

# Projekt Sezon Wyścigowy (RacingSeries)

Piotr Szumański 307356

## Spis treści

<b>1. Opis projektu</b>	2
<b>2. Ogólne</b>	2
2.1. Projekt powinien być przechowywany w repozytorium github	2
2.2. Zastosowanie architektury cebulowej	3
2.3. Powiązanie z interfejsem użytkownika odbywa się za pośrednictwem REST/SOAP	3
2.4. Stosowanie modeli domenowych (Domain-Driven Design)	4
2.5. Logowanie błędów aplikacji przy pomocy dodatkowego frameworka	4
2.6. Dodanie mechanizmu uwierzytelnienia i autoryzacji	4
2.7. Obsługa systemu ról	7
2.8. Identyfikacja użytkowników przy pomocy JSON Web Token	9
2.9. Stworzenie testów jednostkowych jednego repozytorium z mockami bazy	9
2.10. Stworzenie serwisu agregujących kilka operacji (np: dodanie użytkownika do bazy i wysłanie maila)	9
<b>3. UI</b>	9
3.1. Zastosowanie biblioteki bootstrap (obsługa urządzeń mobilnych)	9
3.2. Wyświetlanie (przeglądanie) danych	9
3.3. Filtrowanie danych (AJAX)	10
3.4. Zastosowanie strocniowania (możliwe użycie gotowych kontrolerek ajax	10
3.5. Widok dodania nowego rekordu	10
3.6. Widok edycji rekordu	11
3.7. Opcja usunięcia rekordu	12
3.8. Opcja wgrania zdjęcia z możliwością przestania pliku na server (SignalR)	13
3.9. Opcja instalacji aplikacji jako aplikacja PWA	13
3.10. Wdrożenie nowoczesnego interfejsu użytkownika bazując na szablonie HTML	13
<b>4. Bazy danych</b>	13
4.1. Zastosowanie relacyjnej bazy danych	13
4.2. Użycie minimum 5 tabel	14
4.3. Określenie kluczy głównych i obcych (constraints)	16
4.4. Zastosowanie mechanizmu ORM	17
4.5. Użycie co najmniej jednej relacji 1-*, 1-1, *-*	17

## 1. Opis projektu

W ramach projektu powstała aplikacja Racing Series, która pozwala na przechowywanie oraz edycję (oraz dodawanie oraz usuwanie) danych odnośnie: kierowców, zespołów, wyścigów, torów wyścigowych oraz sponsorów.

RacingSeries Home Drivers Teams Racing Events Tracks Sponsors \*RaceControl Sign in Register

