

Лабораторна робота №2

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.

Номер за списком	Тема проекту	Короткий опис функціоналу
23 (3)	Портал вакансій	Роботодавці можуть публікувати списки вакансій, а кандидати можуть відгукуватися на вакансії та керувати своїми відгуками.

Налаштування HTTP-порту, який ASP.NET Core буде використовувати для прослуховування HTTP-запитів.

Лістинг програми:

```
{
  "iisSettings": {
    "windowsAuthentication": false,
    "anonymousAuthentication": true,
    "iisExpress": {
      "applicationUrl": "http://localhost:5000",
      "sslPort": 0
    }
  },
  "profiles": {
    "SportsStore": {
      "commandName": "Project",
      "dotnetRunMessages": true,
      "launchBrowser": true,
      "applicationUrl": "http://localhost:5000",
    }
  }
}
```

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

```

        "environmentVariables": {
            "ASPNETCORE_ENVIRONMENT": "Development"
        }
    },
    "IIS Express": {
        "commandName": "IISExpress",
        "launchBrowser": true,
        "environmentVariables": {
            "ASPNETCORE_ENVIRONMENT": "Development"
        }
    }
}
}
}

```

Було проведено налаштування сервісів та пайплайну обробки запитів, рушія представлень (View Engine) Razor. Додано файл Razor View під назвою _ViewStart.cshtml до папки Views, файл класу з назвою HomeController.cs у папку /Controllers.

Далі перейдемо до створення першої сутності моделі даних.

Лістинг (Models/User):

```

namespace JobPortal.Models
{
    public class User
    {
        public int Id { get; set; }

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

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

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

        public string Role { get; set; } = string.Empty; //
        "Employer", "Candidate"

        public DateTime CreatedAt { get; set; } = DateTime.UtcNow;
    }
}

```

Запустіть проект для перевірки працездатності.

На рисунку 1.1. зображено результат запуску проекту JobPortal.

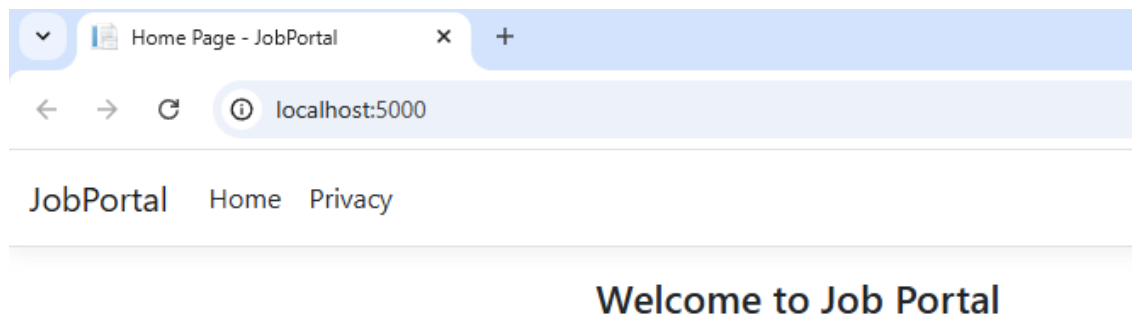


Рис. 1.1. Результат запуску проекту JobPortal

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

Завдання 2. Робота з ORM. Налаштувати роботу з БД

connection string до бази даних, яка буде використовуватися для збереження даних вашого проекту.

Лістинг (appsettings.json):

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "JobPortalConnection":
      "Server=(localdb)\\MSSQLLocalDB;Database=JobPortal;MultipleActiveResultSets=true"
  }
}
```

Реалізуйте клас контексту відповідно до потреб вашого проекту.

Entity Framework Core надає доступ до бази даних через клас контексту. Для проекту JobPortal буде додано файл класу з ім'ям PortalDbContext.cs до папки Models.

Лістинг (PortalDbContext):

```
using Microsoft.EntityFrameworkCore;

namespace JobPortal.Models
{
    public class PortalDbContext : DbContext
    {
        public PortalDbContext(DbContextOptions<PortalDbContext> options) :
            base(options) { }
        public DbSet<User> Users => Set<User>();
    }
}
```

Наступним кроком буде створення інтерфейсу репозиторію та його реалізації.

Лістинг (IPortalRepository):

```
namespace JobPortal.Models
{
    public interface IPortalRepository
    {
        IQueryable<User> Users { get; }
    }
}
```

Щоб створити реалізацію інтерфейсу репозиторію, додамо файл класу з іменем EFPortalRepository.cs в папку Models.

Лістинг (IPortalRepository):

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

