

Documentação Técnica de implementação GLPI-NETWORK

Participantes

Arthur de Paula Moreira Alckmin

Miguel Mendes Ferreira

Willian da Silva

Marcus Rocha

INTRODUÇÃO

Este documento corresponde à versão atualizada da documentação de implementação do sistema GLPI-NETWORK na empresa Chaveiro Ferreira. Nele, serão abordados de forma completa a infraestrutura do projeto, as modificações realizadas, o relatório de requisitos e os custos envolvidos.

De forma resumida, o projeto consiste na implementação de um sistema web de helpdesk, cujo objetivo é facilitar o processo de solicitação de atendimentos domiciliares por parte dos clientes da empresa.

Ao longo deste documento, as informações técnicas serão detalhadas e explicadas minuciosamente, de modo que os colaboradores e os responsáveis pela manutenção do sistema possam compreender sua estrutura e funcionamento. Dessa forma, busca-se tornar os processos de manutenção mais ágeis, eficientes e reduzir possíveis transtornos futuros.

O QUE É O GLPI

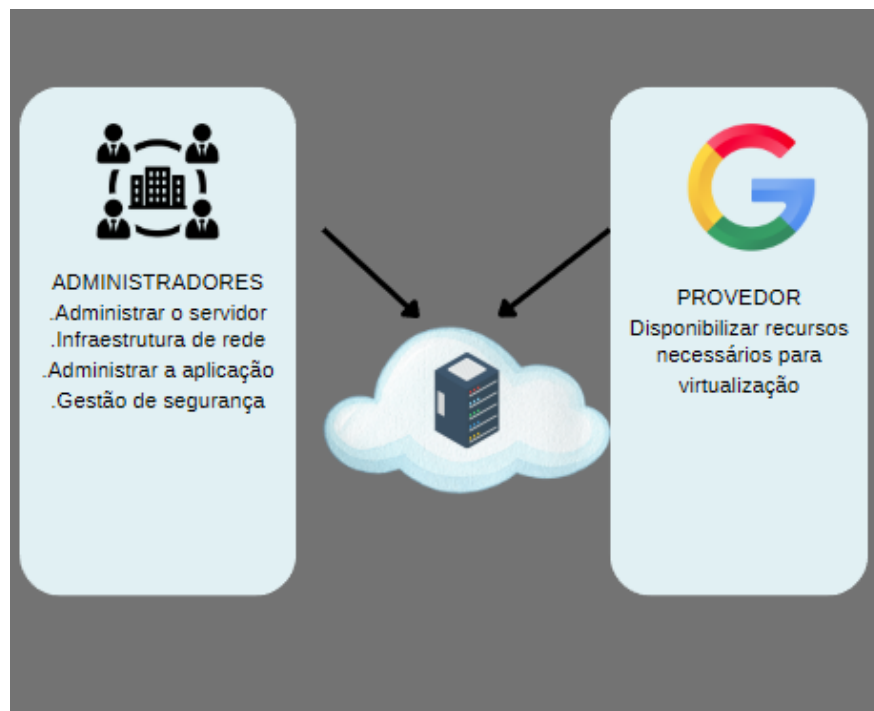
O GLPI-Network é um sistema open source voltado para Help Desk, gestão de serviços e inventário de ativos de TI. Trata-se de uma solução robusta e amplamente utilizada na área de tecnologia. Por ser um software de código aberto, permite a realização de customizações, adaptações e ampliações, possibilitando sua utilização não apenas no setor de TI, mas também em outros setores que trabalham com atendimento, suporte e gestão de demanda

O sistema conta com funcionalidades como abertura e gestão de chamados, controle de SLA, base de conhecimento, gestão de ativos, controle de usuários, geração de relatórios, entre outros.

A escolha dessa solução foi motivada pela necessidade da empresa parceira em otimizar a gestão das solicitações de serviço, proporcionando um controle mais eficiente, organizado e centralizado dos atendimentos recebidos.

ESTRUTURA DO PROJETO

Para implementação da solução de uma maneira mais dinâmica e simples foi utilizado o provedor de hospedagem em nuvem Google Cloud em um plano LAAS para garantir disponibilidade de recursos de maneira escalável, na qual administramos o servidor por completo e utilizamos os recursos da provedora apenas, o que garante que a empresa só arcará com os custos dos recursos que utilizarem. Segue abaixo uma imagem que explica a infraestrutura de hospedagem e função da provedora e dos administradores(participantes do projeto).



No plano adotado LAAS adotado, a divisão das funções dos administradores e da provedora funciona da seguinte forma:

ADMINISTRADORES: Realizam todo processo de configuração do servidor, aplicação, firewall, rede e storage.

PROVEDOR: Disponibiliza os recursos necessários para criação da máquina virtual, endereço ip interno e externo, memória ram, processador e storage.

CONFIGURAÇÃO FÍSICA DO SERVIDOR

.CPU - Intel Broadwell 64 bits
.DISCO - SSD Sata 50GB
.SISTEMA OPERACIONAL - ubuntu-2004-focal-v20250425(Canonical, Ubuntu, 20.04 LTS, amd64 focal image built on 2025-04-25)
.MEMÓRIA RAM - 2,8 GB
.ZONA - Southamerica-east1-c
.IP Público - Automático(definido pela provedora)

Tal configuração física foi realizada com intuito de reduzir custos de hospedagem e garantir funcionamento da aplicação de maneira fluida. Como estamos em fase de homologação, há possibilidade de utilizar uma configuração mais simples.

CUSTO MENSAL: R\$258,00

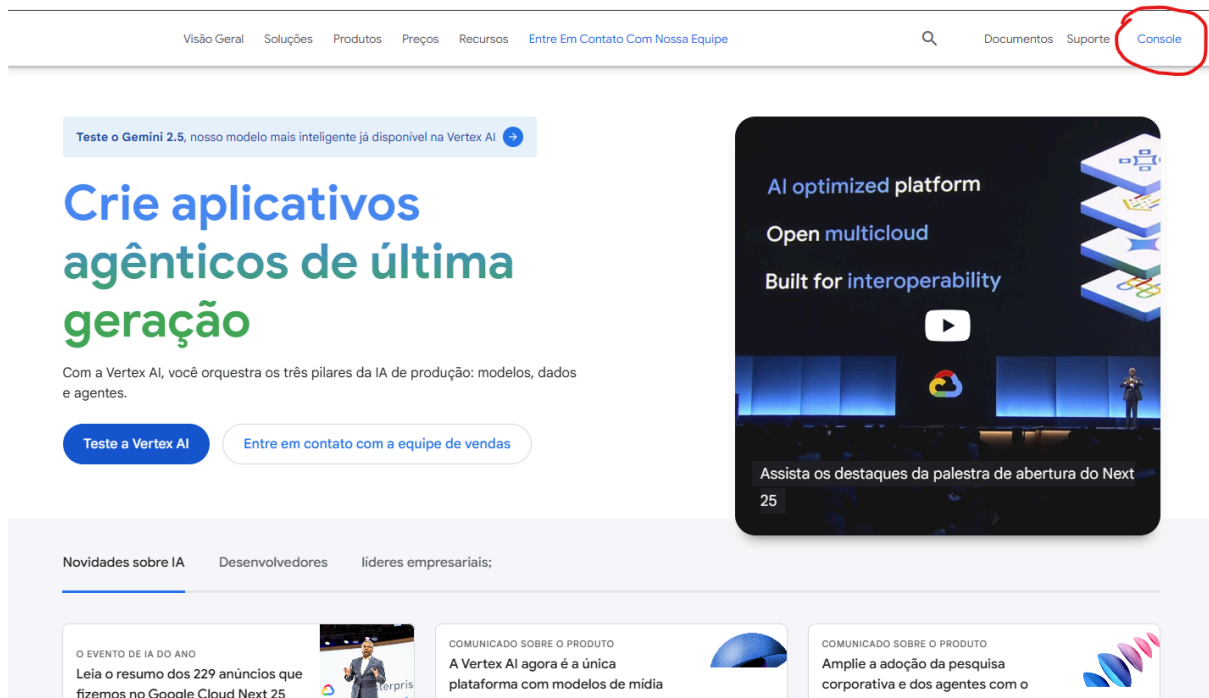
Como dito anteriormente, por estar em fase de homologação a nossa meta é reduzir ainda mais os custos e não perder qualidade de serviço, a empresa parceira não está tendo custo nenhum nessa fase pois estamos utilizando créditos disponibilizados pela própria gcloud.

