Міністерство освіти України

Національний технічний університет "ХПІ"

кафедра "Інформатики та інтелектуальної власності"

**Звіт**

**Лабораторна робота 1**

з дисципліни "Проєктування серверних застосунків"

Виконав: студент групи КН-922В

Шершньов М.О.

Перевірив:

Івашко А.В.

Харків 2025

Зміст

[Завдання 3](#_Toc192421419)

[Контролери textstyle, image, columncount, pagecontroller, visitcount 6](#_Toc192421420)

[Контролер TextStyle 6](#_Toc192421421)

[Контролер Image 7](#_Toc192421422)

[Контролер ColumnCount 7](#_Toc192421423)

[Контролер PageController 8](#_Toc192421424)

[Контролер VisitCount 9](#_Toc192421425)

[Висновок 10](#_Toc192421426)

**Створення простих MVC WEB-застосунків засобами ASP.NET**

**Мета роботи:** придбати практику у створенні простих MVC WEB-додатків, заснованих на зв'язках Model - View Controller з невеликою кількістю конт-ролерів та елементів подання і без використання моделей.

Завдання

1. Створити МѴC WEB-застосунок, головне подання (Index) якого містить нетипізовану (непов'язаною із моделлю) форму (текстові поля, області, списки, радібатони, чекбокси тощо) для задання групи параметрів, які че-рез контролер, що викликається з цієї ж форми, передаються до іншого подання, на якому створюються три блоки (абзаци) тексту, котрі відрізня ються один від одного за розміром, кольором, вмістом, типом вирівню-вання та формою тексту; кольором, зображенням (із його параметрами) фону; розмірами та відступами між блоками. Під час виконання завдання треба врахувати те, що за номером власного варіанту частина інформації (вміст та css-параметри абзаців) має розташовуватися в діалогових об'єктах форми головного подання застосунку, а інша частина css-параметрів має бути зчитана контролером із відповідної таблиці чи таблиць бази даних (БД).

Мій варіант:





1. Створити MVC WEB-застосунок, головне подання якого містить нети-пізовану форму для задання групи параметрів, які через контролер, що викликається з цієї ж форми, передаються до іншого подання, на якому формуються зображення згідно із топологією власного варіанту.

Мій варіант:



Изображение выглядит как снимок экрана, Прямоугольник, линия, текст

Контент, сгенерированный ИИ, может содержать ошибки.

Модифікувати застосунок для реалізації ще однієї топології зображень.

Мій варіант:



Изображение выглядит как Прямоугольник, снимок экрана, текст, линия

Контент, сгенерированный ИИ, может содержать ошибки.

1. Створити MVC WEB-застосунок, головне подання якого містить нетипізо-вану форму для задання групи параметрів, які через контролер, що викли кається з цієї ж форми, передаються до іншого подання, на якому формуються шахова (без об'єднаних комірок) таблиця із вказаними розмірами. Парні номери варіантів кількість рядків задає на формі, а кількість стовпчиків з БД, непарні номери навпаки. Комірки таблиці послідовно запо-внюються цілими числами, починаючи з 1, тобто 1, 2, 3 і т.д. Під час формування таблиці використовуються рядкові змінні в контролері, а не цикли технології Razor.
2. Створити подання із переходами на інші подання через відповідні контролери, відповідно до схеми схемою власного варіанту.

Мій варіант:



Изображение выглядит как диаграмма, зарисовка, Технический чертеж, План

Контент, сгенерированный ИИ, может содержать ошибки.

1. Використовуючи сесійні атрибути (Session) модифікувати минуле за-вдання таким чином, щоб на кожному поданні відображався лічильник, який рахував би кількість відвідувань відповідного подання.

Контролери для роботи:

1. «Налаштування тексту» - **TextStyle**
2. «Топологія зображень» – **Image**
3. «Шахова таблиця» – **ColumnCount**
4. «Перехід по сторінкам» – **PageController**
5. «Відвідування сторінок» – **VisitCount**

Контролери textstyle, image, columncount, pagecontroller, visitcount

Контролер TextStyle

**Index()** (GET) – Отримує стилі з бази даних та передає їх у представлення. Якщо стилів менше трьох, виводить повідомлення про недостатність даних.

