Національний університет “Острозька академія”

Кафедра економіко-математичного моделювання та інформаційних технологій

**Фінальний проект з курсу “Програмування на C#” (2023/2)**

Виконав:

студент групи КН-222

Жарчинський Маріан

Зміст

[**Мета роботи** 1](#_Toc153485019)

[**Завдання** 1](#_Toc153485020)

[**Опис реалізації** 1](#_Toc153485021)

[**Лістинги сутностей:** 1](#_Toc153485022)

[**Клас Тренер:** 1](#_Toc153485023)

[**Клас Контракт тренера:** 2](#_Toc153485024)

[} 2](#_Toc153485025)

[**Клас Футбольна команда:** 2](#_Toc153485026)

[} 2](#_Toc153485027)

[**Клас Гравець:** 2](#_Toc153485028)

[} 3](#_Toc153485029)

[**Клас Контракт гравця :** 3](#_Toc153485030)

[} 3](#_Toc153485031)

[**Клас Позиція:** 3](#_Toc153485032)

[} 3](#_Toc153485033)

[**Клас Трансфер гравця:** 3](#_Toc153485034)

[} 4](#_Toc153485035)

[**Клас Користувачі:** 4](#_Toc153485036)

[} 4](#_Toc153485037)

[**Лістинг репозиторія для роботи з даними та контексту:** 4](#_Toc153485038)

[**Репозиторій для Тренера:** 4](#_Toc153485039)

[**Лістинг контексту:** 5](#_Toc153485040)

[} 5](#_Toc153485041)

[**Опис графічного інтерфейсу:** 5](#_Toc153485042)

[**Висновки** 6](#_Toc153485043)

**Мета роботи**

Мета лабораторної роботи полягає у закріпленні знань у роботі з базами даних у .NET, використанні патерну Repository та ORM Entity Framework та трирівневої архітектури рішення.

Під час виконання завдання студенти повинні самостійно написати програму, підготувати звіт про хід виконання роботи за встановленою структурою та згідно вимог до оформлення, вчасно здати роботу та захистити її.

# **Завдання**

Темою мого фінального проекту є додаток для служби таксі, де клієнт зможе замовити таксі на певний маршрут. Для виконання завдання було створено 4 форми для авторизації та три форми, кожна для одного типу користувачів. Користувачами є адміністратор, що може редагувати всі дані в базі даних, клієнт, що може замовити таксі, скасувати замовлення та поставити оцінку, водій, що може відхилити виклик, або прийняти.

# **Опис реалізації**

## **Лістинги сутностей:**

### **Клас Тренер:**

public class Coach

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public string FullName { get; set; }

public int Age { get; set; }

public int WonTrofies { get; set; }

public int CurrentMatchesAmount { get; set; }

public int WonMatches { get; set; }

public int LoosedMatches { get; set; }

public int DrawnMatches { get; set; }

public virtual ICollection<CoachContract> CoachContracts { get; set; }

public virtual ICollection<FootballTeam> FootballTeams { get; set; }

public override string ToString()

{

return

$" Full name: {FullName}," +

$" Age: {Age}," +

$" Won trofies: {WonTrofies}," +

$" Current matches amount: {CurrentMatchesAmount}," +

$" Won matches: {WonMatches}," +

$" Loosed matches: {LoosedMatches}," +

$" Drawn matches: {DrawnMatches},";

}

public string Name()

{

return $"{FullName}";

}}

### **Клас Контракт тренера:**

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public virtual Coach Coach { get; set; }

public DateTime StartDate { get; set; }

public DateTime EndDate { get; set; }

public int Salary { get; set; }

public override string ToString()

{

return

$" Coach name: {Coach.Name()}," +

$" Start date: {StartDate.ToString("yyyy-MM-dd")}," +

$" End date: {EndDate.ToString("yyyy-MM-dd")}," +

$" Salary: {Salary}";

}

### }

### **Клас Футбольна команда:**

public class FootballTeam

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public string FullName { get; set; }

public DateTime CreationDate { get; set; }

public string Country { get; set; }

public virtual Coach Coach { get; set; }

public virtual ICollection<PlayerContract> PlayerContracts { get; set; }

public override string ToString()

