Міністерство освіти України

Національний технічний університет "ХПІ"

кафедра "Комп’ютерних наук та інформаційних технологій"

**Звіт**

**Лабораторна робота 6**

з дисципліни "Проєктування серверних застосунків"

Виконав: студент групи КН-922В

Мельник Б. В.

Перевірив:

Івашко А. В.

Харків 2025

**Зміст**

[Завдання 1 4](#_Toc195352472)

[Текст програми 4](#_Toc195352473)

[Результати роботи програми 11](#_Toc195352474)

[Алгоритм побудови кодів 11](#_Toc195352475)

[Функціональність 12](#_Toc195352476)

[Висновок 12](#_Toc195352477)

[Завдання 2 14](#_Toc195352478)

[Текст програми 14](#_Toc195352479)

[Результати роботи програми 20](#_Toc195352480)

[Алгоритм побудови кодів 20](#_Toc195352481)

[Функціональність 21](#_Toc195352482)

[Висновок 22](#_Toc195352483)

[Завдання 3 23](#_Toc195352484)

[Текст програми 23](#_Toc195352485)

[Результати роботи програми 31](#_Toc195352486)

[Алгоритм побудови кодів 32](#_Toc195352487)

[Функціональність 32](#_Toc195352488)

[Висновок 33](#_Toc195352489)

[Завдання 4 34](#_Toc195352490)

[Текст програми 34](#_Toc195352491)

[Результати роботи програми 42](#_Toc195352492)

[Алгоритм побудови кодів 44](#_Toc195352493)

[Функціональність 45](#_Toc195352494)

[Висновок 45](#_Toc195352495)

[Завдання 5 47](#_Toc195352496)

[Текст програми 47](#_Toc195352497)

[Результати роботи програми 55](#_Toc195352498)

[Алгоритм побудови кодів 56](#_Toc195352499)

[Функціональність 57](#_Toc195352500)

[Висновок 57](#_Toc195352501)

[Завдання 6 58](#_Toc195352502)

[Текст програми 58](#_Toc195352503)

[Результати роботи програми 70](#_Toc195352504)

[Алгоритм побудови кодів 71](#_Toc195352505)

[Функціональність 72](#_Toc195352506)

[Висновок 73](#_Toc195352507)

[Завдання 7 74](#_Toc195352508)

[Текст програми 74](#_Toc195352509)

[Результати роботи програми 88](#_Toc195352510)

[Алгоритм побудови кодів 91](#_Toc195352511)

[Функціональність 92](#_Toc195352512)

[Висновок 93](#_Toc195352513)

[Завдання 8 94](#_Toc195352514)

[Текст програми 94](#_Toc195352515)

[Результати роботи програми 109](#_Toc195352516)

[Алгоритм побудови кодів 112](#_Toc195352517)

[Функціональність 113](#_Toc195352518)

[Висновок 113](#_Toc195352519)

[Завдання 9 114](#_Toc195352520)

[Текст програми 114](#_Toc195352521)

[Результати роботи програми 128](#_Toc195352522)

[Алгоритм побудови кодів 131](#_Toc195352523)

[Функціональність 132](#_Toc195352524)

[Висновок 133](#_Toc195352525)

[Загальний висновок 134](#_Toc195352526)

**Тема:** Використання моделей (model) в MVC WEB-застосунку.

**Мета роботи:** придбати практику у створенні та використанні моделей даних під час обробки WEB-форм та застосування різних засобів обробки БД у WEB-застосунках.

**Програмні засоби розробки:** середовище програмування (IDE-Integrated De-velopment Environment) MS Visual Studio, програмний фреймворк ASP.NET версії не нижче 6.0 та дуже бажано використання дизайнерського фреймво-рку BOOTSTRAP, або якогось іншого відповідного аналогу, MySQL Server, або якась аналогічна СУБД.

**Індивідуальні завдання**

**Варіант №12**

|  |  |  |  |
| --- | --- | --- | --- |
| *Відомість суспільного транспорту* | | | |
| Вид транс-  порту | № марш-  руту | Протяжність марш-  Руту (км) | *Час в дорозі (хвтлин)* |
| Тр | 12 | 27.55 | *75* |
| Тл | 17 | 13.6 | *57* |
| А | 12а | 57.3 | *117* |
| *Примітка: Тр – трамвай, Тл – тролейбус, А - автобус* | | | |

# **Завдання 1**

Створити застосунок, який за допомогою слабо типізованої форми (ко-жне поле окрема змінна в контролері), що відповідає моделі читав би дані з відповідних полів форми та демонстрував би їх на тому ж подані нижче.

## **Текст програми:**

**Index.cshtml:**

@model Task\_1.Models.Transport

@{

ViewData["Title"] = "Форма транспорту";

}

<h1>Форма введення даних про транспорт</h1>

<!-- Основна форма з методом POST -->

<form method="post">

<div class="mb-3">

<label for="transportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input type="text" class="form-control" id="transportType" name="transportType" placeholder="Введіть вид транспорту" />

</div>

<div class="mb-3">

<label for="routeNumber" class="form-label">Номер маршруту</label>

<input type="text" class="form-control" id="routeNumber" name="routeNumber" placeholder="Введіть номер маршруту" />

</div>

<div class="mb-3">

<label for="routeDistance" class="form-label">Протяжність маршруту (км)</label>

<input type="number" step="0.01" class="form-control" id="routeDistance" name="routeDistance" placeholder="Введіть протяжність" />

</div>

<div class="mb-3">

<label for="travelTime" class="form-label">Час в дорозі (хв)</label>

<input type="number" class="form-control" id="travelTime" name="travelTime" placeholder="Введіть час в дорозі" />

</div>

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

<!-- Кнопка Скинути для перезавантаження сторінки -->

<button type="button" class="btn btn-secondary" onclick="window.location.href='@Url.Action("Index", "Transport")'">Скинути</button>

</form>

<hr />

@if (Model != null)

{

<div class="result">

<h2>Введені дані:</h2>

<!-- Відображення даних моделі -->

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.TransportType</p>

<p><strong>Номер маршруту:</strong> @Model.RouteNumber</p>

<p><strong>Протяжність маршруту (км):</strong> @Model.RouteDistance</p>

<p><strong>Час в дорозі (хв):</strong> @Model.TravelTime</p>

</div>

}

**Transport.cs:**

using System;

namespace Task\_1.Models

{

public class Transport

{

// Ідентифікатор запису

public int Id { get; set; }

// Вид транспорту (Tr - трамвай, Tl - тролейбус, A - автобус)

public string TransportType { get; set; }

// Номер маршруту (наприклад, "12a")

public string RouteNumber { get; set; }

// Протяжність маршруту в кілометрах

public double RouteDistance { get; set; }

// Час в дорозі (хвилини)

public int TravelTime { get; set; }

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_1.Models;

namespace Task\_1.Controllers

{

public class TransportController : Controller

{

// GET: /Transport/Index - повертає порожню форму

[HttpGet]

public IActionResult Index()

{

return View(); // Повернення форми без даних

}

// POST: /Transport/Index - обробка даних форми

[HttpPost]

public IActionResult Index(string transportType, string routeNumber, double routeDistance, int travelTime)

{

// Створення об'єкта Transport з переданими даними

Transport transport = new Transport

{

Id = 1, // Демонстраційне значення

TransportType = transportType,

RouteNumber = routeNumber,

RouteDistance = routeDistance,

TravelTime = travelTime

};

return View(transport); // Відображення даних користувача

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"] - Task\_1</title>

<link rel="stylesheet" href="~/css/site.css" />

<!-- Підключення Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2vBZX3L9/aqG5Uq4fZAS+2jQ0/1KW8fW/5pXz"

crossorigin="anonymous">

</head>

<body>

<!-- Оверлей для покращення читабельності -->

<div class="overlay">

<div class="container">

<main role="main" class="pb-3">

@RenderBody() <!-- Вбудовування контенту -->

</main>

</div>

</div>

<!-- Підключення Bootstrap JS -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

</body>

</html>

**site.css:**

/\* Основний фон з зображенням \*/

body {

background: url('https://images.pexels.com/photos/2144905/pexels-photo-2144905.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=2') no-repeat center center fixed;

background-size: cover;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

animation: fadeInBackground 3s ease-in-out;

}

/\* Анімація появи фону \*/

@keyframes fadeInBackground {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Оверлей для підвищення читабельності \*/

.overlay {

background-color: rgba(255, 255, 255, 0.2);

min-height: 100vh;

padding: 20px;

}

/\* Відступи для контейнера \*/

.container {

padding-top: 30px;

}

/\* Стилізація заголовків з анімацією bounce \*/

h1, h2 {

color: #333;

text-align: center;

animation: bounce 2s;

}

/\* Анімація bounce \*/

@keyframes bounce {

0%, 20%, 50%, 80%, 100% { transform: translateY(0); }

40% { transform: translateY(-20px); }

60% { transform: translateY(-10px); }

}

/\* Стиль форми з анімацією появи \*/

form {

background: rgba(255, 255, 255, 0.95);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

animation: slideIn 1s ease-out;

}

/\* Анімація слайд‑ін \*/

@keyframes slideIn {

from {

transform: translateY(-20px);

opacity: 0;

}

to {

transform: translateY(0);

opacity: 1;

}

}

/\* Плавний перехід для полів форми \*/

.form-control {

transition: border-color 0.3s, box-shadow 0.3s;

}

.form-control:focus {

border-color: #80bdff;

box-shadow: 0 0 0 0.2rem rgba(0,123,255,0.25);

}

/\* Відступи між елементами форми \*/

.mb-3 {

margin-bottom: 1rem;

}

/\* Стилізація кнопок \*/

.btn-primary {

background-color: #007bff;

border-color: #007bff;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

/\* Стиль блоку з результатами \*/

.result {

color: #fff;

}

/\* Персоналізація заголовка в результатах \*/

.result h2 {

color: #fff !important;

}

## **Результати роботи програми:**

Изображение выглядит как текст, небо, корабль, вода

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.1. Результати роботи виконання Завдання 1

Изображение выглядит как текст, корабль, небо, плавсредство

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.2. Результати роботи виконання Завдання 1

## **Алгоритм побудови кодів**

1. **Створення моделі:**
   * Визначено клас Transport з необхідними властивостями: Id, TransportType, RouteNumber, RouteDistance та TravelTime. Цей клас відповідає структурі даних, що вводяться користувачем.
2. **Розробка представлень (View):**
   * Створено файл Index.cshtml, який містить форму для введення даних про транспорт. Кожне поле форми асоціюється з окремою змінною.
   * В представленні передбачено умову для відображення результатів після успішного введення та відправлення форми.
3. **Реалізація контролера:**
   * Контролер TransportController має два методи:
     + **GET метод:** повертає порожню форму для введення даних.
     + **POST метод:** приймає дані з форми (слабо типізовано – кожне поле передається як окремий параметр), створює об’єкт Transport, заповнює його переданими даними та відправляє назад у представлення для відображення.
4. **Інтеграція з макетом та стилізація:**
   * Файл \_Layout.cshtml забезпечує загальне оформлення сторінки, підключаючи стилі Bootstrap та власний CSS.
   * Файл site.css містить додаткову стилізацію, анімації та налаштування фону для поліпшення користувацького інтерфейсу.

## **Функціональність**

* **Введення даних:**

Користувач має змогу заповнити форму, вказуючи тип транспорту, номер маршруту, протяжність маршруту та час в дорозі.

* **Обробка даних:**

За допомогою POST-запиту дані з форми надходять у метод контролера, де створюється об’єкт моделі Transport із заповненими властивостями.

* **Відображення результату:**

Після обробки даних, застосунок повертає ту ж сторінку із заповненими даними, що дозволяє користувачеві побачити результат введення.

* **Інтерфейс користувача:**

За рахунок використання Bootstrap та кастомного CSS забезпечено чистий, сучасний та анімований інтерфейс, який покращує користувацький досвід.

## **Висновок**

* **Ефективність:**

Розроблений застосунок демонструє базову функціональність роботи з формами в ASP.NET MVC, що включає введення, обробку та відображення даних.

* **Розширюваність:**

Структура додатку дозволяє легко інтегрувати додаткові модулі чи розширювати існуючий функціонал, завдяки чітко розділеним шарам (модель, представлення, контролер).

* **Зручність користування:**

Завдяки застосуванню адаптивної верстки та сучасних стилів (Bootstrap та CSS-анімації) користувач отримує зручний та приємний інтерфейс для взаємодії з системою.

# **Завдання 2**

Виконати попереднє завдання, але замість слабо типізованої форми ви-користати суворо типізовану форму, поля якої відповідають властивостям об'єкту моделі в контролері застосунку.

## **Текст програми:**

**Index.cshtml:**

@model Task\_2.Models.Transport

@{

ViewData["Title"] = "Форма транспорту";

}

<h1>Форма введення даних про транспорт</h1>

<!-- Суворо типізована форма з TagHelper-ами -->

<form asp-action="Index" method="post">

<div class="mb-3">

<label asp-for="TransportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input asp-for="TransportType" class="form-control" placeholder="Введіть вид транспорту" />

</div>

<div class="mb-3">

<label asp-for="RouteNumber" class="form-label">Номер маршруту</label>

<input asp-for="RouteNumber" class="form-control" placeholder="Введіть номер маршруту" />

</div>

<div class="mb-3">

<label asp-for="RouteDistance" class="form-label">Протяжність маршруту (км)</label>

<input asp-for="RouteDistance" class="form-control" placeholder="Введіть протяжність" type="number" step="0.01" />

</div>

<div class="mb-3">

<label asp-for="TravelTime" class="form-label">Час в дорозі (хв)</label>

<input asp-for="TravelTime" class="form-control" placeholder="Введіть час в дорозі" type="number" />

</div>

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

<!-- Скидання форми -->

<button type="button" class="btn btn-secondary" onclick="window.location.href='@Url.Action("Index", "Transport")'">Скинути</button>

</form>

@if (Model != null)

{

<hr />

<div class="result">

<h2>Введені дані:</h2>

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.TransportType</p>

<p><strong>Номер маршруту:</strong> @Model.RouteNumber</p>

<p><strong>Протяжність маршруту (км):</strong> @Model.RouteDistance</p>

<p><strong>Час в дорозі (хв):</strong> @Model.TravelTime</p>

</div>

}

**Transport.cs:**

using System;

namespace Task\_2.Models

{

public class Transport

{

// Ідентифікатор

public int Id { get; set; }

// Вид транспорту (Tr - трамвай, Tl - тролейбус, A - автобус)

public string TransportType { get; set; }

// Номер маршруту

public string RouteNumber { get; set; }

// Протяжність маршруту, км

public double RouteDistance { get; set; }

// Час в дорозі, хв

public int TravelTime { get; set; }

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_2.Models;

namespace Task\_2.Controllers

{

public class TransportController : Controller

{

// GET: /Transport/Index - повертає порожню форму

[HttpGet]

public IActionResult Index()

{

return View();

