Міністерство освіти України

Національний технічний університет "ХПІ"

кафедра "Інформатики та інтелектуальної власності"

**Звіт**

**Лабораторна робота 2**

з дисципліни " Проєктування серверних застосунків "

Виконав: студент групи КН-922В

Шершньов М.О.

Перевірив:

Івашко А.В.

Харків 2025

Зміст

[Завдання 3](#_Toc192699939)

[Класи request, iformcollection, weaktypification, stricttypification 6](#_Toc192699940)

[Клас Request 6](#_Toc192699941)

[Клас IFormCollection 8](#_Toc192699942)

[Клас WeakTypification 10](#_Toc192699943)

[Клас StrongTypification 12](#_Toc192699944)

[Висновок 14](#_Toc192699945)

**Шляхи використання можливостей форм у засобах ASP.NET MVC WEB-застосунків. Частина 1.**

**Мета роботи:** набути практику та досвід у використанні різноманітних шля хів отримання даних із форм на поданнях під час розробки та використання MVC WEB-застосунків.

Завдання

1. Створити застосунок, стартове подання (Index) якого містить форму згідно до власного варіанту. За допомогою стандартного класу Request контролер, викликаний кнопкою Відіслати приймає дані з форми та передає їх назад до подання для їхньої демонстрації під формою, кнопка Скинути має повертати форму у «дефолтний» (початковий) стан, тобто очистити текстові поля

Мій варіант:

Изображение выглядит как текст, снимок экрана, Шрифт, Прямоугольник

Контент, сгенерированный ИИ, может содержать ошибки.

1. Створити застосунок, стартове подання (Index) якого містить форму згідно до власного варіанту. За допомогою стандартного класу Request контролер, викликаний кнопкою Відіслати приймає дані з форми та передає їх назад до подання для їхньої демонстрації під формою, кнопка Скинути має повертати форму у «дефолтний» (початковий) стан, тобто відновити комбінацію вибраних радіо-кнопок

Мій варіант:

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Контент, сгенерированный ИИ, может содержать ошибки.

1. Створити застосунок, стартове подання (Index) якого містить форму згідно до власного варіанту. За допомогою стандартного класу Request контролер, викликаний кнопкою Відіслати приймає дані з форми та передає їх назад до подання для їхньої демонстрації під формою, кнопка Скинути має повертати форму у «дефолтний» (початковий) стан, тобто відновити комбінацію вибраних «чекбоксів»

Мій варіант:

Изображение выглядит как текст, снимок экрана, Шрифт, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

1. Створити застосунок, стартове подання (Index) якого містить форму зі списком, що розкривається (combo box, або «комбобокс») з N містами країни згідно свого варіанту та список, що прокручується (list box «листбокс») з аналогічним контентом. За допомогою стандартного класу Request контролер, викликаний кнопкою Відіслати приймає дані з форми та передає їх назад до подання для їхньої демонстрації під формою



1. Виконати функції минулого завдання (кожної з 4 форм), використавши для отримання даних форми замість класу Request колекцію типу IFormCollection
2. Виконати функції першого завдання (кожної з 4 форм), використавши для отримання да-них форми замість класу Request слабку типізацію форми, додавши до кожного поля форми відповідну змінну.
3. Виконати функції першого завдання (кожної з 4 форм), використавши для отримання даних форми замість класу Request сувору типізацію форми, додавши до полів форми відповідну модель.

Класи для роботи:

1. «Запит» - **Request**
2. «Колекція» – **IFormCollection**
3. «Слабка типізація» – **WeakTypification**
4. «Сувора типізація» – **StrongTypification**

Класи request, iformcollection, weaktypification, stricttypification

Клас Request

**Request** – це об'єкт у Controller, який містить усю інформацію про HTTP-запит (GET, POST, Cookies, Headers тощо).

Можна отримувати параметри форми (Request.Form), параметри URL (Request.Query) та заголовки запиту (Request.Headers).

using Microsoft.AspNetCore.Mvc;

public class HomeController : Controller

{

public IActionResult Index()

{

if (Request.Method == "POST")

{

string? name = Request.Form["name"];

string? phone = Request.Form["phone"];

string? email = Request.Form["email"];

string? birthdate = Request.Form["birthdate"];

ViewBag.Name = name ?? "Не вказано";

ViewBag.Phone = phone ?? "Не вказано";

ViewBag.Email = email ?? "Не вказано";

ViewBag.Birthdate = birthdate ?? "Не вказано";

}

return View();

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_5.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewBag.SelectedCity = "Київ";

ViewBag.SelectedTransport = "Літак";

ViewBag.ShowSelection = false;

return View();

}

[HttpPost]

public IActionResult ProcessSelection()

{

string city = Request.Form["city"].FirstOrDefault() ?? "Київ";

string transport = Request.Form["transport"].FirstOrDefault() ?? "Літак";

ViewBag.SelectedCity = city;

ViewBag.SelectedTransport = transport;

ViewBag.ShowSelection = true;

return View("Index");

