

ДОДАТОК А

Ілюстрації програмного застосунку

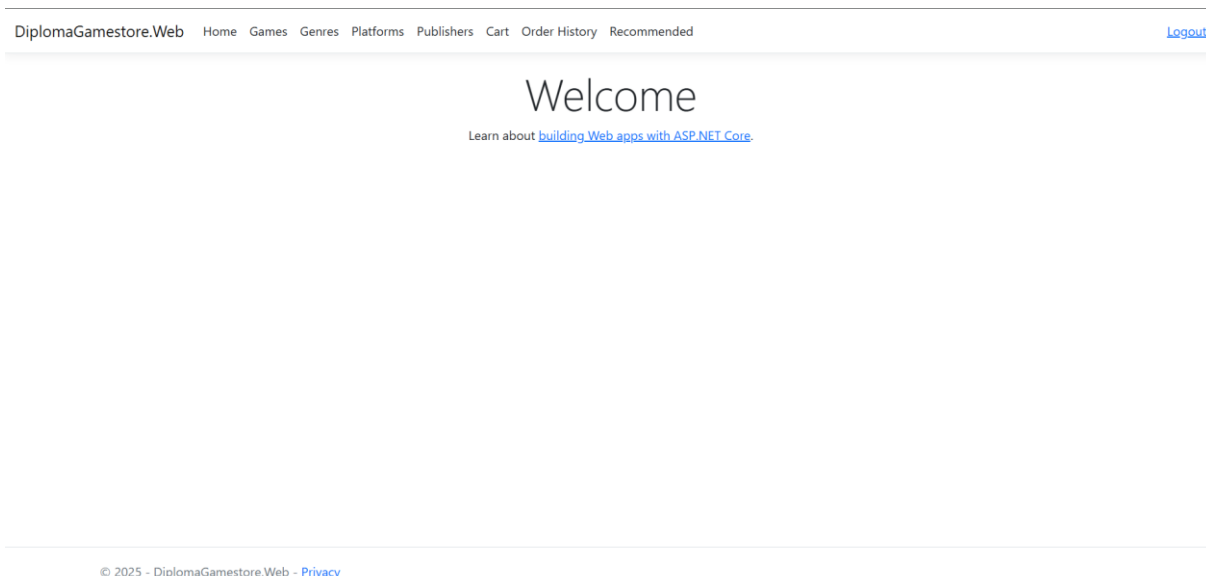


Рисунок А.1 – Головна сторінка

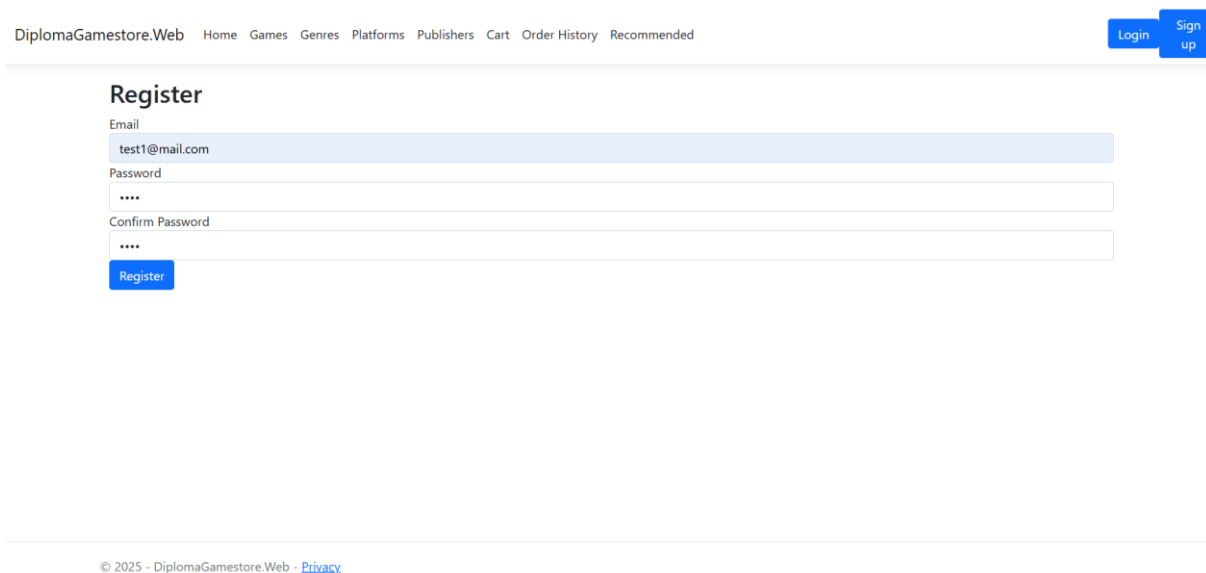


Рисунок А.2 – Сторінка реєстрації

Login

Email
test@mail.com

Password
....

