



INSTITUTO INFNET
GRADUAÇÃO EM REDES DE COMPUTADORES

JEFERSON WILLIAM MAGOLO RODRIGUES

PROJETO DE BLOCO: ARQUITETURA E INFRAESTRUTURA DE APLICAÇÕES

PROF. FABIANO ALVES GISBERT

Projeto de Bloco

Rio de Janeiro
06 de Dezembro 2021

INTRODUÇÃO

Este documento tem como objetivo descrever o processo de implantação de uma aplicação web, utilizando a metodologia DevOps também conhecida como “Integração contínua”, em um ambiente virtualizado visando atender as necessidades da empresa Techinvest tendo nossos serviços como solução.

A Techinvest é uma empresa com mais de 10 anos de experiência no ramo financeiro, com seu escritório localizado na grande São Paulo, atualmente conta com mais de 50 colaboradores em constante desenvolvimento, visando oferecer as melhores oportunidades de investimentos para seus clientes.

Nos últimos anos o brasileiro vem se interessando cada vez mais em cuidar da sua saúde financeira e com esse movimento surgiu o crescimento de empresas concorrentes com menor experiência e excelência que adotam políticas de baixas taxas administrativas para conseguir conquistar novos clientes. Esse tipo de movimento do mercado é uma ameaça para a Techinvest que luta para que seus serviços não se tornem commodities.

Para a solução desse problema, a estratégia encontrada é de se posicionar no mercado como empresa especialista em investimentos em países emergentes, mercado pouco explorado e com grandes possibilidades de multiplicação de patrimônio para seus clientes.

Com a nova estratégia da empresa surge a necessidade da criação de um portal web com objetivo de educar os futuros clientes sobre as oportunidades de crescimento financeiro investindo em países emergentes, com notícias, vídeos e a futura possibilidade de uma área de assinatura para membros pagos.

PROJETO

Para o desenvolvimento do portal foi escolhida a aplicação WordPress, um sistema de gerenciamento de conteúdo (CMS) utilizado em mais de 35% dos sites disponíveis em toda a internet. A ferramenta é OpenSource usa a linguagem PHP como base e seu banco de dados é em MySQL.

O código fonte do portal e suas atualizações serão compartilhados na plataforma GitHub, melhorando o fluxo de trabalho e garantindo o sistema de controle de versão da aplicação. A plataforma de hospedagem de código-fonte permite que interessados cadastrados possam consultar, ajustar e atualizar o código de qualquer lugar pois o conteúdo é armazenado na nuvem.

A infraestrutura para a implantação da aplicação deverá conter um servidor físico com acesso à internet que irá trabalhar de maneira virtualizada com o VMware vSphere, nesse ambiente serão criadas duas máquinas virtuais: a primeira usará o Windows Server 2019 que irá funcionar como servidor de aplicação e a segunda máquina virtual será um Linux Ubuntu contendo MySQL como a função de servidor de dados.

O VMware ESX é um software desenvolvido pela empresa VMware, sua função é fazer o gerenciamento de máquinas virtuais, é executado diretamente no hardware do servidor físico (Bare Metal, tipo 1) o que garante sua alta performance.

Comparado com o VMware ESX outra solução poderia ser o serviço de Nuvem da Amazon AWS rodando uma máquina virtual EC2 com o banco de dados RDS. O serviço é pago mensalmente em dolar sendo assim uma opção arriscada levando em consideração uma possível desvalorização do real.

Para o Armazenamento de dados será necessário um Storage NAS com um processador de 4 núcleos, 16GB de memória ram, um disco de 120GB para o gerenciamento storage, duas interfaces de rede sendo uma disponível para a conexão com o Switch e outra para manutenção e futuras expansões e 10 discos de 5TB totalizando 50TB de armazenamento NAS. criaremos um volume de 50TB formatado em VMFS com dupla redundância (RAID 6), sendo disponível 40TB para armazenamento e 10TB para a redundância.

Máquina virtual de Aplicação especificações:

- 4 cores de processamento.
- 12 GB de memória RAM.
- 1 placa de rede (virtual) para acesso à internet.
- 1 placa de rede (virtual) para acesso ao bando de dados.
- 100 GB de espaço em disco.
- SO: Linux Ubuntu 20.04
- Apps: Apache 2.4, PHP 8.0 e WordPress 5.6

Máquina virtual de Banco de dados especificações:

- 2 cores de processamento.
- 4 GB de memória RAM.
- 1 placa de rede (virtual) para acesso ao servidor de aplicação.
- 100 GB de espaço em disco
- SO: Linux Ubuntu 20.04
- Apps: MySQL 8.0

Especificações Servidor Físico:

- 12 Cores
- 32 GB memória RAM
- 2 placas de rede
- 1 TB de Armazenamento

Storage NAS:

- 4 Cores
- 8 GB de memória RAM
- 120 GB de espaço em disco
- 2 Interfaces de rede
- 10 discos de 5TB = 50TB (40TB de Armazenamento e 10TB de redundância)

Diagrama Físico

Servidor

Cpu de 12 cores
32GB RAM
1TB de Disco
1 Ethernet: Para conexão com o Switch
1 Ethernet: Para futuras expansões e manutenção do servidor

Storage

Cpu 4 Nucleos
8GB RAM
120 GB Disco
1 Ethernet: Para conexão com o Switch
1 Ethernet: Para futuras expansões e manutenção
10 Discos de 5TB

Switch

1 Gbps
8 Portas

Roteador

1 Gbps
4 portas

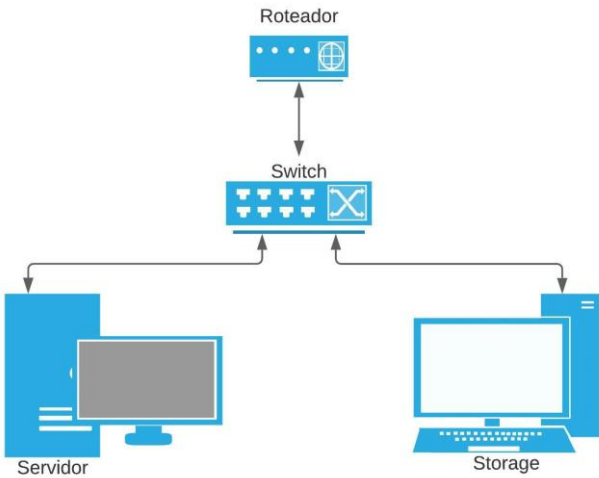
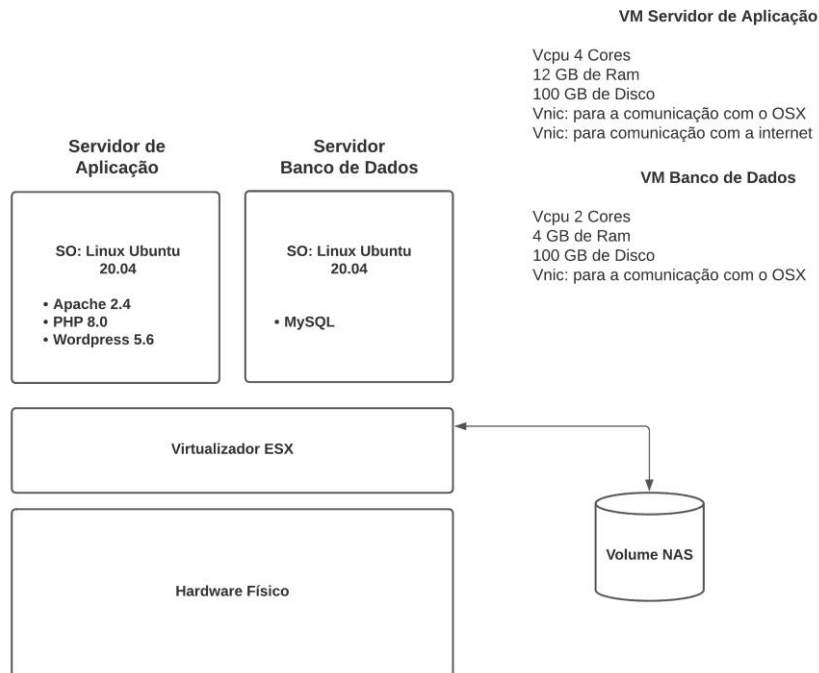


Diagrama das Conexões Virtuais



AUTOMAÇÃO

Com o objetivo de manter a agilidade e dinamismo nas atualizações do conteúdo do portal, utilizaremos o DevOps, conjunto de práticas entre as equipes de desenvolvimento de Software e a equipe de Infraestrutura com o objetivo de produzir resultados mais rápidos e melhores aumentando a confiança da aplicação desejada pela empresa.

Continuous Delivery (Entrega Contínua em português) é uma prática de desenvolvimento de software que utiliza a automação para agilizar o lançamento de um novo código. Estabelece processos no qual os desenvolvedores compartilham os códigos em nosso repositório na GitHub mantendo sempre o mais novo versionamento do código.

Para as boas práticas de DevOps utilizaremos a ferramenta Ansible, software de código aberto com a função de configurar e automatizar inúmeras tarefas de infraestrutura e desenvolvimento em uma central única localizada. Devido a sua grande capacidade de automação o Ansible diminui a demanda de mão de obra reduzindo assim a probabilidade de erro humano.

IMPLANTAÇÃO

Para a demonstração da implantação e automação do nosso projeto iremos utilizar os recursos atualmente disponíveis sendo ele um único computador pessoal de 4 núcleos, 16GB de memória RAM e 500GB de armazenamento SSD.

Toda a nossa estrutura será simulada utilizando o Software de Virtualização VMware ESX em uma rede virtualizada, não sendo esse o ambiente ideal para a implantação real do projeto.

Instalação do VMware ESX no servidor físico

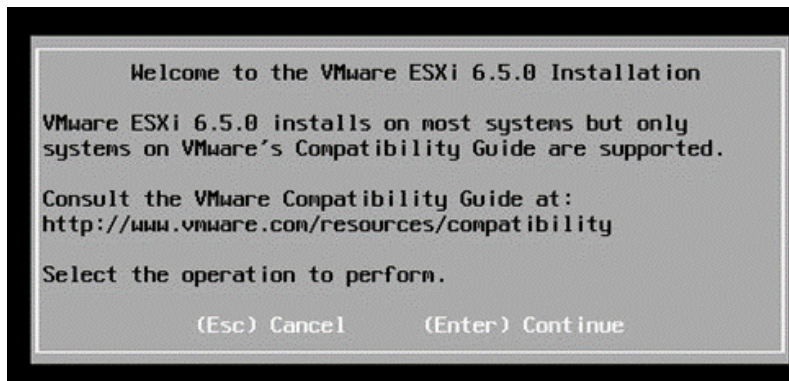
O virtualizador ESX deve ser instalado diretamente no servidor físico.

A Instalação pode ser feita através do CD/DVD de instalação ou baixar o arquivo de instalação direto do site da VMware, neste documento iremos demonstrar a instalação via CD/DVD.

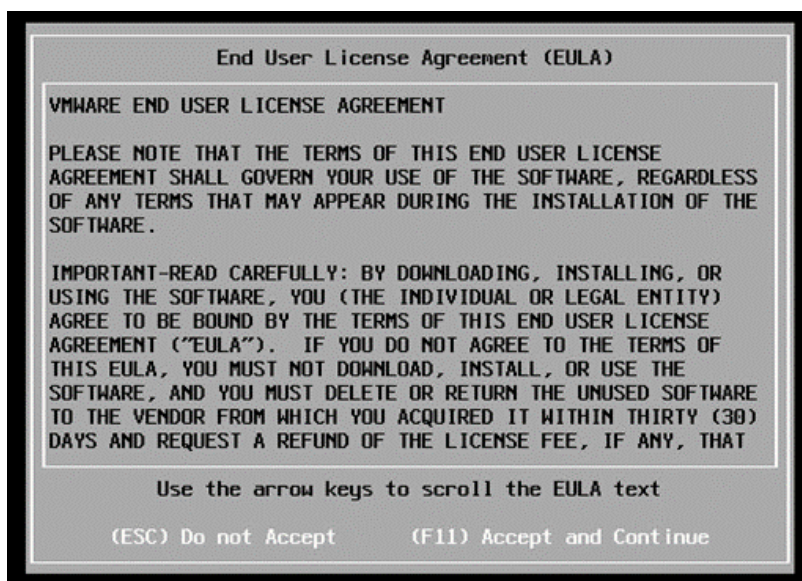
1 – Inicialize o computador através do boot do driver de CD/DVD e espere os arquivos de instalação serem carregados.