{

return

$" Full name: {FullName}," +

$" Creation date: {CreationDate.ToString("yyyy-MM-dd")}," +

$" Country team: {Country}," +

$" Coach name: {Coach.Name()}";

}

public string Name()

{

return $"{FullName}";

}

### }

### **Клас Гравець:**

public class Player

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public string FullName { get; set; }

public int Age { get; set; }

public bool IsCaptain { get; set; }

public string WorkingLeg { get; set; }

public int Goals { get; set; }

public int Assists { get; set; }

public virtual ICollection<PlayerContract> PlayerContracts { get; set; }

public virtual ICollection<Position> Positions { get; set; }

public virtual ICollection<PlayerTransfer> PlayerTransfers { get; set; }

public override string ToString()

{

return

$" Full name: {FullName}," +

$" Age: {Age}," +

$" Is captain: {IsCaptain}," +

$" Working leg: {WorkingLeg}," +

$" Goals scored: {Goals}," +

$" Assists scored: {Assists}";

}

public string Name()

{

return $"{FullName}";

}

### }

### **Клас Контракт гравця :**

public class PlayerContract

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public virtual Player Player { get; set; }

public virtual FootballTeam Team { get; set; }

public DateTime StartDate { get; set; }

public DateTime EndDate { get; set; }

public int Salary { get; set; }

public override string ToString()

{

return

$" Player name: {Player.Name()}," +

$" Football Team: {Team.Name()}," +

$" Start date: {StartDate.ToString("yyyy-MM-dd")}," +

$" End date: {EndDate.ToString("yyyy-MM-dd")}," +

$" Salary: {Salary},";

}

### }

### **Клас Позиція:**

public class Position

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Player> Players { get; set; }

public override string ToString()

{

return

$" Position name: {Name},";

}

### }

### **Клас Трансфер гравця:**

public class PlayerTransfer

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public decimal Price { get; set; }

public string TeamFrom { get; set; }

public string TeamTo { get; set; }

public virtual Player Player { get; set; }

public bool? FullFledged { get; set; }

public bool? Rent { get; set; }

public DateTime? DateRentStart { get; set; }

public DateTime? DateRentEnd { get; set; }

public override string ToString()

{

return

$" Price: {Price}," +

$" Team from: {TeamFrom}," +

$" Team to: {TeamTo}," +

$" Player name: {Player?.Name() ?? "N/A"}," +

$" Is it full fledged: {FullFledged ?? false}," +

$" Is it rent: {Rent ?? false}," +

$" Date contract start: {DateRentStart?.ToString("yyyy-MM-dd") ?? "N/A"}," +

$" Date contract end: {DateRentEnd?.ToString("yyyy-MM-dd") ?? "N/A"},";

}

### }

### **Клас Користувачі:**

public class User

{

[Key]

[DatabaseGenerated(DatabaseGeneratedOption.Identity)]

public int Id { get; set; }

public string Login { get; set; }

public string Password { get; set; }

public bool IsAdmin { get; set; }

## }

## **Лістинг репозиторія для роботи з даними та контексту:**

Лістинг лише одного репозиторія, бо в них ідентичний код просто типи сутностей різні.

### **Репозиторій для Тренера:**

public class CoachContractRep

{

public IEnumerable<CoachContract> GetAll()

{

using (var ctx = new FootballTeamContext())

{

return ctx.CoachContracts.Include(x => x.Coach).ToList();

}

}

public int Add(CoachContract coachContract)

{

using (var ctx = new FootballTeamContext())

{

var obj = ctx.CoachContracts.Add(coachContract);

ctx.SaveChanges();

return obj.Entity.Id;

}

}}

## **Лістинг контексту:**

public class FootballTeamContext: DbContext

{

public DbSet<Coach> Coaches => Set<Coach>();

public DbSet<CoachContract> CoachContracts => Set<CoachContract>();

public DbSet<FootballTeam> FootballTeams => Set<FootballTeam>();

public DbSet<Player> Players => Set<Player>();

public DbSet<PlayerContract> PlayerContracts => Set<PlayerContract>();

public DbSet<PlayerTransfer> PlayerTransfers => Set<PlayerTransfer>();

public DbSet<Position> Positions => Set<Position>();

public DbSet<User> Users => Set<User>();

