Лабораторна робота №2

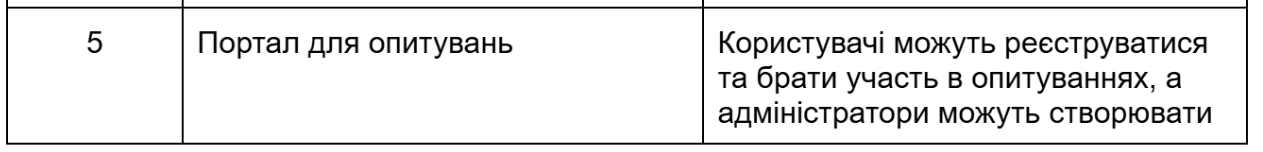
**Тема:** Робота з ASP.NET MVC. Конфігурація проекту. Відображення даних з БД на веб-сторінках.

**Мета:** Ознайомитися з архітектурою MVC, набути навичок створення головних компонентів архітектури MVC, навчитися конфігурувати підключення до БД та виконувати запити на читання даних, набути навичок роботи з представленнями (Views).

Хід роботи:

# Завдання 1. Створення та конфігурація проекту

**Створити проект, який реалізує архітектуру MVC, слідуючи інструкціям наведеним нижче.**

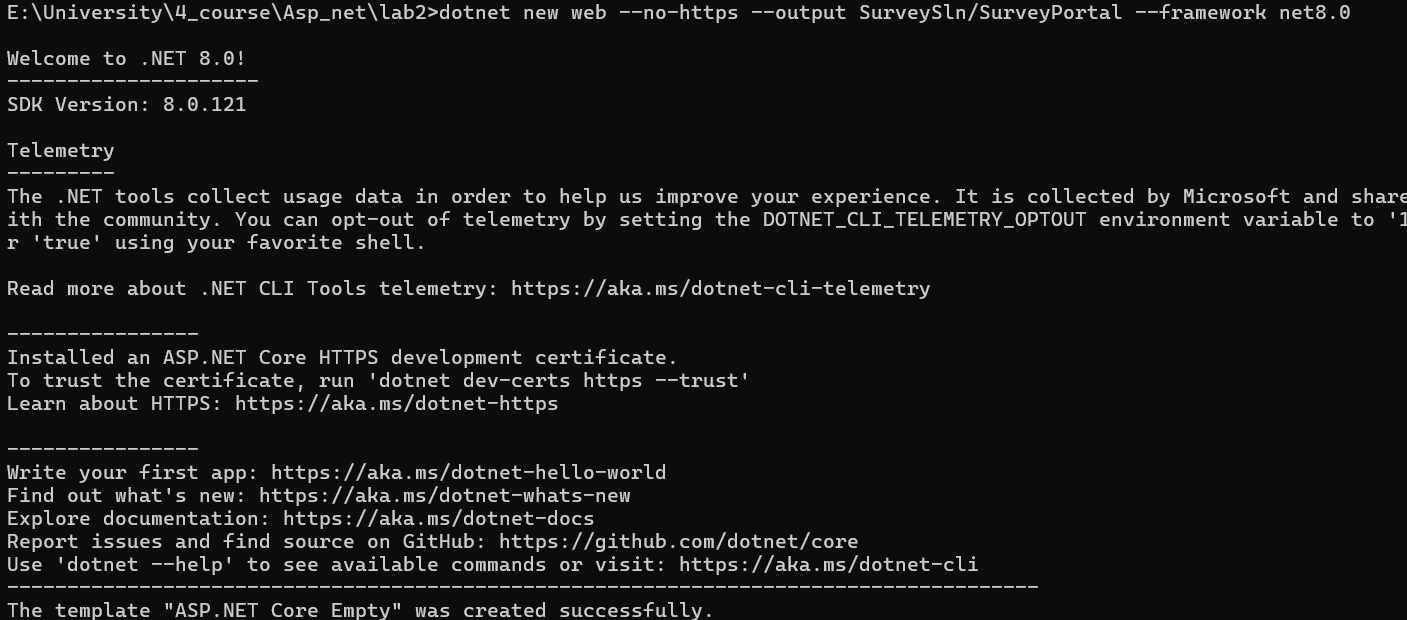


Ця команда створить global.json файл в директорії за шляхом SportsSln/SportsStore. Цей файл потрібен для уникнення проблем при роботі над одним проектом в команді. Він гарантує, що всі розробники, які працюють над проектом, використовують ту саму версію .NET SDK, запобігаючи проблемам із сумісністю.