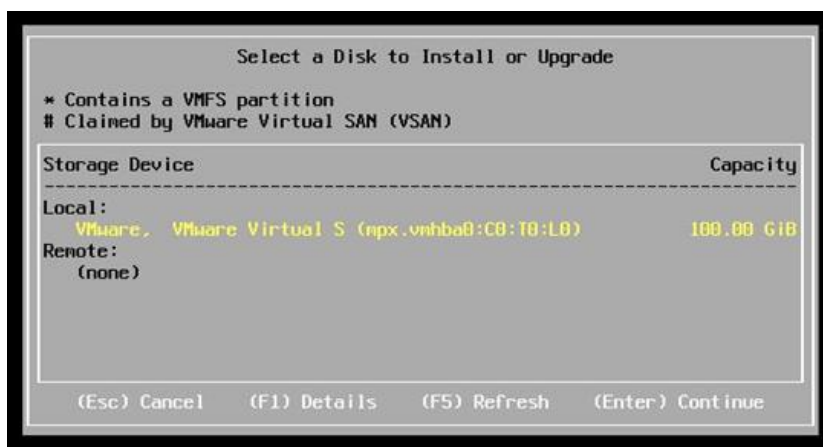
2 – Assim que os arquivos de instalação forem carregados pressione Enter.



3 – Pressione F11 para aceitar o acordo de licença.



4 – Selecione o disco desejado para a Instalação do ESX e pressione Enter.
Atenção: A instalação do ESX irá apagar todos os dados do disco escolhido.



5 – Escolha o tipo de teclado e depois insira a sua senha.

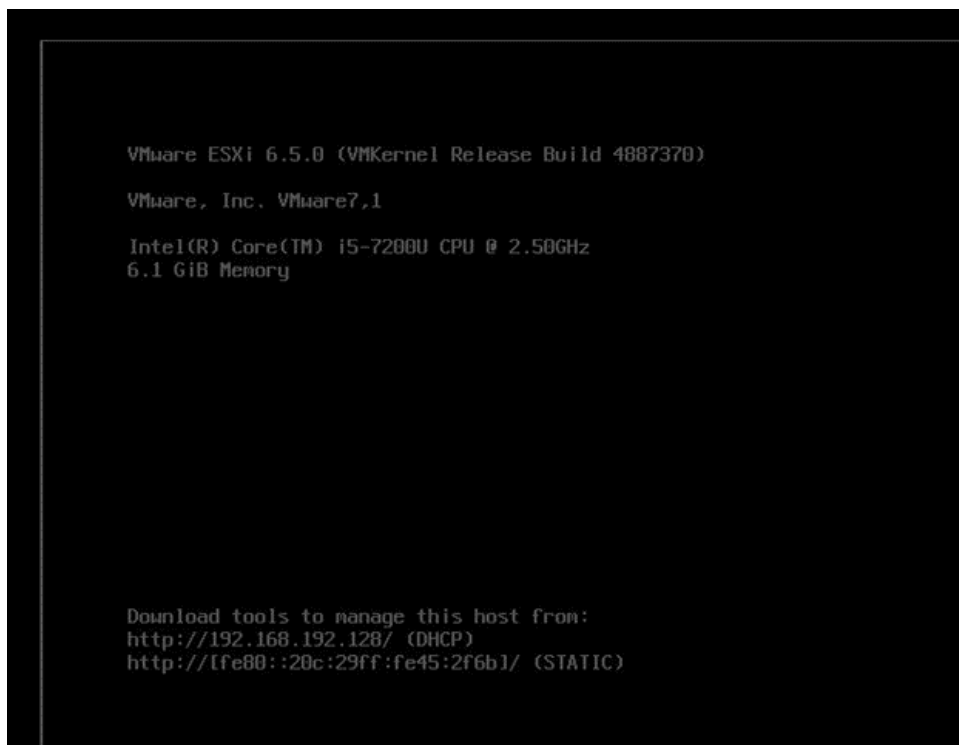
6 – Pressione F11 para iniciar a instalação.



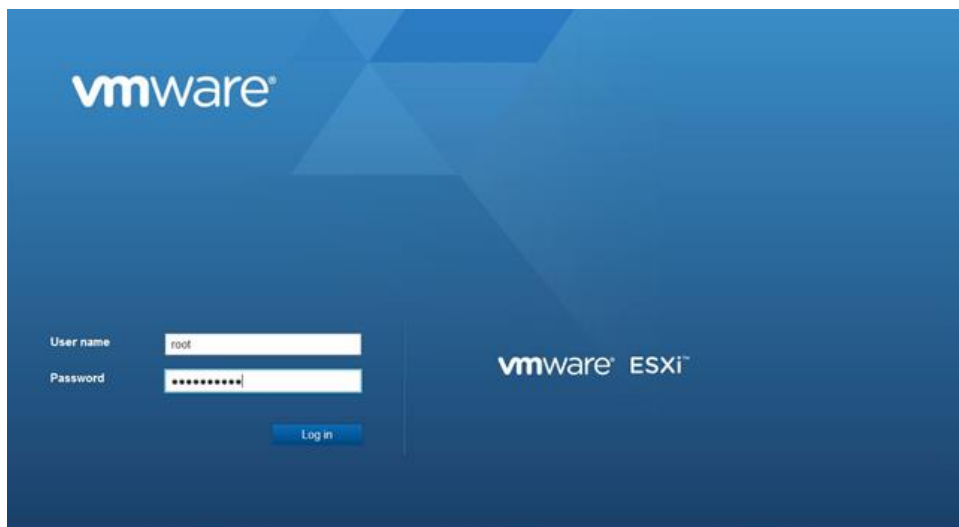
7 – Quando a Instalação estiver completa, remova o CD/DVD e pressione Enter para a reinicialização do sistema.



8 – Após a reinicialização do sistema o ESX irá exibir o IP para a conexão.
Atenção: para o funcionamento correto da aplicação o endereço de IP deve ser configurado como estático evitando assim futuros problemas de indisponibilidade de IP dentro da rede.



09 – Utilize o IP fornecido anteriormente para acessar o ESX pelo seu navegador.

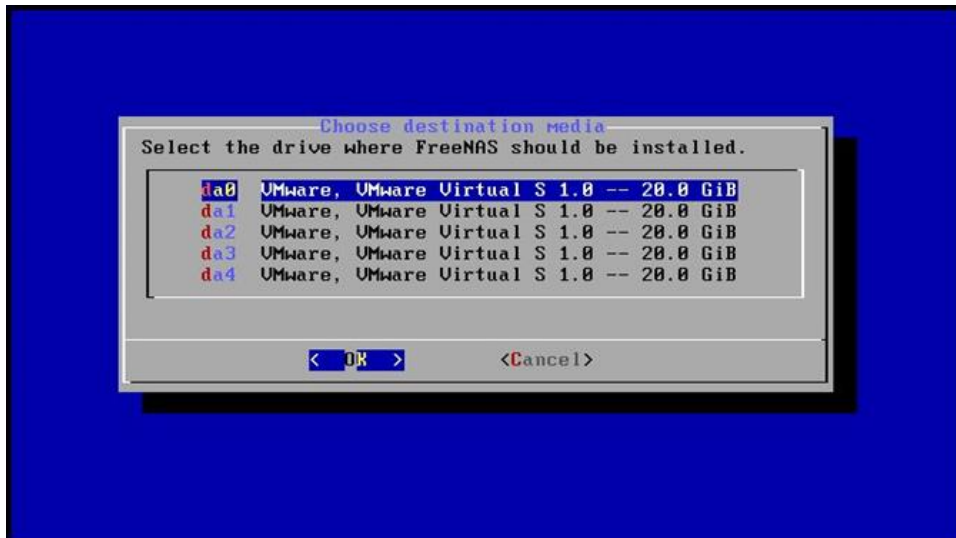


Criação do Volume FreeNas

Requisitos mínimos servidor FreeNas: 1 processador dual core, 8GB de memória RAM, 1 placa de rede física, 1 Disco de 20GB para a instalação do FreeNas. Nosso Servidor Storage irá trabalhar com folga de hardware para possíveis futuras atualizações do sistema.

1 – Selecionar a Opção Install/Upgrade.

2 – Verificar os discos encontrados pelo FreeNas e selecionar OK



3 – O FreeNas exibirá uma mensagem de Atenção dizendo que os dados dos discos serão apagados, selecione OK.



4 – Caso o FreeNas retorne ao início da instação, reinicie a Máquina Virtual.

5 – O FreeNas irá gerar um IP, acesse o IP via o navegador de sua preferência do seu computador.

```
Sun Dec  5 00:42:03 PST 2021

FreeBSD/amd64 (freenas.local) (ttyv0)

Console setup
-----

1) Configure Network Interfaces
2) Configure Link Aggregation
3) Configure VLAN Interface
4) Configure Default Route
5) Configure Static Routes
6) Configure DNS
7) Reset WebGUI login credentials
8) Reset to factory defaults
9) Shell
10) Reboot
11) Shutdown

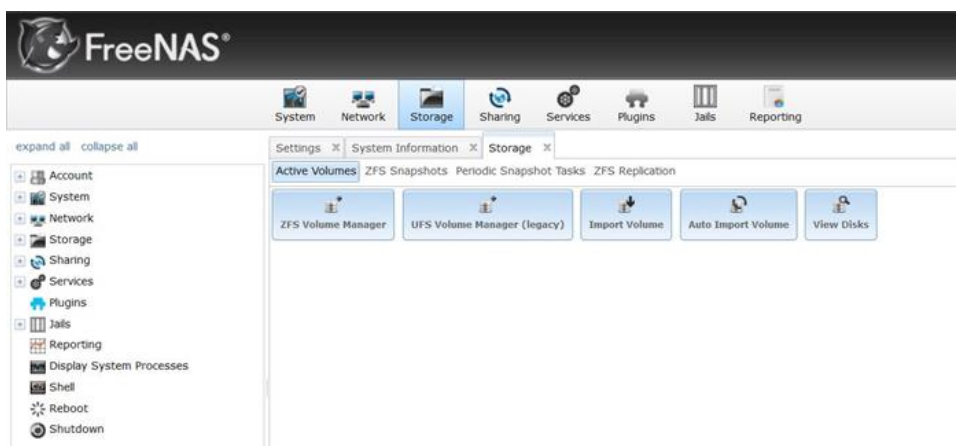
You may try the following URLs to access the web user interface:

http://192.168.192.129

Enter an option from 1-11: █
```

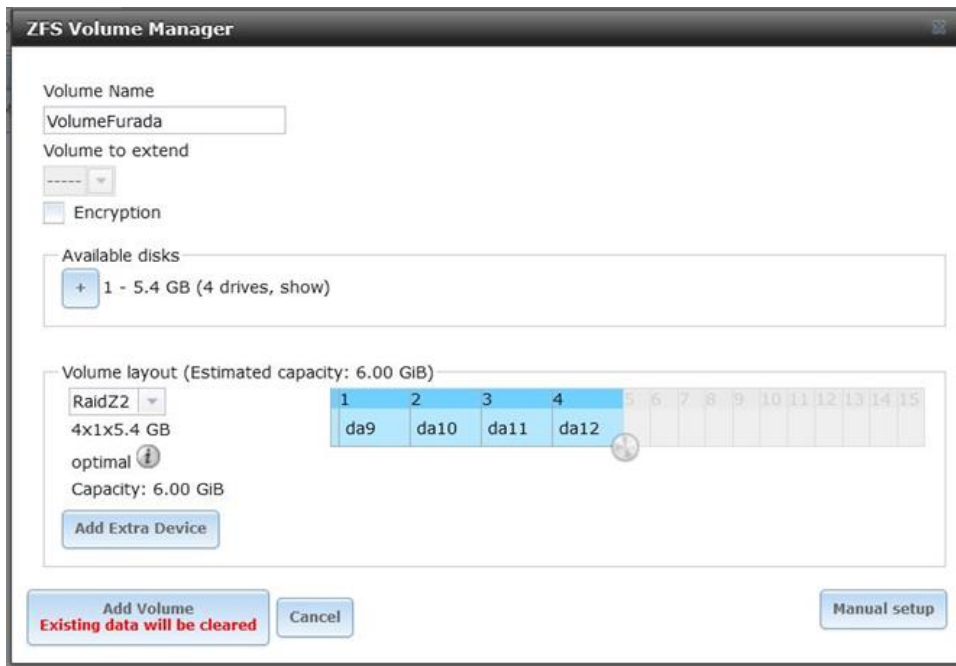
6 – O FreeNas pedirá para criar uma senha, digite a sua senha e clique em Log In.

7 – A aplicação exibirá o console de configuração, clique em Storage e em ZFS Volume Manager para criar o novo volume único de discos rígidos com redundância.