CONFIGURAÇÃO INICIAL DA VM

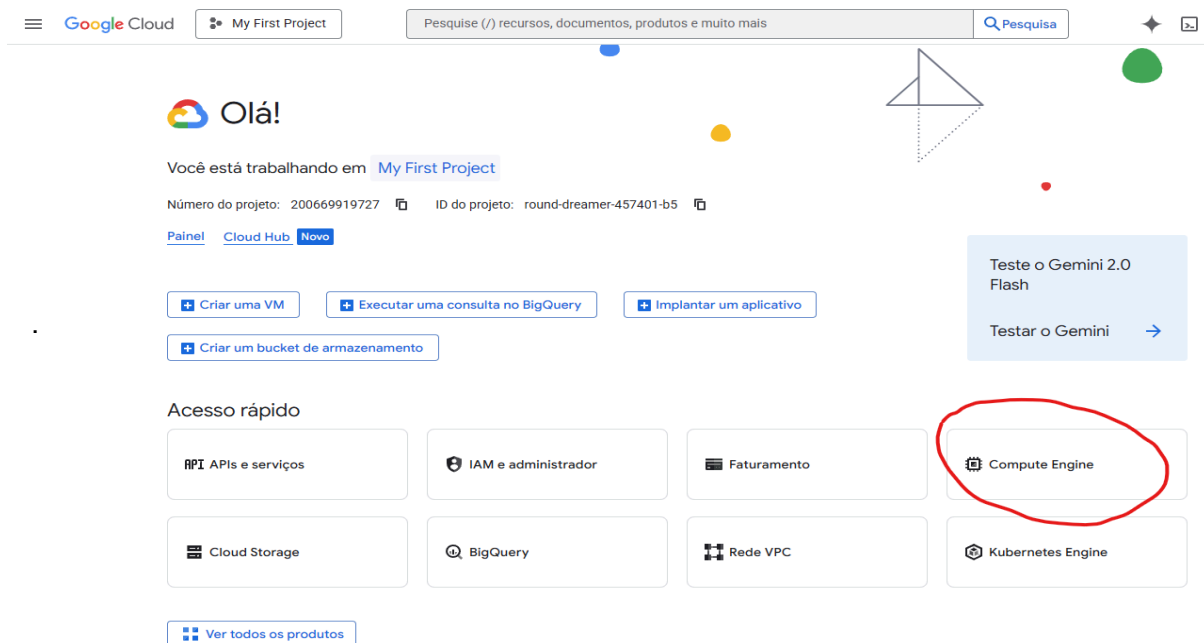
Nessa etapa iremos realizar o tutorial de criação da máquina virtual no ambiente GUI do console google cloud.

.Acesse o console da gcloud para criar a máquina virtual:

LINK DE ACESSO: <https://cloud.google.com/?hl=pt-BR>



.Após acesso do administrador da conta google, acesse o COMPUTE ENGINE(Ferramenta de criação de VM'S da gcloud):



No COMPUTE ENGINE clique na opção “Criar instância”

Menu de navegação (.) Compute engine

Visão geral **Criar instância** Ver instâncias

Esta página foi útil? [👍](#) [👎](#) [Atualizar](#)

Informações gerais

- Visão geral dos riscos ...
- Máquinas virtuais
 - Migrate to Virtual Mach...
 - Instâncias de VM
 - Modelos de instâncias
 - Nós de locatário indivi...
 - Imagens de máquina
 - TPUs
 - Descontos por compro...
- Reservas
- Storage
 - Discos
 - Pools de armazename...
 - Snapshots
 - Imagens
- Marketplace
- Notas de lançamento

Integridade do sistema

Incidentes do Google Cloud

Começar a usar o Service Health

Ative a API Service Health para ver os detalhes de incidentes futuros.

[Ativar a API Service Health](#)

[→ Acessar Personalized Service Health](#)

Uso

Volume do recurso

Atualização em agora, mostrando a tendência diária

Total de VMs	Grupos de instâncias
1	0

Discos	Snapshots
1	0

Imagens	Reservas
0	0

Uso da CPU (cinc...)

UTC-3 22:30 23:00

glpi-serv: 2,76%

1 hora

[→ Ver detalhes em Observabilidade](#)

<https://console.cloud.google.com/compute?hl=pt-BR&inv=1&inv=AbzWnw&project=...> Inventário de ativos

Insira todas as configurações físicas/lógicas essenciais para criação da máquina virtual:

[←](#) Criar uma instância [+ Criar VM de...](#) [Código equivalente](#) [<](#)

Configuração da máquina

Nome *
instance-20250606-020917

Região *
us-central1 (Iowa)

Zona *
Tudo

A região é permanente.

Google vai escolher uma zona para você, ampliando a capacidade de acesso da VM. A zona é permanente.

NOVIDADE: pré-lançamento da série de máquinas C4D de uso geral

✓ Teste a nova série de máquinas C4D com o melhor custo-benefício e recursos [Faça um teste agora](#)

☒ **Uso geral** ☐ Otimização para computação ☐ Otimização de memória ☐ Otimizado para armazenamento ☐ GPUs

Tipos de máquinas para cargas de trabalho comuns, otimizadas para custo e flexibilidade

Series	Descrição	vCPUs	Memory	Plata
<input type="radio"/> C4	Desempenho consistente e constante	2 - 192	4 a 1.488 GB	Intel E
<input type="radio"/> C4A	Alto desempenho consistente baseado em arm	1 - 72	2 a 576 GB	Googl
<input type="radio"/> C4D	Alto desempenho consistente	2 - 384	3 a 3.072 GB	AMD
<input type="radio"/> N4	Flexível e econômico	2 - 80	4 a 640 GB	Intel E
<input type="radio"/> C3	Desempenho consistente constante	4 - 192	8 a 1.536 GB	Intel E
<input type="radio"/> C3D	Desempenho consistente e constante	4 - 360	8 a 2.880 GB	AMD
<input checked="" type="radio"/> E2	Computação diária de baixo custo	0.25 - 32	1 a 128 GB	Intel E

[Criar](#) [Cancelar](#) [Código equivalente](#)

Estimativa mensal

US\$ 25,46

Cerca de US\$ 0,03 por hora

Pague pelo que usar: faturamento por segundo e sem custos iniciais

Item	Estimativa mensal
2 vCPU + 4 GB memory	US\$ 24,46
Disco permanente balanceado com 10 GB	US\$ 1,00
Logging	O custo varia
Monitoring	O custo varia
Programação de snapshots	O custo varia
Total	US\$ 25,46

[Preços do Compute Engine](#)

[Preços do produto Operações do Cloud](#)

[Menos](#)

DESCRIÇÃO DA ETAPA: Essa etapa define todas as configurações que sua instância terá relacionadas a recurso de máquina, sistema operacional, rede e segurança, então é de suma importância que as configurações sejam preenchidas de acordo com a necessidade do serviço que a máquina fará, no caso do projeto referido nesse documento, a configuração da instância ficou da seguinte maneira:

DESCRIÇÃO DA ETAPA: Nessa etapa o usuário google que inicia a sessão do gcloud e tem permissão de administrador para gerenciar aquela instância deve autorizar a conexão via SSH web e assim acenderá ao terminal do servidor. O usuário de acesso já é ADMIN(ROOT) do sistema, então os comandos de admin não precisam ser antecidos do comando SUDO.

INFRAESTRUTURA LÓGICA

Para instalação do sistema e disponibilidade segura do mesmo na web, utilizamos o método de containers docker. Com o docker configuramos a aplicação(GLPI), o banco de dados da aplicação(MariaDB) o subdomínio gratuito(DUCK DNS), o proxy reverso(NGIX PROXY MANAGER) e o gerenciador de containers(PORTAINER) abaixo será anexado as etapas de configuração de cada serviço que explicará o fluxo e a função de cada um serviços executados no servidor:

.Atualize o sistema:

#Antes de tudo atualize o gerenciador de pacotes do sistema e outras dependências com o comando:

```
sudo apt update && sudo apt upgrade -y
```

.Instale o docker com o comando:

```
sudo apt install docker
```

DESCRIÇÃO DA ETAPA: Nessa etapa será realizada a instalação do docker, nela o serviço que possibilitará o funcionamento geral da aplicação é instalado.

.Instale o portainer:

#Crie um volume na partição de armazenamento do sistema para o container do portainer:

```
sudo docker volume create portainer_data
```

#Crie um container para o PORTAINER:

```
sudo docker run -d -p 9000:9000 --name portainer --restart always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:latest
```

DESCRIÇÃO DA ETAPA: Nessa etapa é configurada a interface gráfica que facilitará a administração do sistema. Essa opção é gratuita e foi implementada para facilitar o trabalho e resumir a administração dos containers a poucos cliques.

.Habilitar portas de acesso do portainer no firewall

Como definido na criação do container, o portainer utiliza a porta 9000 do host e direciona para porta 9000 do container, então para acesso da ferramenta via web deve-se permitir o tráfego na porta 9000 da instância criando uma regra de firewall no compute engine:

Segurança da rede

Cloud Armor

Cloud IDS

Cloud NGFW

Componentes comuns

Proxy seguro da Web

Painel de DDoS

Políticas do Cloud Armor

Proteção adaptativa

Nível do serviço Cloud ...

Painel do IDS

Endpoints do IDS

Ameaças do IDS

Painel

Políticas de firewall

Ameaças

Endpoints de firewall

Grupos de endereços

Políticas de firewall

Criar política de firewall

Criar regra de firewall

Saiba mais

Porta SMTP 25 não permitida neste projeto.

