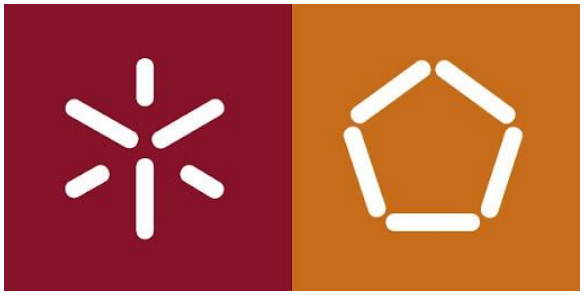


Universidade do Minho

Relatório do Projeto

Ano letivo 2022/2023

Janeiro 2023

Grupo 37

Mestrado em Engenharia Informática

Unidade Curricular de Aplicações e Serviços de Computação em Nuvem

Github: https://github.com/CatarinaOG/Projeto-ASCN

Ana Gonçalves pg50180

Bruno Pereira pg50271

Francisco Toldy pg50379

João Delgado pg50487

Rui Chaves pg47637

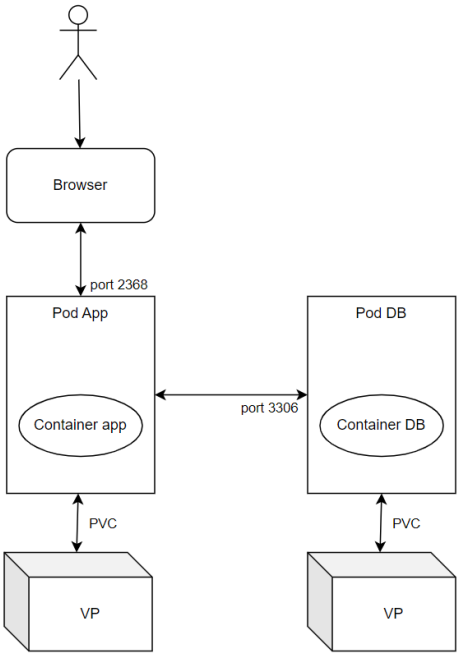
Índice

1. Introdução

Este documento relata o desenvolvimento do trabalho prático da unidade curricular de Aplicações e Serviços de Computação em Nuvem. O enunciado deste trabalho tinha como objetivo que o grupo recorresse aos conhecimentos adquiridos no decorrer da cadeira de forma a automatizar a instalação configuração, monitorização e avaliação da aplicação *ghost*. O enunciado estava dividido em várias tarefas, sendo que o grupo implementou a Tarefa Base, Tarefa de Monotorização, Avaliação e Resiliência, embora as últimas duas com algumas limitações. Assim, neste documento iremos abordar a descrição da arquitetura implementada, bem como as ferramentas e abordagem utilizadas, a replicação e monitorização, terminando com a avaliação experimental e a conclusão.

2. Arquitetura

A aplicação *ghost* está dividida em 2 componentes principais: as funcionalidades da aplicação em si e a base de dados que a suporta. Estes 2 componentes foram implementados em 2 pods diferentes, cada uma com um container. Além das 2 pods, a ambas são suportadas por um volume persistente de forma a assegurar a persistência dos dados armazenados durante a execução da aplicação.
Uma das pods está a executar o container da base de dados e outra pod está a executar o container com as funcionalidades da aplicação. A decisão de separar as 2 componentes em pods diferentes partiu do que foi apresentado no enunciado relativamente a tentativa de separar o máximo possível dos componentes constituintes da aplicação em diferentes pods. Assim, foi esta a divisão que foi feita pelo grupo, considerada adequada para o cenário apresentado.



3. Ferramentas e Abordagens Utilizadas

A instalação automática da aplicação partiu da *code* base fornecida pelos docentes, sendo que o restante foi construído ainda em ansible e em ficheiros *kubernetes*, consoante as necessidades que foram surgindo durante o desenvolvimento do trabalho.
A perceção do que seria necessário incluir numa instalação automática da aplicação foi informada por uma primeira tentativa de instalar o Ghost manualmente, de forma a entender os passos principais e as configurações necessárias. Chegou-se à conclusão que seria necessário seguir um conjunto de factos que serão de seguida especificados:

Em relação ao cluster kubernetes, primeiro era necessário criar um *namespace*, de modo a definir o conjunto de recursos que poderia ser usado por cada container. De seguida, seria necessária a criação de uma *storage class* para que esta pudesse ser utilizada pelo PVC, de modo que fosse possível garantir a persistência dos dados produzidos pela aplicação na Base de Dados juntamente com a criação do PVC que vai permitir dar *bound* do volume persistente à *claim*.

