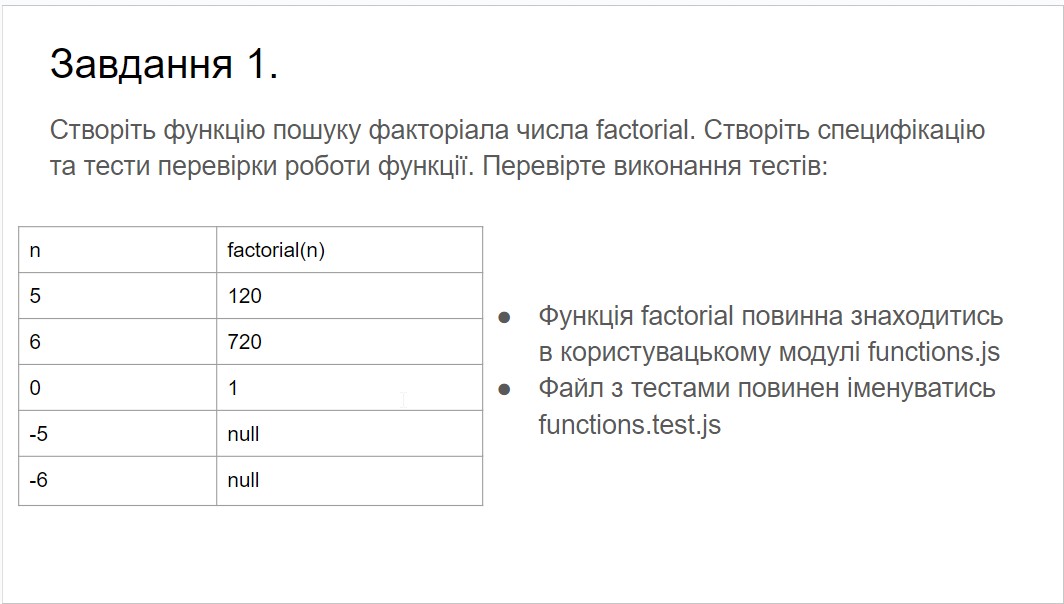
**ЛАБОРАТОРНА РОБОТА № 7**

**Розробка тестів з використанням фреймворка Mocha**

**Хід роботи:**



**Лістинг програми:**

const { factorial } = require('../utils/functions');

const mocha = require('mocha');

const valueDictionary = {

    5: 120,

    6: 720,

    0: 1,

    "-5": null,

    "-6": null

};

describe('Factorial function', () => {

    for (let [number, expected] of Object.entries(valueDictionary)) {

        it(`should return ${expected} for number ${number}`, () => {

            let result = factorial(parseInt(number));

            let correct = expected;

            if (result !== correct) {

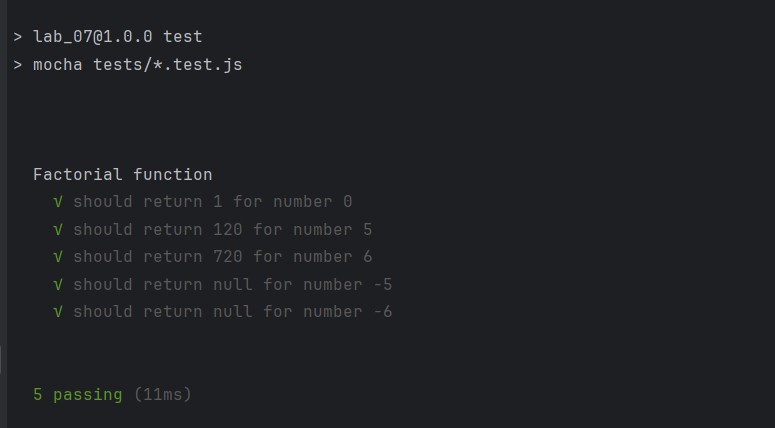
                throw new Error(`Incorrect result of factorial for number ${number}. Expected ${correct}, but got ${result}.`);

