

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Кафедра програмної інженерії
КУРСОВА РОБОТА
ПОЯСНЮВАЛЬНА ЗАПИСКА з дисципліни
“Об’єктно - орієнтоване програмування”
Довідник фаната

Керівник , проф. БОНДАРЄВ В.М
Студент гр. ПЗПІ-22-6 Кісіль Д.С.

Комісія:
Проф. Бондарєв В.М.,
Ст. викл. Черепанова Ю.Ю.,
Ст. викл. Ляпота В.М.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
РАДІОЕЛЕКТРОНІКИ

Кафедра: програмної інженерії

Рівень вищої освіти: перший (бакалаврський)

Дисципліна: Об'єктно-орієнтоване програмування

Спеціальність: 121 Інженерія програмного забезпечення

Освітня програма: Програмна інженерія

КУРС _____ 1 _____ ГРУПА _____ ПЗП-22-6 _____

СЕМЕСТР _____ 2 _____

ЗАВДАННЯ
На курсовий проект студента

Кісіля Дмитра Сергійовича

Тема проекту : Довідник Фаната

Мета проекту: Поглиблення знання в розробці C# проекту.

Використані програми: Visual Studio , платформа – ASP.NET

Зміст

<u>Вступ</u>	<u>4</u>
<u>Опис вимог</u>	<u>6</u>
<u>Тестування</u>	<u>8</u>
<u>Проектування програми</u>	<u>9</u>
<u>Інтерфейс</u>	<u>10</u>
<u>Інструкція</u>	<u>19</u>
<u>Висновок</u>	<u>20</u>
<u>Перелік джерел</u>	<u>26</u>

Зписка до проекту

Вступ

Головними компонентами мого проекту "Довідник фаната" є програма на мові C#, яка взаємодіє з базою даних за допомогою Entity Framework. Для цього я використав відповідні інструменти, що дозволяють ефективно здійснювати звернення до бази даних, отримувати необхідні дані і відображати їх для користувача.

Перед початком розробки програми, я налаштував середовище розробки, встановивши необхідні компоненти, такі як Visual Studio, який надає потужні інструменти для розробки програм на мові C# , також я встановив додаткові розширення для Visual Studio , задля розробки веб-програми з використанням ASP.NET

Основна ідея програми "Довідник фаната" полягає у зручному та інформаційному перегляді інформації про різних спортсменів та клуби. Для цього я створив відповідну структуру бази даних, яка включає всю інформацію додану розробником. У кожному записі бази даних містяться дані про спортсменів, клуби, тренерів. Ця структура дозволяє легко організувати і виконувати пошук за різними критеріями, що спрощує використання програми для користувача.

У програмі "Довідник фаната" я також забезпечив можливість детального перегляду інформації спортсменів.

Крім того, я реалізував функціонал фільтрації спортсменів за різними критеріями, наприклад, за клубом або видом спорту. Завдяки цьому користувачі можуть швидко знайти необхідну їм інформацію.

Код програми на C# включає в себе модуль, який забезпечує взаємодію з базою даних, а також інтерфейс користувача, де відображаються інформацію про спортсменів і відповідно до них фільтрацію.

Загалом, моя програма "Довідник фаната" на C# , дозволяє ефективно та зручно переглядати спортивну інформацію на різні теми.

Мета: Моя мета при розробці програми "Довідник фаната" на C# полягала в створенні зручного та ефективного інструменту для пошуку необхідних даних про спортсменів. Я прагнув створити інтуїтивно зрозумілий інтерфейс з необхідним функціоналом, який дозволяє здійснювати пошук інформації за декілька кліків.

В програмі "Довідник Фаната" забезпечено простий та зрозумілий інтерфейс, який дозволяє користувачам легко шукати потрібну їм інформацію. За допомогою відповідних функцій пошуку та фільтрації, користувачі можуть задати свої критерії пошуку та отримати відповідні результати.

Робота програми повинна виглядати наступним чином:

1. Користувач заходить на сторінку програми та відразу може споглядати інформацію.

