Міністерство освіти України

Національний технічний університет "ХПІ"

кафедра "Комп’ютерних наук та інформаційних технологій"

**Звіт**

**Лабораторна робота 7**

з дисципліни "Проєктування серверних застосунків"

Виконав: студент групи КН-922В

Мельник Б. В.

Перевірив:

Івашко А. В.

Харків 2025

**Зміст**

[Завдання 1 3](#_Toc195967037)

[Текст програми 3](#_Toc195967038)

[Результати роботи програми 16](#_Toc195967039)

[Алгоритм побудови кодів 18](#_Toc195967040)

[Функціональність 19](#_Toc195967041)

[Висновок 20](#_Toc195967042)

[Завдання 2 21](#_Toc195967043)

[Текст програми 21](#_Toc195967044)

[Результати роботи програми 37](#_Toc195967045)

[Алгоритм побудови кодів 39](#_Toc195967046)

[Функціональність 40](#_Toc195967047)

[Висновок 40](#_Toc195967048)

[Завдання 3 42](#_Toc195967049)

[Текст програми 42](#_Toc195967050)

[Результати роботи програми 56](#_Toc195967051)

[Алгоритм побудови кодів 58](#_Toc195967052)

[Функціональність 59](#_Toc195967053)

[Висновок 59](#_Toc195967054)

[Завдання 4 60](#_Toc195967055)

[Текст програми 60](#_Toc195967056)

[Результати роботи програми 76](#_Toc195967057)

[Алгоритм побудови кодів 77](#_Toc195967058)

[Функціональність 78](#_Toc195967059)

[Висновок 79](#_Toc195967060)

[Загальний Висновок 80](#_Toc195967061)

**Тема:** Використання засобів валідації (перевірки на коректність) інформації під час її обробки в MVC WEB-застосунку.

**Мета роботи:** придбати практику у використанні різноманітних засобів валі-дації інформації під час її додання редагування та аналізу у формах та БД МѴС застосунків.

**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**

|  |  |
| --- | --- |
| **№ Варіанту** | **Місце розміщення повідомлень про помилки валідації** |
| 12 | Зверху від першої половини відповідних полів форми та праворуч від другої половини відповідних полів форми |

# **Завдання 1**

Модифікувати підсумкові завдання (4 чи 6 та 9) минулої роботи, дода-вши до їхніх проектів засоби валідації даних через контролери. Розміщення повідомлень про помилки валідації витримати згідно свого варіанту.

**Модифікація Завдання 6 з минулої лабораторної роботи:**

## **Текст програми:**

**SQL-запит:**

USE TransportDbLab7; -- вибір БД

CREATE TABLE Transports (

Id INT PRIMARY KEY IDENTITY(1,1), -- автоінкремент

TransportType NVARCHAR(2) NOT NULL, -- Tr / Tl / A

RouteNumber NVARCHAR(10) NOT NULL, -- номер

RouteDistance FLOAT NOT NULL, -- км

TravelTime INT NOT NULL -- хв

);

-- тестові записи

INSERT INTO Transports (TransportType, RouteNumber, RouteDistance, TravelTime)

VALUES ('Tr','12A',14.5,30),

('Tl','5b', 8.2,25),

('A', '101',20.0,40);

**Index.cshtml:**

@model Task\_1.Models.Transport

@{

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

var transports = ViewBag.Transports as List<Task\_1.Models.Transport>;

}

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

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

<!-- TransportType -->

<div class="mb-3">

<span asp-validation-for="TransportType" class="text-danger" style="display:block;"></span>

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

<input asp-for="TransportType" class="form-control" placeholder="Tr / Tl / A" />

</div>

<!-- RouteNumber -->

<div class="mb-3">

<span asp-validation-for="RouteNumber" class="text-danger" style="display:block;"></span>

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

<input asp-for="RouteNumber" class="form-control" placeholder="12A" />

</div>

<!-- RouteDistance -->

<div class="mb-3">

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

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

<span asp-validation-for="RouteDistance" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- TravelTime -->

<div class="mb-3">

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

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

<span asp-validation-for="TravelTime" class="text-danger" style="margin-left:10px;"></span>

</div>

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

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

</form>

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

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

</form>

@if (transports?.Any() == true)

{

<hr />

<h2>Вміст таблиці Transports</h2>

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

<thead>

<tr>

<th>Id</th>

<th>Вид</th>

<th>Номер</th>

<th>Дистанція</th>

<th>Час</th>

<th></th>

</tr>

</thead>

<tbody>

@foreach (var t in transports)

{

<tr>

<td>@t.Id</td>

<td>@t.TransportType</td>

<td>@t.RouteNumber</td>

<td>@t.RouteDistance</td>

<td>@t.TravelTime</td>

<td>

<a asp-action="Edit" asp-route-id="@t.Id" class="btn btn-sm btn-info btn-edit">Редагувати</a>

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

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

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

</form>

</td>

</tr>

}

</tbody>

</table>

}

else

{

<hr />

<h2>Вміст таблиці Transports</h2>

<p>Таблиця порожня.</p>

}

**Edit.cshtml:**

@model Task\_1.Models.Transport

@{

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

}

<h1>Редагування запису</h1>

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

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

<!-- TransportType -->

<div class="mb-3">

<span asp-validation-for="TransportType" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- RouteNumber -->

<div class="mb-3">

<span asp-validation-for="RouteNumber" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- RouteDistance -->

<div class="mb-3">

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

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

<span asp-validation-for="RouteDistance" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- TravelTime -->

<div class="mb-3">

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

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

<span asp-validation-for="TravelTime" class="text-danger" style="margin-left:10px;"></span>

</div>

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

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

</form>

**Transport.cs:**

namespace Task\_1.Models

{

// POCO‑клас без атрибутів

public class Transport

{

public int Id { get; set; }

public string TransportType { get; set; } = string.Empty; // Tr / Tl / A

public string RouteNumber { get; set; } = string.Empty; // до 10 симв.

public double RouteDistance { get; set; } // км > 0

public int TravelTime { get; set; } // хв >= 1

}

}

**TransportContext.cs:**

using Microsoft.EntityFrameworkCore;

namespace Task\_1.Models

{

public class TransportContext : DbContext

{

public TransportContext(DbContextOptions<TransportContext> options)

: base(options) { }

public DbSet<Transport> Transports => Set<Transport>();

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_1.Models;

using Microsoft.EntityFrameworkCore;

namespace Task\_1.Controllers

{

public class TransportController : Controller

{

private readonly TransportContext \_db;

public TransportController(TransportContext db) => \_db = db;

// -------------------- GET --------------------

public IActionResult Index()

{

ViewBag.Transports = \_db.Transports.AsNoTracking().ToList();

return View(new Transport());

}

// -------------------- POST: створення --------------------

[HttpPost]

public IActionResult Index(Transport transport)

{

Validate(transport); // ➊ власна перевірка

if (!ModelState.IsValid) // ➋ якщо є помилки – повертаємо форму

{

ViewBag.Transports = \_db.Transports.AsNoTracking().ToList();

return View(transport);

