

Лабораторна робота №3

Часткові представлення. Робота з View Components. Навігація.

Фільтрація. Використання сесії.

Мета: отримати навички роботи з частковими представленнями, навчитися працювати з параметрами рядку запиту, набути навичок роботи з сесіями.

Хід роботи

Завдання 1. Створення часткового представлення (Partial View)

1. У папці Views/Shared створіть .cshtml файл для одиничного елемента зі списку сутностей (для JobPortal це буде UserSummary.cshtml)

Лістинг:

```
@model JobPortal.Models.User

<tr>
    <td>@Model.FullName</td>
    <td>@Model.Email</td>
    <td>@Model.Role</td>
    <td>@Model.CreatedAt.ToString("dd.MM.yyyy HH:mm")</td>
</tr>
```

2. Перемістіть код для одиничного елемента з загального View Index.cshtml в створений файл.

Лістинг:

```
@{
    ViewData["Title"] = "Home Page";
}

@model JobPortal.Models.ViewModels.UsersListViewModel

<h2 class="mb-4">Список користувачів</h2>

<table class="table table-striped table-bordered">
    <thead class="table-primary">
        <tr>
            <th>Ім'я</th>
            <th>Email</th>
            <th>Роль</th>
            <th>Дата створення</th>
        </tr>
    </thead>
    <tbody>
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.3							
Змн.	Арк.	№ докум.	Підпис	Дата								
Розроб.		Трофімчук М.О.			Звіт з лабораторної роботи №3				Літ.	Арк.	Аркушіє	
Перевір.		Українець М.О.									1	9
Реценз.									ФІКТ, гр. ІПЗ-22-2			
Н. Контр.												
Зав.каф.												

```

        @foreach (var user in Model.Users)
        {
            <partial name="_UserSummary" model="user" />
        }
    </tbody>
</table>

<nav aria-label="Page navigation" class="mt-4">
    <ul class="pagination justify-content-center">
        @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-page="@i">@i</a>
            </li>
        }
    </ul>
</nav>

```

3. Використайте створений Partial View в Index.cshtml за допомогою Tag Helper'a <partial>

Лістинг:

```

@foreach (var user in Model.Users)
{
    <partial name="_UserSummary" model="user" />
}

```

Завдання 2. Навігація по категоріям

Реалізуйте навігацію за категорією (або за іншою ознакою, яка більше підходить за вашим варіантом), дотримуючись наступних кроків:

1. В папці проекту створіть папку Components, в ній створіть файл з назвою NavigationMenuViewComponent.cs та реалізуйте його

Лістинг:

```

using Microsoft.AspNetCore.Mvc;
using JobPortal.Models;

namespace JobPortal.Components
{
    public class NavigationMenuViewComponent : ViewComponent
    {
        private readonly IPortalRepository repository;

        public NavigationMenuViewComponent(IPortalRepository repo)
        {
            repository = repo;
        }

        public IViewComponentResult Invoke()
        {
            // Отримуємо унікальні ролі користувачів (категорії)
            var roles = repository.Users
                .Select(u => u.Role)
                .Distinct()
                .OrderBy(r => r);
        }
    }
}

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.3	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		2

```

        // Передаємо обрану роль через ViewBag
        ViewBag.SelectedRole = RouteData?.Values["role"];

        return View(roles);
    }
}

```

2. В загальному шаблоні (файл `_Layout.cshtml`) додайте розмітку для сайд-бару. В доданій розмітці додайте виклик створеного `NavigationMenuViewComponent`.

Лістинг:

```

<main role="main" class="flex-grow-1 py-4">
    <div class="container">
        <div class="row">
            <div id="sidebar" class="col-md-3">
                <div class="p-3 bg-white rounded shadow-sm">
                    <h5 class="mb-3 border-bottom pb-2">Навігація</h5>
                    @await Component.InvokeAsync("NavigationMenu")
                </div>
            </div>

            <div id="content" class="col-md-9">
                @RenderBody()
            </div>
        </div>
    </div>
</main>

```

3. Змініть реалізацію `NavigationMenuViewComponent` таким чином, щоб компонент отримував перелік категорій з БД і повертав результат типу `IViewComponentResult`.

