



**Relatório de Situação do Projeto**(1ª Avaliação Periódica)

# Grupo 05

|  |  |  |
| --- | --- | --- |
|  | Nome | Número |
| 1 | André Silva | 58644 |
| 2 | Bernardo Caldas | 58650 |
| 3 | David Carreira | 58629 |
| 4 | Martim Emauz | 58668 |
| 5 | Miguel Martins | 58661 |
| 6 | Tomás Alves | 58633 |

# Arquitetura

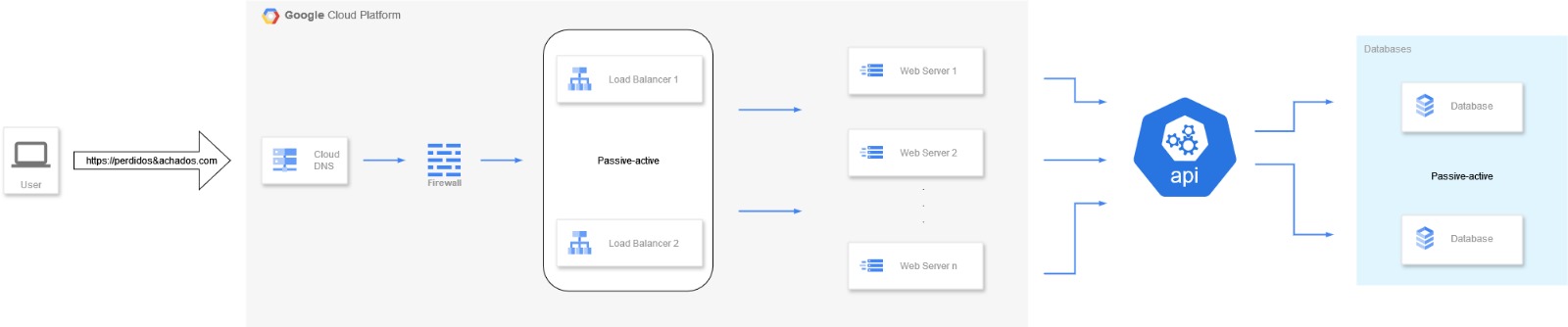
A nossa arquitetura consistirá essencialmente de 2 máquinas virtuais para os 2 balanceadores de carga, 2 máquinas virtuais alocadas horizontalmente para os servidores web (que poderão ser escaladas para ter mais instâncias caso a sobrecarga das iniciais o exigir) e duas bases de dados. Os pedidos serão enviados através de um endereço DNS para os balanceadores, que por sua vez reencaminham os pedidos equitativamente pelos servidores. Os servidores não guardam estado e interagem com a base de dados para registar e aceder a informações.

Figura 1 - Arquitetura do sistema

## Design da arquitetura distribuída

O sistema distribui várias componentes do seu funcionamento por vários nós, ou instâncias, que não só garante a escalabilidade e tolerância a falhas caso estas aconteçam, como garante uma disponibilidade contínua e eficaz destas. As configurações de alta disponibilidade acrescentam a estes benefícios, e o uso de sub-redes faz com que, caso haja algum problema de disponibilidade dos sistemas, seja resolvido sem que o serviço seja interrompido.

Os balanceadores de carga estarão configurados num modo de alta disponibilidade ativo-passivo, de forma que um destes realize o trabalho da gestão dos recursos por si só, e em caso de falha utilizemos a segunda instância.

Os servidores web, serão configurados de forma que possam processar pedidos em paralelo, e só serão geradas mais instâncias caso seja necessário.

Utilizando a Google Cloud Platform (GCP) e o HAProxy em conjunto, garantimos que os diferentes serviços são geridos eficazmente, e conseguimos instanciar várias máquinas em simultâneo sem problemas. A GCP é um serviço bastante fiável, que nos dá todas as ferramentas que precisamos. O HAProxy, além de fácil de usar, permite-nos configurar detalhadamente os modos de alta disponibilidade que pretendemos usar.

