

# UNIVERSIDADE DE ÉVORA

# Trabalho de Sistemas Operativos

Fábio Samina nº 40299

Pedro Amaro nº42348

Ano Letivo 2019/2020

Sistemas Operativos, José Saias

## 1 Descrição do trabalho

O trabalho consiste numa web application que permite aos utilizadores uma autogestão do distanciamento social, ao disponibilizar o grau de ocupação de um espaço numa interface amigável.

Qualquer utilizador tem acesso a consultar o grau de ocupação de um espaço e também a encontrar as informações dos espaços mais perto de si.

Utilizadores autenticados podem também adicionar e remover informações sobre os espaços.

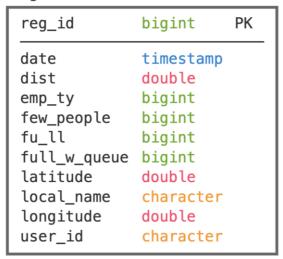
Algumas das sugestões para este projeto foram a utilização de Spring (para gestão do serviço) em conjunto com gradle (para a gestão de dependências).

Como restrições obrigatórias tínhamos a utilização de uma base de dados relacional como forma de armazenamento persistente de dados, interface amigável com os dados representados num mapa (sendo a recomendação o LeafLet).

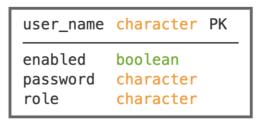
#### 2 Decisões tomadas

Decidimos implementar a aplicação web utilizando a tecnologia Spring. Temos uma base de dados PostgresSQL que seguem o seguinte modelo:

registo



utilizador



A nossa aplicação tem um /home em que o utilizador é exposto a todas as opções que pode executar no sistema.

A nossa aplicação permite:

- -Adicionar um registo de um local (Requer login)
- -Ver e apagar um registo (Requer login e só permite apagar registos associados á conta)
- -Navegar num mapa interativo com todos os locais conhecidos e os seus registos.
  - -Saber qual o local conhecido mais perto do utilizador.

Tomamos em consideração todos os inputs de forma a garantirmos que não são submetidos dados incoerentes (Exemplo: Preencher latitude com letras).

#### 3 Métodos Get/Post

```
@PostMapping("/user/add")
public RedirectView addUser(@RequestParam(name = "name") String name,
@RequestParam(name = "pass") String password)
```

- Esta função é responsável adicionar á base de dados um novo utilizador.
- @ GetMapping("/locals")
  public String getLocals()
- Esta função é responsável por consultar na base de dados todos os locais conhecidos e os seus registos. Esta função invoca o método responsável por apagar registos com mais de 1 hora. Temos noção que caso haja 1M de pedidos para /locals (no mesmo minuto) então haverá 999 999 chamadas da função que apaga os registos antigos desnecessárias, no entanto devido ao feedback sobre o último trabalho duvidamos sobre se seria boa pratica ou não ter uma thread á parte que fizesse essa limpeza a cada hora.
  - Retorna uma string que é o ficheiro html da servlet.

- Esta função adiciona o local ao mapa caso ele não seja conhecido e adiciona o registo.
  - Recebe como argumentos o nome do local, voto e coordenadas. Retorna um redirect para /home

```
@GetMapping("/userReg")
public String getUserRegist()
```

- Esta função é responsável por criar a tabela dos registos do utilizador.
  - Retorna uma string que é o ficheiro html da servlet.

- Esta função é invocada quando o utilizador pretende eliminar um dos seus registos.
  - Retorna um redirect para /home.

- Esta função calcula com base na posição geográfica do utilizador os locais mais perto do mesmo.
  - Retorna uma string que é o ficheiro html da servlet.

## 4 Observações

Após muita reflexão e pesquisa, infelizmente, não conseguimos desenvolver a componente de replicação. No entanto achamos que este trabalho foi muito importante para consolidarmos parte da matéria dada nas aulas praticas e sem dúvida ganhamos novas habilidades bastante valiosas no mundo profissional.