```

namespace JobPortal.Models
{
    public class EFPortalRepository : IPortalRepository
    {
        private PortalDbContext context;
        public EFPortalRepository(PortalDbContext ctx)
        {
            context = ctx;
        }
        public IQueryable<User> Users => context.Users;
    }
}

```

Виконання команди `dotnet ef migrations add Initial`:

```

Time Elapsed 00:00:02.14
PS D:\4 курс\ASP Net\Project\JobPortal> dotnet ef migrations add Initial
Build started...
Build succeeded.
Done. To undo this action, use 'ef migrations remove'
PS D:\4 курс\ASP Net\Project\JobPortal>

```

Рис. 2.1. Результат успішного виконання команди

Наповніть вашу БД початковими даними за зразком нижче.

Щоб наповнити БД даними одразу після створення, реалізуємо статичний клас `SeedData` в папці `Models`. Лістинг файлу `SeedData.cs`:

```

using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.DependencyInjection;
using System;
using System.Linq;
using static System.Net.Mime.MediaTypeNames;

namespace JobPortal.Models
{
    public static class SeedData
    {
        public static void EnsurePopulated(IApplicationBuilder app)
        {
            using var scope = app.ApplicationServices.CreateScope();
            var context =
                scope.ServiceProvider.GetRequiredService<PortalDbContext>();

            // Якщо є незастосовані міграції – застосувати
            if (context.Database.GetPendingMigrations().Any())
            {
                context.Database.Migrate();
            }

            if (!context.Users.Any())
            {
                var user1 = new User
                {
                    FullName = "Іван Петренко",
                    Email = "ivan.petrenko@example.com",
                    PasswordHash = "12345", // тимчасово, для демо
                    Role = "Employer",
                    CreatedAt = DateTime.UtcNow
                };

                var user2 = new User
                {

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.2	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		4

```

        FullName = "Марія Савчук",
        Email = "maria.savchuk@example.com",
        PasswordHash = "12345",
        Role = "Candidate",
        CreatedAt = DateTime.UtcNow
    };

    var user3 = new User
    {
        FullName = "Олег Коваль",
        Email = "oleh.koval@example.com",
        PasswordHash = "12345",
        Role = "Candidate",
        CreatedAt = DateTime.UtcNow
    };

    var user4 = new User
    {
        FullName = "Катерина Катренко",
        Email = "katya@example.com",
        PasswordHash = "123123",
        Role = "Employer",
        CreatedAt = DateTime.UtcNow
    };

    context.Users.AddRange(user1, user2, user3, user4);
    context.SaveChanges();
}
}
}
}
}

```

Лістинг (Progtam.cs (після всіх модифікацій)):

```

using JobPortal.Models;
using Microsoft.EntityFrameworkCore;
using System.Reflection.Metadata.Ecma335;

namespace JobPortal
{
    public class Program
    {
        public static void Main(string[] args)
        {
            var builder = WebApplication.CreateBuilder(args);

            // Add services to the container.
            builder.Services.AddControllersWithViews();

            builder.Services.AddDbContext<PortalDbContext>(opts => {
                opts.UseSqlServer(builder.Configuration["ConnectionStrings:JobPortalCo
nnection"]);
            });

            builder.Services.AddScoped<IPortalRepository,
EFPortalRepository>();

            var app = builder.Build();

            ...

            app.UseHttpsRedirection();
            app.UseStaticFiles();

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.2	Арк.
						5
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        app.UseRouting();

        app.UseAuthorization();

        app.MapControllerRoute(
            name: "default",
            pattern: "{controller=Home}/{action=Index}/{id?}");

        SeedData.EnsurePopulated(app);

        app.Run();
    }
}

```

Завдання 3. Виведення даних на сторінці

Змініть контролер представлення для виведення даних з БД.

Для JobPortal це виглядатиме наступним чином:

Лістинг (HomeController):

```

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

namespace JobPortal.Controllers
{
    public class HomeController : Controller
    {
        private readonly PortalDbContext _context;

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

        public IActionResult Index()
        {
            var users = _context.Users.ToList();
            return View(users);
        }
    }
}