Também com recurso à GCP, estabeleceremos a comunicação através de um endereço DNS único.

A firewall estabelecerá as regras de segurança e será crucial na prevenção de falhas neste aspeto, bem como a filtração do tráfego não legítimo, o que melhora o desempenho do sistema.

Enquanto os clientes comunicam com os balanceadores, numa sub-rede “extnet”, os balanceadores por sua vez comunicam com os servidores numa sub-rede “default”.

As bases de dados serão configuradas num regime ativo-passivo, através do PostgreSQL como os balanceadores de carga, em que a segunda base de dados funcionará caso a primeira falhe.

## Especificar requisitos NF – escalabilidade

-Devem ser usados load balancers para distribuir o volume de tráfego equitativamente pelos containers em uso e/ou ajustar o número de servidores em uso;

-Devem ser simuladas situações como spikes de tráfego ou falhas de servidores para garantir o pleno funcionamento dos load balancers.

-Devem ser monitorizados os níveis de saúde dos sistemas, como o tempo resposta e a taxa de transferência de dados de cada servidor através de scripts.

-Devem ser definidos limites para os tempos de resposta, e para a carga dos servidores sob métricas como uso do CPU e da memória, bem como o número de containers por cada IP.

## Especificar requisitos NF – segurança

-Devem ser configuradas sub-redes de forma a segregar a comunicação interna e externa, e eliminando a dependência da comunicação com os servidores num IP externo;

-Devem ser implementadas políticas de acesso às componentes do sistema como o princípio do menor privilégio, utilizando um sistema hierárquico para reduzir o acesso de dados transversalmente;

-Devem ser gerados e configurados certificados TLS de forma a encriptar a comunicação com os servidores;

-Além da encriptação dos dados, deve ser minimizada a recolha de dados dos utilizadores, e expostas as políticas de recolha e consentimento de dados;

-Devem ser realizados scans e revistas as políticas de acesso semanalmente, de forma a monitorizar a integridade dos sistemas.

## Especificar requisitos NF - tolerância a faltas

-Devem ser isoladas as várias funções do sistema de forma a evitar “cascading failures”;

-Deve ser estabelecido um plano de resposta a fugas de dados, falhas nos componentes ou falhas dos ambientes de trabalho;

-Devem ser configurados alertas através de canais como o email, de fácil acesso, caso exista alguma falha no sistema;

-Deve ser usada uma política de comunicação VRRP (Virtual Router Redundancy Protocol), através de clusters ativo-passivos nos load balancers e bases de dados;

-Devem ser definidos processos de restart automático de componentes em falha.

# Interoperabilidade

A API desenvolvida tem como propósito fazer de ponte de entre o software da aplicação e a base de dados externa ao mesmo, no caso foi utilizado o Swagger Editor para a desenvolver e com isto segue em anexo o link do espaço de desenvolvimento (<https://editor.swagger.io>) bem como um WeTransfer com o conteúdo do código produzido para a API em formato JSON (<https://we.tl/t-iMB14rfqiq>).

# Ambiente de desenvolvimento

Para controlo de versões e partilha de ficheiros criámos um repositório GitHub. Utilizaremos branches pessoais para desenvolver recursos separadamente, cujo nome é o de cada elemento do grupo. Cada funcionalidade ou bug resolvido será desenvolvido na branch de cada um, e integrada na branch principal (main) usando pull requests. Assim ficam também disponíveis revisões de código para os membros das equipas.

Quando uma versão estável da aplicação for alcançada criaremos releases no GitHub, o que nos permitirá marcar versões específicas do nosso código para termos um ponto de referência claro para o estado da aplicação nos vários momentos de desenvolvimento.

Usaremos também ferramentas como a integração contínua e implantação contínua, que inclui a execução de testes automatizados sempre que houver uma nova solicitação de pull.

O link para o repositório é o seguinte: <https://github.com/MartimEmauz/PTI-PTR>

Note-se que o repositório é privado, e por isso, para obter acesso, teremos de adicionar um utilizador à lista de colaboradores do repositório.