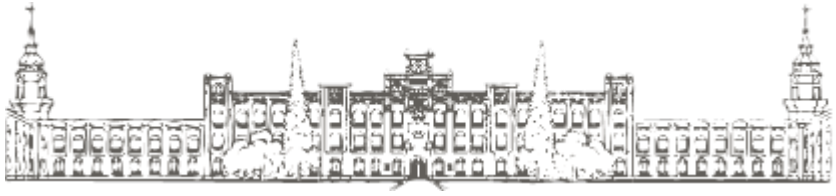


Міністерство освіти і науки України

Національний технічний університет України „КПІ імені Ігоря Сікорського ”

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Звіт до комп’ютерного практикуму №5

З дисципліни «Основи Back-end технологій»

Прийняв:
Зубко Роман Анатолійович

Виконав:
Студент 3 курсу
Кравчук Ілля Володимирович
Група ІМ-13

2024 р.

Лабораторна робота №5

Тема: NodeJS. Робота з БД MongoDB. Додаток що реалізує CRUD операції в БД.

Завдання.

- Створити додаток, що реалізує CRUD операції з БД – додавання, читання, редагування та видалення записів БД.
- Забезпечити роутінг запитів та виведення результатів запитів на WEBсторінку.
- Додати новий роут для виведення інформації у вигляді json-файлу.

Варіант 16. Спроектувати базу даних про договори: назва фірми-клієнта, вид договору, термін дії.

Хід роботи

Створив додаток, що реалізує CRUD операції з БД – додавання, читання, редагування та видалення записів БД. Забезпечив роутінг запитів та виведення результатів запитів на WEBсторінку.

Код програми

index.js

```
import express from 'express';
import mongoose from 'mongoose';
import path, { dirname } from 'path';
import { fileURLToPath } from 'url';
import { create } from 'express-handlebars';
import Handlebars from 'handlebars';
import { allowInsecurePrototypeAccess } from '@handlebars/allow-prototype-access';

import routes from './routes/routes.js';

const url =
'mongodb+srv://ikravchukim13:vru1d2HGSCbAldlT@cluster0.yazpbgq.mongodb.net/';
const PORT = process.env.PORT || 5000;

const app = express();

const hbs = create({
  defaultLayout: 'main',
  extname: 'hbs',
  handlebars: allowInsecurePrototypeAccess(Handlebars)
});

const __filename = fileURLToPath(import.meta.url);
const __dirname = dirname(__filename);

app.engine('hbs', hbs.engine);
```

```
app.set('view engine', 'hbs');
app.set('views', path.join(__dirname, 'views'));
app.use(express.static(path.join(__dirname, 'public')));
app.use(express.urlencoded({ extended: true }));
app.use(express.json());

app.use('/', routes);

const start = async () => {
  try {
    await mongoose.connect(url);
    app.listen(PORT, () => {
      console.log(`Server is running on port: ${PORT}`);
    });
  } catch (error) {
    console.error('Error starting the server:', error);
  }
};

start();
```

Цей код створює сервер на Node.js за допомогою Express.js для обробки HTTP-запитів. Він підключається до бази даних MongoDB через Mongoose, налаштовує шаблонізатор Handlebars для створення HTML-сторінок, визначає статичні файли та URL-кодує дані. Потім він завантажує маршрути та запускає сервер на вказаному порті.

infoNotation.js

```
import { model, Schema } from 'mongoose'

const infoCompaniesSchema = new Schema({
  companyName: {
    type: String,
    required: true
  },
  numberWorkers: {
    type: Number,
    required: true
  },
  contractName: {
    type: String,
    required: true
  },
  startDate: {
    type: Date,
    default: Date.now
  },
  endDate: {
    type: Date,
    default: Date.now
  }
})
```

```
export default model('infoCompanies', infoCompaniesSchema)
```

Цей код визначає структуру даних компаній для бази MongoDB, використовуючи Mongoose. Він встановлює обов'язкові поля, такі як назва компанії, кількість працівників та назва контракту, і додає поля для дат початку та закінчення контракту зі значеннями за замовчуванням.

routes.js

```
import { Router } from 'express';
import Controller from './controller/Controller.js';

const router = Router();

router.get('/', Controller.getMainPage);

router.get('/add', Controller.getAddNotationPage);
router.post('/add', Controller.createNewNotation);

router.get('/notes', Controller.getNotesPage);
router.get('/notes/:id', Controller.getNotationById);

router.get('/edit/:id', Controller.getEditNotationPage);
router.put('/edit/:id', Controller.editNotation);

router.get('/delete/:id', Controller.getDeleteNotationPage);
router.delete('/delete/:id', Controller.deletNotation);

router.get('/api/notes', Controller.getAllInfoJSON);
router.get('/api/notes/:id', Controller.getNotationInfoJSON);

export default router;
```