Remember me?

Login [Sign up](#)

Рисунок А.3 – Сторінка логіну

Genres

[Add Genre](#)

Name	Actions
Action	Details Edit Delete
Adventure	Details Edit Delete
Battle Royale	Details Edit Delete
Card	Details Edit Delete
Co-op	Details Edit Delete
Fighting	Details Edit Delete
Hack and Slash	Details Edit Delete
Horror	Details Edit Delete
MMO	Details Edit Delete
MOBA	Details Edit Delete
Mystery	Details Edit Delete

Рисунок А.4 – Сторінка перегляду жанрів

Publishers

[Add Publisher](#)

Company Name	Description	Home Page	Actions
Bungie			Details Edit Delete
Paradox Interactive			Details Edit Delete
Digital Extremes			Details Edit Delete
Frontier Developments			Details Edit Delete
CD Projekt			Details Edit Delete
Riot Games			Details Edit Delete
Studio Wildcard			Details Edit Delete
Campo Santo			Details Edit Delete
Ludeon Studios			Details Edit Delete
Facepunch Studios			Details Edit Delete
Rockstar Games			Details Edit Delete

Рисунок А.5 – Сторінка перегляду видавців

Platforms

[Add Platform](#)

Type	Actions
Mobile	Details Edit Delete
PC	Details Edit Delete
PS3	Details Edit Delete
PS4	Details Edit Delete
PS5	Details Edit Delete
Switch	Details Edit Delete
VR	Details Edit Delete
Xbox 360	Details Edit Delete
Xbox One	Details Edit Delete
Xbox Series X/S	Details Edit Delete

Рисунок А.6 – Сторінка перегляду платформ

Games

[Add Game](#)

Name	Description	Price	Discount	Publisher	Genres	Platforms	Avg. Rating	View Count	Date Created	Actions
The Witcher 3: Wild Hunt		50	39 %	CD Projekt	Action, RPG	PS4, Xbox One, PC, Switch	3.3	135	4/8/2016	Details Edit Delete Add to Cart
Grand Theft Auto V		20	38 %	Rockstar Games	Action, Adventure	PS5, PS4, Xbox One, PC, Xbox Series X/S	0.0	522	5/13/2017	Details Edit Delete Add to Cart
Red Dead Redemption 2		20	19 %	Rockstar Games	Action, Adventure	PS4, Xbox One, PC	3.0	204	11/7/2021	Details Edit Delete Add to Cart
Cyberpunk 2077		40	19 %	CD Projekt	Action, RPG	PS5, PS4, Xbox One, PC, Xbox Series X/S	3.0	298	11/18/2017	Details Edit Delete Add to Cart
God of War (2018)		41	43 %	Sony Interactive	Action, Adventure	PS4, PC	5.0	580	4/27/2015	Details Edit Delete

Рисунок А.7 – Сторінка перегляду ігор

Your Cart

Game	Price	Subtotal	
Marvel's Spider-Man	18	18	Remove
Halo Infinite	6	6	Remove
Fallout 4	9	9	Remove
		Total:	33

[Checkout](#)[Continue Shopping](#)

Рисунок А.8 – Сторінка перегляду корзини

Order History

Order Date: 6/23/2025 Status: Completed				Total: 60
Game	Price	Your Rating	Action	
Bloodborne	12	Not rated	Rate	
Halo Infinite	6	Your rating: 4	Update Rating	
StarCraft II	42	Not rated	Rate	

Order Date: 6/23/2025 Status: Completed				Total: 30
Game	Price	Your Rating	Action	
The Witcher 3: Wild Hunt	30	Your rating: 4	Update Rating	

[Back to Cart](#)

