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

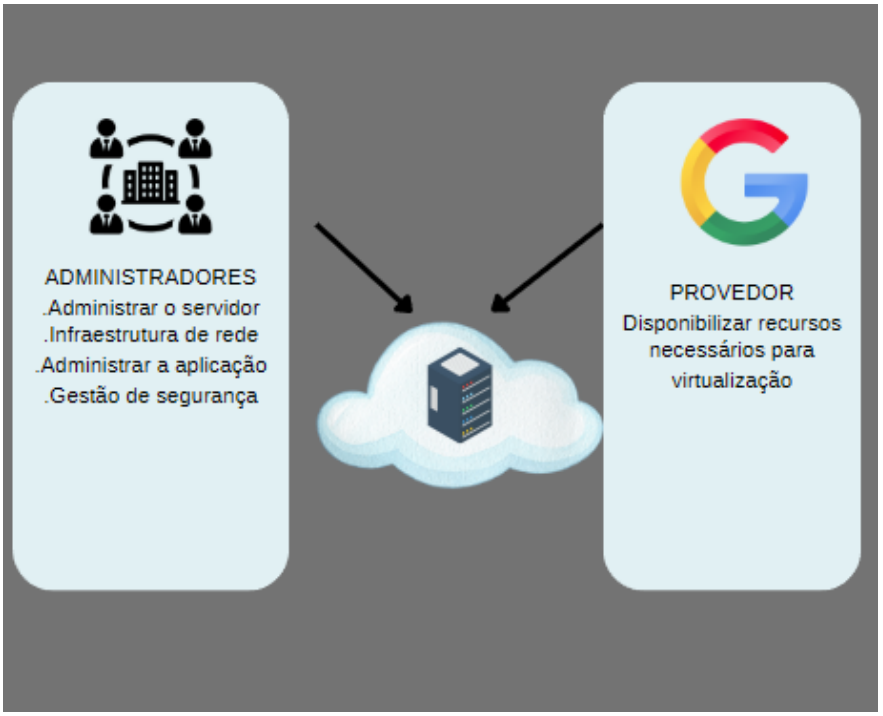
O GLPI-Network é um sistema open source voltado para Help Desk, gestão de serviços e inventário de ativos de TI. Trata-se de uma solução robusta e amplamente utilizada na área de tecnologia. Por ser um software de código aberto, permite a realização de customizações, adaptações e ampliações, possibilitando 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

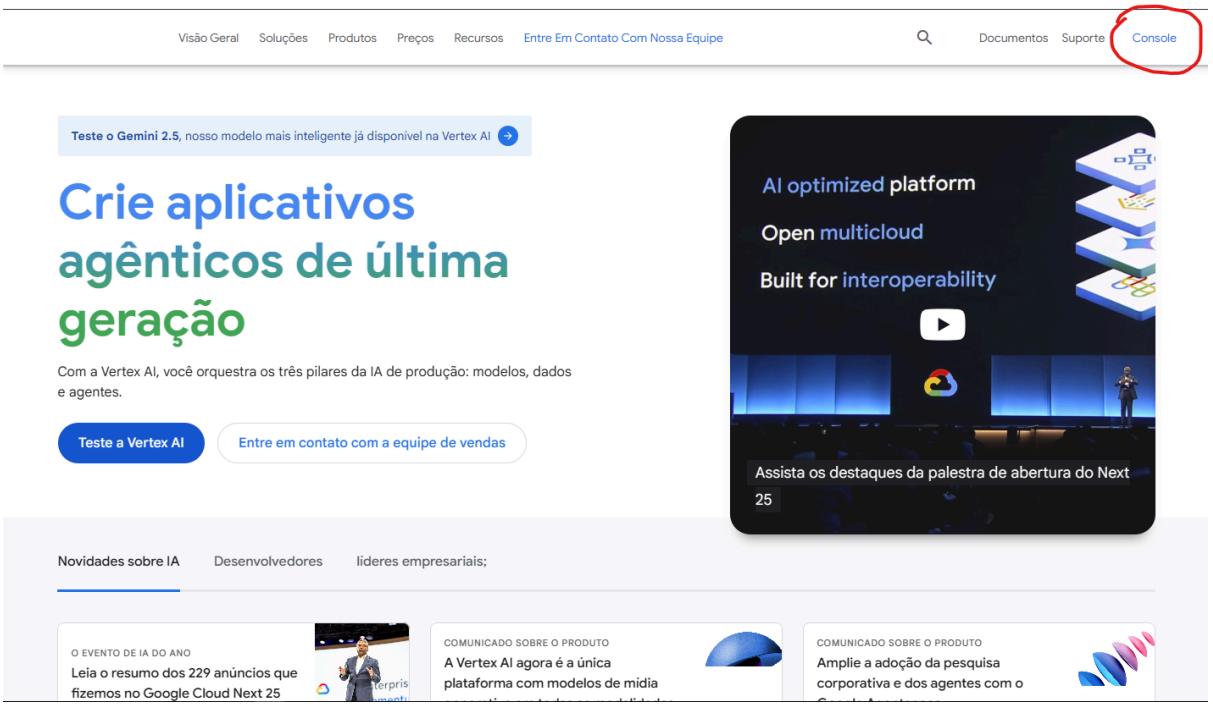
Como dito anteriormente, por estar em fase de homologação a nossa meta é reduzir ainda mais os custos e não perder qualidade de serviço, a empresa parceira não está tendo custo nenhum nessa fase pois estamos utilizando créditos disponibilizados pela própria gcloud.

CONFIGURAÇÃO INICIAL DA VM

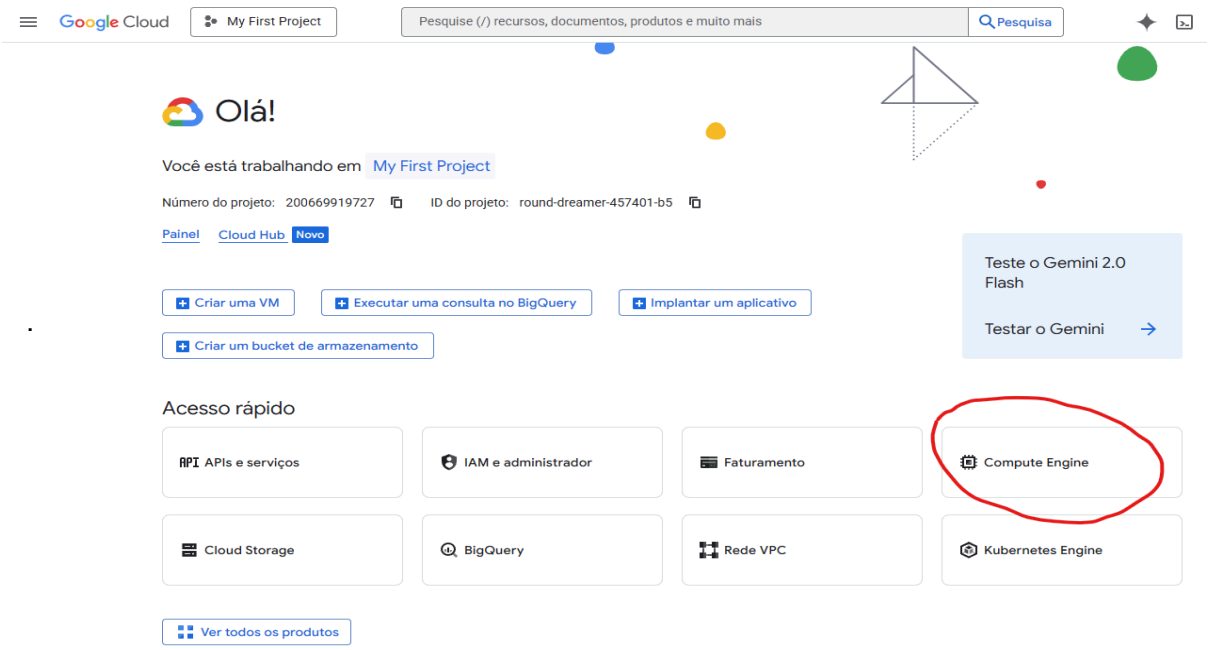
Nessa etapa iremos realizar o tutorial de criação da máquina virtual no ambiente GUI do console google cloud.

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

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

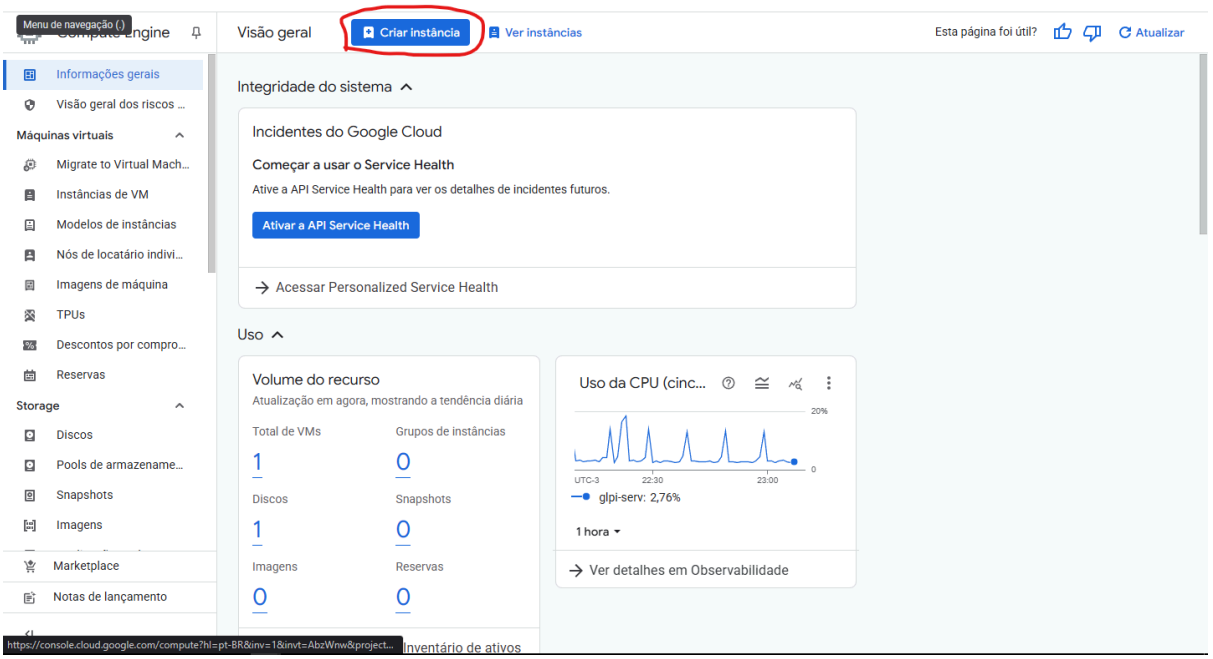


.Após acesso do administrador da conta google, acesse o COMPUTE ENGINE(Ferramenta de criação de VM´S da gcloud):