Лістинг:

```

using Microsoft.AspNetCore.Mvc;
using JobPortal.Models;

namespace JobPortal.Components
{
    public class NavigationMenuViewComponent : ViewComponent
    {
        private readonly PortalDbContext _context;

        public NavigationMenuViewComponent(PortalDbContext context)
        {
            _context = context;
        }

        public IViewComponentResult Invoke()
        {
            // Отримуємо унікальні ролі користувачів
            var roles = _context.Users
                .Select(u => u.Role)

```

```

        .Distinct()
        .OrderBy(r => r)
        .ToList();

        ViewBag.SelectedRole = RouteData?.Values["role"];

        return View(roles);
    }
}

```

- В папці Views/Shared/Components/NavigationMenu створіть файл Default.cshtml. Детальніше про конвенції неймінгу, які використовує Razor для роботи з ViewComponents можна [почитати тут](#).
- В створеному файлі представлення реалізуйте виведення списку посилань, отриманих з ViewComponent'a.

Лістинг:

```

@model IEnumerable<string>

<div class="list-group mb-3">
    <a asp-action="Index" asp-controller="Home"
        class="list-group-item list-group-item-action @(ViewBag.SelectedRole ==
null ? "active" : "")">
        Усі користувачі
    </a>

    @foreach (var role in Model)
    {
        <a asp-action="Index" asp-controller="Home"
            asp-route-role="@role"
            class="list-group-item list-group-item-action @(role ==
ViewBag.SelectedRole ? "active" : "")">
            @role
        </a>
    }
</div>

```

- У файлі Controllers/HomeController.cs в методі Index() реалізуйте підтримку фільтрації по категоріям.

Лістинг:

```

public IActionResult Index(string? role, int page = 1)
{
    var usersQuery = _context.Users.AsQueryable();

    if (!string.IsNullOrEmpty(role))
    {
        usersQuery = usersQuery.Where(u => u.Role == role);
    }

    usersQuery = usersQuery.OrderBy(u => u.Id);

    var model = new UsersListViewModel
    {
        Users = usersQuery
    }
}

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.3	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		4

```

        .Skip((page - 1) * PageSize)
        .Take(PageSize)
        .ToList(),

        PagingInfo = new PagingInfo
        {
            CurrentPage = page,
            ItemsPerPage = PageSize,
            TotalItems = usersQuery.Count()
        },

        CurrentRole = role
    };

    return View(model);
}

```

7. Змініть ViewModel таким чином, щоб мати інформацію про обрану категорію на представленні. Це знадобиться для формування посилань на сторінки з урахуванням обраної категорії.

Лістинг:

```

using JobPortal.Models;
using JobPortal.Views.ViewModels;
using System.Collections.Generic;

namespace JobPortal.Models.ViewModels
{
    public class UsersListViewModel
    {
        public IEnumerable<User> Users { get; set; } = Enumerable.Empty<User>();
        public PagingInfo PagingInfo { get; set; } = new PagingInfo();
        public string? CurrentRole { get; set; }
    }
}

```

8. Реалізуйте виділення категорії, обраної користувачем. Для цього передавайте інформацію про обрану категорію з View Component в View.
9. Змініть реалізацію пейджингу таким чином, щоб він відображав кількість доступних сторінок відповідно до обраної категорії.

Лістинг:

```

<nav aria-label="Page navigation">
    <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-page="@i"
                    asp-route-role="@Model.CurrentRole">@i</a>

            </li>
        }
    </ul>
</nav>

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.3	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		5

```

    }
  </ul>
</nav>

```

У результаті виконання завдання, отримуємо сторінку:

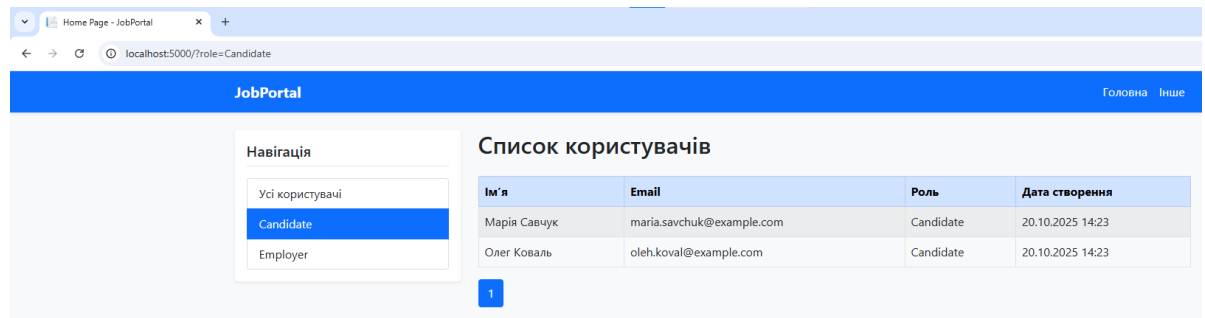


Рис. 3.2.1. Результат реалізації фільтрації

Завдання 3. Сесії

Реалізуйте функціональність відповідно до варіанту вашого проекту з використанням механізму [сесії](#) (табл. 3.1.)

Таблиця 3.1. Функціональності для реалізації

Номер за списком групи	Назва проекту	Опис функціональності
3	Портал вакансій	Збереження незавершених заявок на вакансії у сесії, щоб кандидат міг продовжити заповнення форми після переривання.

Додаємо підтримку сесій у Program.cs.

Лістинг:

```

...
// підтримка сесій
builder.Services.AddDistributedMemoryCache();
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(30); // час життя сесії
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
});

var app = builder.Build();
...
app.UseSession();

```

Створюємо модель заявки Models/ApplicationForm.cs:

ЛІСТИНГ:

```
namespace JobPortal.Models
{
    public class ApplicationForm
    {
        public string FullName { get; set; }
        public string Email { get; set; }
        public string Position { get; set; }
        public string ResumeText { get; set; }
    }
}
```

Створюємо контролер ApplicationController.

ЛІСТИНГ:

```
using JobPortal.Models;
using Microsoft.AspNetCore.Mvc;
using Newtonsoft.Json;

public class ApplicationController : Controller
{
    private const string SessionKey = "ApplicationForm";

    public IActionResult Apply()
    {
        // Якщо є дані в сесії – відновлюємо
        var json = HttpContext.Session.GetString(SessionKey);
        ApplicationForm model = string.IsNullOrEmpty(json)
            ? new ApplicationForm()
            : JsonConvert.DeserializeObject<ApplicationForm>(json);

        return View(model);
    }

    [HttpPost]
    public IActionResult Apply(ApplicationForm model)
    {
        // Зберігаємо поточний стан у сесії
        var json = JsonConvert.SerializeObject(model);
        HttpContext.Session.SetString(SessionKey, json);

        ViewBag.Message = "Заявку збережено в сесії. Можна продовжити пізніше.";
        return View(model);
    }

    public IActionResult Submit()
    {
        // Отримуємо фінальні дані
        var json = HttpContext.Session.GetString(SessionKey);
        if (json == null)
        {
            return RedirectToAction("Apply");
        }
    }
}
```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.3	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		7

```

        var model = JsonConvert.DeserializeObject<ApplicationForm>(json);

        // TODO: збереження в базу даних

        // очищаємо сесію після відправки
        HttpContext.Session.Remove(SessionKey);

        ViewBag.Message = "Заявку успішно подано!";
        return View("Result", model);
    }
}

```

Створюємо відповідне View (Apply.cshtml).

