Посилання на github: <https://github.com/Ilya-Olenchenko/Script-programming.git>

**Лабораторна робота № 4.**

**Mongoose**

Короткі теоретичні відомості

Mongoose представляє спеціальну ODM-бібліотеку (*Object Data Modelling*) для роботи з MongoDB, що дозволяє розміщувати об'єкти класів та документи колекцій з баз даних. Mongoose працює подібним інструментом ORM. Офіційний сайт бібліотеки, де можна переглянути всю необхідну документацію: <http://mongoosejs.com>.

Для роботи з Mongoose продовжуємо роботу з попереднім проектом і спочатку встановимо саму бібліотеку за допомогою команди:

npm install mongoose --save

Хід роботи

**Результати виконання роботи**

Лістинг 1.1 файл app.js.

const mongoose = require("mongoose");

const express = require("express");

const Schema = mongoose.Schema;

const app = express();

const jsonParser = express.json();

const studentScheme = new Schema(

    {

        name: String,

        lastname: String,

        age: Number,

        group: String,

    },

    { versionKey: false }

);

const Student = mongoose.model("Student", studentScheme);

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

mongoose.connect("mongodb://localhost:27017/studentsdb", { useUnifiedTopology: true, useNewUrlParser: true, useFindAndModify: false }, function (err) {

    if (err) return console.log(err);

    app.listen(3000, function () {

        console.log("Сервер запущено...");

    });

});

app.get("/api/students", function (req, res) {

    Student.find({}, function (err, students) {

        if (err) return console.log(err);

        res.send(students)

    });

});

app.get("/api/students/:id", function (req, res) {

    const id = req.params.id;

    Student.findOne({ \_id: id }, function (err, students) {

        if (err) return console.log(err);

        res.send(students);

    });

});

app.post("/api/students", jsonParser, function (req, res) {

    if (!req.body) return res.sendStatus(400);

    const studentName = req.body.name;

    const studentLastName = req.body.lastname;

    const studentAge = req.body.age;

    const studentGroup = req.body.group;

    const student = new Student({ name: studentName, lastname: studentLastName, age: studentAge, group: studentGroup });

    student.save(function (err) {

        if (err) return console.log(err);

        res.send(student);

    });

});

app.delete("/api/students/:id", function (req, res) {

    const id = req.params.id;

    Student.findByIdAndDelete(id, function (err, student) {

        if (err) return console.log(err);

        res.send(student);

    });

});

app.put("/api/students", jsonParser, function (req, res) {

    if (!req.body) return res.sendStatus(400);

    const id = req.body.id;

    const studentName = req.body.name;

    const studentLastName = req.body.lastname;

    const studentAge = req.body.age;

    const studentGroup = req.body.group;

    const newStudent = { age: studentAge, lastname: studentLastName, name: studentName, group: studentGroup };

    Student.findOneAndUpdate({ \_id: id }, newStudent, { new: true }, function (err, student) {

        if (err) return console.log(err);

        res.send(student);

    });

});

На рисунку 1 зображено відкритий **localhost:3000** в браузері.

