Team Management API

# How to run the application.

To start, use the below command to create or update the database schema on your local PC.   
  


# Assumptions:

* Demonstrating OAuth implementation is not needed for this assignment;
* Only basic player data is needed;
* No need for tracking of player positions.

# Reason for specific designs and interfaces:

I decided to implement the Repository-Service Pattern. The benefits of this are:

* It centralizes data logic, business logic and service logic;
* It gives a substitution point for the unit tests;
* It provides flexible architecture;
* It allows for easy modification of the data access logic or business access logic as you don’t need to change the repository logic.

# Obstacles encountered and how I solved them.

It’s been a while since I last had to setup a project and implement a repository-service design pattern from scratch.

I looked up a few examples on the web to refresh my memory and help me get started and continued my implementation from there. See resources below.

# List resources used and relevant references.

Repository-service design pattern:

<https://www.programmingwithwolfgang.com/repository-pattern-net-core/>

EF Core Migration references:

<https://www.entityframeworktutorial.net/efcore/entity-framework-core-migration.aspx>

EF Core Relationship references:

<https://docs.microsoft.com/en-us/ef/core/modeling/relationships?tabs=fluent-api%2Cfluent-api-simple-key%2Csimple-key>

# Time taken to complete the assignment.

It took me roughly 8 hours to complete the assignment.

# If you had more time, what would you do differently? Also, what would you have added additionally?

I initially didn’t intend on doing a proper implementation of DTO’s and Automapper. I later decided to implement this on the project. In hindsight, this decision should have been made beforehand.

Had I had more time, I would have liked to properly secure the API by implementing OAuth.

I also intended on my implementation not to be specific to rugby unions & their structures. That’s why I used naming conventions such as: “business unit”, “employee” and “employee type”. This would facilitate expansion options such as:

* To add functionality to manage fixtures;
* Track scores;
* Manage player positions;
* Use of the application by a variety of team-sport organisations.