Склад програми:

1. Розділи з інформацією про спорт, матчі, спортсменів, стадіони, команди та тренерів
2. Поле введення тексту, для пошуку необхідного спортсмена
3. Фільтр розділів, для зручного пошуку

Опис вимог:

Програма "Довідник фаната" має наступні основні функції:

1. Додавання інформації про спортсменів:

- Назва: Додавання інформації про спортсменів

- Додавання інформації можливе тільки для адміністраторів, для цього потрібно пройти авторизацію.

- Опис:

- Адміністратор заповнює всі обов'язкові поля та натискає кнопку "Create".

- Якщо дані введені правильно, програма зберігає інформацію в базі даних

- Якщо дані введені неправильно або є помилки - База Даних виводить повідомлення про помилку.

2. Перегляд інформації про спортсменів:

- Назва: Перегляд інформації про спортсменів

- Передумови: Користувач має доступ до програми і бази даних

- Опис:

- Користувач запускає програму і може обирати розділ, який цікавить користувача.

- Програма завантажує всю інформацію з бази даних і виводить її списком на екран.

- Користувач може прокручувати список оголошень і переглядати деталі кожного оголошення.

- Користувач може застосовувати фільтри для пошуку оголошень за певними критеріями, такими як назва спорту, вік, ім'я спортсмена.

- Програма показує користувачеві відфільтрований список спортсменів згідно з обраними критеріями.

3. Редагування оголошення:

- Назва: Редагування інформації про спортсменів.

- Редагування інформації можливе тільки для адміністраторів, для цього потрібно пройти авторизацію.

-Опис:

- Адміністратор обирає спортсмена, інформацію про якого потрібно відредагувати.

- Програма відкриває форму з деталями вибраного спортсмена.

- Користувач змінює потрібні поля спортсмена і натискає кнопку "Create".

- Програма перевіряє коректність введених даних і зберігає змінену інформацію про спортсмена в базі даних.

- Якщо дані введені неправильно або є помилки - База Даних виводить повідомлення про помилку.

4. Видалення спортсмена:

- Назва: Видалення спортсмена

-Видалення спортсмена можливе тільки для адміністраторів,для цього потрібно пройти авторизацію

- Опис:

- Адміністратор проходить авторизацію і обирає розділ "Sportsmen".

- Адміністратор обирає спортсмена, якого потрібно видалити.

- Програма відкриває форму з деталями вибраного спортсмена.

- Адміністратор після перегляду натискає "Delete"

ТЕСТУВАННЯ

У рамках цього проєкту було проведено тестування розробленої програми "Купівля Продаж" з використанням C# і Windows Forms. Метою тестування було перевірити функціональність програми та переконатися в її коректній роботі в різних ситуаціях.

Тестування додатку було поділено на два основних типи тестування: модульне тестування та функціональне тестування.

Під час написання коду було проведено модульне тестування для автоматичної перевірки окремих компонентів програми. Для цього використовувалися спеціальні фреймворки, такі як NUnit або Microsoft Unit Testing Framework. Модульне тестування є рекомендованим, але необов'язковим етапом розробки.

Функціональне тестування було проведено для перевірки роботи програми відповідно до умов коректних і некоректних дій користувача. У процесі функціонального тестування було створено тестові сценарії, перевірено введення і виведення даних, обробку помилок та інші аспекти, які відповідають вимогам і очікуванням користувача. Планування функціонального тестування було виконано на етапі проектування, а саме тестування проводили по мірі реалізації окремих функцій програми. Тестування завершилося перевіркою роботи програми в цілому.

На основі результатів тестування було вжито заходів для виправлення виявлених помилок і поліпшення функціональності програми. Після внесення виправлень було проведено повторне тестування для перевірки та підтвердження внесених змін.

Тестування програми "Довідник Фаната" дало змогу забезпечити стабільну та надійну роботу програми, а також збільшити впевненість у її якості та правильній реалізації функцій.