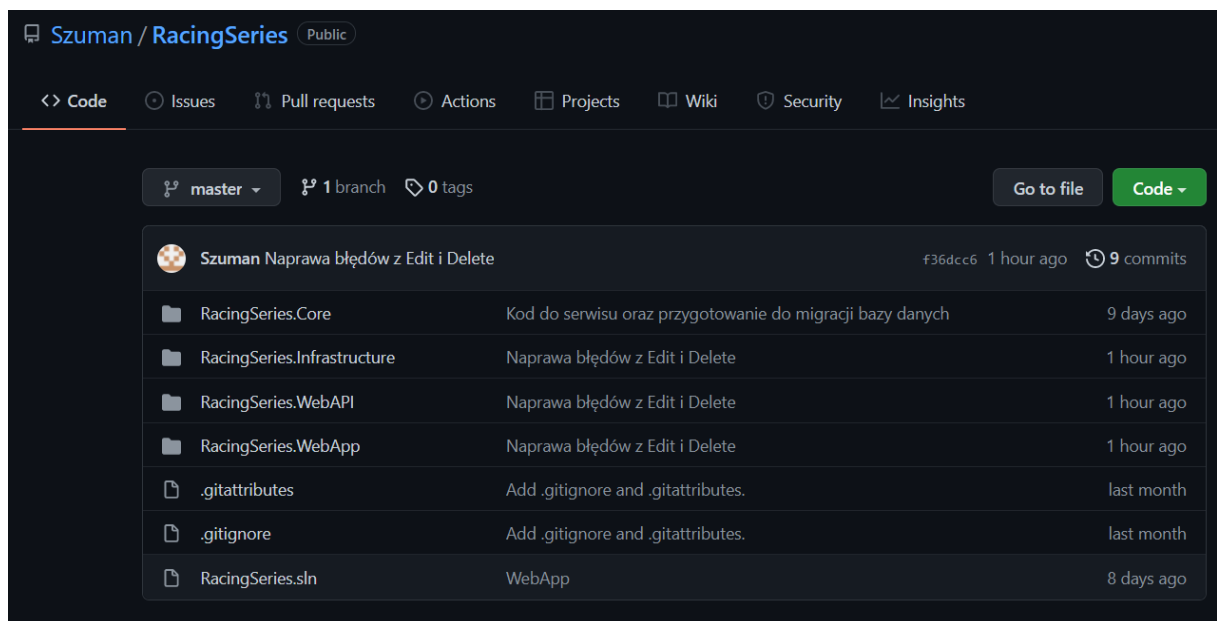
# Welcome to Racing Series

One and only racing series in the world.

© 2022 - RacingSeries.WebApp

## 2. Ogólne

### 2.1. Projekt powinien być przechowywany w repozytorium github



Szuman / RacingSeries (Public)

<> Code Issues Pull requests Actions Projects Wiki Security Insights

master 1 branch 0 tags Go to file Code

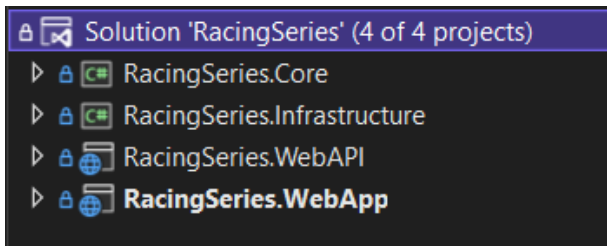
Szuman Naprawa błędów z Edit i Delete #36dcc6 1 hour ago 9 commits

RacingSeries.Core	Kod do serwisu oraz przygotowanie do migracji bazy danych	9 days ago
RacingSeries.Infrastructure	Naprawa błędów z Edit i Delete	1 hour ago
RacingSeries.WebAPI	Naprawa błędów z Edit i Delete	1 hour ago
RacingSeries.WebApp	Naprawa błędów z Edit i Delete	1 hour ago
.gitattributes	Add .gitignore and .gitattributes.	last month
.gitignore	Add .gitignore and .gitattributes.	last month
RacingSeries.sln	WebApp	8 days ago

Link do repozytorium na github:

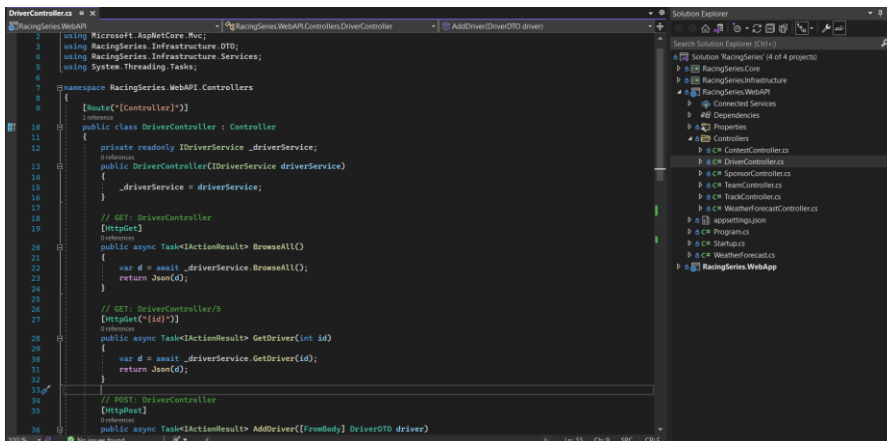
<https://github.com/Szuman/RacingSeries>