}

\_db.Transports.Add(transport); // ➌ усе ґуд – зберігаємо

\_db.SaveChanges();

return RedirectToAction(nameof(Index));

}

// -------------------- GET: редагування --------------------

public IActionResult Edit(int id)

{

var t = \_db.Transports.Find(id);

return t is null ? RedirectToAction(nameof(Index)) : View(t);

}

// -------------------- POST: редагування --------------------

[HttpPost]

public IActionResult Edit(Transport transport)

{

Validate(transport);

if (!ModelState.IsValid) return View(transport);

\_db.Transports.Update(transport);

\_db.SaveChanges();

return RedirectToAction(nameof(Index));

}

// -------------------- POST: видалити все --------------------

[HttpPost]

public IActionResult DeleteAll()

{

\_db.Transports.RemoveRange(\_db.Transports);

\_db.SaveChanges();

return RedirectToAction(nameof(Index));

}

// -------------------- POST: видалити один --------------------

[HttpPost]

public IActionResult DeleteRecord(int id)

{

var rec = \_db.Transports.Find(id);

if (rec != null)

{

\_db.Transports.Remove(rec);

\_db.SaveChanges();

}

return RedirectToAction(nameof(Index));

}

// -------------------- Приватна перевірка --------------------

private void Validate(Transport t)

{

// Вид транспорту

if (string.IsNullOrWhiteSpace(t.TransportType))

ModelState.AddModelError(nameof(t.TransportType),

"Вид транспорту є обов’язковим!");

else if (!new[] { "Tr", "Tl", "A" }.Contains(t.TransportType))

ModelState.AddModelError(nameof(t.TransportType),

"Допустимі значення: Tr, Tl або A");

// Номер маршруту

if (string.IsNullOrWhiteSpace(t.RouteNumber))

ModelState.AddModelError(nameof(t.RouteNumber),

"Номер маршруту обов’язковий");

else if (t.RouteNumber.Length > 10)

ModelState.AddModelError(nameof(t.RouteNumber),

"Максимум 10 символів");

// Дистанція

if (t.RouteDistance <= 0)

ModelState.AddModelError(nameof(t.RouteDistance),

"Протяжність має бути > 0 км");

// Час у дорозі

if (t.TravelTime < 1)

ModelState.AddModelError(nameof(t.TravelTime),

"Час має бути не менше 1 хв");

}

}

}

**\_Layout.cshtml:**

<!DOCTYPE html>

<html lang="uk">

<head>

<meta charset="utf-8" />

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

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

<!-- Підключення Bootstrap (версія 5.x) -->

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

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2B9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous">

</head>

<body>

<div class="overlay">

<div class="container">

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

@RenderBody()

</main>

</div>

</div>

<!-- Підключення Bootstrap bundle (включно з Popper.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>

<!-- Підключення jQuery -->

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

integrity="sha256-/xUj+3OJ+Z+6ZQOJwDS0hp9Z9+4Cl0MiQFv2KTF3n3s="

crossorigin="anonymous"></script>

<!-- Плагіни валідації jQuery -->

<script src="https://cdn.jsdelivr.net/npm/jquery-validation@1.19.5/dist/jquery.validate.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/jquery-validation-unobtrusive@3.2.12/dist/jquery.validate.unobtrusive.min.js"></script>

</body>

</html>

**site.css:**

/\* Приклад фонового зображення \*/

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;

}

}

/\* Прозорий шар \*/

.overlay {

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

min-height: 100vh;

padding: 20px;

}

/\* Анімація заголовків \*/

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);

}

}

/\* Стилі форм \*/

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;

}

}

/\* Поля вводу \*/

.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);

}

/\* Кнопки \*/

.btn-primary:hover {

background-color: #0056b3;

border-color: #004085;

}

.btn-secondary:hover {

background-color: #5a6268;

border-color: #545b62;

}

.btn-danger:hover {

background-color: #bb2d3b;

}

/\* Кнопка "Редагувати" \*/

.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;

}

/\* Кнопка "Видалити" \*/

.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;

}

/\* Вбудована форма \*/

.inline-form {

display: inline-block;

margin: 0;

}

/\* Таблиця \*/

.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;

}

/\* Для .text-danger \*/

.text-danger {

color: red !important;

}

## **Результати роботи програми:**

Изображение выглядит как текст, снимок экрана, небо, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.1. Загальний вигляд форми

Изображение выглядит как текст, снимок экрана, небо, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.2. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, снимок экрана, небо, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.3. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, дерево, небо, на открытом воздухе

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.4. Розміщення помилок у формі редагування даних

Изображение выглядит как текст, дерево, небо, на открытом воздухе

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 1.5. Розміщення помилок у формі редагування даних

## **Алгоритм побудови кодів**

1. **Створення та заповнення БД**
   * Ініціалізуємо *TransportDbLab7* і таблицю **Transports**, одразу додаючи тестові записи, щоб мати дані для перевірки.
2. **Опис моделі** (Transport.cs)
   * Створюємо POCO‑клас без атрибутів — це спрощує подальшу ручну (контролерну) перевірку даних.
3. **Налаштування контексту** (TransportContext.cs)
   * Реєструємо DbSet<Transport> і підключаємо контекст у Program.cs / Startup.cs (AddDbContext) для роботи з LocalDB через EF Core.
4. **Реалізація контролера** (TransportController.cs)
   * **GET Index** — віддає порожню модель та список записів у ViewBag для таблиці.
   * **POST Index** —
     + Викликає власний метод Validate() із ручними правилами (тип, номер, дистанція, час).
     + Якщо ModelState містить помилки — повертає форму з підсвіченими повідомленнями; інакше додає запис до БД.
   * **Edit / Delete / DeleteAll** — стандартні дії CRUD.
5. **Валідація через контролер** (Validate()):
   * Перевірка кожного поля й додавання помилок до ModelState з текстами українською.
   * Розміщення повідомлень здійснюється у **двох зонах** (варіант №12):
     + span з класом text-danger — **зверху** над полем першої половини.
     + span з відступом справа (за допомогою margin-left) — **праворуч** другої половини полів.
6. **Створення подань** (Index.cshtml, Edit.cshtml)
   * Використовуємо тег‑хелпери asp-validation-for і asp-for, щоб автоматично виводити повідомлення, які заповнив контролер.
7. **Оформлення інтерфейсу** (\_Layout.cshtml, site.css)
   * Підключаємо Bootstrap 5, jQuery та unobtrusive‑validation,
   * додаємо анімації й напівпрозорий шар для читабельності тексту на фоні.

## **Функціональність**

* **Додавання**: користувач вводить параметри маршруту; контролер перевіряє їх та зберігає у БД.
* **Перелік**: усі записи показуються у таблиці; доступні кнопки *Редагувати* та *Видалити*.
* **Редагування**: вибір запису відкриває форму з попередніми значеннями; після збереження дані оновлюються.
* **Масове та вибіркове видалення**: можна стерти один запис або всю таблицю.
* **Валідація**:
  + **Тип транспорту** — лише Tr, Tl, A;
  + **Номер маршруту** — обов’язковий, ≤ 10 символів;
  + **Протяжність** — > 0 км;
  + **Час у дорозі** — ≥ 1 хв.  
    Помилки одразу відображаються на формі, не звертаючись до БД.

## **Висновок**

У результаті модифікації Завдання 6 було впроваджено **контролерну валідацію** відповідно до варіанту № 12.  
Рішення забезпечує повноцінний CRUD‑функціонал із чітким розташуванням повідомлень про помилки, що підвищує зручність користувача та гарантує коректність даних ще до їхнього збереження в базі. Отже, поставлена мета досягнута, а застосунок став надійнішим і візуально приємним.

# **Завдання 2**

Модифікувати підсумкові завдання (4 чи 6 та 9) минулої роботи, дода-вши до їхніх проектів засоби валідації даних через контролери. Розміщення повідомлень про помилки валідації витримати згідно свого варіанту.

**Модифікація Завдання 9 з минулої лабораторної роботи:**

## **Текст програми:**

**SQL-запит:**

-- Переходимо до потрібної бази

USE TransportDbLab7\_1;

-- Створюємо таблицю для зберігання інформації про маршрути транспорту

CREATE TABLE Transports

(

Id INT PRIMARY KEY IDENTITY(1,1), -- автоінкремент‑Id (унікальний ключ)

VidTransportu NVARCHAR(2) NOT NULL, -- тип транспорту: Tr, Tl або A

NomMarshruta NVARCHAR(15) NOT NULL, -- номер маршруту

ProtjazhnistMarshruta FLOAT NOT NULL, -- довжина маршруту, км

ChasVDorozi INT NOT NULL, -- час у дорозі, хв

KilkistZupynok INT NOT NULL -- кількість зупинок

);

-- Додаємо стартові дані

INSERT INTO Transports

(VidTransportu, NomMarshruta, ProtjazhnistMarshruta, ChasVDorozi, KilkistZupynok)

VALUES

('Tr', '12A', 14.5, 30, 5), -- трамвай 12A

('Tl', '5b', 8.2, 25, 3), -- тролейбус 5b

('A', '101', 20.0, 40, 7); -- автобус 101

**Index.cshtml:**

@model IEnumerable<Task\_2.Models.Transport>

@{

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

}

<h1>Список маршрутів</h1>

<p>

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

</p>

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

<thead>

<tr>

<th>Id</th>

<th>Вид транспорту</th>

<th>Номер маршруту</th>

<th>Протяжність (км)</th>

<th>Час (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>

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

<form asp-action="Delete" method="get" class="inline-form">

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

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

</form>

</td>

</tr>

}

</tbody>

</table>

**Edit.cshtml:**

@model Task\_2.Models.Transport

@{

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

}

<h1>Редагувати маршрут</h1>

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

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

<div class="mb-3">

<span asp-validation-for="VidTransportu" class="text-danger" style="display:block;"></span>

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

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

</div>

<div class="mb-3">

<span asp-validation-for="NomMarshruta" class="text-danger" style="display:block;"></span>

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

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

</div>

<div class="mb-3">

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

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

<span asp-validation-for="ProtjazhnistMarshruta" class="text-danger" style="margin-left:10px;"></span>

</div>

<div class="mb-3">

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

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

<span asp-validation-for="ChasVDorozi" class="text-danger" style="margin-left:10px;"></span>

</div>

<div class="mb-3">

<label class="form-label">Кількість зупинок</label>

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

<span asp-validation-for="KilkistZupynok" class="text-danger" style="margin-left:10px;"></span>

</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\_2.Models.Transport

@{

ViewData["Title"] = "Деталі 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>

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

<hr />

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

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

</div>

**Delete.cshtml:**

@model Task\_2.Models.Transport

@{

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

}

<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\_2.Models.Transport

@{

ViewData["Title"] = "Додати маршрут";

}

<h1>Додати маршрут</h1>

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

<!-- 1) Вид транспорту -->

<div class="mb-3">

<!-- Помилка над полем -->

<span asp-validation-for="VidTransportu" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- 2) Номер маршруту -->

<div class="mb-3">

<span asp-validation-for="NomMarshruta" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- 3) Протяжність -->

<div class="mb-3">

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

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

<span asp-validation-for="ProtjazhnistMarshruta" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 4) Час в дорозі -->

<div class="mb-3">

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

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

<span asp-validation-for="ChasVDorozi" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 5) Кількість зупинок -->

<div class="mb-3">

<label class="form-label">Кількість зупинок</label>

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

<span asp-validation-for="KilkistZupynok" class="text-danger" style="margin-left:10px;"></span>

</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\_2.Models

{

public class Transport

{

public int Id { get; set; }

public string VidTransportu { get; set; } = string.Empty; // Tr / Tl / A

public string NomMarshruta { get; set; } = string.Empty; // ≤15 симв.

public double ProtjazhnistMarshruta { get; set; } // >0

public int ChasVDorozi { get; set; } // ≥1

public int KilkistZupynok { get; set; } // ≥0

}

}

**ApplicationDbContext.cs:**

using Microsoft.EntityFrameworkCore;

namespace Task\_2.Models

{

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options) { }

public DbSet<Transport> Transports => Set<Transport>();

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Task\_2.Models;

namespace Task\_2.Controllers

{

public class TransportController : Controller

{

private readonly ApplicationDbContext \_db;

public TransportController(ApplicationDbContext db) => \_db = db;

// ========== INDEX ==========

public async Task<IActionResult> Index()

=> View(await \_db.Transports.AsNoTracking().ToListAsync());

// ========== DETAILS ==========

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

{

if (id is null) return NotFound();

var t = await \_db.Transports.FindAsync(id);

return t is null ? NotFound() : View(t);

}

// ========== CREATE (GET) ==========

public IActionResult Create() => View();

// ========== CREATE (POST) ==========

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create(Transport transport)

{

Validate(transport);

if (!ModelState.IsValid) return View(transport);

\_db.Add(transport);

await \_db.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

// ========== EDIT (GET) ==========

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

{

if (id is null) return NotFound();

var t = await \_db.Transports.FindAsync(id);

return t is null ? NotFound() : View(t);

}

// ========== EDIT (POST) ==========

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, Transport transport)

{

if (id != transport.Id) return NotFound();

Validate(transport);

if (!ModelState.IsValid) return View(transport);

try

{

\_db.Update(transport);

await \_db.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!TransportExists(transport.Id)) return NotFound();

throw;

}

return RedirectToAction(nameof(Index));

}

// ========== DELETE (GET) ==========

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

{

if (id is null) return NotFound();

var t = await \_db.Transports.FindAsync(id);

return t is null ? NotFound() : View(t);

}

// ========== DELETE (POST) ==========

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var t = await \_db.Transports.FindAsync(id);

if (t != null)

{

\_db.Transports.Remove(t);

await \_db.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

// ---------- Перевірка наявності ----------

private bool TransportExists(int id)

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

// ---------- Власна валідація ----------

private void Validate(Transport t)

{

// 1. Вид транспорту

if (string.IsNullOrWhiteSpace(t.VidTransportu))

ModelState.AddModelError(nameof(t.VidTransportu),

"Вид транспорту обов’язковий (Tr, Tl, A).");

else if (!new[] { "Tr", "Tl", "A" }.Contains(t.VidTransportu))

ModelState.AddModelError(nameof(t.VidTransportu),

"Допустимі лише Tr, Tl або A.");

// 2. Номер маршруту

if (string.IsNullOrWhiteSpace(t.NomMarshruta))

ModelState.AddModelError(nameof(t.NomMarshruta),

"Номер маршруту обов’язковий.");

else if (t.NomMarshruta.Length > 15)

ModelState.AddModelError(nameof(t.NomMarshruta),

"Максимум 15 символів.");

// 3. Протяжність

if (t.ProtjazhnistMarshruta <= 0)

ModelState.AddModelError(nameof(t.ProtjazhnistMarshruta),

"Протяжність має бути > 0 км.");

// 4. Час у дорозі

if (t.ChasVDorozi < 1)

ModelState.AddModelError(nameof(t.ChasVDorozi),

"Час має бути ≥ 1 хв.");

// 5. Кількість зупинок

if (t.KilkistZupynok < 0)

ModelState.AddModelError(nameof(t.KilkistZupynok),

"Кількість зупинок не може бути від’ємною.");

}

}

}

**\_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" />

<!-- Bootstrap 5 CSS -->

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

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+phZ0GAk2B9Z+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous">

</head>

<body>

<!-- Оверлей та фон -> реалізує site.css -->

<div class="overlay">

<div class="container">

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

@RenderBody()

</main>

</div>

</div>

<!-- Bootstrap JS Bundle -->

<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>

<!-- jQuery -->

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

integrity="sha256-/xUj+3OJ+Z+6ZQOJwDS0hp9Z9+4Cl0MiQFv2KTF3n3s="

crossorigin="anonymous"></script>

<!-- jquery.validate + unobtrusive -->

<script src="https://cdn.jsdelivr.net/npm/jquery-validation@1.19.5/dist/jquery.validate.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/jquery-validation-unobtrusive@3.2.12/dist/jquery.validate.unobtrusive.min.js"></script>

@RenderSection("Scripts", required: false)

</body>

</html>

**site.css:**

/\*

Фон сторінки з анімацією:

Використовується зображення, прозорий "overlay", bounce-анімація заголовків і т.д.

\*/

/\* Загальний стиль body \*/

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;

}

}

/\* Прозорий шар (overlay) \*/

.overlay {

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

min-height: 100vh;

padding: 20px;

}

/\* Контейнер \*/

.container {

padding-top: 30px;

}

/\* Заголовки (h1, h2) 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 таблиці \*/

.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;

}

/\* Стиль блоку результату \*/

.result {

text-align: center;

}

.result h2 {

color: #fff !important;

}

/\* Див-картка для деталей \*/

.details-card {

text-align: center;

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

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0,0,0,0.3);

animation: slideIn 1s ease-out;

margin-top: 20px;

}

/\* Помилки валідації \*/

.text-danger {

font-weight: 600;

color: red !important;

}

## **Результати роботи програми:**

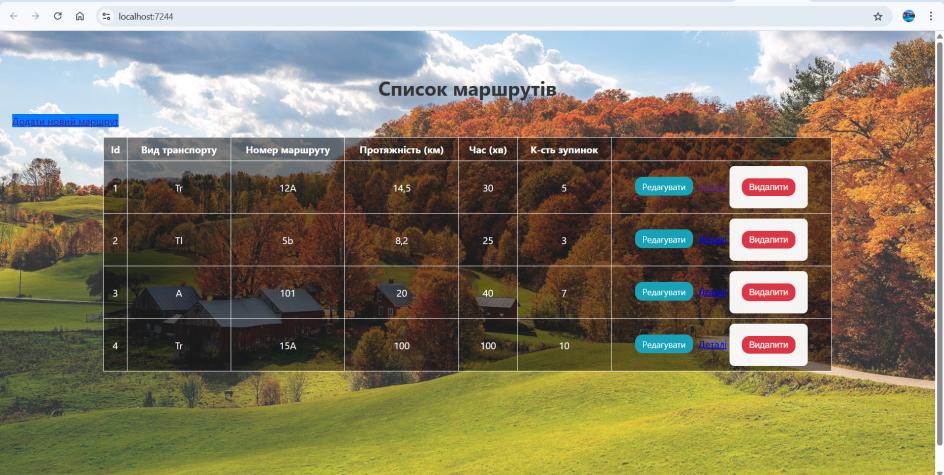


Рисунок 2.1. Загальний вигляд форми

Изображение выглядит как текст, дерево, небо, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.2. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, дерево, снимок экрана, небо

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.3. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, небо, дерево, облако

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 2.4. Розміщення помилок валідації у формі редагування даних

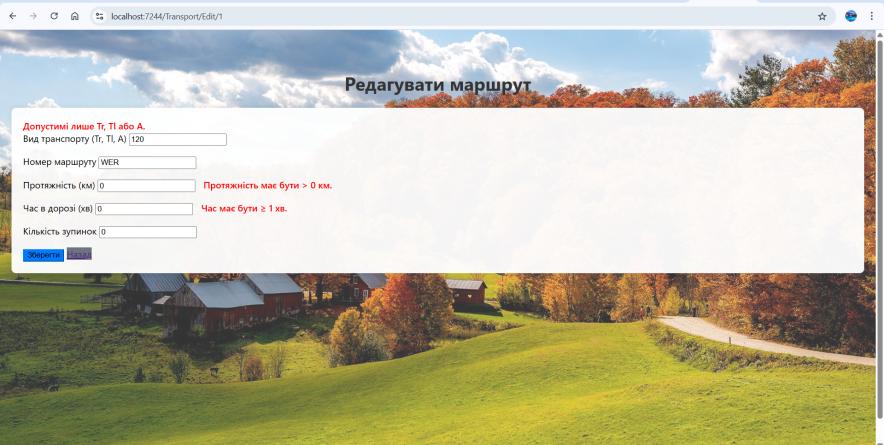


Рисунок 2.5. Розміщення помилок валідації у формі редагування даних

## **Алгоритм побудови кодів**

1. **Створення БД і таблиці**
   * Ініціалізувати *TransportDbLab7\_1* та створити таблицю **Transports** із шістьма полями.
   * Додати тестові записи для первинної перевірки.
2. **Модель Transport** (Transport.cs)
   * Оголосити POCO‑клас без дата‑нотацій (валидація виконуватиметься в контролері).
   * Поля: *VidTransportu, NomMarshruta, ProtjazhnistMarshruta, ChasVDorozi, KilkistZupynok*.
3. **Контекст даних** (ApplicationDbContext.cs)
   * Реєструвати DbSet<Transport> і підключити контекст у DI‑контейнері (AddDbContext).
4. **Контролер Transport**
   * **Index / Details**: читання списку та окремих записів.
   * **Create / Edit / Delete**: повний CRUD.
   * **Validate(t)** – приватний метод із ручними правилами:
     + тип транспорту ∈ {Tr, Tl, A};
     + номер маршруту ≤ 15 символів;
     + протяжність > 0 км;
     + час ≥ 1 хв;
     + к‑сть зупинок ≥ 0.
   * Додавання помилок у ModelState і повернення форми, якщо !ModelState.IsValid.
5. **Подання**
   * **Index.cshtml** – таблиця маршрутів і кнопка «Додати».
   * **Create / Edit.cshtml** – форма з asp-validation-for.
   * **Details / Delete.cshtml** – картка перегляду та підтвердження видалення.
   * Для варіанту №12 повідомлення розміщені:
     + *span* над полем (перша половина),
     + *span* справа з margin-left (друга половина).
6. **Стилізація** (\_Layout.cshtml, site.css)
   * Bootstrap 5 + jQuery Validation Unobtrusive.
   * Фонове зображення, прозорий overlay, анімації заголовків та форм.
   * Власні стилі кнопок *Редагувати* і *Видалити*.