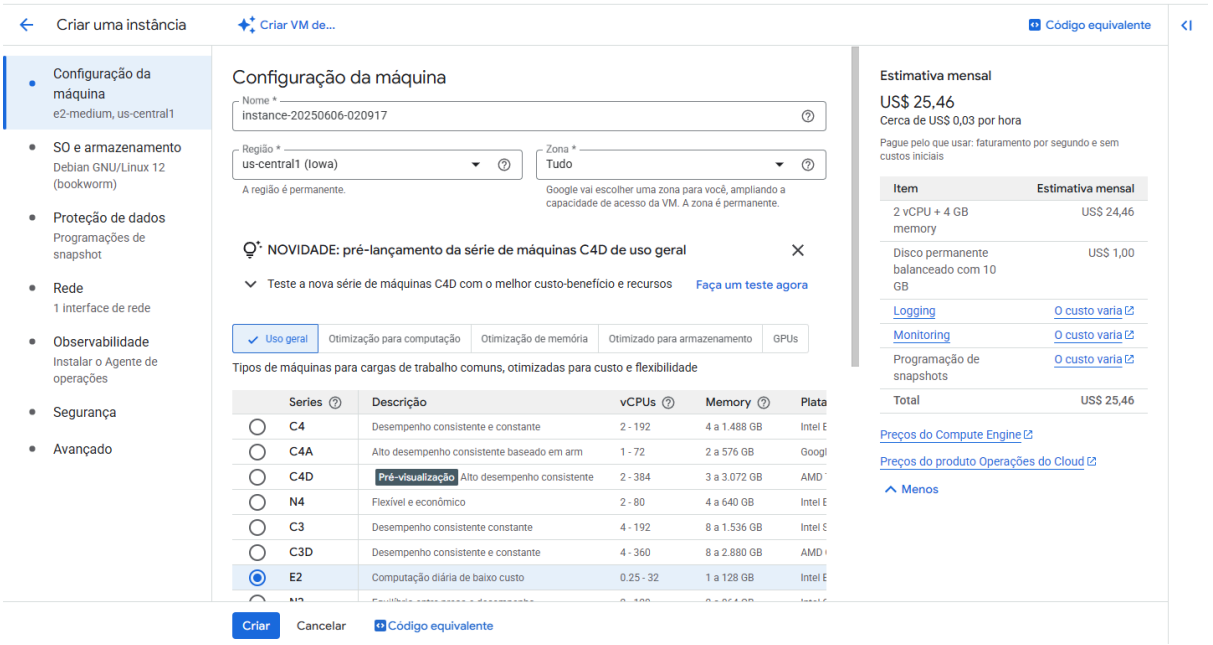
.

.No COMPUTE ENGINE clique na opção “**Criar instância**”



.

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



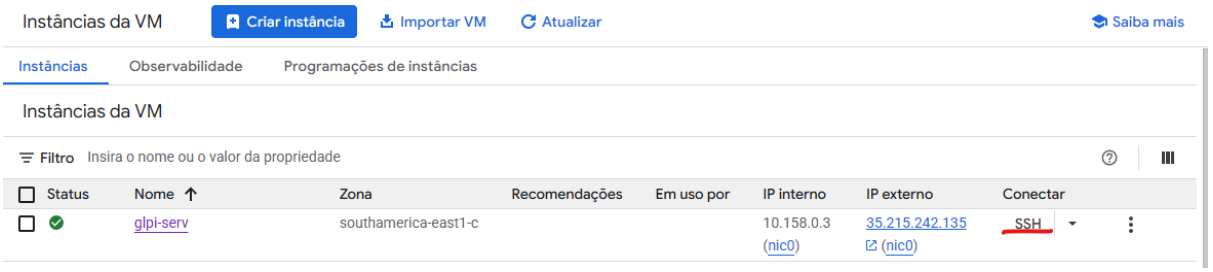
DESCRIÇÃO DA ETAPA: Essa etapa define todas as configurações que sua instância terá relacionadas a recurso de máquina, sistema operacional, rede e segurança, então é de suma importância que as configurações sejam preenchidas dê acordo com a necessidade do serviço que a máquina fará, no caso do projeto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

Série	Sistema Operacional	Disco	Rede	Firewall	Zona
E2	Ubuntu, 20.04 LTS, amd64 focal image	SSD 50GB	IP Público automático	Ativo	southamerica-east1-c

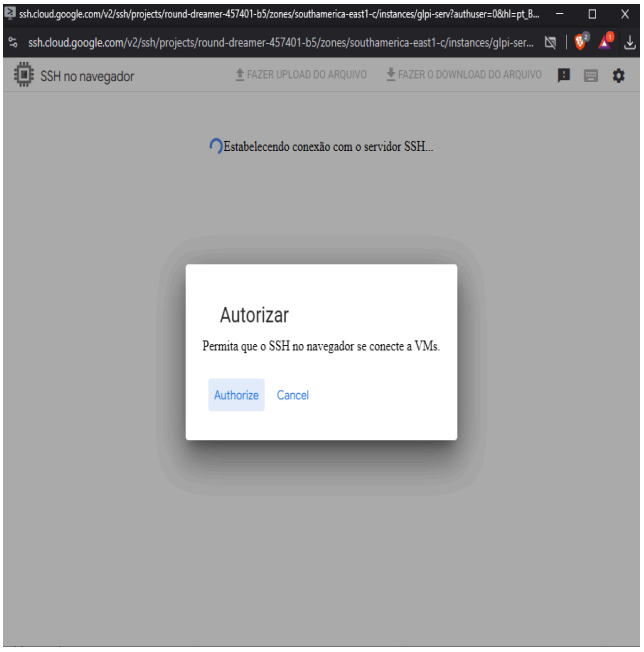
CONFIGURAÇÃO DO SERVIDOR

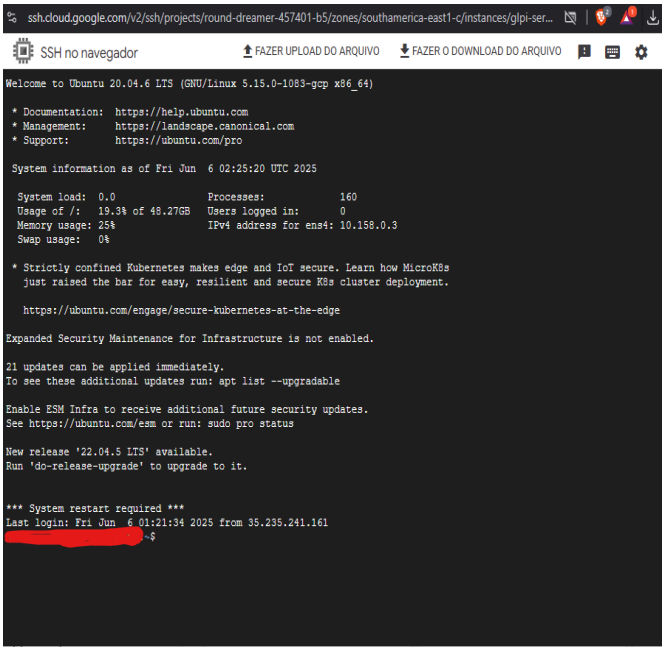
Nessa etapa, como dito antes, foi utilizado um sistema operacional Ubuntu 20.04.LTS fornecido pela Canonical. Para conexão e configuração utiliza-se uma conexão via SSH web totalmente protegida pela provedora de hospedagem. Abaixo está um anexo do método de conexão com terminal do servidor:

.No painel do COMPUTE ENGINE clique na opção SSH de conexão:



.Um terminal web abrirá e o acesso será estabelecido:





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 antecedidos 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
```

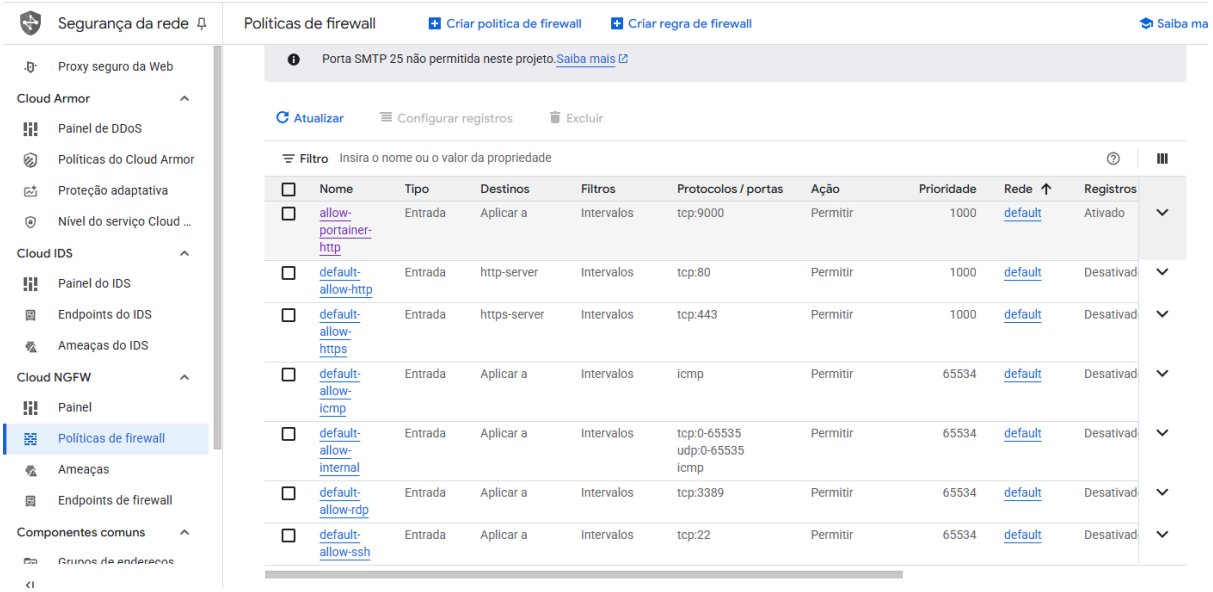
#Crie um container para o PORTAINER:

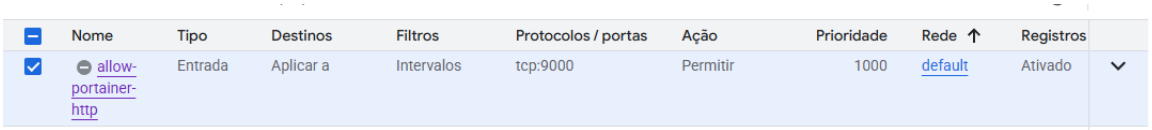
```
sudo docker run -d -p 9000:9000 --name portainer --restart always -v
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data
portainer/portainer-ce:latest
```

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

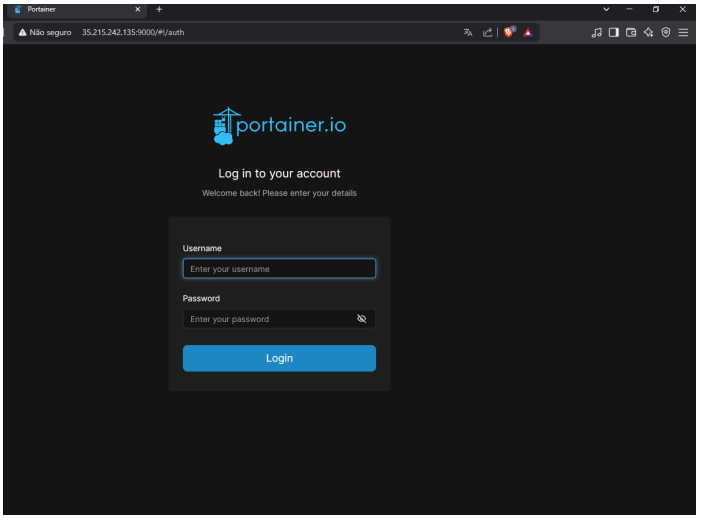
Como definido na criação do container, o portainer utiliza a porta 9000 do host e direciona para porta 9000 do container, então para acesso da ferramenta via web deve-se permitir o tráfego na porta 9000 da instância criando uma regra de firewall no compute engine:





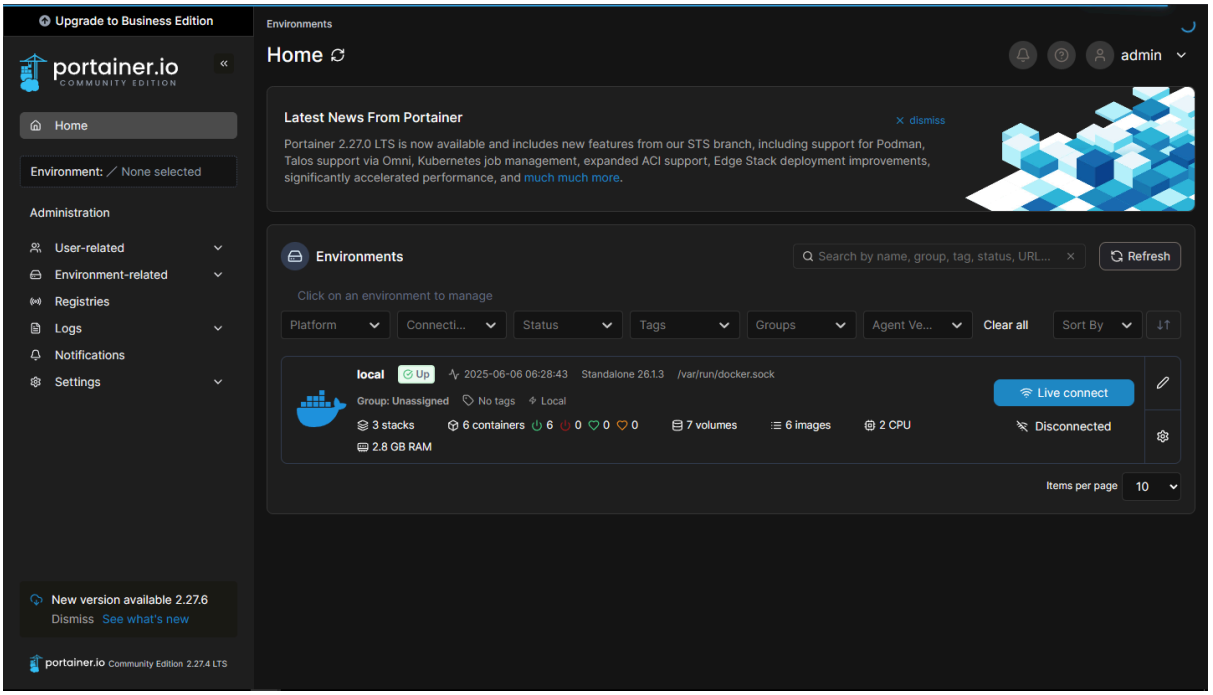
.Acessando o PORTAINER:

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:

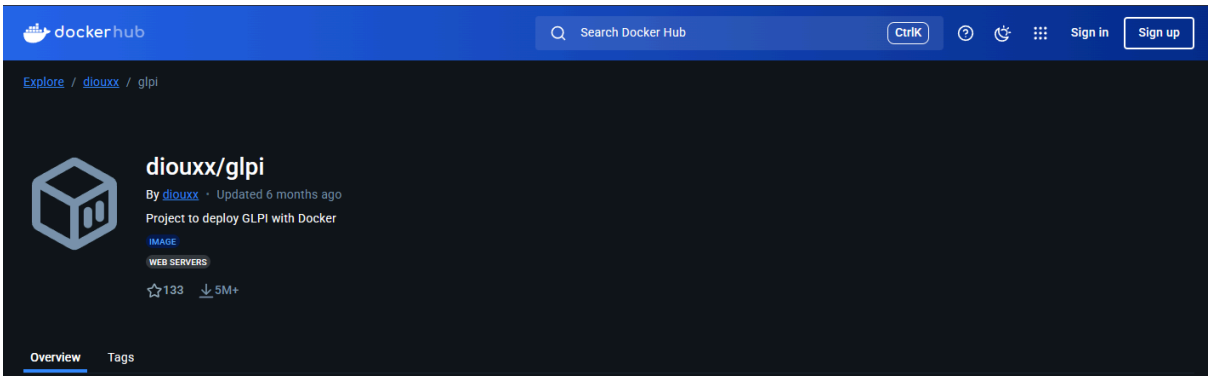
#PAINEL INICIAL



.INSTALAÇÃO DO GLPI

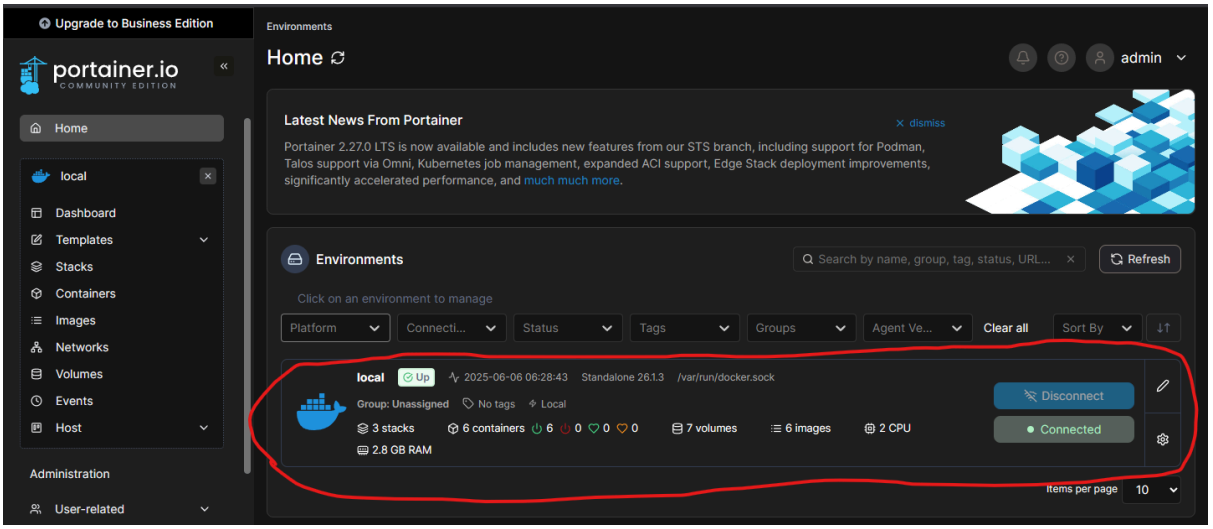
Nessa etapa a aplicação e o banco de dados são instalados via arquivo docker compose fornecido diretamente pela empresa DIOUX.

https://hub.docker.com/r/diouxx/glpi

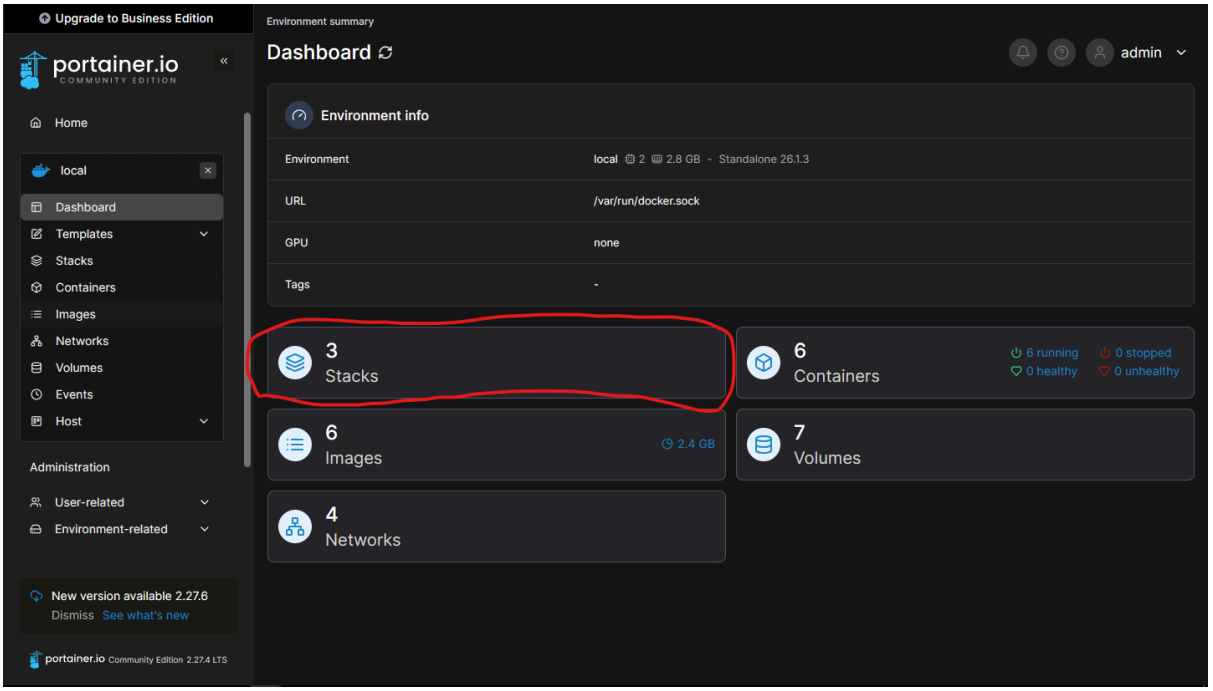


.Criação da pilha(STACK)

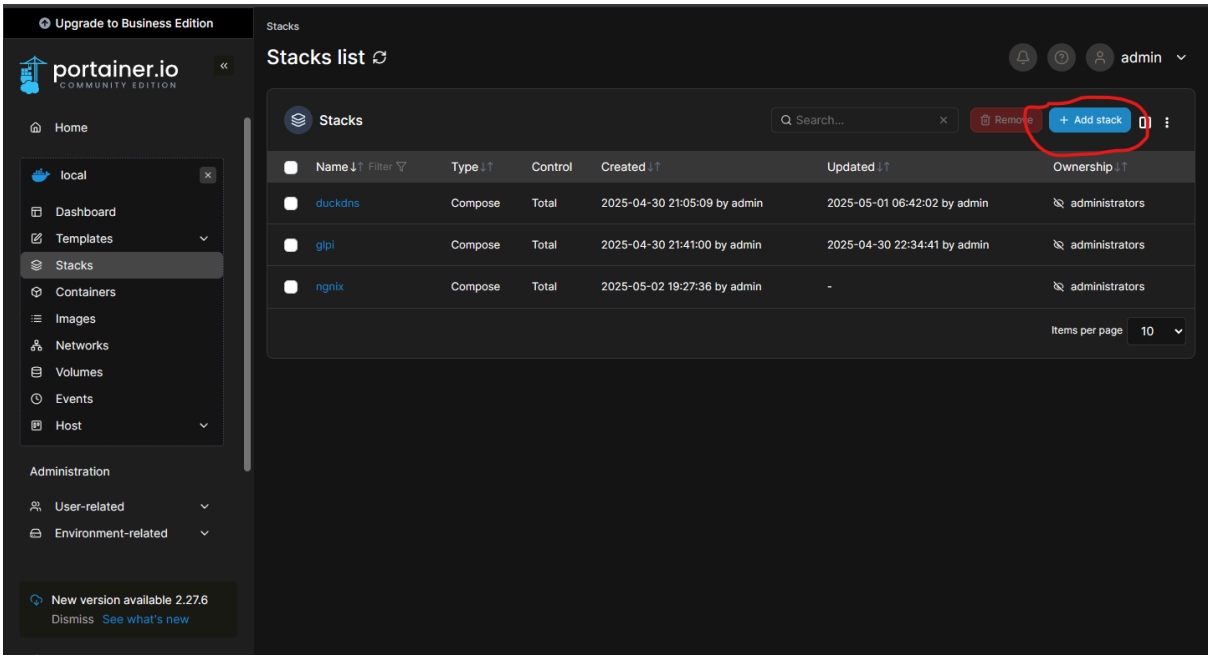
#Selecione o ambiente:



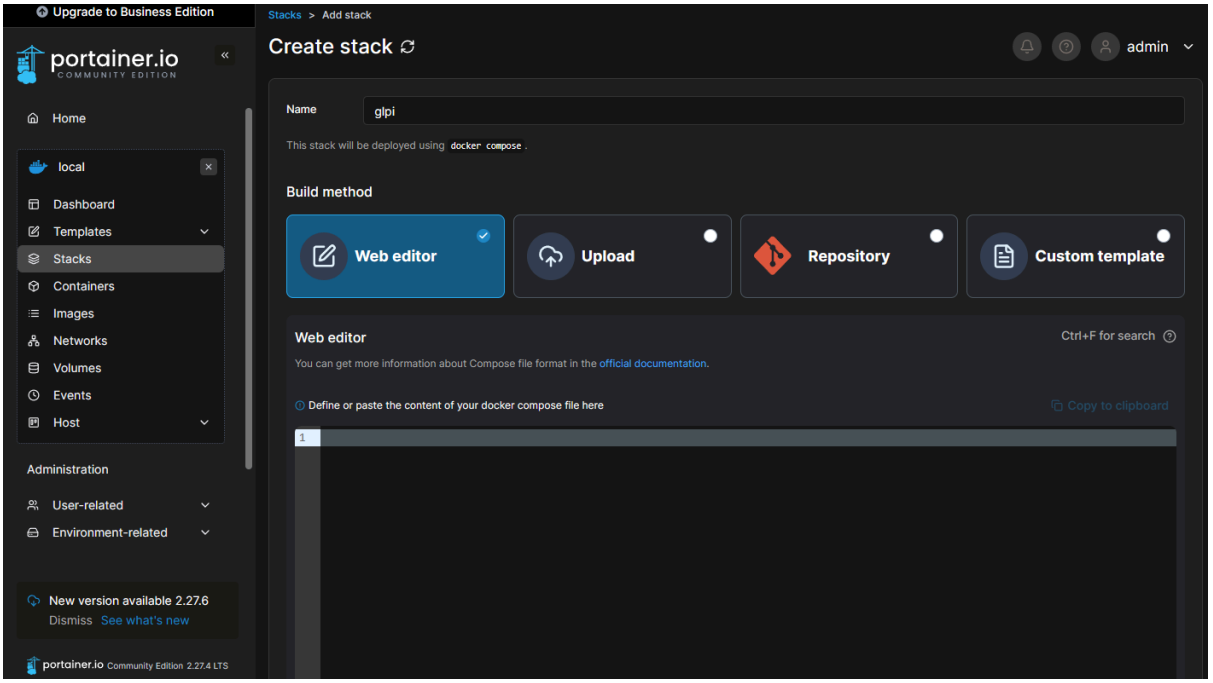
#Clique na opção Stacks no painel direito de menu:



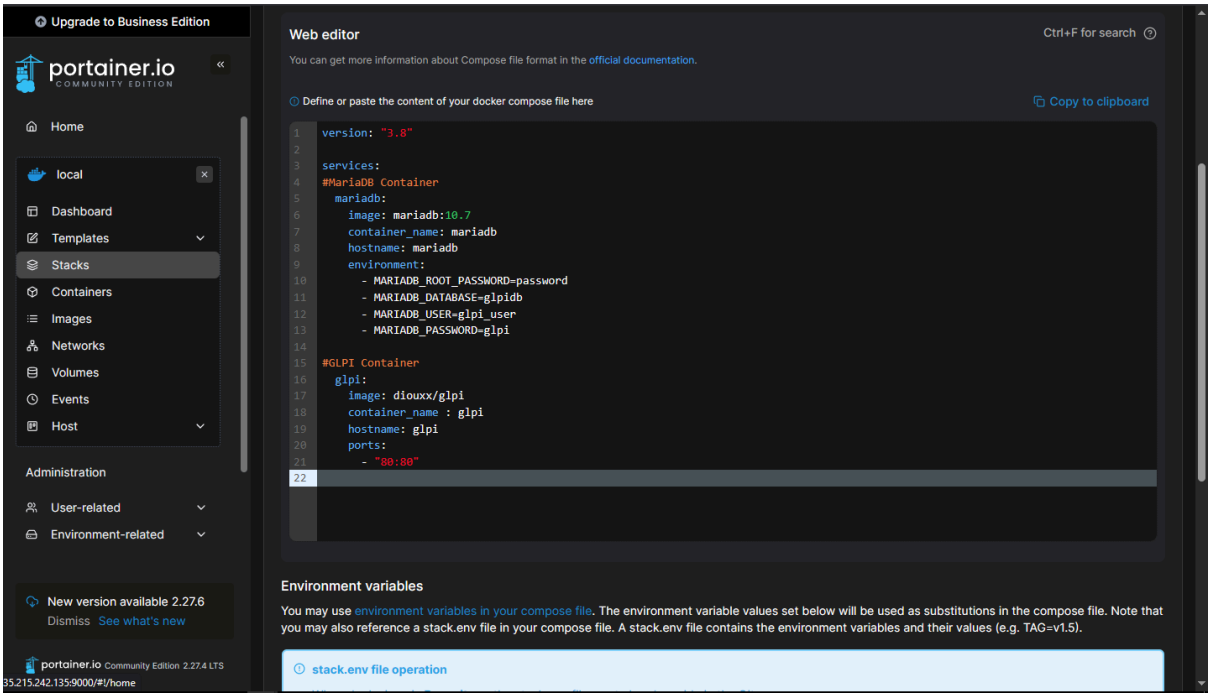
#Clique em adicionar stack



#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 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**

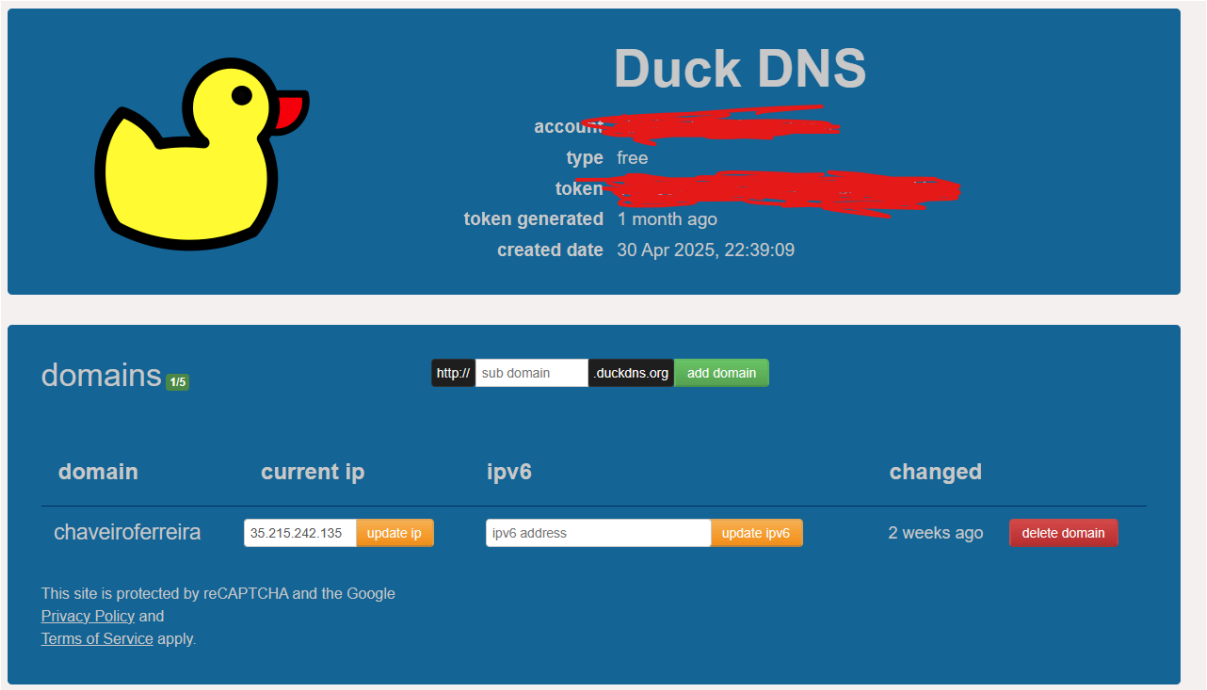
.Instalação do DUCK DNS

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



.Configuração do compose e criação da pilha:

```yaml
services:
  duckdns:
    image: lscr.io/linuxserver/duckdns:latest
    container_name: duckdns
    network_mode: host #optional
    environment:
      - TZ=America/Sao_Paulo
      - SUBDOMAINS=chaveiroferreira.duckdns.org
      - TOKEN=[redacted]
      - UPDATE_IP=ipv4 #optional
      - LOG_FILE=false #optional
    volumes:
      - /path/to/duckdns/config:/config #optional
    restart: unless-stopped
```

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:

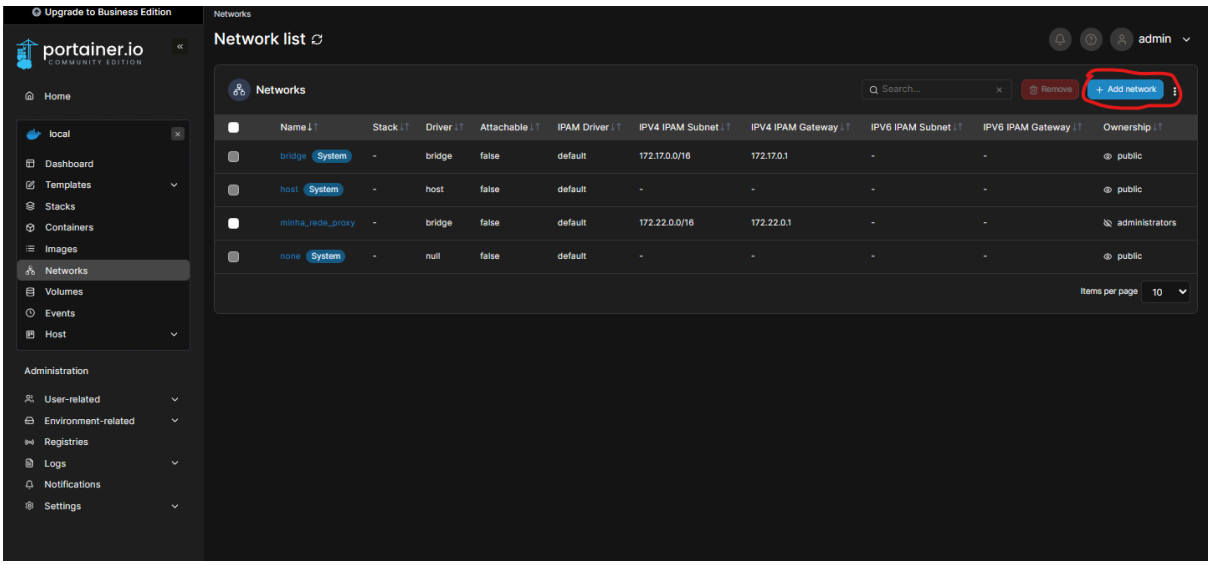
https://nginxproxymanager.com/setup

