**Monitorização de qualidade de serviço em**

**comunicações ferroviárias**

**Organização do dossier de projeto**

Todo o código fonte encontra-se disponível no *GitHub*. O repositório dos componentes Base de Dados, *Web API* e cliente *Web* está disponível em <https://github.com/LikaV1996/ProjectoFinaldeCurso>.

## Requisitos para instalação do projeto

Garantir que o *Node.js* e o *npm* estão instalados (versão 10.9.0 ou superior e versão 5.6.0 ou superior respetivamente), bem como o *IntelliJ IDEA CE* (versão 2018.1.7) com a versão 10 do *SDK* do *Java* e também o *PostgresSQL* (versão 11.2). É também aconselhado apenas por facilitação, instalar o *PGAdmin4* para não ser necessário mexer com a consola do *PostgreSQL*. De seguida encontra-se um vídeo do *Youtube* que ajudou o grupo a inicializar e fazer *setup* a esta componente, <https://www.youtube.com/watch?v=fZQI7nBu32M>.

**Configurações:**

* PostgresSQL: criar uma base de dados e abrir uma *tab Query Tool*. De seguida necessário abrir os ficheiros, na diretoria Database, com todos os scripts, de 1 a 6, e corrê-los por essa ordem. No final a base de dados estará instanciada e populada;
* Web API: aceder ao ficheiro de configuração application.properties, na diretoria Server API/probe-server/src/main/resources, e afectar as variáveis spring.datasource.url, spring.datasource.username e spring.datasource.password de acordo com a base de dados que foi criada anteriormente de forma a que a conexão possa ser estabelecida;
* Cliente Web: instalar todos os módulos node, acendendo na linha de comandos à diretoria Client/my-app e executando o comando *>npm install;*

## Como correr o projecto

***Web API*:** no *IntelliJ* importar o projecto pelo *Maven* (que está na diretoria: Server API\probe-server\pom.xml)*.* Executar um *rebuild project* e correr através da classe Application, localizada na diretoria Server API/probe-server/src/main/java/pt/solvit/probe/server.

***Cliente WEB*:** executar a rotina *>* *ng serve --open*;

## Organização das pastas do código fonte

A diretoria Projectoestá subdividida em três pastas:

* DataBase – está subdividido em seis ficheiros drops, create\_tables, create\_views, create\_triggers, create\_funcs\_and\_procs e init\_tables. Estes ficheiros permitem que a base de dados seja apagada, criada e/ou populada;
* Server API – existem duas pastas: uma resources com os .xml com a dependências do Spring etc. e uma java com o código. Dentro da pasta java temos vários *packages*: config com classes relacionadas com configurações (por exemplo registos de *interceptors* e do CORS), controller com os controladores (classes com os *endpoints* e objetos relacionados), filter com o filtro do CORS, interceptors com os interceptores, model com os objetos , repository com os repositórios (classes que fazem a coneção à DB e *Data Access Objets*), service com os serviços (classes responsáveis por impor restrições de negócio) e util (com classes *util*)
* Client – composto essencialmente pela diretoria my-app/src/app que contém uma pasta \_services onde podem ser encontrados todos os ficheiros que irão estabelecer a ligação com a *Web* *API* para obter os dados e executar ações. A pasta authguard e a httpinterceptor contém os ficheiros dos *interceptors*. Todas as restantes pastas são componentes, que constituem a interface com o utilizador da aplicação.

O ficheiro httproutes.ts contém todas as rotas que foram usadas para fazer pedidos á Web API.

O ficheiro app-routing.module.ts contém todos os *paths* disponibilizados pela aplicação.

Os ficheiros app.component.css, app.component.html, app.component.spec.ts e app.component.ts correspondem ao componente principal da aplicação.