З урахуванням вищевикладеного, можна зробити висновок, що проведене тестування сприяло покращенню програми "Довідник

фаната" і підтвердило її працездатність згідно з вимогами і очікуваннями користувачів.

Проектування

На початковому етапі розробки програми "Довідник фаната" було проведено проектування, що включало прийняття важливих рішень, спрямованих на визначення архітектури, структури та формату зберігання зовнішніх даних. Це дозволило забезпечити ефективну та зручну роботу з програмою.

Крім того, була визначена загальна структура програми, включаючи основні модулі, класи та їх взаємозв'язок. Це дозволило організувати програму за функціональними блоками та визначити взаємодію між ними. Використання діаграм допомагає візуалізувати структуру програми, зв'язки між класами та потоки даних в програмі.

Також важливим аспектом проектування є вибір формату зберігання зовнішніх даних, необхідних для роботи програми. Залежно від вимог програми, це може бути файлова система, база даних або інше джерело даних, яке найкраще відповідає потребам програми. Розробники вирішили, який формат зберігання буде найбільш оптимальним та ефективним для використання.

Проектування окремих частин програми включало створення об'єктної моделі кожної частини та попередній опис класів. Для кожної функціональної частини програми була визначена відповідна об'єктна модель, а класи були описані з урахуванням їх функціональності та взаємозв'язку. Це допомагає розробникам краще розуміти логіку програми та планувати реалізацію різних функцій.

Узагальнюючи, проектування програми "Довідник фаната" було виконано з метою визначення структури програми, взаємозв'язків між її частинами та забезпечення відповідності вимогам користувача. Цей етап є важливим перед переходом до розробки програми, оскільки допомагає зрозуміти загальну концепцію проекту та планувати його реалізацію

Огляд інтерфейсу програми:

1. На рис 1.1 зображено вікно авторизації адміністраторів, за допомогою якого можна додавати, редагувати та видаляти контент.

Login

Email

Password

Login

рис. 1.1. Вікно авторизації

Email – пошта адміністратора, створена під час розробки програми

Password - пароль адміністратора, створений під час розробки програми.

2. На рисунках 1.2 – 1.7 зображено основний інтерфейс програми, де біля кожного елемента програми є детальна інформація, також присутні фільтри сортування, присутній пошук елементів за різними ознаками.

Index

Name	
Football	Details
Basketball	Details
Volleyball	Details
Hockey	Details

рис. 1.2. Інтерфейс розділу “Sports”

FanGuide
Home
Mathces
Sports
Sportsmen
Stadiums
Teams
Trainers
Login

Index

Team name

Sportsman name

Sport name

Date from01.01.0001

Date to27.08.2023

SortingSort by date (new first)


Search

Date	HomeTeam	VisitorTeam	HomeTeamScore	VisitorTeamScore	Sport
27.07.2017 22:00:00	Paris Saint-Germain	Juventus	2	3	Football Details
23.11.2021 22:00:00	Chelsea	Juventus	4	0	Football Details
09.03.2022 22:00:00	Real Madrid	Paris Saint-Germain	3	1	Football Details
17.12.2022 12:00:00	Chicago Bulls	New York Knicks	91	114	Basketball Details
05.01.2023 12:00:00	Los Angeles Lakers	Miami Heat	112	109	Basketball Details
26.03.2023 0:00:00	Carolina Hurricanes	Toronto Maple Leafs	5	3	Hokey Details
29.03.2023 18:00:00	Halkbank	Jastrzębski Węgiel	1	3	Volleyball Details
29.03.2023 20:30:00	ZAKSA Kędzierzyn-Koźle	Sir Safety Perugia	3	1	Volleyball Details
30.03.2023 12:00:00	Chicago Bulls	Los Angeles Lakers	110	121	Basketball Details
05.04.2023 20:30:00	Jastrzębski Węgiel	Halkbank	2	3	Volleyball Details
06.04.2023 20:30:00	Sir Safety Perugia	ZAKSA Kędzierzyn-Koźle	1	3	Volleyball Details
07.04.2023 1:00:00	Boston Bruins	Toronto Maple Leafs	2	1	Hokey Details

