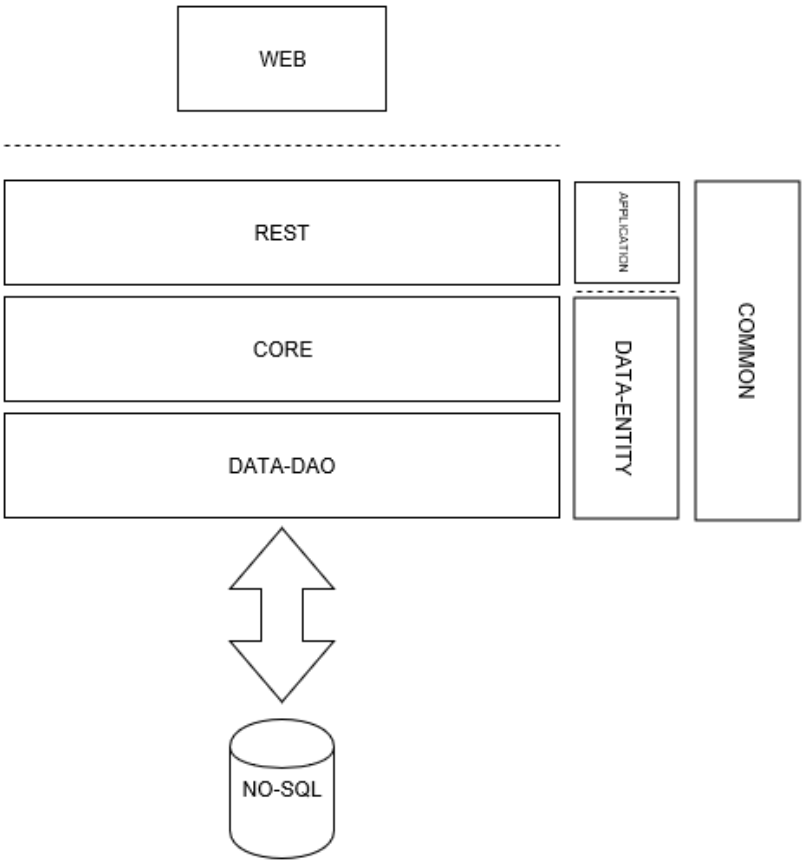
Arquitectura

Se trata de una aplicación basada en front y back, donde el back expone un servicio que es “atacado” por los distintos clientes, en este caso el front o en el futuro dispositivos móviles.



Frontend

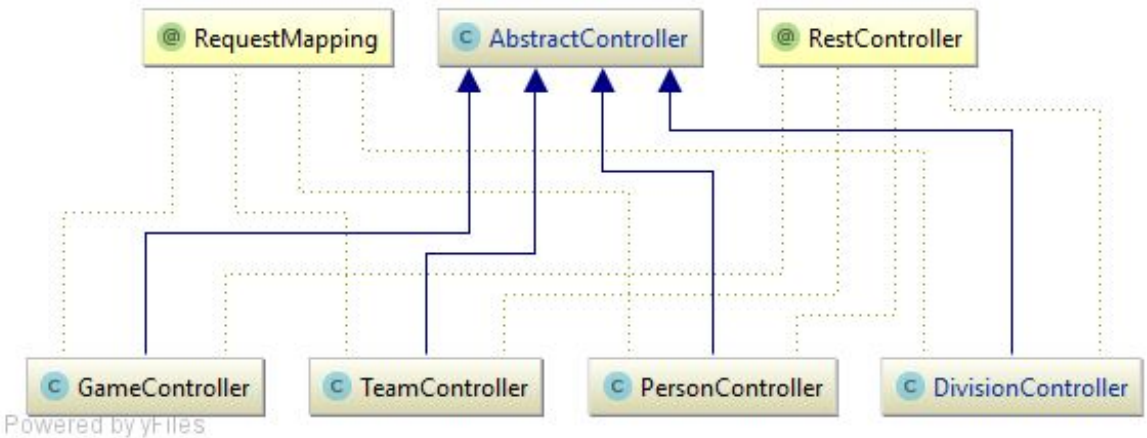
Por decidir

Backend

Desarrollada en java, versión 8, puesto que en el momento de la concepción esta era la versión más actual.