Рисунок А.9 – Сторінка перегляду історії замовлень

Recommended Games

Name	Description	Price	Discount	Publisher	Genres	Platforms	Avg. Rating	View Count	Date Created	Actions
Diablo IV		56	22 %	Blizzard Entertainment	Action, RPG	PS5, PS4, Xbox One, PC, Xbox Series X/S	3.0	515	1/18/2021	Details Add to Cart
Cyberpunk 2077		40	19 %	CD Projekt	Action, RPG	PS5, PS4, Xbox One, PC, Xbox Series X/S	3.0	298	11/18/2017	Details Add to Cart
Overwatch 2		23	75 %	Blizzard Entertainment	Action, Shooter	PS5, PS4, Xbox One, PC, Xbox Series X/S, Switch	0.0	383	4/28/2020	Details Add to Cart
Fallout 4		24	59 %	Bethesda Softworks	Action, RPG	PS4, Xbox One, PC	4.0	692	5/27/2023	Details Add to Cart
Elden Ring		13	8 %	Bandai Namco	Action, RPG	PS5, PS4, Xbox One, PC, Xbox Series X/S	4.0	457	1/13/2022	Details Add to Cart

Previous [1](#) Next

Рисунок А.10 – Сторінка рекомендованих ігор

Game Details

The Witcher 3: Wild Hunt

Price: 50

Discount: 39 %

Publisher: CD Projekt

Genres: Action, RPG

Platforms: PS4, Xbox One, PC, Switch

Average Rating: 3.3

View Count: 136

Date Created: 4/8/2016

[Edit](#)

[Back](#)

Рисунок А.11 – Сторінка перегляду деталей гри

Create Game

Name

Description

Price

Discount

Publisher

- | | | |
|---|--|---|
| <input type="radio"/> Bungie | <input type="radio"/> Paradox Interactive | <input type="radio"/> Digital Extremes |
| <input type="radio"/> Frontier Developments | <input type="radio"/> CD Projekt | <input type="radio"/> Riot Games |
| <input type="radio"/> Studio Wildcard | <input type="radio"/> Campo Santo | <input type="radio"/> Ludeon Studios |
| <input type="radio"/> Facepunch Studios | <input type="radio"/> Rockstar Games | <input type="radio"/> Valve |
| <input type="radio"/> Playdead | <input type="radio"/> Wube Software | <input type="radio"/> Coffee Stain Publishing |
| <input type="radio"/> Sony Interactive | <input type="radio"/> Bethesda Softworks | <input type="radio"/> Kinetic Games |
| <input type="radio"/> Nintendo | <input type="radio"/> Mojang | <input type="radio"/> Electronic Arts |
| <input type="radio"/> Bandai Namco | <input type="radio"/> Re-Logic | <input type="radio"/> Behaviour Interactive |
| <input type="radio"/> 2K Games | <input type="radio"/> Activision | <input type="radio"/> ConcernedApe |
| <input type="radio"/> Innersloth | <input type="radio"/> 2K Sports | <input type="radio"/> 505 Games |
| <input type="radio"/> Supergiant Games | <input type="radio"/> Epic Games | <input type="radio"/> Psyonix |
| <input type="radio"/> Ubisoft | <input type="radio"/> Matt Makes Games | <input type="radio"/> Krafton |
| <input type="radio"/> Sega | <input type="radio"/> Blizzard Entertainment | <input type="radio"/> Team Cherry |
| <input type="radio"/> Studio MDHR | <input type="radio"/> Xbox Game Studios | <input type="radio"/> Motion Twin |
| <input type="radio"/> Humble Games | <input type="radio"/> Unknown Worlds | <input type="radio"/> Capcom |
| <input type="radio"/> Hello Games | <input type="radio"/> Warner Bros. Interactive | |

Genres

- | | | |
|---|------------------------------------|--|
| <input type="checkbox"/> Action | <input type="checkbox"/> Adventure | <input type="checkbox"/> Battle Royale |
| <input type="checkbox"/> Card | <input type="checkbox"/> Co-op | <input type="checkbox"/> Fighting |
| <input type="checkbox"/> Hack and Slash | <input type="checkbox"/> Horror | <input type="checkbox"/> MMO |
| <input type="checkbox"/> MOBA | <input type="checkbox"/> Mystery | <input type="checkbox"/> Party |
| <input type="checkbox"/> Platformer | <input type="checkbox"/> Puzzle | <input type="checkbox"/> Racing |
| <input type="checkbox"/> Roguelike | <input type="checkbox"/> RPG | <input type="checkbox"/> RTS |
| <input type="checkbox"/> Sandbox | <input type="checkbox"/> Shooter | <input type="checkbox"/> Simulation |
| <input type="checkbox"/> Social Deduction | <input type="checkbox"/> Sports | <input type="checkbox"/> Stealth |
| <input type="checkbox"/> Strategy | <input type="checkbox"/> Survival | <input type="checkbox"/> Tactical |
| <input type="checkbox"/> Turn-Based | | |

Platforms

- | | | |
|--|-----------------------------------|-----------------------------------|
| <input type="checkbox"/> Mobile | <input type="checkbox"/> PC | <input type="checkbox"/> PS3 |
| <input type="checkbox"/> PS4 | <input type="checkbox"/> PS5 | <input type="checkbox"/> Switch |
| <input type="checkbox"/> VR | <input type="checkbox"/> Xbox 360 | <input type="checkbox"/> Xbox One |
| <input type="checkbox"/> Xbox Series X/S | | |

Edit Game

Name

Description

Price

Discount

Publisher

- | | | |
|---|--|---|
| <input type="radio"/> Bungie | <input type="radio"/> Paradox Interactive | <input type="radio"/> Digital Extremes |
| <input type="radio"/> Frontier Developments | <input checked="" type="radio"/> CD Projekt | <input type="radio"/> Riot Games |
| <input type="radio"/> Studio Wildcard | <input type="radio"/> Campo Santo | <input type="radio"/> Ludeon Studios |
| <input type="radio"/> Facepunch Studios | <input type="radio"/> Rockstar Games | <input type="radio"/> Valve |
| <input type="radio"/> Playdead | <input type="radio"/> Wube Software | <input type="radio"/> Coffee Stain Publishing |
| <input type="radio"/> Sony Interactive | <input type="radio"/> Bethesda Softworks | <input type="radio"/> Kinetic Games |
| <input type="radio"/> Nintendo | <input type="radio"/> Mojang | <input type="radio"/> Electronic Arts |
| <input type="radio"/> Bandai Namco | <input type="radio"/> Re-Logic | <input type="radio"/> Behaviour Interactive |
| <input type="radio"/> 2K Games | <input type="radio"/> Activision | <input type="radio"/> ConcernedApe |
| <input type="radio"/> Innersloth | <input type="radio"/> 2K Sports | <input type="radio"/> 505 Games |
| <input type="radio"/> Supergiant Games | <input type="radio"/> Epic Games | <input type="radio"/> Psyonix |
| <input type="radio"/> Ubisoft | <input type="radio"/> Matt Makes Games | <input type="radio"/> Krafton |
| <input type="radio"/> Sega | <input type="radio"/> Blizzard Entertainment | <input type="radio"/> Team Cherry |
| <input type="radio"/> Studio MDHR | <input type="radio"/> Xbox Game Studios | <input type="radio"/> Motion Twin |
| <input type="radio"/> Humble Games | <input type="radio"/> Unknown Worlds | <input type="radio"/> Capcom |
| <input type="radio"/> Hello Games | <input type="radio"/> Warner Bros. Interactive | |

Genres

- | | | |
|--|---|--|
| <input checked="" type="checkbox"/> Action | <input type="checkbox"/> Adventure | <input type="checkbox"/> Battle Royale |
| <input type="checkbox"/> Card | <input type="checkbox"/> Co-op | <input type="checkbox"/> Fighting |
| <input type="checkbox"/> Hack and Slash | <input type="checkbox"/> Horror | <input type="checkbox"/> MMO |
| <input type="checkbox"/> MOBA | <input type="checkbox"/> Mystery | <input type="checkbox"/> Party |
| <input type="checkbox"/> Platformer | <input type="checkbox"/> Puzzle | <input type="checkbox"/> Racing |
| <input type="checkbox"/> Roguelike | <input checked="" type="checkbox"/> RPG | <input type="checkbox"/> RTS |
| <input type="checkbox"/> Sandbox | <input type="checkbox"/> Shooter | <input type="checkbox"/> Simulation |
| <input type="checkbox"/> Social Deduction | <input type="checkbox"/> Sports | <input type="checkbox"/> Stealth |
| <input type="checkbox"/> Strategy | <input type="checkbox"/> Survival | <input type="checkbox"/> Tactical |
| <input type="checkbox"/> Turn-Based | | |

Platforms

- | | | |
|--|--|--|
| <input type="checkbox"/> Mobile | <input checked="" type="checkbox"/> PC | <input type="checkbox"/> PS3 |
| <input checked="" type="checkbox"/> PS4 | <input type="checkbox"/> PS5 | <input checked="" type="checkbox"/> Switch |
| <input type="checkbox"/> VR | <input type="checkbox"/> Xbox 360 | <input checked="" type="checkbox"/> Xbox One |
| <input type="checkbox"/> Xbox Series X/S | | |

Рисунок А.13 – Сторінка редагування гри

Create Genre

Name

Рисунок А.14 – Сторінка створення жанру

Create Publisher

CompanyName

Description

HomePage

Рисунок А.15 – Сторінка створення видавця

Create Platform

Type

Рисунок А.16 – Сторінка створення платформи

ДОДАТОК Б

Приклади програмної реалізації

Лістинг Б.1 – Реалізація класу `DiplomaGamestore.BLL.Configure`

```
using DiplomaGamestore.BLL.AzureML;
using DiplomaGamestore.BLL.Mappings;
using DiplomaGamestore.BLL.Services;
using DiplomaGamestore.BLL.Services.Abstracts;
using Microsoft.Extensions.DependencyInjection;
namespace DiplomaGamestore.BLL;
public static class Configure
{
    public static void
ConfigureBusinessLogicLayerServices(this
IServiceCollection services)
    {
        services.AddAutoMapper(typeof(MappingProfile));

        services.AddHttpClient<AzureMLClient>();
        services.AddScoped<IRecommendationService,
AzureMlRecommendationService>();

        services.AddScoped<IGameService, GameService>();
        services.AddScoped<IGenreService,
GenreService>();
        services.AddScoped<IPlatformService,
PlatformService>();
        services.AddScoped<IPublisherService,
PublisherService>();
        services.AddScoped<IRatingService,
RatingService>();
        services.AddScoped<IOrderService,
OrderService>();
    }
}
```

Лістинг Б.2 – Інтерфейс репозиторію

```
using System.Linq.Expressions;
using DiplomaGamestore.DAL.Entities.Abstracts;

namespace DiplomaGamestore.DAL.Repositories.Abstracts;

public interface IRepository<T>
    where T : Entity
{
    Task<ICollection<T>> GetAsync(
        Expression<Func<T, bool>>? predicate = null,
        IEnumerable<string>? includes = null,
        Func<IQueryable<T>, IOrderedQueryable<T>>?
orderBy = null,
        bool asNoTracking = false,
        int? pageNumber = null,
        int? pageSize = null);

    Task<T?> GetByIdAsync(
        Guid id,
        IEnumerable<string>? includes = null,
        bool asNoTracking = false);

    Task<T?> GetSingleAsync(
        Expression<Func<T, bool>>? predicate = null,
        IEnumerable<string>? includes = null,
        bool asNoTracking = false);
    Task<T> CreateAsync(T entity);
    Task<T?> DeleteAsync(Guid id);

    void Delete(T entity);

    T Update(T entity);
}
```

Лістинг Б.3 – Інтерфейс Unit of Work

```

using DiplomaGamestore.DAL.Entities;
using DiplomaGamestore.DAL.Repositories.Abstracts;

namespace DiplomaGamestore.DAL.UnitOfWorks.Abstracts;

public interface IUnitOfWork : IDisposable
{
    IRepository<Game> GameRepository { get; }
    IRepository<Genre> GenreRepository { get; }
    IRepository<Platform> PlatformRepository { get; }
    IRepository<Publisher> PublisherRepository { get; }
    IRepository<Rating> RatingRepository { get; }
    IRepository<Order> OrderRepository { get; }
    IRepository<OrderDetails> OrderDetailsRepository {
get; }
    IRepository<GameGenre> GameGenreRepository { get; }
    IRepository<GamePlatform> GamePlatformRepository {
get; }

    Task SaveAsync();
}

```

Лістинг Б.4 – Реалізація класу GamestoreDbContext

```

using DiplomaGamestore.DAL.Entities;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;

namespace DiplomaGamestore.DAL.Data;

public class GamestoreDbContext :
IdentityDbContext<IdentityUser>
{

```

```

public
GamestoreDbContext (DbContextOptions<GamestoreDbContext>
options) : base(options)
{
}

public DbSet<Game> Games { get; set; }
public DbSet<Genre> Genres { get; set; }
public DbSet<Platform> Platforms { get; set; }
public DbSet<Publisher> Publishers { get; set; }
public DbSet<Rating> Ratings { get; set; }
public DbSet<Order> Orders { get; set; }
public DbSet<OrderDetails> OrderDetails { get; set; }
public DbSet<GameGenre> GameGenres { get; set; }
public DbSet<GamePlatform> GamePlatforms { get; set; }

protected override void OnModelCreating(ModelBuilder
builder)
{
    base.OnModelCreating(builder);

    // Genre
    builder.Entity<Genre>().HasIndex(e => e.Name,
"UX_Genre_Name").IsUnique();

    // Platform
    builder.Entity<Platform>().HasIndex(e => e.Type,
"UX_Platform_Type").IsUnique();

    // Publisher
    builder.Entity<Publisher>().HasIndex(e =>
e.CompanyName, "UX_Publisher_CompanyName").IsUnique();

    // GameGenre
    builder.Entity<GameGenre>()
        .HasIndex(

```

```

        e => new
        {
            e.GameId, e.GenreId
        },
        "UX_GameGenre_GenreId_GameId");

// GameGenre
builder.Entity<GamePlatform>()
    .HasIndex(
        e => new
        {
            e.GameId, e.PlatformId
        },
        "UX_GameGenre_PlatformId_GameId");

// One to Many (Genre - Publisher)
builder.Entity<Game>()
    .HasOne(g => g.Publisher)
    .WithMany(pg => pg.Games)
    .HasForeignKey(g => g.PublisherId)
    .onDelete(DeleteBehavior.Cascade);

// Many to Many (Game - Genre)
builder.Entity<GameGenre>()
    .HasOne(gg => gg.Game)
    .WithMany(g => g.GameGenres)
    .HasForeignKey(gg => gg.GameId)
    .onDelete(DeleteBehavior.Cascade);

builder.Entity<GameGenre>()
    .HasOne(gg => gg.Genre)
    .WithMany(g => g.GameGenres)
    .HasForeignKey(gg => gg.GenreId)
    .onDelete(DeleteBehavior.Cascade);

// Many to Many (Game - Platform)

```

```

builder.Entity<GamePlatform>()
    .HasOne(gp => gp.Game)
    .WithMany(g => g.GamePlatforms)
    .HasForeignKey(gp => gp.GameId)
    .OnDelete(DeleteBehavior.Cascade);

builder.Entity<GamePlatform>()
    .HasOne(gp => gp.Platform)
    .WithMany(p => p.GamePlatforms)
    .HasForeignKey(gp => gp.PlatformId)
    .OnDelete(DeleteBehavior.Cascade);

// Many to Many (Game - Order)
builder.Entity<OrderDetails>()
    .HasOne(od => od.Game)
    .WithMany(g => g.OrderDetails)
    .HasForeignKey(od => od.GameId)
    .OnDelete(DeleteBehavior.Cascade);

builder.Entity<OrderDetails>()
    .HasOne(od => od.Order)
    .WithMany(o => o.OrderDetails)
    .HasForeignKey(od => od.OrderId)
    .OnDelete(DeleteBehavior.Cascade);
    }
}

```

Лістинг Б.5 – Реалізація класу `GameService`

```

public class GameService : IGameService
{
    private readonly IUnitOfWork _unitOfWork;
    private readonly IMapper _mapper;

    public GameService(IUnitOfWork unitOfWork, IMapper
mapper)

```

```

    {
        _unitOfWork = unitOfWork;
        _mapper = mapper;
    }

    public async Task<GameDto?> GetByIdAsync(Guid id)
    {
        var game = await
        _unitOfWork.GameRepository.GetByIdAsync(id, new[] {
            "Publisher", "GameGenres.Genre", "GamePlatforms.Platform",
            "Ratings" });
        return game == null ? null :
        _mapper.Map<GameDto>(game);
    }

    public async Task<ICollection<GameDto>> GetAsync()
    {
        var games = await
        _unitOfWork.GameRepository.GetAsync(null, new[] { "Publisher",
            "GameGenres.Genre", "GamePlatforms.Platform", "Ratings" });
        return
        games.Select(_mapper.Map<GameDto>).ToList();
    }

    public async Task<GameDto> CreateAsync(GameDto
gameDto)
    {
        var game = _mapper.Map<Game>(gameDto);
        await
        _unitOfWork.GameRepository.CreateAsync(game);
        await _unitOfWork.SaveAsync();
        return _mapper.Map<GameDto>(game);
    }

    public async Task<GameDto?> UpdateAsync(GameDto
gameDto)

```

```

    {
        var game = _mapper.Map<Game>(gameDto);
        _unitOfWork.GameRepository.Update(game);
        await _unitOfWork.SaveAsync();
        return _mapper.Map<GameDto>(game);
    }

    public async Task<bool> DeleteAsync(Guid id)
    {
        var result = await
        _unitOfWork.GameRepository.DeleteAsync(id);
        await _unitOfWork.SaveAsync();
        return result != null;
    }
}

```

Лістинг Б.6 – Реалізація класу AzureMLClient

```

public class AzureMLClient
{
    private readonly HttpClient _httpClient;
    private readonly string _endpoint;
    private readonly string _apiKey;

    public AzureMLClient(IConfiguration configuration,
        HttpClient httpClient)
    {
        _httpClient = httpClient;
        _endpoint = configuration["AzureML:Endpoint"]!;
        _apiKey = configuration["AzureML:ApiKey"]!;
        _httpClient.DefaultRequestHeaders.Authorization =
        new AuthenticationHeaderValue("Bearer", _apiKey);
    }

    public async Task<List<Guid>>
    GetRecommendedGameIdsAsync(string userId, int count = 10)

```

```

        {
            var requestPayload = new { UserId = userId,
RecommendationCount = count };
            var content = new
StringContent(JsonSerializer.Serialize(requestPayload),
Encoding.UTF8, "application/json");
            var response = await
_httpClient.PostAsync(_endpoint, content);

            if (!response.IsSuccessStatusCode)
                throw new RequestFailedException($"Azure ML
request failed: {response.StatusCode}");

            var responseContent = await
response.Content.ReadAsStringAsync();
            var result =
JsonSerializer.Deserialize<AzureMLResponse>(responseContent);
            return result?.RecommendedGameIds ?? new
List<Guid>();
        }

        private class AzureMLResponse
        {
            public List<Guid> RecommendedGameIds { get; set; }
        }
    }
}

```

Лістинг Б.6 – Реалізація класу GameController

```

using AutoMapper;
using DiplomaGamestore.BLL.DTOs;
using DiplomaGamestore.BLL.Services.Abstracts;
using DiplomaGamestore.Web.Models;
using Microsoft.AspNetCore.Mvc;

namespace DiplomaGamestore.Web.Controllers;

```

```
public class GameController : Controller
{
    private readonly IGameService _gameService;
    private readonly IGenreService _genreService;
    private readonly IMapper _mapper;
    private readonly IPlatformService _platformService;
    private readonly IPublisherService _publisherService;

    public GameController(
        IGameService gameService,
        IGenreService genreService,
        IPlatformService platformService,
        IPublisherService publisherService,
        IMapper mapper)
    {
        _gameService = gameService;
        _genreService = genreService;
        _platformService = platformService;
        _publisherService = publisherService;
        _mapper = mapper;
    }

    public async Task<IActionResult> Index()
    {
        var games = await _gameService.GetAsync();
        var vms = _mapper.Map<List<GameViewModel>>(games);
        return View(vms);
    }

    public async Task<IActionResult> Details(Guid id)
    {
        var game = await _gameService.GetByIdAsync(id);
        if (game == null) return NotFound();
        var vm = _mapper.Map<GameViewModel>(game);
        return View(vm);
    }
}
```

```

public async Task<IActionResult> Create()
{
    var vm = new GameCreateEditViewModel();
    await PopulateOptions(vm);
    return View(vm);
}

[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult>
Create(GameCreateEditViewModel vm)
{
    if (!ModelState.IsValid)
    {
        await PopulateOptions(vm);
        return View(vm);
    }

    var dto = _mapper.Map<GameDto>(vm);
    dto.PublisherId = vm.PublisherId;
    dto.Genres = vm.SelectedGenreIds.Select(id => new
GenreDto { Id = id }).ToList();
    dto.Platforms = vm.SelectedPlatformIds.Select(id
=> new PlatformDto { Id = id }).ToList();
    await _gameService.CreateAsync(dto);
    return RedirectToAction(nameof(Index));
}

public async Task<IActionResult> Edit(Guid id)
{
    var game = await _gameService.GetByIdAsync(id);
    if (game == null) return NotFound();
    var vm =
_mapper.Map<GameCreateEditViewModel>(game);
    await PopulateOptions(vm);
}