## 2.2. Zastosowanie architektury cebulowej

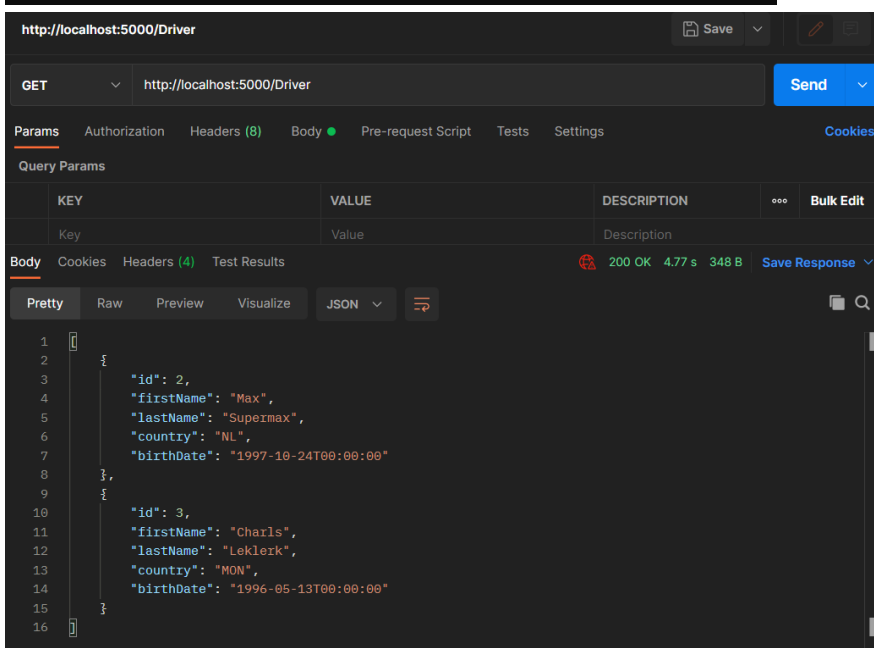


## 2.3. Powiązanie z interfejsem użytkownika odbywa się za pośrednictwem REST/SOAP.

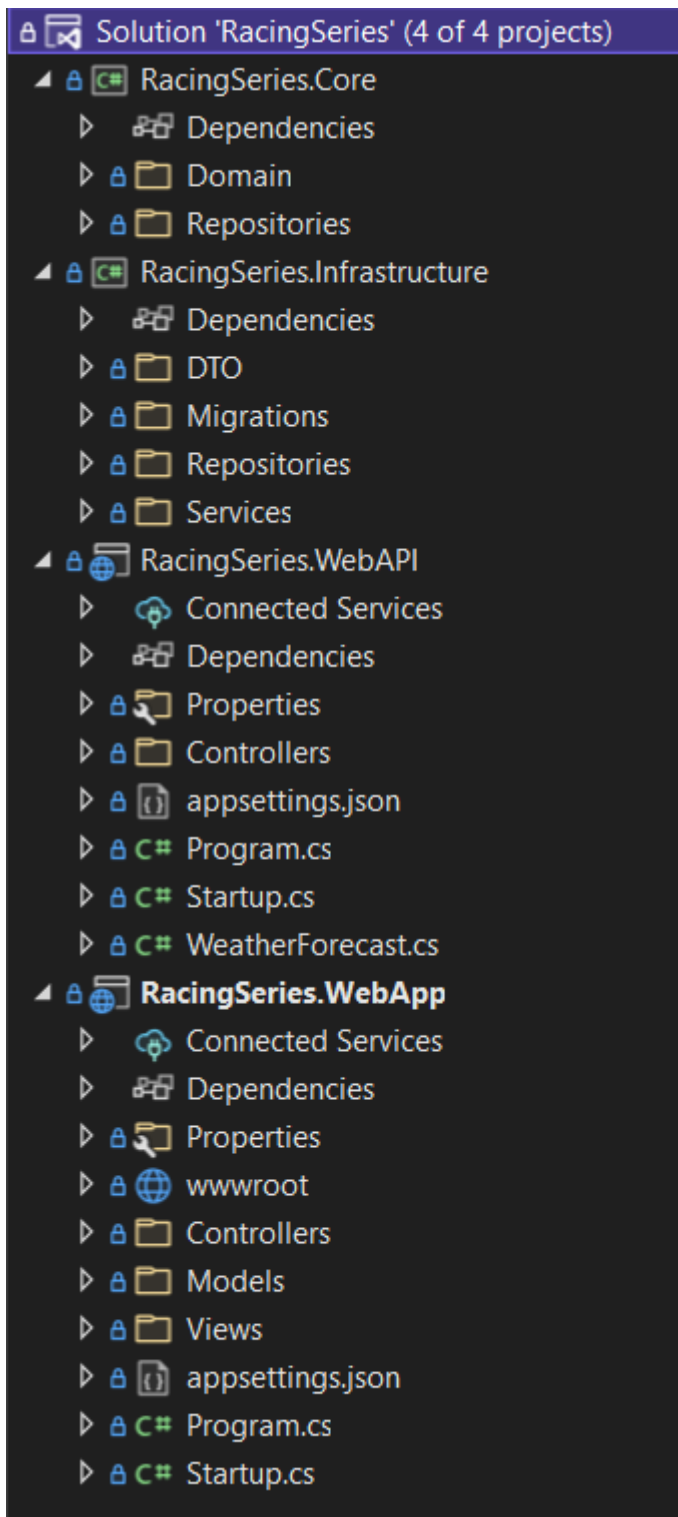
Przykład Driver:



```
PS D:\PAWiM\RacingSeries\RacingSeries.WebAPI> dotnet run
info: Microsoft.Hosting.Lifetime[0]
Now listening on: https://localhost:5001
info: Microsoft.Hosting.Lifetime[0]
Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
Content root path: D:\PAWiM\RacingSeries\RacingSeries.WebAPI
```



## 2.4. Stosowanie modeli domenowych (Domain-Driven Design)



## 2.5. Logowanie błędów aplikacji przy pomocy dodatkowego frameworka

<BRAK>

## 2.6. Dodanie mechanizmu uwierzytelnienia i autoryzacji

WebApp/Controllers/AccountController.cs

```

AccountController.cs
RacingSeries.WebApp
RacingSeries.WebApp.Controllers.AccountController
_signInManager

5
6 namespace RacingSeries.WebApp.Controllers
7 {
8     1 reference
9     public class AccountController : Controller
10    {
11        private readonly SignInManager<IdentityUser> _signInManager;
12        private readonly UserManager<IdentityUser> _userManager;
13
14        0 references
15        public AccountController(SignInManager<IdentityUser> signInManager, UserManager<IdentityUser> userManager)
16        {
17            _signInManager = signInManager;
18            _userManager = userManager;
19        }
20
21        0 references
22        public IActionResult Login()
23        {
24            return View();
25        }
26
27        //Login POST
28        [HttpPost]
29        0 references
30        public async Task<IActionResult> Login(LoginVM loginVM)
31        {
32            if (!ModelState.IsValid)
33                return View(loginVM);
34
35            var user = await _userManager.FindByNameAsync(loginVM.UserName);
36
37            if (user != null)
38            {
39                //login
40                var result = await _signInManager.PasswordSignInAsync(user, loginVM.Password, false, false);
41                if (result.Succeeded)
42                {
43                    return RedirectToAction("Index", "Home");
44                }
45            }
46        }
47    }
48 }

```

WebApp/Controllers/AccountController.cs

```

AccountController.cs
RacingSeries.WebApp
RacingSeries.WebApp.Controllers.AccountController
_signInManager

41 ModelState.AddModelError("", "Incorrect username or password..");
42 return View(loginVM);
43 }
44
45 0 references
46 public IActionResult Register()
47 {
48     return View(new LoginVM());
49 }
50
51 //Register POST
52 [HttpPost]
53 0 references
54 public async Task<IActionResult> Register(LoginVM loginVM)
55 {
56     if (ModelState.IsValid)
57     {
58         var user = new IdentityUser() { UserName = loginVM.UserName };
59         var result = await _userManager.CreateAsync(user, loginVM.Password);
60
61         if (result.Succeeded)
62         {
63             return RedirectToAction("Index", "Home");
64         }
65     }
66     return View(loginVM);
67 }
68
69 //Logout
70 [HttpPost]
71 0 references
72 public async Task<IActionResult> Logout()
73 {
74     await _signInManager.SignOutAsync();
75     return RedirectToAction("Index", "Home");
76 }
77 }

```

## WebApp/Models/Account/Login.cshtml

```

Login.cshtml  X
@model RacingSeries.WebApp.Models.LoginVM

<h3>
    Some operations are available with an account<br>
    No account yet?
    <a asp-action="Register" asp-controller="Account">Register!</a>
    @*<a asp-action="RegisterWithConfirm" asp-controller="Account">zarejestruj się!</a>*@
</h3>

<div class="row">
    <div class="col-md-4">
        <form asp-action="Login" asp-controller="Account" method="post" class="form-horizontal" role="form">

            <div asp-validation-summary="All" class="text-danger"></div>

            <div class="form-group">
                <label asp-for="UserName" class="control-label"></label>
                <input asp-for="UserName" class="form-control" />
                <span asp-validation-for="UserName" class="text-danger"></span>
            </div>

            <div class="form-group">
                <label asp-for="Password" class="control-label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-danger"></span>
            </div>

            <div class="form-group">
                <input type="submit" class="btn btn-primary" value="Sign In" />
            </div>
        </form>
    </div>
</div>

