



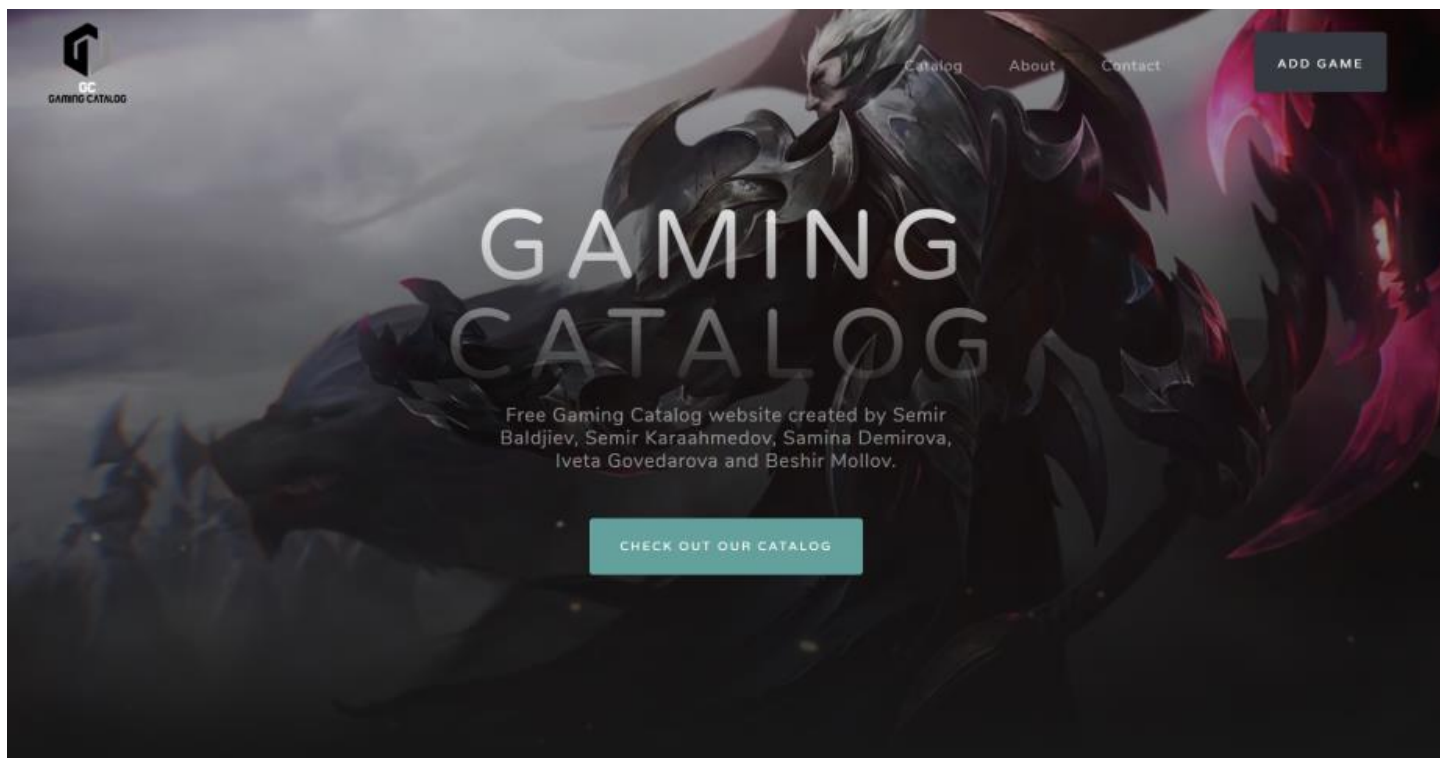
Каталог за игри

MVC уеб приложение

1. Въведение

Gaming Catalog – Web Application

Уеб приложението е направено на **ASP.Net Core MVC** технологията за създаване на **Web Applications**.