}

// POST: /Transport/Index - приймає дані форми

[HttpPost]

public IActionResult Index(Transport transport)

{

// Присвоєння демонстраційного Id

transport.Id = 1;

return View(transport);

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"] - Task\_2</title>

<link rel="stylesheet" href="~/css/site.css" />

<!-- Підключення Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2vBZX3L9/aqG5Uq4fZAS+2jQ0/1KW8fW/5pXz"

crossorigin="anonymous">

</head>

<body>

<!-- Оверлей для покращення читабельності -->

<div class="overlay">

<div class="container">

<main role="main" class="pb-3">

@RenderBody() <!-- Вбудовування контенту -->

</main>

</div>

</div>

<!-- Скрипти Bootstrap -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

</body>

</html>

**site.css:**

/\* Фон сторінки з зображенням \*/

body {

background: url('https://images.pexels.com/photos/186537/pexels-photo-186537.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=2') no-repeat center center fixed;

background-size: cover;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

animation: fadeInBackground 3s ease-in-out;

}

/\* Анімація появи фону \*/

@keyframes fadeInBackground {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Оверлей для кращої читабельності \*/

.overlay {

background-color: rgba(255, 255, 255, 0.2);

min-height: 100vh;

padding: 20px;

}

/\* Відступи для контейнера \*/

.container {

padding-top: 30px;

}

/\* Стилізація заголовків з bounce-анімацією \*/

h1, h2 {

color: #333;

text-align: center;

animation: bounce 2s;

}

/\* Анімація для bounce-ефекту \*/

@keyframes bounce {

0%, 20%, 50%, 80%, 100% { transform: translateY(0); }

40% { transform: translateY(-20px); }

60% { transform: translateY(-10px); }

}

/\* Стиль форми з анімацією появи \*/

form {

background: rgba(255, 255, 255, 0.95);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

animation: slideIn 1s ease-out;

}

/\* Ефект слайд‑ін для форми \*/

@keyframes slideIn {

from { transform: translateY(-20px); opacity: 0; }

to { transform: translateY(0); opacity: 1; }

}

/\* Плавний перехід для полів форми \*/

.form-control {

transition: border-color 0.3s, box-shadow 0.3s;

}

.form-control:focus {

border-color: #80bdff;

box-shadow: 0 0 0 0.2rem rgba(0,123,255,0.25);

}

/\* Відступ між блоками \*/

.mb-3 {

margin-bottom: 1rem;

}

/\* Стилізація кнопок \*/

.btn-primary {

background-color: #007bff;

border-color: #007bff;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

/\* Стиль блоку з результатами \*/

.result {

color: #fff;

}

/\* Коригування кольору заголовка в результатах \*/

.result h2 {

color: #fff !important;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, карта

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.1. Результати роботи виконання Завдання 2

Изображение выглядит как текст, снимок экрана, карта

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.1. Результати роботи виконання Завдання 2

## **Алгоритм побудови кодів**

1. **Створення моделі:**
   * Оголошено клас Transport з властивостями: Id, TransportType, RouteNumber, RouteDistance та TravelTime. Ця модель відповідає структурі даних, що використовуються в застосунку.
2. **Розробка суворо типізованої форми:**
   * У файлі Index.cshtml створено форму, що базується на суворій типізації. Використано TagHelper-и для прив’язки полів форми до відповідних властивостей моделі (asp-for).
   * Форма автоматично генерує елементи вводу (input) для кожного параметра моделі, що спрощує валідацію даних та зв’язування їх з моделлю.
3. **Реалізація контролера:**
   * Контролер TransportController містить два методи:
     + **GET метод:** повертає порожню форму, готову для введення даних.
     + **POST метод:** приймає об’єкт моделі Transport як параметр завдяки механізму автоматичного зв’язування даних. Після присвоєння демонстраційного значення властивості Id об’єкт повертається у представлення для відображення введених даних.
4. **Інтеграція з макетом та оформлення:**
   * Файл \_Layout.cshtml задає загальний вигляд сторінки, включаючи використання Bootstrap 5 і кастомних стилів у файлі site.css для створення сучасного інтерфейсу з анімаціями.

## **Функціональність**

* **Автоматичне зв’язування даних:**
  + Завдяки суворій типізації, дані, введені користувачем, автоматично зв’язуються з властивостями моделі Transport через TagHelper-и, що спрощує обробку і зменшує ризик помилок.
* **Обробка і відображення даних:**
  + Після натискання кнопки «Надіслати» дані передаються до POST‑методу контролера, де здійснюється їх обробка і демонстрація на тій же сторінці, що дозволяє користувачеві одразу бачити введену інформацію.
* **Зручний інтерфейс:**
  + Використання Bootstrap і сучасних CSS‑анимацій створює привабливе та інтуїтивно зрозуміле середовище для введення даних

## **Висновок**

* **Покращена надійність:**

Завдяки суворій типізації форми, використовується автоматичне зв’язування даних, що зменшує кількість помилок при передачі інформації між виглядом і контролером.

* **Простота підтримки:**

Структура застосунку, де модель, представлення та контролер чітко розділені, сприяє зрозумілості коду та його розширеності у майбутньому.

* **Покращений користувацький досвід:**

Сучасна верстка і використання TagHelper-ів спрощують роботу з формою, роблячи інтерфейс більш дружнім до користувача та ефективним у взаємодії.

# **Завдання 3**

Модифікувати попереднє завдання додавши до форми кнопку додання даних до таблиці відповідної MySQL БД з подальшою демонстрацією вмісту цієї таблиці під формою подання.

## **Текст програми:**

**SQL-скрипт:**

USE TransportDB;

CREATE TABLE Transports (

Id INT IDENTITY(1,1) PRIMARY KEY,

TransportType NVARCHAR(50) NOT NULL,

RouteNumber NVARCHAR(50) NOT NULL,

RouteDistance FLOAT NOT NULL,

TravelTime INT NOT NULL

);

**Index.cshtml:**

@model Task\_3.Models.Transport

@{

ViewData["Title"] = "Форма транспорту";

var transports = ViewBag.Transports as List<Task\_3.Models.Transport>; // Отримання списку записів з ViewBag

}

<h1>Форма введення даних про транспорт</h1>

<!-- Суворо типізована форма з використанням TagHelper-ів -->

<form asp-action="Index" method="post">

<div class="mb-3">

<label asp-for="TransportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input asp-for="TransportType" class="form-control" placeholder="Введіть вид транспорту" />

</div>

<div class="mb-3">

<label asp-for="RouteNumber" class="form-label">Номер маршруту</label>

<input asp-for="RouteNumber" class="form-control" placeholder="Введіть номер маршруту" />

</div>

<div class="mb-3">

<label asp-for="RouteDistance" class="form-label">Протяжність маршруту (км)</label>

<input asp-for="RouteDistance" class="form-control" placeholder="Введіть протяжність" type="number" step="0.01" />

</div>

<div class="mb-3">

<label asp-for="TravelTime" class="form-label">Час в дорозі (хв)</label>

<input asp-for="TravelTime" class="form-control" placeholder="Введіть час в дорозі" type="number" />

</div>

<button type="submit" class="btn btn-primary">Додати до БД</button>

<!-- Скидання форми через перезавантаження сторінки -->

<button type="button" class="btn btn-secondary" onclick="window.location.href='@Url.Action("Index", "Transport")'">Скинути</button>

</form>

@if (transports != null && transports.Any())

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<!-- Таблиця з даними -->

<table class="table custom-bordered-table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність маршруту (км)</th>

<th>Час в дорозі (хв)</th>

</tr>

</thead>

<tbody>

@foreach (var item in transports)

{

<tr>

<td>@item.Id</td>

<td>@item.TransportType</td>

<td>@item.RouteNumber</td>

<td>@item.RouteDistance</td>

<td>@item.TravelTime</td>

</tr>

}

</tbody>

</table>

</div>

}

**Transport.cs:**

namespace Task\_3.Models

{

public class Transport

{

// ID запису (генерується автоматично в БД)

public int Id { get; set; }

// Вид транспорту (Tr, Tl, A)

public string TransportType { get; set; }

// Номер маршруту, напр. "12a"

public string RouteNumber { get; set; }

// Протяжність маршруту в км

public double RouteDistance { get; set; }

// Час в дорозі в хвилинах

public int TravelTime { get; set; }

}

}

**TransportContext.cs:**

using Microsoft.EntityFrameworkCore;

namespace Task\_3.Models

{

public class TransportContext : DbContext

{

public TransportContext(DbContextOptions<TransportContext> options)

: base(options)

{

}

// Таблиця з транспортними записами

public DbSet<Transport> Transports { get; set; }

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_3.Models;

using System.Linq;

namespace Task\_3.Controllers

{

public class TransportController : Controller

{

private readonly TransportContext \_context;

public TransportController(TransportContext context)

{

\_context = context; // Ініціалізація контексту бази даних

}

// GET: /Transport/Index - показує форму і таблицю записів

[HttpGet]

public IActionResult Index()

{

var model = new Transport(); // Порожня модель для форми

var transports = \_context.Transports.ToList(); // Отримання записів з БД

ViewBag.Transports = transports;

return View(model);

}

// POST: /Transport/Index - додає новий запис до БД

[HttpPost]

public IActionResult Index(Transport transport)

{

\_context.Transports.Add(transport); // Додавання запису

\_context.SaveChanges(); // Збереження змін

var transports = \_context.Transports.ToList(); // Оновлення списку записів

ViewBag.Transports = transports;

return View(transport);

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"] - Task\_3</title>

<link rel="stylesheet" href="~/css/site.css" />

<!-- Підключення Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2vBZX3L9/aqG5Uq4fZAS+2jQ0/1KW8fW/5pXz"

crossorigin="anonymous">

</head>

<body>

<!-- Оверлей для зручного перегляду -->

<div class="overlay">

<div class="container">

<main role="main" class="pb-3">

@RenderBody() <!-- Основний контент сторінки -->

</main>

</div>

</div>

<!-- Скрипти Bootstrap -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

</body>

</html>

**site.css:**

/\* Фон сторінки із зображенням \*/

body {

background: url('https://www.hdwallpapers.in/download/usa\_vermont\_forest\_with\_house\_and\_road\_between\_field\_and\_trees\_under\_cloudy\_sky\_during\_fall\_hd\_nature-1920x1080.jpg') no-repeat center center fixed;

background-size: cover;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

animation: fadeInBackground 3s ease-in-out;

}

/\* Анімація появи фону \*/

@keyframes fadeInBackground {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Оверлей для покращення читаємості \*/

.overlay {

background-color: rgba(255, 255, 255, 0.2);

min-height: 100vh;

padding: 20px;

}

/\* Відступи для контейнера \*/

.container {

padding-top: 30px;

}

/\* Стилізація заголовків із bounce-анімацією \*/

h1, h2 {

color: #333;

text-align: center;

animation: bounce 2s;

}

@keyframes bounce {

0%, 20%, 50%, 80%, 100% { transform: translateY(0); }

40% { transform: translateY(-20px); }

60% { transform: translateY(-10px); }

}

/\* Стилізація форми з анімацією появи \*/

form {

background: rgba(255, 255, 255, 0.95);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

animation: slideIn 1s ease-out;

}

@keyframes slideIn {

from { transform: translateY(-20px); opacity: 0; }

to { transform: translateY(0); opacity: 1; }

}

/\* Плавний перехід для полів форми \*/

.form-control {

transition: border-color 0.3s, box-shadow 0.3s;

}

.form-control:focus {

border-color: #80bdff;

box-shadow: 0 0 0 0.2rem rgba(0,123,255,0.25);

}

/\* Відступи між блоками форми \*/

.mb-3 {

margin-bottom: 1rem;

}

/\* Стилізація кнопок \*/

.btn-primary {

background-color: #007bff;

border-color: #007bff;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

/\* Стилізація блоку з результатами \*/

.result {

text-align: center;

}

.result h2 {

color: #fff !important;

}

/\* Таблиця з білими границями та текстом \*/

.custom-bordered-table {

border-collapse: collapse;

margin: 0 auto;

background-color: rgba(0, 0, 0, 0.5);

color: #fff;

width: 80%;

}

.custom-bordered-table th,

.custom-bordered-table td {

border: 1px solid #fff !important;

padding: 8px;

}

.custom-bordered-table th {

font-weight: bold;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.1. Результати роботи виконання Завдання 3 (Створена БД)

Изображение выглядит как текст, дерево, снимок экрана, небо

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.2. Результати роботи виконання Завдання 3

Изображение выглядит как текст, дерево, снимок экрана, небо

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.3. Результати роботи виконання Завдання 3

## **Алгоритм побудови кодів**

1. **Налаштування бази даних:**
   * Запуск SQL-скрипта для створення БД TransportDB та таблиці **Transports** з необхідними полями, де поле **Id** генерується автоматично.
2. **Створення моделі:**
   * Оголошено клас Transport з властивостями: Id, TransportType, RouteNumber, RouteDistance та TravelTime, що відповідають структурі таблиці БД.
3. **Налаштування контексту даних:**
   * Реалізовано клас TransportContext на основі DbContext, який містить DbSet<Transport> для взаємодії із записами таблиці.
4. **Розробка представлення (View):**
   * В Index.cshtml використано суворо типізовану форму з TagHelper-ами для введення даних. Додано кнопку «Додати до БД», яка у POST‑методі контролера зберігає новий запис.
   * Після форми розміщено таблицю для демонстрації вмісту таблиці **Transports**, дані якої передаються через ViewBag.
5. **Реалізація контролера:**
   * Контролер TransportController містить:
     + **GET метод:** Створює порожню модель і отримує список записів із БД, що передається через ViewBag до представлення.
     + **POST метод:** Приймає модель, додає новий запис до БД та зберігає зміни, після чого оновлюється список записів для відображення.

## **Функціональність**

* **Введення та збереження даних:**
  + Суворо типізована форма дозволяє користувачу вводити дані про транспорт, які автоматично зв’язуються з моделлю Transport.
  + Після натискання кнопки «Додати до БД» введені дані зберігаються в таблиці MySQL, що забезпечує збереження інформації.
* **Динамічне відображення записів:**
  + Дані, що збережені в БД, отримуються за допомогою Entity Framework і відображаються у вигляді таблиці під формою. Це дозволяє користувачеві бачити поточний вміст таблиці та перевіряти успішність додавання нового запису.
* **Інтуїтивний інтерфейс:**
  + Використання Bootstrap і CSS-анімацій забезпечує сучасний та зручний дизайн сторінки, що сприяє кращій взаємодії користувача із застосунком.

## **Висновок**

* **Розширення функціоналу:**
  + Додавання можливості збереження даних у MySQL БД значно розширює можливості застосунку. Тепер користувач може не тільки вводити та переглядати інформацію, а й зберігати історію транспортних записів.
* **Підвищення надійності та гнучкості:**
  + Використання Entity Framework дозволяє спростити роботу із базою даних, забезпечує автоматичне управління зв’язуванням даних і полегшує подальше масштабування застосунку.
* **Оптимальний користувацький досвід:**
  + Зручний інтерфейс з суворо типізованою формою і динамічним відображенням даних гарантує інтуїтивну роботу користувача із застосунком, сприяючи оперативному перегляду результатів внесених змін.