Atualizar

Configurar registros

Excluir

Filtro

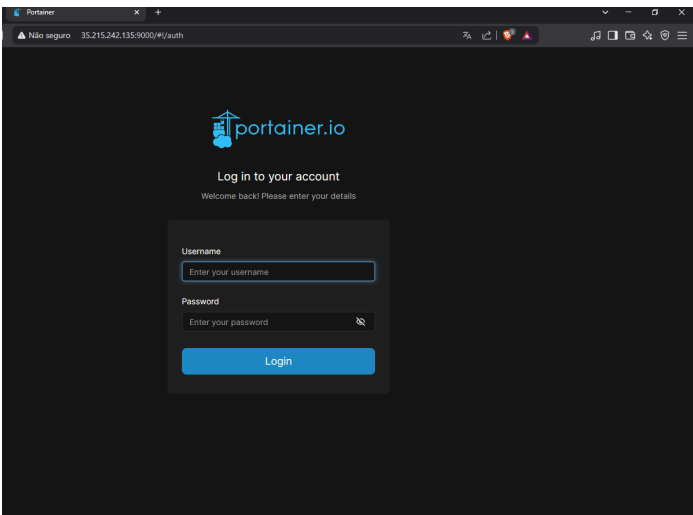
Insira o nome ou o valor da propriedade

	Nome	Tipo	Destinos	Filtros	Protocolos / portas	Ação	Prioridade	Rede	Registros	
<input type="checkbox"/>	allow-portainer-http	Entrada	Aplicar a	Intervalos	tcp:9000	Permitir	1000	default	Ativado	▼
<input type="checkbox"/>	default-allow-http	Entrada	http-server	Intervalos	tcp:80	Permitir	1000	default	Desativado	▼
<input type="checkbox"/>	default-allow-https	Entrada	https-server	Intervalos	tcp:443	Permitir	1000	default	Desativado	▼
<input type="checkbox"/>	default-allow-icmp	Entrada	Aplicar a	Intervalos	icmp	Permitir	65534	default	Desativado	▼
<input type="checkbox"/>	default-allow-internal	Entrada	Aplicar a	Intervalos	tcp:0-65535 udp:0-65535 icmp	Permitir	65534	default	Desativado	▼
<input type="checkbox"/>	default-allow-rdp	Entrada	Aplicar a	Intervalos	tcp:3389	Permitir	65534	default	Desativado	▼
<input type="checkbox"/>	default-allow-ssh	Entrada	Aplicar a	Intervalos	tcp:22	Permitir	65534	default	Desativado	▼

	Nome	Tipo	Destinos	Filtros	Protocolos / portas	Ação	Prioridade	Rede	Registros	
<input checked="" type="checkbox"/>	allow-portainer-http	Entrada	Aplicar a	Intervalos	tcp:9000	Permitir	1000	default	Ativado	▼

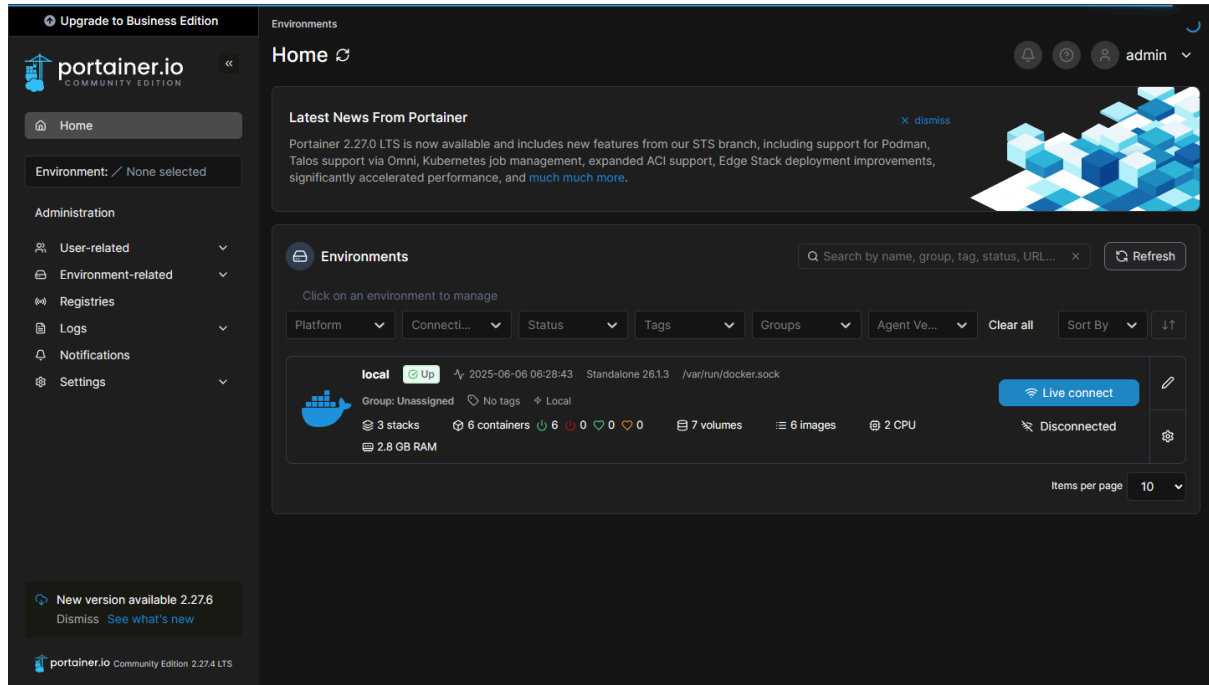
.Acessando o PORTAINER:

Após ativar a regra que libera o tráfego de entrada da porta 9000, acesse o portainer com o IP EXTERNO:9000(Direcionando para porta 9000).



No primeiro acesso o usuário define senha e usuário da aplicação, no caso do projeto descrito as credenciais serão reservadas aos administradores:

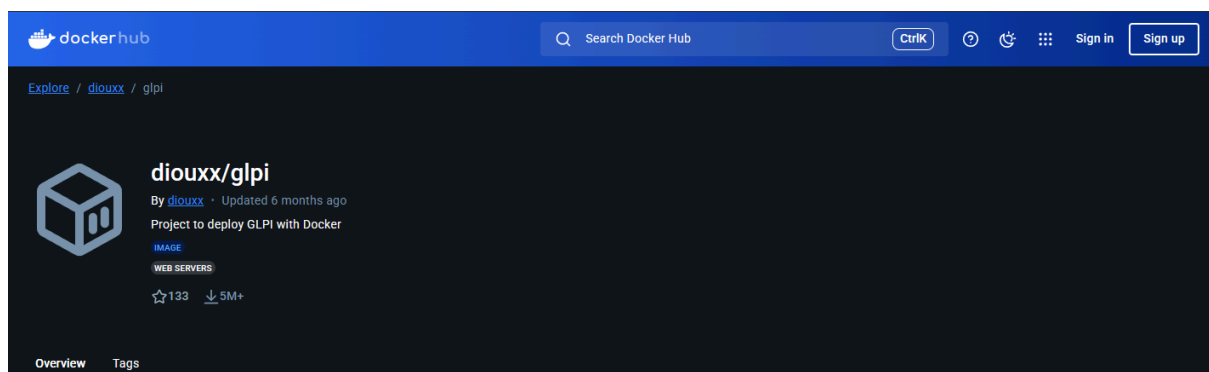
#PAINEL INICIAL



.INSTALAÇÃO DO GLPI

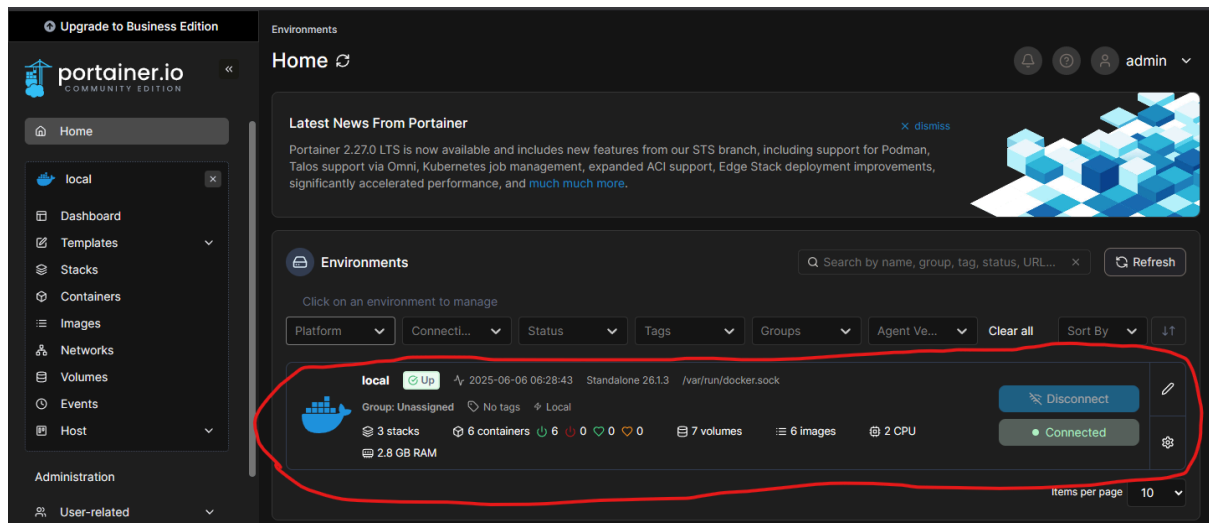
Nessa etapa a aplicação e o banco de dados são instalados via arquivo docker compose fornecido diretamente pela empresa DIOUX.

