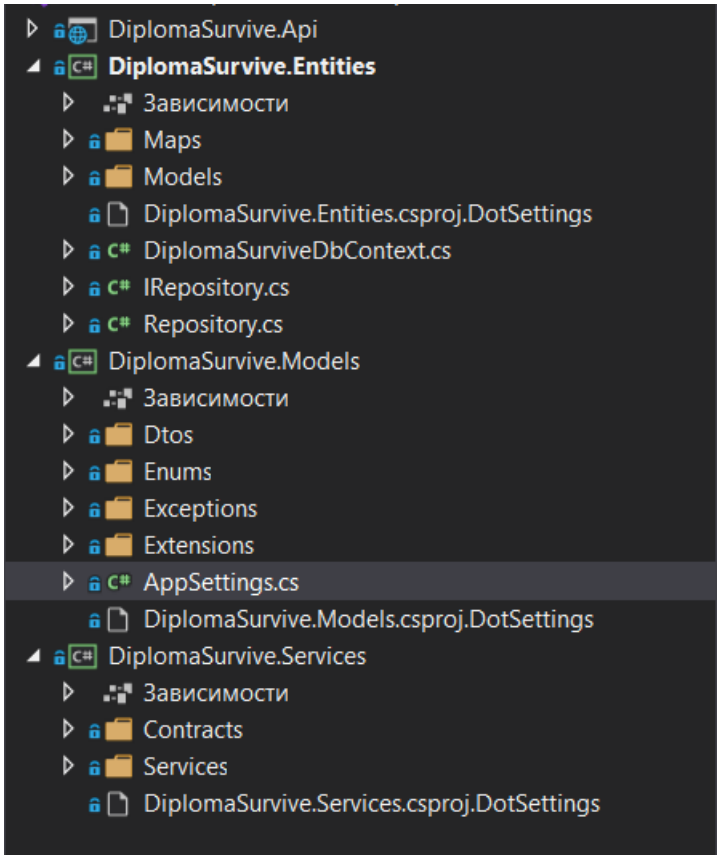
Серверна частина.

Була необхідність реалізувати серверну частину для даної гри для створення «лідербордів», синхронізації набраних балів та керування змаганнями. Для описаної задачі був вибраний такий стек технологій: PostgreSQL і .NET Core 3.0. Завдяки кроссплатформеності сервер був розгорнутий на убунтовій машині разом з базою даних.

Логіка лідербордів працює таким чином: налаштовується подія, що приурочена до змагання. До прикладу, Різдво – приурочена подія, визначається дата початку та дата кінця – проміжок часу, що відповідає цьому турніру, а також налаштовуються подарунки за відповідні місця. Коли наступає дата фінішу, то відбувається фіналізація: гравці сортуються за їхніми балами і переможці отримують свої подарунки.

Також, при старті сервера запускається «шедулер», який дістає з бази даних ці налаштування і таким чином отримує інформацію, коли саме потрібно виконати фіналізацію балів (нагородження), до того ж, завдяки шедулеру, усі лідерборди, що старші за 1 місяць очищуються з бази даних. Для шедулера була використана стороння бібліотека – FluentScheduler. Окрім цього, запрограмовано, що з певною періодичністю налаштування перевіряються на нові, якщо такі знайдені, то старі перезаписуються знайденими, усі заплановані джоби (а саме на фіналізацію та очистку) відміняються та створюються нові.

Архітектура побудована таким чином, що у проекті знаходиться декілька бібліотек та вхідна точка:



Структура проекту.

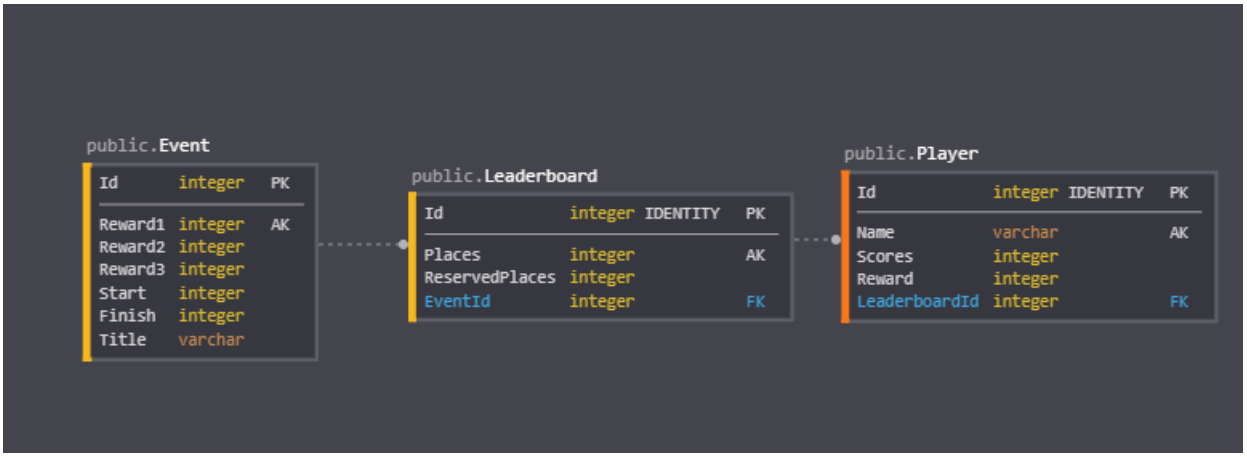
DiplomaSurvival.Api – це точка входу, тут усі контролери та «зв'язучі зі світом» для додатку, через цей проект приймаються запити та надається на них відповідь.