```yaml
services:
  app:
    image: 'jc21/nginx-proxy-manager:latest'
    restart: unless-stopped
    ports:
      - '80:80'
      - '443:443'
      - '81:81'
    environment:
      # Mysql/Maria connection parameters:
      DB_MYSQL_HOST: "db"
      DB_MYSQL_PORT: 3306
      DB_MYSQL_USER: "npm"
      DB_MYSQL_PASSWORD: "npm"
      DB_MYSQL_NAME: "npm"
      # Uncomment this if IPv6 is not enabled on your host
      # DISABLE_IPV6: 'true'
    volumes:
      - ./data:/data
      - ./letsencrypt:/etc/letsencrypt
    depends_on:
      - db
    networks:
      - minha_rede_proxy
```

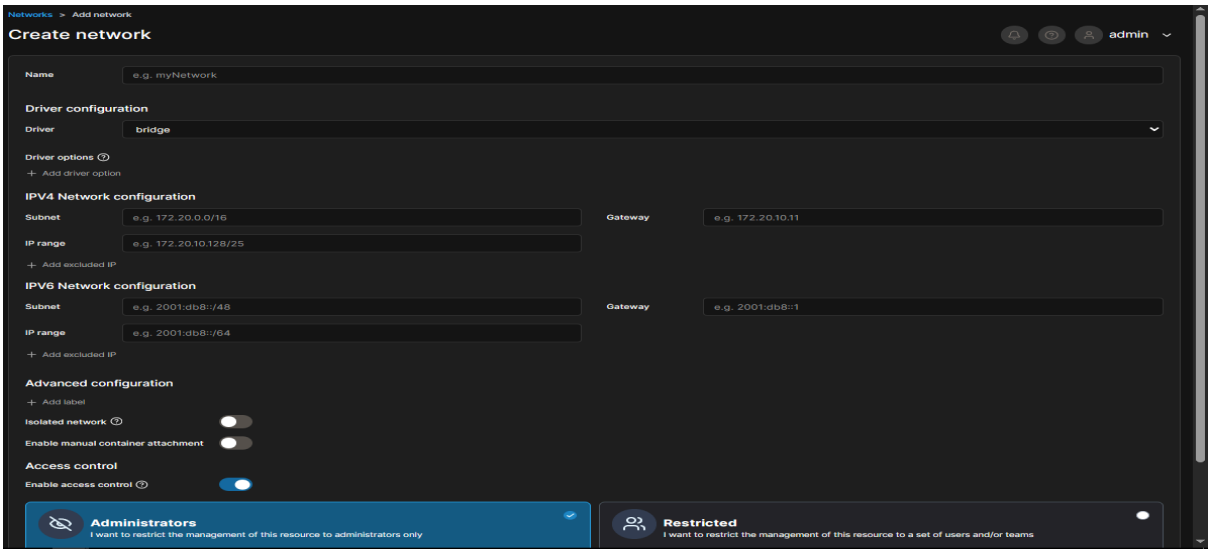
```
networks:
  minha_rede_proxy:
    external: true
    name: minha_rede_proxy
```

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:



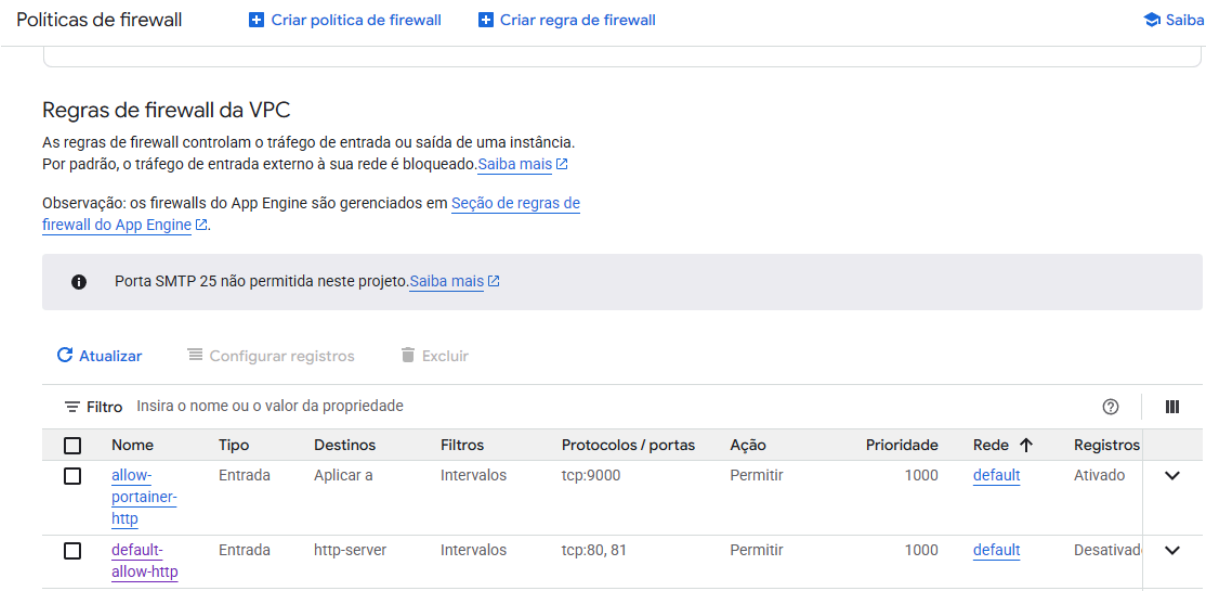
#Inserir o parâmetro de configuração de rede:

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
```

