Node.js

Лабораторна робота №3

Mongoose

Rest API

Лістинг програми(app.js):

const express = require("express");

const mongoose = require("mongoose");

const userRouter = require("./routers/user");

const taskRouter = require("./routers/task");

require("dotenv").config();

const app = express();

const url = process.env.MONGO\_URL;

const port = process.env.PORT;

mongoose.connect(url);

app.use(express.json());

app.get("/", (req, res) => {

res.send("Ласкаво просимо до API користувачів та завдань");

});

app.use(userRouter);

app.use("/tasks", taskRouter);

app.listen(port, () => {

console.log(`Сервер слухає на порті ${port}`);

});

module.exports = app;

Лістинг програми(models/user):

const mongoose = require("mongoose");

const validator = require("validator");

const userSchema = new mongoose.Schema({

name: {

type: String,

required: true,

trim: true

},

age: {

type: Number,

default: 0,

validate(value) {

if (value < 0) {

throw new Error("Age must be a positive number");

}

},

required: true

},

email: {

type: String,

required: [true, 'Email is required'],

unique: true,

lowercase: true,

validate(value) {

if (!validator.isEmail(value)) {

throw new Error("Email is not correct");

}

}

},

password: {

type: String,

required: true,

minlength: 6

}

});

const User = mongoose.model('User', userSchema);

module.exports = User;

Лістинг програми(routers/user):

const express = require('express');

const User = require("../models/user");

const router = new express.Router();

router.get('/test', (req, res) => {

res.send("From a new File");

});

router.post("/", async (req, res) => {

try {

const user = new User(req.body);

await user.save();

res.json(user);

} catch (error) {

res.status(400).send(error.message);

}

});

router.get("/:id", async (req, res) => {

try {

const user = await User.findOne({ \_id: req.params.id });

if (!user) {

res.status(404).send("Користувач не знайдений");

return;

}

res.json(user);

} catch (error) {

res.status(400).send(error.message);

}

});

router.get("/", async (req, res) => {

try {

const users = await User.find();

res.json(users);

} catch (error) {

res.status(400).send(error.message);

}

});

router.patch("/:id", async (req, res) => {

try {

const user = await User.findOne({ \_id: req.params.id });

if (!user) {

res.status(404).send("Користувач не знайдений");

return;

}

const fields = ["name", "age", "email", "password"];

fields.forEach(field => {

if (req.body[field]) {

user[field] = req.body[field];

}

});

await user.save();

res.json(user);

} catch (error) {

res.status(400).send(error.message);

}

});

router.delete("/:id", async (req, res) => {

try {

const user = await User.findOneAndDelete({ \_id: req.params.id });

if (!user) {

res.status(404).send("Користувач не знайдений");

return;

}

res.json(user);

} catch (error) {

res.status(400).send(error.message);

}

});

router.delete("/", async (req, res) => {

try {

const result = await User.deleteMany();

res.json(result);

} catch (error) {

res.status(400).send(error.message);

}

});

module.exports = router;

Лістинг програми(models/task):

const express = require("express");

const Task = require("../models/task");

const router = new express.Router();

router.post("/", async (req, res) => {

try {

const task = new Task(req.body);

await task.save();

res.status(201).send(task);

} catch (error) {

res.status(400).send(error.message);

}

});

router.get("/", async (req, res) => {

try {

const tasks = await Task.find();

res.send(tasks);

} catch (error) {

res.status(500).send(error.message);

}

});

module.exports = router;

Лістинг програми(routers/task):

const mongoose = require("mongoose");

const taskSchema = new mongoose.Schema({

title: {

type: String,

required: true,

trim: true

},

description: {

type: String,

required: true,

trim: true

},

completed: {

type: Boolean,

default: false

}

});

const Task = mongoose.model("Task", taskSchema);

module.exports = Task;

Результат виконання програми:





