Отчёт по лабораторной работе №4

Тема: «Веб-сервисы»

Выполнил

студент ИСТ-120

Голубев Д. О.

Проверил

Данилов В. В.

Содержание

[Цель работы 2](#_Toc101441361)

[Выполнение работы 3](#_Toc101441362)

[1 Создание API 3](#_Toc101441363)

[1.1 Создание модели 3](#_Toc101441364)

[1.2 Создание интерфейса DAO 3](#_Toc101441365)

[1.3 Создание конкретного DAO 3](#_Toc101441366)

[1.4 Конфигурирование проекта 4](#_Toc101441367)

[1.5 Создание контроллера 5](#_Toc101441368)

[2 Создание Web-приложения для работы с сервером 5](#_Toc101441369)

[2.1 Подключение HttpServer в Program.cs 5](#_Toc101441370)

[2.2 Создание клиентского хранилища 5](#_Toc101441371)

[2.3 Создание запросов к веб-серверу 6](#_Toc101441372)

[2.4 Запуск приложения 6](#_Toc101441373)

[Вывод 7](#_Toc101441374)

Цель работы

Познакомиться с технологией создания и использования веб-сервисов.

Выполнение работы

1 Создание API

1.1 Создание модели

namespace lw4API.Models  
{  
 public class Game  
 {  
 public Game(string title, string developer, string genre)  
 {  
 Title = title;  
 Developer = developer;  
 Genre = genre;  
 }  
  
 public string Title { get; set; }  
 public string Developer { get; set; }  
 public string Genre { get; set; }  
 }  
}

1.2 Создание интерфейса DAO

namespace lw4API.DAO  
{  
 public interface IDao<TEntity>  
 {  
 public void Create(TEntity entity);  
 public List<TEntity> GetAll();  
 }  
}

1.3 Создание конкретного DAO

namespace lw4API.DAO  
{  
 public class GameDao : IDao<Game>  
 {  
 private readonly Database \_database;  
 private readonly List<Game> \_games = new();  
  
 public GameDao(IConfiguration configuration)  
 {  
 \_database = new Database(configuration);  
 }  
  
 public void Create(Game game)  
 {  
 NpgsqlCommand cmd = new("INSERT INTO game (title, developer, genre, date) VALUES (" +  
 "@Title, @Developer, @Genre)");  
 cmd.Parameters.AddWithValue("@Title", game.Title);  
 cmd.Parameters.AddWithValue("@Developer", game.Developer);  
 cmd.Parameters.AddWithValue("@Genre", game.Genre);  
  
 \_database.ExecuteSQLRequest\_withParams(cmd);  
 }  
  
 public List<Game> GetAll()  
 {  
 string sql = "SELECT \* FROM game";  
 \_database.ExecuteSqlRequest(sql);  
 if (\_database.Reader is not null && \_database.Reader.HasRows)  
 {  
 foreach (DbDataRecord line in \_database.Reader)  
 {  
 Game game = new Game(line.GetString(0), line.GetString(1), line.GetString(2));  
 \_games.Insert(0, game);  
 }  
 }  
  
 return \_games;  
 }  
 }  
}

1.4 Конфигурирование проекта

1.4.1 Создание класса для работы с базой данных

namespace lw4API.Configuration  
{  
 public class Database  
 {  
 private readonly Server \_server;  
 public NpgsqlDataReader? Reader;  
  
 public Database(IConfiguration configuration)  
 {  
 \_server = new Server(configuration);  
 }  
  
 public void ExecuteSQLRequest\_withParams(NpgsqlCommand cmd)  
 {  
 cmd.Connection = \_server.Connection;  
 cmd.ExecuteNonQuery();  
 }  
  
 public void ExecuteSqlRequest(string sql)  
 {  
 NpgsqlCommand cmd = new NpgsqlCommand(sql, \_server.Connection);  
 Reader = cmd.ExecuteReader();  
 }  
 }  
  
 public class Server  
 {  
 public readonly NpgsqlConnection Connection;  
  
 public Server(IConfiguration configuration)  
 {  
 string conn = string.*Format*(configuration.GetConnectionString("Server"));  
 Connection = new NpgsqlConnection(conn);  
 Connection.Open();  
 }  
 }  
}

1.4.2 Создание файла конфигурации

{  
 "Logging": {  
 "LogLevel": {  
 "Default": "Information",  
 "Microsoft.AspNetCore": "Warning"  
 }  
 },  
 "AllowedHosts": "\*",  
 "ConnectionStrings": {  
 "Server": "Server=localhost;Port=5432;Database=forlabs;UserId=postgres;Password=1234"  
 }  
}

1.5 Создание контроллера

namespace lw4API.Controllers  
{  
 [ApiController]  
 [Route("/Game")]  
 public class GameController : ControllerBase  
 {  
 readonly GameDao \_gameDao = new(Configuration.Configuration.*config*);  
  
 [HttpGet]  
 public List<Game> GetGames()  
 {  
 return \_gameDao.GetAll();  
 }  
  
 [HttpPost]  
 public void CreateGame(Game game)  
 {  
 \_gameDao.Create(game);  
 }  
 }  
}

2 Создание Web-приложения для работы с сервером

2.1 Подключение HttpServer в Program.cs

builder.Services.AddSingleton<HttpClient>();

2.2 Создание клиентского хранилища

{  
 public static List<Game> *Games* = new();  
}

2.3 Создание запросов к веб-серверу

public class GameModel : PageModel  
{  
 private HttpClient? \_client;  
  
 public GameModel(HttpClient client)  
 {  
 \_client = client;  
 }  
  
 [BindProperty] public Game Game { get; set; }  
  
 public async void OnGet()  
 {  
 if (\_client is not null)  
 {  
 ViewStore.*Games* = await \_client.GetFromJsonAsync<List<Game>>("http://localhost:5124/Game");  
 }  
 }  
  
 public async void OnPostAsync()  
 {  
 if (!ModelState.IsValid)  
 {  
 return;  
 }  
  
 ViewStore.*Games*.Insert(0, Game);  
 if (\_client is not null)  
 {  
 HttpResponseMessage response = await \_client.PostAsJsonAsync("http://localhost:5124/Game", Game);  
 }  
 }  
  
 public List<Game> GetGames()  
 {  
 return ViewStore.*Games*;  
 }  
}

2.4 Запуск приложения

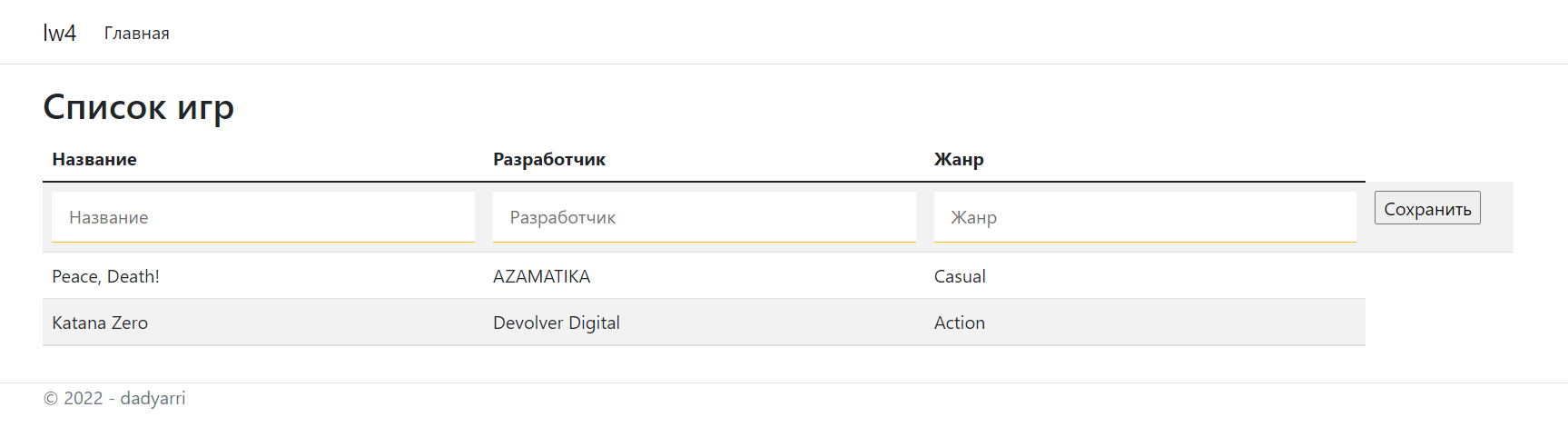


Рисунок 2.1 — Скриншот запущенного приложения

Вывод

В ходе выполнения лабораторной работы было произведено знакомство с технологией создания и использования веб-сервисов.