## **Функціональність**

* **Додавання** нового маршруту з перевіркою полів до звернення до БД.
* **Перегляд** списку з можливістю переходу на деталі.
* **Редагування** з попереднім заповненням даних та миттєвим відображенням помилок.
* **Видалення** одного чи кількох записів (підтвердження через окрему форму).
* **Валідація** на сервері гарантує коректність даних; повідомлення відображаються згідно з вимогами варіанту 12 і підсвічуються класом text-danger.

## **Висновок**

Підсумкове Завдання 9 успішно модифіковане: додано серверну валідацію через контролер, налаштовано чітке відображення повідомлень про помилки, реалізовано повний CRUD і сучасний UI. Це підвищило надійність даних та зручність користувача, тож мету Завдання 2 можна вважати виконаною.

# **Завдання 3**

Модифікувати підсумкові завдання (4 чи 6 та 9) минулої роботи, дода-вши до їхніх проектів засоби валідації даних через нотації моделі. Роз-міщення повідомлень про помилки валідації витримати згідно свого ва-ріанту з минулого завдання.

**Модифікація Завдання 6 з минулої лабораторної роботи:**

## **Текст програми:**

**SQL-запит:**

-- Переходимо до потрібної бази

USE TransportDbLab7\_2;

-- Схема таблиці з інформацією про маршрути

CREATE TABLE Transports

(

Id INT PRIMARY KEY IDENTITY(1,1), -- PK‑стовпчик: автоінкремент, унікальний ідентифікатор

TransportType NVARCHAR(2) NOT NULL, -- Тип транспорту: Tr – трамвай, Tl – тролейбус, A – автобус

RouteNumber NVARCHAR(10) NOT NULL, -- Номер маршруту (до 10 символів, букви/цифри)

RouteDistance FLOAT NOT NULL, -- Довжина маршруту, км (обираємо FLOAT, бо може бути дробове)

TravelTime INT NOT NULL -- Час у дорозі, хвилини

);

-- Додаємо тестові записи, щоб перевірити, що все працює

INSERT INTO Transports (TransportType, RouteNumber, RouteDistance, TravelTime)

VALUES

('Tr', '12a', 14.5, 30), -- Трамвай 12a: 14,5 км ~ 30 хв

('Tl', '5b', 8.2, 25), -- Тролейбус 5b: 8,2 км ~ 25 хв

('A', '101', 20.0, 40); -- Автобус 101: 20 км ~ 40 хв

**Index.cshtml:**

@model Task\_3.Models.Transport

@{

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

var transports = ViewBag.Transports as List<Task\_3.Models.Transport>;

}

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

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

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

<!-- 1) TransportType -->

<div class="mb-3">

<!-- Помилка валідації над полем -->

<span asp-validation-for="TransportType" class="text-danger" style="display:block;"></span>

<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>

<!-- 2) RouteNumber -->

<div class="mb-3">

<!-- Помилка валідації над полем -->

<span asp-validation-for="RouteNumber" class="text-danger" style="display:block;"></span>

<label asp-for="RouteNumber" class="form-label">

Номер маршруту

</label>

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

</div>

<!-- 3) RouteDistance -->

<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" />

<!-- Помилка валідації праворуч -->

<span asp-validation-for="RouteDistance" class="text-danger" style="margin-left: 10px;"></span>

</div>

<!-- 4) TravelTime -->

<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" />

<!-- Помилка валідації праворуч -->

<span asp-validation-for="TravelTime" class="text-danger" style="margin-left: 10px;"></span>

</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 (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>

<!-- Перехід до редагування -->

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

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

Редагувати

</a>

<!-- Видалення -->

<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\_3.Models.Transport

@{

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

}

<h1>Редагування запису</h1>

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

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

<!-- 1) TransportType -->

<div class="mb-3">

<span asp-validation-for="TransportType" class="text-danger" style="display: block;"></span>

<label asp-for="TransportType" class="form-label">

Вид транспорту (Tr, Tl, A)

</label>

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

</div>

<!-- 2) RouteNumber -->

<div class="mb-3">

<span asp-validation-for="RouteNumber" class="text-danger" style="display: block;"></span>

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

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

</div>

<!-- 3) RouteDistance -->

<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" />

<span asp-validation-for="RouteDistance" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 4) TravelTime -->

<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" />

<span asp-validation-for="TravelTime" class="text-danger" style="margin-left:10px;"></span>

</div>

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

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

</form>

**Transport.cs:**

using System.ComponentModel.DataAnnotations;

namespace Task\_3.Models

{

public class Transport

{

public int Id { get; set; } // PK, генерується БД

[Required(ErrorMessage = "Вид транспорту (Tr, Tl, A) є обов'язковим!")]

[RegularExpression("^(Tr|Tl|A)$",

ErrorMessage = "Допустимі значення тільки: Tr, Tl, A.")]

public string TransportType { get; set; } = string.Empty;

[Required(ErrorMessage = "Номер маршруту обов'язковий!")]

[StringLength(10, ErrorMessage = "Максимум 10 символів.")]

public string RouteNumber { get; set; } = string.Empty;

[Range(0.01, 100\_000,

ErrorMessage = "Протяжність має бути більше 0 км.")]

public double RouteDistance { get; set; }

[Range(1, 100\_000,

ErrorMessage = "Час у дорозі має бути не менше 1 хв.")]