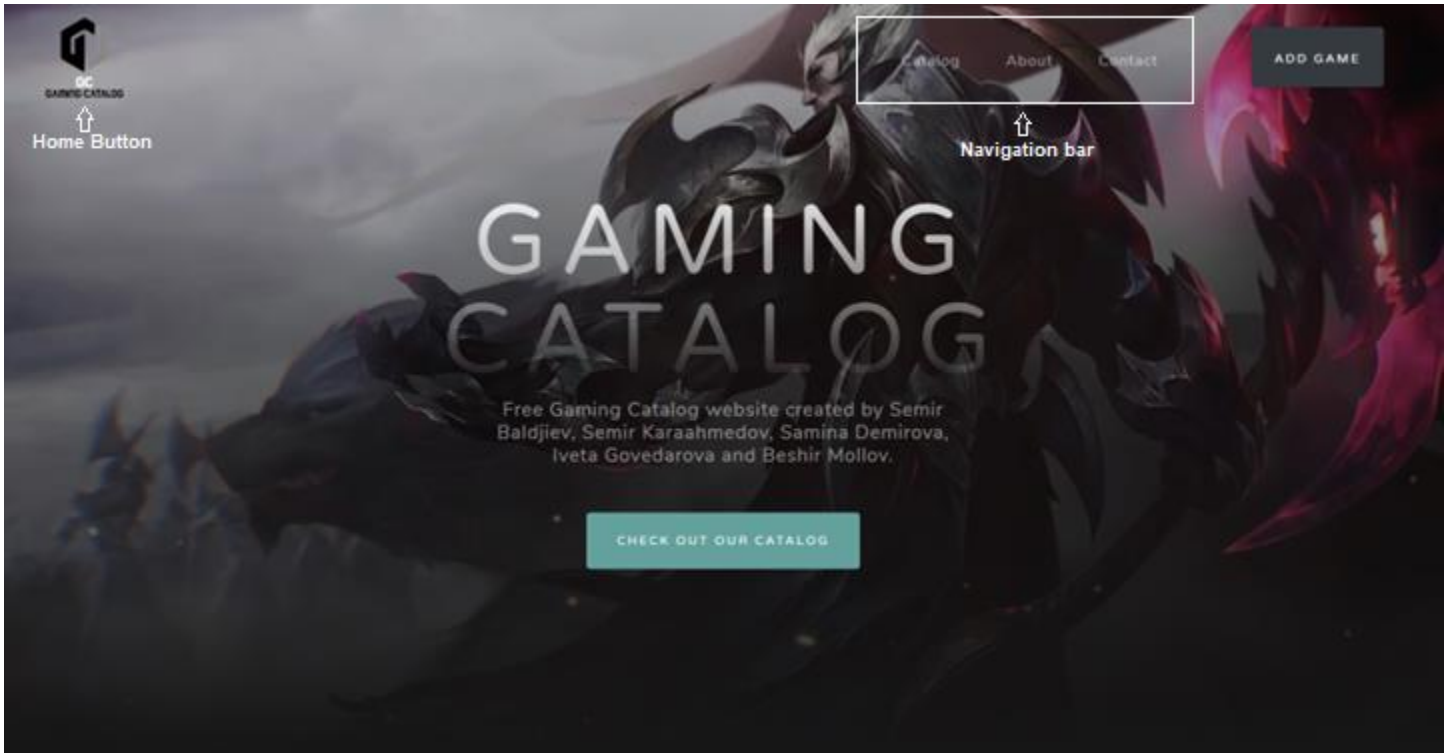


В това приложение ще се разгледат действия като създаване, четене, редактиране и премахване или още познати като (**CRUD**) операции на стойностите в каталога.

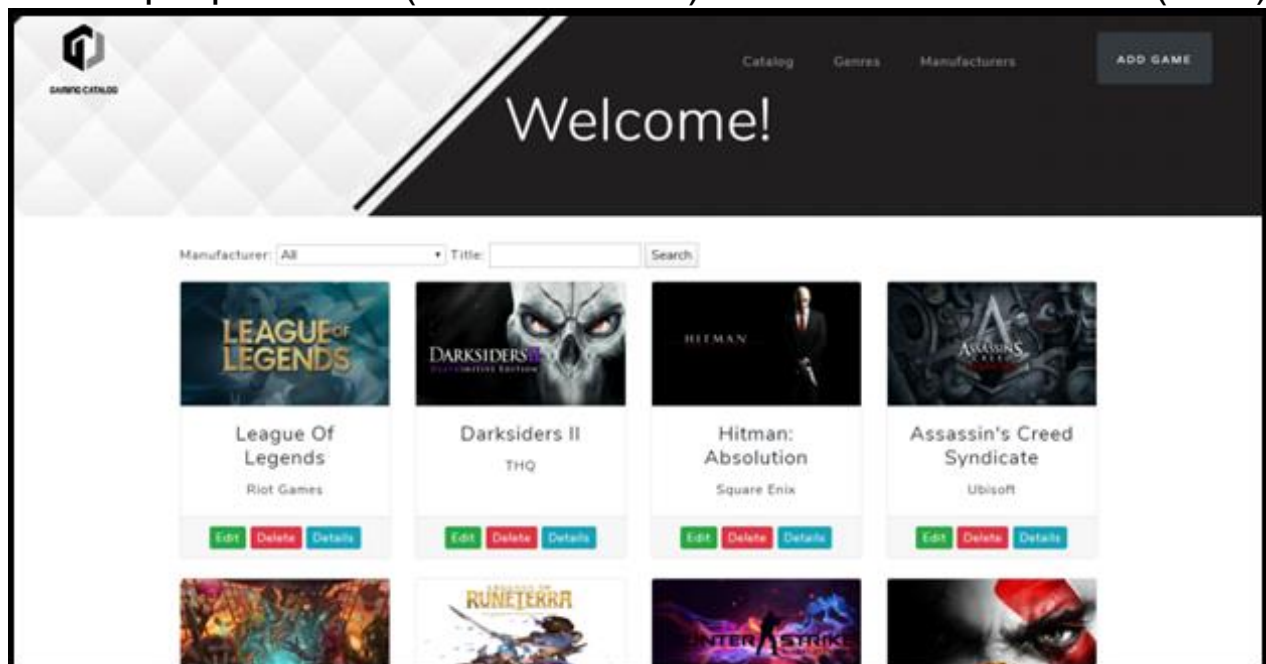
2. Преглед на уеб приложението

При стартиране на уеб приложението се отваря първоначалната страница или **Home Screen**–а от който се започва уебсайта.

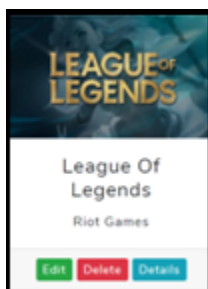
В горната страна на сайта са бутоните които ще използваме.