}

}

}

using Microsoft.AspNetCore.Mvc;

using System.Linq;

namespace Task\_9.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewData["SelectedValues"] = "Спорт";

return View();

}

[HttpPost]

public IActionResult ProcessInterests()

{

var interests = Request.Form["interests"];

string selectedInterests = interests.Where(i => !string.IsNullOrEmpty(i)).Any()

? string.Join(", ", interests.Where(i => !string.IsNullOrEmpty(i)))

: "Спорт";

ViewData["SelectedInterests"] = selectedInterests;

ViewData["SelectedValues"] = selectedInterests;

return View("Index");

}

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_13.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewData["SelectedCity"] = null;

ViewData["SelectedCities"] = new string[] { };

return View();

}

[HttpPost]

public IActionResult ProcessSelection()

{

string selectedCity = Request.Form["comboBoxCities"].FirstOrDefault() ?? "";

string[] selectedCities = Request.Form["listBoxCities"].ToArray();

ViewData["SelectedCity"] = selectedCity;

ViewData["SelectedCities"] = selectedCities;

return View("Index");

}

}

}

Клас IFormCollection

**IFormCollection –** це колекція всіх даних POST-форми, яка дозволяє отримувати значення у вигляді Dictionary<string, StringValues>.

Використовується для отримання всіх полів форми одним параметром.

using Microsoft.AspNetCore.Mvc;

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

return View();

}

[HttpPost]

public IActionResult Index(IFormCollection form)

{

string? name = form["name"];

string? phone = form["phone"];

string? email = form["email"];

string? birthdate = form["birthdate"];

ViewBag.Name = name ?? "Не вказано";

ViewBag.Phone = phone ?? "Не вказано";

ViewBag.Email = email ?? "Не вказано";

ViewBag.Birthdate = birthdate ?? "Не вказано";

return View();

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_6.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewBag.SelectedCity = "Київ";

ViewBag.SelectedTransport = "Літак";

ViewBag.ShowSelection = false;

return View();

}

[HttpPost]

public IActionResult Index(IFormCollection form)

{

string city = form["city"].ToString() ?? "Київ";

string transport = form["transport"].ToString() ?? "Літак";

ViewBag.SelectedCity = city;

ViewBag.SelectedTransport = transport;

ViewBag.ShowSelection = true;

return View();

}

}

}

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Http;

using System.Linq;

namespace Task\_10.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewData["SelectedValues"] = new string[] { "Спорт" };

return View();

}

[HttpPost]

public IActionResult Index(IFormCollection form)

{

var selectedInterests = form["interests"].Where(interest => !string.IsNullOrEmpty(interest)).ToArray();

if (selectedInterests.Length == 0)

{

selectedInterests = new string[] { "Спорт" };

}

ViewData["SelectedInterests"] = string.Join(", ", selectedInterests);

ViewData["SelectedValues"] = selectedInterests;

return View();

}

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_14.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewData["SelectedCity"] = null;

ViewData["SelectedCities"] = new string[] { };

return View();

}

[HttpPost]

public IActionResult ProcessSelection(IFormCollection form)

{

string selectedCity = form["comboBoxCities"].FirstOrDefault() ?? "";

string[] selectedCities = form["listBoxCities"].ToArray();

ViewData["SelectedCity"] = selectedCity;

ViewData["SelectedCities"] = selectedCities;

return View("Index");

}

}

}

Клас WeakTypification

Слабка типізація означає, що дані передаються у вигляді string без явного зв’язку з типом.

Використовується Request.Form або IFormCollection.

using Microsoft.AspNetCore.Mvc;

using Microsoft.AspNetCore.Mvc;

public class HomeController : Controller

{

public IActionResult Index()

{

if (Request.Method == "POST")

{

var form = Request.Form;

string name = form["name"].ToString();

string phone = form["phone"].ToString();

string email = form["email"].ToString();

string birthdate = form["birthdate"].ToString();

ViewBag.Name = !string.IsNullOrEmpty(name) ? name : "Не вказано";

ViewBag.Phone = !string.IsNullOrEmpty(phone) ? phone : "Не вказано";

ViewBag.Email = !string.IsNullOrEmpty(email) ? email : "Не вказано";

ViewBag.Birthdate = !string.IsNullOrEmpty(birthdate) ? birthdate : "Не вказано";

}

return View();

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_7.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewBag.SelectedCity = "Київ";

ViewBag.SelectedTransport = "Літак";

ViewBag.ShowSelection = false;

return View();

}

[HttpPost]

public IActionResult Index(string? city, string? transport)

{

ViewBag.SelectedCity = city ?? "Київ";

ViewBag.SelectedTransport = transport ?? "Літак";

ViewBag.ShowSelection = true;

return View();

}

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_11.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewBag.SelectedInterests = null;

return View();

}

[HttpPost]

public IActionResult ProcessInterests(string? sport, string? travel, string? crafting, string? drawing)