```

Лістинг (Views/Home/Index.cshtml):

```

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

@model IEnumerable<JobPortal.Models.User>

<h2>Список користувачів</h2>

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

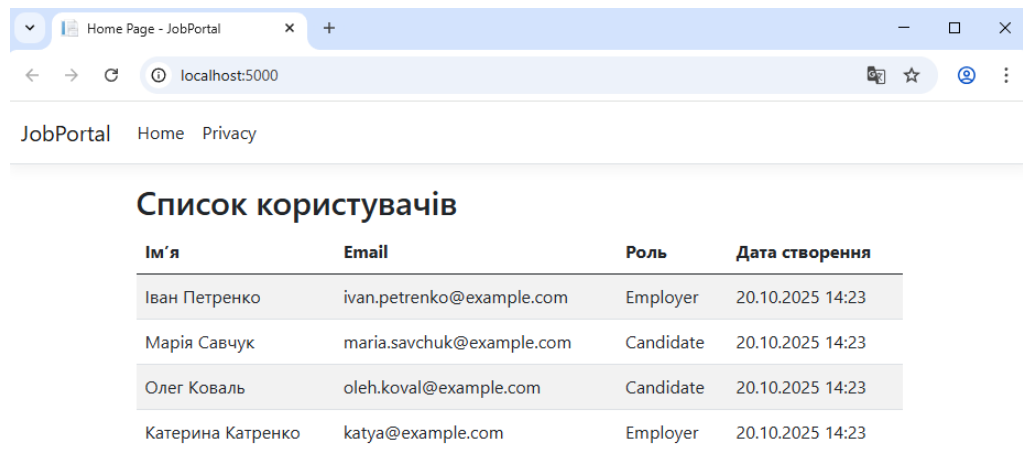
```

```

<tbody>
    @foreach (var user in Model ??
        Enumerable.Empty<JobPortal.Models.User>())
    {
        <tr>
            <td>@user.FullName</td>
            <td>@user.Email</td>
            <td>@user.Role</td>
            <td>@user.CreatedAt.ToLocalTime().ToString("dd.MM.yyyy
                HH:mm")</td>
        </tr>
    }
</tbody>
</table>

```

Після запуску проекту отримаємо результат зображений на рисунку 3.1.



Ім'я	Email	Роль	Дата створення
Іван Петренко	ivan.petrenko@example.com	Employer	20.10.2025 14:23
Марія Савчук	maria.savchuk@example.com	Candidate	20.10.2025 14:23
Олег Коваль	oleh.koval@example.com	Candidate	20.10.2025 14:23
Катерина Катренко	katya@example.com	Employer	20.10.2025 14:23

Рис. 3.1. Результат виведення даних на сторінці

Завдання 4. Реалізуйте посторінковий вивід записів.

1. У папці Models створіть підпапку ViewModels. View Model – це сутність, створена спеціально для передачі даних між контролером та представленням, яка створюється під час роботи додатку і не зберігається в БД.
2. В папці ViewModels створіть клас PagingInfo для збереження даних про пейджинг (кількість записів на сторінці, загальна кількість записів, кількість сторінок тощо).

Лістинг:

```

namespace JobPortal.Views.ViewModels
{
    public class PagingInfo
    {
        public int TotalItems { get; set; }
        public int ItemsPerPage { get; set; }
        public int CurrentPage { get; set; }

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

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.2	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		7

```
    }
}
```

3. В папці ViewModels створіть View Model для посторінкового відображення списку сутностей. Вона повинна включати в собі колекцію сутностей та інформацію про пейджинг.

Лістинг:

```
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();
    }
}
```

4. Додайте підтримку пейджингу в контролер. Для цього змініть метод Index таким чином, щоб він приймав номер поточної сторінки та реалізуйте в ньому запит до БД для отримання елементів саме для цієї сторінки. У представлення передавайте екземпляр створеної View Model на кроці номер 3.

Лістинг:

```
using Microsoft.AspNetCore.Mvc;
using JobPortal.Models;
using JobPortal.Models.ViewModels;
using JobPortal.Views.ViewModels;

namespace JobPortal.Controllers
{
    public class HomeController : Controller
    {
        private readonly PortalDbContext _context;
        private const int PageSize = 3; // кількість користувачів на сторінці

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

        public IActionResult Index(int page = 1)
        {
            var usersQuery = _context.Users.OrderBy(u => u.Id);

            var model = new UsersListViewModel
            {
                Users = usersQuery
                    .Skip((page - 1) * PageSize)
                    .Take(PageSize)
                    .ToList(),
            }
        }
    }
}
```



```

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

    return View(model);
}
}
}
}

```

5. Адаптуйте представлення для відображення списку продуктів з View Model.

Лістинг:

```

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

@model JobPortal.Models.ViewModels.UsersListViewModel

<h2>Список користувачів</h2>

<table class="table table-striped">
    <thead>
        <tr>
            <th>Ім'я</th>
            <th>Email</th>
            <th>Роль</th>
            <th>Дата створення</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var user in Model.Users)
        {
            <tr>
                <td>@user.FullName</td>
                <td>@user.Email</td>
                <td>@user.Role</td>
                <td>@user.CreatedAt.ToLocalTime().ToString("dd.MM.yyyy
HH:mm")</td>
            </tr>
        }
    </tbody>
</table>