```

```

        return View(vm);
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> Edit(Guid id,
GameCreateEditViewModel vm)
    {
        if (id != vm.Id) return BadRequest();
        if (!ModelState.IsValid)
        {
            await PopulateOptions(vm);
            return View(vm);
        }

        var dto = _mapper.Map<GameDto>(vm);
        dto.PublisherId = vm.PublisherId;
        dto.Genres = vm.SelectedGenreIds.Select(gid => new
GenreDto { Id = gid }).ToList();
        dto.Platforms = vm.SelectedPlatformIds.Select(pid
=> new PlatformDto { Id = pid }).ToList();
        await _gameService.UpdateAsync(dto);
        return RedirectToAction(nameof(Index));
    }

    public async Task<IActionResult> Delete(Guid id)
    {
        var game = await _gameService.GetByIdAsync(id);
        if (game == null) return NotFound();
        var vm = _mapper.Map<GameViewModel>(game);
        return View(vm);
    }

    [HttpPost]
    [ActionName("Delete")]
    [ValidateAntiForgeryToken]

```

```

        public async Task<IActionResult> DeleteConfirmed(Guid
id)
        {
            await _gameService.DeleteAsync(id);
            return RedirectToAction(nameof(Index));
        }

        private async Task
PopulateOptions(GameCreateEditViewModel vm)
        {
            vm.AllPublishers = (await
_publisherService.GetAsync())
                .Select(p => new PublisherViewModel { Id =
p.Id, CompanyName = p.CompanyName }).ToList();
            vm.AllGenres = (await _genreService.GetAsync())
                .Select(g => new GenreViewModel { Id = g.Id,
Name = g.Name }).ToList();
            vm.AllPlatforms = (await
_platformService.GetAsync())
                .Select(p => new PlatformViewModel { Id =
p.Id, Type = p.Type }).ToList();
        }
    }
}

```

Лістинг Б.7 – Реалізація класу GameViewModel

```

using System.ComponentModel.DataAnnotations;

namespace DiplomaGamestore.Web.Models;

public class GameViewModel
{
    public Guid Id { get; set; }

    [Required] [StringLength(50)] public string Name {
get; set; }
}

```

```

public string? Description { get; set; }

[Range(0, int.MaxValue)] public int Price { get; set;
}

[Range(0, 100)] public int Discount { get; set; }

public long ViewCount { get; set; }
public DateTime DateCreated { get; set; }

public Guid PublisherId { get; set; }
public string PublisherName { get; set; }

public List<Guid> SelectedGenreIds { get; set; } =
new();
public List<Guid> SelectedPlatformIds { get; set; } =
new();

public List<string> GenreNames { get; set; } = new();
public List<string> PlatformTypes { get; set; } =
new();

public double AverageRating { get; set; }
}

```

Лістинг Б.8 – Реалізація класу GameCreateEditViewModel

```

using System.ComponentModel.DataAnnotations;

namespace DiplomaGamestore.Web.Models;

public class GameCreateEditViewModel
{
    public Guid Id { get; set; }
}

```

```
[Required] [StringLength(50)] public string Name {
get; set; }

public string? Description { get; set; }

[Range(0, int.MaxValue)] public int Price { get; set;
}

[Range(0, 100)] public int Discount { get; set; }

[Required] public Guid PublisherId { get; set; }

[MinLength(1, ErrorMessage = "At least one genre must
be selected.")]
public List<Guid> SelectedGenreIds { get; set; } =
new();

[MinLength(1, ErrorMessage = "At least one platform
must be selected.")]
public List<Guid> SelectedPlatformIds { get; set; } =
new();

// For rendering options
public List<PublisherViewModel> AllPublishers { get;
set; } = new();
public List<GenreViewModel> AllGenres { get; set; } =
new();
public List<PlatformViewModel> AllPlatforms { get;
set; } = new();
}
```