# **Завдання 4**

Модифікувати попереднє завдання додавши до форми кнопку видалення усіх записів таблиці відповідної MySQL БД з подальшою демонстрацією порожньої таблиці під формою подання.

## **Текст програми:**

**Index.cshtml:**

@model Task\_4.Models.Transport

@{

ViewData["Title"] = "Форма транспорту";

var transports = ViewBag.Transports as List<Task\_4.Models.Transport>; // Отримуємо список записів

}

<h1>Форма введення даних про транспорт</h1>

<!-- Форма для додавання запису -->

<form asp-action="Index" method="post">

<div class="mb-3">

<label asp-for="TransportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input asp-for="TransportType" class="form-control" placeholder="Введіть вид транспорту" />

</div>

<div class="mb-3">

<label asp-for="RouteNumber" class="form-label">Номер маршруту</label>

<input asp-for="RouteNumber" class="form-control" placeholder="Введіть номер маршруту" />

</div>

<div class="mb-3">

<label asp-for="RouteDistance" class="form-label">Протяжність маршруту (км)</label>

<input asp-for="RouteDistance" class="form-control" placeholder="Введіть протяжність" type="number" step="0.01" />

</div>

<div class="mb-3">

<label asp-for="TravelTime" class="form-label">Час в дорозі (хв)</label>

<input asp-for="TravelTime" class="form-control" placeholder="Введіть час в дорозі" type="number" />

</div>

<button type="submit" class="btn btn-primary">Додати до БД</button>

<!-- Скидання форми шляхом перезавантаження -->

<button type="button" class="btn btn-secondary" onclick="window.location.href='@Url.Action("Index", "Transport")'">Скинути</button>

</form>

<!-- Форма для видалення всіх записів -->

<form asp-action="DeleteAll" method="post" style="margin-top: 15px;">

<button type="submit" class="btn btn-danger">Видалити всі записи</button>

</form>

@if (transports != null && transports.Any())

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<!-- Таблиця з даними -->

<table class="table custom-bordered-table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність маршруту (км)</th>

<th>Час в дорозі (хв)</th>

</tr>

</thead>

<tbody>

@foreach (var item in transports)

{

<tr>

<td>@item.Id</td>

<td>@item.TransportType</td>

<td>@item.RouteNumber</td>

<td>@item.RouteDistance</td>

<td>@item.TravelTime</td>

</tr>

}

</tbody>

</table>

</div>

}

else

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<p>Таблиця порожня.</p>

</div>

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_4.Models;

using System.Linq;

namespace Task\_4.Controllers

{

public class TransportController : Controller

{

private readonly TransportContext \_context;

public TransportController(TransportContext context)

{

\_context = context; // Ініціалізація контексту даних

}

// GET: /Transport/Index

[HttpGet]

public IActionResult Index()

{

var model = new Transport(); // Порожня модель для форми

var transports = \_context.Transports.ToList(); // Завантаження даних з БД

ViewBag.Transports = transports;

return View(model);

}

// POST: /Transport/Index - Додавання нового запису

[HttpPost]

public IActionResult Index(Transport transport)

{

\_context.Transports.Add(transport); // Додавання запису до БД

\_context.SaveChanges(); // Збереження змін

var transports = \_context.Transports.ToList(); // Оновлення даних

ViewBag.Transports = transports;

return View(transport);

}

// POST: /Transport/DeleteAll - Видалення всіх записів

[HttpPost]

public IActionResult DeleteAll()

{

\_context.Transports.RemoveRange(\_context.Transports); // Видалення всіх записів

\_context.SaveChanges(); // Фіксація змін

var model = new Transport(); // Порожня модель після видалення

ViewBag.Transports = \_context.Transports.ToList(); // Завантаження порожнього списку

return View("Index", model);

}

}

}

**Transport.cs:**

namespace Task\_4.Models

{

public class Transport

{

// ID запису (генерується в базі)

public int Id { get; set; }

// Вид транспорту (Tr, Tl, A)

public string TransportType { get; set; }

// Номер маршруту

public string RouteNumber { get; set; }

// Протяжність маршруту (км)

public double RouteDistance { get; set; }

// Час в дорозі (хв)

public int TravelTime { get; set; }

}

}

**TransportContext.cs:**

using Microsoft.EntityFrameworkCore;

using System.Collections.Generic;

namespace Task\_4.Models

{

public class TransportContext : DbContext

{

public TransportContext(DbContextOptions<TransportContext> options)

: base(options)

{

}

// Таблиця для зберігання записів транспорту

public DbSet<Transport> Transports { get; set; }

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"] - Task\_4</title>

<link rel="stylesheet" href="~/css/site.css" />

<!-- Підключення Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2vBZX3L9/aqG5Uq4fZAS+2jQ0/1KW8fW/5pXz"

crossorigin="anonymous">

</head>

<body>

<!-- Контейнер для основного контенту -->

<div class="overlay">

<div class="container">

<main role="main" class="pb-3">

@RenderBody() <!-- Основна частина сторінки -->

</main>

</div>

</div>

<!-- Скрипти Bootstrap -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

</body>

</html>

**site.css:**

/\* Основний фон сторінки \*/

body {

background: url('https://www.hdwallpapers.in/download/usa\_vermont\_forest\_with\_house\_and\_road\_between\_field\_and\_trees\_under\_cloudy\_sky\_during\_fall\_hd\_nature-1920x1080.jpg') no-repeat center center fixed;

background-size: cover;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

animation: fadeInBackground 3s ease-in-out;

}

/\* Анімація появи фону \*/

@keyframes fadeInBackground {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Оверлей для покращення читабельності \*/

.overlay {

background-color: rgba(255, 255, 255, 0.2);

min-height: 100vh;

padding: 20px;

}

/\* Відступи зверху для контейнера \*/

.container {

padding-top: 30px;

}

/\* Стилізація заголовків з bounce-анімацією \*/

h1, h2 {

color: #333;

text-align: center;

animation: bounce 2s;

}

@keyframes bounce {

0%, 20%, 50%, 80%, 100% { transform: translateY(0); }

40% { transform: translateY(-20px); }

60% { transform: translateY(-10px); }

}

/\* Стиль форми з анімацією появи \*/

form {

background: rgba(255, 255, 255, 0.95);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

animation: slideIn 1s ease-out;

}

@keyframes slideIn {

from { transform: translateY(-20px); opacity: 0; }

to { transform: translateY(0); opacity: 1; }

}

/\* Плавні переходи для полів форми \*/

.form-control {

transition: border-color 0.3s, box-shadow 0.3s;

}

.form-control:focus {

border-color: #80bdff;

box-shadow: 0 0 0 0.2rem rgba(0,123,255,0.25);

}

/\* Відступи між блоками форми \*/

.mb-3 {

margin-bottom: 1rem;

}

/\* Стилізація кнопок \*/

.btn-primary {

background-color: #007bff;

border-color: #007bff;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

.btn-danger {

background-color: #dc3545;

border-color: #dc3545;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-danger:hover {

background-color: #bb2d3b;

border-color: #b02a37;

}

/\* Стилізація блоку для таблиці \*/

.result {

text-align: center;

}

.result h2 {

color: #fff !important;

}

/\* Таблиця з білими границями та текстом \*/

.custom-bordered-table {

border-collapse: collapse;

margin: 0 auto; /\* Центрування таблиці \*/

background-color: rgba(0, 0, 0, 0.5); /\* Напівпрозорий темний фон \*/

color: #fff;

width: 80%;

}

.custom-bordered-table th,

.custom-bordered-table td {

border: 1px solid #fff !important;

padding: 8px;

}

.custom-bordered-table th {

font-weight: bold;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.1. Результати роботи виконання Завдання 4 (Створена БД)

Изображение выглядит как текст, снимок экрана, небо, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.2. Результати роботи виконання Завдання 4

Изображение выглядит как текст, снимок экрана, небо, трава

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.3. Результати роботи виконання Завдання 4

Изображение выглядит как текст, небо, снимок экрана, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.4. Результати роботи виконання Завдання 4

## **Алгоритм побудови кодів**

1. **Налаштування бази даних та моделі:**
   * Використовується SQL‑скрипт для створення бази даних (TransportDB) та таблиці **Transports** з полями: Id, TransportType, RouteNumber, RouteDistance та TravelTime.
   * Оголошено клас Transport, що відображає структуру таблиці, де Id генерується автоматично базою даних.
2. **Налаштування контексту даних:**
   * Створено клас TransportContext, який наслідує від DbContext. Він містить DbSet<Transport>, що дозволяє виконувати операції з таблицею через Entity Framework.
3. **Розробка представлення (View):**
   * Файл Index.cshtml містить дві форми:
     + Перша форма – суворо типізована, з використанням TagHelper‑ів для додавання нового запису до бази.
     + Друга форма – для видалення всіх записів з таблиці (дія DeleteAll).
   * Після форм розміщується блок, в якому перевіряється наявність записів. Якщо записи присутні, вони виводяться у вигляді таблиці; інакше виводиться повідомлення «Таблиця порожня».
4. **Реалізація контролера:**
   * **GET метод Index:** Створює порожню модель для форми, завантажує всі записи з бази та передає їх через ViewBag у вигляді списку.
   * **POST метод Index:** Приймає модель із форми, додає новий запис до бази та оновлює список записів.
   * **POST метод DeleteAll:** Видаляє всі записи з таблиці за допомогою методу RemoveRange і викликає збереження змін, після чого повертає представлення з порожнім списком.

## **Функціональність**

* **Додавання даних:**

Суворо типізована форма дозволяє користувачу вводити дані про транспорт, які автоматично зв’язуються з моделлю та зберігаються у MySQL базі даних.

* **Видалення всіх записів:**

Додана кнопка «Видалити всі записи» запускає спеціальний POST‑метод, що очищує таблицю в базі. Після видалення користувач бачить повідомлення про те, що таблиця порожня.

* **Динамічне відображення результатів:**

Після кожної операції (додавання або видалення) контролер завантажує актуальний список записів із бази даних, який виводиться під формою. Це дозволяє користувачу в режимі реального часу перевірити успішність виконання операції.

* **Інтуїтивний інтерфейс:**

Завдяки використанню Bootstrap і кастомних стилів (site.css), інтерфейс застосунку є сучасним і зручним, що покращує взаємодію користувача з системою.

## **Висновок**

* **Розширення функціоналу:**

Додавання можливості видалення всіх записів дозволяє демонструвати не лише операцію додавання даних, але й управління вмістом БД, що робить застосунок більш гнучким та динамічним.

* **Модульність та підтримуваність:**

Чітке розділення логіки між моделлю, контекстом даних, контролером і представленням сприяє легшій підтримці та можливому розширенню застосунку у майбутньому.

* **Покращений користувацький досвід:**

Динамічне оновлення відображення записів у таблиці дозволяє користувачу одразу бачити результати своїх дій – після додавання або видалення даних. Це підвищує інтерактивність застосунку та сприяє комфортному використанню.

# **Завдання 5**

Модифікувати попереднє завдання додавши до форми засоби видалення бажаного запису з таблиці відповідної MySQL БД з подальшою демонстрацією вмісту таблиці без видаленого запису під формою по-дання.

## **Текст програми:**

**Index.cshtml:**

@model Task\_5.Models.Transport

@{

ViewData["Title"] = "Форма транспорту";

// Отримуємо список записів з ViewBag

var transports = ViewBag.Transports as List<Task\_5.Models.Transport>;

}

<h1>Форма введення даних про транспорт</h1>

<!-- Форма для додавання запису -->

<form asp-action="Index" method="post">

<div class="mb-3">

<label asp-for="TransportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input asp-for="TransportType" class="form-control" placeholder="Введіть вид транспорту" />

</div>

<div class="mb-3">

<label asp-for="RouteNumber" class="form-label">Номер маршруту</label>

<input asp-for="RouteNumber" class="form-control" placeholder="Введіть номер маршруту" />

</div>

<div class="mb-3">

<label asp-for="RouteDistance" class="form-label">Протяжність маршруту (км)</label>

<input asp-for="RouteDistance" class="form-control" placeholder="Введіть протяжність" type="number" step="0.01" />

</div>

<div class="mb-3">

<label asp-for="TravelTime" class="form-label">Час в дорозі (хв)</label>

<input asp-for="TravelTime" class="form-control" placeholder="Введіть час в дорозі" type="number" />

</div>

<button type="submit" class="btn btn-primary">Додати до БД</button>

<!-- Скидання форми шляхом перезавантаження сторінки -->

<button type="button" class="btn btn-secondary" onclick="window.location.href='@Url.Action("Index", "Transport")'">Скинути</button>

</form>

<!-- Форма для видалення всіх записів -->

<form asp-action="DeleteAll" method="post" style="margin-top: 15px;">

<button type="submit" class="btn btn-danger">Видалити всі записи</button>

</form>

@if (transports != null && transports.Any())

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<table class="table custom-bordered-table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність маршруту (км)</th>

<th>Час в дорозі (хв)</th>

<th>Дії</th>

</tr>

</thead>

<tbody>

@foreach (var item in transports)

{

<tr>

<td>@item.Id</td>

<td>@item.TransportType</td>

<td>@item.RouteNumber</td>

<td>@item.RouteDistance</td>

<td>@item.TravelTime</td>

<td>

<!-- Форма для видалення конкретного запису -->

<form asp-action="DeleteRecord" method="post" class="inline-form">

<input type="hidden" name="id" value="@item.Id" />

<button type="submit" class="btn-delete">Видалити</button>

</form>

</td>

</tr>

}

</tbody>

</table>

</div>

}

else

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<p>Таблиця порожня.</p>

</div>

}

**TransportContext.cs:**

using Microsoft.EntityFrameworkCore;

using System.Collections.Generic;

namespace Task\_5.Models

{

public class TransportContext : DbContext

{

public TransportContext(DbContextOptions<TransportContext> options)

: base(options)

{

}

// Таблиця для записів транспорту

public DbSet<Transport> Transports { get; set; }

}

}

**Transport.cs:**

namespace Task\_5.Models

{

public class Transport

{

// ID (генерується автоматично)

public int Id { get; set; }

// Вид транспорту (наприклад, Tr, Tl, A)

public string TransportType { get; set; }

// Номер маршруту, напр. "12a"

public string RouteNumber { get; set; }

// Протяжність маршруту (км)

public double RouteDistance { get; set; }

// Час в дорозі (хв)

public int TravelTime { get; set; }

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"] - Task\_5</title>

<link rel="stylesheet" href="~/css/site.css" />

<!-- Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2vBZX3L9/aqG5Uq4fZAS+2jQ0/1KW8fW/5pXz"

crossorigin="anonymous">

</head>

<body>

<!-- Обгортка для основного контенту -->

<div class="overlay">

<div class="container">

<main role="main" class="pb-3">

@RenderBody() <!-- Рендеринг основного контенту -->

</main>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

</body>

</html>

**site.css:**

/\* Фон сторінки із зображенням \*/

body {

background: url('https://www.hdwallpapers.in/download/usa\_vermont\_forest\_with\_house\_and\_road\_between\_field\_and\_trees\_under\_cloudy\_sky\_during\_fall\_hd\_nature-1920x1080.jpg') no-repeat center center fixed;

background-size: cover;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

animation: fadeInBackground 3s ease-in-out;