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

var connectionString = "Server=.;Database=Football\_Team;Integrated Security=True;Encrypt=True;TrustServerCertificate=True";

optionsBuilder.UseSqlServer(connectionString);

base.OnConfiguring(optionsBuilder);

}

# }

# **Опис графічного інтерфейсу:**



Рис. 1. Скріншот форми логіну користувача

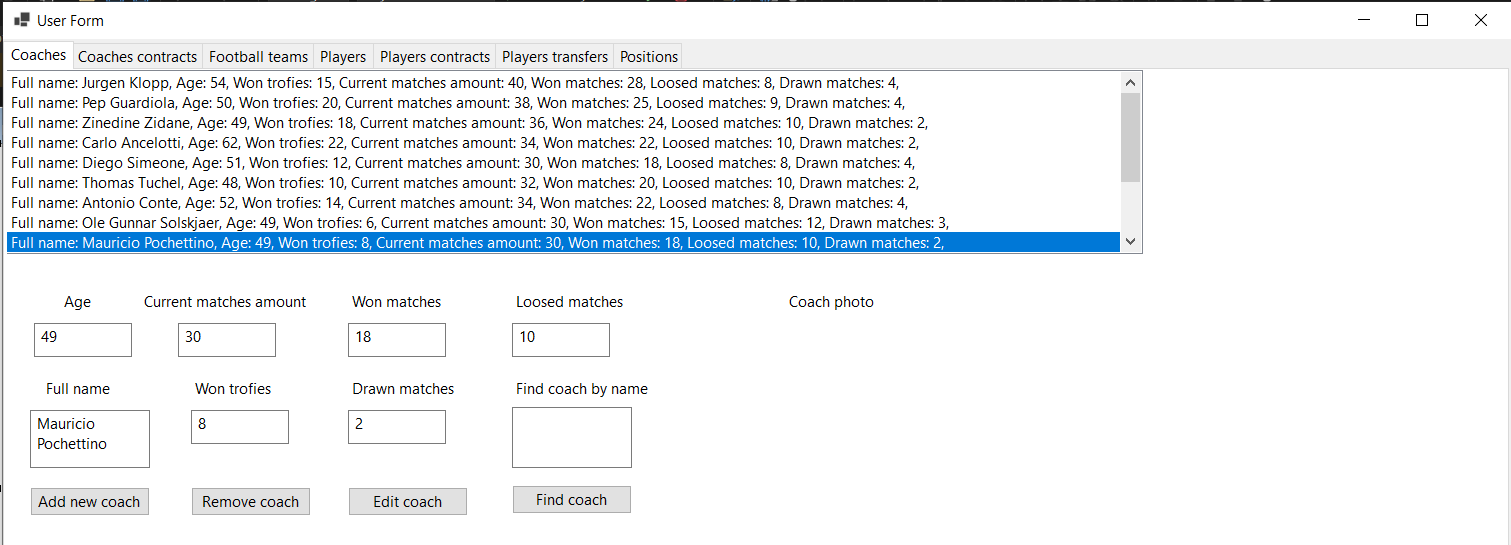


Рис. 2 Скріншот форми взаємодії користувача з програмою

Графічний інтерфейс містить 7 вкладок для роботи з кожною сутністю . Можна змінювати кожний атрибут сутності , додавати нову сутність та видаляти. Також можна вводити певну назву або ім’я сутності – тоді буде виводиться лише певні сутності , які є по цьому фільтру.

# **Висновки**

У ході виконання лабораторної роботи була розроблена функціональність додавання нових сутностей у системі керування футбольною командою. Для цього було використано мову програмування C# та інструменти розробки .NET Framework. Основна мета лабораторної роботи полягала в реалізації взаємодії із базою даних для збереження нових сутностей у футбольну базу даних, введення необхідної інформації з форми та відображення повідомлень про результати дій користувача. Крім того, була використана обробка винятків для відображення повідомлень про можливі помилки.