Mestrado em Engenharia Informática e de Computadores (MEIC)
Mestrado em Engenharia Informática e Multimédia (MEIM)

Unidade Curricular de Computação Distribuída (CD-2324)

Laboratório 01

Objetivo: Criação e experimentação sobre uma infraestrutura de máquinas virtuais na plataforma Google Cloud Platform e de *Containers* Docker para alojar e executar serviços ao longo da Unidade Curricular de CD, incluindo os trabalhos de avaliação prática.

1) Seguindo as instruções no ficheiro em anexo, <**CD 2324-Registo na Google Cloud Platform como Aluno.pdf**> cada grupo de alunos deve criar um projeto GCP em que todos os alunos do grupo podem gerir recursos nesse projeto (*role owner* do projeto).

2) Seguindo as instruções no ficheiro em anexo, <**CD2324-CriarVms-GCP.pdf**>, crie uma máquina virtual (VM) no serviço *Compute Engine* da GCP. Por questão de uniformização, a VM deve ter o sistema operativo Linux CentOS versão Stream 8 e a instalação do Java OpenJDK 11.

3) Em anexo encontra o artefacto <**`ServiceREST.jar`**> que é um serviço HTTP REST que aceita pedidos no porto 7500 através das seguintes rotas http:

```
http://<host ip>:7500/ping
http://<host ip>:7500/hello/<some name>
http://<host ip>:7500/calc/number{+,-,*}number
      Ex: http://<host ip>:7500/calc/5*3 retorna 15
```

   a) Teste na sua máquina local (IP localhost) o serviço chamando as rotas num browser;

   b) Faça upload do artefacto para a VM criada no ponto 2) e, garantindo que a VM tem o porto 7500 aberto numa *firewall rule*, execute o serviço HTTP REST e verifique que consegue chamar as várias rotas do serviço a partir de um browser;

   c) Verifique que, durante a execução, o serviço gera um ficheiro de *logs* `<ServiceRestlogs>` na diretoria corrente onde é executado o serviço. Visualize o conteúdo do ficheiro.

4) Instale o *runtime* Docker na VM de acordo com o slide 16, dos slides (**CD-03 VMs-Containers**) conforme foi explicado na aula teórica;

5) Verifique que está tudo a funcionar executando e interpretando o seguinte comando:

```
docker run fedora cat /etc/os-release
```

6) Crie o ficheiro `Dockerfile` e a imagem Docker de um *container* para executar o serviço HTTP REST com o comando `<docker build>;`

7) Execute o comando `<docker images>` e verifique que o repositório local da VM contém a imagem que gerou.

8) Execute dois *containers* (docker run) baseados na imagem que gerou e que ficam disponíveis nos portos 8000 e 8001 da VM (*host machine*), garantindo que esses portos estão abertos nas *firewall rules* da VM. Verifique que consegue chamar os dois serviços em execução nos dois *containers*.

9) No conteúdo do ficheiro Dockerfile (ver slides da aula) a diretoria WORKDIR (*working diretory*) é onde o *container* executa o serviço REST, por exemplo `/usr/servicerest`. Será aí que é criado o ficheiro de *logging* `ServiceRestlogs` que regista eventuais erros ou as chamadas às rotas `do serviço`. No entanto, fora do *container* e após o *container* terminar não conseguiremos ler o ficheiro dado que cada *container* tem um *file system* próprio e que não é persistido na máquina *host*.

10) Crie um `volume` docker de nome *logging* e volte a lançar os *containers* mapeando a diretoria de trabalho dentro do container para esse volume.

11) Execute o comando <*docker volume ls*> para visualizar os volumes existentes;

12) Execute o comando <`docker volume inspect logging`> para verificar a diretoria do *file system* do *host* onde está o volume. Em princípio será (`/var/lib/docker/volumes/logging/_data`)

13) Verifique que agora já consegue visualizar os *logs* durante a execução ou após o *container* ter terminado, acedendo à diretoria do *host* onde está o volume `logging:`

```
sudo ls -al /var/lib/docker/volumes/logging/_data
sudo cat /var/lib/docker/volumes/logging/_data/ServiceRestlogs
```

14) Cada grupo de alunos deve criar uma conta no `dockerhub partilhada` para poderem publicar (*push*) as imagens e posteriormente poderem reutilizar (*pull*) da imagem e executar outros *containers* em qualquer máquina com docker.

