Aplicações distribuídas sobre a internet – Entrega intermédia

Rui Cavalheiro - 85229

Diogo Cardoso - 85224

Outubro 2021

Grupo 1

rui.b.cavalheiro@tecnico.ulisboa.pt

diogo.carrola.cardoso@tecnico.ulisboa.pt

Índice

1 Data Model

1.1 Modelo ORM

1.1.1 Class Gate

Esta classe representa o objecto que ira guardar a informação sobre as gates do sistema.
Os atributos são:

- tablename = 'gate' → Nome da tabela
- id (Integer) → ID do gate que será utilizado como primary key
- secret (String) → Segredo para aceder á gate
- location (String) → Breve descrição da localização da gate
- count (Integer) → Contador do numero de utilizações da gate

Possui um método "as_json" para transformar o atributos numa representação em JSON

1.1.2 Class User

Esta classe representa o objecto que ira guardar a informação sobre as gates do sistema.
Os atributos são:

- tablename = 'user' → Nome da tabela
- id (Integer) → ID do utilizador que será utilizado como primary key
- code (Integer) → Código para entrar na gate
- time_stamp (String) → Data do pedido do código

Possui um método "as_json" para transformar o atributos numa representação em JSON

2 Documentação REST API

2.1 Service

'GET' → "/users/<path:id>/code"

Descrição: Gerar um novo codigo para o utilizador usar na gate
Input: None
Return:

- Em caso de sucesso retorna um JSON com o seguinte formato:

```
{
    'errorCode': <int>,
    'errorDescription': <string>,
    'code': <new code>
}
```

- Em caso de erro retorna um JSON com 2 parametros:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

'GET' → "/gates/id"

Descrição: Autentificar a existência e segredo da gate
Input: JSON com formato:

```
{
    'id': <int>,
    'secret': <string>
}
```

Return:JSON com formato:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

'GET' → "/gates/code"

Descrição: Autentificar o código para abertura da gate.

Input: JSON com formato:

```
{
    'id':<string>,
    'code':<string>,
    'gate_id':<int>
}
```

Return: JSON com formato:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

2.2 AdminWebApp

'GET' → "/"

Descrição: Retorna um HTML com 2 opções; uma para a criação de um novo gate e outra para apresentar a lista de gates.

Return: Ficheiro HTML "index.html"

'GET' → "/newGate"

Descrição: Retorna um HTML para preenchimento do formulário de criação de um novo gate

Return: Ficheiro HTML "newGate.html" para preenchimento de um formulário

'POST' → "/createGate"

Descrição: Cria um gate com os dados preenchidos no formulário.

Return:

- Em caso de sucesso retorna um JSON com o segredo no tipo "<segredo>".
- Em caso de erro retorna um JSON com 2 parametros:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

'GET' → "/listGate"

Descrição: Apresenta uma lista em JSON dos gates
Return: Lista JSON com os gates e respetivas informações (id, segredo, localização, count).

2.3 GateData Service

'PUT' → "/users/<path:id>/code"

Descrição: Introduzir um novo codigo de utilizador para acesso ás gates.
Input: JSON com formato:

```
{
    'code': <string>
}
```

Return: JSON com formato:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

'GET' → "/users/<path:id>/code"

Descrição: Confirma o código de abertura da gate.
Input: JSON com formato:

```
{
    'id':<string>,
    'code':<string>,
    'gate_id':<int>
}
```

Return: JSON com formato:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

'GET' → "/gates/id"

Descrição:
Input: JSON com formato:

```
{
    'id': <int>,
    'secret': <string>
}
```

Return: JSON com formato:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

'GET' → "/gates"

Descrição: Get list of gates.
Input: None
Return: Lista JSON com formato:

```
{
    'id':<int>,
    'secret': <string>,
    'location': <string>,
    'count':<int>,
}
```

'PUT' → "/gates"

Descrição: Create and save a new gate in database
Input: JSON com formato:

```
{
    'id': <string>,
    'location': <string>
}
```

Return:

- Em caso de sucesso retorna um JSON com o segredo no tipo "<segredo>".
- Em caso de erro retorna um JSON com 2 parametros:

```
{
    'errorCode': <int>,
    'errorDescription': <string>
}
```

3 Decisões de desenvolvimento

Para esta entrega intermédia decidimos criar uma tabela no mesmo serviço da base de dados dos gates para guardar os utilizadores, respectivos códigos de entrada e a hora a que foram gerados a fim de simular um acesso a uma base de dados de utilizadores. Esta tabela é inicialmente populada com um utilizador 1111 como é esperado de existir na entrega intermédia.

Para anular um código utilizado é colocado o seu time_stamp 100 semanas atrás.

3.1 códigos de erros retornados pelo dataService

Do dataService é esperado receber algum destes códigos sempre que se acede a algum endpoint. Nem todos os endpoints retornam todos os códigos.

- 0 → sucesso
- 1 → The secret is not valid for this gate.,
- 2 → This Code Has Been Used Already.,
- 3 → No gate found for this ID.
- 4 → Gate id not available.
- 5 → database had an error with JSON input.

- 6 → Failed to receive code.

3.2 Códigos de erros retornados pelo server

Do server é esperado receber algum destes códigos sempre que se acede a algum endpoint. Nem todos os endpoints retornam todos os códigos

- 7 → Couldn´t access database.
- 8 → Lacking arguments.
- 9 → !!! Bad form !!!
- 10 → No ID or secret.
- 11 → Server had an error with JSON input.

Para alem destes 5 códigos, o server pode também retornar algum dos 7 códigos que o dataServide retorna.