# Exercise: Introduction to .NET Core and EF Core

This document defines the **in-class exercises** assignments for the ["C# Web Development Basics" course @ Software University](https://softuni.bg/courses/csharp-web-development-basics). You can submit your code in the course page.

# Student System

## Create models

Your task is to create a database for the **Student System**, using the **Entity Framework Code First** approach. Separate the **models**, **data** and **client** into **different layers** (projects). Model the following tables:

* **Students**: id, name, phone number (**optional**), registration date, birthday (**optional**)
* **Courses**: id, name, description (**optional**), start date, end date, price
* **Resources**: id, name, type of resource (**video / presentation / document / other**), URL
* **Homework**: id, content, content-type (**application/pdf/zip**), submission date

Table relations:

* **Students** can be in **many course**s
* **Courses** can have **many students**
* **Courses** can have **many resources**
* **One course** can have **many homework submissions**
* **Homework submissions** have a **student**



Add **navigational properties** in all models to simplify navigation. Annotate the data models with the appropriate **attributes** and validations and **enable code first migrations**.

## Working with the Database

Write a console application that works with the EF data layer and performs some CRUD operations.

### Tasks

1. Lists **all students** and their **homework submissions**. Print only their **names** and for each homework - **content** and **content-type**.
2. List **all courses** with their corresponding **resources**. Print the **course name** and **description** and everything for each **resource**. Order the courses by start date (ascending), then by end date (descending).
3. List **all courses** with **more than 5 resources**. Order them by **resources count** (descending), then by **start date** (descending). Print only the **course name** and the **resource count**.
4. \*List all **courses** which were active on a **given date** (choose the date depending on the data seeded to ensure there are results), and for each course count the **number of students enrolled**.   
   Print the **course name**, **start** and **end date**, **course duration** (difference between end and start date) and **number of students enrolled**. Order the results by the **number of students** enrolled (in descending order), then by **duration** (descending).
5. For each student, calculate the **number of courses** he/she has enrolled in, the **total price** of these courses and the **average price per course** for the student.  
   Print the **student name**, **number of courses**, **total price** and **average price**. Order the results by **total price** (descending), then by **number of courses** (descending) and then by the **student's name** (ascending).

# Football Betting

## Database Models

Your task is to create a database for the **Football Bookmaker System**, using the **Entity Framework Code First** approach. Model the following tables:

* **Teams** – Id, Name, Logo, 3 letter Initials (JUV, LIV, ARS…), Primary Kit Color, Secondary Kit Color, Town, Budget
* **Colors** – Id, Name
* **Towns** – Id, Name, Country
* **Countries** – Id (3 letters – for example BUL, USA, GER, FRA, ITA…), Name, Continent
* **Continents** – Id, Name
* **Players** – Id, Name, Squad Number, Team, Position, Is Currently Injured
* **Position** – Id (2 letters – GK, DF, MF, FW…), position description (for example – goal keeper, defender…)
* **PlayerStatistics** – Game, Player, Scored Goals, Player Assists, Played Minutes During Game, (PK = Game + Player)
* **Games** – Id, Home Team, Away Team, Home Goals, Away Goals, Date and Time of Game, Home team Win bet rate, Away Team Win Bet Rate, Draw Game Bet Rate, Round, Competition)
* **Rounds** – Id, Name (for example Groups, League, 1/8 Final, 1/4 Final, Semi-Final, Final…)
* **Competitions** – Id, Name, Type (local, national, international)
* **CompetitionTypes** –Id, Name
* **BetGame** – Game, Bet, Result Prediction (PK = Game + Bet)
* **Bets** – Id, Bet Money, Date and Time of Bet, User
* **ResultPrediction** – Id, Prediction (possible values - Home Team Win, Draw Game, Away Team Win)
* **Users** – Id, Username, Password, Email, Full Name, Balance

## Table relationships

* Team has one primary kit color and one secondary kit color
* Team resident in one town
* Each town can host several teams
* Town can be placed in one country and a country can have many towns
* Country can be placed in several continents and a continent can have many countries
* Player can play for one team and one team can have many players that play for it
* Player can play at one position and one position can be played by many players
* Player can play in many games and in each game, many players take part
* Additionally, for each player for given game is kept statistics such as scored goals, goal assists and minutes played during given game
* A game can be played in one round and in one round many games can be played
* A game can be played in one competition and in one competition many games can be played
* On a game, many bets can be placed and one bet can be on several games
* Each bet for given game must have prediction result
* A bet can be placed by only one user and one user can place many bets

Add **navigational properties** in all models to simplify navigation. Annotate the data models with the appropriate **attributes** and validations and **enable code first migrations**.

### Hint - Database Schema