рис. 1.3. Інтерфейс розділу “Matches”

Index

0
 0

Sorting
 Sort by surname (A-Z)
 

Search

Team	Name	Surname	Age	Nationally	Sport	
Halkbank	Nimir	Abdel-Aziz	31	Dutch	Volleyball	Details
Real Madrid	Karim	Benzema	35	France	Football	Details
Juventus	Leonardo	Bonucci	36	Italy	Football	Details
New York Knicks	Jalen	Brunson	26	USA	Basketball	Details
Miami Heat	Jimmy	Butler	33	USA	Basketball	Details
New Jersey Devils	Nico	Hischier	24	Switzerland	Hokey	Details
Los Angeles Lakers	LeBron	James	38	USA	Basketball	Details
Boston Bruins	Brad	Marchand	35	Canada	Hokey	Details
Toronto Maple Leafs	Mitchell	Marnier	26	Canada	Hokey	Details
Carolina Hurricanes	Jordan	Martinook	30	Canada	Hokey	Details
Paris Saint-Germain	Kylian	Mbappé	24	France	Football	Details
Chelsea	Mykhailo	Mudryk	22	Ukraine	Football	Details

рис. 1.4. Інтерфейс розділу “Sportsmen”

Index

Name	City	StadiumCapacity	HomeTeam	
Santiago Bernabéu	Madrid , Spain	83186	Real Madrid	Details
Stamford Bridge	London, England	40343	Chelsea	Details
Parc des Princes	Paris , France	47929	Paris Saint-Germain	Details
Juventus Stadium	Turin , Italy	41507	Juventus	Details
United Center	Chicago, Illinois, United States	23129	Chicago Bulls	Details
Crypto.com Arena	Los Angeles, California	19079	Los Angeles Lakers	Details
Madison Square Garden	New York City, New York	19812	New York Knicks	Details
Kaseya Center	Miami, Florida	19600	Miami Heat	Details
PalaEvangelisti	Perugia , Italy	5300	Sir Safety Perugia	Details
Hala Azoty	Kędzierzyn-Koźle , Poland	3375	ZAKSA Kędzierzyn-Koźle	Details
HWS	Jastrzębie-Zdrój , Poland	3112	Jastrzębski Węgiel	Details
Başkent Voleybol Salonu	Beşevler, Yenimahalle, Ankara, Turkey	7600	Halkbank	Details
TD Garden	Boston , Massachusetts	17850	Boston Bruins	Details
Scotiabank Arena	Toronto , Ontario	18800	Toronto Maple Leafs	Details
PNC Arena	Raleigh, North Carolina	18680	Carolina Hurricanes	Details


рис. 1.5. Інтерфейс розділу “Stadiums”

Index

Name	City	CreateDate	Sport	
Real Madrid	Madrid , Spain	06.03.1902 0:00:00	Football	Details
Chelsea	London , England	10.03.1905 0:00:00	Football	Details
Paris Saint-Germain	Paris , France	12.08.1970 0:00:00	Football	Details
Juventus	Turin , Italy	01.11.1897 0:00:00	Football	Details
Chicago Bulls	Chicago, Illinois	16.01.1996 0:00:00	Basketball	Details
Los Angeles Lakers	Los Angeles, California	12.05.1960 0:00:00	Basketball	Details
New York Knicks	New York City , New York	12.05.1946 0:00:00	Basketball	Details
Miami Heat	Miami , Florida	27.03.1988 0:00:00	Basketball	Details
Sir Safety Perugia	Perugia , Italy	24.06.2001 0:00:00	Volleyball	Details
ZAKSA Kędzierzyn-Koźle	Kędzierzyn-Koźle , Poland	12.11.1947 0:00:00	Volleyball	Details
Jastrzębski Węgiel	Jastrzębie-Zdrój , Poland	25.01.1961 0:00:00	Volleyball	Details
Halkbank	Ankara , Turkey	21.07.1983 0:00:00	Volleyball	Details
Boston Bruins	Boston, Massachusetts	27.08.1924 0:00:00	Hokey	Details
Toronto Maple Leafs	Toronto, Ontario	01.06.1917 0:00:00	Hokey	Details
Carolina Hurricanes	Raleigh, North Carolina	08.10.1972 0:00:00	Hokey	Details