Така изглежда каталога, който има опцията да търси игри спрямо техния разработчик (**Manufacturer**) или тяхното заглавие (**Title**).



Всяка въведена игра в базата данни може да се редактира (**Edit**), изтрива (**Delete**) или да и се гледат детайлите (**Details**) въведени в избраната игра.




Edit бутона отваря прозорец за редактиране на данните в играта.

A screenshot of a web application's 'Edit Game' modal. The modal has a white background and a black border. In the top left corner is a logo for 'GAMING CATALOG'. In the top right corner are links for 'Catalog', 'Genres', and 'Manufacturers', and a dark grey 'Add Game' button. The main area contains several input fields: 'Title' (with 'League Of Legends' entered), 'Price' (with '0.00' entered), 'Rating' (with '8.6' entered), 'Release Date' (with '10/27/2009' entered), 'Platform' (with 'PC' selected), and 'Manufacturer' (with 'Riot Games' selected). There is a 'Description' text area with the text 'League of Legends (LoL) is a multiplayer online battle arena video game developed and published by Riot Games for Microsoft Windows and macOS.' Below this is an 'Image' section with a 'Choose File' button and the text 'No file chosen'. To the right of the image section is a 'Genres' dropdown menu with 'Hack and Slash', 'Action role-playing', 'MOBA', and 'Stealth' options. At the bottom center is a blue 'Save' button. In the bottom left corner is a blue link 'Back to List'.

Delete бутона отваря прозорец за изтриване на играта.


A screenshot of a web application's 'Delete Game' modal. The modal has a white background and a black border. In the top left corner is a logo for 'GAMING CATALOG'. In the top right corner are links for 'Catalog', 'Genres', and 'Manufacturers', and a dark grey 'Add Game' button. Below the header is a question: 'Are you sure you want to delete this game?'. Below this is a list of game details: 'Title' (League Of Legends), 'Price' (\$0.00), 'Rating' (8.6), 'Release Date' (10/27/2009), 'Platform' (PC), 'Description' (League of Legends (LoL) is a multiplayer online battle arena video game developed and published by Riot Games for Microsoft Windows and macOS. Inspired by the Warcraft III: The Frozen Throne mod Defense of the Ancients, the game follows a freemium model and is supported by microtransactions. In League of Legends, players assume the role of a "champion" with unique abilities and battle against a team of other player- or computer-controlled champions.), 'Image' (a small image of the League of Legends logo), 'Manufacturer' (Riot Games), and 'Genres' (MOBA). At the bottom left are two buttons: a red 'Delete' button and a blue 'Back to List' link.

A **Details** бутона отваря прозорец за преглеждане на данните въведени за конкретната игра.

 GAMING CATALOG

Catalog Genres Manufacturers Add Game

Details

Title	League Of Legends
Price	\$0.00
Rating	8.6
Release Date	10/27/2009
Platform	PC
Description	League of Legends (LoL) is a multiplayer online battle arena video game developed and published by Riot Games for Microsoft Windows and macOS. Inspired by the Warcraft III: The Frozen Throne mod Defense of the Ancients, the game follows a freemium model and is supported by microtransactions. In League of Legends, players assume the role of a "champion" with unique abilities and battle against a team of other player- or computer-controlled champions.
Image	
Manufacturer	Riot Games
Genres	MOBA

[Edit](#) | [Back to List](#)

За добавяне на жанрове или разработчици в базата данни използваме бутоните **Genres** и **Manufacturers** които ни водят до отделни прозорци при които да въвеждаме данните които искаме.

Genre

Catalog Genres Manufacturers Add Game

Name

Create

[Back to List](#)