}

@keyframes fadeInBackground {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Оверлей для підвищення читабельності \*/

.overlay {

background-color: rgba(255, 255, 255, 0.2);

min-height: 100vh;

padding: 20px;

}

/\* Відступи для контейнера \*/

.container {

padding-top: 30px;

}

/\* Заголовки з bounce-анімацією \*/

h1, h2 {

color: #333;

text-align: center;

animation: bounce 2s;

}

@keyframes bounce {

0%, 20%, 50%, 80%, 100% { transform: translateY(0); }

40% { transform: translateY(-20px); }

60% { transform: translateY(-10px); }

}

/\* Стиль форми з ефектом появи \*/

form {

background: rgba(255, 255, 255, 0.95);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

animation: slideIn 1s ease-out;

margin-bottom: 0;

}

@keyframes slideIn {

from { transform: translateY(-20px); opacity: 0; }

to { transform: translateY(0); opacity: 1; }

}

/\* Плавний перехід для полів форми \*/

.form-control {

transition: border-color 0.3s, box-shadow 0.3s;

}

.form-control:focus {

border-color: #80bdff;

box-shadow: 0 0 0 0.2rem rgba(0,123,255,0.25);

}

/\* Відступи між блоками \*/

.mb-3 {

margin-bottom: 1rem;

}

/\* Стили кнопок \*/

.btn-primary {

background-color: #007bff;

border-color: #007bff;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

.btn-secondary {

background-color: #6c757d;

border-color: #6c757d;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-secondary:hover {

background-color: #5a6268;

border-color: #545b62;

}

.btn-danger {

background-color: #dc3545;

border-color: #dc3545;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-danger:hover {

background-color: #bb2d3b;

border-color: #b02a37;

}

/\* Блок результату \*/

.result {

text-align: center;

}

.result h2 {

color: #fff !important;

}

/\* Таблиця з білими границями і текстом \*/

.custom-bordered-table {

border-collapse: collapse;

margin: 0 auto;

background-color: rgba(0, 0, 0, 0.5);

color: #fff;

width: 80%;

}

.custom-bordered-table th,

.custom-bordered-table td {

border: 1px solid #fff !important;

padding: 8px;

vertical-align: middle;

text-align: center;

}

.custom-bordered-table th {

font-weight: bold;

}

/\* Кнопка для видалення запису \*/

.btn-delete {

background-color: #dc3545;

border: 1px solid #dc3545;

color: #fff;

border-radius: 12px;

padding: 6px 12px;

font-size: 0.875rem;

cursor: pointer;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-delete:hover {

background-color: #bb2d3b;

border-color: #b02a37;

}

/\* Inline форма для окремого запису \*/

.inline-form {

display: inline-block;

margin: 0;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5.1. Результати роботи виконання Завдання 5 (Створена БД)

Изображение выглядит как текст, снимок экрана, небо, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5.2. Результати роботи виконання Завдання 5

Изображение выглядит как текст, снимок экрана, небо, облако

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 5.3. Результати роботи виконання Завдання 5

## **Алгоритм побудови кодів**

1. **Налаштування моделі та контексту даних:**
   * Створюється клас **Transport** з властивостями: Id, TransportType, RouteNumber, RouteDistance і TravelTime, що ідентифікують кожен запис.
   * Реалізовано клас **TransportContext** (усередині якого використовується DbSet для таблиці **Transports**) для роботи з базою даних через Entity Framework.
2. **Розробка представлення (View):**
   * У файлі **Index.cshtml** створено форму для додавання запису із використанням TagHelper‑ів.
   * Виводиться таблиця з поточними записами із БД, де для кожного рядка додається окрема форма з кнопкою «Видалити», яка передає значення Id конкретного запису через прихований елемент.
3. **Реалізація контролера:**
   * **GET метод (Index):** Завантажує порожню модель для форми та список записів із БД (передається через ViewBag для відображення).
   * **POST метод (Index):** Приймає дані з форми, додає новий запис до БД і оновлює список записів.
   * **POST метод (DeleteRecord):** Приймає ідентифікатор запису, який потрібно видалити, знаходить відповідний елемент, видаляє його з БД, фіксує зміни і повертає оновлений список записів на представлення.

## **Функціональність**

* **Додавання записів:**

Завдяки суворо типізованій формі користувач може вводити дані про транспорт, які автоматично зв’язуються із моделлю і додаються до бази даних.

* **Індивідуальне видалення:**

Для кожного запису в таблиці є окрема форма з кнопкою «Видалити», що дозволяє видалити бажаний запис. Після видалення запис оновлюється, і користувач бачить таблицю без видаленого елемента.

* **Динамічне оновлення відображення:**

Після виконання операцій (додавання або видалення) контролер завантажує актуальний список записів із БД, який відображається під формою – це забезпечує миттєве відображення змін.

## **Висновок**

* **Розширення функціоналу:**

Модифікація завдання з додаванням засобів видалення окремих записів дозволяє користувачу краще управляти даними в базі – як додавати, так і видаляти непотрібні записи.

* **Підвищення інтерактивності:**

Динамічне оновлення списку записів після кожної операції гарантує, що користувач завжди бачить актуальний стан бази даних, що сприяє підвищенню зручності використання застосунку.

* **Модульність та підтримуваність:**

Розділення логіки на моделі, контролер і представлення відповідає принципу MVC, що спрощує подальшу підтримку, розширення функціоналу та інтеграцію додаткових можливостей у застосунок.

# **Завдання 6**

Модифікувати попереднє завдання додавши до форми засоби редагу-вання бажаного запису з таблиці відповідної MySQL БД в окремому поданні з подальшою демонстрацією вмісту таблиці без видаленого запису під формою попереднього подання.

## **Текст програми:**

**Index.cshtml:**

@model Task\_6.Models.Transport

@{

ViewData["Title"] = "Форма транспорту";

// Отримуємо записи з ViewBag для відображення таблиці

var transports = ViewBag.Transports as List<Task\_6.Models.Transport>;

}

<h1>Форма введення даних про транспорт</h1>

<!-- Форма для додавання нового запису -->

<form asp-action="Index" method="post">

<div class="mb-3">

<label asp-for="TransportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input asp-for="TransportType" class="form-control" placeholder="Введіть вид транспорту" />

</div>

<div class="mb-3">

<label asp-for="RouteNumber" class="form-label">Номер маршруту</label>

<input asp-for="RouteNumber" class="form-control" placeholder="Введіть номер маршруту" />

</div>

<div class="mb-3">

<label asp-for="RouteDistance" class="form-label">Протяжність маршруту (км)</label>

<input asp-for="RouteDistance" class="form-control" placeholder="Введіть протяжність" type="number" step="0.01" />

</div>

<div class="mb-3">

<label asp-for="TravelTime" class="form-label">Час в дорозі (хв)</label>

<input asp-for="TravelTime" class="form-control" placeholder="Введіть час в дорозі" type="number" />

</div>

<button type="submit" class="btn btn-primary">Додати до БД</button>

<!-- Перезавантаження сторінки для скидання форми -->

<button type="button" class="btn btn-secondary" onclick="window.location.href='@Url.Action("Index", "Transport")'">Скинути</button>

</form>

<!-- Форма для видалення всіх записів -->

<form asp-action="DeleteAll" method="post" style="margin-top: 15px;">

<button type="submit" class="btn btn-danger">Видалити всі записи</button>

</form>

@if (transports != null && transports.Any())

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<table class="table custom-bordered-table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність маршруту (км)</th>

<th>Час в дорозі (хв)</th>

<th>Дії</th>

</tr>

</thead>

<tbody>

@foreach (var item in transports)

{

<tr>

<td>@item.Id</td>

<td>@item.TransportType</td>

<td>@item.RouteNumber</td>

<td>@item.RouteDistance</td>

<td>@item.TravelTime</td>

<td>

<!-- Посилання для переходу до редагування -->

<a href="@Url.Action("Edit", "Transport", new { id = item.Id })"

class="btn btn-sm btn-info btn-edit">

Редагувати

</a>

<!-- Форма для видалення запису -->

<form asp-action="DeleteRecord" method="post" class="inline-form">

<input type="hidden" name="id" value="@item.Id" />

<button type="submit" class="btn-delete">Видалити</button>

</form>

</td>

</tr>

}

</tbody>

</table>

</div>

}

else

{

<hr />

<div class="result">

<h2>Вміст таблиці Transports:</h2>

<p>Таблиця порожня.</p>

</div>

}

**Edit.cshtml:**

@model Task\_6.Models.Transport

@{

ViewData["Title"] = "Редагування запису";

}

<h1>Редагування запису</h1>

<!-- Форма редагування запису -->

<form asp-action="Edit" method="post">

<input type="hidden" asp-for="Id" />

<div class="mb-3">

<label asp-for="TransportType" class="form-label">Вид транспорту (Tr, Tl, A)</label>

<input asp-for="TransportType" class="form-control" />

</div>

<div class="mb-3">

<label asp-for="RouteNumber" class="form-label">Номер маршруту</label>

<input asp-for="RouteNumber" class="form-control" />

</div>

<div class="mb-3">

<label asp-for="RouteDistance" class="form-label">Протяжність маршруту (км)</label>

<input asp-for="RouteDistance" class="form-control" type="number" step="0.01" />

</div>

<div class="mb-3">

<label asp-for="TravelTime" class="form-label">Час в дорозі (хв)</label>

<input asp-for="TravelTime" class="form-control" type="number" />

</div>

<button type="submit" class="btn btn-primary">Зберегти зміни</button>

<!-- Посилання для повернення до головної форми -->

<a href="@Url.Action("Index", "Transport")" class="btn btn-secondary">Відмінити</a>

</form>

**TransportContext.cs:**

using Microsoft.EntityFrameworkCore;

using System.Collections.Generic;

namespace Task\_6.Models

{

public class TransportContext : DbContext

{

public TransportContext(DbContextOptions<TransportContext> options)

: base(options)

{

}

// DB набір для записів транспорту

public DbSet<Transport> Transports { get; set; }

}

}

**Transport.cs:**

namespace Task\_6.Models

{

public class Transport

{

// ID, створюється автоматично в БД

public int Id { get; set; }

// Вид транспорту (Tr, Tl, A)

public string TransportType { get; set; }

// Номер маршруту (наприклад, "12a")

public string RouteNumber { get; set; }

// Протяжність маршруту в км

public double RouteDistance { get; set; }

// Час в дорозі в хвилинах

public int TravelTime { get; set; }

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_6.Models;

using System.Linq;

namespace Task\_6.Controllers

{

public class TransportController : Controller

{

private readonly TransportContext \_context;

public TransportController(TransportContext context)

{

\_context = context; // Ініціалізація доступу до БД

}

// GET: /Transport/Index – відображаємо форму та записи

[HttpGet]

public IActionResult Index()

{

var model = new Transport(); // Створюємо порожню модель

var transports = \_context.Transports.ToList(); // Отримуємо записи з БД

ViewBag.Transports = transports;

return View(model);

}

// POST: /Transport/Index – додаємо новий запис

[HttpPost]

public IActionResult Index(Transport transport)

{

\_context.Transports.Add(transport); // Додавання запису

\_context.SaveChanges(); // Збереження змін

var transports = \_context.Transports.ToList(); // Оновлюємо список записів

ViewBag.Transports = transports;

return View(transport);

}

// POST: /Transport/DeleteAll – видаляємо всі записи

[HttpPost]

public IActionResult DeleteAll()

{

\_context.Transports.RemoveRange(\_context.Transports); // Видалення всіх записів

\_context.SaveChanges();

var model = new Transport(); // Порожня модель для форми

ViewBag.Transports = \_context.Transports.ToList(); // Оновлений список (має бути порожнім)

return View("Index", model);

}

// POST: /Transport/DeleteRecord – видаляємо конкретний запис

[HttpPost]

public IActionResult DeleteRecord(int id)

{

var record = \_context.Transports.FirstOrDefault(t => t.Id == id);

if (record != null)

{

\_context.Transports.Remove(record); // Видаляємо знайдений запис

\_context.SaveChanges();

}

var model = new Transport();

ViewBag.Transports = \_context.Transports.ToList();

return View("Index", model);

}

// GET: /Transport/Edit/{id} – відображення форми редагування

[HttpGet]

public IActionResult Edit(int id)

{

var record = \_context.Transports.FirstOrDefault(t => t.Id == id);

if (record == null)

{

return RedirectToAction("Index"); // Якщо запис не знайдено

}

return View(record);

}

// POST: /Transport/Edit – збереження відредагованих даних

[HttpPost]

public IActionResult Edit(Transport transport)

{

if (ModelState.IsValid)

{

\_context.Transports.Update(transport); // Оновлення даних

\_context.SaveChanges();

return RedirectToAction("Index");

}

return View(transport);

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<title>@ViewData["Title"] - Task\_6</title>

<link rel="stylesheet" href="~/css/site.css" />

<!-- Підключення Bootstrap 5 -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2vBZX3L9/aqG5Uq4fZAS+2jQ0/1KW8fW/5pXz"

crossorigin="anonymous">

</head>

<body>

<!-- Обгортка для кращої читабельності -->

<div class="overlay">

<div class="container">

<main role="main" class="pb-3">

@RenderBody() <!-- Основний контент -->

</main>

</div>

</div>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

</body>

</html>

**site.css:**

/\* Фон сторінки із зображенням \*/

body {

background: url('https://www.hdwallpapers.in/download/usa\_vermont\_forest\_with\_house\_and\_road\_between\_field\_and\_trees\_under\_cloudy\_sky\_during\_fall\_hd\_nature-1920x1080.jpg') no-repeat center center fixed;

background-size: cover;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

margin: 0;

padding: 0;

animation: fadeInBackground 3s ease-in-out;

}

@keyframes fadeInBackground {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Оверлей для покращення читабельності \*/

.overlay {

background-color: rgba(255, 255, 255, 0.2);

min-height: 100vh;

padding: 20px;

}

/\* Відступи зверху для контейнера \*/

.container {

padding-top: 30px;

}

/\* Заголовки з bounce-анімацією \*/

h1, h2 {

color: #333;

text-align: center;

animation: bounce 2s;

}

@keyframes bounce {

0%, 20%, 50%, 80%, 100% { transform: translateY(0); }

40% { transform: translateY(-20px); }

60% { transform: translateY(-10px); }

}

/\* Стиль форми з ефектом появи \*/

form {

background: rgba(255, 255, 255, 0.95);

