**Лабораторна робота №3**

Node.js та MongoDB.

**Мета роботи:**

Ознайомитись з mongoDB,

**Результат виконання:**

Лістинг 1 - app.js

const express = require("express");

const app = express();

const bodyParser = require("body-parser");

const jsonParser = express.json();

const {

  connectMongoDB,

  closeMongoDB,

  getUsers,

  getUser,

  insertUser,

  removeUserByName,

  updateUser,

  removeAllUsers,

} = require("./mongo");

const host = process.env.API\_URL;

const port = process.env.API\_PORT;

app.use(bodyParser.urlencoded({ extended: false }));

app.use(bodyParser.json());

app.use(express.static("public"));

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

  await connectMongoDB();

  const users = await getUsers();

  await closeMongoDB();

  res.json(users);

});

app.get("/api/users/:id", async (req, res) => {

  await connectMongoDB();

  const user = await getUser(req.params.id);

  await closeMongoDB();

  res.json(user);

});

app.post("/api/users", jsonParser, async (req, res) => {

  await connectMongoDB();

  await insertUser(req.body);

  await closeMongoDB();

  res.json(req.body);

});

app.delete("/api/users/", async (req, res) => {

  await connectMongoDB();

  await removeAllUsers();

  await closeMongoDB();

  res.json({ message: "All users removed" });

});

app.delete("/api/users/:id", async (req, res) => {

  await connectMongoDB();

  await removeUserByName(req.params.id);

  await closeMongoDB();

  res.json({ message: "User removed", \_id: req.params.id });

});

app.put("/api/users/", jsonParser, async (req, res) => {

  await connectMongoDB();

  const response = await updateUser(req.body);

  await closeMongoDB();

  res.json(response);

});

app.listen(port, () => {

  console.log(`Server is running at ${host}:${port}`);

});

Лістинг 2 – index.html

<!DOCTYPE html>

<html>

<head>

    <meta charset="utf-8" />

    <meta name="viewport" content="width=device-width" />

    <title>Список пользователей</title>

    <link href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" rel="stylesheet" />

</head>

<body>

    <h2>Список пользователей</h2>

    <form name="userForm">

        <input type="hidden" name="id" value="0" />

        <div class="form-group">

            <label for="name">Имя:</label>

            <input class="form-control" name="name" />

        </div>

        <div class="form-group">

            <label for="age">Возраст:</label>

            <input class="form-control" name="age" />

        </div>

        <div class="panel-body">

            <button type="submit" class="btn btn-sm btn-primary">Сохранить</button>

            <button type="reset" class="btn btn-sm btn-primary">Сбросить</button>

            <button id="remove-all" class="btn btn-sm btn-primary">Удалить все</button>

        </div>

    </form>

    <table class="table table-condensed table-striped table-bordered">

        <thead>

            <tr>

                <th>Id</th>

                <th>Имя</th>

                <th>возраст</th>

                <th></th>

            </tr>

        </thead>

        <tbody>

        </tbody>

    </table>

    <style>

        body {

            margin: 20px;

        }

    </style>

    <script>

        // Получение всех пользователей

        async function GetUsers() {

            // отправляет запрос и получаем ответ

            const response = await fetch("/api/users", {

                method: "GET",

                headers: { "Accept": "application/json" }

            });

            // если запрос прошел нормально

            if (response.ok === true) {

                // получаем данные

                const users = await response.json();

                let rows = document.querySelector("tbody");

                users.forEach(user => {

                    // добавляем полученные элементы в таблицу

                    rows.append(row(user));

                });

            }

        }

        // Получение одного пользователя

        async function GetUser(id) {

            const response = await fetch("/api/users/" + id, {

                method: "GET",

                headers: { "Accept": "application/json" }

            });

            if (response.ok === true) {

                const user = await response.json();

                const form = document.forms["userForm"];

                form.elements["id"].value = user.\_id;

                form.elements["name"].value = user.name;

                form.elements["age"].value = user.age;

            }

        }

        // Добавление пользователя

        async function CreateUser(userName, userAge) {

            const response = await fetch("api/users", {

                method: "POST",

                headers: { "Accept": "application/json", "Content-Type": "application/json" },

                body: JSON.stringify({

                    name: userName,

                    age: parseInt(userAge, 10)

                })

            });

            if (response.ok === true) {

                const user = await response.json();

                reset();

                document.querySelector("tbody").append(row(user));

            }

        }

        // Изменение пользователя

        async function EditUser(userId, userName, userAge) {

            const response = await fetch("api/users", {

                method: "PUT",

                headers: { "Accept": "application/json", "Content-Type": "application/json" },

                body: JSON.stringify({

                    \_id: userId,

                    name: userName,

                    age: parseInt(userAge, 10)

                })

            });

            if (response.ok === true) {

                const user = await response.json();

                reset();

                console.log(user);

                document.querySelector("tr[data-rowid='" + user.\_id + "']").replaceWith(row(user));

            }

        }

        // Удаление пользователя

        async function DeleteUser(id) {

            const response = await fetch("/api/users/" + id, {

                method: "DELETE",

                headers: { "Accept": "application/json" }

            });

            if (response.ok === true) {

                const user = await response.json();

                console.log(user);

                document.querySelector("tr[data-rowid='" + user.\_id + "']").remove();

            }

        }

        // сброс формы

        function reset() {

            const form = document.forms["userForm"];

            form.reset();

            form.elements["id"].value = 0;