<https://hub.docker.com/r/dioux/glpi>

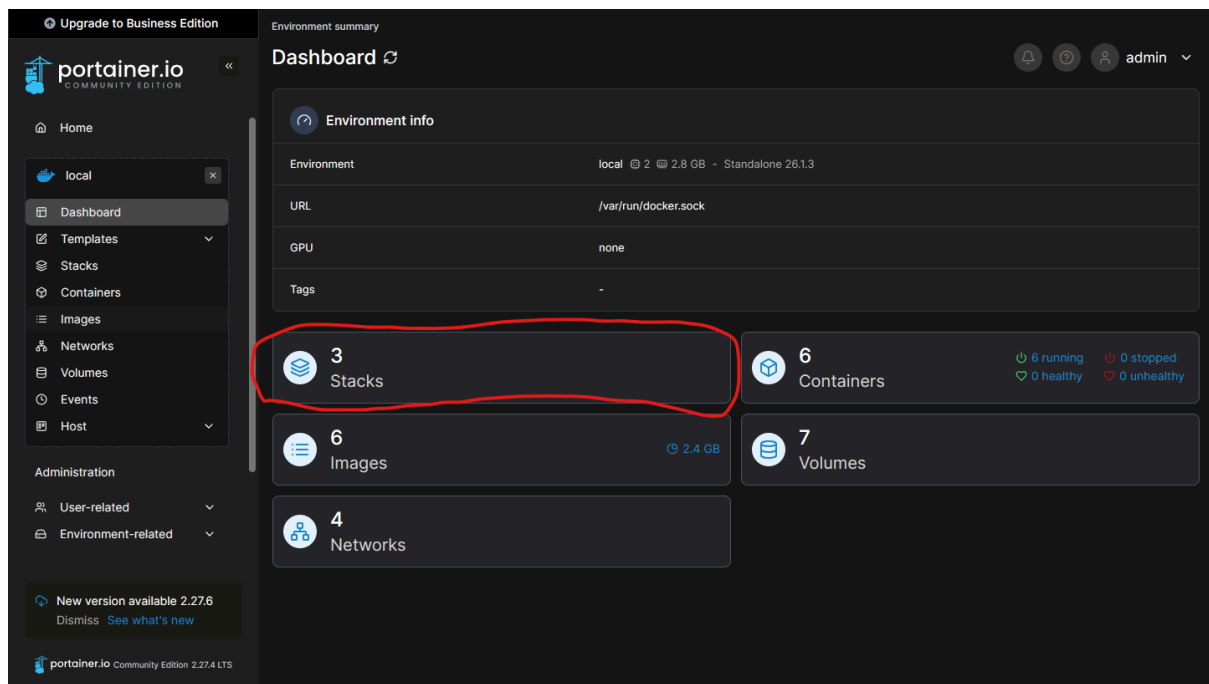


.Criação da pilha(STACK)

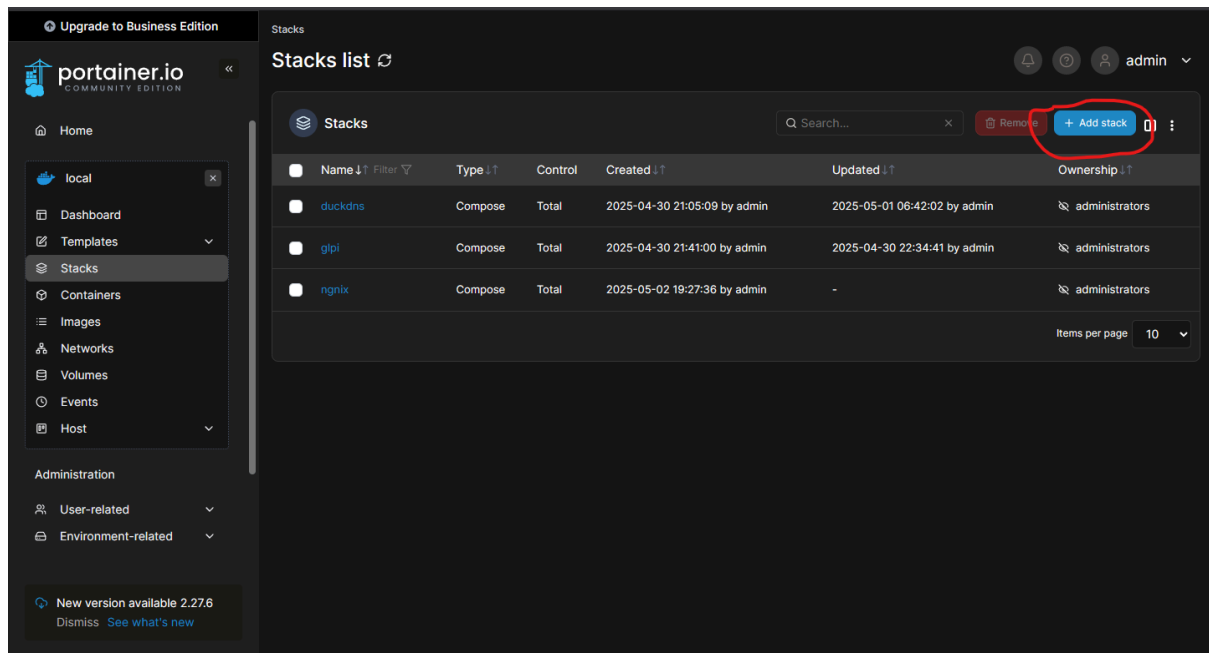
#Selecione o ambiente:



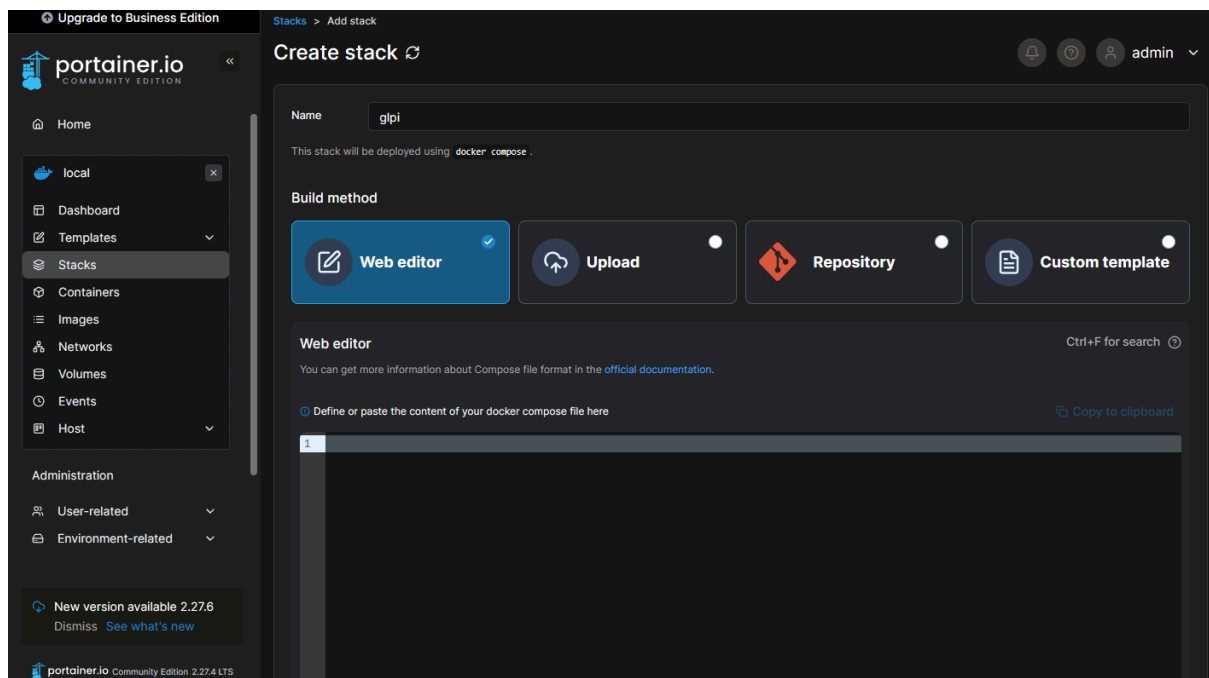
#Clique na opção Stacks no painel direito de menu:



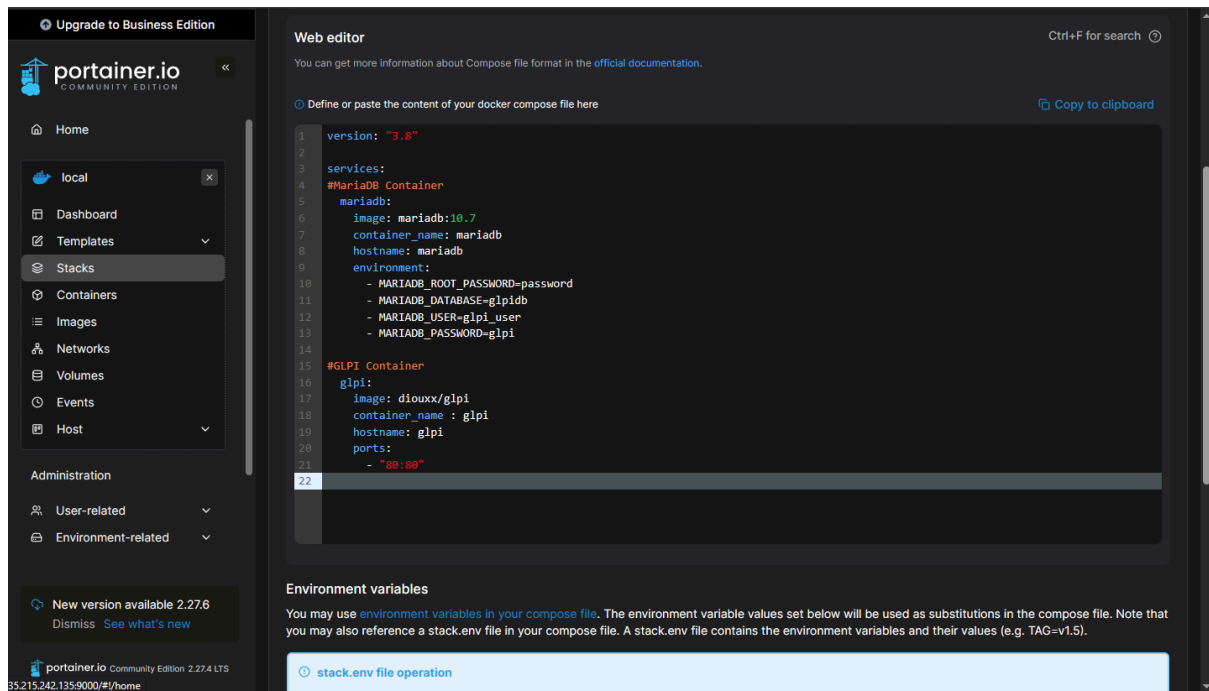
#Clique em adicionar stack



#Defina o nome da STACK(normalmente o nome do serviço)