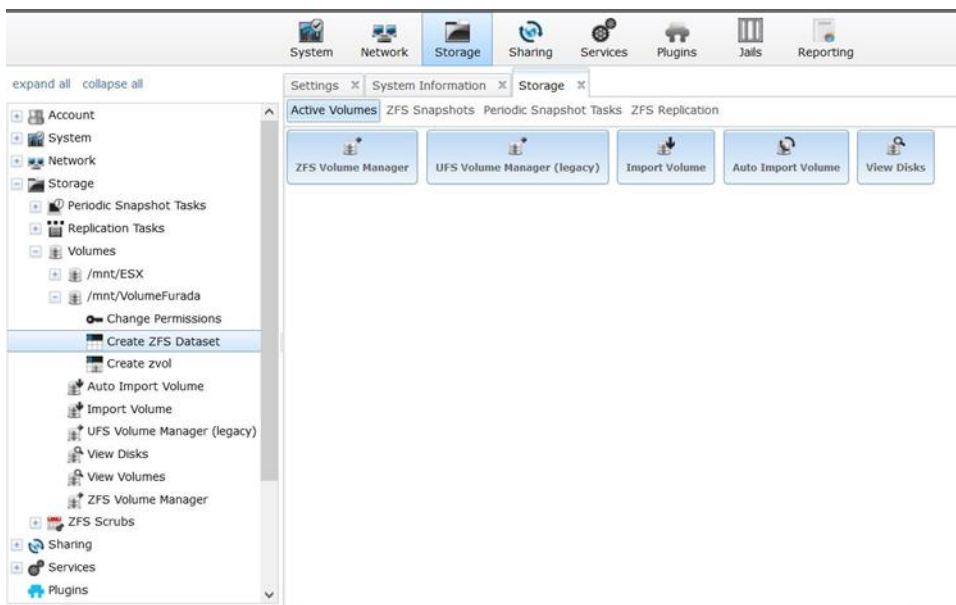


8 – Nomeie o volume e selecione 10 Discos de 5TB. Selecione o RaidZ2 (Raid 6) como o layout do volume. Essa configuração vai trabalhar com a disponibilidade de 40TB (8 discos de 5TB) e garantirá a dupla redundância de 10TB (2 discos de 5TB) como segurança.

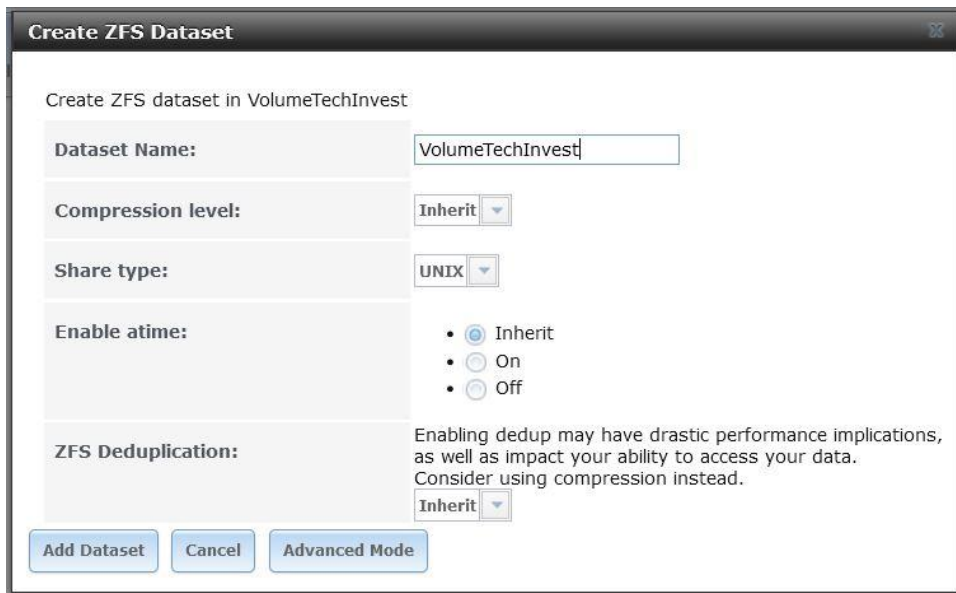
Clique em Add Volume para criar o volume.



9 – Agora devemos criar uma estrutura visível para os Sistemas Operacionais (Dataset). Na aba de navegação da esquerda encontre o volume que acabou de ser criado e clique em **Create ZFS Dataset**.



10 – Crie um nome para o volume, mantenha as configurações pré-selecionadas e clique em **Add Dataset**. Após esse passo o sistema operacional será capaz de enxergar o RAID de quatro discos como um único volume.



Create ZFS dataset in VolumeTechInvest

Dataset Name:

Compression level:

Share type:

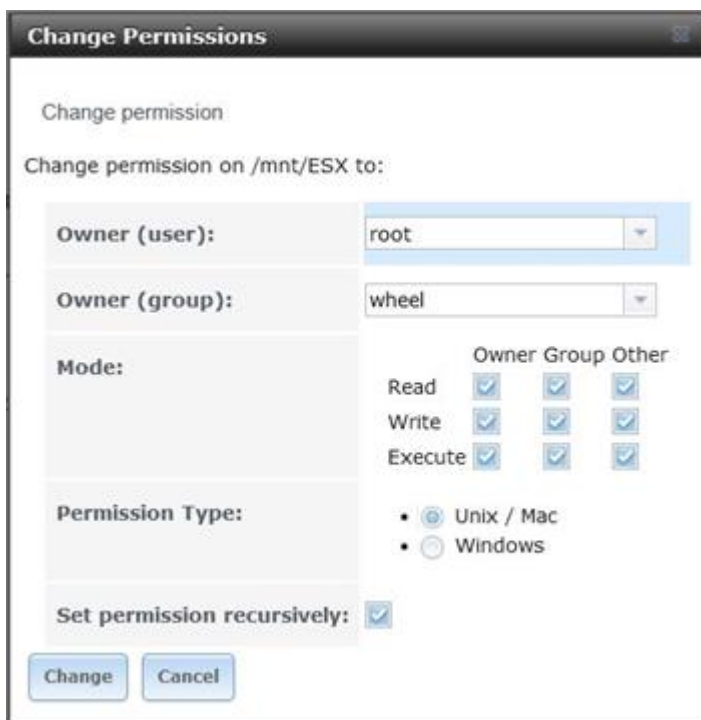
Enable atime:

- ☒ Inherit
- ☐ On
- ☐ Off

ZFS Deduplication:
Enabling dedup may have drastic performance implications, as well as impact your ability to access your data. Consider using compression instead.

11 – Vá até a aba de navegação a esquerda e clique em Change Permissions. Agora devemos habilitar o ESX para fazer gravação no disco, na linha write marque todas as opções.

ATENÇÃO: Você deve habilitar a opção Set Permission Recursively, depois clique em Change.



Change permission

Change permission on /mnt/ESX to:

Owner (user):

Owner (group):

Mode:

	Owner	Group	Other
Read	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Write	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Execute	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

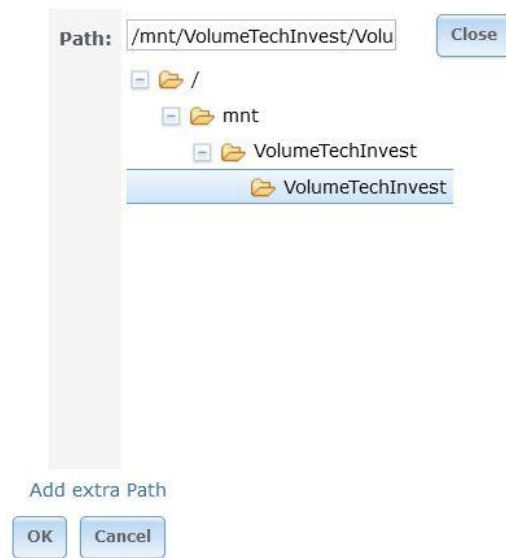
Permission Type:

- ☒ Unix / Mac
- ☐ Windows

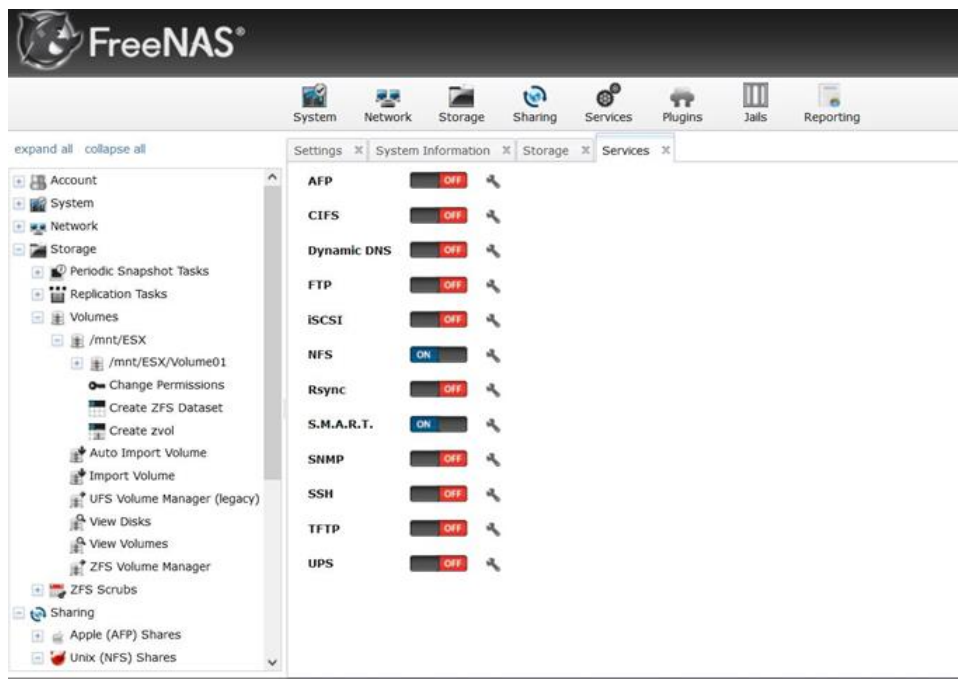
Set permission recursively: ☒

12 – Dentro do FreeNas na parte superior, vá em Sharing > UNIX (NFS) > Add unix (NFS Share) 13 – Aponte o Volume a ser compartilhado como na imagem abaixo e

clique em OK. Logo em seguida será exibido na tela uma caixa perguntando se gostaríamos de compartilhar o serviço, clique em OK.



14 – Em seguida será exibido as opções habilitadas no FreeNas



Criação de um Datastore no Vmware ESXi usando o volume criado no FreeNas

01 – Abra o ESXi, na aba esquerda de navegação vá em Storage e clique na Opção New Datastore.

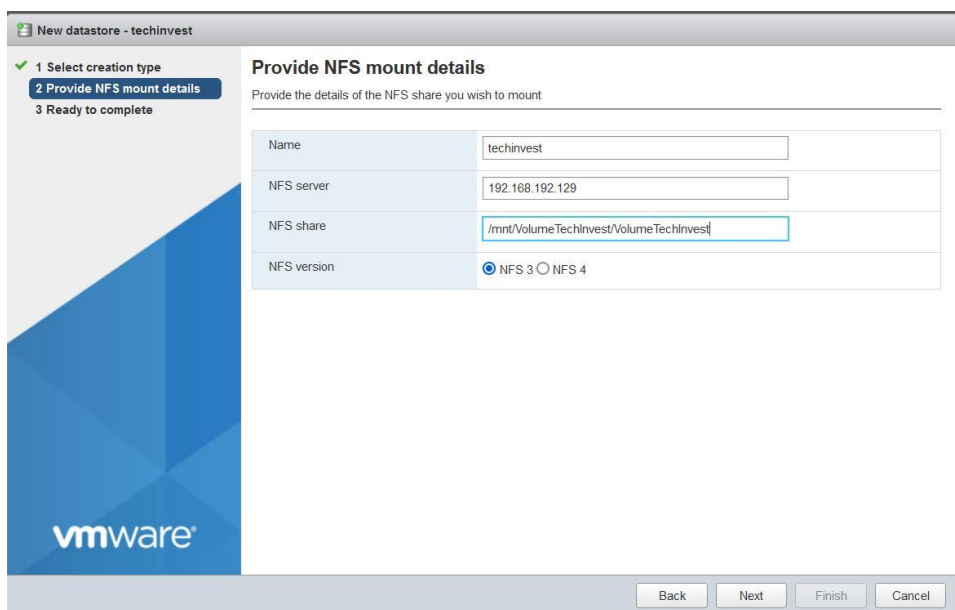


02 – Selecione a Opção Mount NFS Datastore e clique em Next.

03 – Crie um nome para o Storage, no servidor NFS você deve colocar o IP fornecido pelo FreeNas.

04 - Em NFS Share é preciso descrever o caminho do volume do FreeNas.

Atenção! É preciso diferenciar as letras minúsculas e maiúsculas para evitar erros. Depois habilite a versão NFS 3 e clique em Next.