        }

        // создание строки для таблицы

        function row(user) {

            const tr = document.createElement("tr");

            tr.setAttribute("data-rowid", user.\_id);

            const idTd = document.createElement("td");

            idTd.append(user.\_id);

            tr.append(idTd);

            const nameTd = document.createElement("td");

            nameTd.append(user.name);

            tr.append(nameTd);

            const ageTd = document.createElement("td");

            ageTd.append(user.age);

            tr.append(ageTd);

            const linksTd = document.createElement("td");

            const editLink = document.createElement("a");

            editLink.setAttribute("data-id", user.\_id);

            editLink.setAttribute("style", "cursor:pointer;padding:15px;");

            editLink.append("Изменить");

            editLink.addEventListener("click", e => {

                e.preventDefault();

                console.log(user.\_id);

                GetUser(user.\_id);

            });

            linksTd.append(editLink);

            const removeLink = document.createElement("a");

            removeLink.setAttribute("data-id", user.\_id);

            removeLink.setAttribute("style", "cursor:pointer;padding:15px;");

            removeLink.append("Удалить");

            removeLink.addEventListener("click", e => {

                e.preventDefault();

                DeleteUser(user.\_id);

            });

            linksTd.append(removeLink);

            tr.appendChild(linksTd);

            return tr;

        }

        document.querySelector("#remove-all").addEventListener("click", e => {

            e.preventDefault();

            fetch("/api/users", {

                method: "DELETE",

                headers: { "Accept": "application/json" }

            }).then(response => {

                if (response.ok === true) {

                    document.querySelector("tbody").innerHTML = "";

                }

            });

        });

        // отправка формы

        document.forms["userForm"].addEventListener("submit", e => {

            e.preventDefault();

            const form = document.forms["userForm"];

            const id = form.elements["id"].value;

            const name = form.elements["name"].value;

            const age = form.elements["age"].value;

            if (id == 0)

                CreateUser(name, age);

            else

                EditUser(id, name, age);

        });

        // загрузка пользователей

        GetUsers();

    </script>

</body>

</html>

Лістинг 3 – mongo.js  
const { MongoClient, ObjectId } = require("mongodb");

const url = "mongodb://localhost:27017/";

const mongoClient = new MongoClient(url);

const connectMongoDB = async () => {

  try {

    await mongoClient.connect();

  } catch (e) {

    console.log(e);

  }

};

const closeMongoDB = async () => {

  try {

    await mongoClient.close();

  } catch (e) {

    console.log(e);

  }

};

const insertUser = async (user) => {

  try {

    const db = mongoClient.db("usersdb");

    const collection = db.collection("users");

    await collection.insertOne(user);

  } catch (e) {

    console.log(e);

  }

};

const getUser = async (id) => {

  try {

    const db = mongoClient.db("usersdb");

    const collection = db.collection("users");

    return await collection.findOne({ \_id: ObjectId.createFromHexString(id) });

  } catch (e) {

    console.log(e);

  }

};

const getUsers = async () => {

  try {

    const db = mongoClient.db("usersdb");

    const collection = db.collection("users");

    return await collection.find({}).toArray();

  } catch (e) {

    console.log(e);

  }

};

const removeUserByName = async (id) => {

  try {

    const db = mongoClient.db("usersdb");

    const collection = db.collection("users");

    await collection.deleteOne({ \_id: ObjectId.createFromHexString(id) });

  } catch (e) {

    console.log(e);

  }

};

const removeAllUsers = async () => {

  try {

    const db = mongoClient.db("usersdb");

    const collection = db.collection("users");

    await collection.deleteMany({});

  } catch (e) {

    console.log(e);

  }

};

const updateUser = async (user) => {

  try {

    const db = mongoClient.db("usersdb");

    const collection = db.collection("users");

    const userId = user.\_id;

    delete user.\_id;

    const response = await collection.findOneAndUpdate(

      { \_id: ObjectId.createFromHexString(userId) },

      { $set: user },

      { returnDocument: "after" }

    );

    return response;

  } catch (e) {

    console.log(e);

  }

};

module.exports = {

  connectMongoDB,

  closeMongoDB,

  insertUser,

  getUser,

  getUsers,

  removeUserByName,

  updateUser,

  removeAllUsers,

};

В результаті отримали додаток, який працює з БД mongoDB у хмарі

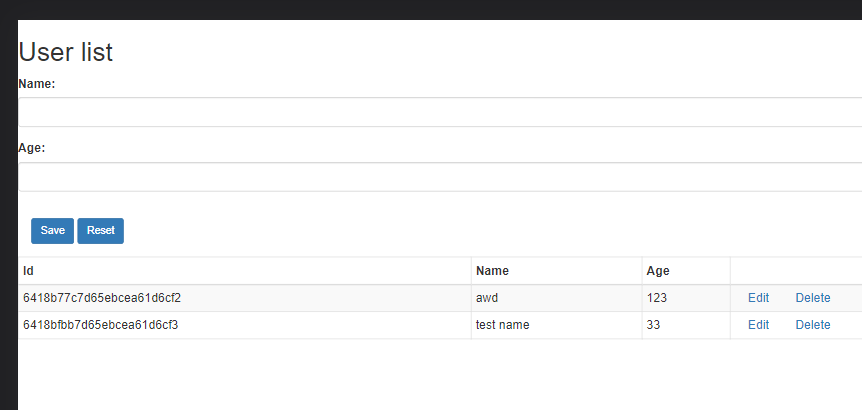


Рисунок 1 – робочий додаток

**Висновок:** Ознайомились з mongoDB. Створили додаток для роботи з БД.