```

## WebApp/Models/Account/Register.cshtml

```

Register.cshtml  X
@model RacingSeries.WebApp.Models.LoginVM

<h3>
    Register into the system<br>
    If you have an account already
    <a asp-action="Login" asp-controller="Account">Login!</a>
</h3>

<form asp-action="Register" asp-controller="Account" method="post" class="form-horizontal" role="form">

    <div asp-validation-summary="All" class="text-danger"></div>

    <div class="form-group">
        <label asp-for="UserName" class="col-md-2 control-label"></label>
        <div class="col-md-10">
            <input asp-for="UserName" class="form-control" />
            <span asp-validation-for="UserName" class="text-danger"></span>
        </div>
    </div>

    <div class="form-group">
        <label asp-for="Password" class="col-md-2 control-label"></label>
        <div class="col-md-10">
            <input asp-for="Password" class="form-control" />
            <span asp-validation-for="Password" class="text-danger"></span>
        </div>
    </div>

    <div class="form-group">
        <div class="col-md-offset-2 col-md-10">
            <input type="submit" class="btn btn-primary" value="Register" />
        </div>
    </div>
</form>

```

<https://localhost:44379/Account/Login>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

Some operations are available with an account  
No account yet? [Register!](#)

Username

Password

Sign In

<https://localhost:44379/Account/Register>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

Register into the system  
If you have an account already [login!](#)

Username

Password

Register

## 2.7. Obsługa systemu ról

WebApp/Models/Shared/\_Layout.cshtml

```
@{
    if (User.IsInRole("admin") || User.IsInRole("Masi"))
    {
        <li class="nav-item">
            <a class="nav-link text-dark" asp-area="" asp-controller="Secret" asp-action="Index">Race Control</a>
        </li>
    }
    else //tylko wyswietl nieaktywną opcję
    {
        <li class="nav-link text-dark">
            <span style="color:darkgray">*RaceControl</span>
        </li>
    }
}
```

WebApp/Controllers/SecretController.cs

```
SecretController.cs  Output
RacingSeries.WebApp
1  using Microsoft.AspNetCore.Authorization;
2  using Microsoft.AspNetCore.Mvc;
3
4  namespace RacingSeries.WebApp.Controllers
5  {
6      [Authorize(Roles = "admin,Masi")]
7      public class SecretController : Controller
8      {
9          public IActionResult Index()
10         {
11             return View();
12         }
13     }
14 }
15
```

WebApp/Models/Secret/Index.cshtml

```
Index.cshtml  SecretController.cs  Output
1  @{
2      ViewData["Title"] = "Race Control";
3  }
4  <h1>@ViewData["Title"]</h1>
5
6  <p>PAGE for admins and other secret roles</p>
```

RaceControl jest stroną, która jest zablokowana dla zwykłych użytkowników

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

<https://localhost:44379/Account/Login>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

Some operations are available with an account  
No account yet? [Register!](#)

Username

admin

Password

.....

Sign In



Po zalogowaniu się na konto admina – odblokowane jest RaceControl

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [Race Control](#) [Logout](#)

<https://localhost:44379/Secret>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [Race Control](#) [Logout](#)

## Race Control

PAGE for admins and other secret roles

### 2.8. Identyfikacja użytkowników przy pomocy JSON Web Token

<BRAK>

### 2.9. Stworzenie testów jednostkowych jednego repozytorium z mockami bazy

<BRAK>

### 2.10. Stworzenie serwisu agregujących kilka operacji (np: dodanie użytkownika do bazy i wysłanie maila)

<BRAK>

## 3. UI

### 3.1. Zastosowanie biblioteki bootstrap (obsługa urządzeń mobilnych)

<ZASTOSOWANA GWIAZDKA> - \*

### 3.2. Wyświetlanie (przeglądanie) danych

WebApp/Controller/Driver

```
5 references
public ContentResult GetHostUrl()
{
    var result = Configuration["RestApiUrl:HostUrl"];
    return Content(result);
}

5 references
private string CN()
{
    string cn = ControllerContext.RouteData.Values["controller"].ToString();
    return cn;
}

// GET: Driver
3 references
public async Task<IActionResult> Index()
{
    string _restpath = GetHostUrl().Content + CN();
    List<DriverVM> drivers = new List<DriverVM>();
    using (var httpClient = new HttpClient(clientHandler))
    {
        using (var response = await httpClient.GetAsync(_restpath))
        {
            string apiResponse = await response.Content.ReadAsStringAsync();
            drivers = JsonConvert.DeserializeObject<List<DriverVM>>(apiResponse);
        }
    }
    return View(drivers);
}
```

