

 <p>instituto superior de engenharia de lisboa</p>	<p>Mestrado em Engenharia Informática e de Computadores (MEIC) Mestrado em Engenharia Informática e Multimédia (MEIM)</p> <p>Unidade Curricular de Computação Distribuída (CD-2526)</p>
---	---

Laboratório 3

Objetivos:

- Experimentação de modelos de comunicação por eventos/mensagens utilizando um servidor RabbitMQ alojado num *container* Docker;
- Desenvolvimento em Java, usando a API *amqp-client*, de aplicações Produtoras e Consumidoras de mensagens usando conceitos do modelo *Publish/Subscribe*

Este laboratório pressupõe que cada grupo tem um projeto GCP e uma VM com o runtime Docker instalado de acordo com o Laboratório 01.

- 1) Garanta que a sua VM tem regras de Firewall para abrir os portos TCP/IP **5672** e **15672**. Para o efeito deve validar e acrescentar firewall rules no serviço "VPC network" da GCP aplicáveis a todas as instâncias de VM do seu projeto.
- 2) Execute um *container* que aloja um servidor (Broker) RabbitMQ com o seguinte comando:

```
$ docker run -d --hostname rabbithost --name rabbitmq \
-p 5672:5672 -p 15672:15672 rabbitmq:management
```
- 3) Execute no seu *browser* a aplicação de gestão do Broker no seguinte URL:

```
http://<public IP da VM>:15672
```

 - a) Faça Login com User: **guest** e password: **guest**
- 4) Faça *download* no Moodle do ficheiro RabbitMQ-Examples.zip que contém 3 projectos Maven (aplicação *Configurator* de configuração de *Exchanges*, Filas *Bindings* etc e aplicações de produção (*Producer*) e consumo (*Consumer*) de mensagens.
- 5) Usando as aplicações atrás referidas, com eventuais modificações que ache necessárias, concretize os diferentes padrões de encaminhamento de mensagens (Tipos de *exchange* e *Bindings*), apresentados nos slides e demonstrados na aula teórica, utilizando a aplicação web no browser de gestão do RabbitMQ para ir confirmando as configurações que efetua.
- 6) Usando exclusivamente o modelo *Publish/Subscribe* e o servidor RabbitMQ implemente uma aplicação distribuída de troca de mensagens entre múltiplos utilizadores (*PubSubChat*) nos seguintes cenários:
 - a) Todos os utilizadores recebem todas as mensagens;
 - b) Um utilizador só recebe mensagens associadas a um único dos seguintes temas (Desporto, Cinema, Tecnologia);
 - c) Generalizar para que cada utilizador possa receber mensagens classificadas com qualquer subconjunto de temas, por exemplo Desporto e Tecnologia;
 - d) Considere uma solução que permita aos receptores da mensagem enviar uma mensagem de resposta que só será consumida pelo utilizador que enviou a mensagem inicial. Para tal, a aplicação que envia a mensagem deverá ser consumidora de uma *queue* privada associada a um *exchange*, para publicação da resposta, e enviar na mensagem o nome desse *exchange*.