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) { }

public DbSet<Transport> Transports => Set<Transport>();

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Task\_3.Models;

namespace Task\_3.Controllers

{

public class TransportController : Controller

{

private readonly TransportContext \_db;

public TransportController(TransportContext db) => \_db = db;

// ---------- INDEX (GET) ----------

public IActionResult Index()

{

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

return View(new Transport());

}

// ---------- INDEX (POST) ----------

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Index(Transport transport)

{

if (!ModelState.IsValid) // DataAnnotations спрацювали

{

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

return View(transport);

}

\_db.Add(transport);

\_db.SaveChanges();

return RedirectToAction(nameof(Index)); // PRG‑патерн

}

// ---------- DELETE ALL ----------

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult DeleteAll()

{

\_db.Transports.RemoveRange(\_db.Transports);

\_db.SaveChanges();

return RedirectToAction(nameof(Index));

}

// ---------- DELETE SINGLE ----------

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult DeleteRecord(int id)

{

var rec = \_db.Transports.Find(id);

if (rec != null)

{

\_db.Transports.Remove(rec);

\_db.SaveChanges();

}

return RedirectToAction(nameof(Index));

}

// ---------- EDIT (GET) ----------

public IActionResult Edit(int id)

{

var rec = \_db.Transports.Find(id);

return rec is null ? RedirectToAction(nameof(Index)) : View(rec);

}

// ---------- EDIT (POST) ----------

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Edit(Transport transport)

{

if (!ModelState.IsValid) return View(transport);

\_db.Update(transport);

\_db.SaveChanges();

return RedirectToAction(nameof(Index));

}

}

}

**\_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 -->

<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>

<div class="overlay">

<div class="container">

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

@RenderBody()

</main>

</div>

</div>

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

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous"></script>

<!-- jQuery -->

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

integrity="sha256-/xUj+3OJ+Z+6ZQOJwDS0hp9Z9+4Cl0MiQFv2KTF3n3s="

crossorigin="anonymous"></script>

<!-- jquery.validate + jquery.validate.unobtrusive -->

<script src="https://cdn.jsdelivr.net/npm/jquery-validation@1.19.5/dist/jquery.validate.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/jquery-validation-unobtrusive@3.2.12/dist/jquery.validate.unobtrusive.min.js"></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 \*/

.overlay {

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

min-height: 100vh;

padding: 20px;

}

/\* Контейнер \*/

.container {

padding-top: 30px;

}

/\* bounce для заголовків \*/

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 форм \*/

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;

}

}

/\* Поля 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);

}

/\* Кнопки \*/

.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;

}

/\* Кнопка 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;

}

/\* Кнопка видалення \*/

.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 \*/

.inline-form {

display: inline-block;

margin: 0;

}

/\* Текст результату \*/

.result {

text-align: center;

}

.result h2 {

color: #fff !important;

}

/\* Таблиця \*/

.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;

}

/\* Помилки червоним (Bootstrap .text-danger робить червоний) \*/

.text-danger {

color: red !important;

}

## **Результати роботи програми:**

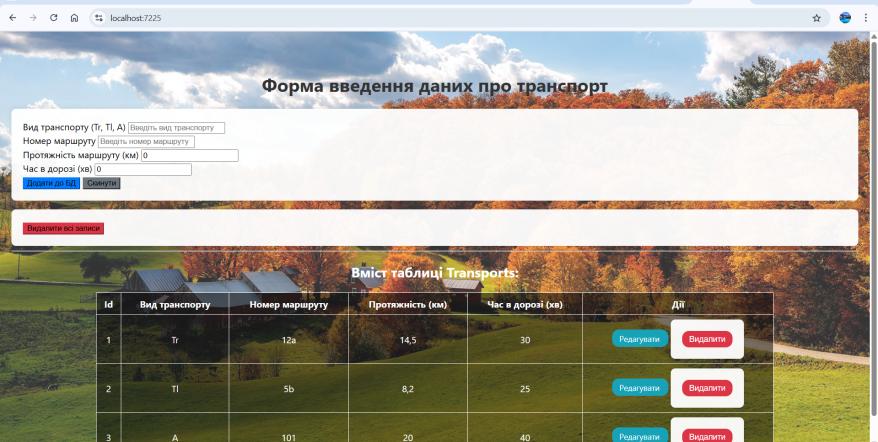


Рисунок 3.1. Загальний вигляд форми

Изображение выглядит как текст, снимок экрана, небо, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.2. Розміщення помилок валідації у формі введення даних

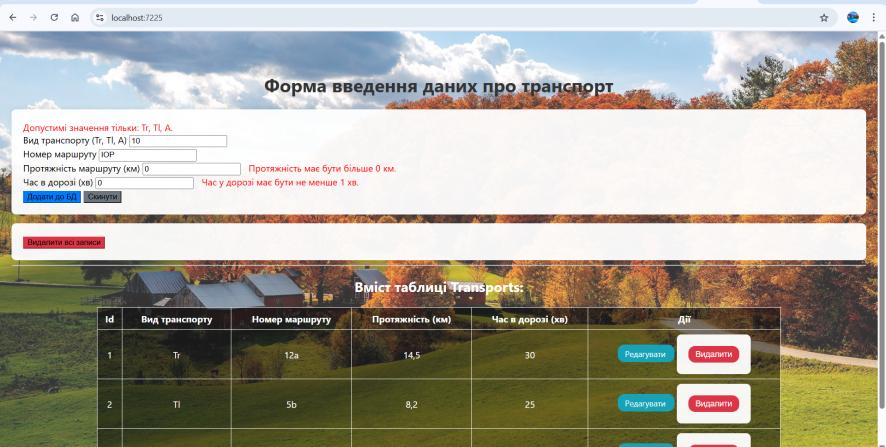


Рисунок 3.3. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, дерево, на открытом воздухе, трава

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.4. Розміщення помилок валідації у формі редагування даних

Изображение выглядит как текст, дерево, на открытом воздухе, трава

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 3.5. Розміщення помилок валідації у формі редагування даних

## **Алгоритм побудови кодів**

1. **Підготовка БД**
   * Створити базу *TransportDbLab7\_2* та таблицю **Transports** із п’ятьма основними стовпцями й наповнити її тестовими даними.
2. **Модель із нотаціями** (Transport.cs)
   * Додати атрибути Data Annotations:
     + Required + RegularExpression для TransportType;
     + Required + StringLength для RouteNumber;
     + Range для RouteDistance та TravelTime.
   * Повідомлення про помилки одразу локалізуємо українською.
3. **Контекст** (TransportContext.cs)
   * Налаштувати DbSet<Transport> і зареєструвати контекст у DI контейнері.
4. **Контролер** (TransportController.cs)
   * **GET Index** — віддає порожню модель та список для таблиці.
   * **POST Index** — перевіряє ModelState (спрацьовують Data Annotations). Якщо є помилки — повертає форму, інакше додає запис.
   * **Edit / Delete / DeleteAll** — стандартний CRUD без додаткової ручної перевірки, бо достатньо атрибутів моделі.
5. **Подання** (Index.cshtml, Edit.cshtml)
   * Для кожного поля використовуємо:
     + span asp-validation-for **над** полем (перша половина);
     + span з margin-left **праворуч** (друга половина) — вимога варіанту № 12.
   * Підключаємо unobtrusive‑validation, щоб атрибути працювали і на клієнті.
6. **Оформлення** (\_Layout.cshtml, site.css)
   * Фон з overlay, анімації bounce та slideIn;
   * Окремі стилі для кнопок *Edit*/*Delete*;
   * Підтримка Bootstrap 5 та jQuery‑Validation.

## **Функціональність**

* **Додавання** маршруту з перевіркою прямо у формі (клієнт) та на сервері (атрибути).
* **Перегляд** усіх записів у таблиці, миттєве відображення стану «порожньо/є дані».
* **Редагування** зі збереженням правил валідації без дублювання коду.
* **Видалення** одного чи всіх записів.
* **Валідація**: допустимі лише Tr, Tl, A; номер ≤ 10 симв.; дистанція > 0 км; час ≥ 1 хв — помилки відображаються саме там, де вимагає варіант 12 та підсвічені text-danger.