            }

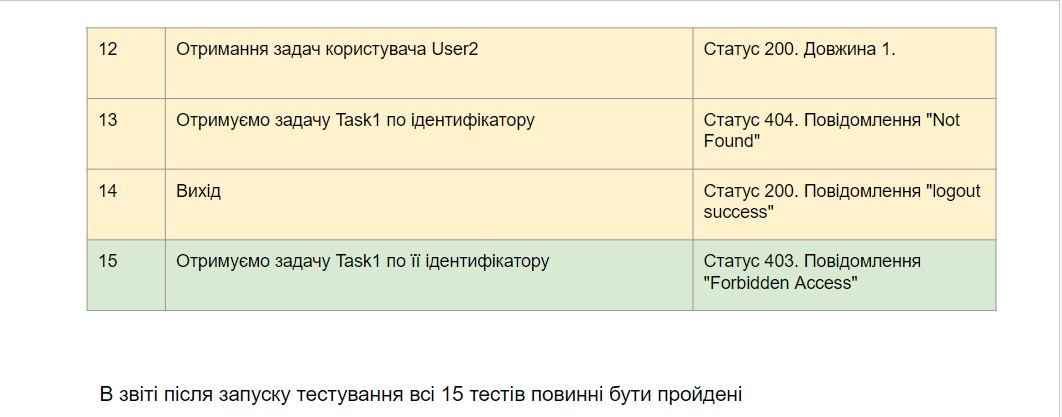
        });

    }

});







**Лістинг програми:**

process.env.NODE\_ENV = 'test';

let mongoose = require("mongoose");

let User = require('../src/models/user');

let Task = require('../src/models/task');

let chai = require('chai');

let chaiHttp = require('chai-http');

let server = require('../src/app');

chai.should();

let token = "";

let task1Id = "";

chai.use(chaiHttp);

describe('Тести', () => {

    before(async () => {

        try {

            await User.deleteMany({});

        } catch (error) {

            throw new Error(error.message);

        }

    });

    it('1. Реєстрація користувача User1 з помилкою валідації => Статус 401 та назва помилки', (done) => {

        const user = {

            name: 'User1',

            email: 'user1user1',

            password: '1234567890'

        }

        chai.request(server)

            .post('/users')

            .send(user)

            .end((err, res) => {

                res.should.have.status(401);

                const errorMessage = res.body.message;

                errorMessage.should.match(/Email is invalid/);

                done();

            });

    });

    it('2. Реєстрація користувача User1 без помилок => Статус 200. Отриманий об\'єкт user з властивістю \_id', (done) => {

        let user = {

            name: 'User1',

            email: 'user1@example.com',

            password: '1234567890'

        }

        chai.request(server)

            .post('/users')

            .send(user)

            .end((err, res) => {

                res.should.have.status(200);

                token = res.body.token;

                done();

            })

    });

    it('3. Реєстрація користувача User2 без помилок => Статус 200 і отриманий об\'єкт user з ідентифікатором', (done) => {

        let user = {

            name: 'User2',

            email: 'user2@example.com',

            password: '1234567890'

        }

        chai.request(server)

            .post('/users')

            .send(user)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.should.be.a('object');

                res.body.should.have.property('\_id');

                done();

            })

    });

    it('4. Вхід під User1 з вірними даними => Статус 200 і отримане повідомлення success\n', (done) => {

        chai.request(server)

            .post('/users/login')

            .send({ email: 'user1@example.com', password: '1234567890' })

            .end((err, res) => {

                res.should.have.status(200);

                res.body.message.should.equal('Login successful');

                token = res.body.token;

                done();

            })

    });

    before(async () => {

        try {

            await Task.deleteMany({});

        } catch (error) {

            throw new Error(error.message);

        }

    });

    it('5. Додавання задачі Task1 => Статус 200 і отриманий об\'єкт task з ідентифікатором', (done) => {

        const task = {

            title: 'Task1',

            description: 'Task1 description',

            completed: false

        }

        chai.request(server)

            .post('/tasks')

            .set('Authorization', `Bearer ${token}`)

            .send(task)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.should.be.a('object');

                res.body.should.have.property('\_id');

                task1Id = res.body.\_id;

                done();

            });

    });

    it('6. Додавання задачі Task2 => Статус 200 і отриманий об\'єкт task з ідентифікатором', (done) => {

        const task = {

            title: 'Task2',

            description: 'Task2 description',

            completed: false

        }

        chai.request(server)

            .post('/tasks')

            .set('Authorization', `Bearer ${token}`)

            .send(task)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.should.be.a('object');

                res.body.should.have.property('\_id');

                task1Id = res.body.\_id;

                done();

            });

    });

    it('7 Отримання задач користувача User1 => Статус 200. Довжина 2', (done) => {

        chai.request(server)

            .get('/tasks')

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.length.should.equal(2);

                done();

            });

    });

    it('8. Отримуємо задачу Task1 по ідентифікатору => Статус 200. Об\'єкт task з властивостями title і completed.', (done) => {

        chai.request(server)

            .get(`/tasks/${task1Id}`)

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.should.be.a('object');

                res.body.should.have.property('title');

                res.body.should.have.property('completed');

                done();

            });

    });

    it('9. Вихід => Повідомлення "logout success"\n', (done) => {

        chai.request(server)