Manufacturer

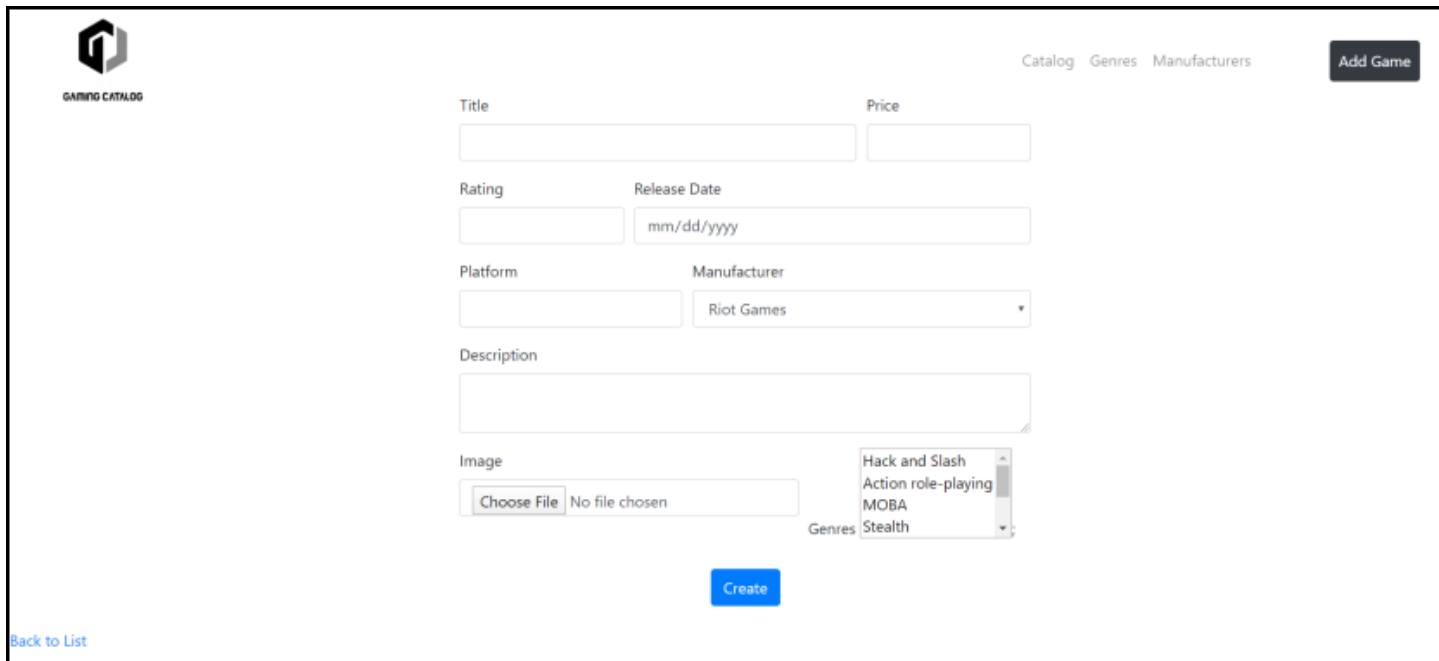
Catalog Genres Manufacturers Add Game

Name

Create

[Back to List](#)

За добавяне на игра в базата данни използваме бутона **Add Game** който ни отваря нов прозорец с полета за въвеждане на данни.



The screenshot shows the 'Add Game' form in the GamingCatalog application. The form includes the following fields and controls:

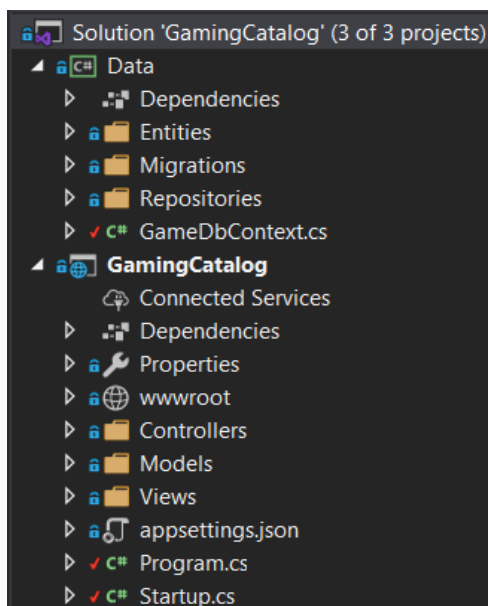
- Title**: Text input field.
- Price**: Text input field.
- Rating**: Text input field.
- Release Date**: Text input field with a date mask 'mm/dd/yyyy'.
- Platform**: Text input field.
- Manufacturer**: Dropdown menu with 'Riot Games' selected.
- Description**: Text area.
- Image**: File upload control with a 'Choose File' button and 'No file chosen' text.
- Genres**: Dropdown menu with 'Hack and Slash', 'Action role-playing', 'MOBA', and 'Stealth' options.
- Create**: Blue button to submit the form.
- Back to List**: Link at the bottom left.

The top navigation bar includes links for 'Catalog', 'Genres', and 'Manufacturers', along with the 'Add Game' button.

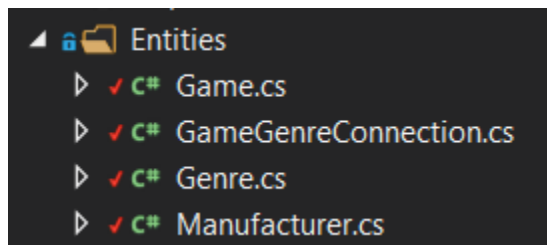
3. Структура на уеб приложението

Приложението се състои от едно конзолно приложение с име **Data** и един **ASP.Net Core MVC** проект с име **GamingCatalog**. **Data** съдържа папки **Entities**, **Migrations**, **Repositories** и **class GameDbContext.cs** в които се състои кода на създадената база данни която се използва от приложението.