Цей код використовує Express.js для створення веб-сервера. Він налаштовує маршрути для різних сторінок та функцій додатку. Кожен маршрут пов'язується з відповідним обробником запитів з контролера, який містить функції для виконання відповідних дій. Коли сервер отримує запит на певний маршрут, він викликає відповідний обробник, який обробляє запит та повертає відповідь клієнт

Controller.js

```
import infoNotation from "../../models/infoNotation.js";

export default class Controller {
  static async getMainPage(req, res) {
    res.render('index', {
      title: 'Головна сторінка',
      isMain: true
    });
  }

  static async getAddNotationPage(req, res) {
```

```
    res.render('addNotation', {
      title: 'Додати нову картку',
      notationAddedMessage: req.query.notationAdded,
      isAdd: true
    });
  }

  static async getNotesPage(req, res) {
    const notation = await infoNotation.find();
    res.render('notes', {
      title: 'Записи',
      notation,
      isNotation: true
    });
  }

  static async createNewNotation(req, res) {
    try {
      const { companyName, numberWorkers, contractName, startDate, endDate } =
req.body;
      const newNotation = new infoNotation({ companyName, numberWorkers,
contractName, startDate, endDate });
      await newNotation.save();
      res.redirect('/add?notationAdded=true');
    } catch (e) {
      console.error(e);
      res.status(500).send('Помилка сервера');
    }
  }

  static async getNotationById(req, res) {
    const selectedNotation = await infoNotation.findById(req.params.id);
    res.render('viewNotation', {
      title: 'Технологічна картка',
      selectedNotation,
      isViewNotation: true
    });
  }

  static async getEditNotationPage(req, res) {
    const selectedNotation = await infoNotation.findById(req.params.id);
    let dateToStr = formatDate(selectedNotation.startDate);
    let dateToStr2 = formatDate(selectedNotation.endDate);
    res.render('editNotation', {
      title: 'Редагувати картку',
      selectedNotation,
      dateToStr,
      dateToStr2,
      isEdit: true
    });

    function formatDate(date) {
      const d = new Date(date);
```

```
      const year = d.getFullYear();
      let month = '' + (d.getMonth() + 1);
      let day = '' + d.getDate();
      if (month.length < 2) month = '0' + month;
      if (day.length < 2) day = '0' + day;
      return [year, month, day].join('-');
    }
  }

  static async editNotation(req, res) {
    try {
      const notationId = req.params.id;
      delete req.body.notationdId;
      await infoNotation.findByIdAndUpdate(notationId, req.body);
      res.json({ status: 0, notationId });
    } catch (e) {
      console.error(e);
      res.status(500).json({ status: 1, error: e });
    }
  }

  static async getDeleteNotationPage(req, res) {
    res.render('deleteNotation', {
      title: 'Видалити картку',
      layout: 'main',
      notationId: req.params.id,
      isDelete: true
    });
  }

  static async deletNotation(req, res) {
    try {
      const notationId = req.params.id;
      await infoNotation.findByIdAndDelete(notationId);
      res.json({ status: 0 });
    } catch (e) {
      console.error(e);
      res.status(500).json({ status: 1, error: e });
    }
  }

  static async getNotationInfoJSON(req, res) {
    const notationId = req.params.id;
    const selectedNotation = await infoNotation.findById(notationId);
    res.json(selectedNotation);
  }

  static async getAllInfoJSON(req, res) {
    const notation = await infoNotation.find();
    res.json(notation);
  }
}
```

Цей код визначає клас Controller, який містить різні методи для обробки запитів HTTP у веб-додатку на Node.js з використанням Express. Метод getMainPage відповідає за відображення головної сторінки, getAddNotationPage - за сторінку додавання нового запису, а getNotesPage - за сторінку з усіма записами. Метод createNewNotation додає новий запис до бази даних, а getNotationById відображає конкретний запис за його ідентифікатором. getEditNotationPage відображає сторінку редагування запису, а editNotation змінює вже існуючий запис. Метод getDeleteNotationPage відображає сторінку для видалення запису, і deleteNotation видаляє конкретний запис. Нарешті, getNotationInfoJSON повертає інформацію про запис у форматі JSON, а getAllInfoJSON - всі записи у форматі JSON.

delete.js

```
const hiddenInput = document.querySelector('input[type="hidden"][name="id"]')
const confirmButton = document.querySelector('input[type="button"][value="Так"]');

const notationId = hiddenInput.value

confirmButton.addEventListener('click', (event) => {
  fetch(`/delete/${notationId}`, {
    method: 'DELETE',
  })
  .then(res => res.json())
  .then(res => {
    if(res.status === 0) {
      window.location.href = '/notes'
    } else {
      console.log(res.error)
    }
  })
})
```

Цей код використовує на сторінці поле з ім'ям "id" і кнопку підтвердження. Він використовує значення цього поля для видалення запису на сервері за допомогою запиту DELETE. Після отримання відповіді перенаправляє користувача на сторінку /notes, якщо операція пройшла успішно, або виводить помилку в консоль, якщо її виникла.