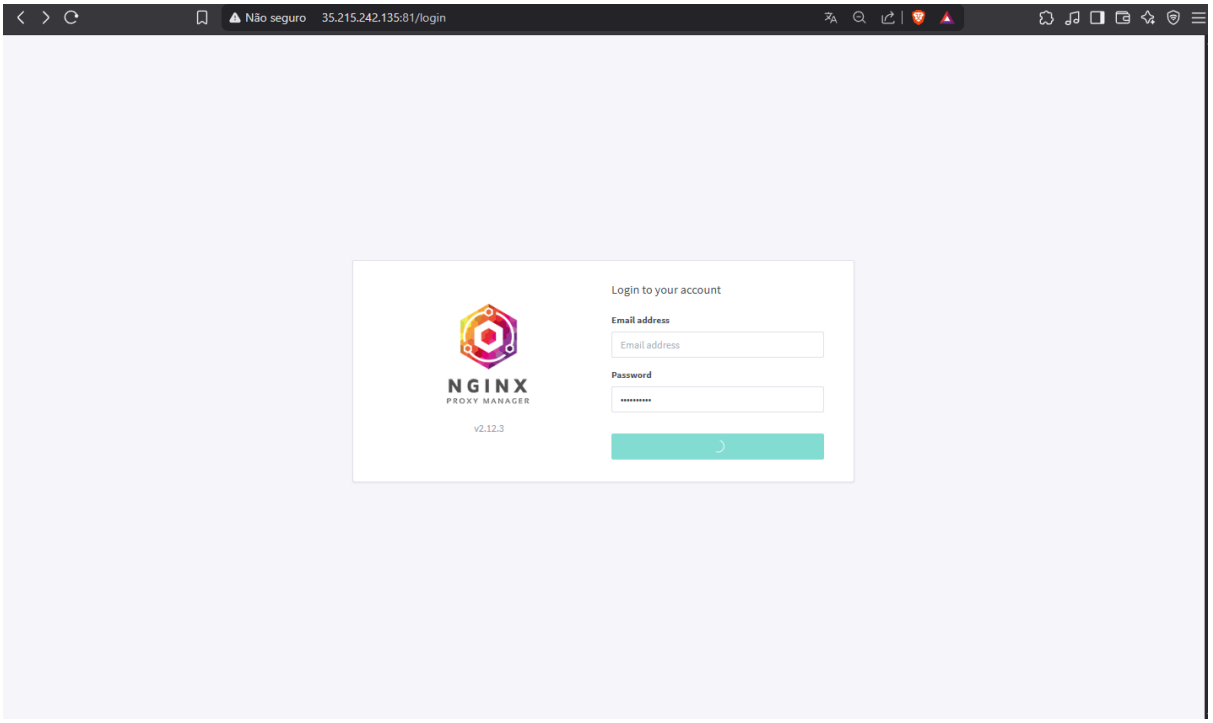
.Configuração do NGINX:

#Primeiro libere a porta 81 no firewall da instância:



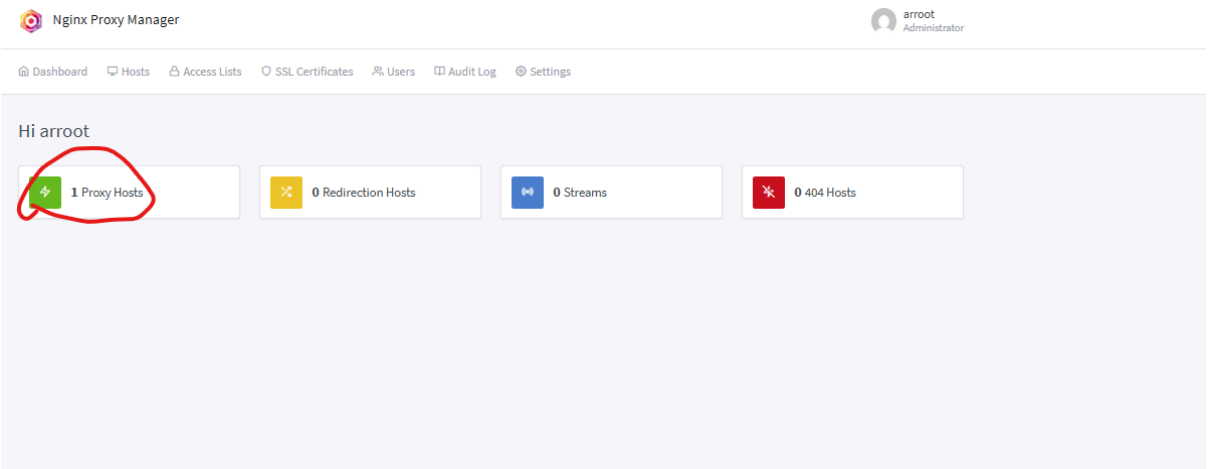
#Acesse o painel web do nginx:

IP EXTERNO:Apontando para porta 81:

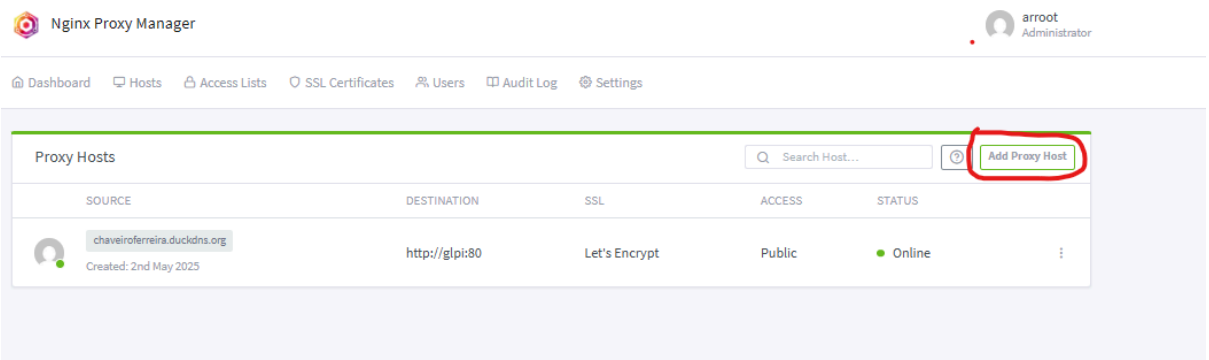


#Após inserir as informações de login(credenciais ficarão resguardadas ao grupo administrador)

#Clique em Proxy Hosts:

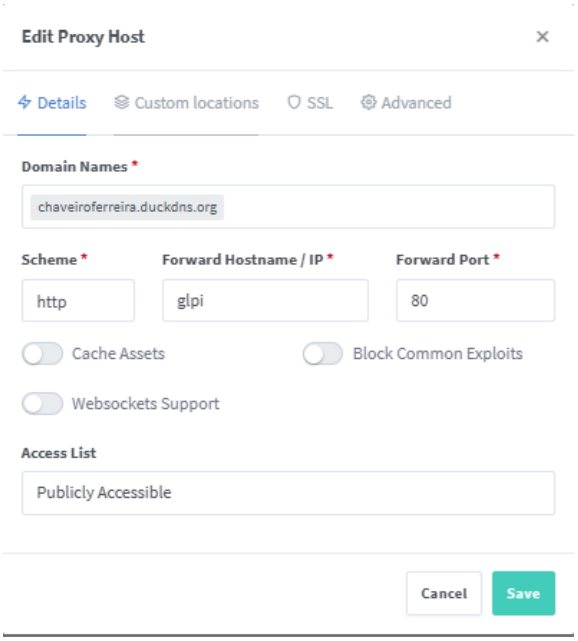


#Add Proxy hosts:

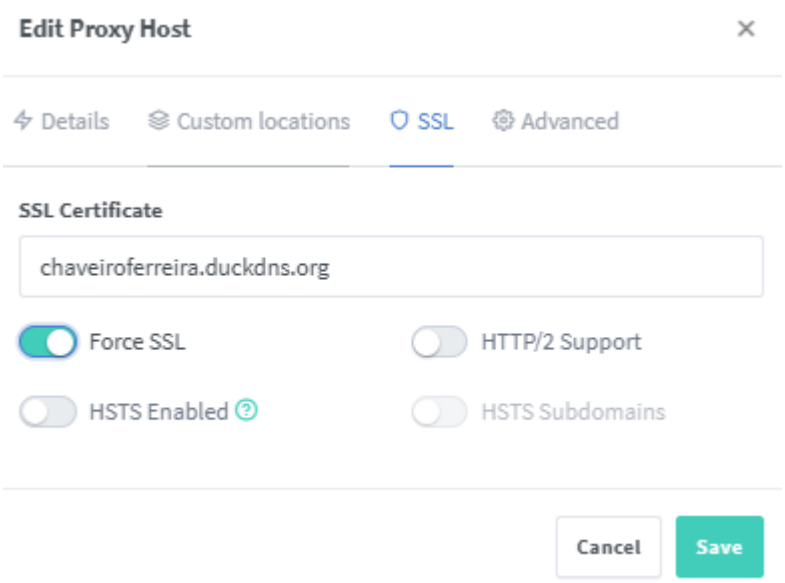


#Insira as seguintes informações:

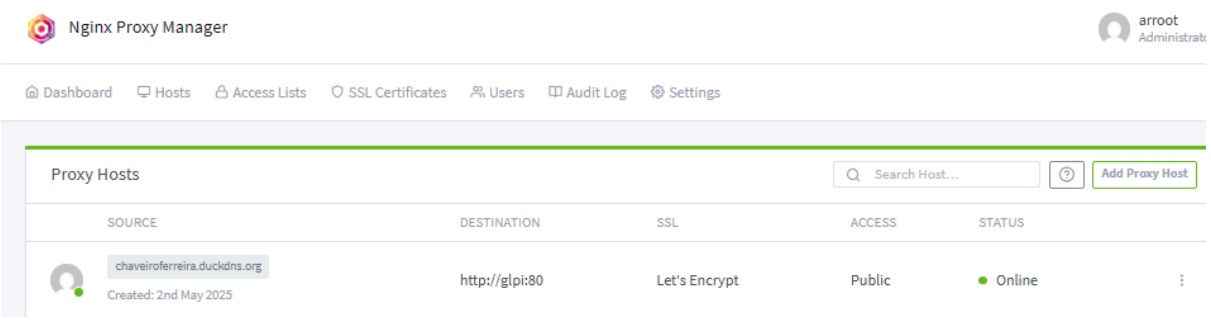
nome de domínio(registrado no DUCK DNS)
hostname do servidor(do container no caso do projeto referido)
Porta Padrão de acesso(GLPI:80)



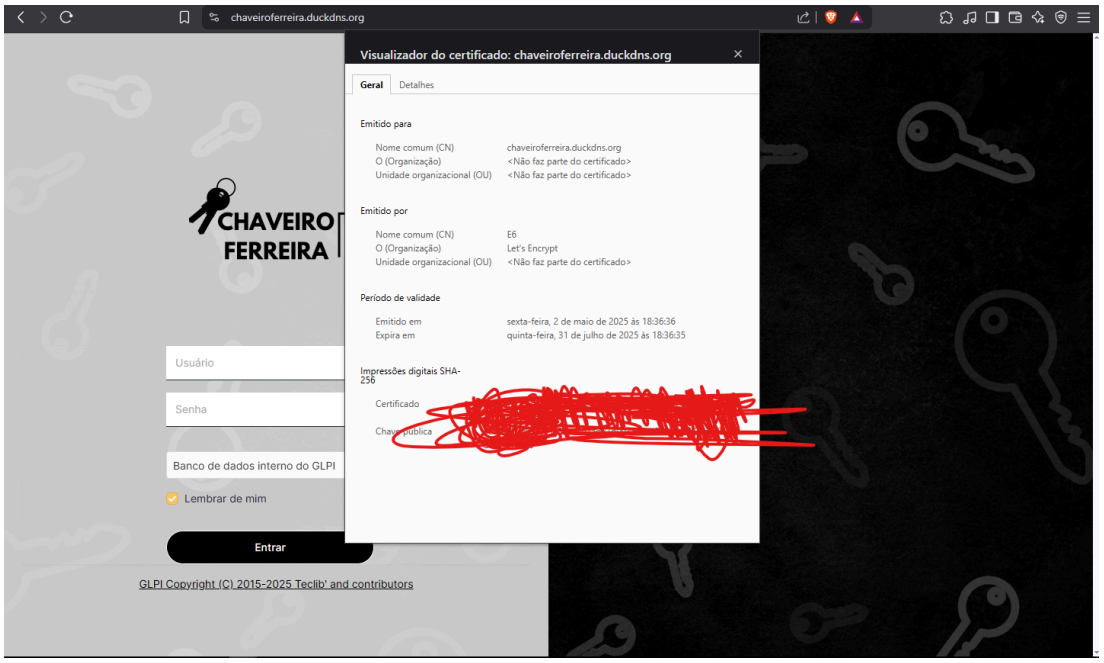
#Solicite um certificado SSL para o domínio desejado:



#Painel pós criação:



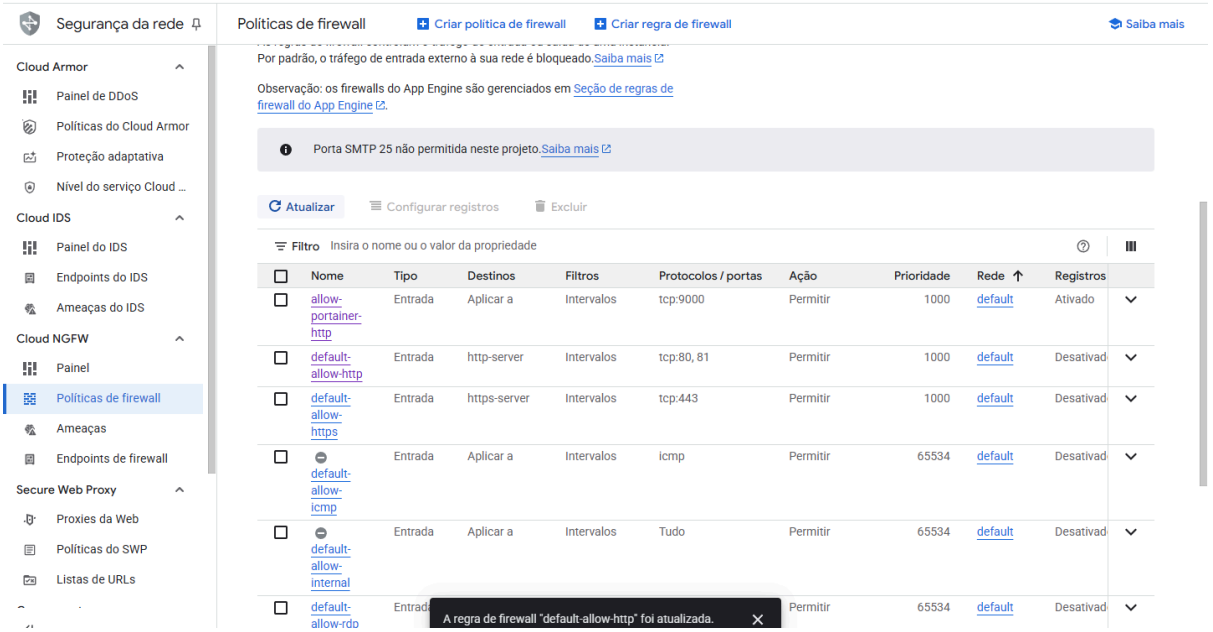
DESCRIÇÃO DA ETAPA: Nessa etapa o funcionamento da URL já é estabelecido e agora a conexão estabelecida por ela está criptografada, o proxy atua como um filtro de segurança que conduz o usuário até a aplicação de maneira criptografada com certificado SSL válido:



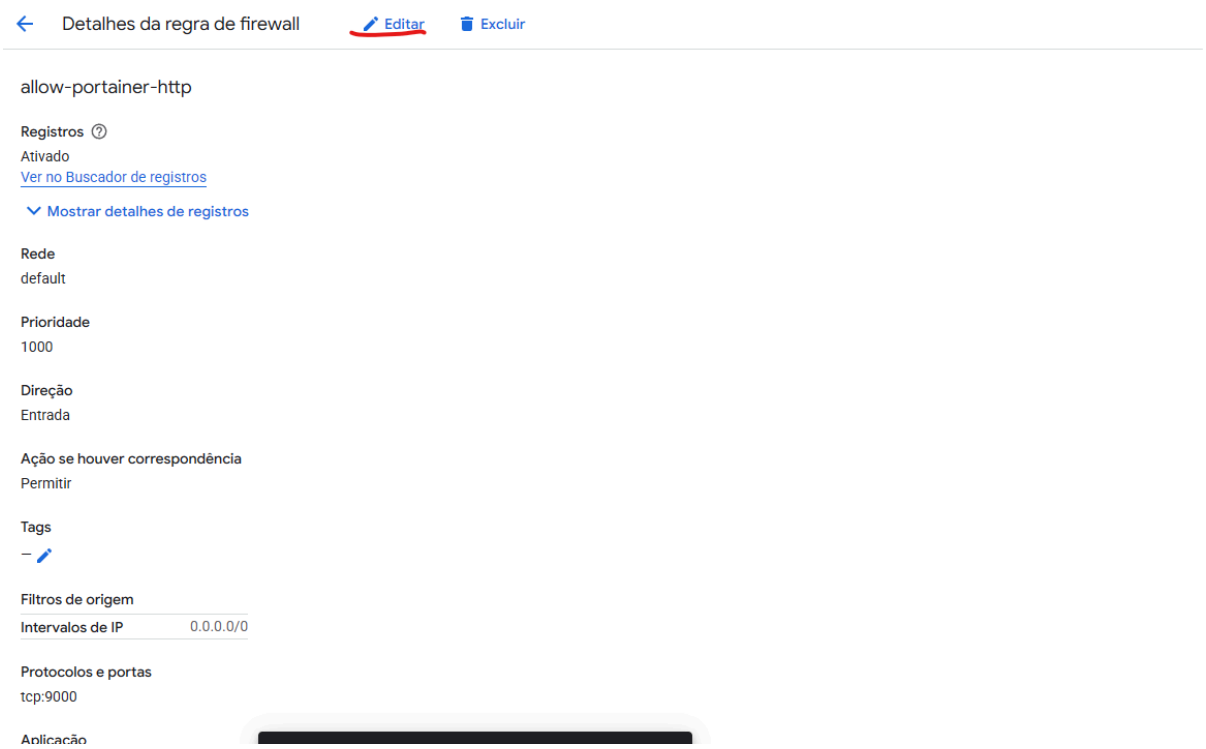
.Fechar as portas do firewall:

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.

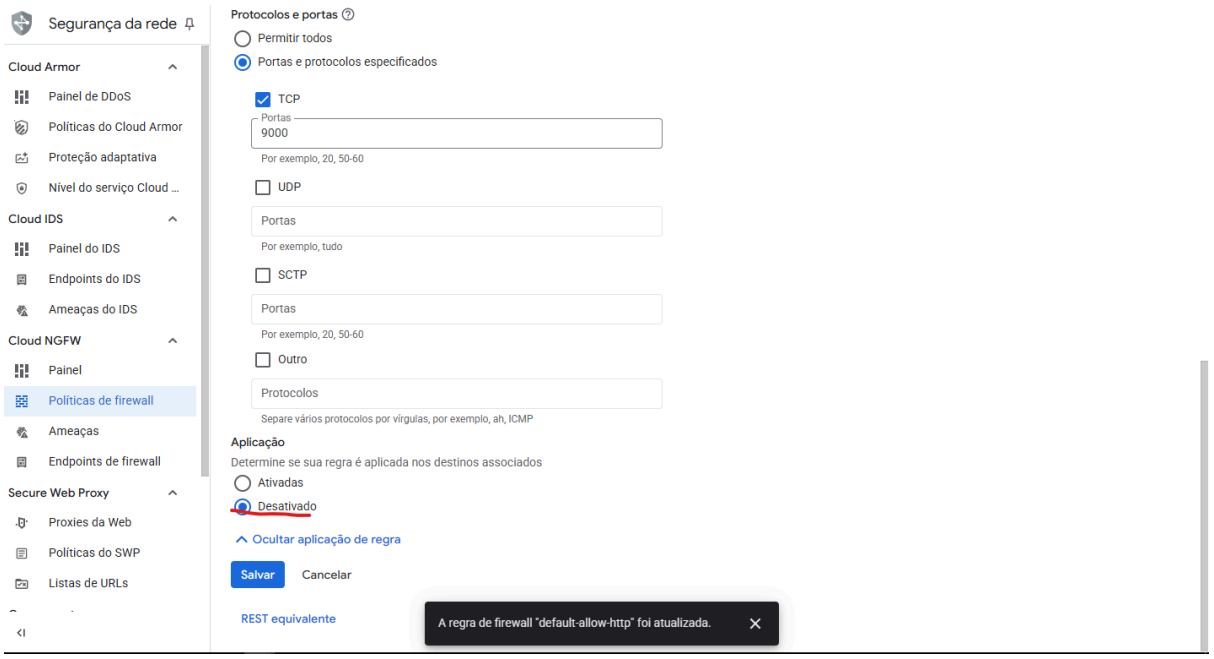
#Clique na regra



#Clique em editar:

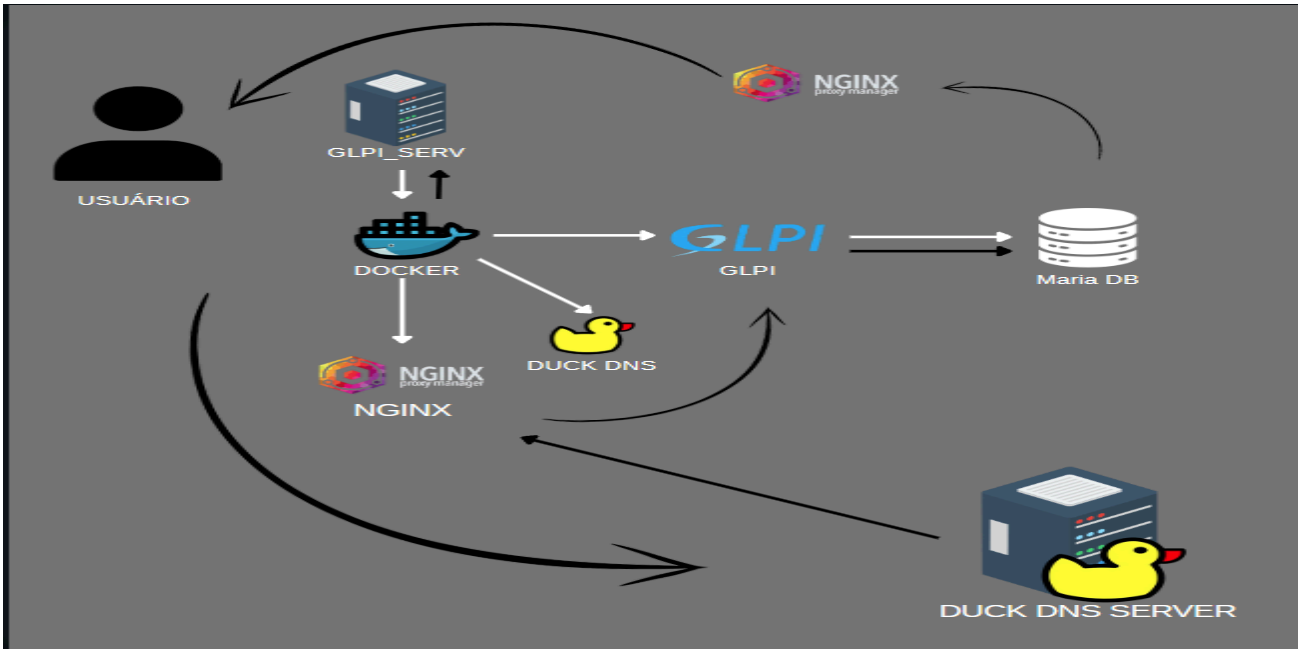


#Desative a regra:



DESCRIÇÃO DA ETAPA: Nessa etapa realizamos a configuração final que disponibilza 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`.

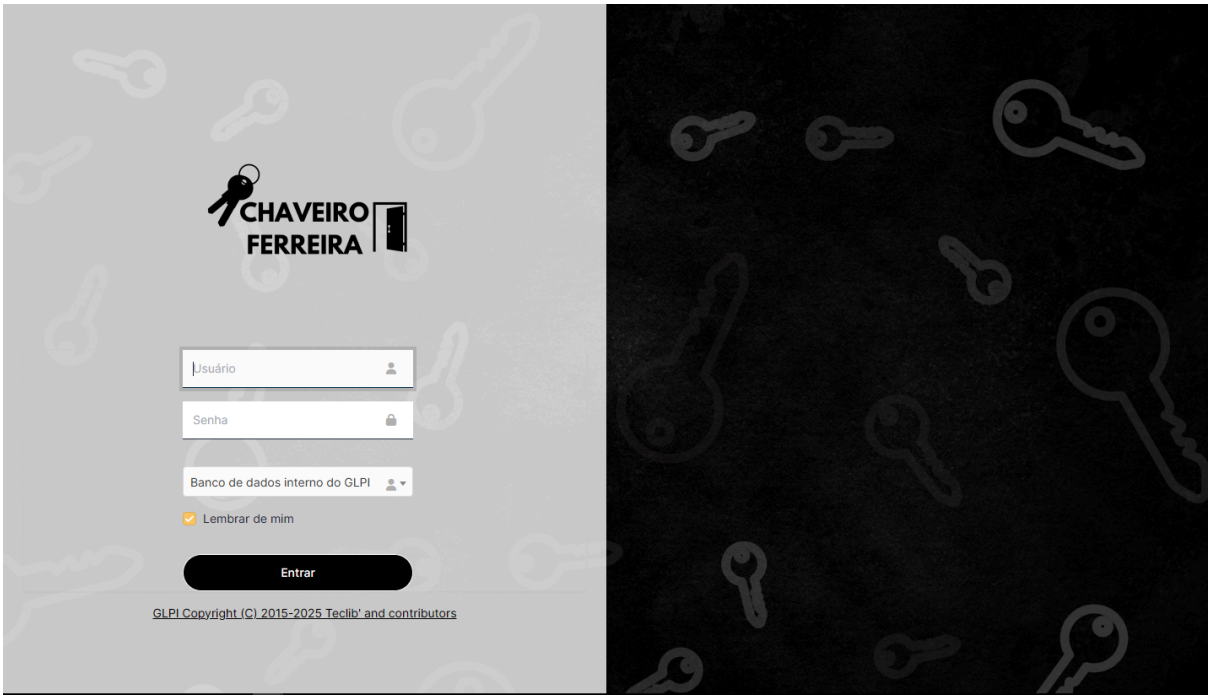
#NGINX Proxy Manager entra em ação

.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.