### <https://localhost:44379/Driver>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	FirstName	LastName	Country	BirthDate	
2	Max	Supermax	NL	24.10.1997 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
5	Nowy	Kubica	PL	21.12.1987 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

### <https://localhost:44379/Contest>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	Name	DateTime	
2	Aus GP	03.03.2022 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

### <https://localhost:44379/Sponsor>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	Name	Description	
2	Orlen	Polish Hot Dog Industry	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

### 3.3. Filtrowanie danych (AJAX)

<ZASTOSOWANA GWIAZDKA> - \*

### 3.4. Zastosowanie stronicowania (możliwe użycie gotowych kontroltek ajax)

<ZASTOSOWANA GWIAZDKA> - \*

### 3.5. Widok dodania nowego rekordu (DONE)

WebApp/Controller/DriverController.cs

```
// GET: Driver/Create
0 references
public IActionResult Create()
{
    DriverVM driverVM = new DriverVM();
    return View(driverVM);
}

// POST: Driver/Create
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Create(DriverVM driverVM)
{
    string _restpath = GetHostUrl().Content + CN();
    DriverVM result = new DriverVM();
    try
    {
        using (var httpClient = new HttpClient(clientHandler))
        {
            string jsonString = System.Text.Json.JsonSerializer.Serialize(driverVM);
            var content = new StringContent(jsonString, Encoding.UTF8, "application/json");

            using (var response = await httpClient.PostAsync(_restpath, content))
            {
                string apiResponse = await response.Content.ReadAsStringAsync();
                result = JsonConvert.DeserializeObject<DriverVM>(apiResponse);
            }
        }
    }
    catch (Exception ex)
    {
        return View(ex);
    }
    return RedirectToAction(nameof(Index));
}
```

<https://localhost:44379/Driver/Create>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

## DriverVM

Id

FirstName

LastName

Country

BirthDate



Create

[Back to List](#)

<https://localhost:44379/Driver>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	FirstName	LastName	Country	BirthDate	
2	Max	Supermax	NL	24.10.1997 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
3	Charls	Leklerk	MON	13.05.1996 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
5	Nowy	Kubica	PL	21.12.1987 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

### 3.6. Widok edycji rekordu

WebApp/Controller/DriverController.cs

```
// GET: Driver/Edit/5
0 references
public async Task<ActionResult> Edit(int id)
{
    string _restpath = GetHostUrl().Content + CN();
    DriverVM driverVM = new DriverVM();
    using (var httpClient = new HttpClient(clientHandler))
    {
        using (var response = await httpClient.GetAsync($"{_restpath}/{id}"))
        {
            string apiResponse = await response.Content.ReadAsStringAsync();
            driverVM = JsonConvert.DeserializeObject<DriverVM>(apiResponse);
        }
    }
    return View(driverVM);
}

// POST: Driver/Edit/5
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<ActionResult> Edit(int id, DriverVM driverVM)
{
    string _restpath = GetHostUrl().Content + CN();
    DriverVM result = new DriverVM();
    try
    {
        using (var httpClient = new HttpClient(clientHandler))
        {
            string jsonString = System.Text.Json.JsonSerializer.Serialize(driverVM);
            var content = new StringContent(jsonString, Encoding.UTF8, "application/json");

            using (var response = await httpClient.PutAsync($"{_restpath}/{driverVM.Id}", content))
            {
                string apiResponse = await response.Content.ReadAsStringAsync();
                result = JsonConvert.DeserializeObject<DriverVM>(apiResponse);
            }
        }
    }
}
```

<https://localhost:44379/Driver/Edit>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

## DriverVM

Id

3

FirstName

Charls

LastName

LeklerEKKk

Country

MON

BirthDate

13.05.1996 00:00



Save

[Back to List](#)

<https://localhost:44379/Driver>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	FirstName	LastName	Country	BirthDate	
2	Max	Supermax	NL	24.10.1997 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
3	Charls	LeklerEKKk	MON	13.05.1996 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
5	Nowy	Kubica	PL	21.12.1987 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

### 3.7. Opcja usunięcia rekordu

WebApp/Controller/DriverController.cs

```
// POST: Driver/Delete/5
0 references
public async Task<IActionResult> Delete(int id)
{
    string _restpath = GetHostUrl().Content + CN();
    DriverVM driverVM = new DriverVM();
    using (var httpClient = new HttpClient(clientHandler))
    {
        using (var response = await httpClient.DeleteAsync($"{_restpath}/{id}"))
        {
            string apiResponse = await response.Content.ReadAsStringAsync();
            driverVM = JsonConvert.DeserializeObject<DriverVM>(apiResponse);
        }
    }

    return RedirectToAction(nameof(Index));
}
```