edit.js

```
const submitButton = document.querySelector('button[type="button"]')

submitButton.addEventListener('click', (event) => {
  const form = document.querySelector('.edit-form');

  const notationId = form.querySelector('#notationId').value;
  const companyName = form.querySelector('#companyName').value;
  const numberWorkers = form.querySelector('#numberWorkers').value;
  const contractName = form.querySelector('#contractName').value;
  const startDate = form.querySelector('#startDate').value;
  const endDate = form.querySelector('#endDate').value;
```

```
  const updatedData = {
    companyName,
    numberWorkers,
    contractName,
    startDate,
    endDate
  };

  fetch(`/edit/${notationId}`, {
    method: 'PUT',
    headers: {
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(updatedData)
  })
  .then(res => res.json())
  .then(res => {
    if(res.status === 0) {
      window.location.href = `/notes/${notationId}`
    } else {
      console.log(res.error)
    }
  })
})
```

Цей JavaScript-код відповідає за оновлення даних у формі на сторінці. При кліку на кнопку він отримує дані з полів форми, виконує запит на сервер з цими даними для оновлення, і в разі успіху перенаправляє користувача на сторінку з деталями нотатки, або виводить помилку у консоль

main.hbs

```
{{>header}}

<body>
  {{{body}}}
  {{>footer}}
</body>

</html>
```

footer.hbs

```
<script src="/delete.js"></script>
<script src="/edit.js"></script>
```

header.hbs

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
  {{#if isMain}}
    <link rel="stylesheet" href="/index.css">
  {{/if}}
  {{#if isAdd}}
    <link rel="stylesheet" href="/addNewNotation.css">
  {{/if}}
  {{#if isNotation}}
    <link rel="stylesheet" href="/notes.css">
  {{/if}}
  {{#if isViewNotation}}
    <link rel="stylesheet" href="/viewNotation.css">
  {{/if}}
  {{#if isEdit}}
    <link rel="stylesheet" href="/editNotation.css">
  {{/if}}
  {{#if isDelete}}
    <link rel="stylesheet" href="/deleteNotation.css">
  {{/if}}
</head>
```

Файл header.hbs визначає заголовок сторінки і посилається на різні CSS-файли в залежності від того, на якій сторінці (головній, додавання, перегляду тощо) знаходиться користувач, використовуючи умовні блоки {{#if}}. Це дозволяє підключати відповідні стилі для кожної сторінки.

addNotation.hbs

```
<header>
        <ul>
                <li><a href="/">Головна сторінка</a></li>
                <li><a href="/notes">Записи</a></li>
                <li><a href="/add">Створити запис</a></li>
        </ul>
</header>
<div class="container">
    <form action="/add" method="post">
        <div class="form-group">
            <label for="companyName">Назва Компанії</label>
            <input type="text" id="companyName" name="companyName">
        </div>
        <div class="form-group">
            <label for="numberWorkers">Кількість працівників</label>
            <input type="text" id="numberWorkers" name="numberWorkers"
pattern="\d+" title="Please enter only numbers">
        </div>
        <div class="form-group">
            <label for="contractName">Контракти</label>
            <input type="text" id="contractName" name="contractName">
        </div>
        <div class="form-group">
            <label for="startDate">Дата початку контракту</label>
            <input type="text" id="startDate" name="startDate" pattern="\d{4}-
\d{2}-\d{2}" title="Please enter date in YYYY-MM-DD format">
        </div>
```

```
        <div class="form-group">
            <label for="endDate">Дата закінчення контракту</label>
            <input type="text" id="endDate" name="endDate" pattern="\d{4}-\d{2}-
\d{2}" title="Please enter date in YYYY-MM-DD format">
        </div>
        <button type="submit">Додати</button>
    </form>
</div>
```

Сторінка додавання записів.

deleteNotation.hbs

```
<header>
        <ul>
                <li><a href="/">Головна сторінка</a></li>
                <li><a href="/notes">Записи</a></li>
                <li><a href="/add">Створити запис</a></li>
        </ul>
</header>
<form action="/delete" method="post">
  <div class="confirmation-container">
    <div class="confirmation-message">
      <h1>Ви точно хочете видалити запис з ID: {{notationId}}?</h1>
      <div class="confirmation-buttons">
        <input type="hidden" name="id" value="{{notationId}}">
        <input type="button" value="Так">
        <a href="/notes">Ні</a>
      </div>
    </div>
  </div>
</form>
```

Сторінка уточнення про видалення запису.

editNotation.hbs

```
<header>
        <ul>
                <li><a href="/">Головна сторінка</a></li>
                <li><a href="/notes">Записи</a></li>
                <li><a href="/add">Створити запис</a></li>
        </ul>
</header>
<div class="container">
    <form class="edit-form">
        <input type="hidden" id="notationId" name="notationId"
value="{{selectedNotation._id}}">
        <div class="form-group">
            <label for="companyName">Назва Компанії</label>
            <input type="text" id="companyName" name="companyName"
value="{{selectedNotation.companyName}}">
        </div>
        <div class="form-group">
            <label for="numberWorkers">Кількість працівників</label>
```