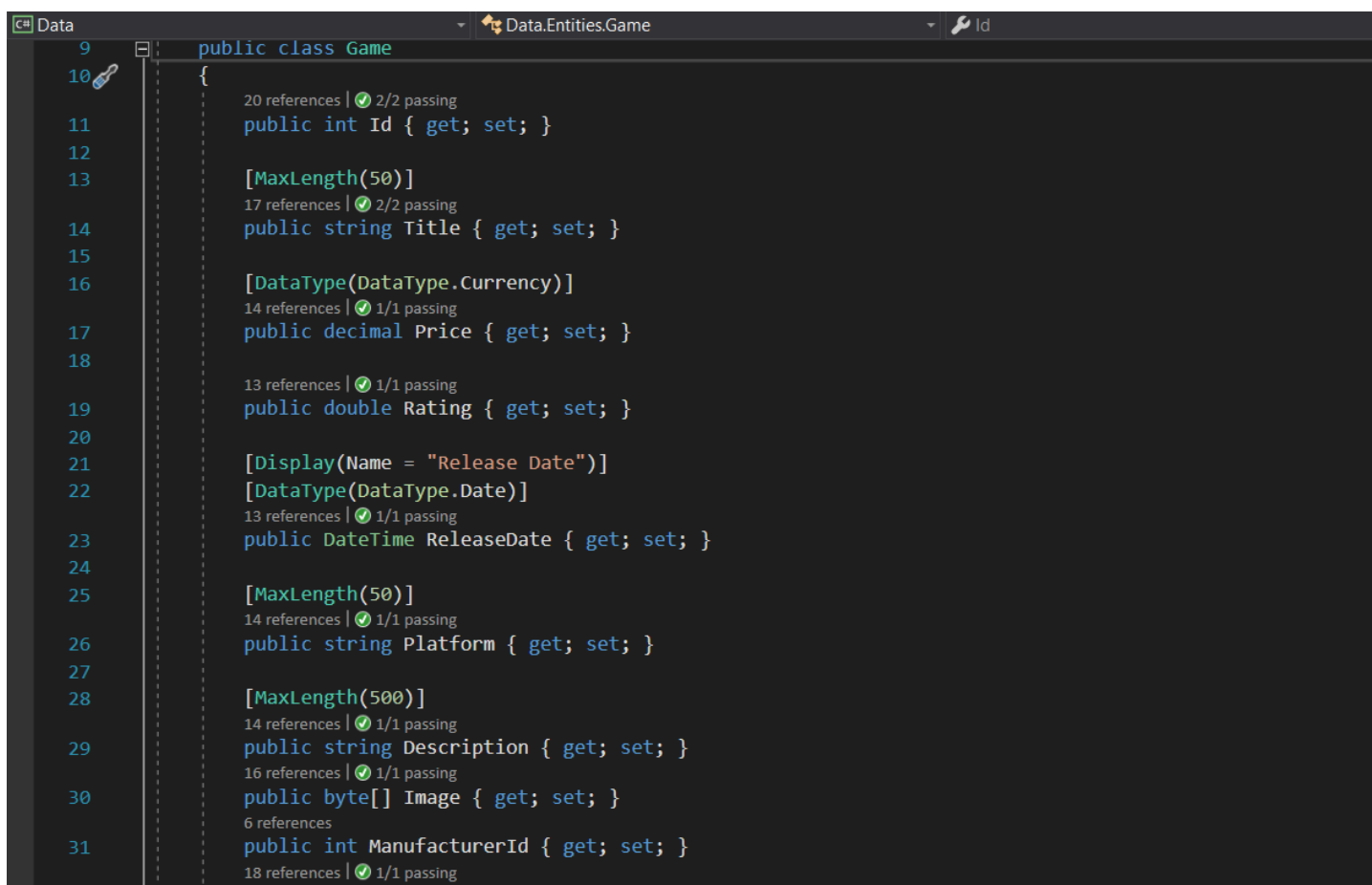
GamingCatalog съдържа папки **wwwroot**, **Controllers**, **Views** и два **cs** файла **Program.cs** и **Startup.cs**.



Папка **Entities** се състои от 4 **cs** файла **Game.cs**, **GameGenreConnection.cs**, **Genre.cs** и **Manufacturer.cs**.



В **Game.cs** се намират свойствата за създаване на игра в базата данни.



В **GameGenreConnetion.cs** се намира връзката между **Game.cs** и **Genre.cs**.

```
Data | Data.Entities.GameGenreConnection | Gameld
1 namespace Data.Entities
2 {
3     //Here is the connection between game and genre.
4     8 references
5     public class GameGenreConnection
6     {
7         6 references
8         public int GameId { get; set; }
9         3 references
10        public Game Game { get; set; }
11
12        3 references
13        public int GenreId { get; set; }
14        5 references
15        public Genre Genre { get; set; }
16    }
17 }
```

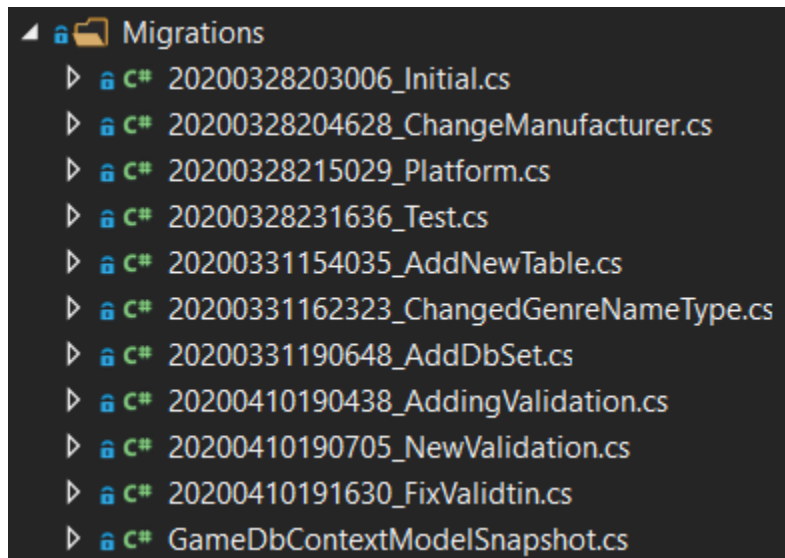
В **Genre.cs** се намират свойствата за въвеждане на жанр на играта в базата данни.

```
Data | Data.Entities.Genre | Id
1 using System.Collections.Generic;
2
3 namespace Data.Entities
4 {
5
6     //Here we create a genre model for the Database.
7     30 references
8     public class Genre
9     {
10         15 references | 1/1 passing
11         public int Id { get; set; }
12         19 references | 1/1 passing
13         public string Name { get; set; }
14
15         1 reference
16         public ICollection<GameGenreConnection> GameGenres { get; set; } = new HashSet<GameGenreConnection>();
17     }
18 }
```