<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">@i</a>
            </li>
        }
    </ul>
</nav>

```

Результат роботи програми:



1. Встановіть фреймворк Bootstrap, виконавши команди в папці проекту:

libman init -p cdnjs

2. Змініть базовий шаблон сторінки (`_Layout.cshtml`). Додайте елементи для хедера та футера.

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0"
  />
  <title>@ ViewData["Title"] - JobPortal</title>

  <link rel="stylesheet"
    href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
  <link rel="stylesheet" href="~/css/site.css" asp-append-version="true"
  />
  <link rel="stylesheet" href="~/JobPortal.styles.css" asp-append-
    version="true" />
```

```

</head>
<body class="d-flex flex-column min-vh-100 bg-light">

    <header>
        <nav class="navbar navbar-expand-sm navbar-dark bg-primary shadow-sm">
            <div class="container">
                <a class="navbar-brand fw-bold text-white" asp-area="" asp-controller="Home" asp-action="Index">JobPortal</a>
                <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="collapse navbar-collapse" id="navbarNav">
                    <ul class="navbar-nav ms-auto">
                        <li class="nav-item">
                            <a class="nav-link text-white" asp-controller="Home" asp-action="Index">Головна</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-white" asp-controller="Home" asp-action="Privacy">Політика</a>
                        </li>
                    </ul>
                </div>
            </div>
        </nav>
    </header>

    <main role="main" class="flex-grow-1 py-4">
        <div class="container">
            @RenderBody()
        </div>
    </main>

    <footer class="bg-primary text-white mt-auto py-3">
        <div class="container text-center small">
            &copy; 2025 - JobPortal | <a class="text-white text-decoration-underline" asp-controller="Home" asp-action="Privacy">Політика</a>
        </div>
    </footer>

    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    @await RenderSectionAsync("Scripts", required: false)
</body>
</html>

```

3. Стилiзуйте список сутностей та пейджинг.

Лiстинг:

```

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

@model JobPortal.Models.ViewModels.UsersListViewModel

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

<table class="table table-striped table-bordered">
    <thead class="table-primary">
        <tr>
            <th>Им'я</th>

```

					ЖИТОМИРСЬКА ПОЛІТЕХНІКА.25.121.23.000 – Лр.2	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		11

```

        <th>Email</th>
        <th>Роль</th>
        <th>Дата створення</th>
    </tr>
</thead>
<tbody>
    @foreach (var user in Model.Users)
    {
        <tr>
            <td>@user.FullName</td>
            <td>@user.Email</td>
            <td>@user.Role</td>
            <td>@user.CreatedAt.ToLocalTime().ToString("dd.MM.yyyy
HH:mm")</td>
        </tr>
    }
</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>

```

Результат:

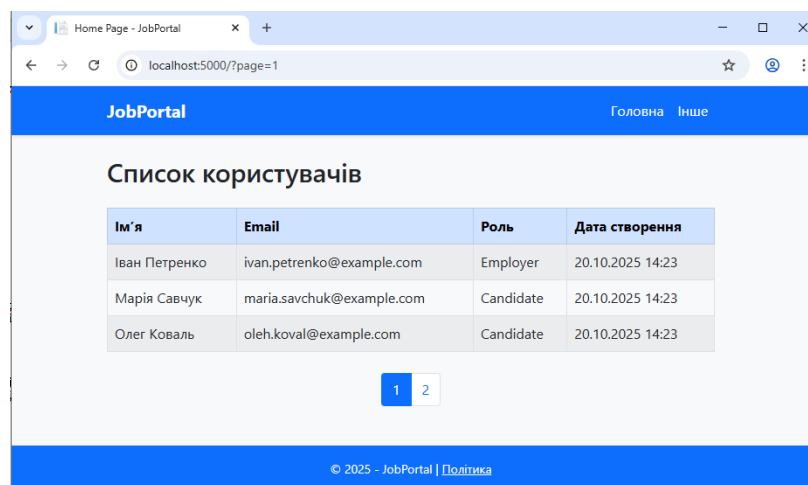


Рис. 5.1. Результат стилізації сторінок за допомогою bootstrap

Висновок: в ході виконання лабораторної роботи було ознайомлено з архітектурою MVC, набуто навичок створення головних компонентів архітектури MVC, навчено конфігурувати підключення до БД та виконувати запити на читання даних, набуто навичок роботи з представленнями (Views). Роботу виконано в повному обсязі.