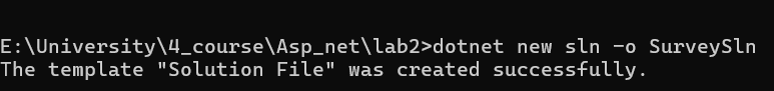
Ця команда створює новий проект типу ASP.NET Core minimal web API, опція --no-https відключає налаштування для HTTPS

Наступна команда:



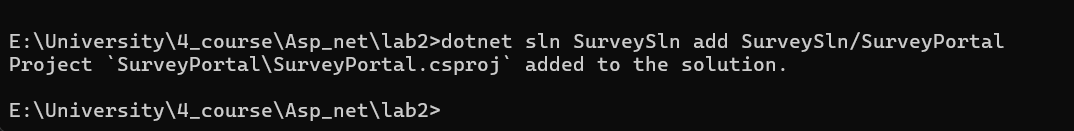
Створення файлу рішення (.sln):

dotnet new sln -o SurveySln

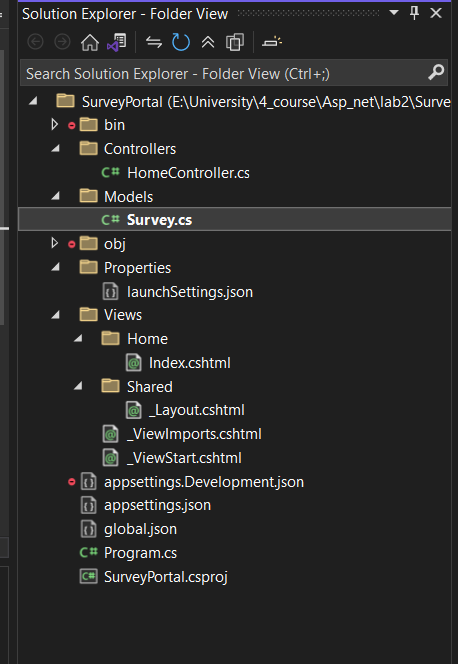


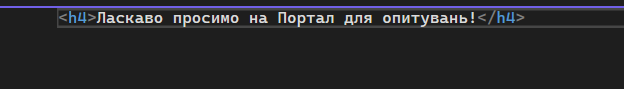
Додавання проекту до рішення:

dotnet sln SurveySln add SurveySln/SurveyPortal

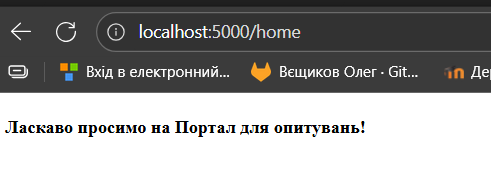


Створена початкова структура програми:

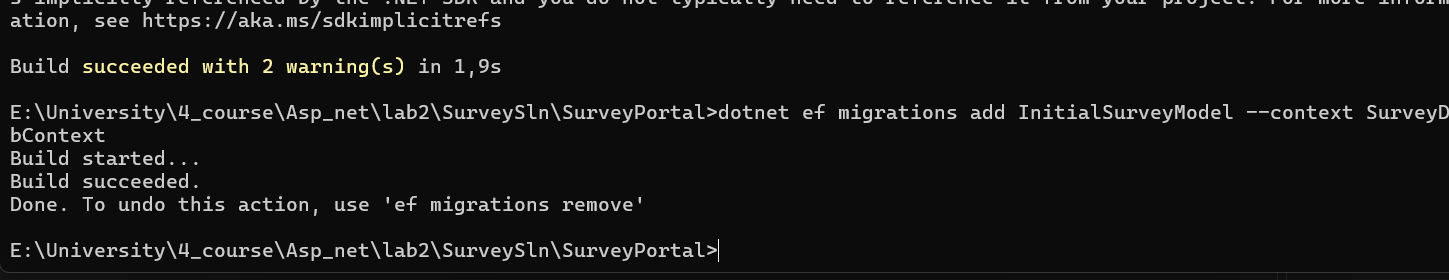


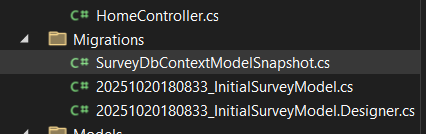


Результат роботи в браузері:



Міграція успішно створена:





**Завдання 3. Виведення даних на сторінці**

**3.1. Зміна контролера**

Оновлення файлу Controllers/HomeController.cs для використання репозиторію та передачі даних у представлення:

using Microsoft.AspNetCore.Mvc;

using SurveyPortal.Models;

namespace SurveyPortal.Controllers

{

public class HomeController : Controller

{

private ISurveyRepository repository;

// Dependency Injection репозиторію через конструктор

public HomeController(ISurveyRepository repo)

{

repository = repo;

}

// Передача всіх опитувань у представлення

public IActionResult Index() => View(repository.Surveys);

}

}

**3.2. Оновлення представлення**

Оновіть файл Views/Home/Index.cshtml для відображення списку опитувань:

@model IQueryable<SurveyPortal.Models.Survey>

<h4>Доступні опитування</h4>

@foreach (var s in Model ?? Enumerable.Empty<SurveyPortal.Models.Survey>())

{

<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 15px;">

<h3>@s.Title</h3>

<p>@s.Description</p>

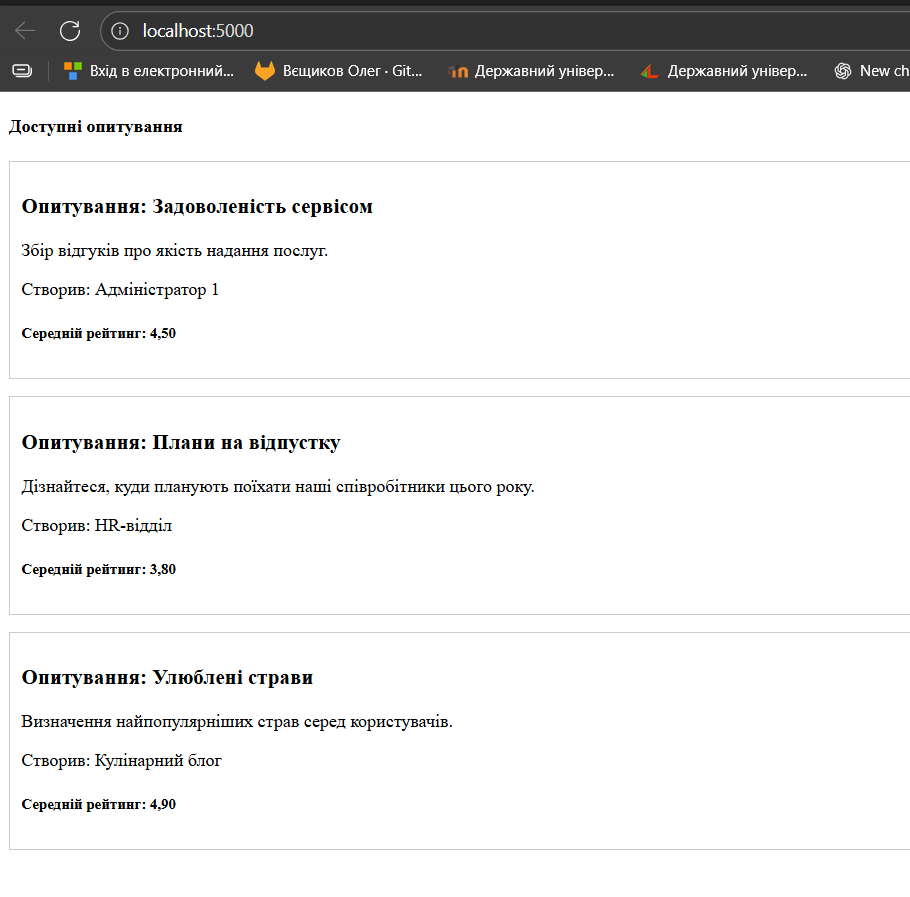
<p>Створив: @s.Creator</p>

<h5>Середній рейтинг: @s.AverageRating.ToString("F2")</h5>

</div>

}

Результат:



**Завдання 4. Реалізація посторінкового виводу записі:**

**4.1. Створення ViewModels**

Створення файлу Models/ViewModels/PagingInfo.cs

namespace SurveyPortal.Models.ViewModels

{

public class PagingInfo

{

public int TotalItems { get; set; }

public int ItemsPerPage { get; set; } = 4; // Припустимо, 4 опитування на сторінку

public int CurrentPage { get; set; }

public int TotalPages => (int)Math.Ceiling((decimal)TotalItems / ItemsPerPage);