05 – Se os passos anteriores foram executados corretamente o Volume deverá ser exibido como na figura abaixo.

New datastore		Increase capacity	Register a VM	Datastore browser	Refresh	Actions	Search	
Name	Drive Type	Capacity	Provisio...	Free	Type	Thin pro...	Access	
datastore1	SSD	72.5 GB	972 MB	71.55 GB	VMFS5	Supported	Single	
techinvest	Unknown	34.09 GB	208 KB	34.09 GB	NFS	Supported	Single	
								2 items

Criação e Configuração do Servidor Virtual de Aplicação

01 – Na tela inicial do ESX clique no menu Virtual Machines e clique em Create / Register VM, dê um nome para a VM e selecione o sistema operacional Ubuntu.

New virtual machine - Servidor de Aplicação (ESXi 6.5 virtual machine)

1 Select creation type
2 Select a name and guest OS
 3 Select storage
 4 Customize settings
 5 Ready to complete

Select a name and guest OS

Specify a unique name and OS

Name
 Servidor de Aplicação

Virtual machine names can contain up to 80 characters and they must be unique within each ESXi instance.

Identifying the guest operating system here allows the wizard to provide the appropriate defaults for the operating system installation.

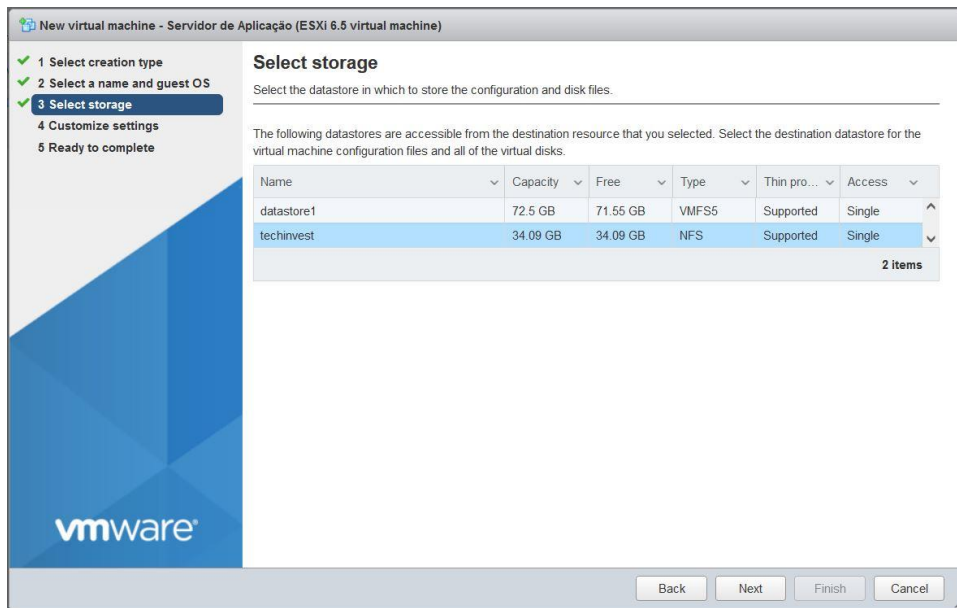
Compatibility
 ESXi 6.5 virtual machine

Guest OS family
 Linux

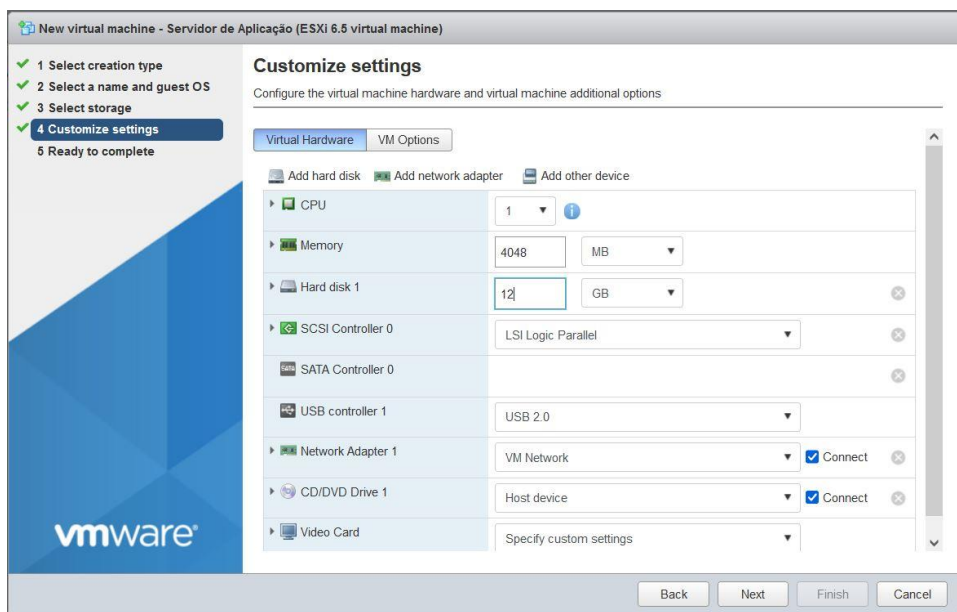
Guest OS version
 Ubuntu Linux (64-bit)

Back Next Finish Cancel

02 – Selecione o Datastore TechInvest Criado anteriormente para o armazenamento da nossa VM.



03 – Selecione as configurações indicadas e selecione a imagem do Sistema Operacional para a Instalação e clique em Next.



04 – A Criação da VM deve aparecer como na figura abaixo.

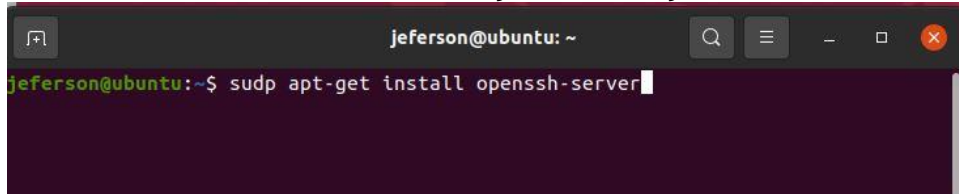
Create / Register VM Console Power on Power off Suspend Refresh Actions							
	Virtual machine	Status	Used space	Guest OS	Host name	Host CPU	Host memory
<input type="checkbox"/>	PB_Ansible_test	Normal	1.15 KB	CentOS 4/5 or later (64-bit)	Unknown	0 MHz	0 MB
<input type="checkbox"/>	Servidor de Aplicação	Normal	1.5 KB	Ubuntu Linux (64-bit)	Unknown	0 MHz	0 MB

2 items

Instalação do Ansible

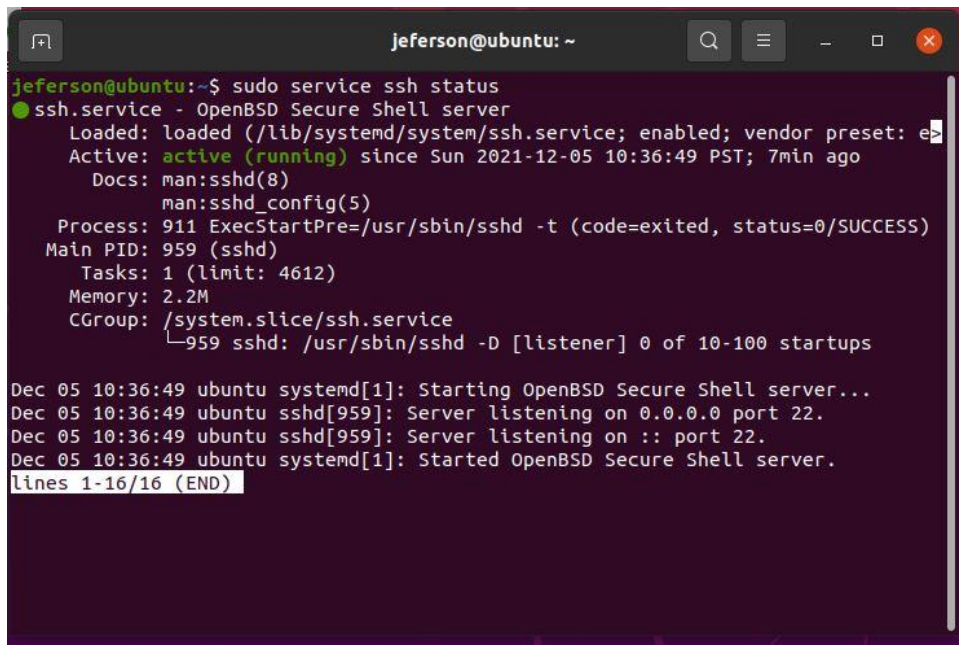
Instalação do Serviço SSH

1 – Primeiro vamos fazer a instalação do serviço SSH com o comando abaixo



```
jefferson@ubuntu: ~  
jefferson@ubuntu:~$ sudo apt-get install openssh-server
```

2 – O segundo passo é verificar se o serviço foi corretamente instalado.



```
jefferson@ubuntu: ~  
jefferson@ubuntu:~$ sudo service ssh status  
● ssh.service - OpenBSD Secure Shell server  
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: e  
   Active: active (running) since Sun 2021-12-05 10:36:49 PST; 7min ago  
     Docs: man:sshd(8)  
           man:sshd_config(5)  
   Process: 911 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)  
    Main PID: 959 (sshd)  
       Tasks: 1 (limit: 4612)  
      Memory: 2.2M  
     CGroup: /system.slice/ssh.service  
             └─959 sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups  
  
Dec 05 10:36:49 ubuntu systemd[1]: Starting OpenBSD Secure Shell server...  
Dec 05 10:36:49 ubuntu sshd[959]: Server listening on 0.0.0.0 port 22.  
Dec 05 10:36:49 ubuntu sshd[959]: Server listening on :: port 22.  
Dec 05 10:36:49 ubuntu systemd[1]: Started OpenBSD Secure Shell server.  
lines 1-16/16 (END)
```

3 – Agora devemos criar uma chave ssh utilizando o comando “ssh-keygen” pressione enter nas opções que aparecerem pois assim a nossa chave será configurada sem a necessidade de utilizar senha.

4 – Com o comando “cp -p ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys” armazenamos a nossa chave em um diretório padrão para o serviço SSH.

5 – Agora podemos testar o acesso ao nosso servidor sem senha usando o comando “ssh localhost”.

```
jeferson@ubuntu: ~  
jeferson@ubuntu:~$ ssh localhost  
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-38-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
183 updates can be applied immediately.  
103 of these updates are standard security updates.  
To see these additional updates run: apt list --upgradable  
  
Your Hardware Enablement Stack (HWE) is supported until April 2025.  
Last login: Wed Oct 20 09:57:49 2021 from 127.0.0.1  
jeferson@ubuntu:~$
```

Criando um playbook no Ansible para a instalação do Wordpress

Para a criação do nosso Playbook é importante certificar-se que o Ansible foi instalado e configurado corretamente digitando o comando “ping localhost” no terminal do Ubuntu, a resposta deve aparecer como na figura abaixo:

```
jeferson@ubuntu: ~/Desktop  
usr/share/ansible/plugins/modules']  
ansible python module location = /usr/lib/python3/dist-packages/ansible  
executable location = /usr/bin/ansible  
python version = 3.8.10 (default, Jun 2 2021, 10:49:15) [GCC 9.4.0]  
jeferson@ubuntu:~/Desktop$ ping localhost  
PING localhost (127.0.0.1) 56(84) bytes of data.  
64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.082 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.112 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=3 ttl=64 time=0.102 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=4 ttl=64 time=0.064 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=5 ttl=64 time=0.110 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=6 ttl=64 time=0.104 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=7 ttl=64 time=0.115 ms  
64 bytes from localhost (127.0.0.1): icmp_seq=8 ttl=64 time=0.072 ms  
^C  
--- localhost ping statistics ---  
8 packets transmitted, 8 received, 0% packet loss, time 7148ms  
rtt min/avg/max/mdev = 0.064/0.095/0.115/0.018 ms  
jeferson@ubuntu:~/Desktop$ ansible localhost -m ping -u jeferson  
localhost | SUCCESS => {  
  "changed": false,  
  "ping": "pong"  
}  
jeferson@ubuntu:~/Desktop$
```

1 – Para organizar melhor o nosso projeto vamos criar um diretório chamado “wordpress-ansible”. Dentro desse diretório vamos criar um arquivo para o nosso playbook e o outro arquivo para o armazenamento do host Wordpress.

```
jeferson@ubuntu: ~/wordpress-ansible
jeferson@ubuntu:~/Desktop$ cd ..
jeferson@ubuntu:~$ mkdir wordpress-ansible
jeferson@ubuntu:~$ cd wordpress-ansible
jeferson@ubuntu:~/wordpress-ansible$ touch playbook.yml
jeferson@ubuntu:~/wordpress-ansible$ touch hosts
jeferson@ubuntu:~/wordpress-ansible$
```

2 – Para o uso das melhores práticas iremos dividir o nosso playbook em quatro módulos chamados “roles”, as roles serão:

I – Server: Armazena as informações de configuração do nosso servidor.