<https://localhost:44379/Driver>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	FirstName	LastName	Country	BirthDate	
2	Max	Supermax	NL	24.10.1997 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
3	Charls	LeklerEKKk	MON	13.05.1996 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
5	Nowy	Kubica	PL	21.12.1987 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

[RacingSeries](#) [Home](#) [Drivers](#) [Teams](#) [Racing Events](#) [Tracks](#) [Sponsors](#) [\\*RaceControl](#) [Sign in](#) [Register](#)

[Create New](#)

Id	FirstName	LastName	Country	BirthDate	
2	Max	Supermax	NL	24.10.1997 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>
5	Nowy	Kubica	PL	21.12.1987 00:00:00	<a href="#">Edit</a>   <a href="#">Details</a>   <a href="#">Delete</a>

### 3.8. Opcja wgrania zdjęcia z możliwością przestania pliku na server (SignalR)

<ZASTOSOWANA GWIAZDKA> - \*

### 3.9. Opcja instalacji aplikacji jako aplikacja PWA

<ZASTOSOWANA GWIAZDKA> - \*

### 3.10. Wdrożenie nowoczesnego interfejsu użytkownika bazując na szablonie HTML

<BRAK>

## 4. Bazy danych

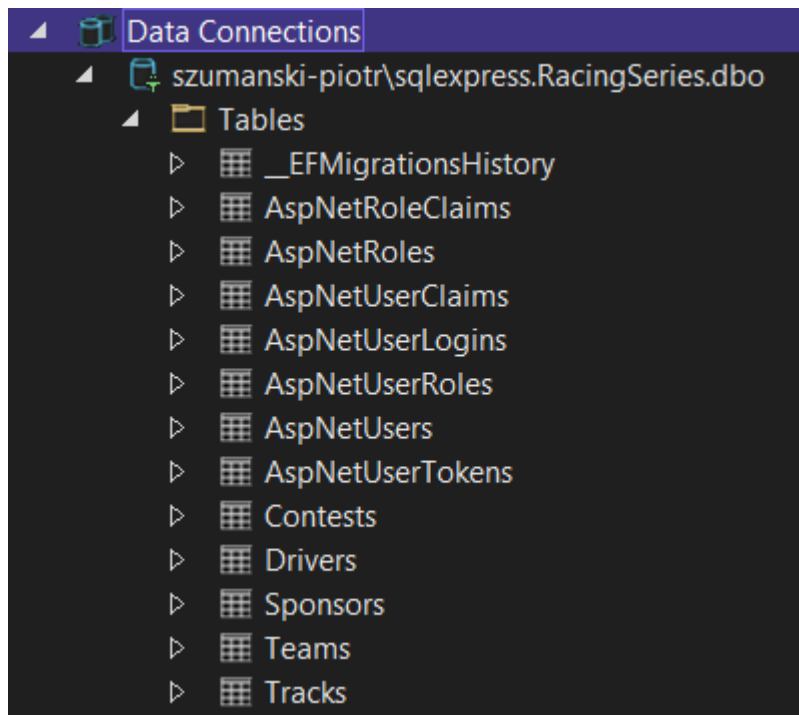
### 4.1. Zastosowanie relacyjnej bazy danych

Infrastructure/Repositories/AppDbContext.cs

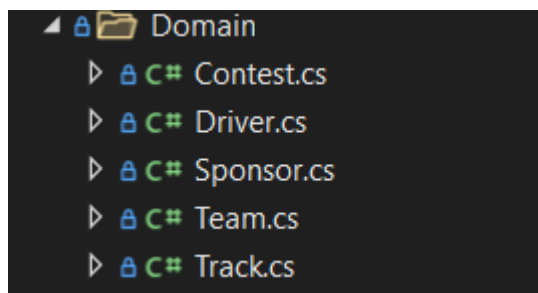
```
AppDbContext.cs
using System;
using System.Collections.Generic;
using System.Text;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
using Microsoft.EntityFrameworkCore;
using RacingSeries.Core.Domain;

namespace RacingSeries.Infrastructure.Repositories
{
    public class AppDbContext : IdentityDbContext<IdentityUser>
    {
        public AppDbContext(DbContextOptions<AppDbContext> options) : base(options)
        {
        }

        public DbSet<Driver> Drivers { get; set; }
        public DbSet<Contest> Contests { get; set; }
        public DbSet<Sponsor> Sponsors { get; set; }
        public DbSet<Team> Teams { get; set; }
        public DbSet<Track> Tracks { get; set; }
    }
}
```