}

}

Створення файлу Models/ViewModels/SurveysListViewModel.cs:

using System.Collections.Generic;

namespace SurveyPortal.Models.ViewModels

{

public class SurveysListViewModel

{

public IEnumerable<Survey> Surveys { get; set; } = Enumerable.Empty<Survey>();

public PagingInfo PagingInfo { get; set; } = new PagingInfo();

}

}

**4.2. Додавання підтримки пейджингу в контролер**

Оновлення Controllers/HomeController.cs для реалізації логіки пейджингу:

using Microsoft.AspNetCore.Mvc;

using SurveyPortal.Models;

using SurveyPortal.Models.ViewModels; // Додайте using

namespace SurveyPortal.Controllers

{

public class HomeController : Controller

{

private ISurveyRepository repository;

public HomeController(ISurveyRepository repo)

{

repository = repo;

}

public IActionResult Index(int surveyPage = 1)

{

var itemsPerPage = 4; // Кількість елементів на сторінку

return View(new SurveysListViewModel

{

Surveys = repository.Surveys

.OrderBy(s => s.SurveyID) // Сортування для стабільного пейджингу

.Skip((surveyPage - 1) \* itemsPerPage)

.Take(itemsPerPage),

PagingInfo = new PagingInfo

{

CurrentPage = surveyPage,

ItemsPerPage = itemsPerPage,

TotalItems = repository.Surveys.Count()

}

});

}

}

}

**4.3. Адаптація представлення**

Оновлення Views/Home/Index.cshtml для використання SurveysListViewModel:

@using SurveyPortal.Models.ViewModels

@model SurveysListViewModel

<h4>Доступні опитування</h4>

@foreach (var s in Model?.Surveys ?? Enumerable.Empty<SurveyPortal.Models.Survey>())

{

<div style="border: 1px solid #ccc; padding: 10px; margin-bottom: 15px;">

<h3>@s.Title</h3>

<p>@s.Description</p>

<p>Створив: @s.Creator</p>

<h5>Середній рейтинг: @s.AverageRating.ToString("F2")</h5>

</div>

}

<div id="paging-controls" style="margin-top: 20px;">

<!-- Елемент для вибору сторінки буде тут -->

</div>

**4.4. Додавання елемента для вибору сторінки (Пейджинг)**

логіка для елементів пейджингу в Views/Home/Index.

<div id="paging-controls" style="margin-top: 20px; display: flex; gap: 5px;">

@for (int i = 1; i <= Model.PagingInfo.TotalPages; i++)

{

<a href="@Url.Action("Index", new { surveyPage = i })"

style="padding: 8px 12px; border: 1px solid #007bff; text-decoration: none;

background-color: @(i == Model.PagingInfo.CurrentPage ? "#007bff" : "white");

color: @(i == Model.PagingInfo.CurrentPage ? "white" : "#007bff");

border-radius: 4px;">

@i

</a>

}

</div>

**Завдання 5. Стилізація**

5.1. Встановлення Bootstrap

Виконайте команди (в кореневій папці проекту, наприклад, SurveySln/SurveyPortal):

dotnet tool install --global Microsoft.Web.LibraryManager.Cli

libman init -p cdnjs

libman install bootstrap -d wwwroot/lib/bootstrap

Це створить папку wwwroot/lib/bootstrap з файлами Bootstrap.

**5.2. Зміна базового шаблону (\_Layout.cshtml)**

Оновлення файлу Views/Shared/\_Layout.cshtml, додавши посилання на Bootstrap CSS/JS, хедер та футер:

<!DOCTYPE html>

<html>

<head>

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

<title>Портал для опитувань</title>

<!-- Підключення Bootstrap CSS -->

<link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />

</head>

<body>

<header class="bg-primary text-white p-3 mb-4 shadow-sm">

<div class="container-fluid">

<span class="h3">Портал для опитувань</span>

</div>

</header>

<div class="container-fluid">

@RenderBody()

</div>

<!-- Підключення Bootstrap JS (якщо потрібний інтерактив) -->

<script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>

</body>

</html>

**5.3. Стилізація списку сутностей та пейджингу**

Оновлення файлу Views/Home/Index.cshtml, використовуючи класи Bootstrap:

@using SurveyPortal.Models.ViewModels

@model SurveysListViewModel

<h4 class="mb-4">Доступні опитування</h4>

@foreach (var s in Model?.Surveys ?? Enumerable.Empty<SurveyPortal.Models.Survey>())