II – PHP: Armazena as configurações PHP usadas pelo WordPress

III – Mysql: Relacionado as configurações do Banco de Dados.

IV – Wordpress: Finaliza a configuração detalhando o funcionamento da aplicação.

3 – Dentro da pasta “wordpress-ansible” vamos criar um diretório para as nossas roles, na pasta roles usaremos comandos Ansible para criar a estrutura básica da nossa aplicação

```
jeferson@ubuntu: ~/wordpress-ansible/roles
jeferson@ubuntu:~/Desktop$ cd ..
jeferson@ubuntu:~$ mkdir wordpress-ansible
jeferson@ubuntu:~$ cd wordpress-ansible
jeferson@ubuntu:~/wordpress-ansible$ touch playbook.yml
jeferson@ubuntu:~/wordpress-ansible$ touch hosts
jeferson@ubuntu:~/wordpress-ansible$ mkdir roles
jeferson@ubuntu:~/wordpress-ansible$ cd roles
jeferson@ubuntu:~/wordpress-ansible/roles$ ansible-galaxy unit server
usage: ansible-galaxy role [-h] ROLE ACTION ...
ansible-galaxy role: error: argument ROLE_ACTION: invalid choice: 'unit' (choose from 'init', 'remove', 'delete', 'list', 'search', 'import', 'setup', 'login', 'info', 'install')
jeferson@ubuntu:~/wordpress-ansible/roles$ ansible-galaxy init server
- Role server was created successfully
jeferson@ubuntu:~/wordpress-ansible/roles$ ansible-galaxy init php
- Role php was created successfully
jeferson@ubuntu:~/wordpress-ansible/roles$ ansible-galaxy init mysql
- Role mysql was created successfully
jeferson@ubuntu:~/wordpress-ansible/roles$ ansible-galaxy init wordpress
- Role wordpress was created successfully
jeferson@ubuntu:~/wordpress-ansible/roles$
```

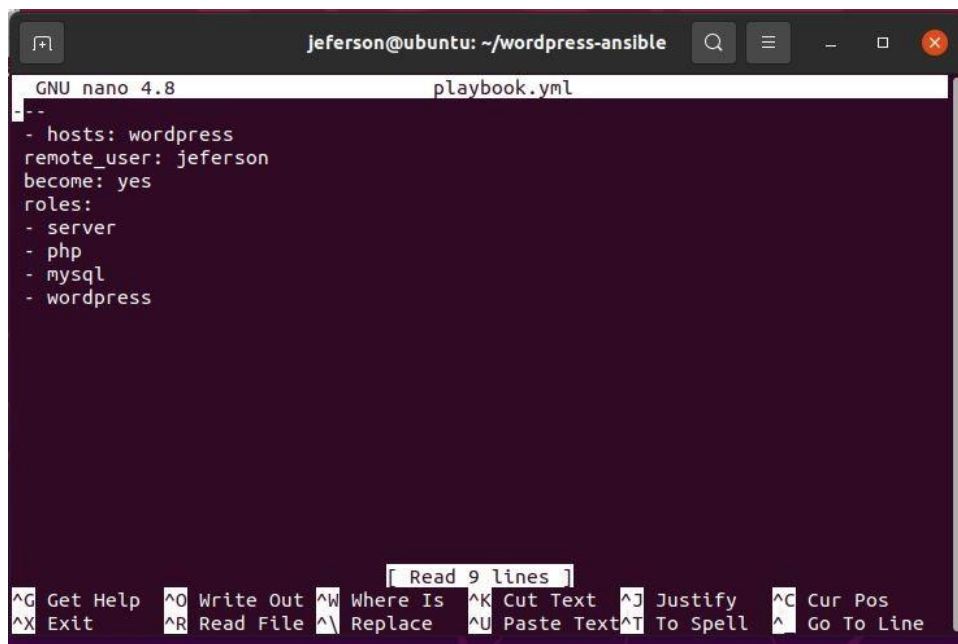
4 – Vamos escrever o nosso arquivo de inventário com o endereço IP da máquina onde vamos fazer a instalação, na pasta “roles” podemos editar o arquivo com comando “nano ~/wordpress-ansible/hosts”. Dentro do arquivo podemos indicar o endereço de vários servidores para a instalação automatizada da nossa aplicação, no nosso caso iremos utilizar somente o endereço de LoopBack 127.0.0.1.



```
jeferson@ubuntu: ~/wordpress-ansible/roles
GNU nano 4.8 /home/jeferson/wordpress-ansible/hosts
wordpress
127.0.0.1

Read 2 lines
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

5 – Para que o nosso playbook combine com as roles criadas anteriormente com o comando “ansible-galaxy” produzindo as configurações do WordPress. Na pasta “wordpress-ansible” vamos executar o comando “nano playbook.yml” e adicionar o conteúdo abaixo.



```
jeferson@ubuntu: ~/wordpress-ansible
GNU nano 4.8 playbook.yml
- -
- hosts: wordpress
  remote_user: jeferson
  become: yes
  roles:
  - server
  - php
  - mysql
  - wordpress

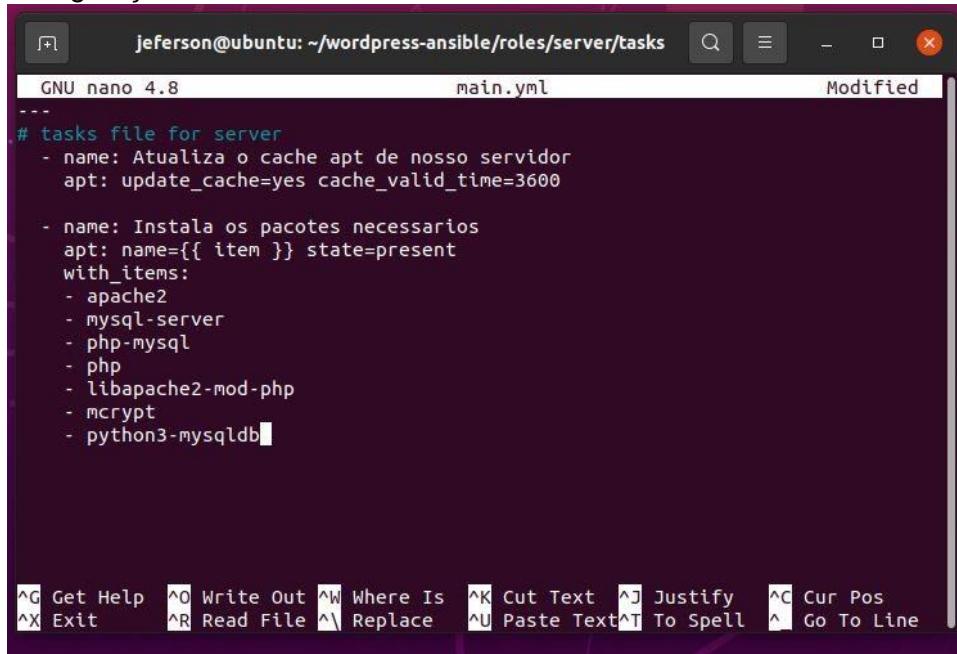
Read 9 lines
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

Escrevendo as Roles

Server

6 - Na pasta “wordpress-ansible/roles/server/tasks” vamos fazer as configurações básicas do nosso servidor com o comando “nano main.yml”, dentro do arquivo escreva a seguinte

configuração:



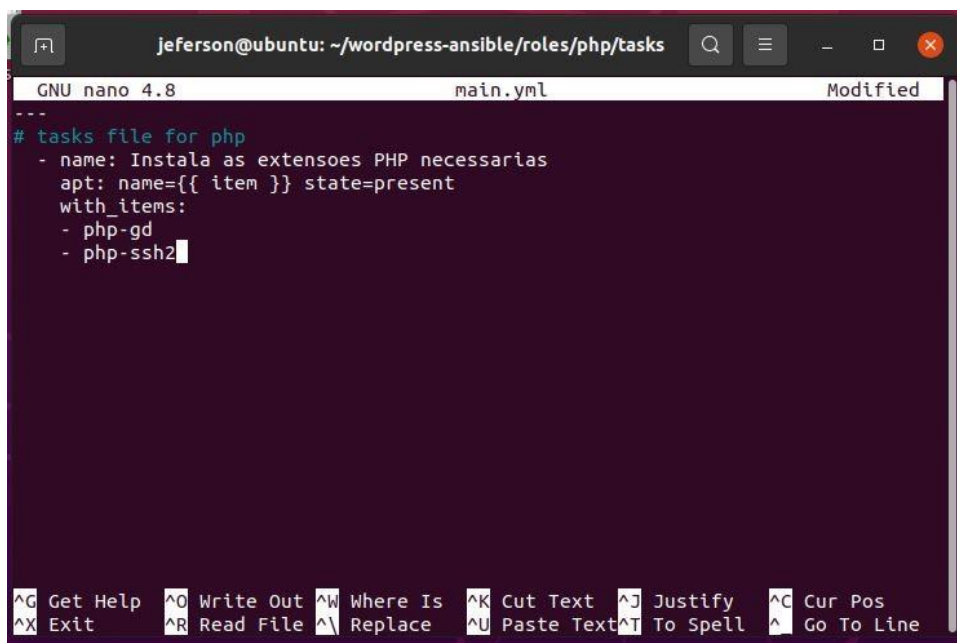
```
jefferson@ubuntu: ~/wordpress-ansible/roles/server/tasks
GNU nano 4.8      main.yml      Modified
---
# tasks file for server
- name: Atualiza o cache apt de nosso servidor
  apt: update_cache=yes cache_valid_time=3600

- name: Instala os pacotes necessarios
  apt: name={{ item }} state=present
  with_items:
    - apache2
    - mysql-server
    - php-mysql
    - php
    - libapache2-mod-php
    - mcrypt
    - python3-mysqldb
```

Na primeira parte do arquivo será feito um update baixando as configurações mais recentes dos pacotes disponíveis. Na segunda parte são instalados os pacotes necessários para o funcionamento do WordPress.

PHP

7 - Agora vamos fazer a instalação das extensões PHP, na pasta “wordpress-ansible/roles/php/tasks” vamos editar o arquivo main com o comando “nano main.yml” utilizando o seguinte código.

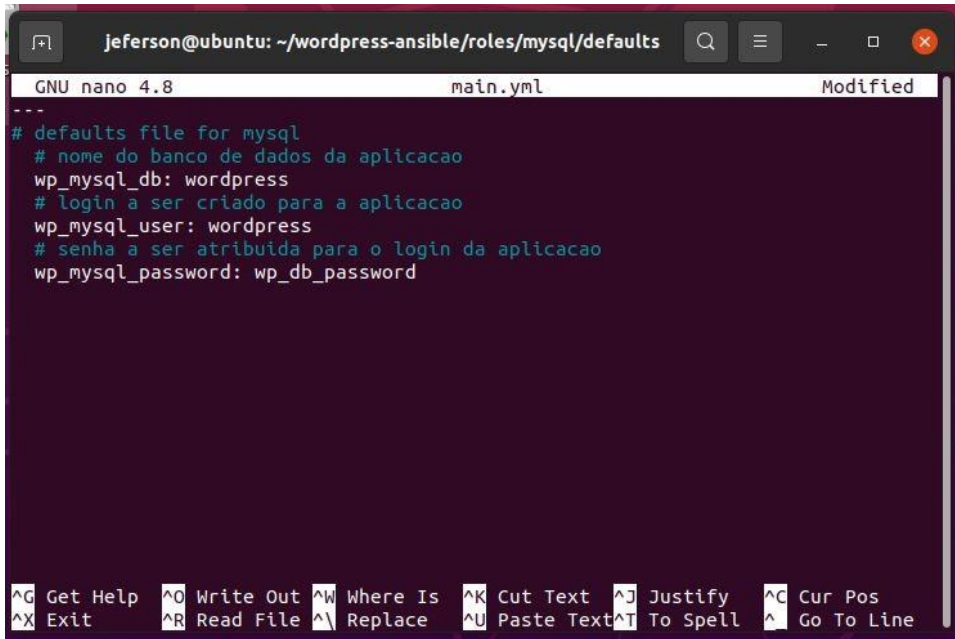


```
jefferson@ubuntu: ~/wordpress-ansible/roles/php/tasks
GNU nano 4.8      main.yml      Modified
---
# tasks file for php
- name: Instala as extensoes PHP necessarias
  apt: name={{ item }} state=present
  with_items:
    - php-gd
    - php-ssh2
```

MySQL

8 – Para a instalação do Banco de dados MySQL devemos acessar o diretório “wordpress-ansible/roles/mysql/defaults” e editar o arquivo main.yml.

