Guião de Demonstração

Grupo A17

F1 - Funcionamento Normal da Replicação

1. Iniciar 3 estações com atrasos diferentes com:
   mvn exec:java -Dlag=<atraso em ms> -Dws.i=<nº da estação>
2. Iniciar o binas-ws normalmente:
   mvn exec:java
3. Correr a demonstração em binas-ws-cli:
   mvn exec:java
4. Carregar ENTER sempre que pedido, sem reiniciar nenhuma das estações/binas
5. Esperar que a demonstração acabe.

Para informação sobre o teste, ver anexo.

F2 - Tolerância a Faltas (Estação/Binas)

1. Iniciar 3 estações com atrasos diferentes com:
   mvn exec:java -Dlag=<atraso em ms> -Dws.i=<nº da estação>
2. Iniciar o binas-ws normalmente:
   mvn exec:java
3. Correr a demonstração em binas-ws-cli:
   mvn exec:java
4. Quando pedido para carregar ENTER:
   a. terminar UMA estação **k** $\in \{2,3\}$, carregando ENTER no respectivo terminal.
   b. mvn exec:java -Dlag=<atraso em ms> -Dws.i=<**k**> (no mesmo terminal)
5. Carregar no botão ENTER.
6. Quando pedido para carregar ENTER novamente:
   a. Reiniciar o servidor binas-ws, carregando ENTER no respectivo terminal.
   b. mvn exec:java (no mesmo terminal)
7. Esperar que a demonstração acabe.

Para informação sobre o teste, ver anexo.

Anexo

Caso de demonstração **F1:**

Inicialmente:

1. Ativa 3 utilizadores com emails diferentes - activateUser
2. Pede uma bina para cada utilizador, em estações diferentes - getBina
3. Verifica se o saldo dos utilizadores é decrementado - getCredit
4. Retorna a bina de cada utilizador - returnBina
5. Verifica se o saldo dos utilizadores é atualizado com o bónus respetivo - getCredit

Output binas-ws e stations (esperado):

ACTIVATE USER

Como o utilizador não se encontra na cache, pergunta primeiro às estações se conhecem o utilizador e apenas depois de se certificar que não existe cria-o:

(binas)

```
CALL 0 GetBalanceAsync: LucasRafael@tecnico.ulisboa.pt
CALL 1 GetBalanceAsync: LucasRafael@tecnico.ulisboa.pt
CALL 2 GetBalanceAsync: LucasRafael@tecnico.ulisboa.pt
RESPONSE getBalanceAsync: Invalid User
RESPONSE getBalanceAsync: Invalid User
CALL setBalanceAsync: 0, LucasRafael@tecnico.ulisboa.pt, 10
CALL setBalanceAsync: 0, LucasRafael@tecnico.ulisboa.pt, 10
CALL setBalanceAsync: 0, LucasRafael@tecnico.ulisboa.pt, 10
RESPONSE setBalanceAsync: OK
RESPONSE setBalanceAsync: OK
```

(station)

```
CALL getBalance (LucasRafael@tecnico.ulisboa.pt)
RETURN getBalance InvalidUser_Exception
CALL setBalance (0, LucasRafael@tecnico.ulisboa.pt, 10)
RETURN setBalance (void)
```

GET BINA

Como saldo encontra-se na cache apenas é necessário atualizar o saldo:

(binas)

```
CALL setBalanceAsync: 3, LucasRafael@tecnico.ulisboa.pt, 9
CALL setBalanceAsync: 3, LucasRafael@tecnico.ulisboa.pt, 9
CALL setBalanceAsync: 3, LucasRafael@tecnico.ulisboa.pt, 9
RESPONSE setBalanceAsync: OK
RESPONSE setBalanceAsync: OK
```

(station)

```
CALL setBalance (3, LucasRafael@tecnico.ulisboa.pt, 9)
RETURN setBalance (void)
```

GET CREDIT

Não há output pois respondemos imediatamente a estes pedidos apenas perguntando às estações no caso de o utilizador não estar na cache.

RETURN BINA

Igual a getBinas, é apenas necessário atualizar o saldo nas estações.

Caso de demonstração **F2:**

Inicialmente:

1. Ativa 3 utilizadores com emails diferentes - activateUser

Depois repete 3 vezes o seguinte:

1. Pede uma bina para cada utilizador, em estações diferentes - getBina
2. Verifica se o saldo dos utilizadores é decrementado - getCredit
3. Retorna a bina de cada utilizador - returnBina
4. Verifica se o saldo dos utilizadores é atualizado com o bónus respetivo - getCredit
5. Pede para o utilizador carregar ENTER para prosseguir à próxima iteração.

O utilizador deve reiniciar/terminar uma estação/binas quando indicado.

CHAMADAS

Todas as chamadas devem proceder de forma semelhante com a excepção do caso do binas ter sido reiniciado. Neste caso esperamos primeiro um getBalance de cada vez que o novo binas recebe um utilizador novo.