```
            <input type="text" id="numberWorkers" name="numberWorkers"
pattern="\d+" title="Please enter only numbers"
value="{{selectedNotation.numberWorkers}}">
        </div>
        <div class="form-group">
            <label for="contractName">Контракти</label>
            <input type="text" id="contractName" name="contractName"
value="{{selectedNotation.contractName}}">
        </div>
        <div class="form-group">
            <label for="startDate">Дата початку контракту</label>
            <input type="text" id="startDate" name="startDate" pattern="\d{4}-
\d{2}-\d{2}" title="Please enter date in YYYY-MM-DD format" value="{{dateToStr}}">
        </div>
        <div class="form-group">
            <label for="endDate">Дата закінчення контракту</label>
            <input type="text"id="endDate" name="endDate" pattern="\d{4}-\d{2}-
\d{2}" title="Please enter date in YYYY-MM-DD format" value="{{dateToStr}}">
        </div>
        <button type="button">Застосувати зміни</button>
        <a href="/notes/{{selectedNotation._id}}" class="back-button">Відхилити
зміни</a>
    </form>
</div>
```

Сторінка редагування записів.

index.hbs

```
<header>
        <ul>
                <li><a href="/">Головна сторінка</a></li>
                <li><a href="/notes">Записи</a></li>
                <li><a href="/add">Створити запис</a></li>
        </ul>
</header>
<h1>Лабораторна робота 5, Кравчук Ілля, ІМ-13</h1>
<h2>Варіант 16. Спроектувати базу даних про договори: назва фірми-клієнта, вид
договору, термін дії.</h2>
```

Головна сторінка.

notes.hbs

```
<header>
        <ul>
                <li><a href="/">Головна сторінка</a></li>
                <li><a href="/notes">Записи</a></li>
                <li><a href="/add">Створити запис</a></li>
        </ul>
</header>
<div class="container">
    <div class="notes-list">
        {{#if notation.length}}
```