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

animation: slideIn 1s ease-out;

margin-bottom: 0;

}

@keyframes slideIn {

from { transform: translateY(-20px); opacity: 0; }

to { transform: translateY(0); opacity: 1; }

}

/\* Плавний перехід для полів форми \*/

.form-control {

transition: border-color 0.3s, box-shadow 0.3s;

}

.form-control:focus {

border-color: #80bdff;

box-shadow: 0 0 0 0.2rem rgba(0,123,255,0.25);

}

/\* Відступи між блоками форми \*/

.mb-3 {

margin-bottom: 1rem;

}

/\* Стилі для кнопок \*/

.btn-primary {

background-color: #007bff;

border-color: #007bff;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

.btn-secondary {

background-color: #6c757d;

border-color: #6c757d;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-secondary:hover {

background-color: #5a6268;

border-color: #545b62;

}

.btn-danger {

background-color: #dc3545;

border-color: #dc3545;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-danger:hover {

background-color: #bb2d3b;

border-color: #b02a37;

}

/\* Стиль для кнопки редагування \*/

.btn-edit {

background-color: #17a2b8;

border-color: #17a2b8;

color: #fff;

border-radius: 12px;

padding: 6px 12px;

font-size: 0.875rem;

margin-right: 4px;

text-decoration: none;

transition: background-color 0.3s, border-color 0.3s;

}

.btn-edit:hover {

background-color: #138496;

border-color: #117a8b;

}

/\* Стиль для кнопки видалення запису \*/

.btn-delete {

background-color: #dc3545;

border: none;

color: #fff;

border-radius: 12px;

padding: 6px 12px;

font-size: 0.875rem;

cursor: pointer;

transition: background-color 0.3s, border-color 0.3s;

outline: none;

box-shadow: none;

}

.btn-delete:hover {

background-color: #bb2d3b;

}

/\* Inline форма для окремого запису \*/

.inline-form {

display: inline-block;

margin: 0;

}

/\* Стиль блоку результату \*/

.result {

text-align: center;

}

.result h2 {

color: #fff !important;

}

/\* Таблиця з білими границями та текстом \*/

.custom-bordered-table {

border-collapse: collapse;

margin: 0 auto;

background-color: rgba(0, 0, 0, 0.5);

color: #fff;

width: 80%;

}

.custom-bordered-table th,

.custom-bordered-table td {

border: 1px solid #fff !important;

padding: 8px;

vertical-align: middle;

text-align: center;

}

.custom-bordered-table th {

font-weight: bold;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6.1. Результати роботи виконання Завдання 6 (Створена БД)

Изображение выглядит как текст, снимок экрана, небо, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6.2. Результати роботи виконання Завдання 6

Изображение выглядит как дерево, текст, снимок экрана, небо

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6.3. Результати роботи виконання Завдання 6

Изображение выглядит как текст, снимок экрана, небо, облако

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 6.4. Результати роботи виконання Завдання 6

## **Алгоритм побудови кодів**

1. **Налаштування моделі та контексту даних:**
   * Створено клас **Transport** з властивостями: Id, TransportType, RouteNumber, RouteDistance та TravelTime.
   * Реалізовано клас **TransportContext** на основі DbContext, що містить DbSet для роботи з таблицею записів.
2. **Розробка представлень:**
   * **Index.cshtml:**
     + Містить форму для додавання нового запису з полями, що зв’язуються з моделлю через TagHelper-и.
     + Відображає таблицю поточних записів із БД, де для кожного запису додано елементи дій: посилання для переходу до редагування (Edit) та форма для видалення конкретного запису.
   * **Edit.cshtml:**
     + Створено окреме подання для редагування вибраного запису. Форма отримує початкові дані запису, а після зміни даних користувач може зберегти оновлення через POST-запит.
3. **Реалізація контролера:**
   * **GET метод Index:** Завантажує порожню модель та список записів із БД, які передаються через ViewBag для відображення.
   * **POST метод Index:** Приймає дані із форми для додавання нового запису та оновлює список записів.
   * **POST метод DeleteRecord:** Видаляє конкретний запис за переданим Id і повертає оновлений список записів.
   * **GET метод Edit:** Відображає форму редагування для запису за заданим Id.
   * **POST метод Edit:** Приймає відредаговані дані, оновлює запис у базі та перенаправляє користувача на головну форму, де відображається оновлений список.

## **Функціональність**

* **Додавання записів:**

Суворо типізована форма дозволяє користувачу вводити дані про транспорт, які зберігаються в MySQL базі даних та відображаються у таблиці.

* **Редагування запису:**

Кожен запис у таблиці має кнопку «Редагувати», що веде до окремого подання, де користувач може змінити потрібні поля. Після збереження змін, оновлений запис відображається у списку без видаленого (редагованого) варіанту.

* **Видалення окремих записів:**

Для кожного запису передбачено окрему форму з кнопкою «Видалити», що дозволяє індивідуально видаляти запис із бази даних із динамічним оновленням вмісту таблиці.

* **Загальне управління даними:**

Комбінація операцій додавання, редагування та видалення дозволяє комплексно управляти даними у застосунку, забезпечуючи актуальність і коректність відображуваної інформації.

## **Висновок**

* **Покращення функціоналу:**

Додавання можливості редагування окремих записів дає змогу оперативно коригувати дані, що підвищує гнучкість та зручність керування інформацією.

* **Інтерактивність і зручність:**

Завдяки впровадженню окремого подання для редагування та наявності засобів для видалення записів, користувач отримує повноцінний механізм CRUD-операцій, що дозволяє працювати з даними ефективно та інтуїтивно.

* **Модульність та підтримуваність:**

Чітке розділення логіки між моделлю, контролером і представленнями відповідає принципам MVC, що спрощує подальшу підтримку та розширення функціоналу застосунку.

# **Завдання 7**

За допомогою Entity Framework (EF) створити застосунок обробки (додання, редагування, видалення) Microsoft SQL БД, яка зберігається в файлі \*.mdf.

## **Текст програми:**

**Index.cshtml:**

@model IEnumerable<Task\_7\_1.Models.Transport>

@{

ViewData["Title"] = "Список маршрутів";

// Модель містить список маршрутів

}

<h1>Список маршрутів</h1>

<p>

<!-- Кнопка для переходу до створення нового маршруту -->

<a class="btn btn-primary" asp-action="Create">Додати новий маршрут</a>

</p>

<table class="table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність (км)</th>

<th>Час в дорозі (хв)</th>

<th>Дії</th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr>

<td>@item.Id</td>

<td>@item.VidTransportu</td>

<td>@item.NomMarshruta</td>

<td>@item.ProtjazhnistMarshruta</td>

<td>@item.ChasVDorozi</td>

<td>

<!-- Посилання для редагування, деталей та видалення маршруту -->

<a class="btn btn-secondary" asp-action="Edit" asp-route-id="@item.Id">Редагувати</a>

<a class="btn btn-info" asp-action="Details" asp-route-id="@item.Id">Деталі</a>

<a class="btn btn-danger" asp-action="Delete" asp-route-id="@item.Id">Видалити</a>

</td>

</tr>

}

</tbody>

</table>

**Edit.cshtml:**

@model Task\_7\_1.Models.Transport

@{

ViewData["Title"] = "Редагувати маршрут";

// Модель містить дані маршруту для редагування

}

<h1>Редагувати маршрут</h1>

<form asp-action="Edit" method="post">

<input type="hidden" asp-for="Id" />

<div class="form-group">

<label asp-for="VidTransportu">Вид транспорту</label>

<input asp-for="VidTransportu" class="form-control" />

</div>

<div class="form-group">

<label asp-for="NomMarshruta">Номер маршруту</label>

<input asp-for="NomMarshruta" class="form-control" type="number" />

</div>

<div class="form-group">

<label asp-for="ProtjazhnistMarshruta">Протяжність (км)</label>

<input asp-for="ProtjazhnistMarshruta" class="form-control" type="number" step="0.01" />

</div>

<div class="form-group">

<label asp-for="ChasVDorozi">Час в дорозі (хв)</label>

<input asp-for="ChasVDorozi" class="form-control" type="number" />

</div>

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

<!-- Кнопка повернення до списку -->

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

**Details.cshtml:**

@model Task\_7\_1.Models.Transport

@{

ViewData["Title"] = "Деталі маршруту";

// Модель містить дані маршруту для перегляду

}

<div class="details-card">

<h2>Деталі маршруту</h2>

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.VidTransportu</p>

<p><strong>Номер маршруту:</strong> @Model.NomMarshruta</p>

<p><strong>Протяжність (км):</strong> @Model.ProtjazhnistMarshruta</p>

<p><strong>Час в дорозі (хв):</strong> @Model.ChasVDorozi</p>

<hr />

<!-- Кнопки для повернення та редагування -->

<a class="btn btn-secondary" asp-action="Index">Назад</a>

<a class="btn btn-primary" asp-action="Edit" asp-route-id="@Model.Id">Редагувати</a>

</div>

**Delete.cshtml:**

@model Task\_7\_1.Models.Transport

@{

ViewData["Title"] = "Видалити маршрут";

// Модель містить дані маршруту для підтвердження видалення

}

<h1>Видалення маршруту</h1>

<div>

<h4>Ви дійсно бажаєте видалити цей маршрут?</h4>

<hr />

<dl class="dl-horizontal">

<dt>Id</dt>

<dd>@Model.Id</dd>

<dt>Вид транспорту</dt>

<dd>@Model.VidTransportu</dd>

<dt>Номер маршруту</dt>

<dd>@Model.NomMarshruta</dd>

<dt>Протяжність (км)</dt>

<dd>@Model.ProtjazhnistMarshruta</dd>

<dt>Час в дорозі (хв)</dt>

<dd>@Model.ChasVDorozi</dd>

</dl>

<form asp-action="Delete" method="post">

<input type="hidden" asp-for="Id" />

<button type="submit" class="btn btn-danger">Видалити</button>

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

</div>

**Create.cshtml:**

@model Task\_7\_1.Models.Transport

@{

ViewData["Title"] = "Додати маршрут";

// Модель використовується для створення нового маршруту

}

<h1>Додати маршрут</h1>

<form asp-action="Create" method="post">

<div class="form-group">

<label asp-for="VidTransportu">Вид транспорту</label>

<input asp-for="VidTransportu" class="form-control" />

</div>

<div class="form-group">

<label asp-for="NomMarshruta">Номер маршруту</label>

<input asp-for="NomMarshruta" class="form-control" type="number" />

</div>

<div class="form-group">

<label asp-for="ProtjazhnistMarshruta">Протяжність (км)</label>

<input asp-for="ProtjazhnistMarshruta" class="form-control" type="number" step="0.01" />

</div>

<div class="form-group">

<label asp-for="ChasVDorozi">Час в дорозі (хв)</label>

<input asp-for="ChasVDorozi" class="form-control" type="number" />

</div>

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

<!-- Кнопка повернення до списку -->

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

**Transport.cs:**

namespace Task\_7\_1.Models

{

public class Transport

{

public int Id { get; set; } // Первинний ключ

public string VidTransportu { get; set; } // Вид транспорту (наприклад, "Tr", "Tl", "A")

public int NomMarshruta { get; set; } // Номер маршруту

public float ProtjazhnistMarshruta { get; set; } // Протяжність маршруту (км)

public int ChasVDorozi { get; set; } // Час в дорозі (хв)

}

}

**Data/ApplicationDbContext.cs:**

using Microsoft.EntityFrameworkCore;

using Task\_7\_1.Models;

namespace Task\_7\_1.Data

{

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

// Набір даних для транспорту

public DbSet<Transport> Transports { get; set; }

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Task\_7\_1.Data;

using Task\_7\_1.Models;

namespace Task\_7\_1.Controllers

{

public class TransportController : Controller

{

private readonly ApplicationDbContext \_context;

public TransportController(ApplicationDbContext context)

{

\_context = context; // Ініціалізація контексту БД

}

// GET: Transport - вивід списку маршрутів

public async Task<IActionResult> Index()

{

var transports = await \_context.Transports.ToListAsync();

return View(transports);

}

// GET: Transport/Details/5 - перегляд деталей маршруту

public async Task<IActionResult> Details(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FirstOrDefaultAsync(m => m.Id == id);

if (transport == null)

return NotFound();

return View(transport);

}

// GET: Transport/Create - форма для створення маршруту

public IActionResult Create()

{

return View();

}

// POST: Transport/Create - збереження нового маршруту

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("Id,VidTransportu,NomMarshruta,ProtjazhnistMarshruta,ChasVDorozi")] Transport transport)

{

if (ModelState.IsValid)

{

\_context.Add(transport);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(transport);

}

// GET: Transport/Edit/5 - форма редагування маршруту

public async Task<IActionResult> Edit(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FindAsync(id);

if (transport == null)

return NotFound();

return View(transport);

}

// POST: Transport/Edit/5 - збереження змін маршруту

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("Id,VidTransportu,NomMarshruta,ProtjazhnistMarshruta,ChasVDorozi")] Transport transport)

{

if (id != transport.Id)

return NotFound();

if (ModelState.IsValid)

{

try

{

\_context.Update(transport);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!TransportExists(transport.Id))

return NotFound();

else

throw;

}

return RedirectToAction(nameof(Index));

}

return View(transport);

}

// GET: Transport/Delete/5 - підтвердження видалення маршруту

public async Task<IActionResult> Delete(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FirstOrDefaultAsync(m => m.Id == id);

if (transport == null)

return NotFound();

return View(transport);

}

// POST: Transport/Delete/5 - видалення маршруту

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var transport = await \_context.Transports.FindAsync(id);

if (transport != null)

{

\_context.Transports.Remove(transport);

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

// Перевірка наявності маршруту

private bool TransportExists(int id)

{

return \_context.Transports.Any(e => e.Id == id);

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>@ViewData["Title"] - Task\_7\_1</title>

<link rel="stylesheet" href="~/css/site.css" />

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<!-- Фонове слайдшоу -->

<div class="background-slideshow">

<div class="slide slide1"></div>

<div class="slide slide2"></div>

<div class="slide slide3"></div>

<div class="slide slide4"></div>

<div class="slide slide5"></div>

</div>

<header>

<nav class="navbar">

<!-- Меню навігації -->

<a class="nav-link" href="@Url.Action("Index", "Transport")">Головна</a>

<a class="nav-link" href="@Url.Action("Create", "Transport")">Додати запис</a>

</nav>

</header>

<div class="container">

@RenderBody() <!-- Основний контент -->

</div>

<footer>

<p>&copy; @DateTime.Now.Year - Task\_7\_1</p>

</footer>

@RenderSection("Scripts", required: false)

</body>

</html>

**site.css:**

/\* Скидання стилів та основний фон \*/

body, html {

margin: 0;

padding: 0;

height: 100%;

font-family: Arial, sans-serif;

}

/\* Фонове слайдшоу \*/

.background-slideshow {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

z-index: -1;

overflow: hidden;