Esse arquivo utilizara variáveis como parâmetros para uma fácil reutilização no futuro.

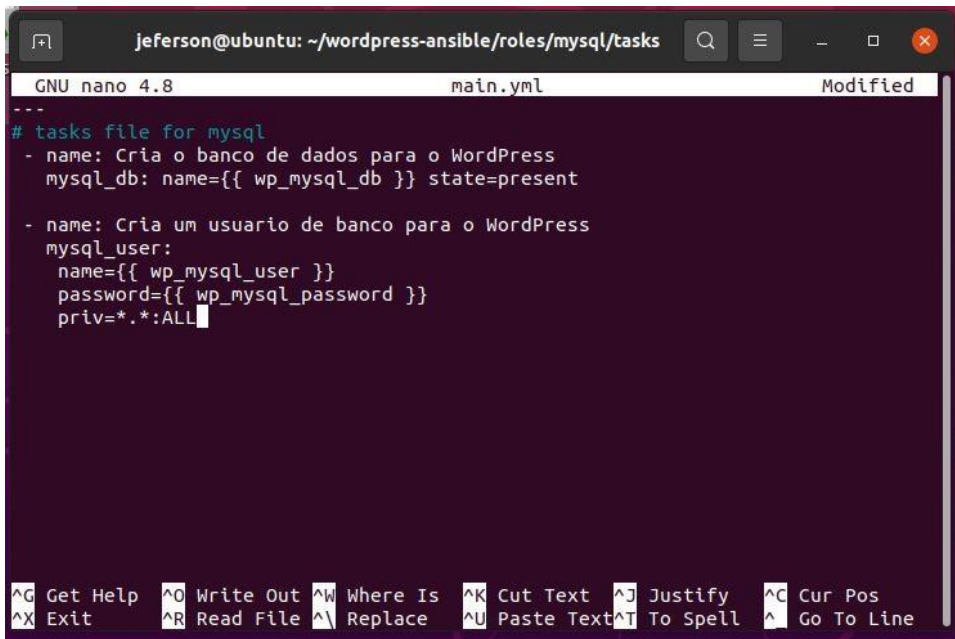


```
jefferson@ubuntu: ~/wordpress-ansible/roles/mysql/defaults
GNU nano 4.8 main.yml Modified
---
# defaults file for mysql
# nome do banco de dados da aplicacao
wp_mysql_db: wordpress
# login a ser criado para a aplicacao
wp_mysql_user: wordpress
# senha a ser atribuida para o login da aplicacao
wp_mysql_password: wp_db_password

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line
```

9 – No diretório Tasks “wordpress-ansible/roles/mysql/tasks” vamos editar o arquivo “main.yml”.

Essa role cria o banco de dados contendo as informações do wordpress, também cria uma conta de acesso dentro do bando de dados com login e senha já estabelecidos.



```
jefferson@ubuntu: ~/wordpress-ansible/roles/mysql/tasks
GNU nano 4.8 main.yml Modified
---
# tasks file for mysql
- name: Cria o banco de dados para o WordPress
  mysql_db: name={{ wp_mysql_db }} state=present

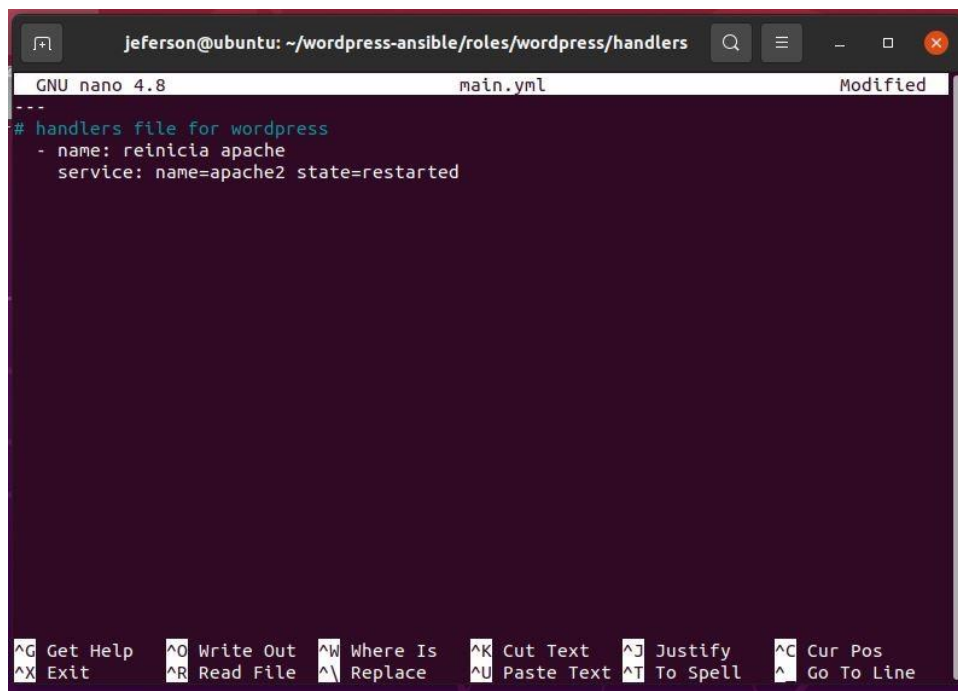
- name: Cria um usuario de banco para o WordPress
  mysql_user:
    name={{ wp_mysql_user }}
    password={{ wp_mysql_password }}
    priv=*.*:ALL

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line
```

WordPress

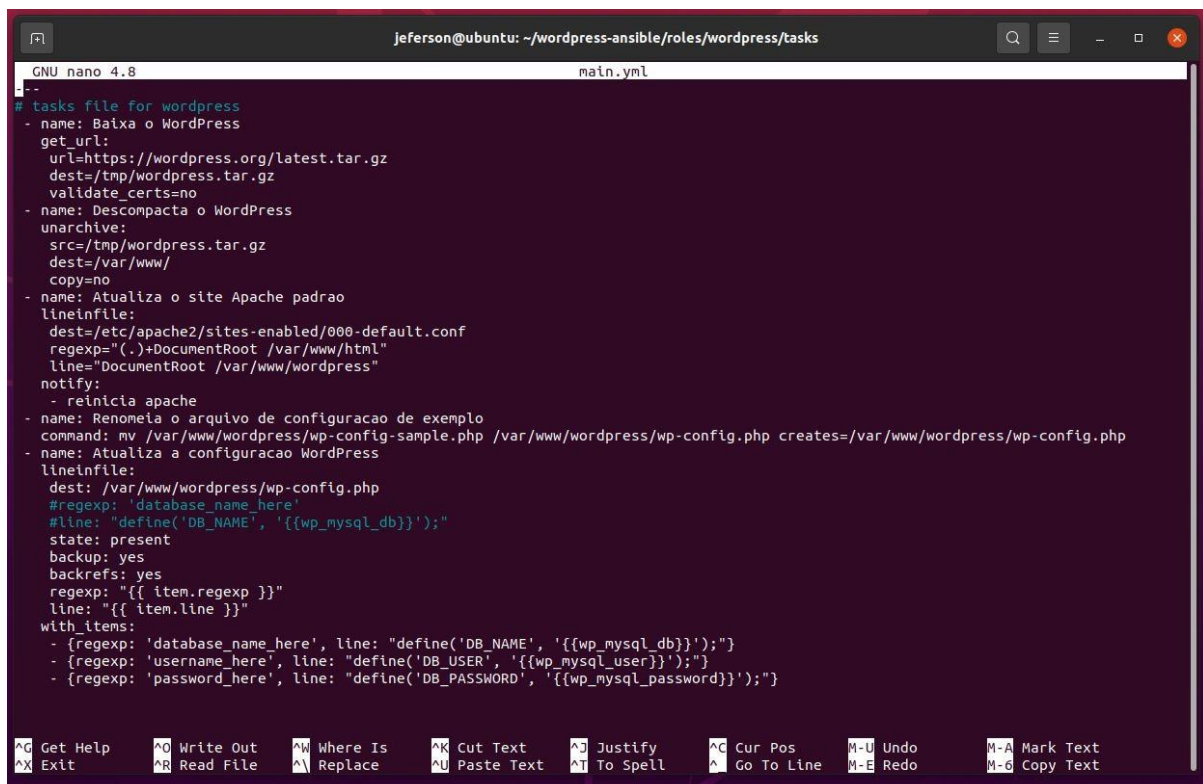
10 – Agora com todos os pré-requisitos definidos vamos fazer a instalação do WordPress acessando o diretório “wordpress-ansible/roles/wordpress/handlers” e editar o arquivo main.yml.

Esse código configura e reinicia o nosso servidor apache.



```
jefferson@ubuntu: ~/wordpress-ansible/roles/wordpress/handlers
GNU nano 4.8 main.yml Modified
---
# handlers file for wordpress
- name: reinicia apache
  service: name=apache2 state=restarted
```

11 – No diretório “wordpress-ansible/roles/wordpress/tasks” daremos o seguinte código para o arquivo main.yml:



```
jefferson@ubuntu: ~/wordpress-ansible/roles/wordpress/tasks
GNU nano 4.8 main.yml
---
# tasks file for wordpress
- name: Baixa o WordPress
  get_url:
    url=https://wordpress.org/latest.tar.gz
    dest=/tmp/wordpress.tar.gz
    validate_certs=no
- name: Descompacta o WordPress
  unarchive:
    src=/tmp/wordpress.tar.gz
    dest=/var/www/
    copy=no
- name: Atualiza o site Apache padrao
  lineinfile:
    dest=/etc/apache2/sites-enabled/000-default.conf
    regexp="(.)+DocumentRoot /var/www/html"
    line="DocumentRoot /var/www/wordpress"
  notify:
    - reinicia apache
- name: Renomeia o arquivo de configuracao de exemplo
  command: mv /var/www/wordpress/wp-config-sample.php /var/www/wordpress/wp-config.php creates=/var/www/wordpress/wp-config.php
- name: Atualiza a configuracao WordPress
  lineinfile:
    dest: /var/www/wordpress/wp-config.php
    #regexp: 'database_name_here'
    #line: "define('DB_NAME', '{{wp_mysql_db}}');"
    state: present
    backup: yes
    backrefs: yes
    regexp: "{{ item.regexp }}"
    line: "{{ item.line }}"
  with_items:
    - {regexp: 'database_name_here', line: "define('DB_NAME', '{{wp_mysql_db}}');"}
    - {regexp: 'username_here', line: "define('DB_USER', '{{wp_mysql_user}}');"}
    - {regexp: 'password_here', line: "define('DB_PASSWORD', '{{wp_mysql_password}}');"}
M-U Undo M-A Mark Text
M-E Redo M-G Copy Text
```


Esse código é responsável por baixar e descompactar o WordPress, atualizar e reiniciar o site apache e atualizar as configurações do WordPress, O código também informa à aplicação dados de login e senha para o acesso do nosso Banco de Dados MySQL.