REST

Es una capa basada en Spring MVC proporciona el acceso a la aplicación a través de servicios REST.
Añade además tecnologías como Swagger que permiten la generación de un cliente html accesible vía web, generado específicamente para los contratos expuestos por los diferentes recursos REST.
Existen unos controladores donde se definen los diferentes recursos

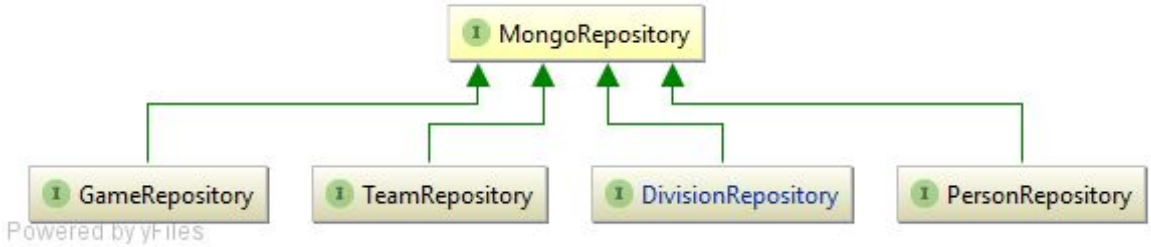


Por otro lado tenemos DTOs que se convierten desde las entidades, para esto nos apoyamos en la librería ModelMapper con métodos que se exponen desde la clase AbstractController.

Capa DAO

Esta es la capa de acceso a datos, utiliza el framework Spring Data Mongo Db. Este framework nos abstrae las operaciones boiler plate con la base de datos, la generación de consultas para la BD subyacente y entre otras cosas el map de tipos de datos que utiliza la BD subyacente y los tipos del lenguaje (java 8).

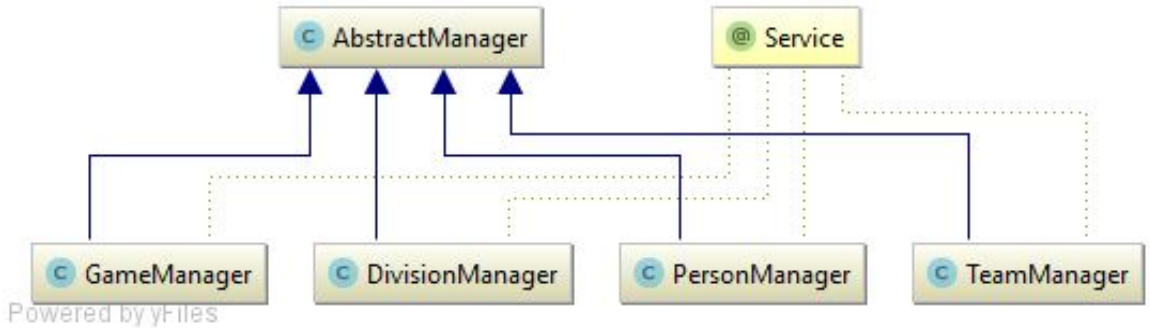
Los DAOs o Repositories son interfaces que heredan de la interfaz MongoRepository de Spring Data Mongo Db



Capa Core

Esta capa contiene la lógica de negocio de la aplicación comparte las entidades con la capa dao.

Existe unos Managers que son usados para facilitar el acceso a la capa de datos y que contendrán la lógica de las operaciones con entidades de base de datos.



Module Common

Ofrece servicios, tipos de datos, test globales a toda la aplicación Podemos encontrar dos partes diferenciados:

- Capa Common Tests.
- Capa de Dominio común.

Common tests

Esta capa tiene los test generales de todo el backend. Se utiliza el framework de Spock para realizar los test y el framework de mockito para realizar mocks principalmente métodos estáticos e instanciaciones de objetos que necesite el test.

Common Domain

Se definen métodos de utilidad y tipos de datos como enumerables etc para todo el backend.

Módule Data Entity

Define los objetos de la capa de negocio que serán compartidos en todas las capas. Estos objetos serán trasvasados a los DTO en la capa Rest en última instancia.

Otras tecnologías

Aspectos

Son usados para el loguear información de ciertos métodos a nivel global o definir implementaciones de interfaces, que se pueden. En la rama itd_mixing_aspect se utilizan para emular la “multiherencia” con versiones anteriores a java 8.

Mongeez

https://github.com/mongeez/mongeez
Se usa para cargar con datos la base de datos mongo.

Gestor de dependencias

El proyecto ha sido concebido con maven, pero se espera migrar a gradle por su versatilidad.

Maven: gestor de dependencias, usado para gestionar las librerías de la aplicación. Es en general un xml, que se gestiona desde un archivo llamando pom.xml. Cada pom.xml define un nuevo módulo, por lo que se usa también para ordenar la arquitectura de la aplicación.

Gradle: es igual que maven, pero en lenguaje groovy y mucho más versátil, sólo limitado por el propio groovy.