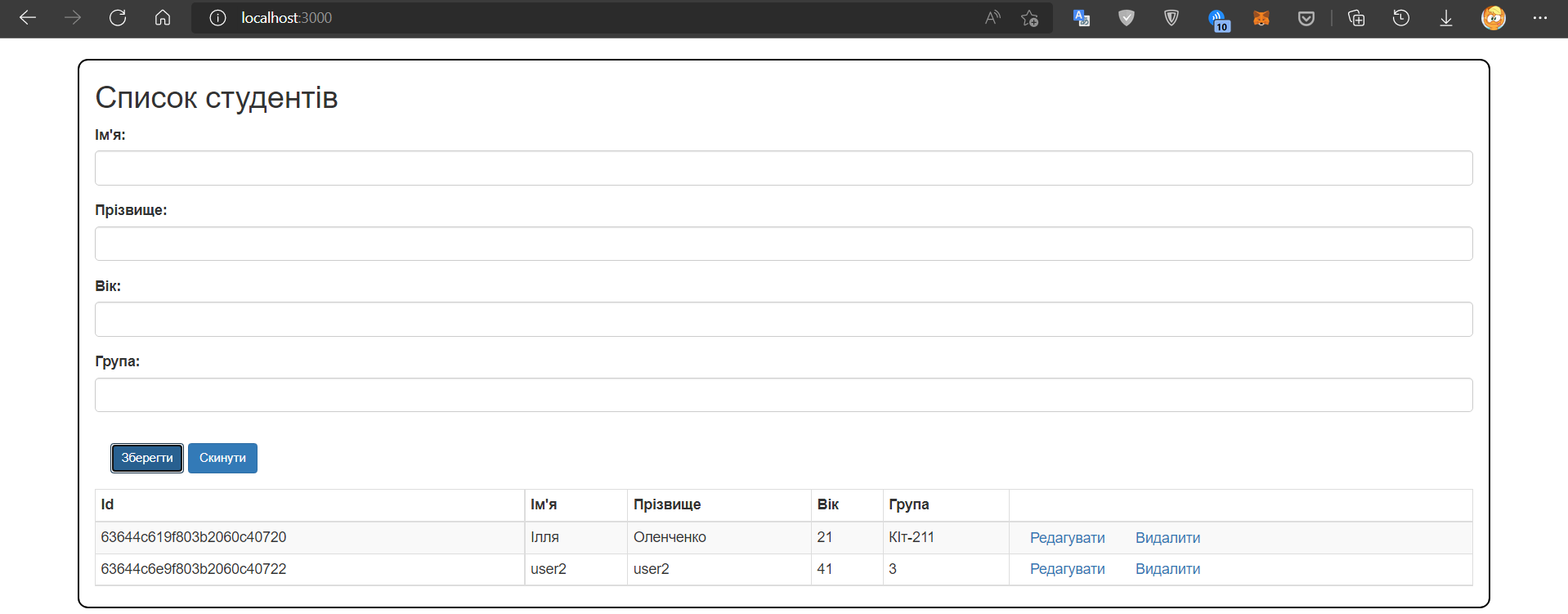


Рисунок 1 - Зображено запущений localhost:3000

На рисунку 2 зображено заповнену БД в MongoDB Compass.

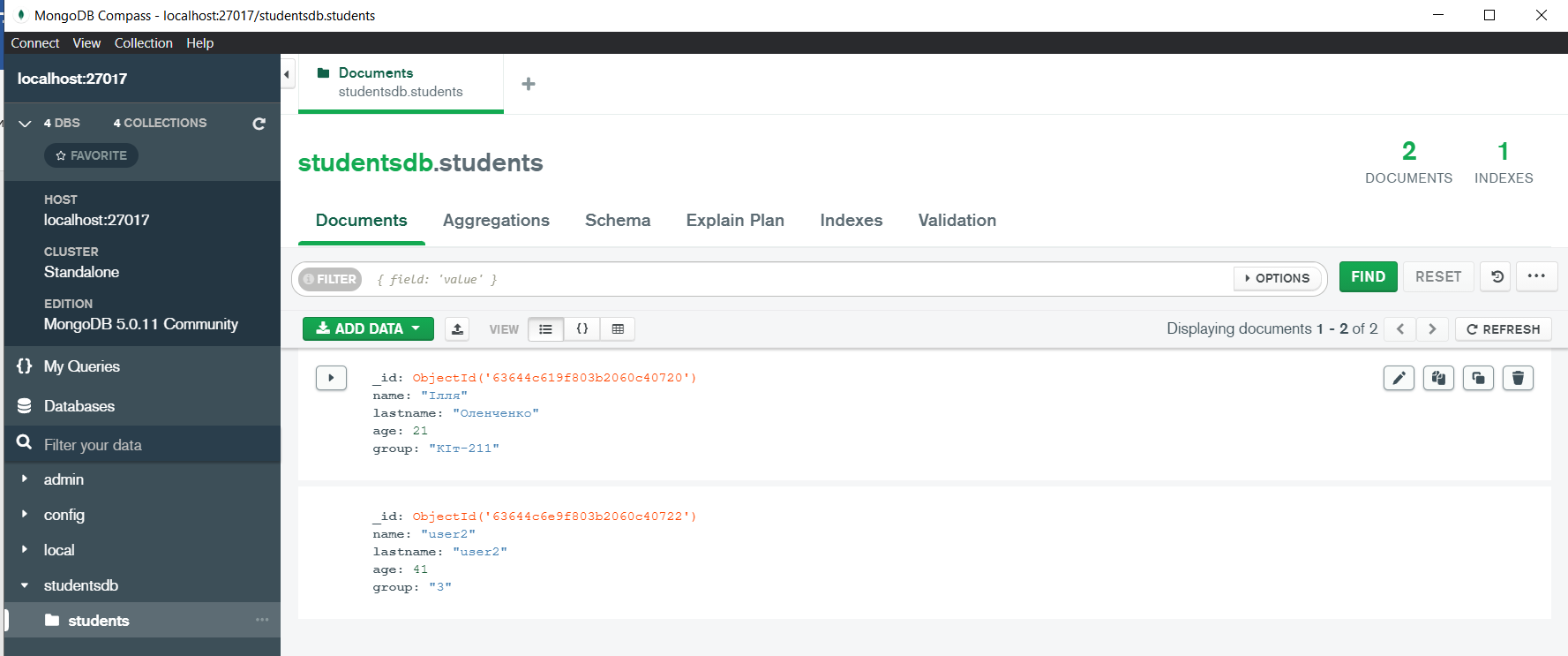


Рисунок 2 - Зображено БД в MongoDB Compass

**Висновок:** моодіфікували створений додаток з використанням mongoose і власної бази MongoDB.