Test2: Providing correct information email and password.

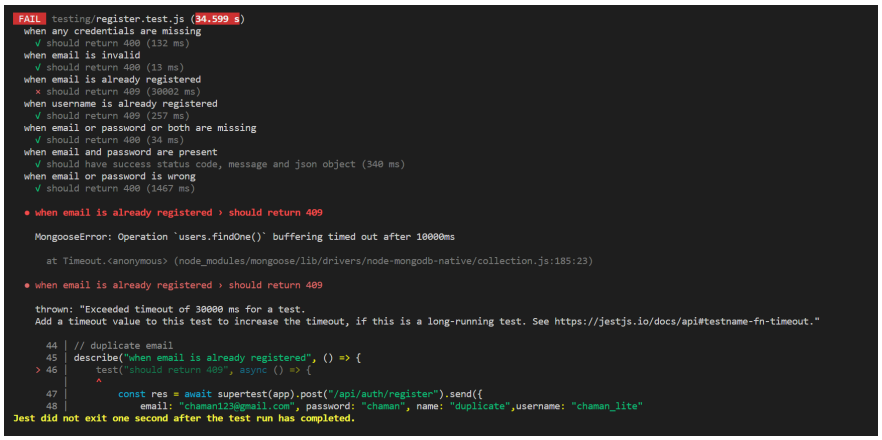
```
100 // on successfull login
101 describe("when email and password are present", () => {
102     test("should have success status code, message and json object", async () => {
103         const res = await supertest(app).post("/api/auth/login").send({
104             email: "chaman123@gmail.com",
105             password: "chaman",
106         });
107         expect(res.statusCode).toEqual(200);
108         expect(res.body).toHaveProperty("message","User logged in successfully.");
109         expect(res.headers["content-type"]).toEqual(expect.stringContaining("json"));
110     }, 30000);
111 });
112
```

Test3 : when we provide all the field but email or password is wrong.

```
114 // should return 400 for invalid login credentials
115 describe("when email or password is wrong", () => {
116     test("should return 400", async () => {
117         const bodyData = [
118             {email: "123@gmail.com", password: "chaman"},
119             {email: "chaman123@gmail.com", password: "123"},
120             {email: "123@gmail.com", password: "123"}
121         ]
122         for( const data of bodyData){
123             const res = await supertest(app).post("/api/auth/login").send(data);
124             expect(res.statusCode).toEqual(409);
125         }
126     }, 10000);
127 });
128
```

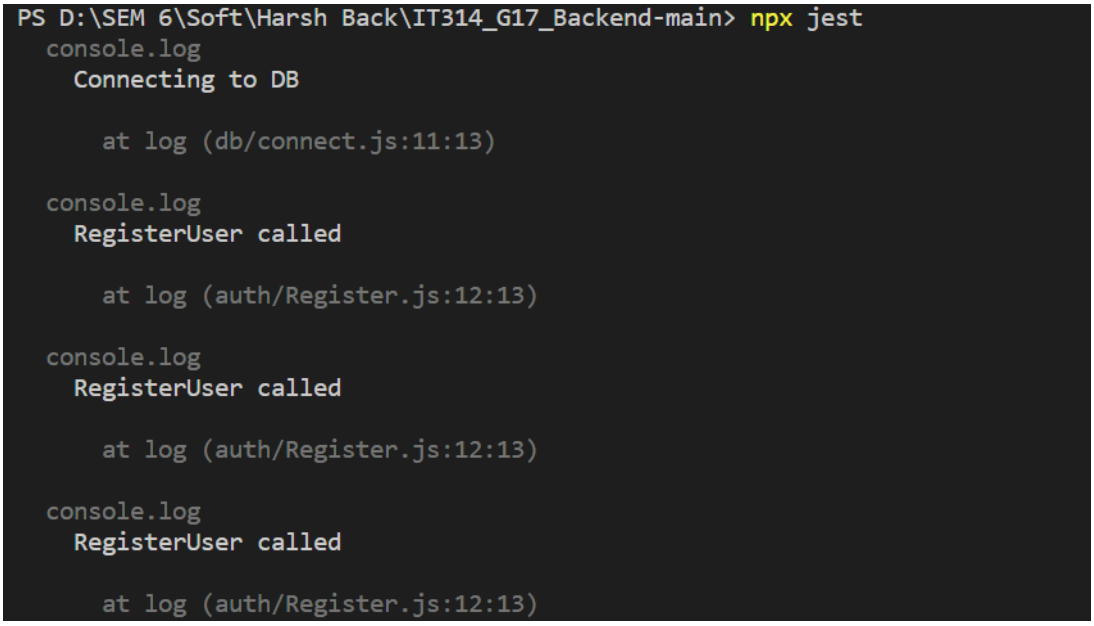
- Execute the test cases by using the unit testing framework.

For the unit testing of the given modules, we used Jest frame work to test the nodejs API and running the testcases.



Since after writing the initial testcases we got almost all the tescases passed except 3rd which is “email is already registered”.

To resolve the above bug from the API codebase we debug the code and fix that but and below is the console for that debugging.



And below is the screenshot of the terminal for passing all the testcases.

```
  console.log
    RegisterUser called

      at log (auth/Register.js:12:13)

  console.log
    RegisterUser called

      at log (auth/Register.js:12:13)

  console.log
    Server listening on port 3000...

      at Server.log (server.js:41:17)

  console.log
    RegisterUser called

      at log (auth/Register.js:12:13)
  when email and password are present
    √ should have success status code, message and json object (332 ms)
  when email or password is wrong
    √ should return 400 (775 ms)

Test Suites: 1 passed, 1 total
Tests:       7 passed, 7 total
Snapshots:   0 total
Time:        9.774 s, estimated 35 s
Ran all test suites.
```

For the sign-in module, we got all the test cases passed after running the test file.

```
PS D:\SEM 6\Soft\Harsh Back\IT314_G17_Backend-main> npx jest
  console.log
    Connecting to DB
  when email or password or both are missing
    √ should return 400 (88 ms)
  when email and password are present
    √ should have success status code, message and json object (7213 ms)
  when email or password is wrong
    √ should return 400 (5400 ms)

Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        15.625 s, estimated 19 s
Ran all test suites.
```

- Once the test case failed, How you fixed that?

In the module sign-up, there is one test case which got failed. As terminology suggests the defect is reported by the developer, the bug is addressed, and the developer found a typo error this was not caught by the static analysis tool as both the typo variable name is already present in the file, so by fixing the bug, test case passed.