рис. 1.6. Інтерфейс розділу “Teams”

Index

Trainer name	<input type="text"/>	Trainer surname	<input type="text"/>	Team name	<input type="text"/>
Trainer age from	<input type="text" value="0"/>	Trainer age to	<input type="text" value="0"/>	Sorting	Sort by trainer surname (A-Z) 
<input type="button" value="Search"/>					

Team	Name	Surname	Age	
Sir Safety Perugia	Angelo Lorenzetti	Volleyball coach	59	Details
Chicago Bulls	Billy Donovan	Basketball coach	58	Details
Real Madrid	Carlo Ancelotti	Football coach	64	Details
Los Angeles Lakers	Darvin Ham	Basketball coach	49	Details
Miami Heat	Erik Spoelstra	Basketball coach	52	Details
Boston Bruins	Jim Montgomery	Hockey coach	54	Details
New Jersey Devils	Lindy Ruff	Hockey coach	63	Details
Paris Saint-Germain	Luis Enrique	Football coach	53	Details
Jastrzębski Węgiel	Marcelo Méndez	Volleyball coach	59	Details
Juventus	Massimiliano Allegri	Football coach	55	Details
Chelsea	Mauricio Pochettino	Football coach	51	Details
Carolina Hurricanes	Rod Brind'Amour	Hockey coach	52	Details

рис. 1.7. Інтерфейс розділу “Trainers”

Запуск Проєкту

Для початку роботи треба розпакувати архів з проєктом в бажану вам папку.

1.Після запуску проєкту в Visual Studio,натискаємо кнопку “Start”,після цього ми потрапляємо на головну сторінку програми.

2.Програма розділена на різні розділи такі як :
Sports,Matches,Sportsmen,Stadiums,Teams,Trainers.

3.Користувач може переглянути будь-який розділ та застосувати фільтри пошуку за для зручного пошуку інформації.

Детальний розбір програми

1.На рисунку 2.1. зображено вікно авторизації адміністраторів.

Login

Email

Password

Login

рис. 2.1. Вікно авторизації

2. На рисунку 2.2. зображений повний список спортсменів

FanGuide

Home

Mathces

Sports

Sportsmen

Stadiums

Teams

Trainers

Login

Index

Sportsman name

Sportsman surname

Team name

Sport name

Sportsman age from0

Sportsman age to0

SortingSort by surname (A-Z)

Search

Team	Name	Surname	Age	Nationally	Sport	
Halkbank	Nimir	Abdel-Aziz	31	Dutch	Volleyball	Details
Real Madrid	Karim	Benzema	35	France	Football	Details
Juventus	Leonardo	Bonucci	36	Italy	Football	Details
New York Knicks	Jalen	Brunson	26	USA	Basketball	Details
Miami Heat	Jimmy	Butler	33	USA	Basketball	Details
New Jersey Devils	Nico	Hischier	24	Switzerland	Hokey	Details
Los Angeles Lakers	LeBron	James	38	USA	Basketball	Details
Boston Bruins	Brad	Marchand	35	Canada	Hokey	Details
Toronto Maple Leafs	Mitchell	Marner	26	Canada	Hokey	Details
Carolina Hurricanes	Jordan	Martinook	30	Canada	Hokey	Details
Paris Saint-Germain	Kylian	Mbappé	24	France	Football	Details
Chelsea	Mykhailo	Mudryk	22	Ukraine	Football	Details

рис. 2.2. Список спортсменів

На рисунку 2.2. можна побачити у списку спортсменів наступну інформацію:

- 1.Team – Команда в якій грає спортсмен.
- 2.Name – Ім'я спортсмена.
- 3.Surname – Прізвище спортсмена.
- 4.Age – Вік спортсмена.
- 5.Nationally – Національність.
- 6.Sport – Вид спорту в якому виступає спортсмен.
- 7.Details – Додаткова інформація про спортсмена