}

.background-slideshow .slide {

position: absolute;

width: 100%;

height: 100%;

background-size: cover;

background-position: center center;

opacity: 0;

animation: fade 25s infinite;

}

/\* Фонові зображення з stagger delay \*/

.background-slideshow .slide1 {

background-image: url('https://i.imgur.com/RGka78l.jpg');

animation-delay: 0s;

}

.background-slideshow .slide2 {

background-image: url('https://wallpaper-mania.com/wp-content/uploads/2018/09/High\_resolution\_wallpaper\_background\_ID\_77701304695.jpg');

animation-delay: 5s;

}

.background-slideshow .slide3 {

background-image: url('https://wallpapers.com/images/hd/1920-x-1080-nature-desktop-chafsybseyq3lp0n.jpg');

animation-delay: 10s;

}

.background-slideshow .slide4 {

background-image: url('https://wallpapers.com/images/hd/1920x1080-full-hd-nature-landscape-54q8pleyp8lhhbu7.jpg');

animation-delay: 15s;

}

.background-slideshow .slide5 {

background-image: url('https://images.unsplash.com/photo-1469474968028-56623f02e42e?fm=jpg&q=60&w=3000');

animation-delay: 20s;

}

@keyframes fade {

0% { opacity: 0; }

10% { opacity: 1; }

30% { opacity: 1; }

40% { opacity: 0; }

100% { opacity: 0; }

}

/\* Контейнер для контенту \*/

.container {

position: relative;

z-index: 2;

margin: 20px;

background: rgba(255, 255, 255, 0.85);

padding: 20px;

border-radius: 8px;

}

/\* Навігаційне меню \*/

.navbar {

background: rgba(0, 0, 0, 0.6);

padding: 10px 20px;

position: relative;

z-index: 3;

}

.navbar .nav-link {

color: #fff;

text-decoration: none;

margin-right: 15px;

transition: color 0.3s ease;

}

.navbar .nav-link:hover {

color: #f0a500;

}

/\* Футер \*/

footer {

text-align: center;

padding: 10px;

background: rgba(0, 0, 0, 0.6);

color: #fff;

position: fixed;

bottom: 0;

width: 100%;

z-index: 3;

}

/\* Анімація форм \*/

form {

animation: slideIn 1s ease-out;

}

@keyframes slideIn {

from { opacity: 0; transform: translateY(-20px); }

to { opacity: 1; transform: translateY(0); }

}

/\* Таблиця \*/

.table {

width: 100%;

border-collapse: collapse;

animation: fadeIn 1s ease-in;

}

.table th, .table td {

padding: 10px;

border: 1px solid #ccc;

}

@keyframes fadeIn {

from { opacity: 0; }

to { opacity: 1; }

}

/\* Стилі кнопок \*/

.btn {

padding: 8px 12px;

border: none;

border-radius: 4px;

cursor: pointer;

margin: 5px;

transition: background 0.3s;

}

.btn-primary {

background: #007bff;

color: #fff;

}

.btn-primary:hover {

background: #0056b3;

}

.btn-secondary {

background: #6c757d;

color: #fff;

}

.btn-secondary:hover {

background: #5a6268;

}

.btn-danger {

background: #dc3545;

color: #fff;

}

.btn-danger:hover {

background: #c82333;

}

/\* Додаткові стилі для формових елементів \*/

.form-group {

margin-bottom: 15px;

}

label {

display: block;

margin-bottom: 5px;

}

.form-control {

width: 100%;

padding: 8px;

box-sizing: border-box;

}

/\* Картка деталей маршруту \*/

.details-card {

margin: 20px auto;

max-width: 600px;

background: rgba(255, 255, 255, 0.9);

padding: 20px;

border-radius: 8px;

box-shadow: 0 2px 6px rgba(0, 0, 0, 0.2);

animation: fadeIn 1.2s ease-in-out;

}

.details-card h2 {

margin-top: 0;

margin-bottom: 15px;

}

.details-card p {

margin: 10px 0;

}

## **Результати роботи програми:**

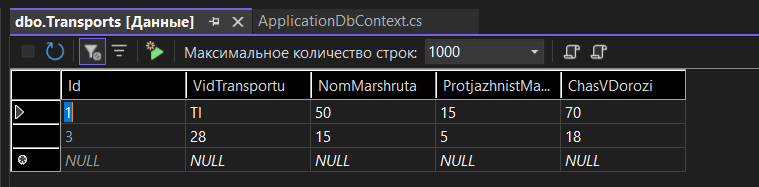


Рисунок 7.1. Результати роботи виконання Завдання 7 (Створена БД)

Изображение выглядит как текст, дерево, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.2. Результати роботи виконання Завдання 7 (Головна сторінка)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.3. Результати роботи виконання Завдання 7 (Додавання маршруту)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.4. Результати роботи виконання Завдання 7 (Результат додавання маршруту)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.5. Результати роботи виконання Завдання 7 (Редагування)

Изображение выглядит как текст, снимок экрана, программное обеспечение, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.6. Результати роботи виконання Завдання 7 (Результати редагування)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.7. Результати роботи виконання Завдання 7 (Деталі маршруту)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.8. Результати роботи виконання Завдання 7 (Видалення маршруту)

Изображение выглядит как текст, снимок экрана, вода, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 7.9. Результати роботи виконання Завдання 7 (Результати видалення)

## **Алгоритм побудови кодів**

1. **Налаштування моделі та контексту даних:**
   * Створено клас **Transport** із властивостями: Id, VidTransportu, NomMarshruta, ProtjazhnistMarshruta та ChasVDorozi, що описують дані маршруту.
   * Реалізовано клас **ApplicationDbContext** (розташований у Data), який наслідується від DbContext і містить DbSet<Transport>. База даних зберігається у файлі \*.mdf, що забезпечує інтеграцію з Microsoft SQL Server.
2. **Створення представлень (Views):**
   * **Index.cshtml:** Відповідає за відображення списку маршрутів. За допомогою циклу foreach показано всі записи з бази даних із використанням таблиці. Для кожного запису передбачено кнопки для виконання CRUD‑операцій (редагувати, переглянути деталі, видалити).
   * **Create.cshtml:** Форма для створення нового маршруту, яка використовує модель для зв’язування даних.
   * **Edit.cshtml:** Подання для редагування існуючого запису. Форма попередньо заповнюється даними обраного маршруту, що дозволяє вносити зміни.
   * **Details.cshtml:** Подання для перегляду докладної інформації про конкретний маршрут.
   * **Delete.cshtml:** Подання для підтвердження видалення маршруту з відображенням деталей запису.
3. **Реалізація контролера:**
   * **TransportController:** Використовує ApplicationDbContext для роботи з базою даних.
     + **Index:** Асинхронний метод, який завантажує всі записи з таблиці та передає їх у подання як модель.
     + **Create:** Методи GET та POST для створення нового маршруту. При успішній валідації даних новий запис додається до бази, після чого здійснюється перенаправлення на Index.
     + **Edit:** Методи GET та POST для редагування запису. При зверненні GET – формується форма із початковими даними, а при POST – збереження змін та перенаправлення.
     + **Details:** Метод для перегляду інформації про обраний маршрут.
     + **Delete:** Методи GET (для відображення підтвердження) та POST (для видалення запису). Видалення здійснюється, після чого користувач перенаправляється на список маршрутів.

## **Функціональність**

* **Виведення даних:**

Список маршрутів завантажується з бази даних (файл \*.mdf), і всі записи відображаються у таблиці на головній сторінці, що дозволяє швидко ознайомитися з інформацією про транспорт.

* **Додавання записів:**

Форма створення нового маршруту дозволяє вводити дані про транспорт, які потім додаються до бази даних. Після успішного додавання користувач перенаправляється до списку маршрутів.

* **Редагування та перегляд деталей:**

Для кожного запису доступні окремі подання редагування та перегляду деталей. Це забезпечує можливість внесення коректив у дані маршруту або перегляду його повної інформації.

* **Видалення записів:**

Видалення маршруту здійснюється через подання для підтвердження. При видаленні запис миттєво оновлюється, що забезпечує актуальність списку маршрутів.

## **Висновок**

* **Інтеграція з Microsoft SQL:**

Використання Entity Framework забезпечує ефективну роботу зі збереженням, оновленням та видаленням даних у базі даних, яка зберігається у файлі \*.mdf. Це спрощує процес управління даними і забезпечує стабільність роботи застосунку.

* **Повноцінні CRUD‑операції:**

Завдяки реалізації повного набору операцій (додавання, редагування, перегляд, видалення) користувач отримує можливість комплексного керування записами маршруту, що полегшує адміністрування даних.

* **Модульна структура:**

Розділення логіки між моделлю, представленнями та контролером відповідає архітектурі MVC. Це сприяє легкій підтримці, розширенню застосунку та подальшій інтеграції нових можливостей, що забезпечує сучасний та зручний інтерфейс для роботи з даними.

# **Завдання 8**

Модифікувати минуле завдання зробивши початкове заповнення таб-лиці БД трьома записами з таблиці власного варіанту.

## **Текст програми:**

**Index.cshtml:**

@model IEnumerable<Task\_8.Models.Transport>

@{

ViewData["Title"] = "Список маршрутів";

// Використовуємо модель, яка містить список маршрутів

}

<h1>Список маршрутів</h1>

<p>

<!-- Кнопка для переходу до створення нового маршруту -->

<a class="btn btn-primary" asp-action="Create">Додати новий маршрут</a>

</p>

<table class="table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність (км)</th>

<th>Час в дорозі (хв)</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr>

<td>@item.Id</td>

<td>@item.VidTransportu</td>

<td>@item.NomMarshruta</td>

<td>@item.ProtjazhnistMarshruta</td>

<td>@item.ChasVDorozi</td>

<td>

<!-- Посилання для редагування, перегляду деталей та видалення -->

<a class="btn btn-secondary" asp-action="Edit" asp-route-id="@item.Id">Редагувати</a> |

<a class="btn btn-info" asp-action="Details" asp-route-id="@item.Id">Деталі</a> |

<a class="btn btn-danger" asp-action="Delete" asp-route-id="@item.Id">Видалити</a>

</td>

</tr>

}

</tbody>

</table>

**Edit.cshtml:**

@model Task\_8.Models.Transport

@{

ViewData["Title"] = "Редагувати маршрут";

// Модель містить дані маршруту для редагування

}

<h1>Редагувати маршрут</h1>

<form asp-action="Edit" method="post">

<input type="hidden" asp-for="Id" />

<div class="form-group">

<label>Вид транспорту</label>

<input asp-for="VidTransportu" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Номер маршруту</label>

<input asp-for="NomMarshruta" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Протяжність (км)</label>

<input asp-for="ProtjazhnistMarshruta" class="form-control" type="number" step="0.01" />

</div>

<div class="form-group">

<label>Час в дорозі (хв)</label>

<input asp-for="ChasVDorozi" class="form-control" type="number" />

</div>

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

<!-- Кнопка повернення до списку маршрутів -->

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

**Details.cshtml:**

@model Task\_8.Models.Transport

@{

ViewData["Title"] = "Деталі маршруту";

// Відображення даних вибраного маршруту

}

<div class="details-card">

<h2>Деталі маршруту</h2>

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.VidTransportu</p>

<p><strong>Номер маршруту:</strong> @Model.NomMarshruta</p>

<p><strong>Протяжність (км):</strong> @Model.ProtjazhnistMarshruta</p>

<p><strong>Час в дорозі (хв):</strong> @Model.ChasVDorozi</p>

<hr />

<!-- Кнопки для повернення та редагування маршруту -->

<a class="btn btn-secondary" asp-action="Index">Назад</a>

<a class="btn btn-primary" asp-action="Edit" asp-route-id="@Model.Id">Редагувати</a>

</div>

**Delete.cshtml:**

@model Task\_8.Models.Transport

@{

ViewData["Title"] = "Видалити маршрут";

// Підтвердження видалення маршруту

}

<div class="details-card">

<h2>Видалення маршруту</h2>

<p>Ви дійсно бажаєте видалити цей маршрут?</p>

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.VidTransportu</p>

<p><strong>Номер маршруту:</strong> @Model.NomMarshruta</p>

<p><strong>Протяжність (км):</strong> @Model.ProtjazhnistMarshruta</p>

<p><strong>Час в дорозі (хв):</strong> @Model.ChasVDorozi</p>

<hr />

<form asp-action="Delete" method="post">

<input type="hidden" asp-for="Id" />

<button type="submit" class="btn btn-danger">Видалити</button>

<!-- Кнопка повернення до списку -->

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

</div>

**Create.cshtml:**

@model Task\_8.Models.Transport

@{

ViewData["Title"] = "Додати маршрут";

// Форма для створення нового маршруту

}

<h1>Додати маршрут</h1>

<form asp-action="Create" method="post">

<div class="form-group">

<label>Вид транспорту</label>

<input asp-for="VidTransportu" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Номер маршруту</label>

<input asp-for="NomMarshruta" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Протяжність (км)</label>

<input asp-for="ProtjazhnistMarshruta" class="form-control" type="number" step="0.01" />

</div>

<div class="form-group">

<label>Час в дорозі (хв)</label>

<input asp-for="ChasVDorozi" class="form-control" type="number" />

</div>

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

<!-- Кнопка повернення до списку маршрутів -->

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

**Transport.cs:**

namespace Task\_8.Models

{

public class Transport

{

public int Id { get; set; } // Первинний ключ

public string VidTransportu { get; set; } // "Tr", "Tl", "A"

public string NomMarshruta { get; set; } // Номер маршруту (якщо є букви, це string)

public float ProtjazhnistMarshruta { get; set; } // Протяжність маршруту (км)

public int ChasVDorozi { get; set; } // Час в дорозі (хв)

}

}

**ApplicationDbContext.cs:**

using Microsoft.EntityFrameworkCore;

using Task\_8.Models;

namespace Task\_8.Data

{

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

// DB набір для зберігання маршрутів

public DbSet<Transport> Transports { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

base.OnModelCreating(modelBuilder);

// Початкові дані для демонстрації

modelBuilder.Entity<Transport>().HasData(

new Transport

{

Id = 1,

VidTransportu = "Tr",

NomMarshruta = "12",

ProtjazhnistMarshruta = 27.55f,

ChasVDorozi = 75

},

new Transport

{

Id = 2,

VidTransportu = "Tl",

NomMarshruta = "17",

ProtjazhnistMarshruta = 13.6f,

ChasVDorozi = 57

},

new Transport

{

Id = 3,

VidTransportu = "A",

NomMarshruta = "12a",

ProtjazhnistMarshruta = 57.3f,

ChasVDorozi = 117

}

);

}

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Task\_8.Data;

using Task\_8.Models;

namespace Task\_8.Controllers

{

public class TransportController : Controller

{

private readonly ApplicationDbContext \_context;

public TransportController(ApplicationDbContext context)

{

\_context = context; // Ініціалізація доступу до БД

}

// GET: Transport - отримання списку маршрутів

public async Task<IActionResult> Index()

{

var transports = await \_context.Transports.ToListAsync();

return View(transports);