#### 4.2. Użycie minimum 5 tabel



```
namespace RacingSeries.Core.Domain
{
    14 references
    public class Contest
    {
        10 references
        public int Id { get; set; }
        6 references
        public string Name { get; set; }
        0 references
        public Track Track { get; set; }
        6 references
        public DateTime DateTime { get; set; }
        0 references
        public List<Driver> Drivers { get; set; }
    }
}
```

```

namespace RacingSeries.Core.Domain
{
    15 references
    public class Driver
    {
        10 references
        public int Id { get; set; }
        6 references
        public string FirstName { get; set; }
        6 references
        public string LastName { get; set; }
        4 references
        public string Country { get; set; }
        6 references
        public DateTime BirthDate { get; set; }
        0 references
        public Team Team { get; set; }
    }
}

```

```

namespace RacingSeries.Core.Domain
{
    15 references
    public class Sponsor
    {
        10 references
        public int Id { get; set; }
        6 references
        public string Name { get; set; }
        6 references
        public string Description { get; set; }
    }
}

```

```

namespace RacingSeries.Core.Domain
{
    15 references
    public class Team
    {
        10 references
        public int Id { get; set; }
        6 references
        public string Name { get; set; }
        2 references
        public string Country { get; set; }
        6 references
        public DateTime EntryDate { get; set; }
        0 references
        public List<Sponsor> Sponsors { get; set; }
    }
}

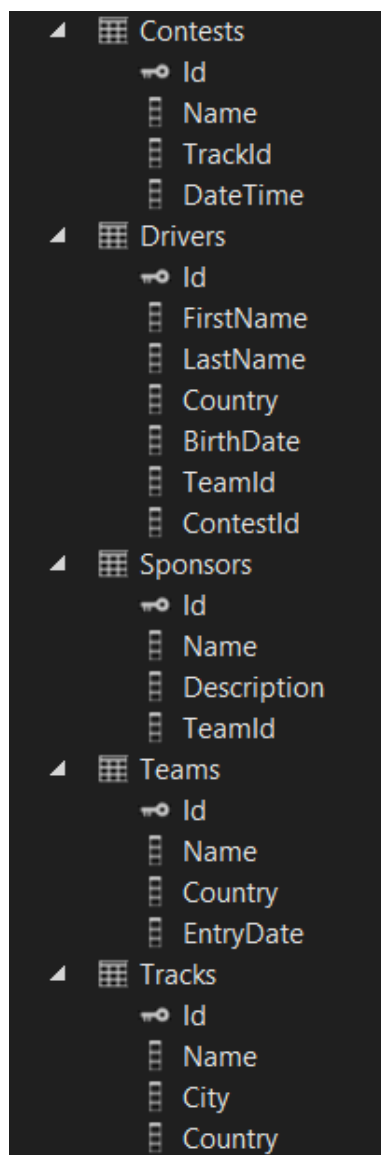
```

```

namespace RacingSeries.Core.Domain
{
    15 references
    public class Track
    {
        10 references
        public int Id { get; set; }
        6 references
        public string Name { get; set; }
        6 references
        public string City { get; set; }
        6 references
        public string Country { get; set; }
    }
}

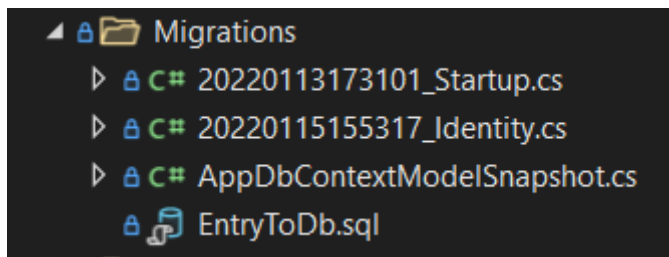
```

#### 4.3. Określenie kluczy głównych i obcych (constraints)





#### 4.4. Zastosowanie mechanizmu ORM



#### 4.5. Użycie co najmniej jednej relacji 1-\*, 1-1, \*-\*

Założenie relacji bazy danych:

Team 1-\* Driver

Event \*-\* Driver

Event 1-1 Track

Sponsor \*-\* Team