```
        {{#each notation}}
                <div class="notes">
                <h2>Запис: {{_id}}</h2>
                <a href="/notes/{{_id}}" class="edit-button">Редагувати</a>
                <a href="/delete/{{_id}}" class="delete-button">Видалити</a>
                </div>
        {{/each}}
        {{else}}
            <h1>Записи відсутні!</h1>
        {{/if}}
    </div>
</div>
```

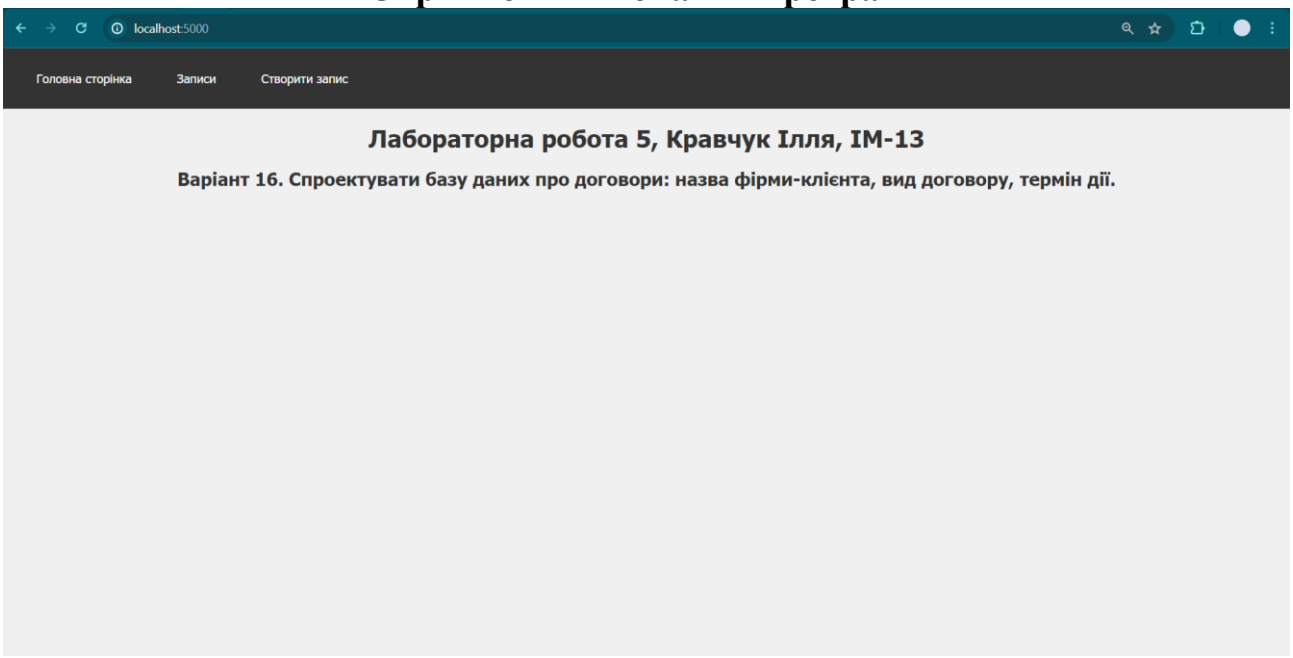
Сторінка перегляду записів.

viewNotation.hbs

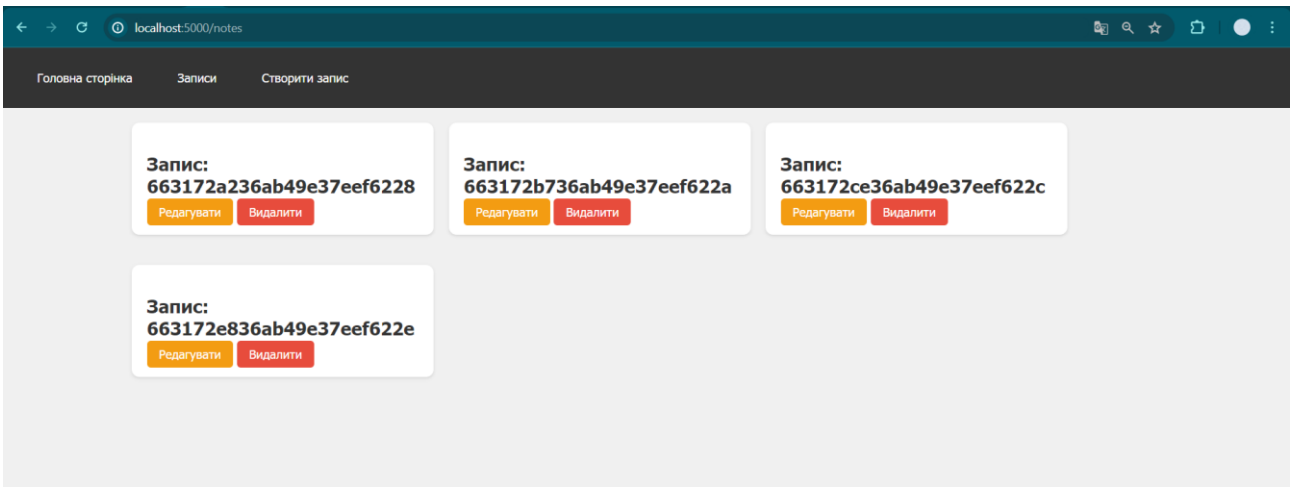
```
<header>
        <ul>
                <li><a href="/">Головна сторінка</a></li>
                <li><a href="/notes">Записи</a></li>
                <li><a href="/add">Створити запис</a></li>