ЛІСТИНГ:

```

@model ApplicationForm

<h2>Заявка на вакансію</h2>

<form asp-action="Apply" method="post">
    <div>
        <label>ПІБ:</label>
        <input asp-for="FullName" />
    </div>

    <div>
        <label>Email:</label>
        <input asp-for="Email" />
    </div>

    <div>
        <label>Посада:</label>
        <input asp-for="Position" />
    </div>

    <div>
        <label>Резюме:</label>
        <textarea asp-for="ResumeText"></textarea>
    </div>

    <button type="submit">Зберегти у сесії</button>
</form>

<form asp-action="Submit" method="post">
    <button type="submit">Надіслати</button>
</form>

@if (ViewBag.Message != null)
{
    <p style="color:green">@ViewBag.Message</p>
}

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.3	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		8

Результат роботи:

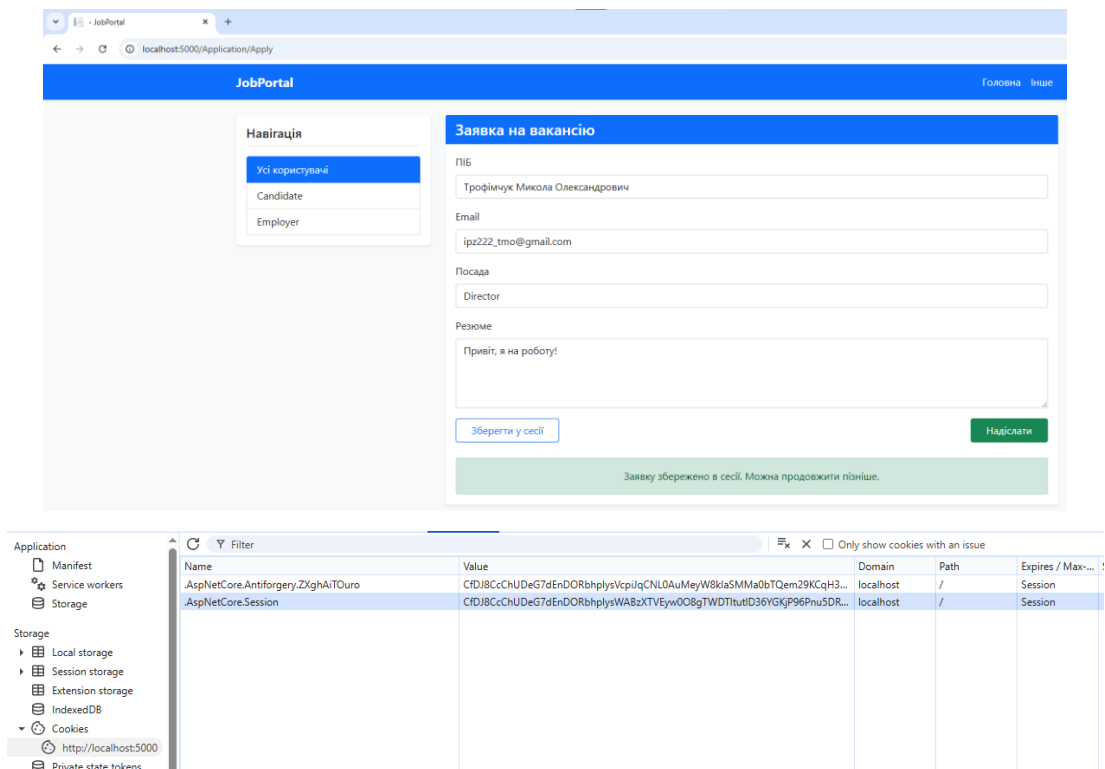


Рис. 3.3.1-2. Результат функціональності сесій

Висновок: в ході виконання лабораторної роботи було отримано навички роботи з частковими представленнями, навчено працювати з параметрами рядку запиту, набуто навичок роботи з сесіями. Роботу виконано в повному обсязі.