**Display()** (POST) – Приймає введені користувачем абзаци та параметри стилю (колір, шрифт, вирівнювання), передає їх у представлення для відображення форматованого тексту.

using Microsoft.AspNetCore.Mvc;

using Microsoft.EntityFrameworkCore;

using Task\_1.Data;

using Task\_1.Models;

namespace Task\_1.Controllers

{

public class HomeController : Controller

{

private readonly ApplicationDbContext \_context;

public HomeController(ApplicationDbContext context)

{

\_context = context;

}

public IActionResult Index()

{

var styles = \_context.TextStyles.ToList();

if (styles.Count < 3)

{

return Content("В базі недостатньо стилів для відображення.");

}

ViewBag.Styles = styles;

return View();

}

[HttpPost]

public IActionResult Display(string paragraph1, string paragraph2, string paragraph3, string textColor, string fontSize, string fontStyle, string textAlign)

{

var styles = \_context.TextStyles.ToList();

ViewBag.Paragraphs = new List<string> { paragraph1, paragraph2, paragraph3 };

ViewBag.Styles = styles;

ViewBag.TextColor = textColor;

ViewBag.FontSize = fontSize;

ViewBag.FontStyle = fontStyle;

ViewBag.TextAlign = textAlign;

return View();

}

}

}

Контролер Image

**Index()** (GET) – Відображає головну сторінку з формою для введення URL-адрес.

**Submit()** (POST) – Приймає список URL з форми, оновлює відповідні записи в базі та передає їх у представлення для відображення.

using Microsoft.AspNetCore.Mvc;

using Task\_2.Data;

public class HomeController : Controller

{

private readonly ApplicationDbContext \_context;

public HomeController(ApplicationDbContext context)

{

\_context = context;

}

public IActionResult Index()

{

return View();

}

[HttpPost]

public IActionResult Submit(List<string> urls)

{

var images = \_context.Images.ToList();

for (int i = 0; i < images.Count && i < urls.Count; i++)

{

images[i].Url = urls[i];

}

return View("Display", images);

}

}

Контролер ColumnCount

**Index()** (GET) – Відображає форму для введення кількості рядків.

**GenerateTable(int rows)** (POST) – Отримує кількість рядків, зчитує кількість стовпців з бази даних, генерує HTML-код таблиці та передає його у представлення.

using Microsoft.AspNetCore.Mvc;

using Task\_3.Data;

public class HomeController : Controller

{

private readonly ApplicationDbContext \_context;

public HomeController(ApplicationDbContext context)

{

\_context = context;

}

[HttpGet]

public IActionResult Index()

{

return View();

}

[HttpPost]

public IActionResult GenerateTable(int rows)

{

var columns = \_context.ColumnCounts.FirstOrDefault()?.Count ?? 5;

string tableHtml = GenerateTableHtml(rows, columns);

ViewData["TableHtml"] = tableHtml;

return View("Display");

}

private string GenerateTableHtml(int rows, int columns)

{

int number = 1;

string table = "<table class='chess-table'>";

for (int i = 0; i < rows; i++)

{

table += "<tr>";

for (int j = 0; j < columns; j++)

{

string cellClass = (i + j) % 2 == 0 ? "white-cell" : "black-cell";

table += $"<td class='{cellClass}'>{number++}</td>";

}

table += "</tr>";

}

table += "</table>";

return table;

}

}

Контролер PageController

**BaseController.Index()** – Повертає представлення відповідної сторінки для всіх контролерів-нащадків.

**Контролери (HomeController, Page2Controller, Page3Controller, Page4Controller, Page5Controller)**

* + Використовують метод Index() базового контролера для відображення сторінок.

using Microsoft.AspNetCore.Mvc;

public class BaseController : Controller

{

public IActionResult Index()

{

return View();

}

}

public class HomeController : BaseController { }

public class Page2Controller : BaseController { }

public class Page3Controller : BaseController { }

public class Page4Controller : BaseController { }

public class Page5Controller : BaseController { }

Контролер VisitCount

**Index() у BaseController**

* + Підраховує кількість відвідувань сторінки та зберігає значення у сесії.
  + Передає значення в ViewData, щоб відобразити його на сторінці.

using Microsoft.AspNetCore.Mvc;

public class BaseController : Controller

{

public virtual IActionResult Index()

{

string key = $"VisitCount\_{this.GetType().Name}";

int count = HttpContext.Session.GetInt32(key) ?? 0;

count++;

HttpContext.Session.SetInt32(key, count);

ViewData["VisitCount"] = count;

return View();

}

}

public class HomeController : BaseController { }

public class Page2Controller : BaseController { }

public class Page3Controller : BaseController { }

public class Page4Controller : BaseController { }

public class Page5Controller : BaseController { }

Висновок

У ході написання цієї лабораторної роботи я ознайомився з принципами розробки простих MVC WEB-застосунків засобами ASP.NET. Я навчився працювати з контролерами та поданнями в середовищі ASP.NET Core. Реалізував п’ять контролерів: TextStyle, Image, ColumnCount, PageController, VisitCount, використав Bootstrap і CSS для стилізації інтерфейсу.

Також я освоїв принципи створення та використання контролерів у ASP.NET MVC. Контролери відповідають за обробку запитів, взаємодію з моделями та передачу даних у відповідні подання.

Реалізація контролерів та їх взаємодія з поданнями дала змогу зрозуміти, як обробляються запити користувачів і формується відповідь. Загалом, ця робота допомогла мені краще зрозуміти архітектуру MVC та її практичне застосування в ASP.NET.