{

<div class="card mb-3 shadow-sm">

<div class="card-body">

<h5 class="card-title">@s.Title</h5>

<p class="card-text">@s.Description</p>

<div class="d-flex justify-content-between align-items-center">

<small class="text-muted">Створив: @s.Creator</small>

<span class="badge bg-success p-2">Рейтинг: @s.AverageRating.ToString("F2")</span>

</div>

</div>

</div>

}

<!-- Стилізований елемент для вибору сторінки -->

<nav class="mt-4">

<ul class="pagination">

@for (int i = 1; i <= Model.PagingInfo.TotalPages; i++)

{

<li class="page-item @(i == Model.PagingInfo.CurrentPage ? "active" : "")">

<a class="page-link"

asp-action="Index"

asp-route-surveyPage="@i">

@i

</a>

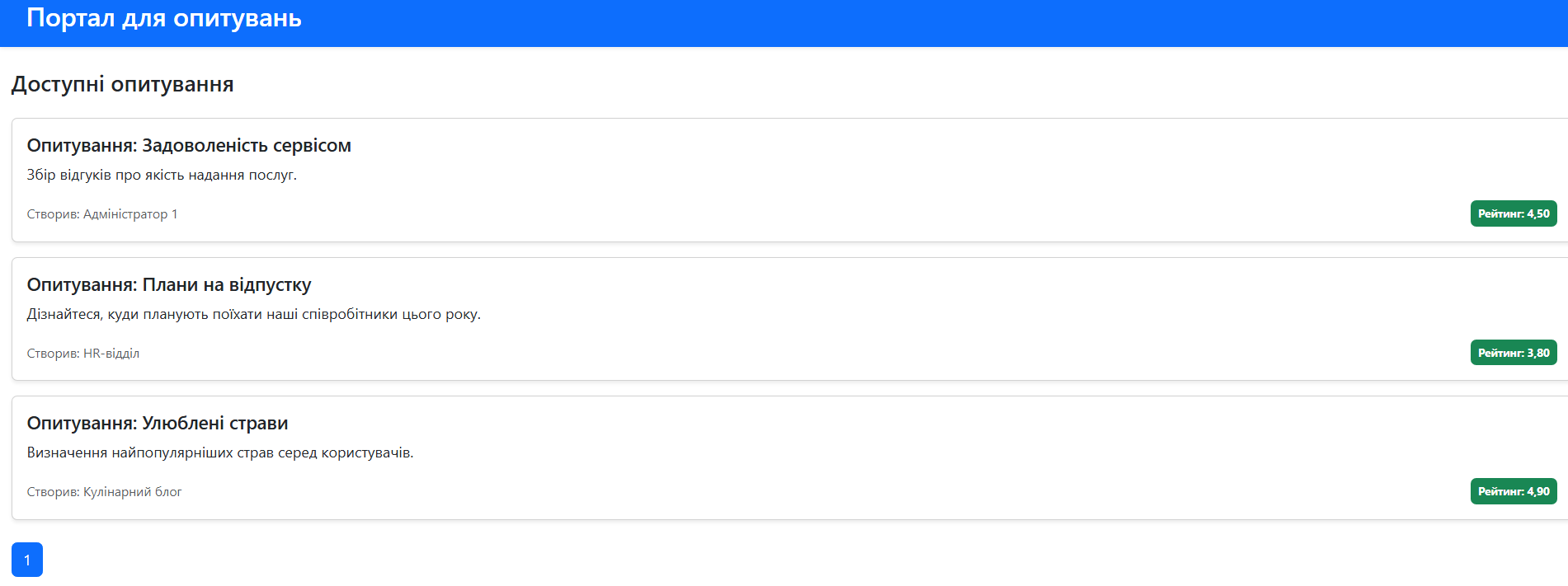
</li>

}

</ul>

</nav>

Результат:



Висновок:

**Що було зроблено:**

1. **Налаштування проекту:** Я створив ASP.NET Core MVC-проект і вирішив усі проблеми з конфігурацією, що виникали через використання мінімального шаблону (включно з виправленням .csproj для коректної роботи Razor та Entity Framework Core).
2. **Робота з БД:** Я інтегрував Entity Framework Core, створив контекст бази даних (SurveyDbContext) та модель даних (Survey), а також реалізував початкове наповнення даними (SeedData).
3. **MVC та Логіка:** Я реалізував патерн Репозиторій та використав Dependency Injection для доступу до даних.
4. **Фронтенд:** Я вивів список опитувань на сторінку, реалізував складний функціонал **посторінкового виводу (Paging)** за допомогою View Models, а також стилізував інтерфейс, підключивши Bootstrap.