FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

Departamento de Engenharia Informática

Sistemas Distribuídos

Projeto 2

2021/2022 - 2º Semestre

Web Framework:

scoreDEI

Relatório realizado por:

- Alexandre Gameiro Leopoldo – 2019219929
- Luís Miguel Gomes Batista – 2019214869

Índice

1 Introdução

O projeto 2 da disciplina de Sistemas Distribuídos consiste na elaboração de uma Web Framework capaz de fornecer funcionalidades a diversos utilizadores, tais como funcionalidades de gestão administrativa.

A aplicação tem como objetivo possibilitar a consulta de resultados desportivos em tempo real. Para isto, os administradores criam jogos e os utilizadores autenticados têm a possibilidade de registar eventos. Para além disto, é possível a um cliente não registado aceder às páginas informativas, isto é, lista de jogos e estatísticas. É também possível obter informações relativas às equipas e jogadores pertencentes a um jogo.

A realização deste trabalho foi possibilitada através dos vários conceitos lecionados ao longo das aulas teóricas e práticas, tendo sido aplicados conceitos como utilização de tokens de autenticação e hashing de passwords.

Com a execução deste trabalho foi possível aos elementos do grupo, tal como pretendido, aprender e consolidar conceitos de sistemas Web; programação de Base de dados através da Java Persistence Application Programming Interface (JPA); e integração com serviços externos através da tecnologia REST, no caso, com o serviço API Sports.

2 Arquitetura

A plataforma *scoreDEI* utiliza uma arquitetura baseada em *Frontend* e *Backend*. O *Frontend* está relacionado com as diferentes páginas HTML que o utilizador tem possibilidade de interagir. Estas interações levam a parte *Frontend* a enviar informações ao *Backend* de modo a serem processadas e obter uma resposta consoante o resultado.

Para a concretização do *Frontend* é utilizado o mecanismo de *templates* Thymeleaf e as linguagens de programação HTML, CSS e JavaScript.

Para o *Backend* foi utilizada a *framework* Spring Boot, uma base de dados PostgreSQL e a linguagem de programação Java.

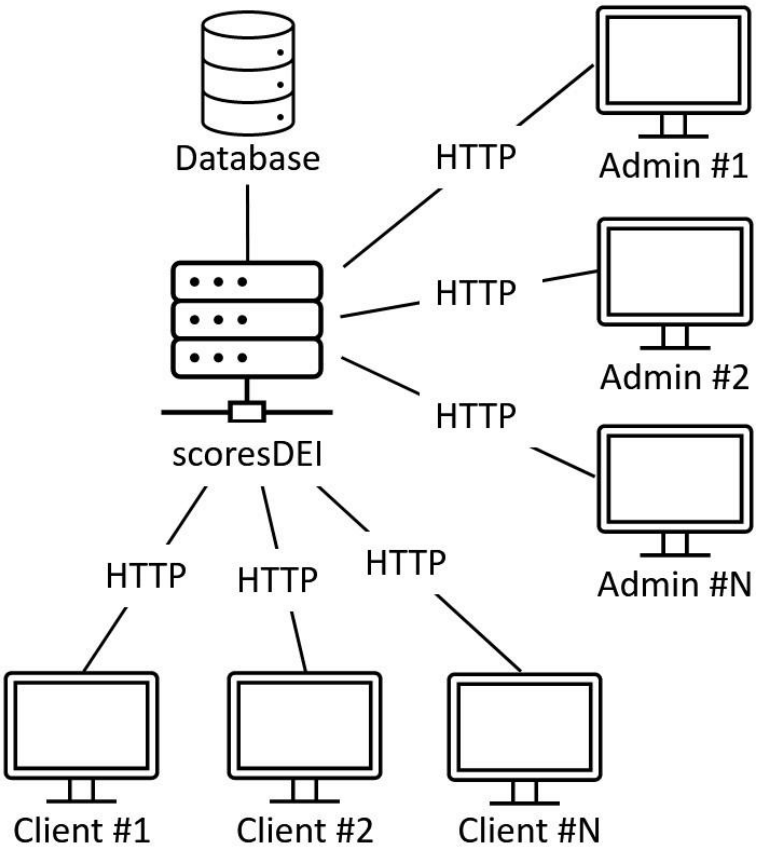

Figura 1 - Arquitetura da Web Framework scoreDei

A comunicação Cliente-Servidor é realizada através de HTTP, sendo que o servidor comunica posteriormente com a base de dados para a obtenção de dados.

Outro aspeto a ter em conta é a existência de vistas específicas para diferentes tipos de utilizadores, podendo um utilizador não estar registado; estar autenticado; ou ser um administrador, sendo esta a *role* máxima.