}

// GET: Transport/Details/5 - перегляд деталей маршруту

public async Task<IActionResult> Details(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FirstOrDefaultAsync(m => m.Id == id);

if (transport == null)

return NotFound();

return View(transport);

}

// GET: Transport/Create - форма додавання маршруту

public IActionResult Create()

{

return View();

}

// POST: Transport/Create - збереження нового маршруту

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("Id,VidTransportu,NomMarshruta,ProtjazhnistMarshruta,ChasVDorozi")] Transport transport)

{

if (ModelState.IsValid)

{

\_context.Add(transport);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(transport);

}

// GET: Transport/Edit/5 - форма редагування маршруту

public async Task<IActionResult> Edit(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FindAsync(id);

if (transport == null)

return NotFound();

return View(transport);

}

// POST: Transport/Edit/5 - збереження змін маршруту

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("Id,VidTransportu,NomMarshruta,ProtjazhnistMarshruta,ChasVDorozi")] Transport transport)

{

if (id != transport.Id)

return NotFound();

if (ModelState.IsValid)

{

try

{

\_context.Update(transport);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!TransportExists(transport.Id))

return NotFound();

else

throw;

}

return RedirectToAction(nameof(Index));

}

return View(transport);

}

// GET: Transport/Delete/5 - підтвердження видалення маршруту

public async Task<IActionResult> Delete(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FirstOrDefaultAsync(m => m.Id == id);

if (transport == null)

return NotFound();

return View(transport);

}

// POST: Transport/Delete/5 - видалення маршруту

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var transport = await \_context.Transports.FindAsync(id);

if (transport != null)

{

\_context.Transports.Remove(transport);

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

// Перевірка наявності маршруту

private bool TransportExists(int id)

{

return \_context.Transports.Any(e => e.Id == id);

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>@ViewData["Title"] - Task\_8</title>

<link rel="stylesheet" href="~/css/site.css" />

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<!-- Фонове слайдшоу -->

<div class="background-slideshow">

<div class="slide slide1"></div>

<div class="slide slide2"></div>

<div class="slide slide3"></div>

<div class="slide slide4"></div>

<div class="slide slide5"></div>

</div>

<header>

<nav class="navbar">

<!-- Навігаційне меню -->

<a class="nav-link" href="@Url.Action("Index", "Transport")">Головна</a>

<a class="nav-link" href="@Url.Action("Create", "Transport")">Додати запис</a>

</nav>

</header>

<div class="container">

@RenderBody() <!-- Основний контент -->

</div>

<footer>

<p>&copy; @DateTime.Now.Year - Task\_8</p>

</footer>

@RenderSection("Scripts", required: false)

</body>

</html>

**site.css:**

/\* Скидання базових стилів \*/

body, html {

margin: 0;

padding: 0;

height: 100%;

font-family: Arial, sans-serif;

}

/\* Фонове слайдшоу \*/

.background-slideshow {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

z-index: -1;

overflow: hidden;

}

.background-slideshow .slide {

position: absolute;

width: 100%;

height: 100%;

background-size: cover;

background-position: center center;

opacity: 0;

animation: fade 25s infinite;

}

.background-slideshow .slide1 {

background-image: url('https://i.imgur.com/RGka78l.jpg');

animation-delay: 0s;

}

.background-slideshow .slide2 {

background-image: url('https://wallpaper-mania.com/wp-content/uploads/2018/09/High\_resolution\_wallpaper\_background\_ID\_77701304695.jpg');

animation-delay: 5s;

}

.background-slideshow .slide3 {

background-image: url('https://wallpapers.com/images/hd/1920-x-1080-nature-desktop-chafsybseyq3lp0n.jpg');

animation-delay: 10s;

}

.background-slideshow .slide4 {

background-image: url('https://wallpapers.com/images/hd/1920x1080-full-hd-nature-landscape-54q8pleyp8lhhbu7.jpg');

animation-delay: 15s;

}

.background-slideshow .slide5 {

background-image: url('https://images.unsplash.com/photo-1469474968028-56623f02e42e?fm=jpg&q=60&w=3000&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxzZWFyY2h8N3x8MTkyMHgxMDgwJTIwd2FsbHBhcGVyfGVufDB8fDB8fHww');

animation-delay: 20s;

}

@keyframes fade {

0% { opacity: 0; }

10% { opacity: 1; }

30% { opacity: 1; }

40% { opacity: 0; }

100% { opacity: 0; }

}

/\* Контейнер для основного контенту \*/

.container {

position: relative;

z-index: 2;

margin: 20px;

background: rgba(255, 255, 255, 0.85);

padding: 20px;

border-radius: 8px;

}

/\* Навігаційне меню \*/

.navbar {

background: rgba(0, 0, 0, 0.6);

padding: 10px 20px;

position: relative;

z-index: 3;

}

.navbar .nav-link {

color: #fff;

text-decoration: none;

margin-right: 15px;

transition: color 0.3s ease;

}

.navbar .nav-link:hover {

color: #f0a500;

}

/\* Футер \*/

footer {

text-align: center;

padding: 10px;

background: rgba(0, 0, 0, 0.6);

color: #fff;

position: fixed;

bottom: 0;

width: 100%;

z-index: 3;

}

/\* Анімації для форм \*/

form {

animation: slideIn 1s ease-out;

}

@keyframes slideIn {

from { opacity: 0; transform: translateY(-20px); }

to { opacity: 1; transform: translateY(0); }

}

/\* Стилізація таблиць \*/

.table {

width: 100%;

border-collapse: collapse;

animation: fadeIn 1s ease-in;

}

@keyframes fadeIn {

from { opacity: 0; }

to { opacity: 1; }

}

.table th, .table td {

padding: 10px;

border: 1px solid #ccc;

}

/\* Стилі кнопок \*/

.btn {

padding: 8px 12px;

border: none;

border-radius: 4px;

cursor: pointer;

margin: 5px;

transition: background 0.3s;

}

.btn-primary {

background: #007bff;

color: #fff;

}

.btn-primary:hover {

background: #0056b3;

}

.btn-secondary {

background: #6c757d;

color: #fff;

}

.btn-secondary:hover {

background: #5a6268;

}

.btn-danger {

background: #dc3545;

color: #fff;

}

.btn-danger:hover {

background: #c82333;

}

/\* Додаткові стилі для формових елементів \*/

.form-group {

margin-bottom: 15px;

}

label {

display: block;

margin-bottom: 5px;

}

.form-control {

width: 100%;

padding: 8px;

box-sizing: border-box;

}

/\* Картка для деталей \*/

.details-card {

margin: 20px auto;

max-width: 600px;

background: rgba(255, 255, 255, 0.9);

padding: 20px;

border-radius: 8px;

box-shadow: 0 2px 6px rgba(0, 0, 0, 0.2);

animation: fadeIn 1.2s ease-in-out;

}

.details-card h2 {

margin-top: 0;

margin-bottom: 15px;

}

.details-card p {

margin: 10px 0;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.1. Результати роботи виконання Завдання 8 (Створена БД)

Изображение выглядит как текст, снимок экрана, программное обеспечение, вода

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.2. Результати роботи виконання Завдання 8 (Головна сторінка)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.3. Результати роботи виконання Завдання 8 (Додавання)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.4. Результати роботи виконання Завдання 8 (Результат додавання)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.5. Результати роботи виконання Завдання 8 (Редагування)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.6. Результати роботи виконання Завдання 8 (Результати редагування)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.7. Результати роботи виконання Завдання 8 (Деталі)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.8. Результати роботи виконання Завдання 8 (Видалення)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 8.9. Результати роботи виконання Завдання 8 (Результати видалення)

## **Алгоритм побудови кодів**

1. **Налаштування моделі та контексту:**
   * Створено клас **Transport** із властивостями: Id, VidTransportu, NomMarshruta, ProtjazhnistMarshruta та ChasVDorozi.
   * Реалізовано клас **ApplicationDbContext**, який наслідується від DbContext. У методі **OnModelCreating** додано початкове заповнення таблиці трьома записами для демонстрації (seed data).
2. **Розробка представлень (Views):**
   * **Index.cshtml:** Відображає список маршрутів у таблиці та містить посилання для створення нового маршруту. За допомогою циклу foreach показуються всі записи з бази даних.
   * **Create.cshtml:** Форма для створення нового маршруту, де користувач вводить потрібні дані.
   * **Edit.cshtml:** Подання для редагування обраного запису, яке попередньо заповнюється існуючими даними маршруту.
   * **Details.cshtml:** Відображає деталі вибраного маршруту.
   * **Delete.cshtml:** Подання для підтвердження видалення маршруту.
3. **Реалізація контролера:**
   * **TransportController:** Виконує CRUD‑операції з використанням ApplicationDbContext.
     + **Index:** Асинхронно завантажує список маршрутів для відображення.
     + **Create:** Методи GET і POST для створення нового запису.
     + **Edit:** Методи GET і POST для редагування існуючого запису.
     + **Details:** Завантаження та відображення повної інформації про маршрут.
     + **Delete:** Методи GET (для підтвердження) та POST (для видалення) маршруту.
   * Завдяки початковому заповненню бази (seed data), застосунок автоматично містить три записи з моменту розгортання.

## **Функціональність**

* **Початкове заповнення даними:**

Застосунок використовує метод OnModelCreating для автоматичного додавання трьох записів у таблицю, що дає змогу відразу побачити наявні маршрути при першому завантаженні.

* **Базові CRUD‑операції:**

Завдяки реалізації методів для створення, редагування, перегляду деталей і видалення, користувач може повноцінно керувати маршрутами. Інтерфейс забезпечує зручне переглядання та маніпулювання даними.

* **Інтерактивний інтерфейс:**

Список маршрутів відображається у вигляді таблиці зі зручним відображенням даних і кнопками для виконання операцій, що покращує користувацький досвід.

## **Висновок**

* **Автоматизоване початкове заповнення:**

Додавання seed‑даних дозволяє одразу продемонструвати функціонал застосунку, що особливо корисно для тестування та демонстрації роботи.

* **Комплексність функціоналу:**

Завдяки реалізації повного набору CRUD‑операцій застосунок дозволяє ефективно керувати даними маршрутів, що сприяє підвищенню зручності адміністрування інформацією.

* **Архітектурна чистота:**

Розділення логіки між моделлю, представленнями і контролером згідно з принципами MVC сприяє легкій підтримці та можливості подальшого розширення функціоналу, забезпечуючи сучасний та зручний інтерфейс для користувача.

# **Завдання 9**

Модифікувати минуле завдання додавши до структури таблиці БД будь яке поле за змістом предметної сфери. Перевірити працездатність застосунку

## **Текст програми:**

**Index.cshtml:**

@model IEnumerable<Task\_9.Models.Transport>

@{

ViewData["Title"] = "Список маршрутів";

// Модель – перелік маршрутів

}

<h1>Список маршрутів</h1>

<p>

<!-- Кнопка переходу до створення нового маршруту -->

<a class="btn btn-primary" asp-action="Create">Додати новий маршрут</a>

</p>

<table class="table">

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність (км)</th>

<th>Час в дорозі (хв)</th>

<th>К-сть зупинок</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var item in Model)

{

<tr>

<td>@item.Id</td>

<td>@item.VidTransportu</td>

<td>@item.NomMarshruta</td>

<td>@item.ProtjazhnistMarshruta</td>

<td>@item.ChasVDorozi</td>

<td>@item.KilkistZupynok</td>

<td>

<!-- Дії: редагування, деталі, видалення -->

<a class="btn btn-secondary" asp-action="Edit" asp-route-id="@item.Id">Редагувати</a> |

<a class="btn btn-info" asp-action="Details" asp-route-id="@item.Id">Деталі</a> |

<a class="btn btn-danger" asp-action="Delete" asp-route-id="@item.Id">Видалити</a>

</td>

</tr>

}

</tbody>

</table>

**Edit.cshtml:**

@model Task\_9.Models.Transport

@{

ViewData["Title"] = "Редагувати маршрут";

// Дані маршруту для редагування

}

<h1>Редагувати маршрут</h1>

<form asp-action="Edit" method="post">

<input type="hidden" asp-for="Id" />

<div class="form-group">

<label>Вид транспорту</label>

<input asp-for="VidTransportu" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Номер маршруту</label>

<input asp-for="NomMarshruta" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Протяжність (км)</label>

<input asp-for="ProtjazhnistMarshruta" class="form-control" type="number" step="0.01" />

</div>

<div class="form-group">

<label>Час в дорозі (хв)</label>

<input asp-for="ChasVDorozi" class="form-control" type="number" />

</div>

<div class="form-group">

<label>Кількість зупинок</label>

<input asp-for="KilkistZupynok" class="form-control" type="number" />

</div>

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

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

**Details.cshtml:**

@model Task\_9.Models.Transport

@{

ViewData["Title"] = "Деталі маршруту";

// Детальна інформація про маршрут

}

<div class="details-card">

<h2>Деталі маршруту</h2>

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.VidTransportu</p>

<p><strong>Номер маршруту:</strong> @Model.NomMarshruta</p>

<p><strong>Протяжність (км):</strong> @Model.ProtjazhnistMarshruta</p>

<p><strong>Час в дорозі (хв):</strong> @Model.ChasVDorozi</p>

<p><strong>Кількість зупинок:</strong> @Model.KilkistZupynok</p>

<hr />

<a class="btn btn-secondary" asp-action="Index">Назад</a>

<a class="btn btn-primary" asp-action="Edit" asp-route-id="@Model.Id">Редагувати</a>

</div>

**Delete.cshtml:**

@model Task\_9.Models.Transport

@{

ViewData["Title"] = "Видалити маршрут";

// Підтвердження видалення маршруту

}

<div class="details-card">

<h2>Видалення маршруту</h2>

<p>Ви дійсно бажаєте видалити цей маршрут?</p>

<p><strong>Id:</strong> @Model.Id</p>

<p><strong>Вид транспорту:</strong> @Model.VidTransportu</p>

<p><strong>Номер маршруту:</strong> @Model.NomMarshruta</p>

<p><strong>Протяжність (км):</strong> @Model.ProtjazhnistMarshruta</p>

<p><strong>Час в дорозі (хв):</strong> @Model.ChasVDorozi</p>

<p><strong>Кількість зупинок:</strong> @Model.KilkistZupynok</p>

<hr />

<form asp-action="Delete" method="post">

<input type="hidden" asp-for="Id" />

<button type="submit" class="btn btn-danger">Видалити</button>

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

</div>

**Create.cshtml:**

@model Task\_9.Models.Transport

@{

ViewData["Title"] = "Додати маршрут";

// Форма створення нового маршруту

}

<h1>Додати маршрут</h1>

<form asp-action="Create" method="post">

<div class="form-group">

<label>Вид транспорту</label>

<input asp-for="VidTransportu" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Номер маршруту</label>

<input asp-for="NomMarshruta" class="form-control" type="text" />

</div>

<div class="form-group">

<label>Протяжність (км)</label>

<input asp-for="ProtjazhnistMarshruta" class="form-control" type="number" step="0.01" />

