PUBG Stats Backend

# Overview

As a good player of PUGB, you want to know if you belong to the elite of PUBG players. So you will develop a backend in order to retrieve the leaderboard of the different game modes and be able to monitor the lifetime stats of the best players in the game.

To do this, you will need:

1. Register to the developer portal of PUGB in order to obtain a free API key. (<https://developer.pubg.com/apps?locale=en>)
2. Docker compose in order to run a multi-container backend application.
3. Use Redis or similar to cache requests.
4. Use NoSQL database.
5. All backend logic developed in ASP.Net core.

The general things that will be evaluated are:

* Best coding practices: dependency injection, SOLID principles …
* Unit testing skills: Using Nunit or similar, Moq …
* Use of Environment variables for connection strings instead of hardcoded ones.
* Generate a swagger file for the ASP.Net core API backend, that will be available in /swagger endpoint.
* Good Rate Limit management, proactive instead of reactive.
* High performance API.
* Git usage



# Components

## ASP.Net Core backend

The backend should process the less information as possible and **will not be the responsible to make the API calls to the PUGB api**.

All of the following endpoints will have:

* 4 hours CACHE
* Token based authentication via HTTP header: Authentication

Endpoints list:

* /swagger
* /leaderboard/{gamemode}
  + GET’s the top 100 players from a given game mode.
* /player/stats/{accountid}
  + GET’s the lifetime stats of the specified player by account id.
* /player/stats/by-name/{player-name}
  + GET’s the lifetime stats of the specified player by name
  + It should return information even when there is not available in the database but the player exists in the PUGB API.

**Note:** For the /player/stats/… endpoints if the stats wasn’t updated in 5 hours it should update them again.

## Leaderboard stats updater service

A long running backend service that will update the leaderboard and the lifetime stats of the first 100 player in the leaderboard every 4 hours.

* This service is the responsible to make all the api calls to the official PUGB API, and update a no SQL database with the requested information.
* This service should manage the API rate limitation in a proactive manner.
* This service can receive requests from only the ASP.NET core API backend.

# Deliverable

The code must be provided in an accessible github/bitbucket/gitlab repository. This must contain a solution with the different components and the docker compose file used to validate the application.

The repository must have a README file to explain all the components and the necessary steps to run the backend.