{

List<string> selectedInterests = new();

if (!string.IsNullOrEmpty(sport)) selectedInterests.Add("Спорт");

if (!string.IsNullOrEmpty(travel)) selectedInterests.Add("Мандрівки");

if (!string.IsNullOrEmpty(crafting)) selectedInterests.Add("Майстрування");

if (!string.IsNullOrEmpty(drawing)) selectedInterests.Add("Малювання");

ViewBag.SelectedInterests = selectedInterests.Count > 0 ? selectedInterests : null;

return View("Index");

}

[HttpPost]

public IActionResult Reset()

{

ViewBag.SelectedInterests = null;

return View("Index");

}

}

}

using Microsoft.AspNetCore.Mvc;

namespace Task\_15.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

ViewData["SelectedCity"] = null;

ViewData["SelectedCities"] = null;

return View();

}

[HttpPost]

public IActionResult ProcessSelection(string? comboBoxCities, string[]? listBoxCities)

{

string selectedCity = comboBoxCities ?? "Братислава";

string[] selectedCities = listBoxCities ?? new string[] { };

ViewData["SelectedCity"] = selectedCity;

ViewData["SelectedCities"] = selectedCities;

return View("Index");

}

}

}

Клас StrongTypification

Сувора типізація використовує **класи-моделі** (Model), де чітко визначені типи полів.

Використовується asp-for у <input> для автоматичної прив'язки даних.

using Microsoft.AspNetCore.Mvc;

using Task\_4.Models;

public class HomeController : Controller

{

public IActionResult Index()

{

return View();

}

[HttpPost]

public IActionResult Index(UserFormModel model)

{

ViewBag.Name = !string.IsNullOrEmpty(model.Name) ? model.Name : "Не вказано";

ViewBag.Phone = !string.IsNullOrEmpty(model.Phone) ? model.Phone : "Не вказано";

ViewBag.Email = !string.IsNullOrEmpty(model.Email) ? model.Email : "Не вказано";

ViewBag.Birthdate = !string.IsNullOrEmpty(model.Birthdate) ? model.Birthdate : "Не вказано";

return View();

}

}

using Microsoft.AspNetCore.Mvc;

using Task\_8.Models;

namespace Task\_8.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

return View(new SelectionViewModel());

}

[HttpPost]

public IActionResult Index(SelectionViewModel model)

{

model.ShowSelection = true;

return View(model);

}

[HttpPost]

public IActionResult Reset()

{

return View("Index", new SelectionViewModel());

}

}

}

using Microsoft.AspNetCore.Mvc;

using Task\_12.Models;

namespace Task\_12.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

return View(new InterestModel());

}

[HttpPost]

public IActionResult Index(InterestModel model)

{

List<string> selectedInterests = new();

if (model.Sport) selectedInterests.Add("Спорт");

if (model.Travel) selectedInterests.Add("Мандрівки");

if (model.Crafting) selectedInterests.Add("Майстрування");

if (model.Drawing) selectedInterests.Add("Малювання");

ViewData["SelectedInterests"] = selectedInterests.Count > 0

? string.Join(", ", selectedInterests)

: null;

return View(model);

}

[HttpPost]

public IActionResult Reset()

{

return View("Index", new InterestModel());

}

}

}

using Microsoft.AspNetCore.Mvc;

using Task\_16.Models;

namespace Task\_16.Controllers

{

public class HomeController : Controller

{

[HttpGet]

public IActionResult Index()

{

return View(new CitySelectionModel());

}

[HttpPost]

public IActionResult ProcessSelection(CitySelectionModel model)

{

if (model.ListBoxCities == null || model.ListBoxCities.Count == 0)

{

model.ListBoxCities = new List<string> { "Нічого не вибрано" };

}

ViewData["SelectedCity"] = model.ComboBoxCity;

ViewData["SelectedCities"] = string.Join(", ", model.ListBoxCities);

return View("Index", model);

}

}

}

Висновок

У ході виконання цієї лабораторної роботи я ознайомився зі шляхами використання можливостей форм у засобах ASP.NET MVC WEB-застосунків. Я навчився отримувати та обробляти дані форм за допомогою різних методів, таких як Request, IFormCollection, слабка та сувора типізація.

Під час роботи я реалізував обробку форм через контролери, використовуючи Request.Form для швидкого отримання даних, IFormCollection для гнучкого доступу до всіх полів форми, слабку типізацію (string, string[]) для простих форм і сувору типізацію (моделі, наприклад, CitySelectionModel) для структурованого підходу та валідації.

Я також використовував asp-for у Razor-розмітці для зв’язку елементів форми з моделями, що значно спрощує передачу даних між поданням і контролером. Завдяки цьому я краще зрозумів, як у ASP.NET MVC обробляються введені користувачем дані та формується відповідь сервера.

Загалом, ця робота допомогла мені освоїти принципи роботи з формами, їх інтеграцію з контролерами, а також навчила вибирати відповідний метод обробки залежно від вимог конкретного застосунку.