            .post('/users/logout')

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.message.should.equal('Logout successful');

                done();

            });

    });

    it('10. Вхід під User2 з вірними даними => Статус 200 і отримане повідомлення success', (done) => {

        chai.request(server)

            .post('/users/login')

            .send({ email: 'user2@example.com', password: '1234567890' })

            .end((err, res) => {

                res.should.have.status(200);

                res.body.message.should.equal('Login successful');

                token = res.body.token;

                done();

            })

    });

    it('11. Додавання задачі Task3 => Статус 200 і отриманий об\'єкт task з ідентифікатором', (done) => {

        const task = {

            title: 'Task3',

            description: 'Task3 description',

            completed: false

        }

        chai.request(server)

            .post('/tasks')

            .set('Authorization', `Bearer ${token}`)

            .send(task)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.should.be.a('object');

                res.body.should.have.property('\_id');

                done();

            })

    });

    it('12. Отримання задач користувача User2 => Статус 200. Довжина 1.', (done) => {

        chai.request(server)

            .get('/tasks')

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.length.should.equal(1);

                done();

            });

    });

    it('13. Отримуємо задачу Task1 по ідентифікатору => Статус 404. Повідомлення "Not Found"', (done) => {

        chai.request(server)

            .get(`/tasks/${task1Id}`)

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(404);

                res.body.message.should.equal('Not found');

                done();

            });

    });

    it('14. Вихід => Статус 200. Повідомлення "logout success"', (done) => {

        chai.request(server)

            .post('/users/logout')

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(200);

                res.body.message.should.equal('Logout successful');

                token = null;

                done();

            });

    });

    it('15. Отримуємо задачу Task1 по її ідентифікатору => Статус 403. Повідомлення "Forbidden Access"\n', (done) => {

        chai.request(server)

            .get(`/tasks/${task1Id}`)

            .set('Authorization', `Bearer ${token}`)

            .end((err, res) => {

                res.should.have.status(401);

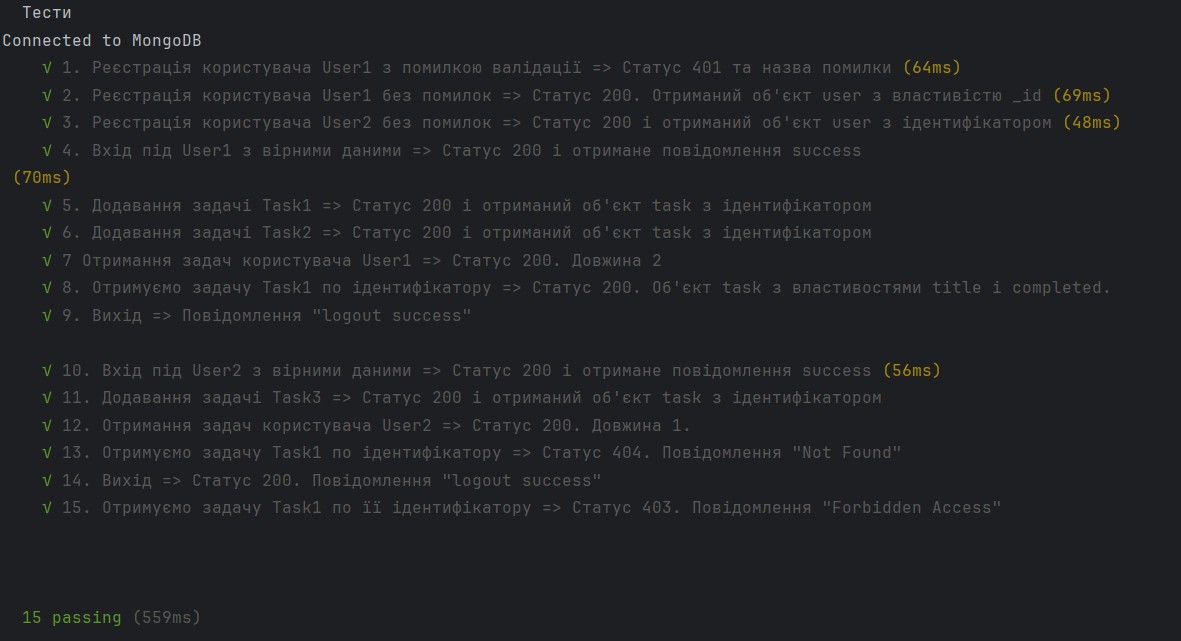
                //res.body.should.have.property('message').eql('Forbidden Access');

                done();

            });

    });

});



**Висновок:** Навчилася розробці тестів з використанням фреймворка Mocha.