15) Execute os comandos docker que permitem:

   a) Listar os *containers* em execução <`docker ps -a`>
   b) Fazer *start*, *stop*, *kill* e *remove* (rm) de *containers*;
   c) Listar e remover imagens no repositório local
   d) Criar uma imagem com tags diferentes e no formato exigido para poder realizar *push* de imagens no repositório dockerhub <`user`>/<`image name`>[:<`tag`>]

   e) Após realizar *push* da imagem para o repositório dockerhub, apague a imagem no repositório local (comando `$docker image rm <image_name>`) e faça *pull* da imagem a partir do dockerhub. Execute um novo *container* com essa imagem;

   f) Outra possibilidade é criar uma segunda VM para verificar que se pode fazer *push* numa VM e fazer *pull* na outra VM para se executarem *containers*. Na Figura 1, ilustra-se como se pode, na plataforma GCP, criar novas VMs com a mesma configuração onde se instalou o Docker. Existe em GCP a possibilidade de criar uma <machine image> e depois facilmente criar novas VMs com a mesma configuração.

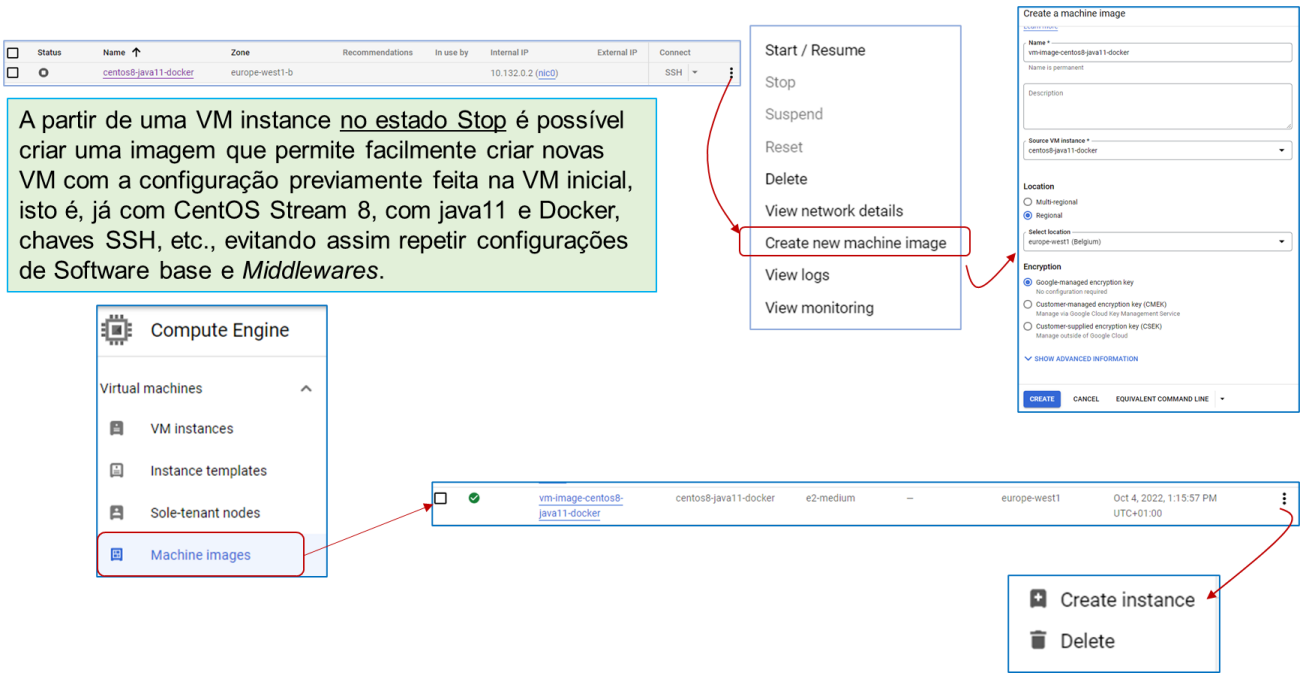

Figura 1: Criar imagem de uma VM, permitindo a criação de novas VMs com a mesma configuração