DiplomaSurvival.Services – цей проект зроблений для сервісів – прошарок між базою даних та контролерами.

DiplomaSurvival.Models – використовується як бібліотека моделей - різних data transfer object – ів, перечислень та констант проекту.

DiplomaSurvival.Entities - тут знаходиться опис і маппинг таблиць. Для маппингу обраний – fluent mapping.

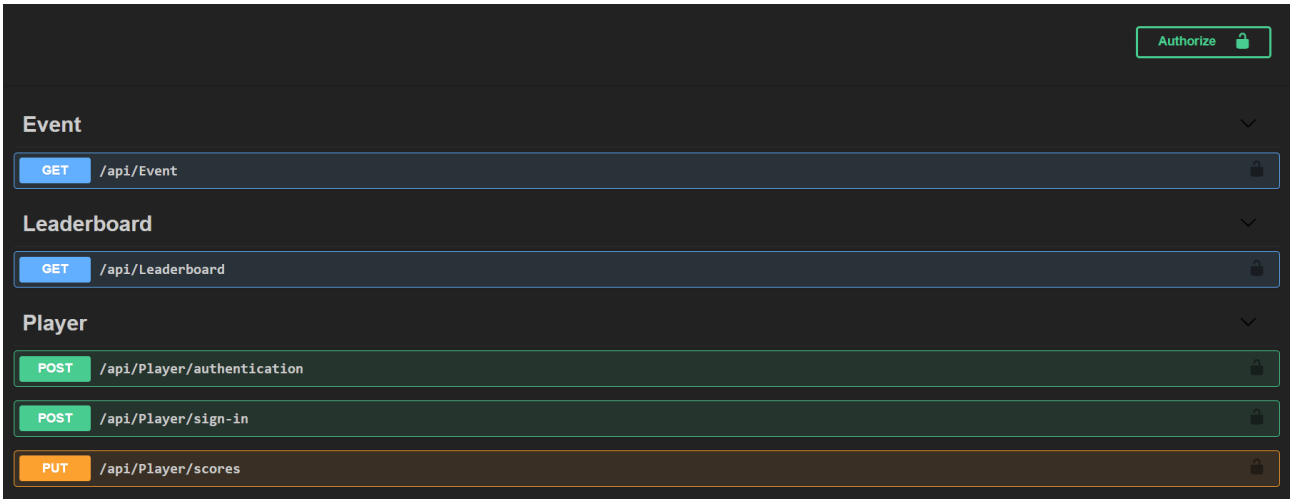
Структура бази даних досить нескладна: усього 3 таблиці, що зберігають у собі інформацію про події (нагорода за перше місце, нагорода за друге місце, нагорода за третє місце, дата початку, дата кінця та заголовок), про лідерборди (кількість виділений місць, кількість зайнятих місць та посилання на приурочену подію через зовнішній ключ) та про гравців (ім'я, бали, нагорода та посилання на лідерборд через зовнішній ключ).



Структура бази даних.

Для швидкодії код написаний з урахуванням асинхронності та паралелізму.

Для взаємодією з API за допомогою swagger продемонстровано такі методи:



Методи у API.

Event:

1. GET: api/event – запит для отримання усіх подій на даний момент.

Leaderboard:

1. GET: api/leaderboard – запит для отримання лідерборду гравця (розпізнавання гравця відбувається за токеном).

Player:

1. POST: api/player/authentication – запит для аутенфікації гравця (отримання токену).
2. POST: api/player/sign-in – запит для реєстрації гравця.
3. PUT: api/player/scores – запит для синхронізації балів гравця з сервером.