12 - Agora podemos verificar se toda a nossa automação feita no ansible está em pleno funcionamento com o comando “ansible-playbook -i hosts playbook.yml”

O resultado deve ser parecido como o das figuras abaixo:

```
jeferson@ubuntu:~/wordpress-ansible$ ansible-playbook -i hosts playbook.yml

PLAY [wordpress] *****

TASK [Gathering Facts] *****
ok: [127.0.0.1]

TASK [server : Atualiza o cache apt de nosso servidor] *****
ok: [127.0.0.1]

TASK [server : Instala os pacotes necessarios] *****
[DEPRECATION WARNING]: Invoking "apt" only once while using a loop via squash_actions is deprecated. Instead of using a loop to supply multiple items and specifying 'name: "{{ item }}"', please use 'name: ['apache2', 'mysql-server', 'php-mysql', 'php', 'libapache2-mod-php', 'mcrypt', 'python3-mysqldb']' and remove the loop. This feature will be removed in version 2.11. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
changed: [127.0.0.1] => (item=['apache2', 'mysql-server', 'php-mysql', 'php', 'libapache2-mod-php', 'mcrypt', 'python3-mysqldb'])

TASK [php : Instala as extensoes PHP necessarias] *****
[DEPRECATION WARNING]: Invoking "apt" only once while using a loop via squash_actions is deprecated. Instead of using a loop to supply multiple items and specifying 'name: "{{ item }}"', please use 'name: ['php-gd', 'php-ssh2']' and remove the loop. This feature will be removed in version 2.11. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
changed: [127.0.0.1] => (item=['php-gd', 'php-ssh2'])

TASK [mysql : Cria o banco de dados para o WordPress] *****
changed: [127.0.0.1]

TASK [mysql : Cria um usuario de banco para o WordPress] *****
[WARNING]: Module did not set no_log for update_password
changed: [127.0.0.1]

TASK [wordpress : Baixa o WordPress] *****
changed: [127.0.0.1]

TASK [wordpress : Descompacta o WordPress] *****
changed: [127.0.0.1]

TASK [wordpress : Atualiza o site Apache padrao] *****
changed: [127.0.0.1]

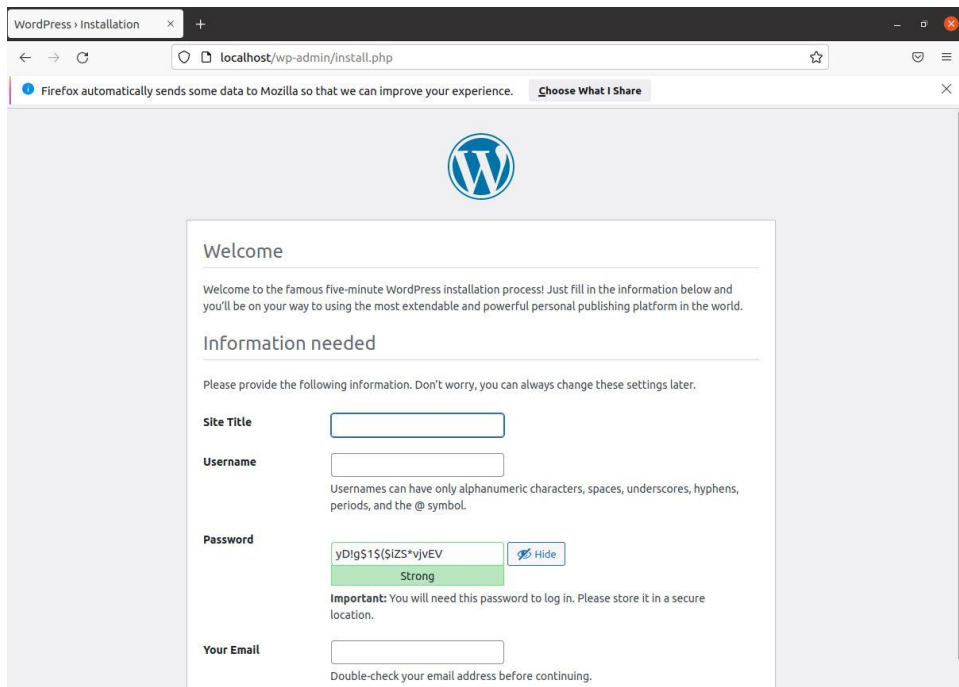
TASK [wordpress : Renomeia o arquivo de configuracao de exemplo] *****
changed: [127.0.0.1]

TASK [wordpress : Atualiza a configuracao WordPress] *****
changed: [127.0.0.1] => (item={'regexp': 'database_name_here', 'line': "define('DB_NAME', 'wordpress');"})
changed: [127.0.0.1] => (item={'regexp': 'username_here', 'line': "define('DB_USER', 'wordpress');"})
changed: [127.0.0.1] => (item={'regexp': 'password_here', 'line': "define('DB_PASSWORD', 'wp_db_password');"})

RUNNING HANDLER [wordpress : reinicia apache] *****
changed: [127.0.0.1]

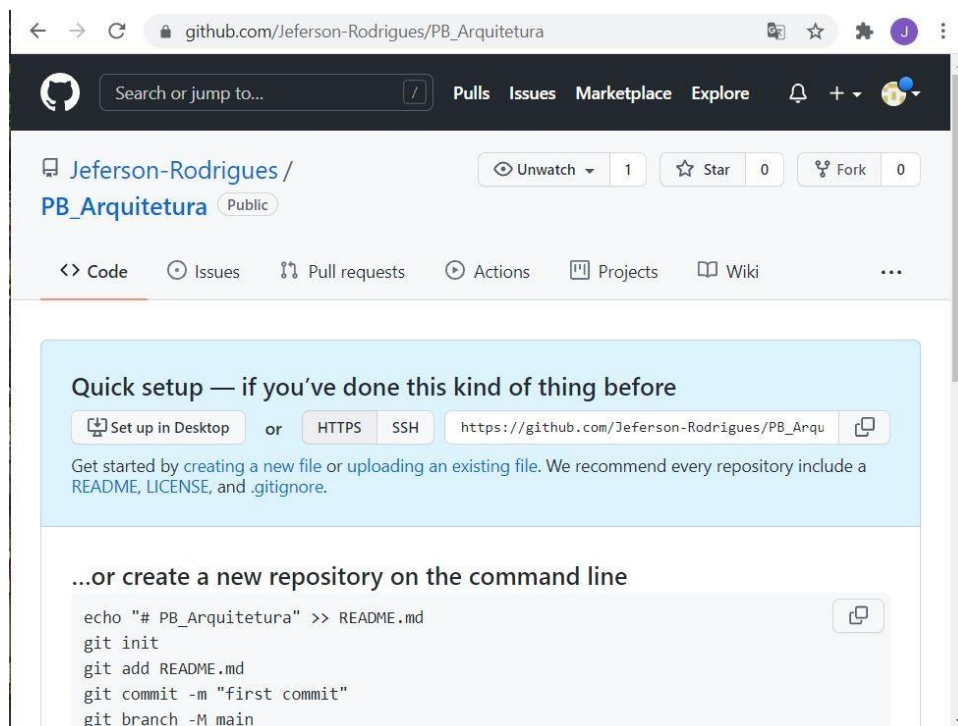
PLAY RECAP *****
127.0.0.1 : ok=12 changed=10 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0
```

13 – Agora finalmente podemos acessar a nossa aplicação WordPress pelo nosso navegador preferido digitando o “localhost” na barra de endereços.



Gerenciamento do Código-Fonte na GitHub

01 – A criação de um novo repositório na conta GitHub é muito simples e intuitiva, basta estar logado e clicar no botão com o sinal de adição na parte superior da tela e clicar em criar um novo repositório.



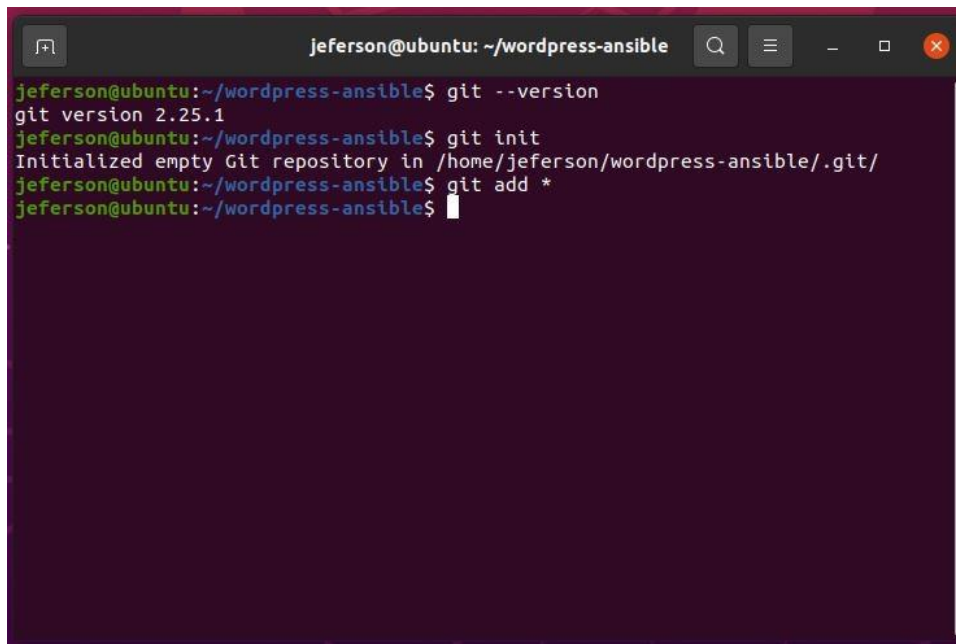
2 – Para que seja possível o acesso à nosso repositório do nosso computador local, será necessário criar uma chave de acesso Token.

Com a página do repositório aberto clique na sua figura de usuário no canto superior da tela > Settings > Developer Settings.

Crie um nome para o nosso token, defina uma data de expiração de acordo com o projeto e habilite todas as opções de acesso, por ultimo clique em Generate Token.

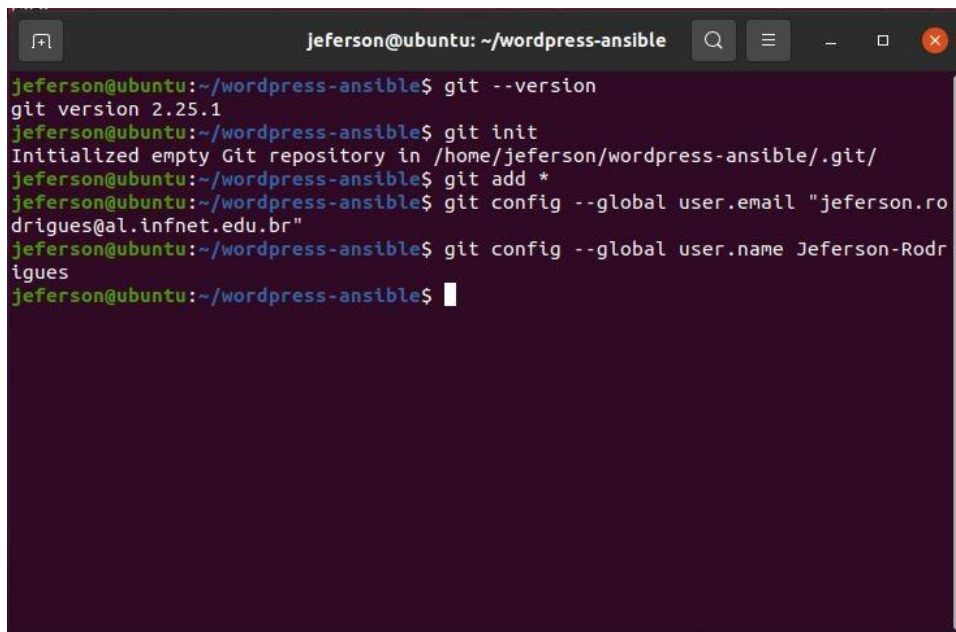
3 – Pronto, o nosso token foi criado, copie o código pois iremos utilizá-lo mais tarde.

2 – Dentro do nosso diretório local “wordpress-ansible” vamos fazer a preparação para utilizar o ansible com o comando “git init” e adicionar a nossa estrutura de arquivos com o comando “git add*”.

A terminal window titled 'jeferson@ubuntu: ~/wordpress-ansible' with search, menu, and window control icons. The terminal shows the following commands and output:

```
jeferson@ubuntu:~/wordpress-ansible$ git --version
git version 2.25.1
jeferson@ubuntu:~/wordpress-ansible$ git init
Initialized empty Git repository in /home/jeferson/wordpress-ansible/.git/
jeferson@ubuntu:~/wordpress-ansible$ git add *
jeferson@ubuntu:~/wordpress-ansible$
```

3 – Em seguida devemos informar o nosso email e usuário para a autenticação no GitHub como mostra a figura abaixo:

A terminal window titled 'jeferson@ubuntu: ~/wordpress-ansible' with search, menu, and window control icons. The terminal shows the following commands and output:

```
jeferson@ubuntu:~/wordpress-ansible$ git --version
git version 2.25.1
jeferson@ubuntu:~/wordpress-ansible$ git init
Initialized empty Git repository in /home/jeferson/wordpress-ansible/.git/
jeferson@ubuntu:~/wordpress-ansible$ git add *
jeferson@ubuntu:~/wordpress-ansible$ git config --global user.email "jeferson.rodrigues@al.infnet.edu.br"
jeferson@ubuntu:~/wordpress-ansible$ git config --global user.name Jeferson-Rodrigues
jeferson@ubuntu:~/wordpress-ansible$
```

4 – Agora vamos fazer o nosso commit com todos os nossos arquivos configurados do diretório “wordpress-ansible” para a criação do nosso projeto local.


```
jeferson@ubuntu: ~/wordpress-ansible
jeferson@ubuntu:~/wordpress-ansible$ git commit -m "importando playbook WordPress"
[master (root-commit) 41cff6d] importando playbook WordPress
40 files changed, 620 insertions(+)
create mode 100644 echo
create mode 100644 hosts
create mode 100644 hosts.yml
create mode 100644 playbook.yml
create mode 100644 roles/mysql/.travis.yml
create mode 100644 roles/mysql/README.md
create mode 100644 roles/mysql/defaults/main.yml
create mode 100644 roles/mysql/handlers/main.yml
create mode 100644 roles/mysql/meta/main.yml
create mode 100644 roles/mysql/tasks/main.yml
create mode 100644 roles/mysql/tests/inventory
create mode 100644 roles/mysql/tests/test.yml
create mode 100644 roles/mysql/vars/main.yml
create mode 100644 roles/php/.travis.yml
create mode 100644 roles/php/README.md
create mode 100644 roles/php/defaults/main.yml
create mode 100644 roles/php/handlers/main.yml
create mode 100644 roles/php/meta/main.yml
create mode 100644 roles/php/tasks/main.yml
create mode 100644 roles/php/tests/inventory
create mode 100644 roles/php/tests/test.yml
```

