UNIVERSIDADE
DE ÉVORA

Sistemas Distribuídos
2020-2021

2º Trabalho Prático
Sistema nacional de vacinação

Trabalho realizado por:

- Rui Roque nº42720
- Tomás Dias nº42784

Introdução

Foi-nos proposto na UC de Sistemas Distribuídos a realização de um trabalho prático em que consistia na implementação de um sistema nacional de vacinação cujo sistema distribuído incluía os seguintes módulos:

- **Aplicação do cidadão**
- **Centro de vacinação**
- **Módulo central para a DGS**

Deveriam ser satisfeitas as seguintes operações:

- **Aplicação do cidadão**:
  - Consultar centros de vacinação (1)
  - Autoagendamento (2)
  - Notificação do estado do agendamento (3)

- **Centro de vacinação**:
  - Registar o seu funcionamento (4)
  - Registar a realização de uma vacina (5)
  - Comunicar uma lista de vacinações à DGS (6)
  - Obter número de vacinas da DGS e confirmar/desmarcar agendamentos (7)

- **Módulo central para a DGS**:
  - Manter lista de centros e a sua capacidade (8)
  - Distribuição de vacinas (9)
  - Listar nº vacinados por tipo de vacina (10)

Os dados teriam de ser armazenados de forma persistente numa base de dados *PostgreSQL*.

Para a realização do trabalho foi utilizado o framework *Spring*, sendo que aplicação se encontra alojada num servidor *Tomcat*.

A gestão de dependências, compilação, execução e deployment da aplicação foram efetuadas utilizando o *Gradle*.

Desenvolvimento

Modelo de dados

O primeiro passo do desenvolvimento foi decidir como estruturar a informação que seria utilizada pelo sistema e como esta seria persistida.

A ferramenta *Spring Data JPA* teve um papel fundamental nesse sentido, permitindo a criação de tabelas e relações utilizando *Java*, dispensando a utilização de comandos *SQL* na maior parte dos casos. As classes correspondentes às entidades são as seguintes:

- ***Centre*:** contém o nome do centro de vacinação, data de funcionamento, capacidade diária e disponibilidade.

- ***Vaccines*:** contém o nome das vacinas existentes.

- ***User*:** contém o email e a password do utilizador.

- ***Appointment*:** contém o id, email, password, nome e email do utente assim como a data e o centro para o autoagendamento.

- ***Vaccinated*:** contém o id (igual ao id do utente que estava previamente na tabela appointment), email do utente vacinado, data em que foi vacinado (igual, em caso de confirmação, da data de agendamento do utente vacinado) e tipo de vacina.

- ***Notification*:** contém o id e email do utilizador e a mensagem da notificação.

Acesso e manipulação de dados

As classes dedicadas ao acesso e manipulação de dados (chamadas à base de dados) bem como à lógica das operações encontram-se na diretoria *services*. Essas classes são iniciadas com a anotação *@Service* e são posteriormente chamadas nos Controllers através de injeção de dependências (utilizando a anotação *@Autowired*). Os serviços implementados foram os seguintes:

- ***RegisterService*:** serviço que implementa a operação de registo do utente, para visualização de notificações e a permissão de autoagendamentos.

- ***ListCentreService*:** serviço que implementa a operação (1).

- ***DoAppointmentService*:** serviço que implementa a operação (2).
- ***ShowNotificationsService*:** serviço que implementa a operação (3).
- ***UpdateCentreOperationService*:** serviço que implementa a operação (4).
- ***RegisterVaccinatedUserService*:** serviço que implementa a operação (5).
- ***VaccinesDistributionService*:** serviço que implementa a operação (9).
- ***ListVaccinatedUsersService*:** serviço que implementa a operação (10).

Receção e resposta a pedidos

É na diretoria *controllers* que se encontram as classes responsáveis por receber e responder a pedidos. Sendo que se utilizou a arquitetura *REST*, estas classes são iniciadas com a anotação *@RestController*. Os *controllers* implementados foram os seguintes:

- ***DoAppointmentController*:** redireciona o pedido de um autoagedamento para o endpoint ***/appointment***, via *HTTP POST*.
- ***ListCentresController*:** redireciona o pedido para consulta dos centros disponíveis para o endpoint ***/centres***, via *HTTP GET*.
- ***ListVaccinatedUsersController*:** redireciona o pedido para consulta do número de vacinados por tipo de vacina para o endpoint ***/vaccinated*** via *HTTP GET*.
- ***RegisterController*:** redireciona o pedido de registo do utilizador para o endpoint ***/register***, via *HTTP POST*.
- ***RegisterVaccinatedUserController*:** redireciona o pedido de registo de um utente como vacinado para o endpoint ***/registed-vaccinated-user***, via *HTTP POST*.
- ***ShowNotificationController*:** redireciona o pedido de consulta das notificações para o endpoint ***/notifications***, via *HTTP GET*.
- ***UpdateCentreOperationController*:** redireciona o pedido de registo do funcionamento de um centro para o endpoint ***/update-centre-operation***, via *HTTP POST*.

- ***VaccinesDistributionController*:** redireciona o pedido de distribuição das vacinas para o endpoint ***/vaccines-distribution***, via *HTTP POST*.

Interfaces de utilizador

De modo a facilitar a realização das operações de cada um dos módulos, foram criados alguns ficheiros *HTML* de forma a permitir que os pedidos sejam executados a partir de aplicações web (cidadão, centro de vacinação e DGS) através da ocorrência de eventos (formulários).

Os menus das aplicações do Cliente, Centro de Vacinação e DGS podem ser encontrados na diretoria *resources/static* através dos ficheiros *index.html*, *cv-index.html* e *dgs-index.html*, respetivamente, bem como outros *HTML* utilizados.

De modo a facilitar a apresentação da informação proveniente de um determinado *endpoint* de forma dinâmica e em tempo real, utilizou-se as bibliotecas *JQuery* e *AJAX*.

Informações adicionais

As tabelas e alguns *inserts* encontram-se no ficheiro *db.sql*.

Alguns parâmetros de configuração (como por exemplo parâmetros de ligação à base de dados) encontram-se no ficheiro de propriedades (*application.properties*).