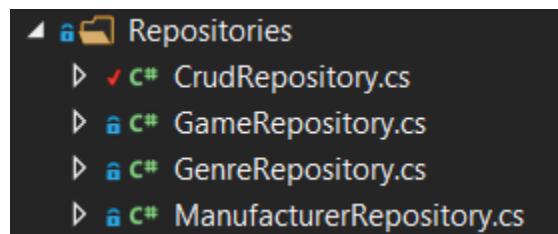
В **Manufacturer.cs** се намират свойствата за въвеждане на разработчик на играта в базата данни.

```
Data | Data.Entities.Manufacturer | Id
1 using System.Collections.Generic;
2
3 namespace Data.Entities
4 {
5
6     //Here we create a manufacturer model for the Database.
7     32 references
8     public class Manufacturer
9     {
10         15 references | 1/1 passing
11         public int Id { get; set; }
12
13         27 references | 2/2 passing
14         public string Name { get; set; }
15
16         1 reference
17         public ICollection<Game> Games { get; set; } = new HashSet<Game>();
18     }
19 }
```

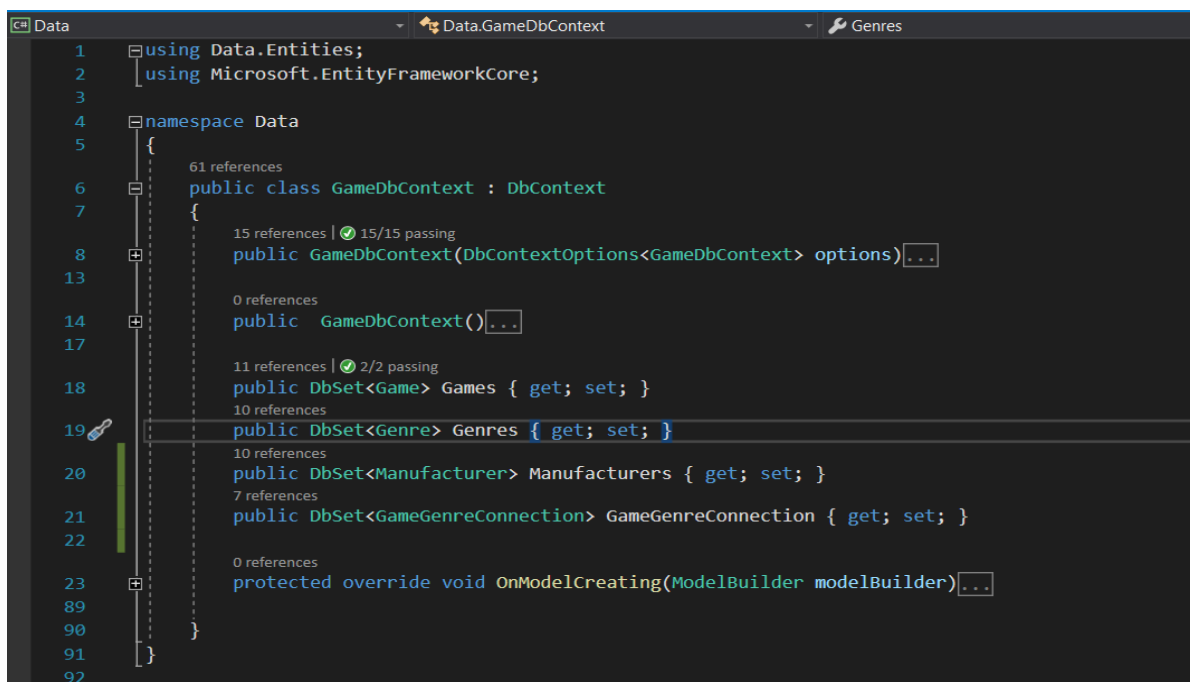
В папката **Migrations** се намират всички миграции към базата данни.



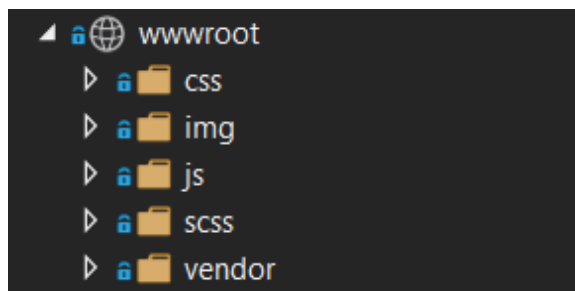
В папката **Repositories** се намират методите на **CRUD** операциите които се изпълняват в приложението при създаване на игра.



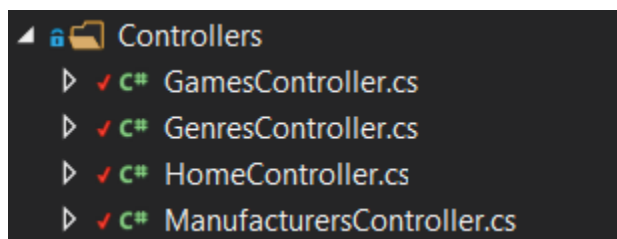
И в последния файл **GameDbContext.cs** се създават таблиците и се описват връзките между данните в **Entities** папките.