#Cole e configure o arquivo docker compose fornecido no docker hub da aplicação:



.Instalação do GLPI

Para acessar o GLPI inicialmente utiliza-se o endereço externo apontando para porta 80, no primeiro acesso utilize as credenciais definidas no arquivo docker-compose na criação da stack e instale e integre o banco de dados a aplicação:

Dados necessários:

- MARIADB_ROOT_PASSWORD=password
- MARIADB_DATABASE=glpidb
- MARIADB_USER=glpi_user
- MARIADB_PASSWORD=glpi



DESCRIÇÃO DA ETAPA: Essa etapa de configuração chamada “criação da stack” é basicamente a criação de um pacote com serviços interligados, no caso do projeto referido, a stack cria ao mesmo tempo dois containers interligados, o container do GLPI com definição de HOST e Porta de serviço(Padrão 80) e o banco de dados que é executado em background juntamente com a aplicação. **TODAS AS INFORMAÇÕES NECESSÁRIAS PARA INSTALAÇÃO ESTÃO NO ARQUIVO COMPOSE. ATENÇÃO**

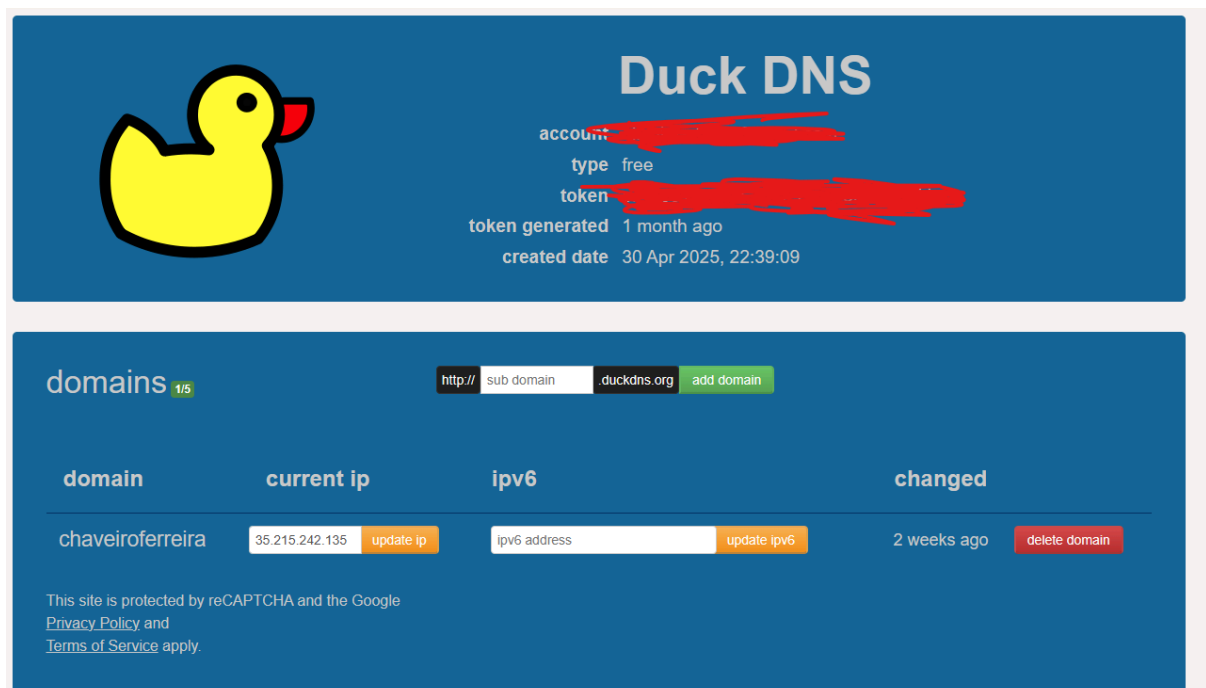
.Instalação do DUCK DNS

Explicação: O duck dns é um servidor de subdomínios, no qual um usuário com uma conta google registra seus subdomínios, aponta para o ip do servidor que contém a aplicação desejada e recebe um token de acesso para cada um dos servidores registrados. No caso do projeto referido, o token de acesso rodando no container duck atualiza o endereço ip do servidor no servidor de subdomínios e faz com que o fato do ip público ser trocado a todo instante pela provedora não impacte na disponibilidade da aplicação:

#Registro do servidor no DUCKDNS

Registre o domínio e o ip do servidor que hospeda a aplicação:

LINK DE ACESSO: <https://www.duckdns.org>



The screenshot shows the Duck DNS website interface. At the top, there is a yellow duck logo on the left and account information on the right. The account information includes: account: [redacted], type: free, token: [redacted], token generated: 1 month ago, and created date: 30 Apr 2025, 22:39:09. Below this, there is a section titled 'domains' with a progress indicator '1/5'. It features a form to add a new domain with fields for 'http://', 'sub domain', and '.duckdns.org', followed by an 'add domain' button. Below the form is a table listing existing domains. The table has columns for 'domain', 'current ip', 'ipv6', and 'changed'. One domain is listed: 'chaveiroferreira' with current ip '35.215.242.135' and a '2 weeks ago' change date. There are buttons for 'update ip', 'update ipv6', and 'delete domain'. At the bottom, there is a reCAPTCHA notice and links to 'Privacy Policy' and 'Terms of Service'.

domain	current ip	ipv6	changed
chaveiroferreira	35.215.242.135 update ip	<input type="text" value="ipv6 address"/> update ipv6	2 weeks ago delete domain

.Configuração do compose e criação da pilha:

```
1 services:
2   duckdns:
3     image: lscr.io/linuxserver/duckdns:latest
4     container_name: duckdns
5     network_mode: host #optional
6     environment:
7       - TZ=America/Sao_Paulo
8       - SUBDOMAINS=chaveiroferreira.duckdns.org
9       - TOKEN=
10      - UPDATE_IP=ipv4 #optional
11      - LOG_FILE=false #optional
12     volumes:
13       - /path/to/duckdns/config:/config #optional
14     restart: unless-stopped
15
```

DESCRIÇÃO DA ETAPA: Nessa etapa é criado o container do DUCKDNS que atua como um conversor de IP para nome de domínio. Nessa etapa a criação da stack utiliza o mesmo processo da criação da stack do GLPI, apenas as informações do docker compose são atualizadas. A partir daqui a aplicação já tem uma URL, para que a mesma funcione iremos para próxima etapa que é a configuração do NGINX servidor web e proxy reverso.

ATENÇÃO:

As tags em azul são parâmetros de configuração do docker compose, os mesmos devem ser editados de acordo com as configurações do serviço.

.INSTALAÇÃO DO NGINX

#Crie um stack com o nginx proxy manager

Link do compose:

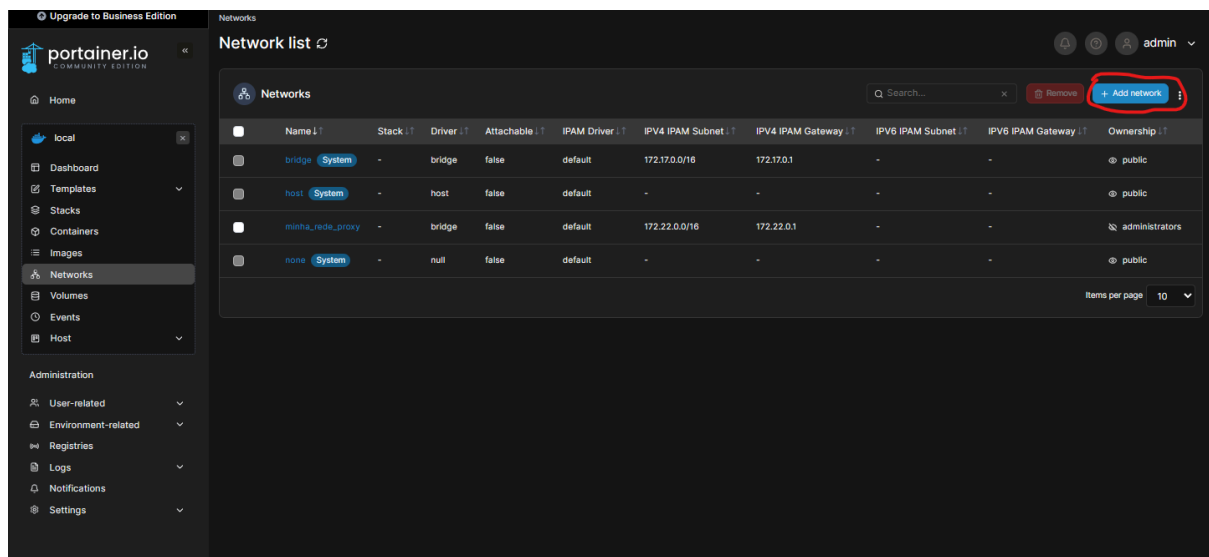
<https://nginxproxymanager.com/setup>

```
1 services:
2   app:
3     image: 'jc21/nginx-proxy-manager:latest'
4     restart: unless-stopped
5     ports:
6       - '80:80'
7       - '443:443'
8       - '81:81'
9     environment:
10      # Mysql/Maria connection parameters:
11      DB_MYSQL_HOST: "db"
12      DB_MYSQL_PORT: 3306
13      DB_MYSQL_USER: "npm"
14      DB_MYSQL_PASSWORD: "npm"
15      DB_MYSQL_NAME: "npm"
16      # Uncomment this if IPV6 is not enabled on your host
17      # DISABLE_IPV6: 'true'
18     volumes:
19       - ./data:/data
20       - ./letsencrypt:/etc/letsencrypt
21     depends_on:
22       - db
23     networks:
24       - minha_rede_proxy
25
```

```
networks:
  minha_rede_proxy:
    external: true
    name: minha_rede_proxy
```