        </ul>
</header>
<div class="container">
   <div class="object-details">
      <h1>Вміст запису</h1>
        <p><strong>ID: </strong>{{selectedNotation._id}}</p>
        <p><strong>Назва Компанії: </strong>{{selectedNotation.companyName}}</p>
        <p><strong>Кількість працівників:
</strong>{{selectedNotation.numberWorkers}}</p>
        <p><strong>Контракти: </strong>{{selectedNotation.contractName}}</p>
        <p><strong>Дата початку контракту:
</strong>{{selectedNotation.startDate}}</p>
        <p><strong>Дата закінчення контракту:
</strong>{{selectedNotation.endDate}}</p>
        <a href="/edit/{{selectedNotation._id}}" class="edit-button">Змінити</a>
    </div>
</div>
```

Сторінка перегляду певного запису.

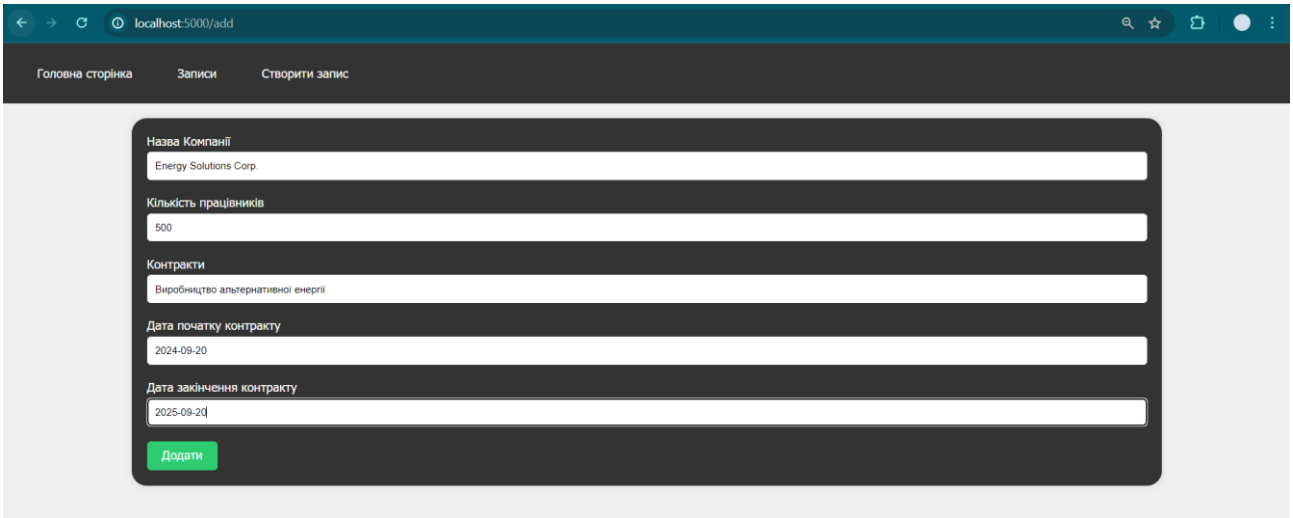
Скріншоти виконання програми



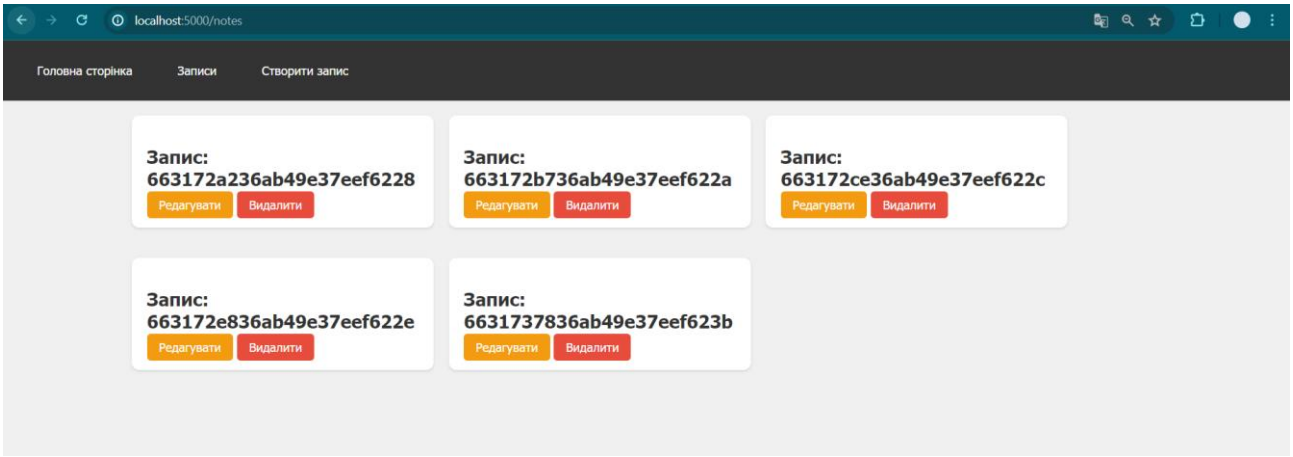
Головна сторінка



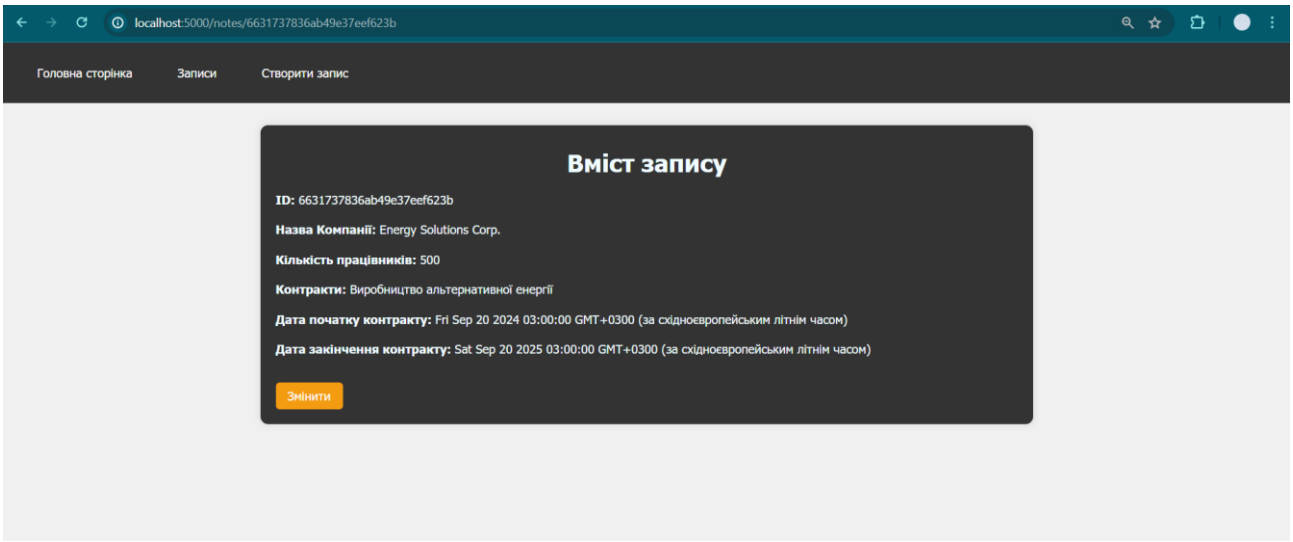
Перегляд записів



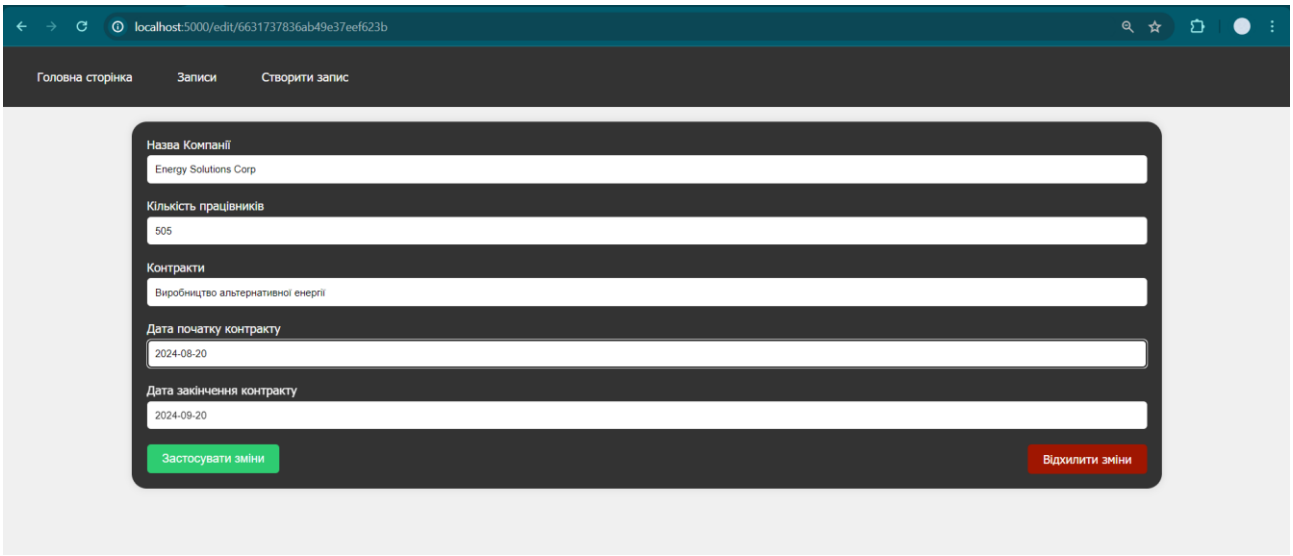
Додавання нового запису



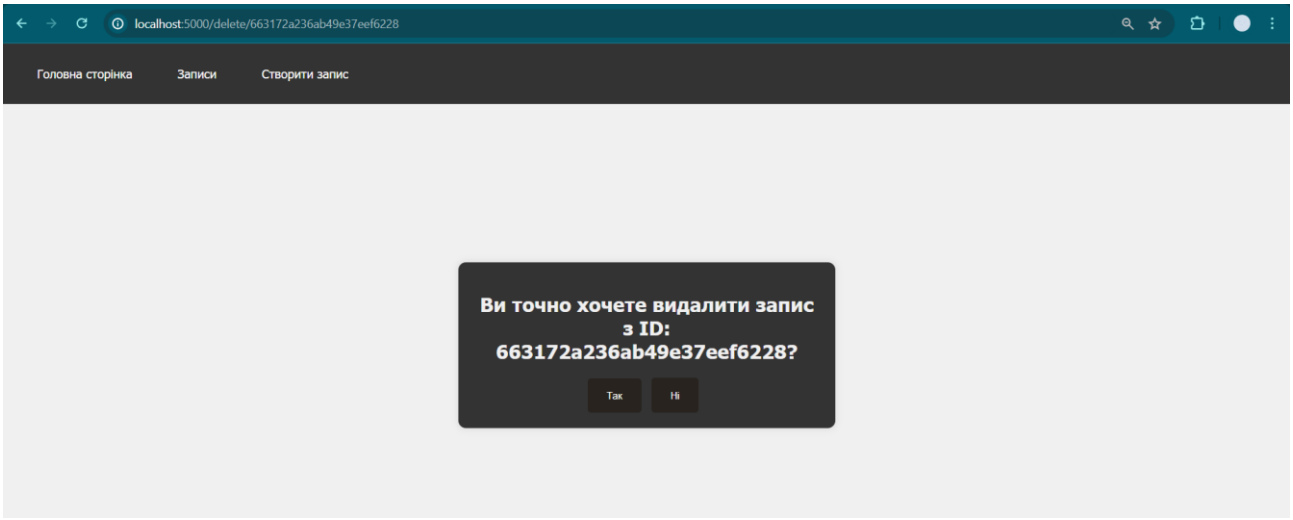
Запис успішно додано!



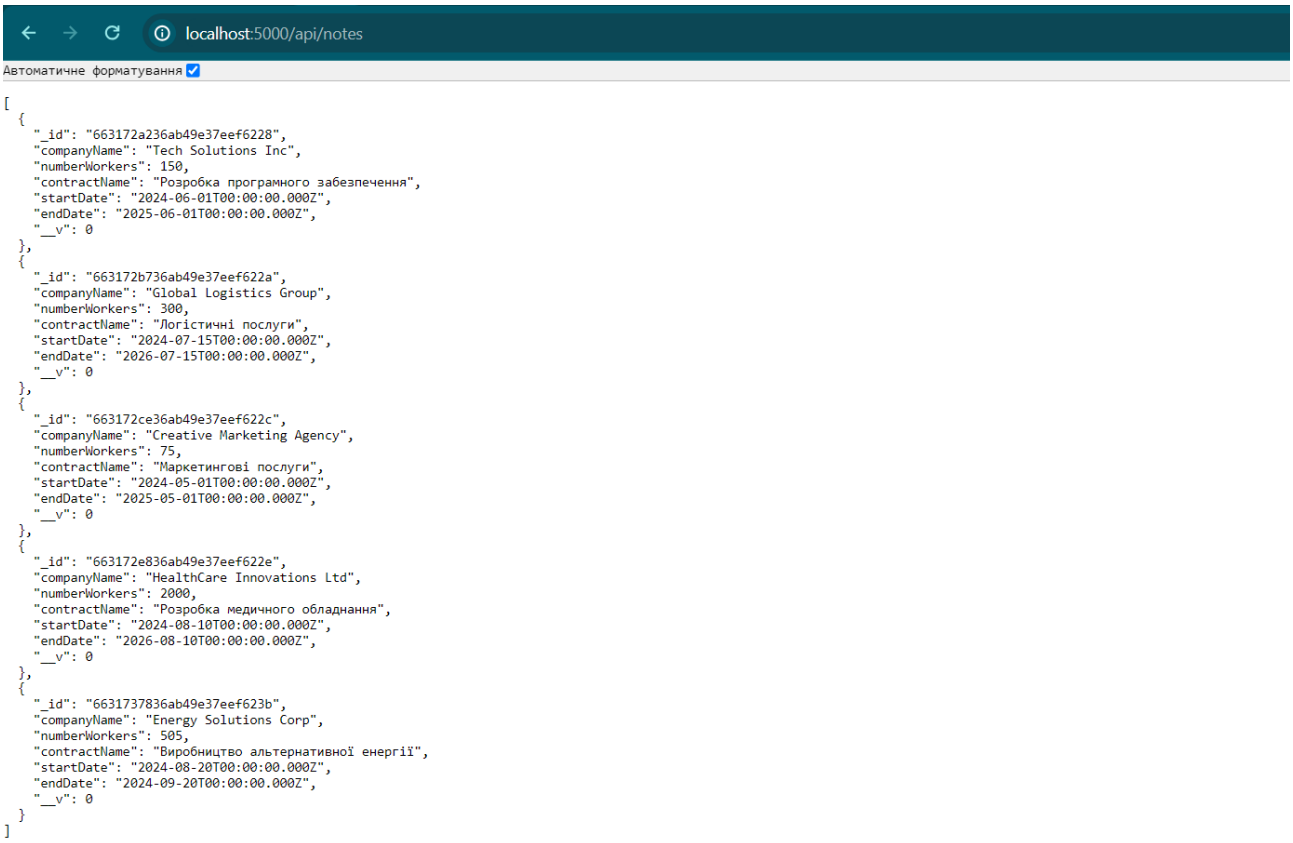
Перегляд запис для редагування



Редагування запису



Видалення запису



Виведення інформації у вигляді json-файлу

Висновок

У результаті лабораторної роботи був розроблений веб-додаток на основі Node.js та Express.js для взаємодії з базою даних MongoDB. Додаток забезпечує можливість виконання CRUD операцій (створення, читання, оновлення, видалення) з записами про договори. Веб-сервер налаштований на обробку HTTP-запитів, а шаблонізатор Handlebars використовується для генерації HTML-сторінок. Маршрутизація запитів виконується за допомогою Express.js, а зв'язок з базою даних MongoDB здійснюється за допомогою Mongoose. Контролери містять методи для обробки різних типів запитів, включаючи

додавання, редагування, видалення та отримання інформації про записи. Результати запитів можуть виводитись як у вигляді HTML-сторінок, так і у форматі JSON. Для взаємодії з користувачем на сторінках використовуються скрипти JavaScript. Таким чином, лабораторна робота успішно виконала поставлені завдання і реалізувала функціональність веб-додатку для роботи з базою даних MongoDB.