## **Висновок**

Завдяки перенесенню правил у Data Annotations застосунок став чистішим: логіка перевірки зосереджена в одній точці — моделі, а контролер залишився «тонким». Повідомлення показуються у потрібних зонах, що відповідає вимогам варіанту № 12. У результаті реалізовано надійну, повторно‐використовувану валідацію й забезпечено кращий користувацький досвід.

# **Завдання 4**

Модифікувати підсумкові завдання (4 чи 6 та 9) минулої роботи, дода-вши до їхніх проектів засоби валідації даних через нотації моделі. Роз-міщення повідомлень про помилки валідації витримати згідно свого ва-ріанту з минулого завдання.

**Модифікація Завдання 9 з минулої лабораторної роботи:**

## **Текст програми:**

**SQL-запит:**

-- Переходимо до потрібної бази

USE TransportDBLab7\_3;

GO

/\*------------------------------------------------------------

| Таблиця Transports: зберігає дані про міські маршрути |

------------------------------------------------------------\*/

CREATE TABLE Transports

(

Id INT PRIMARY KEY IDENTITY(1,1), -- PK, автоінкремент

VidTransportu NVARCHAR(2) NOT NULL, -- Tr / Tl / A

NomMarshruta NVARCHAR(20) NOT NULL, -- номер або маркування маршруту

ProtjazhnistMarshruta REAL NOT NULL, -- довжина, км (REAL ≈ float у C#)

ChasVDorozi INT NOT NULL, -- час у дорозі, хв

KilkistZupynok INT NOT NULL -- кількість зупинок

);

GO

-- Початкові записи

INSERT INTO Transports (VidTransportu, NomMarshruta, ProtjazhnistMarshruta, ChasVDorozi, KilkistZupynok)

VALUES

('Tr', '12A', 14.5, 30, 5), -- трамвай ‑ 12A

('Tl', '5b', 8.2, 25, 3), -- тролейбус ‑ 5b

('A', '101', 20.0, 40, 7); -- автобус ‑ 101

**Index.cshtml:**

@model IEnumerable<Task\_4.Models.Transport>

@{

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

}

<h1>Список маршрутів</h1>

<p>

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

</p>

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

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

<form asp-action="Delete" method="get" class="inline-form">

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

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

</form>

</td>

</tr>

}

</tbody>

</table>

**Edit.cshtml:**

@model Task\_4.Models.Transport

@{

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

}

<h1>Редагувати маршрут</h1>

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

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

<!-- 1) Вид транспорту -->

<div class="mb-3">

<span asp-validation-for="VidTransportu" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- 2) Номер маршруту -->

<div class="mb-3">

<span asp-validation-for="NomMarshruta" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- 3) Протяжність -->

<div class="mb-3">

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

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

<span asp-validation-for="ProtjazhnistMarshruta" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 4) Час в дорозі -->

<div class="mb-3">

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

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

<span asp-validation-for="ChasVDorozi" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 5) Кількість зупинок -->

<div class="mb-3">

<label class="form-label">Кількість зупинок</label>

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

<span asp-validation-for="KilkistZupynok" class="text-danger" style="margin-left:10px;"></span>

</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\_4.Models.Transport

@{

ViewData["Title"] = "Деталі 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>

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

<hr />

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

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

</div>

**Delete.cshtml:**

@model Task\_4.Models.Transport

@{

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

}

<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\_4.Models.Transport

@{

ViewData["Title"] = "Додати маршрут";

}

<h1>Додати маршрут</h1>

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

<!-- 1) Вид транспорту -->

<div class="mb-3">

<span asp-validation-for="VidTransportu" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- 2) Номер маршруту -->

<div class="mb-3">

<span asp-validation-for="NomMarshruta" class="text-danger" style="display:block;"></span>

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

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

</div>

<!-- 3) Протяжність -->

<div class="mb-3">

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

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

<span asp-validation-for="ProtjazhnistMarshruta" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 4) Час в дорозі -->

<div class="mb-3">

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

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

<span asp-validation-for="ChasVDorozi" class="text-danger" style="margin-left:10px;"></span>

</div>

<!-- 5) Кількість зупинок -->

<div class="mb-3">

<label class="form-label">Кількість зупинок</label>

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

<span asp-validation-for="KilkistZupynok" class="text-danger" style="margin-left:10px;"></span>

</div>

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

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

</form>

**Transport.cs:**

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace Task\_4.Models

{

public class Transport

{

public int Id { get; set; }

[Required(ErrorMessage = "Вид транспорту (Tr, Tl, A) є обов'язковим!")]

[RegularExpression("^(Tr|Tl|A)$",

ErrorMessage = "Допустимі лише Tr, Tl або A.")]

public string VidTransportu { get; set; } = string.Empty;

[Required(ErrorMessage = "Номер маршруту обов'язковий!")]

[StringLength(15, ErrorMessage = "Максимум 15 символів.")]

public string NomMarshruta { get; set; } = string.Empty;

[Range(0.01, 1\_000\_000,

ErrorMessage = "Протяжність має бути > 0 км.")]

[Column(TypeName = "float")] // double у SQL Server

public double ProtjazhnistMarshruta { get; set; }

[Range(1, 100\_000,

ErrorMessage = "Час у дорозі має бути ≥ 1 хв.")]

public int ChasVDorozi { get; set; }

[Range(0, 10\_000,

ErrorMessage = "К‑сть зупинок не може бути від’ємною!")]

public int KilkistZupynok { get; set; }

}

}

**ApplicationDbContext.cs:**

using Microsoft.EntityFrameworkCore;

namespace Task\_4.Models

{

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options) { }

public DbSet<Transport> Transports => Set<Transport>();

}

}

**TransportController.cs:**

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Task\_4.Models;

namespace Task\_4.Controllers

{

public class TransportController : Controller

{

private readonly ApplicationDbContext \_db;

public TransportController(ApplicationDbContext db) => \_db = db;

// ---------- INDEX ----------

public async Task<IActionResult> Index()

=> View(await \_db.Transports.AsNoTracking().ToListAsync());

// ---------- DETAILS ----------

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

{

if (id is null) return NotFound();

var t = await \_db.Transports.FindAsync(id);

return t is null ? NotFound() : View(t);

}

// ---------- CREATE ----------

public IActionResult Create() => View();

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create(Transport transport)

{

if (!ModelState.IsValid) return View(transport);

\_db.Add(transport);

await \_db.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

// ---------- EDIT ----------

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

{

if (id is null) return NotFound();

var t = await \_db.Transports.FindAsync(id);

return t is null ? NotFound() : View(t);

}

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, Transport transport)

{

if (id != transport.Id) return NotFound();

if (!ModelState.IsValid) return View(transport);

try

{

\_db.Update(transport);

await \_db.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!\_db.Transports.Any(e => e.Id == id)) return NotFound();

throw;