5 - Agora devemos resgatar o link do nosso repositório no GitHub.

```
jeferson@ubuntu: ~/wordpress-ansible
create mode 100644 roles/php/tasks/main.yml
create mode 100644 roles/php/tests/inventory
create mode 100644 roles/php/tests/test.yml
create mode 100644 roles/php/vars/main.yml
create mode 100644 roles/server/.travis.yml
create mode 100644 roles/server/README.md
create mode 100644 roles/server/defaults/main.yml
create mode 100644 roles/server/handlers/main.yml
create mode 100644 roles/server/meta/main.yml
create mode 100644 roles/server/tasks/main.yml
create mode 100644 roles/server/tests/inventory
create mode 100644 roles/server/tests/test.yml
create mode 100644 roles/server/vars/main.yml
create mode 100644 roles/wordpress/.travis.yml
create mode 100644 roles/wordpress/README.md
create mode 100644 roles/wordpress/defaults/main.yml
create mode 100644 roles/wordpress/handlers/main.yml
create mode 100644 roles/wordpress/meta/main.yml
create mode 100644 roles/wordpress/tasks/main.yml
create mode 100644 roles/wordpress/tests/inventory
create mode 100644 roles/wordpress/tests/test.yml
create mode 100644 roles/wordpress/vars/main.yml
jeferson@ubuntu:~/wordpress-ansible$ git remote add origin https://github.com/Jeferson-Rodrigues/PB_Arquitetura
```

6 - Vamos enviar a nossa Git local para o nosso repositório remoto com o comando "git push -u origin master"

```
jeferson@ubuntu: ~/wordpress-ansible
create mode 100644 roles/php/tests/inventory
create mode 100644 roles/php/tests/test.yml
create mode 100644 roles/php/vars/main.yml
create mode 100644 roles/server/.travis.yml
create mode 100644 roles/server/README.md
create mode 100644 roles/server/defaults/main.yml
create mode 100644 roles/server/handlers/main.yml
create mode 100644 roles/server/meta/main.yml
create mode 100644 roles/server/tasks/main.yml
create mode 100644 roles/server/tests/inventory
create mode 100644 roles/server/tests/test.yml
create mode 100644 roles/server/vars/main.yml
create mode 100644 roles/wordpress/.travis.yml
create mode 100644 roles/wordpress/README.md
create mode 100644 roles/wordpress/defaults/main.yml
create mode 100644 roles/wordpress/handlers/main.yml
create mode 100644 roles/wordpress/meta/main.yml
create mode 100644 roles/wordpress/tasks/main.yml
create mode 100644 roles/wordpress/tests/inventory
create mode 100644 roles/wordpress/tests/test.yml
create mode 100644 roles/wordpress/vars/main.yml
jeferson@ubuntu:~/wordpress-ansible$ git remote add origin https://github.com/Jeferson-Rodrigues/PB_Arquitetura
jeferson@ubuntu:~/wordpress-ansible$ git push -u origin master
```

7 – Nessa etapa será requerido o nosso usuário e depois nossa senha, no campo password vamos colar a nossa Chave Token criada anteriormente.

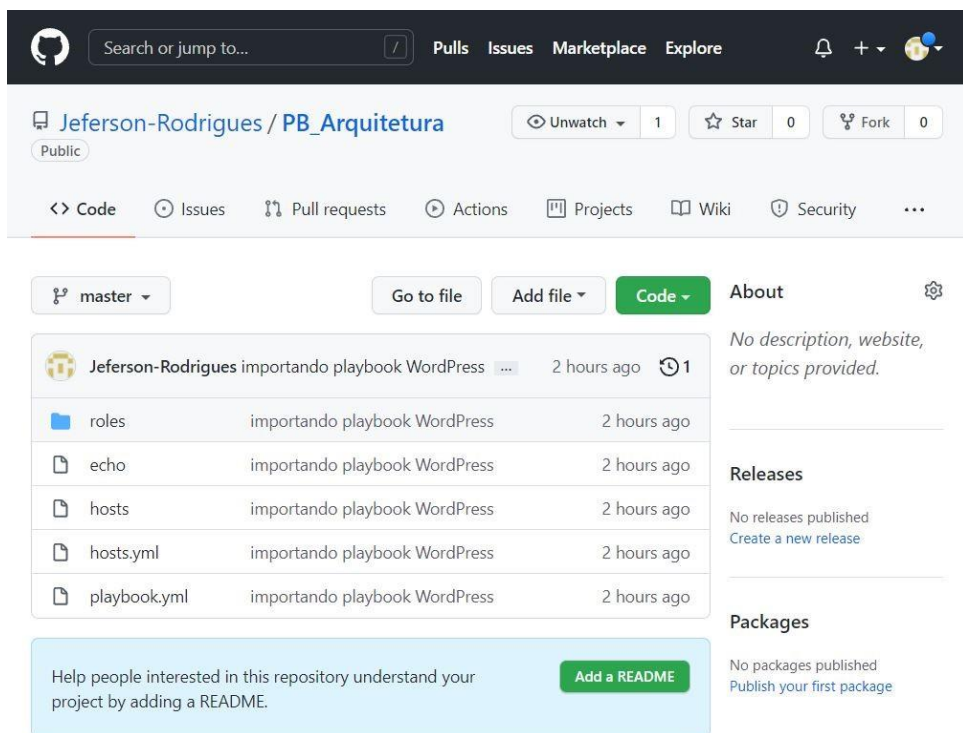
Atenção: A chave token é oculta e não irá aparecer depois de colar, pressione Enter.

```
jeferson@ubuntu: ~/wordpress-ansible
create mode 100644 roles/php/vars/main.yml
create mode 100644 roles/server/.travis.yml
create mode 100644 roles/server/README.md
create mode 100644 roles/server/defaults/main.yml
create mode 100644 roles/server/handlers/main.yml
create mode 100644 roles/server/meta/main.yml
create mode 100644 roles/server/tasks/main.yml
create mode 100644 roles/server/tests/inventory
create mode 100644 roles/server/tests/test.yml
create mode 100644 roles/server/vars/main.yml
create mode 100644 roles/wordpress/.travis.yml
create mode 100644 roles/wordpress/README.md
create mode 100644 roles/wordpress/defaults/main.yml
create mode 100644 roles/wordpress/handlers/main.yml
create mode 100644 roles/wordpress/meta/main.yml
create mode 100644 roles/wordpress/tasks/main.yml
create mode 100644 roles/wordpress/tests/inventory
create mode 100644 roles/wordpress/tests/test.yml
create mode 100644 roles/wordpress/vars/main.yml
jeferson@ubuntu:~/wordpress-ansible$ git remote add origin https://github.com/Jeferson-Rodrigues/PB_Arquitetura
jeferson@ubuntu:~/wordpress-ansible$ git push -u origin master
Username for 'https://github.com': Jeferson-Rodrigues
Password for 'https://Jeferson-Rodrigues@github.com':
```

10 – Caso não apareça nenhuma mensagem de erro o nosso procedimento foi concluído com sucesso.

```
jeferson@ubuntu: ~/wordpress-ansible
create mode 100644 roles/wordpress/README.md
create mode 100644 roles/wordpress/defaults/main.yml
create mode 100644 roles/wordpress/handlers/main.yml
create mode 100644 roles/wordpress/meta/main.yml
create mode 100644 roles/wordpress/tasks/main.yml
create mode 100644 roles/wordpress/tests/inventory
create mode 100644 roles/wordpress/tests/test.yml
create mode 100644 roles/wordpress/vars/main.yml
jeferson@ubuntu:~/wordpress-ansible$ git remote add origin https://github.com/Jeferson-Rodrigues/PB_Arquitetura
jeferson@ubuntu:~/wordpress-ansible$ git push -u origin master
Username for 'https://github.com': Jeferson-Rodrigues
Password for 'https://Jeferson-Rodrigues@github.com':
Enumerating objects: 56, done.
Counting objects: 100% (56/56), done.
Delta compression using up to 2 threads
Compressing objects: 100% (25/25), done.
Writing objects: 100% (56/56), 6.16 KiB | 573.00 KiB/s, done.
Total 56 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/Jeferson-Rodrigues/PB_Arquitetura
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
jeferson@ubuntu:~/wordpress-ansible$
```

11 – Agora podemos conferir se os arquivos foram enviados para o nosso repositório GitHub na nuvem.



Link do repositório: https://github.com/Jeferson-Rodrigues/PB_Arquitetura

A solução ideal para o projeto (TO BE)

As soluções do projeto apresentadas foram baseadas nos recursos disponibilizados até o momento pela empresa, a seguir vamos apresentar melhorias sobre uma visão técnica qual seria o ambiente ideal para uma solução completa do nosso cliente.

Por motivos de redundância seriam necessários dois servidores físicos trabalhando com virtual protection garantindo a disponibilidade das maquina virtuais mesmo em caso de falha em um dos servidores.

A solução total para a proteção dos dados seria utilizar o serviço RDS da Cloud AWS como cópia do banco de dados (backup)

Para a Rede física a melhor opção seria utilizar 2 Switches e 2 Roteadores interligando os servidores físicos e o Banco de Dados, além de 2 links de conexão com a internet para manter os serviços funcionando caso um provedor entre em manutenção.

Diagrama Físico Solução Ideal

(2x) Servidor

Cpu de 12 cores
32GB RAM
1TB de Disco
1 Ethernet: Para conexão com o Switch
1 Ethernet: Para futuras expansões e manutenção do servidor

Storage

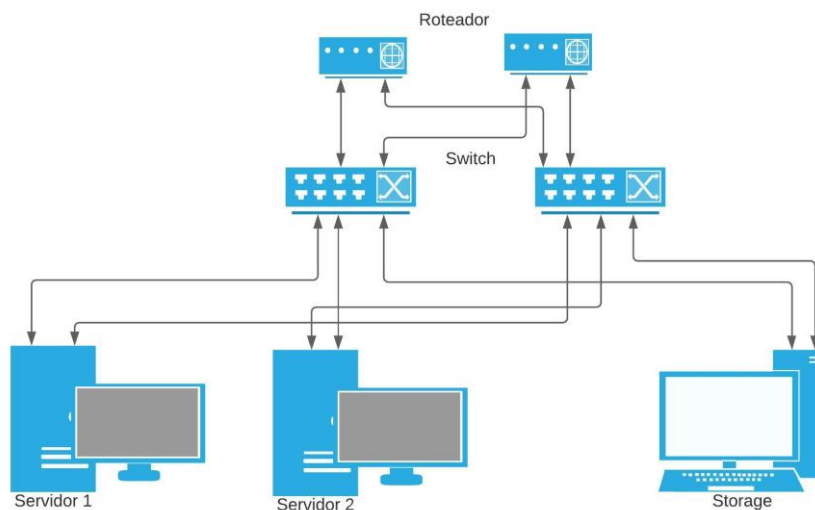
Cpu 4 Nucleos
8GB RAM
120 GB Disco
1 Ethernet: Para conexão com o Switch
1 Ethernet: Para futuras expansões e manutenção
10 Discos de 5TB

(2x) Switch

1 Gbps
8 Portas

(2x) Roteador

1 Gbps
4 portas



CRONOGRAMA

19/11 - Entrega do esboço da documentação

30/11 - Entrega da documentação completa

07/12 - Apresentação do Projeto

Os prazos estabelecidos no início do projeto foram adequados porém ocorreram atrasos na entrega da documentação definida para o dia 31/11 pois ocorreram problemas durante os testes da implantação da aplicação.

Conclusão

Com a conclusão do projeto podemos avaliar que os recursos planejados foram suficientes para colocar a aplicação em perfeito funcionamento.

O novo portal foi um sucesso, a Techinvest está conseguindo transitar seu nicho de atuação de maneira sólida e cada vez mais se posiciona como empresa referência quando o assunto é investimento no exterior em países emergentes. Esse novo posicionamento promove uma vantagem de mercado para a companhia podendo aumentar suas taxas de administração mantendo a excelência na entrega de seus serviços. O site bem posicionado aliado a uma campanha de marketing acertava a Techinvest continua captando novos clientes em todo o mundo e todos esses benefícios se refletem no aumento do faturamento da empresa.

Links da apresentação:

[01 - Apresentação Projeto de Bloco - TechInvest](#)

[02 - Demonstração da Automação da Aplicação - TechInvest](#)