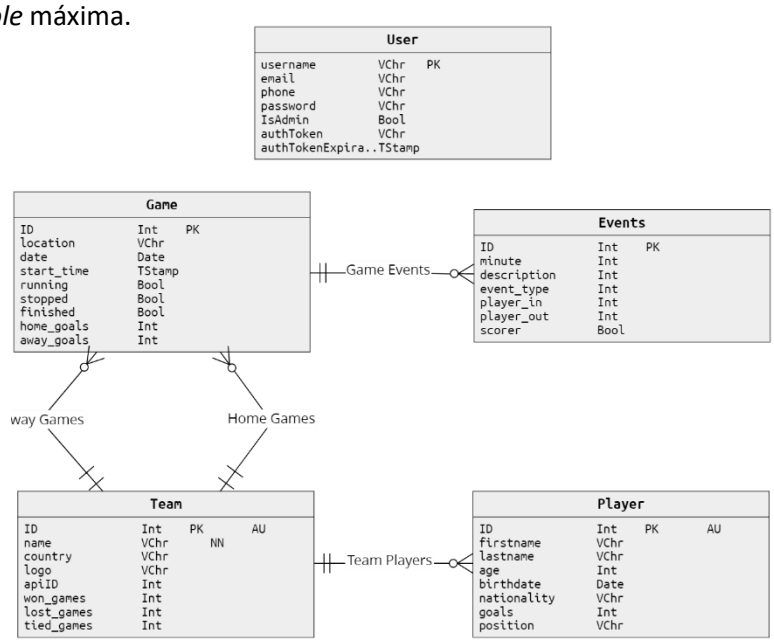

Figura 2 – Relações entre Tabelas da Base de Dados

3 Implementação do Backend

A implementação da estrutura de Backend foi realizada de forma meticulosa, já que todos os dados recebidos de utilizadores são validados de forma a evitar qualquer tipo de exceção.

Foram criadas entidades para armazenar todos os dados necessários ao funcionamento da plataforma. O Spring Boot, através da anotação ***@Entity***, faz o relacionamento entre classes Java e tabelas da base de dados. As classes criadas foram: ***User***, ***Team***, ***Player***, ***Game*** e ***Event***. Todas as entidades fazem uso de uma interface disponibilizada pelo Java Persistence API, de forma a fazer *queries* de modo rápido e simples, facilitando o acesso à base de dados. Para isto foram criadas as classes ***Repository***, que fazem parte da camada de dados. A camada de apresentação utiliza os ***Services***, que pertencem à camada de negócio.

Através de anotações como ***@NotNull*** e ***@NotEmpty***, é possível fazer a validação de todos os dados, também de forma simples. Utilizando estas anotações, todos os dados recebidos de utilizadores são validados, e todos os erros encontrados são devolvidos e apresentados ao utilizador.

Relativamente ao funcionamento mais geral da aplicação, foi desenvolvida uma função administrativa que permite a obtenção e criação de alguns dos dados necessários através de uma API REST denominada API Sports. Esta funcionalidade é baseada em pedidos HTTP aos quais o serviço responde com os dados pretendidos. A inicialização de equipas e jogadores é, então, realizada através desta mesma função.

Estando a base de dados preenchida com alguns jogadores e equipas, é possível aos administradores a criação de jogos. Com isto, um utilizador autenticado tem a possibilidade de registar um evento em qualquer jogo que se encontre por iniciar ou a decorrer. Já a utilizadores não registados é apenas possível a visualização dos eventos e dos jogos.

Passando mais detalhadamente aos eventos, o primeiro evento tem de ser, obrigatoriamente, um evento de início de jogo. Este evento apenas pode ser registado caso o jogo ainda não tiver começado nem terminado. Os restantes eventos só podem ser registados, de modo geral, se o jogo ainda não tiver terminado. No caso de existir um evento de pausa de jogo, apenas o evento de retoma de jogo é aceite.

O registo de utilizadores pode apenas ser feito por administradores. Por questões de segurança, a password de cada utilizador é guardada em forma de *hash* (através do mecanismo de *hashing* SHA-256), de modo a garantir que a obtenção da password original seja impossível. Outro mecanismo de segurança utilizado são os *tokens* de autenticação, que são validos durante 30 minutos sendo que, após este tempo, o cliente é obrigado a repetir a autenticação. Sempre que um utilizador se autentica, é criado um novo *token* e é guardado na base de dados um *timestamp* com a validade já referida a partir da hora atual.

Quando um utilizador tenta aceder a uma página, o *token* é enviado para o servidor e comparado com o que está presente na base dados. Para além disto, também é verificado se a validade já expirou ou não.

4 Testes Realizados

Descrição do teste	*Pass/Fail*
Criação, gestão de utilizadores e login	Pass
Criação e gestão de equipas	Pass
Criação e gestão de jogadores	Pass
Criação e gestão de jogos	Pass
Criação de eventos para um jogo	Pass
Visualizar detalhes de um jogo	Pass
Estatísticas ordenáveis de equipas por vitórias, derrotas e empates	Pass
Estatísticas melhor marcador	Pass
Importação de jogadores de API externa	Pass
Importação de equipas de API externa	Pass
Tratamento de exceção com a alteração de um tipo de parâmetro do URL	Pass
Tratamento de exceção com inserção de URLs inválidos	Pass
Acessos inválidos a páginas que necessitam de permissões	Pass
(...)	

5 Conclusão

Com a realização deste projeto, foi possível compreender o modo de funcionamento da *framework* Spring Boot e do mecanismo de *templates* Thymeleaf, assim como o modo de interligação das várias camadas de uma Aplicação Web.

Apesar da inexistente valorização da parte estética do projeto, o grupo aproveitou a realização do trabalho para aprofundar os conhecimentos de *Frontend*.

Concluindo, o projeto 2 de SD possibilitou a integração de uma Aplicação Web bem estruturada e desenvolvida, na qual ambos os elementos do grupo tiveram a oportunidade de trabalhar em equipa e simultaneamente, sendo que enquanto um desenvolvia código *Frontend*, o outro trabalhava em código *Backend* e vice-versa.

6 Referências

Change select options based on another selected option
https://stackoverflow.com/questions/65908339/change-select-options-based-on-another-selected-option

Redirect from @ExceptionHandler
https://stackoverflow.com/questions/54418269/redirect-from-exceptionhandler-doesnt-work

Hashing with MessageDigest
https://www.baeldung.com/sha-256-hashing-java

Spring Boot data validation
https://www.baeldung.com/spring-boot-bean-validation