3. На рисунку 2.3. зображено фільтри пошуку спортсменів за різними критеріями

FanGuide Home Mathces Sports Sportsmen Stadiums Teams Trainers Login

Index

Sportsman name

Sportsman surname

Team name

Sport name

Sportsman age from 0

Sportsman age to 0

Sorting

Sort by surname (A-Z)

Search

Team	Name	Surname	Age	Nationally	Sport	
Halkbank	Nimir	Abdel-Aziz	31	Dutch	Volleyball	Details
Real Madrid	Karim	Benzema	35	France	Football	Details
Juventus	Leonardo	Bonucci	36	Italy	Football	Details
New York Knicks	Jalen	Brunson	26	USA	Basketball	Details
Miami Heat	Jimmy	Butler	33	USA	Basketball	Details
New Jersey Devils	Nico	Hischier	24	Switzerland	Hokey	Details
Los Angeles Lakers	LeBron	James	38	USA	Basketball	Details
Boston Bruins	Brad	Marchand	35	Canada	Hokey	Details
Toronto Maple Leafs	Mitchell	Marner	26	Canada	Hokey	Details
Carolina Hurricanes	Jordan	Martinook	30	Canada	Hokey	Details
Paris Saint-Germain	Kylian	Mbappé	24	France	Football	Details
Chelsea	Mykhailo	Mudryk	22	Ukraine	Football	Details

рис. 2.3. Фільтри пошуку спортсменів

- 1.Sportsman name – Знайти спортсмена за ім'ям.
- 2.Sportsman surname – Знайти спортсмена за прізвищем.
- 3.Team name – Знайти спортсмена за командою.
- 4.Sport name – Знайти спортсмена за видом спорту.
- 5.Sportsman age from – Знайти спортсмена з віком більшим або однаковим від заданого.
- 6.Sportsman age to – Знайти спортсмена з віком меншим або однаковим від заданого.

4. На рисунку 2.4. зображено сортування спортсменів за різними критеріями.

Index

Sportsman name

Sportsman surname

Team name

Sport name

Sportsman age from 24

Sportsman age to 0

Search

Team	Name	Surname	Age	Nationally
Halkbank	Nimir	Abdel-Aziz	31	Dutch
Real Madrid	Karim	Benzema	35	France

Sorting

Sort by name (A-Z)

Sort by name (Z-A)

Sort by surname (A-Z)

Sort by surname (Z-A)

Sort by team name (A-Z)

Sort by team name (Z-A)

Sort by age (new first)

Sort by age (old first)

Sort by weight (heavy first)

Sort by weight (lungs first)

Інструкція

- 1.Скачати архів з програмою.
- 2.Розпакувати архів.
- 3.Відкрити програму у Visual Studio.
- 4.Після запуску проєкту в Visual Studio, натискаємо кнопку “Start”, після цього ми потрапляємо на головну сторінку програми.
- 5.Програма розділена на різні розділи такі як : Sports, Matches, Sportsmen, Stadiums, Teams, Trainers.
- 6.Користувач може переглянути будь-який розділ та застосувати фільтри пошуку за для зручного пошуку інформації

Висновок

У даній курсовій роботі був розроблений та реалізований функціональний довідник фанату з використанням мови програмування C# та платформи ASP.NET. Основною метою проєкту було створення веб-додатку, який надаватиме користувачам зручний інструмент для отримання інформації про їхніх улюблених спортсменів, спортивних команд.

Процес розробки включав аналіз вимог, вибір технологій (C# та ASP.NET), створення бази даних для зберігання інформації про об'єкти, реалізацію функціональності для додавання, редагування, видалення об'єктів, а також їхнього перегляду та пошуку.

Важливою частиною процесу була розробка бази даних для зберігання інформації про об'єкти. Використання технології Entity Framework дозволило створити зручну та ефективну структуру бази даних та забезпечити зручний доступ до інформації.