Finalmente, para a instalação completa da base de dados é necessário disponibilizá-la através de um serviço para que esta possa ser acedida pelos restantes nodos do cluster) *kubernetes* (definido o tipo do serviço: **Cluster IP**) e fazer o seu *deployment* efetivo, permitindo a especificação de variáveis de ambientes da imagem utilizada:

- **MYSQL_DATABASE**: ghost, que será o nome da Base de Dados usada pela aplicação ghost
- **MYSQL_USER**: ghostuser, usado para criar um utilizador na Base de Dados com este nome
- **MYSQL_PASSWORD**: pass, a password do utilizador criado

Numa segunda fase, a instalação da aplicação em si, vai ser necessário criar outra *claim* para o volume persistente juntamente com a exposição do *ghost* como um serviço, mas neste caso, para que pudesse ser acedido a partir do exterior do cluster (definido o tipo de serviço: **LoadBalancer**).

Só a partir da criação do *service* é que vai ser possível obter o IP do serviço fornecido pelo *ghost*, que vai ser necessário para permitir o *deployment*, especificando o *url* com base neste valor.
Quanto ao ficheiro deployment escrito no módulo kubernetes do ansible, foram especificadas as seguintes variáveis de ambiente necessárias para a instalação do ghost e respetiva configuração:

- **database__client:** será o mysql, motor de base de dados usado
- **database__connection__host:** mysql-service, que será o pod onde estará a correr a Base de Dados. O nome será traduzido para um IP pelo DNS kubernetes
- **database__connection__user:** ghostuser, que especifica o utilizador da Base de Dados criado na instalação da mesma
- **database__connection__password:** pass, password do utilizador
- **database__connection__database:** ghost, nome da Base de Dados criada na instalação da mesma
- **url:** "http:// {{ ghost_ip }}:{{ ghost_port }}"
- **mail__from:** endereço email de onde serão enviados os emails de confirmação para os utilizadores
- **mail__transport:** "SMTP", protocolo de transporte usado
- **mail__options__host:** "smtp.gmail.com", domínio onde está hospedado o serviço
- **mail__options__service:** "Gmail", serviço de email usado
- **mail__options__port:** 465, porta usada para comunicar com o serviço de email
- **mail__options__secure:** flag de segurança
- **mail__options__auth__user:** Username usado para fazer a autenticação do serviço
- **mail__options__auth__pass:** Password usado para fazer a autenticação do serviço

Pelo valor das variáveis é possível observar que o serviço de Mail configurado para a aplicação foi o Gmail, uma vez que foi o serviço que se revelou mais fácil de configurar e utilizar.

Finalmente, para terminar a configuração da aplicação, era necessário fazer a criação do utilizador "admin", de modo que a aplicação pudesse funcionar como esperado. Para o fazer, foi feita uma task, que corria um comando "kubectl exec" para fazer a criação direta no pod mysql, especificado com várias variáveis guardadas anteriormente no inventory.

4. Replicação

Relativamente à escalabilidade e resiliência, foi possível criar um playbook que permite alterar o número de réplicas para o deployment da aplicação ghost. O grupo apostou na replicação da aplicação ghost e não na correspondente base de dados pois concordou que seria mais vantajoso existir uma maior escalabilidade e resiliência por parte da aplicação comparativamente à vantagem mais significativa da base de dados: a resiliência fornecida por mais réplicas. Para além deste fator, o acesso à aplicação em si implica um Round Trip Time maior do pedido, devido à longa distância entre o cliente e o recurso físico onde está a correr a aplicação.

Idealmente, a alteração do número de réplicas deveria ser feita de acordo com os alertas de ultrapassagem de thresholds da utilização de certos recursos, disponibilizados pela monitorização, no entanto não foi possível, sendo então a versão mais simples (alteração para um número de réplicas desejado) preservada.

5. Monitorização

De modo a realizar a monitorização da aplicação, optou-se pela utilização das ferramentas de monitorização da GCP. A automatização da criação das dashboards é feita partindo de um ficheiro json resultante de uma configuração inicial na plataforma por parte do grupo de trabalho. Esse ficheiro json foi então convertido pelo grupo para um ficheiro yml, sendo este invocado na task "create kubernetes monitoring dashboards" criada no role **monitoring_dashboards**

Inicialmente foi criada a disponibilização de dashboards com métricas relevantes que vão ser essenciais para permitir futuramente uma avaliação experimental sobre a o funcionamento da aplicação, com os seguintes nomes:

- Utilização do PV pelo Ghost – Foi tomada a decisão de monitorizar a utilização do persistente volume pelo Ghost de forma a poder acompanhar a forma como este recurso era utilizado, de forma a poder averiguar se este precisaria ou não de ser escalado.
- Utilização do PV pela BD.
- Bytes recebidos pela pod do Ghost – A monitorização dos bytes recebidos pela pod do Ghost é útil para o administrador ter noção da escala de dados a ser entregues no Pod para processamento num dado momento
- Bytes recebidos pela pod do MySQL
- Fração de uso do PV pelo Ghost – A monitorização da fração de uso do persistent volume é importante para controlar a saúde do sistema, sendo uma métrica complementar à primeira listada.
- Fração de uso do PV pelo MySQL
- Tempo de uso do CPU pelo container Ghost – Esta métrica permite averiguar em tempo real o impacto que o container está a ter na sua unidade de processamento, de forma a perceber se é necessário escalar horizontalmente os recursos em uso.
- Tempo de uso do CPU pelo container MySQL
- Bytes transmitidos para o Pod Ghost– Permite monitorizar a taxa de transmissão dos dados para fora da pod, de forma a, complementarmente à métrica de Bytes recebidos, poder avaliar se existe um bottleneck no funcionamento da aplicação
- Bytes transmitidos para o Pod MySQL
- RTT Latencies – Esta métrica permite acompanhar o tempo de resposta a pedidos ao longo do funcionamento do sistema. Caso a latência comece a aumentar então o administrador do sistema perceberá que o sistema está a entrar em estado de thrashing.

O conjunto destas métricas consegue então oferecer uma visão detalhada sobre várias vertentes do sistema, de forma a informar o administrador do estado dos vários recursos implementados. Isto permitirá informar decisões sobre alocação de mais recursos se necessário, ou redução dos mesmos, se notável que estes não estão a ser aproveitados por inteiro.

Para complementar as dashboards, foi adicionada à role **monitoring_dashboards** a task "create kubernetes monitoring policy". Esta executa um comando que configura a criação de uma política de monitorização que irá alertar o administrador caso a utilização do CPU ultrapasse um threshold de 80% durante um período contínuo de 5 minutos, com base num

ficheiro de configuração json. Esta política tem associada ainda a dashboard de alerta “Exceeded CPU Utilisation”. Esta seria útil de modo a ser combinada com o ajuste de réplicas da aplicação web, por exemplo, para lançar uma nova réplica caso a política se verificasse, tomando a aplicação características de escalabilidade, face a uma maior utilização de clientes da aplicação.

6. Avaliação Experimental

Esta tarefa do projeto teve como objetivo testar e validar experimentalmente a correta instalação da aplicação do Ghost e respetiva Base de Dados e dados persistentes. Desta forma, tendo por base os testes já disponibilizados pelos docentes da Unidade Curricular que se concentram na testagem da disponibilidade da aplicação Ghost, efetuando um pedido HTTP GET à página principal do Blog, aquando do *deployment* inicial e um segundo em que se removem os dados persistentes, foi necessário ter um pensamento crítico no sentido de verificar o que teria de ser testado.

Os testes que idealizamos consistiram em recorrer à Admin API disponibilizada pelo Ghost e através desta fazer autenticação através de uma sessão. Esta autenticação requer um email e uma password e, de acordo com o *role* do mesmo, este teria acesso às ações e *endpoints* da API. Aquando do estabelecimento da sessão, os dados de autenticação seriam substituídos por um cookie que iria ser usado para manter a sessão. Conseguindo extrair a string do cookie poderíamos assim fazer um conjunto de operações vastas ao associá-la ao header dos pedidos POST. A operações que pretendemos fazer consiste na criação de um post. O teste corre devidamente.

7. Conclusão

Este trabalho é o culminar de um estudo minucioso que exigiu uma análise e uma reflexão profunda sobre a matéria, com destaque para a automatização da configuração e deployment em pods, que, apesar da enorme e variada quantidade de informação disponível na internet, arranjar fontes adequadas ao contexto em que o projeto se insere tornou-se uma tarefa árdua, pelo que houve a necessidade de discutir algumas dúvidas com o professor para nos esclarecer em relação a algumas dúvidas que nos foram surgindo ao longo do desenvolvimento da solução.

Como trabalho futuro, idealmente, gostávamos de completar a avaliação experimental com todos os testes possíveis (mostrando todas as funcionalidades ativas) como também a configuração de mais alertas de threshold e consequentemente implementá-los na replicação dinâmica e automática.

Não obstante, podemos confiantemente dizer que aprendemos bastante acerca do que nos foi lecionado nas últimas semanas contribuindo, assim, para estarmos mais motivados nos próximos desafios.