</div>

<div class="form-group">

<label>Час в дорозі (хв)</label>

<input asp-for="ChasVDorozi" class="form-control" type="number" />

</div>

<div class="form-group">

<label>Кількість зупинок</label>

<input asp-for="KilkistZupynok" class="form-control" type="number" />

</div>

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

<a asp-action="Index" class="btn btn-secondary">Назад</a>

</form>

**Transport.cs:**

namespace Task\_9.Models

{

public class Transport

{

public int Id { get; set; } // Первинний ключ

public string VidTransportu { get; set; } // "Tr", "Tl", "A"

public string NomMarshruta { get; set; } // Номер маршруту (якщо є букви, тип string)

public float ProtjazhnistMarshruta { get; set; } // Протяжність (км)

public int ChasVDorozi { get; set; } // Час в дорозі (хв)

public int KilkistZupynok { get; set; } // Кількість зупинок

}

}

**ApplicationDbContext.cs:**

using Microsoft.EntityFrameworkCore;

using Task\_9.Models;

namespace Task\_9.Data

{

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

// Зберігання маршрутів

public DbSet<Transport> Transports { get; set; }

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Task\_9.Data;

using Task\_9.Models;

namespace Task\_9.Controllers

{

public class TransportController : Controller

{

private readonly ApplicationDbContext \_context;

public TransportController(ApplicationDbContext context)

{

\_context = context; // Доступ до БД

}

// GET: Transport – вивід списку маршрутів

public async Task<IActionResult> Index()

{

var transports = await \_context.Transports.ToListAsync();

return View(transports);

}

// GET: Transport/Details/5 – деталі маршруту

public async Task<IActionResult> Details(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FirstOrDefaultAsync(m => m.Id == id);

if (transport == null)

return NotFound();

return View(transport);

}

// GET: Transport/Create – форма додавання маршруту

public IActionResult Create()

{

return View();

}

// POST: Transport/Create – збереження нового маршруту

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("Id,VidTransportu,NomMarshruta,ProtjazhnistMarshruta,ChasVDorozi,KilkistZupynok")] Transport transport)

{

if (ModelState.IsValid)

{

\_context.Add(transport);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(transport);

}

// GET: Transport/Edit/5 – форма редагування маршруту

public async Task<IActionResult> Edit(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FindAsync(id);

if (transport == null)

return NotFound();

return View(transport);

}

// POST: Transport/Edit/5 – збереження змін маршруту

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("Id,VidTransportu,NomMarshruta,ProtjazhnistMarshruta,ChasVDorozi,KilkistZupynok")] Transport transport)

{

if (id != transport.Id)

return NotFound();

if (ModelState.IsValid)

{

try

{

\_context.Update(transport);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!TransportExists(transport.Id))

return NotFound();

else

throw;

}

return RedirectToAction(nameof(Index));

}

return View(transport);

}

// GET: Transport/Delete/5 – підтвердження видалення маршруту

public async Task<IActionResult> Delete(int? id)

{

if (id == null)

return NotFound();

var transport = await \_context.Transports.FirstOrDefaultAsync(m => m.Id == id);

if (transport == null)

return NotFound();

return View(transport);

}

// POST: Transport/Delete/5 – видалення маршруту

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var transport = await \_context.Transports.FindAsync(id);

if (transport != null)

{

\_context.Transports.Remove(transport);

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

// Перевірка наявності маршруту

private bool TransportExists(int id)

{

return \_context.Transports.Any(e => e.Id == id);

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>@ViewData["Title"] - Task\_9</title>

<link rel="stylesheet" href="~/css/site.css" />

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

</head>

<body>

<!-- Фонове слайдшоу -->

<div class="background-slideshow">

<div class="slide slide1"></div>

<div class="slide slide2"></div>

<div class="slide slide3"></div>

<div class="slide slide4"></div>

<div class="slide slide5"></div>

</div>

<header>

<nav class="navbar">

<!-- Навігація -->

<a class="nav-link" href="@Url.Action("Index", "Transport")">Головна</a>

<a class="nav-link" href="@Url.Action("Create", "Transport")">Додати запис</a>

</nav>

</header>

<div class="container">

@RenderBody() <!-- Основний контент -->

</div>

<footer>

<p>&copy; @DateTime.Now.Year - Task\_9</p>

</footer>

@RenderSection("Scripts", required: false)

</body>

</html>

**site.css:**

/\* Загальні стилі для сторінки \*/

body, html {

margin: 0;

padding: 0;

height: 100%;

font-family: Arial, sans-serif;

}

/\* Фонове слайдшоу \*/

.background-slideshow {

position: fixed;

top: 0;

left: 0;

width: 100%;

height: 100%;

z-index: -1;

overflow: hidden;

}

.background-slideshow .slide {

position: absolute;

width: 100%;

height: 100%;

background-size: cover;

background-position: center center;

opacity: 0;

animation: fade 25s infinite;

}

.background-slideshow .slide1 {

background-image: url('https://i.imgur.com/RGka78l.jpg');

animation-delay: 0s;

}

.background-slideshow .slide2 {

background-image: url('https://wallpaper-mania.com/wp-content/uploads/2018/09/High\_resolution\_wallpaper\_background\_ID\_77701304695.jpg');

animation-delay: 5s;

}

.background-slideshow .slide3 {

background-image: url('https://wallpapers.com/images/hd/1920-x-1080-nature-desktop-chafsybseyq3lp0n.jpg');

animation-delay: 10s;

}

.background-slideshow .slide4 {

background-image: url('https://wallpapers.com/images/hd/1920x1080-full-hd-nature-landscape-54q8pleyp8lhhbu7.jpg');

animation-delay: 15s;

}

.background-slideshow .slide5 {

background-image: url('https://images.unsplash.com/photo-1469474968028-56623f02e42e?fm=jpg&q=60&w=3000&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxzZWFyY2h8N3x8MTkyMHgxMDgwJTIwd2FsbHBhcGVyfGVufDB8fDB8fHww');

animation-delay: 20s;

}

@keyframes fade {

0% { opacity: 0; }

10% { opacity: 1; }

30% { opacity: 1; }

40% { opacity: 0; }

100% { opacity: 0; }

}

/\* Контейнер основного контенту \*/

.container {

position: relative;

z-index: 2;

margin: 20px;

background: rgba(255, 255, 255, 0.85);

padding: 20px;

border-radius: 8px;

}

/\* Стилі для навігаційного меню \*/

.navbar {

background: rgba(0, 0, 0, 0.6);

padding: 10px 20px;

position: relative;

z-index: 3;

}

.navbar .nav-link {

color: #fff;

text-decoration: none;

margin-right: 15px;

transition: color 0.3s ease;

}

.navbar .nav-link:hover {

color: #f0a500;

}

/\* Футер \*/

footer {

text-align: center;

padding: 10px;

background: rgba(0, 0, 0, 0.6);

color: #fff;

position: fixed;

bottom: 0;

width: 100%;

z-index: 3;

}

/\* Анімації для форм \*/

form {

animation: slideIn 1s ease-out;

}

@keyframes slideIn {

from { opacity: 0; transform: translateY(-20px); }

to { opacity: 1; transform: translateY(0); }

}

/\* Стилізація таблиць \*/

.table {

width: 100%;

border-collapse: collapse;

animation: fadeIn 1s ease-in;

}

@keyframes fadeIn {

from { opacity: 0; }

to { opacity: 1; }

}

.table th, .table td {

padding: 10px;

border: 1px solid #ccc;

}

/\* Стилі кнопок \*/

.btn {

padding: 8px 12px;

border: none;

border-radius: 4px;

cursor: pointer;

margin: 5px;

transition: background 0.3s;

}

.btn-primary {

background: #007bff;

color: #fff;

}

.btn-primary:hover {

background: #0056b3;

}

.btn-secondary {

background: #6c757d;

color: #fff;

}

.btn-secondary:hover {

background: #5a6268;

}

.btn-danger {

background: #dc3545;

color: #fff;

}

.btn-danger:hover {

background: #c82333;

}

/\* Додаткові стилі для формових елементів \*/

.form-group {

margin-bottom: 15px;

}

label {

display: block;

margin-bottom: 5px;

}

.form-control {

width: 100%;

padding: 8px;

box-sizing: border-box;

}

/\* Картка для деталей і видалення \*/

.details-card {

margin: 20px auto;

max-width: 600px;

background: rgba(255, 255, 255, 0.9);

padding: 20px;

border-radius: 8px;

box-shadow: 0 2px 6px rgba(0, 0, 0, 0.2);

animation: fadeIn 1.2s ease-in-out;

}

.details-card h2 {

margin-top: 0;

margin-bottom: 15px;

}

.details-card p {

margin: 10px 0;

}

## **Результати роботи програми:**

Изображение выглядит как снимок экрана, текст, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.1. Результати роботи виконання Завдання 9 (Створена БД)

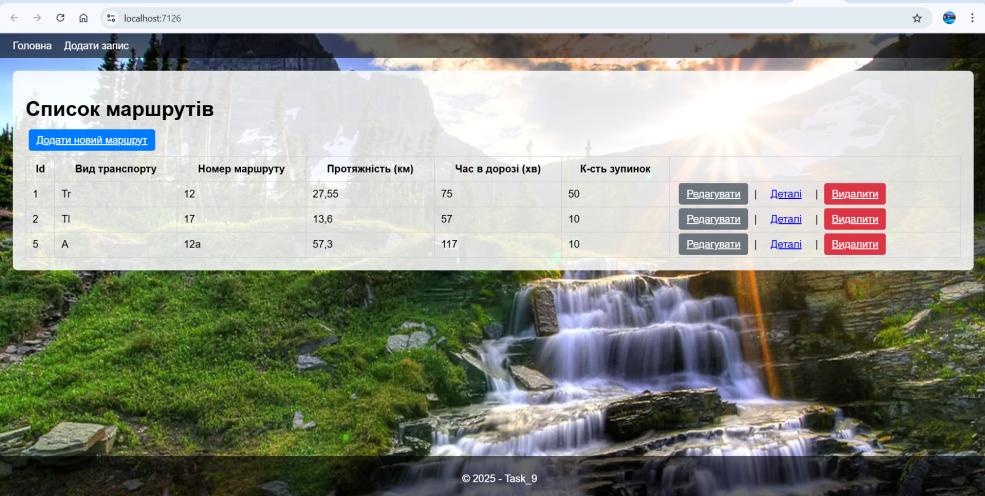


Рисунок 9.2. Результати роботи виконання Завдання 9 (Головна сторінка)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.3. Результати роботи виконання Завдання 9 (Додавання)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.4. Результати роботи виконання Завдання 9 (Результати додавання)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.5. Результати роботи виконання Завдання 9 (Редагування)

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.6. Результати роботи виконання Завдання 9 (Результати редагування)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.7. Результати роботи виконання Завдання 9 (Деталі)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.8. Результати роботи виконання Завдання 9 (Видалення)

Изображение выглядит как текст, снимок экрана, Мультимедийное программное обеспечение, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 9.9. Результати роботи виконання Завдання 9 (Результати видалення)

## **Алгоритм побудови кодів**

1. **Модифікація моделі та контексту даних:**
   * **Модель Transport:**

Розширено клас моделі Transport, додавши нове поле **KilkistZupynok** (кількість зупинок) поряд з уже наявними властивостями: Id, VidTransportu, NomMarshruta, ProtjazhnistMarshruta та ChasVDorozi.

* + **Контекст ApplicationDbContext:**

Налаштовано DbContext для роботи з таблицею маршрутів. Жодних додаткових змін у контексті не потрібно, оскільки розширення моделі автоматично відображається в структурі таблиці за умов належної міграції.

1. **Розробка представлень (Views):**
   * **Index.cshtml:**

Подання для відображення списку маршрутів доповнено стовпчиком для нового поля. Таблиця відображає значення кількості зупинок для кожного запису.

* + **Create.cshtml та Edit.cshtml:**

Форма введення/редагування маршруту доповнена новим полем «Кількість зупинок». Це поле забезпечує можливість введення даних щодо зупинок.

* + **Details.cshtml та Delete.cshtml:**

Подання для перегляду деталей та підтвердження видалення маршруту також відображають значення нового поля.

1. **Реалізація контролера:**
   * У **TransportController** реалізовано всі типові CRUD‑операції (створення, редагування, видалення, перегляд).
   * Методи **Create**, **Edit**, **Details** та **Delete** працюють із розширеною моделлю, що включає поле **KilkistZupynok**, забезпечуючи коректну обробку та відображення додаткових даних.

## **Функціональність**

* **Повноцінні операції із записами:**

Завдяки використанню розширеної моделі, застосунок забезпечує створення, редагування, видалення та перегляд маршрутів разом із відображенням нової інформації – кількості зупинок.

* **Відображення додаткових даних:**

Нове поле додається до таблиці, що дозволяє користувачу одразу бачити додаткову інформацію про маршрут, що може бути важливою з точки зору предметної сфери.

* **Уніфікація інтерфейсу:**

Всі подання (створення, редагування, перегляд деталей і видалення) оновлено таким чином, щоб коректно працювати з розширеною структурою даних, забезпечуючи зручний і послідовний користувацький досвід.

## **Висновок**

* **Розширення предметної логіки:**

Додавши додаткове поле (кількість зупинок), застосунок набув більшої інформативності, що дозволяє краще відображати особливості транспортної системи або маршруту.

* **Підвищення гнучкості:**

Модифікація моделі та відповідне оновлення представлень і контролера демонструють, наскільки просто можна адаптувати систему під нові вимоги за допомогою EF і ASP.NET MVC.

* **Стабільна працездатність:**

Перевірка працездатності застосунку після внесення змін підтверджує, що додаткове поле інтегровано правильно, і всі CRUD‑операції працюють без збоїв, що є ознакою надійної розробки.

# **Загальний висновок**

Лабораторна робота №6 демонструє практичне впровадження архітектури MVC у веб-додатках на базі ASP.NET. Протягом роботи було послідовно розроблено застосунок, який починається з простого введення даних через як слабко, так і суворо типізовані форми, забезпечуючи наочне представлення введеної інформації.

Подальша інтеграція з базою даних MySQL дозволила додати функціонал збереження записів, їх відображення, а також реалізувати операції видалення як усіх записів, так і окремих елементів. Використання Entity Framework значно спростило процес роботи з БД, дозволивши динамічно оновлювати інформацію на сторінці.

Застосування сучасних технологій верстки (Bootstrap, CSS-анімації) сприяло створенню зручного і привабливого інтерфейсу, що позитивно впливає на користувацький досвід. Чітке розділення логіки між моделлю, представленням і контролером забезпечує модульність та легкість майбутнього розширення застосунку.

Ця лабораторна робота дала змогу оволодіти основами побудови сучасного серверного застосунку, інтегруючи всі ключові елементи: введення даних, їх обробку, збереження та управління базою даних, що є важливим аспектом у розробці інформаційних систем.