Процес розробки супроводжувався тестуванням для перевірки коректності роботи програми. Результатом є функціональний довідник, який забезпечує легкий доступ.

У результаті роботи було успішно реалізовано функціональний довідник фанату, який забезпечує зручний спосіб отримання інформації. Реалізація на платформі ASP.NET дозволяє забезпечити доступ до довідника з будь-якого пристрою з веб-браузером, що робить його дуже зручним для користувачів.

Вихідний код

DataBaseContext.cs

```
using FanGuide.Models;
using Microsoft.EntityFrameworkCore;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide
{
    public class DataBaseContext : DbContext
    {
        public DbSet<Match> Matches { get; set; }
        public DbSet<Sport> Sports { get; set; }
        public DbSet<Sportsman> Sportsmen { get; set; }
        public DbSet<Stadium> Stadiums { get; set; }
        public DbSet<Team> Teams { get; set; }
        public DbSet<Trainer> Trainers { get; set; }

        public DataBaseContext()
        {
            // Database.EnsureDeleted();
            Database.EnsureCreated();
        }

        protected override void OnConfiguring(DbContextOptionsBuilder
optionsBuilder)
        {
            optionsBuilder.UseSqlServer("Server=(localdb)\\mssqllocaldb;Database=FanGuide;Tru
sted_Connection=True;");
        }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            modelBuilder.Entity<Team>().HasMany(c => c.HomeMatches).WithOne(c =>
c.HomeTeam)
                .HasForeignKey(c =>
c.HomeTeamId).IsRequired().OnDelete(DeleteBehavior.ClientSetNull);
            modelBuilder.Entity<Team>().HasMany(c => c.VisitorMatches).WithOne(c
=> c.VisitorTeam)
                .HasForeignKey(c =>
c.VisitorTeamId).IsRequired().OnDelete(DeleteBehavior.ClientSetNull);
            modelBuilder.Entity<Team>().HasMany(c => c.HomeMatches).WithOne(c =>
c.HomeTeam)
                .HasForeignKey(c =>
c.HomeTeamId).IsRequired().OnDelete(DeleteBehavior.ClientSetNull);
            modelBuilder.Entity<Team>().HasMany(c => c.VisitorMatches).WithOne(c
=> c.VisitorTeam)
                .HasForeignKey(c =>
c.VisitorTeamId).IsRequired().OnDelete(DeleteBehavior.ClientSetNull);
        }
    }
}
```

Program.cs

```
using Microsoft.AspNetCore.Hosting;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.Hosting;
using Microsoft.Extensions.Logging;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide
{
    public class Program
    {
        public static void Main(string[] args)
        {
            CreateHostBuilder(args).Build().Run();
        }

        public static IHostBuilder CreateHostBuilder(string[] args) =>
            Host.CreateDefaultBuilder(args)
                .ConfigureWebHostDefaults(webBuilder =>
                {
                    webBuilder.UseStartup<Startup>();
                })
        }
    }
}
```

Startup.cs

```
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.HttpsPolicy;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide
{
    public class Startup
    {
        public Startup(IConfiguration configuration)
        {
            Configuration = configuration;
        }

        public IConfiguration Configuration { get; }

        // This method gets called by the runtime. Use this method to add
        // services to the container.
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddDbContext<DataBaseContext>();
            services.AddSession();
        }
    }
}
```

```

        services.AddControllersWithViews();
    }

    // This method gets called by the runtime. Use this method to configure
    the HTTP request pipeline.
    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
        }
        else
        {
            app.UseExceptionHandler("/Home/Error");
            // The default HSTS value is 30 days. You may want to change this
            for production scenarios, see https://aka.ms/aspnetcore-hsts.
            app.UseHsts();
        }
        app.UseHttpsRedirection();
        app.UseStaticFiles();

        app.UseRouting();

        app.UseSession();

        app.UseAuthorization();

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