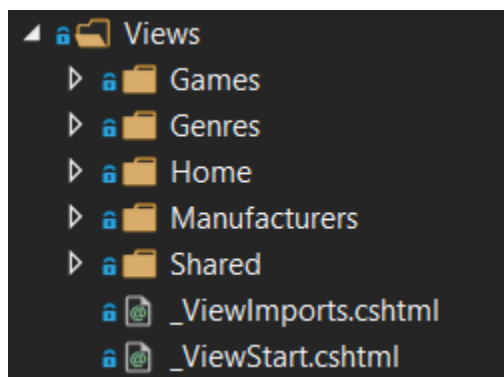
Първата папка от **GamingCatalog** проекта с която започваме е **wwwroot**. В нея се намира **front-end**-а на уеб приложението.



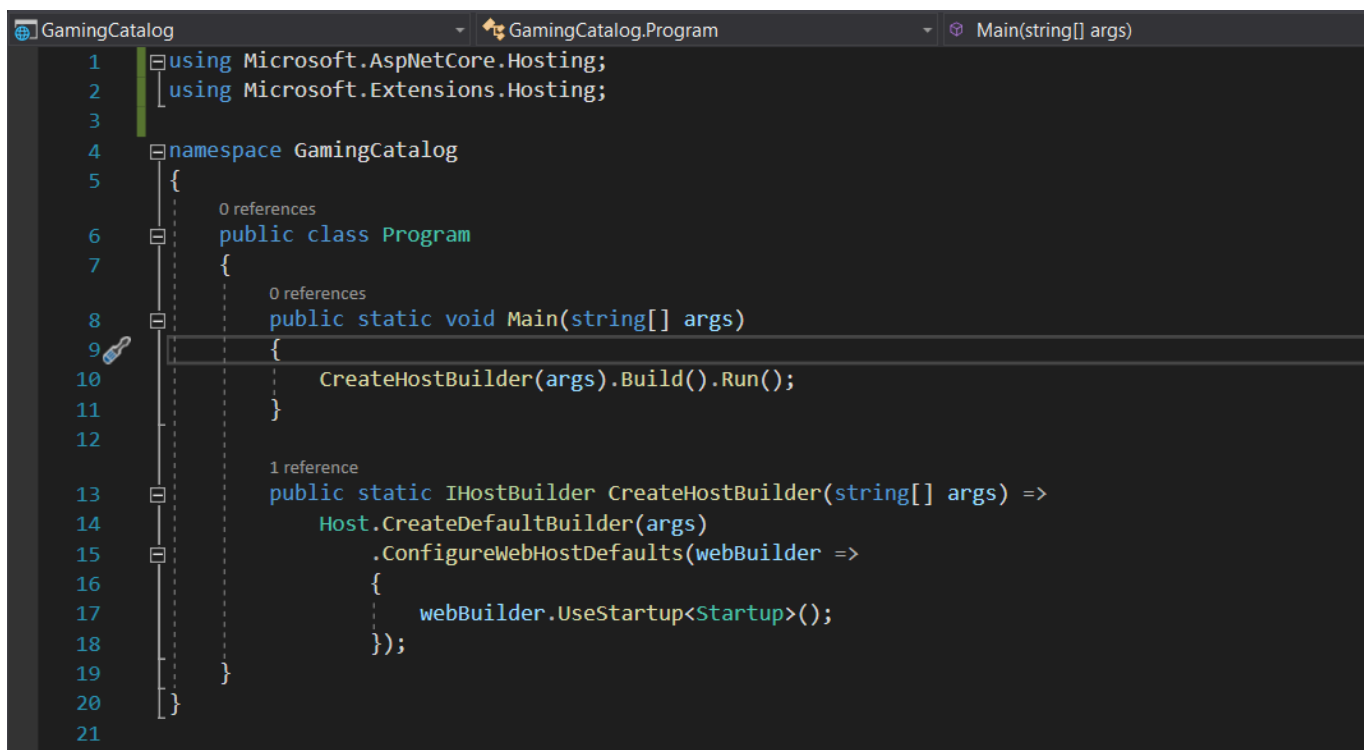
Следващата папка в проекта е **Controllers**. В нея се намират методите на отделните страници в уеб приложението.



И в папката **Views** се намира **Html** код за **front-end**-а на отделните страници.

