Running Events – v2.0 web

Introdução

No âmbito da UC de **Tecnologias Web** foi proposta a implementação de uma interface web para gerir eventos de corrida.
Esta solução deve possuir as seguintes funcionalidades:

- Uma divisão do site em perfis, com:
  - Uma parte publica (sem necessidade de login), onde seja possível registar utilizadores (com o perfil ATLETA), pesquisar por eventos por nome ou data, participantes inscritos num evento, classificações num consultar a situação de um atleta num determinado ponto;
  - Uma parte reservada para o perfil "ATLETA", onde seja possível realizar inscrições em eventos, bem como consultar e fazer o pagamento das mesmas;
  - Uma parte reservada para o perfil "STAFF", onde seja possível registar eventos e tempos de participantes nos eventos.

A solução foi desenvolvida em **Java.**

Descrição da Solução Encontrada

De modo a conseguir encontrar uma solução para o problema pedido foram primeiro analisadas todas as tecnologias e ferramentas estudadas e utilizadas nas aulas. Após analisar tudo, ficou decidido que iria criar-se um serviço com as seguintes configurações:

- Pedidos com REST;
- Autenticação com Spring JPA;
- Acessos à base de dados PostgreSQL com JDBC;
- Paginas com conteúdo Web em JSP;
- Arquitetura MVC.

Classes

A solução desenvolvida foi feita com as seguintes classes:

- **DatabaseConfig** – Contem as configurações da base de dados a utilizar;
- **SecurityConfig** – Contém as configurações de segurança da webapp, como o tipo de autenticação, bem como quais os perfis que têm acesso a que parte da webapp;
- **SpringSecurityFormBasedJdbcAuthApp** – Classe onde é iniciada a webapp;
- **PostGresConnect** – Classe utilizada para conectar e operar sobre uma BD PostGreSQL através de JDBC;
- **RegisterManager** - Classe onde são feitos pedidos e inserções à BD, e onde os resultados dos pedidos são processados;
- **SpringSecurityController** – Controlador da webapp, trata os pedidos da vista, e adiciona atributos ao modelo quando necessário.

Páginas JSP

As paginas JSP utilizadas na **webapp** foram os seguintes:

- **about_us** – Página de "Sobre Nós";
- **classification** – Página onde são mostradas as classificações;
- **confirmations** – Página onde são mostradas confirmações das operações (como inscrições);
- **error** – Página de erro, sucede caso um utilizador não autorizado tente aceder a uma parte da webapp para a qual não tem permissões;
- **event_info** – Página onde são mostradas as informações sobre os eventos;
- **future_events** – Página onde são mostrados os eventos futuros;
- **index** – Página inicial da webapp;

- **inscrições_atleta** – Página de registo na webapp;
- **live_events** – Página onde são mostrados os eventos a decorrer;
- **login –** Página de login do site;
- **newevent** – Página onde é registado um novo evento;
- **newinscricao** – Página onde é registada uma inscrição num evento;
- **newtime** – Página onde é registado um tempo de um participante num setor;
- **newuser** – Página onde é criado um novo utilizador com o perfil ATLETA na webapp;
- **payment** – Página onde é feito o pagamento;
- **previous_events** – Página onde são mostrados os eventos passados;
- **search_events** – Página de pesquisa de eventos por nome ou data;
- **track_participants** – Página que apresenta os corredores que já chegaram a certo ponto.

Tabelas BD

Para a representação deste sistema, foi utilizada uma BD PostGres, nela foram criadas as seguintes tabelas:

- **userinfo** – Contem as informações sobre cada utilizador da plataforma;
- **user_role** – Contem o role de cada utilizador (que pode ser STAFF ou ATLETA);
- **events** – Contem a informação sobre um evento, nomeadamente o seu nome, descrição, data e custo de inscrição;
- **inscricoes** – Contem cada uma das inscrições em cada um dos eventos;
- **refs** – Contem as referencias MB geradas para pagamento das inscrições;
- **paid** – Contem a informação referente a quais os utilizadores que pagaram a sua inscrição num evento, isto é, quais os utilizadores que já confirmaram a sua inscrição.
- **times**- Contem todos os tempos registados em secções de eventos.

Funcionamento do programa

O funcionamento do programa é muito simples e consiste na realização dos seguintes passos:

1. Criar uma BD **Postgres** com as tabelas necessárias (descritas acima);
2. Compilar o projeto num **IDE** (como o **NetBeans**) ou com o comando "**mvn compile**" num terminal na pasta base do projeto.
3. Alterar, no ficheiro **config.properties**, os valores **db** (BD a usar), **host** (endereço da BD), **user** (username da BD), **password** (password da BD);
4. Executar o projeto num **IDE** (como o **NetBeans**) ou com o comando "**mvn spring-boot:run**" num terminal na pasta base do projeto.
5. Entrar no link **https::/localhost/8080** para aceder à pagina com os conteúdos.

Balanço Crítico

O trabalho entregue **cumpre** com as seguintes funcionalidades pedidas:

- Existência de vários perfis, todos eles com conteúdo reservado dependendo do perfil;
- Existência de uma secção publica onde não seja necessário ter um perfil com sessão ativa;
- Existência de métodos para registar novos perfis para utilizadores, e para autenticar os mesmos;
    i. Implementação de todas as funcionalidades pedidas para o perfil **ATLETA** (inscrições em eventos, consulta de inscrições e pagamento de inscrições);
    ii. Implementação de todas as funcionalidades pedidas para o perfil **STAFF** (registo de novos eventos e registo de tempos em eventos);
    iii. Implementação das seguintes funcionalidades pedidas para a parte publica (pesquisa de eventos, consulta de participantes num evento, consulta de classificações em eventos e consulta de situações de atletas).

O trabalho entregue **não cumpre/omite** as seguintes funcionalidades:

- Nem todos os pedidos são feitos em **POST**, isto é, alguns dos pedidos da aplicação são feitos em **GET**, o que pode levar a alguns problemas de segurança.