```

Match.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide.Models
{
    public class Match
    {
        public int Id { get; set; }
        public DateTime Date { get; set; }
        public Team HomeTeam { get; set; }
        [Required(ErrorMessage = "Type something")]
        public int HomeTeamId { get; set; }
        public Team VisitorTeam { get; set; }
        [Required(ErrorMessage = "Type something")]
        public int VisitorTeamId { get; set; }
        [Range(0, 1000, ErrorMessage = "Invalid score")]
        public int HomeTeamScore { get; set; }
        [Range(0, 1000, ErrorMessage = "Invalid score")]
        public int VisitorTeamScore { get; set; }
    }
}

```

```
    }  
}
```

Sport.cs

```
using System;  
using System.Collections.Generic;  
using System.ComponentModel.DataAnnotations;  
using System.Linq;  
using System.Threading.Tasks;  
  
namespace FanGuide.Models  
{  
    public class Sport  
    {  
        public int Id { get; set; }  
        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length  
must be between 3 and 50 characters")]  
        public string Name { get; set; }  
    }  
}
```

Sportsman.cs

```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Threading.Tasks;  
  
namespace FanGuide.Models  
{  
    public class Sportsman : UserBase  
    {  
        public int Weight { get; set; }  
        public int Height { get; set; }  
        public string Nationally { get; set; }  
        public string Records { get; set; }  
        public string Description { get; set; }  
        public Team Team { get; set; }  
        public int TeamId { get; set; }  
    }  
}
```

Stadium.cs

```
using System;  
using System.Collections.Generic;  
using System.ComponentModel.DataAnnotations;  
using System.Linq;  
using System.Threading.Tasks;  
  
namespace FanGuide.Models  
{  
    public class Stadium  
    {  
        public int Id { get; set; }  
        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length  
must be between 3 and 50 characters")]  
        public string Name { get; set; }  
    }  
}
```



```

        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length
must be between 3 and 50 characters")]
        public string City { get; set; }
        public int StadiumCapacity { get; set; }
        public Team HomeTeam { get; set; }
        public int HomeTeamId { get; set; }
    }
}

```

Team.cs

```

using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide.Models
{
    public class Team
    {
        public int Id { get; set; }
        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length
must be between 3 and 50 characters")]
        public string Name { get; set; }
        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length
must be between 3 and 50 characters")]
        public string City { get; set; }
        public DateTime CreateDate { get; set; }
        public List<Trainer> Trainers { get; set; } = new();
        public List<Sportsman> Sportsmen { get; set; } = new();
        public List<Match> HomeMatches { get; set; } = new();
        public List<Match> VisitorMatches { get; set; } = new();
        public Sport Sport { get; set; }
        public int SportId { get; set; }
    }
}

```

Trainer.cs

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide.Models
{
    public class Trainer : UserBase
    {
        public Team Team { get; set; }
        public int TeamId { get; set; }
    }
}

```

UserBase.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Threading.Tasks;

namespace FanGuide.Models
{
    public abstract class UserBase
    {
        public int Id { get; set; }
        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length
must be between 3 and 50 characters")]
        public string Name { get; set; }
        [StringLength(50, MinimumLength = 3, ErrorMessage = "The string length
must be between 3 and 50 characters")]
        public string Surname { get; set; }
        [Range(1, 110, ErrorMessage = "Invalid age")]
        public int Age { get; set; }
    }
}
```

Джерела

1. Metanit [Електронний ресурс] – Режим доступу до ресурсу:
<https://metanit.com/> – 29.04.2022 р.
2. Microsoft docs [Електронний ресурс] – Режим доступу до ресурсу:
<https://docs.microsoft.com/ru-ru/aspnet/overview> - 01.05.2022 р.
3. Троелсен, Э, Джепикс, Ф. Язык программирования C# 7 и платформы .NET и .NET Core, 8-е изд. : Пер. с англ. СПб. : ООО “Диалектика”, 2018 — 1328 с. 978-5-6040723-1-8.
4. C# / BestProg. Програмування: теорія та практика. URL:
https://www.bestprog.net/uk/sitemap_ua/c-3/