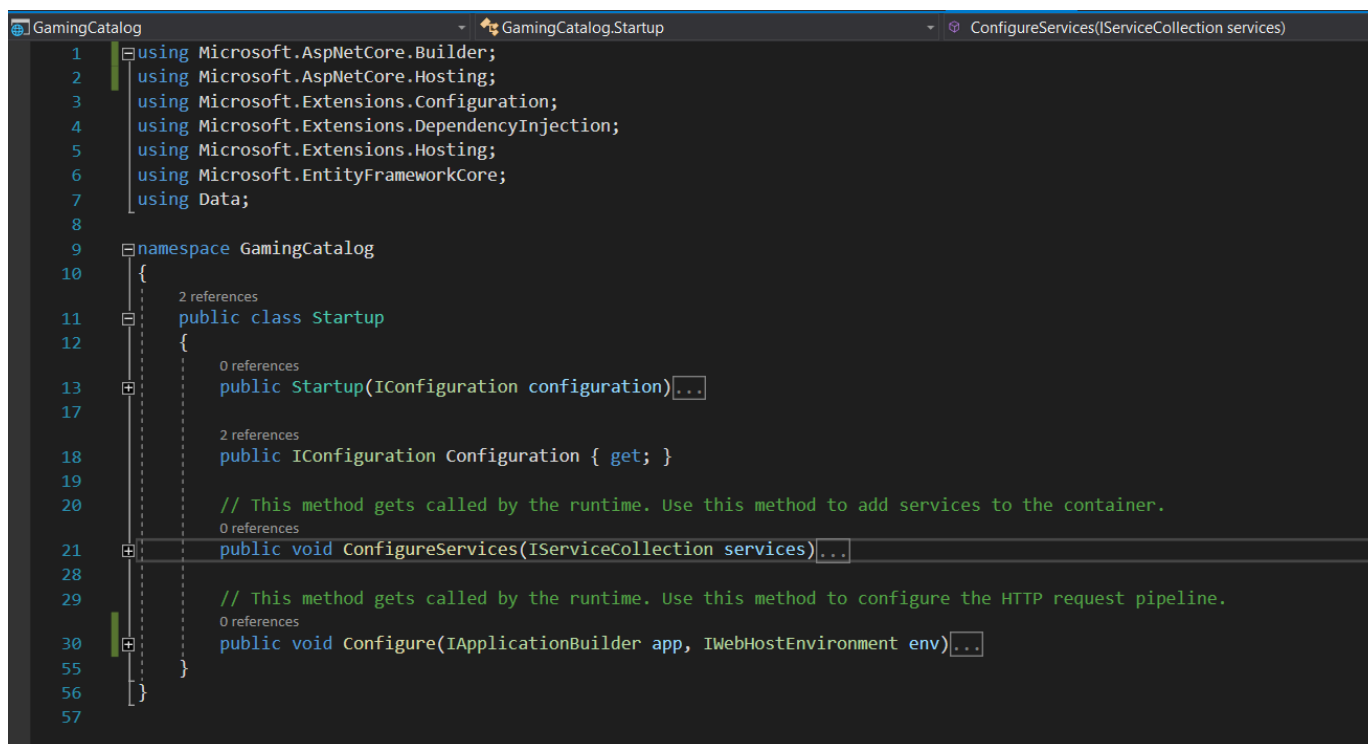


Program.cs е файла от който се стартира цялото приложение

A screenshot of a Visual Studio code editor showing the Program.cs file for a project named 'GamingCatalog'. The file is part of the 'GamingCatalog.Program' namespace and contains a 'Main' method. The code includes using statements for Microsoft.AspNetCore.Hosting and Microsoft.Extensions.Hosting. It defines a 'GamingCatalog' namespace containing a 'Program' class with a 'Main' method. The 'Main' method calls 'CreateHostBuilder(args).Build().Run()'. There is also a 'CreateHostBuilder' method that returns an 'IHostBuilder' by calling 'Host.CreateDefaultBuilder' and configuring web host defaults.

```
1 using Microsoft.AspNetCore.Hosting;
2 using Microsoft.Extensions.Hosting;
3
4 namespace GamingCatalog
5 {
6     public class Program
7     {
8         public static void Main(string[] args)
9         {
10             CreateHostBuilder(args).Build().Run();
11         }
12
13         public static IHostBuilder CreateHostBuilder(string[] args) =>
14             Host.CreateDefaultBuilder(args)
15                 .ConfigureWebHostDefaults(webBuilder =>
16                 {
17                     webBuilder.UseStartup<Startup>();
18                 });
19     }
20 }
21
```

А в **Startup.cs** файла са конфигурациите които са дадени за изпълнение от приложението.

A screenshot of a Visual Studio code editor showing the Startup.cs file for a project named 'GamingCatalog'. The file is part of the 'GamingCatalog.Startup' namespace and contains a 'Startup' class. The code includes using statements for Microsoft.AspNetCore.Builder, Microsoft.AspNetCore.Hosting, Microsoft.Extensions.Configuration, Microsoft.Extensions.DependencyInjection, Microsoft.Extensions.Hosting, Microsoft.EntityFrameworkCore, and Data. It defines a 'GamingCatalog' namespace containing a 'Startup' class. The 'Startup' class has a constructor that takes an 'IConfiguration' object. It also has a 'ConfigureServices' method that takes an 'IServiceCollection' object and a 'Configure' method that takes an 'IApplicationBuilder' and an 'IWebHostEnvironment' object. There are comments explaining the purpose of these methods.

```
1 using Microsoft.AspNetCore.Builder;
2 using Microsoft.AspNetCore.Hosting;
3 using Microsoft.Extensions.Configuration;
4 using Microsoft.Extensions.DependencyInjection;
5 using Microsoft.Extensions.Hosting;
6 using Microsoft.EntityFrameworkCore;
7 using Data;
8
9 namespace GamingCatalog
10 {
11     public class Startup
12     {
13         public Startup(IConfiguration configuration)
14         {
15             Configuration = configuration;
16         }
17
18         public IConfiguration Configuration { get; }
19
20         // This method gets called by the runtime. Use this method to add services to the container.
21         public void ConfigureServices(IServiceCollection services)
22         {
23             // Add services here
24         }
25
26         // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
27         public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
28         {
29             // Configure the request pipeline here
30         }
31     }
32 }
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
```

Изготвено от: Семир Балджиев, Семир Караахмедов, Самина Демирова и Ивета Говедарова