DESCRIÇÃO DA ETAPA: O proxy reverso atua como intermediário entre o cliente e o container da aplicação. Ele recebe o tráfego criptografado (HTTPS) na porta 443, termina a conexão segura, e então encaminha internamente as requisições via HTTP (geralmente na porta 80 ou uma porta customizada) para o container GLPI. Com isso, o tráfego na rede externa permanece protegido por SSL/TLS, enquanto o tráfego interno entre os containers é mantido simples e eficiente. Além disso, o uso de uma rede Docker externa dedicada permite que tanto o proxy quanto os serviços a serem expostos compartilhem o mesmo domínio de comunicação, garantindo roteamento correto e seguro das requisições. Nessa etapa foi criada uma rede interna para os containers, chamada: **minha_rede_proxy** e os outros containers foram adicionados a ela:

#Criação da rede



Clique em “adicionar rede” e insira as informações necessárias para criação. Lembre-se de utilizar o Driver de rede em modo “bridge” para permitir a comunicação entre os containers adicionados:

The screenshot shows the 'Create network' form in Portainer.io. The form is titled 'Create network' and has a 'Name' field with the value 'e.g. myNetwork'. The 'Driver configuration' section shows the 'Driver' set to 'bridge'. The 'IPv4 Network configuration' section has fields for 'Subnet' (e.g. 172.20.0.0/16), 'IP range' (e.g. 172.20.10.128/25), and 'Gateway' (e.g. 172.20.10.11). The 'IPv6 Network configuration' section has fields for 'Subnet' (e.g. 2001:db8::/48), 'IP range' (e.g. 2001:db8::/64), and 'Gateway' (e.g. 2001:db8::1). The 'Advanced configuration' section has checkboxes for 'Isolated network', 'Enable manual container attachment', and 'Access control'. The 'Access control' checkbox is checked. At the bottom, there are two radio buttons: 'Administrators' (selected) and 'Restricted'.

#Inserir o parâmetro de configuração de rede:

Inserir o seguinte parâmetro no compose da aplicação GLPI e do proxy Manager.

```
networks:
  minha_rede_proxy:
    external: true
    name: minha_rede_proxy # O nome da rede que você já criou no Docker
```

.Configuração do NGINX:

#Primeiro libere a porta 81 no firewall da instância:

Políticas de firewall [+ Criar política de firewall](#) [+ Criar regra de firewall](#) [Saiba](#)

Regras de firewall da VPC

As regras de firewall controlam o tráfego de entrada ou saída de uma instância. Por padrão, o tráfego de entrada externo à sua rede é bloqueado. [Saiba mais](#)

Observação: os firewalls do App Engine são gerenciados em [Seção de regras de firewall do App Engine](#).

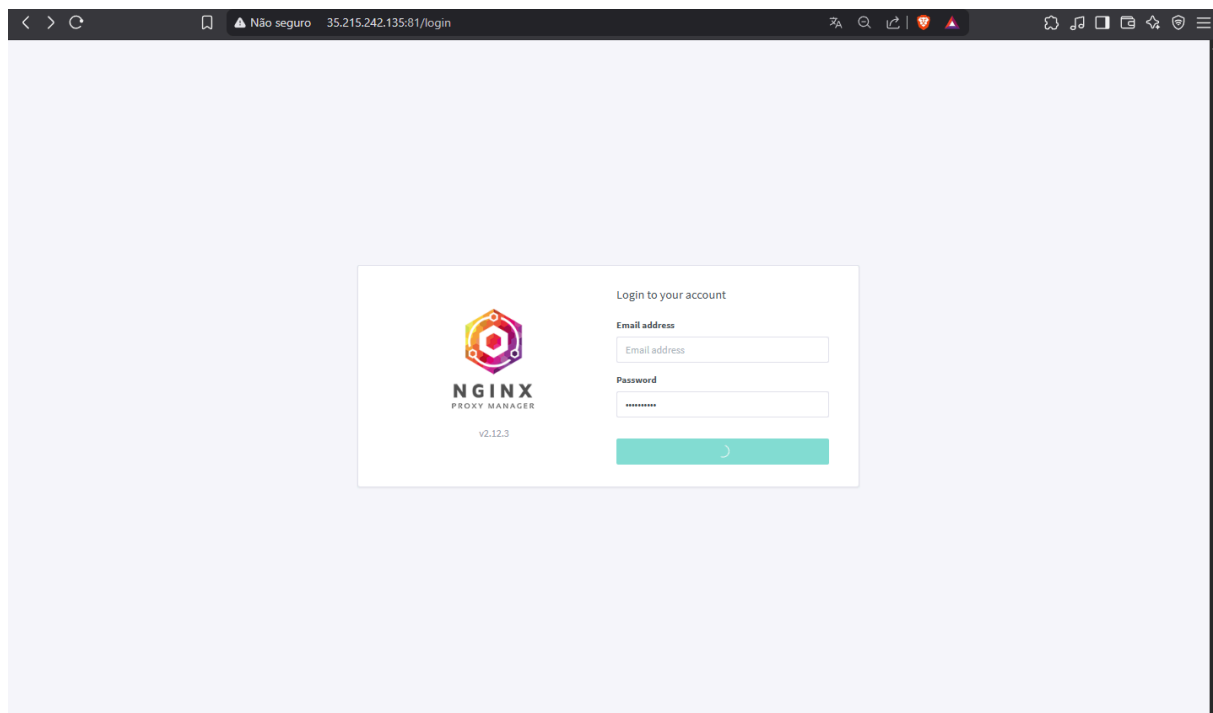
i Porta SMTP 25 não permitida neste projeto. [Saiba mais](#)

[Atualizar](#) [Configurar registros](#) [Excluir](#)

Filtro	Nome	Tipo	Destinos	Filtros	Protocolos / portas	Ação	Prioridade	Rede	Registros
<input type="checkbox"/>	allow-portainer-http	Entrada	Aplicar a	Intervalos	tcp:9000	Permitir	1000	default	Ativado
<input type="checkbox"/>	default-allow-http	Entrada	http-server	Intervalos	tcp:80,81	Permitir	1000	default	Desativado

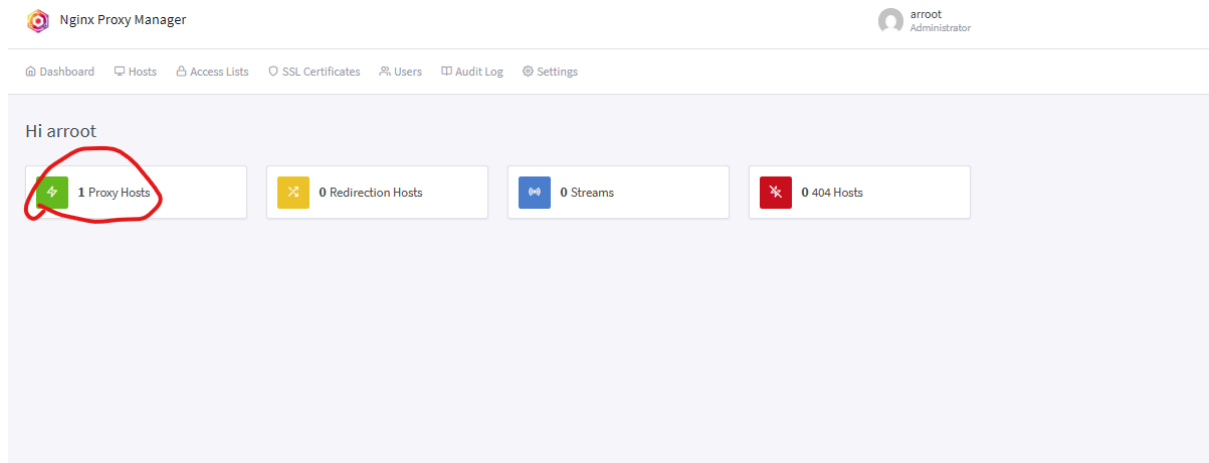
#Acesse o painel web do nginx:

IP EXTERNO:Apontando para porta 81:

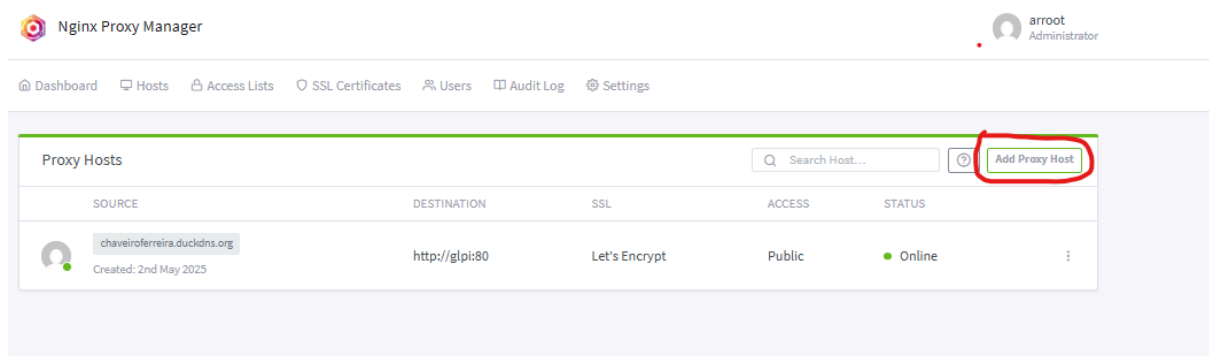


#Após inserir as informações de login(credenciais ficarão resguardadas ao grupo administrador)

#Clique em Proxy Hosts:



#Add Proxy hosts:



#Insira as seguintes informações:

nome de domínio(registrado no DUCK DNS)
hostname do servidor(do container no caso do projeto referido)
Porta Padrão de acesso(GLPI:80)

The screenshot shows the 'Edit Proxy Host' form. It has tabs for Details, Custom locations, SSL, and Advanced. The 'Domain Names' field contains 'chaveiroferreira.duckdns.org'. The 'Scheme' is set to 'http', 'Forward Hostname / IP' is 'glpi', and 'Forward Port' is '80'. There are three toggle switches: 'Cache Assets', 'Block Common Exploits', and 'Websockets Support', all of which are currently turned off. The 'Access List' field contains 'Publicly Accessible'. At the bottom, there are 'Cancel' and 'Save' buttons.

#Solicite um certificado SSL para o domínio desejado:

Edit Proxy Host

Details

Custom locations

SSL

Advanced

SSL Certificate

chaveiroferreira.duckdns.org

Force SSL

HTTP/2 Support

HSTS Enabled ?

HSTS Subdomains

Cancel

Save

#Painel pós criação:

Ngix Proxy Manager

arroot
Administrat

Dashboard

Hosts

Access Lists

SSL Certificates

Users

Audit Log

Settings

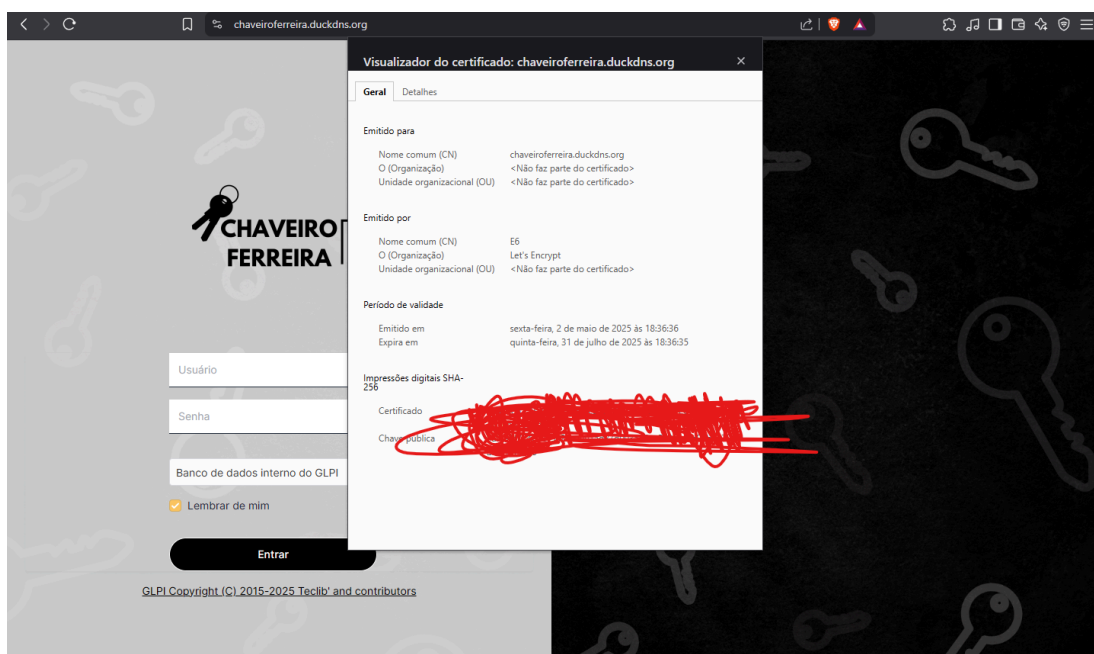
Proxy Hosts

Search Host...

Add Proxy Host

SOURCE	DESTINATION	SSL	ACCESS	STATUS
<div><div></div>chaveiroferreira.duckdns.org</div> <div>Created: 2nd May 2025</div>	http://glpi:80	Let's Encrypt	Public	<div>Online</div>

DESCRIÇÃO DA ETAPA: Nessa etapa o funcionamento da URL já é estabelecido e agora a conexão estabelecida por ela está criptografada, o proxy atua como um filtro de segurança que conduz o usuário até a aplicação de maneira criptografada com certificado SSL válido:



.Fechar as portas do firewall:

Após a configuração é necessário fechar as portas do servidor que ficaram abertasl. Permitindo tráfego apenas na porta 443 que é a porta em que o proxy está atuando.

#Clique na regra

Segurança da rede

Políticas de firewall

Porta SMTP 25 não permitida neste projeto.

Nome	Tipo	Destinos	Filtros	Protocolos / portas	Ação	Prioridade	Rede	Registros
allow-portainer-http	Entrada	Aplicar a	Intervalos	tcp:9000	Permitir	1000	default	Ativado
default-allow-http	Entrada	http-server	Intervalos	tcp:80, 81	Permitir	1000	default	Desativado
default-allow-https	Entrada	https-server	Intervalos	tcp:443	Permitir	1000	default	Desativado
default-allow-icmp	Entrada	Aplicar a	Intervalos	icmp	Permitir	65534	default	Desativado
default-allow-internal	Entrada	Aplicar a	Intervalos	Tudo	Permitir	65534	default	Desativado
default-allow-rdp	Entrada	Aplicar a	Intervalos	Tudo	Permitir	65534	default	Desativado

#Clique em editar:

Detalhes da regra de firewall

allow-portainer-http

Registros

Ativado

Ver no Buscador de registros

Mostrar detalhes de registros

Rede

default

Prioridade

1000

Direção

Entrada

Ação se houver correspondência

Permitir

Tags

Filtros de origem

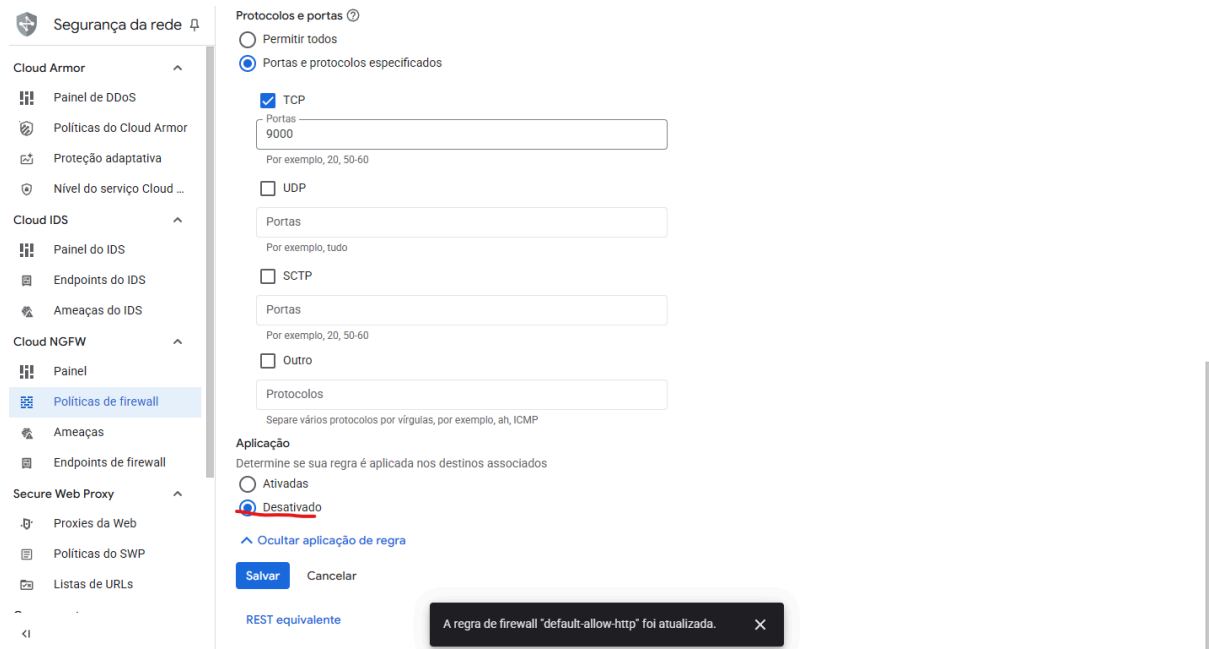
Intervalos de IP 0.0.0.0/0

Protocolos e portas

tcp:9000

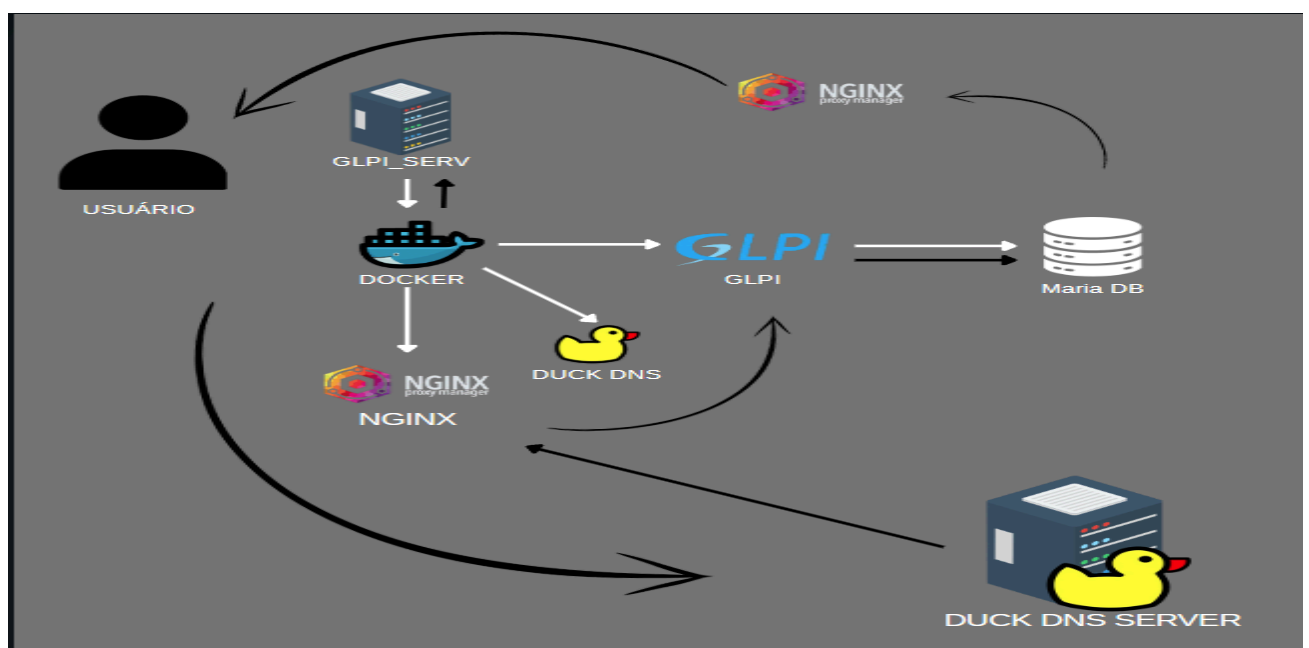
Aplicação

#Desative a regra:



DESCRIÇÃO DA ETAPA: Nessa etapa realizamos a configuração final que disponibiliza a aplicação na web de maneira segura. Como estamos trabalhando com containers, sempre que for necessário realizar a manutenção dos mesmo o admin pode utilizar CLI diretamente no servidor ou GUI pelo portainer, lembrando sempre de ativar a regra de liberação da porta 9000 e fechar a mesma assim que terminar para evitar problemas de acesso indevido e cyberataques.

Arquitetura Interna do Servidor



(Setas Brancas)

#GLPI Container (Docker)

- .Container principal da aplicação GLPI.
- .Executado dentro do ambiente Docker no servidor **GLPI_SERV**.
- .Comunica-se com o banco de dados MariaDB.
- .Expõe a aplicação GLPI para o ambiente interno.

#MariaDB Container

- .Banco de dados usado pela aplicação GLPI.
- .Armazena todas as informações e dados da aplicação GLPI.
- .Comunica-se diretamente com o container GLPI.

#NGINX Proxy Manager (Container)

- .Gerencia os acessos via proxy reverso.
- .Redireciona o tráfego de entrada para o container GLPI.
- .Utiliza configurações definidas para fazer o mapeamento do domínio para o serviço interno.

#DuckDNS (Container)

- .Atualiza dinamicamente o IP público do servidor com o domínio DuckDNS.
- .Garante que o nome de domínio (ex: **seudominio.duckdns.org**) esteja sempre apontando para o IP correto da rede onde está o **GLPI_SERV**.

Fluxo do Usuário (Setas Pretas)

#Usuário → DuckDNS Server

- .O usuário acessa o sistema GLPI por meio de um domínio DuckDNS (ex: **glpi.duckdns.org**).
- .O DuckDNS resolve esse domínio para o IP público atual do servidor.

Usuário → IP do Servidor (GLPI_SERV)

- .Após a resolução DNS, o navegador do usuário se conecta ao IP do **GLPI_SERV**.

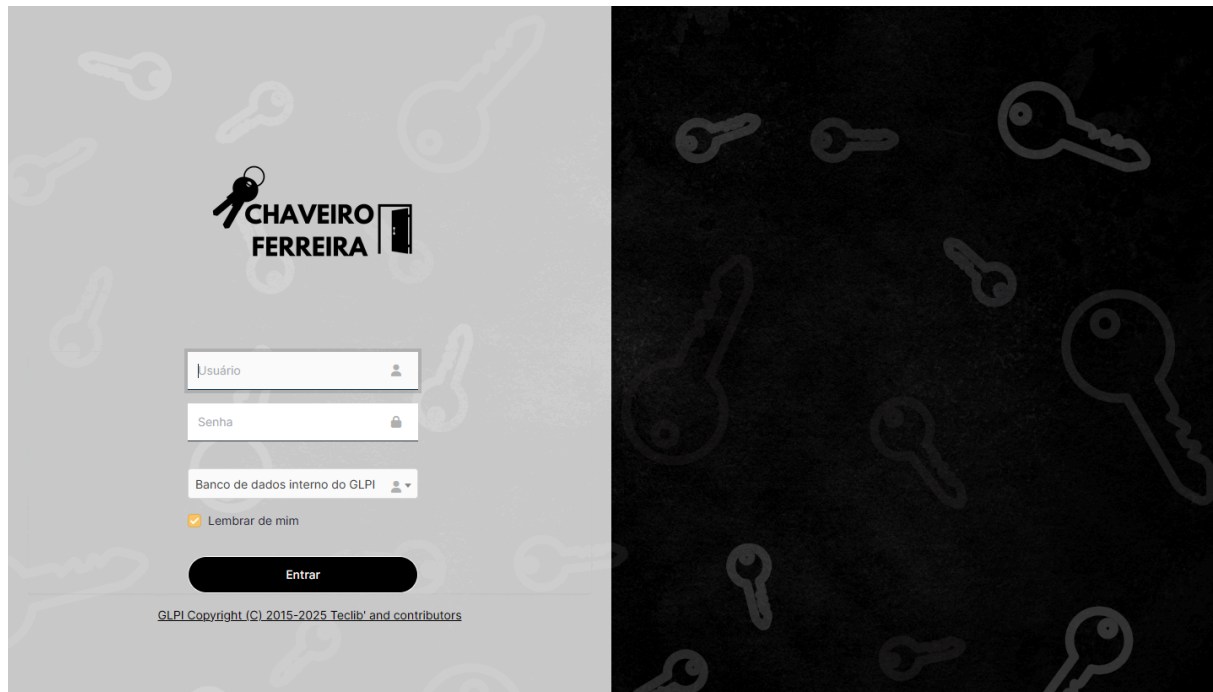
#NGINX Proxy Manager entra em ação

- .O NGINX (rodando em container) recebe a requisição do usuário.
- .Verifica a rota e redireciona a requisição para o container do GLPI.

#GLPI Container responde ao usuário

- .O GLPI responde à requisição, podendo buscar dados do MariaDB.
- .A resposta é encaminhada de volta ao usuário.

LINK DA APLICAÇÃO: chaveiroferreira.duckdns.org



CONCLUSÃO

A implementação do sistema GLPI-Network na infraestrutura da empresa Chaveiro Ferreira representa um importante avanço no processo de digitalização e organização dos atendimentos prestados ao público. Por meio da utilização de containers Docker, foi possível criar um ambiente modular, seguro, de fácil manutenção e escalável, atendendo às necessidades atuais e permitindo futuras adaptações com baixo impacto.

A adoção de ferramentas como Portainer, NGINX Proxy Manager, Duck DNS e MariaDB possibilitou a entrega de um sistema web totalmente funcional, acessível via internet com conexão criptografada (HTTPS), sem custos adicionais à empresa durante o período de homologação, graças aos créditos fornecidos pela Google Cloud. Além disso, o uso de uma infraestrutura em nuvem garante maior disponibilidade, redundância e flexibilidade de gerenciamento, mesmo com recursos limitados.

Com a conclusão deste projeto, a Chaveiro Ferreira passa a contar com um sistema moderno de abertura e gerenciamento de chamados, melhorando a eficiência do atendimento, reduzindo falhas operacionais e criando uma base sólida para o crescimento e expansão tecnológica da empresa. A documentação aqui apresentada garante que futuras manutenções, expansões ou correções possam ser realizadas de maneira rápida e segura, por qualquer membro autorizado da equipe técnica.