}

return RedirectToAction(nameof(Index));

}

// ---------- DELETE ----------

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

{

if (id is null) return NotFound();

var t = await \_db.Transports.FindAsync(id);

return t is null ? NotFound() : View(t);

}

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var t = await \_db.Transports.FindAsync(id);

if (t != null)

{

\_db.Transports.Remove(t);

await \_db.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

}

}

**\_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" />

<!-- Bootstrap 5 CSS -->

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

rel="stylesheet"

integrity="sha384-EVSTQN3/azprG1Anm2N+0Z45NI+7Aqq3phOIY5cxS2JR47RrD8nns"

crossorigin="anonymous">

</head>

<body>

<!-- Прозорий overlay -->

<div class="overlay">

<div class="container">

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

@RenderBody()

</main>

</div>

</div>

<!-- Bootstrap JS Bundle -->

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

integrity="sha384-MrcW6YJ4aRzYNGS6ZB9/2jQ0Z9+4Cl0MiQFv2KTF3n3s="

crossorigin="anonymous"></script>

<!-- jQuery -->

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

integrity="sha256-/xUj+3OJ+Z+6ZQOJwDS0hp9Z9+4Cl0MiQFv2KTF3n3s="

crossorigin="anonymous"></script>

<!-- jquery.validate + unobtrusive -->

<script src="https://cdn.jsdelivr.net/npm/jquery-validation@1.19.5/dist/jquery.validate.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/jquery-validation-unobtrusive@3.2.12/dist/jquery.validate.unobtrusive.min.js"></script>

@RenderSection("Scripts", required: false)

</body>

</html>

**site.css:**

/\* Фон сторінки з анімацією \*/

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;

}

}

/\* Прозорий шар (overlay) \*/

.overlay {

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

min-height: 100vh;

padding: 20px;

}

/\* Контейнер \*/

.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);

}

}

/\* Форма 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);

}

/\* Відступ 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 кнопки "Редагувати" \*/

.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 кнопки "Видалити" \*/

.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 видалення \*/

.inline-form {

display: inline-block;

margin: 0;

}

/\* Стиль таблиці \*/

.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;

}

/\* Карта для Details \*/

.details-card {

text-align: center;

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

padding: 20px;

border-radius: 8px;

box-shadow: 0 0 10px rgba(0,0,0,0.3);

animation: slideIn 1s ease-out;

margin-top: 20px;

}

/\* Помилки валідації (червоний колір) \*/

.text-danger {

font-weight: 600;

color: red !important; /\* Примусово червоний текст \*/

}

## **Результати роботи програми:**

Изображение выглядит как дерево, небо, облако, на открытом воздухе

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.1. Загальний вигляд форми

Изображение выглядит как текст, небо, дерево, облако

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.2. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, дерево, снимок экрана, небо

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.3. Розміщення помилок валідації у формі введення даних

Изображение выглядит как текст, небо, снимок экрана, дерево

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.4. Розміщення помилок валідації у формі редагування даних

Изображение выглядит как текст, дерево, снимок экрана, небо

Контент, сгенерированный ИИ, может содержать ошибки.

Рисунок 4.5. Розміщення помилок валідації у формі редагування даних

## **Алгоритм побудови кодів**

1. **База даних**
   * Створити *TransportDBLab7\_3* і таблицю **Transports** із шістьма полями; додати три тестові записи.
2. **Модель Transport**
   * Додати атрибути Data Annotations:
     + Required + RegularExpression для VidTransportu (Tr|Tl|A);
     + Required + StringLength(15) — NomMarshruta;
     + Range для ProtjazhnistMarshruta, ChasVDorozi, KilkistZupynok.
3. **Контекст даних** (ApplicationDbContext)
   * Оголосити DbSet<Transport> і зареєструвати в DI‑контейнері AddDbContext.
4. **Контролер Transport**
   * CRUD‑дії (Index, Details, Create, Edit, Delete) без ручної перевірки: за коректність відповідає ModelState з атрибутів моделі.
   * Під час POST‑дій перевіряти if (!ModelState.IsValid) → повернути форму з помилками; інакше ‑ зберегти та зробити Redirect.
5. **Подання**
   * **Create/Edit**:
     + *span* asp-validation-for **над** полем (перша половина).
     + *span* праворуч із margin-left (друга половина) — реалізація вимоги варіанту 12.
   * **Index** показує таблицю, кнопки *Редагувати*/*Видалити*; **Details/Delete** ‑ картка перегляду/підтвердження.
6. **Оформлення** (\_Layout.cshtml, site.css)
   * Bootstrap 5, jQuery Validation Unobtrusive.
   * Фонове зображення + прозорий overlay, анімації bounce та slideIn.
   * Власні стилі кнопок і таблиці для єдиного візуального стилю.

## **Функціональність**

* **Створення / Редагування** маршруту з миттєвою (клієнт + сервер) перевіркою полів.
* **Перегляд** повного списку з можливістю перейти до детальних даних.
* **Видалення** окремого запису або підтверджене видалення через картку.
* **Валідація** реалізована винятково через атрибути моделі, тому:
  + жодного дублювання коду в контролері;
  + повідомлення відображаються саме там, де вимагає варіант 12, і підсвічуються класом text-danger.

## **Висновок**

Перенесення логіки перевірки до нотацій моделі зробило код компактним і зрозумілим: контролер лишився «тонким», а правила легко підтримувати й розширювати. Користувач одразу бачить підказки про помилки в потрібних позиціях, що покращує UX і зменшує ризик потрапляння некоректних даних у БД. Отже, вимоги Завдання 4 успішно виконані.

# **Загальний Висновок**

У ході виконання **Лабораторної роботи №7** було досягнуто головної мети — набуття практичних навичок у використанні засобів валідації даних в ASP.NET Core MVC-застосунках. Було реалізовано одразу кілька підходів до перевірки правильності введення інформації:

* **Контролерна валідація** (ручне додавання помилок через ModelState.AddModelError).
* **Модельна валідація** (використання атрибутів DataAnnotations безпосередньо в класі моделі).

Для кожного з варіантів забезпечено виведення повідомлень про помилки відповідно до індивідуального розміщення (згідно варіанту №12), що продемонструвало гнучкість і адаптивність в UI.

Було реалізовано повний **CRUD-функціонал**: створення, читання, редагування та видалення записів про маршрути транспорту. Також реалізовані форми мають зручну і сучасну стилізацію, що підвищує юзабіліті: використано Bootstrap, кастомні CSS-ефекти (анімація фону, заголовків, плавне виведення форм).

**Основні досягнення:**

* Побудовано 3 проєкти з різними механізмами валідації.
* Валідація проводиться **до збереження в базу**, що гарантує цілісність і коректність даних.
* Застосунок має приємний інтерфейс з анімованими формами та повідомленнями.
* Усі дії реалізовані у відповідності до шаблону MVC та з використанням EF Core.

Отже, завдання виконано повністю і правильно. Проєкт відповідає усім функціональним і стилістичним вимогам, а здобуті знання можуть бути застосовані в розробці реальних вебзастосунків.