

INSTITUTO INFNET

Rodrigo de Oliveira Menezes



Projeto de Bloco: Arquitetura de Infraestrutura de Aplicações ASSESSMENT

Rio de Janeiro, 02 De Dezembro de 2021.

ÍNDICE.

CAPITULO I

1.1-Introdução-----	Pag. 03
1.2 Planejamento de Infraestrutura.-----	Pag. 04.
Diagrama Físico-----	Pag. 04.
Diagrama Virtual-----	Pag.05.
1.3 Atualizações, Correções de Bugs e requisição de Desenvolvimento---	Pag. 08.

CAPITULO II

2.1 Razões para o uso de VMs em servidores próprios e não AWS.-----	Pag. 09.
2.2 Previsão Orçamentária.-----	Pag. 10.
2.2 Planejamento e Cronograma.-----	Pag. 10

CAPITULO III

3.1 Ref. dos Downloads e versões dos componentes da solução.-----	Pag. 11
3.2 Implementação de Infraestrutura e Configuração de Aplicação.-----	Pag. 12
3.3 Procedimento de Criação de Repositório Público Utilizando GitHub.---	Pag. 29

CAPITULO IV

4.1 Conclusão-----	Pag. 42
To be-----	Pag. 42

1-Capítulo I

1.1-Introdução:

O escritório de advocacia SIQUEIRA&MENEZES é um escritório corporativo com mais de 20 (vinte) advogados prestando acessória, consultoria e representação jurídica na área cível e criminal.

Para trabalho integrado dos advogados e melhor troca de experiência em todos, será desenvolvido uma aplicação que busca facilitar a troca de experiência de sucesso em processos através de um banco de Petições com peças processuais que poderão ser acessadas e baixadas, aumentando assim a produtividade do trabalho e a chance de êxito nos processos.

Cada advogado usuário poderá cadastrar no Banco de Petições, uma ou mais petição, para que outro colega, acessando o banco de petições, possa baixar a petição e adapta lá para o seu caso concreto.

O Banco de Petições terá as petições catalogadas e divididas em dois grandes grupos sendo o grupo Petições Cível e o grupo Petições Criminais.

O presente documento tem por objetivo descrever o processo de implementação dessa aplicação através da metodologia de integração contínua DevOps.

1.2 Planejamentos de Infraestrutura.

O acesso ao Banco de Petições se dará através do computador da empresa, o qual o colaborador advogado entrará utilizando-se de login e senha de acesso pré-cadastrado. Uma vez acessado o servidor, o advogado, após inserir seu Login e senha, terá acesso a um banco de Petições salvo dentro do servidor, conforme projeto abaixo:

Diagrama Físico:

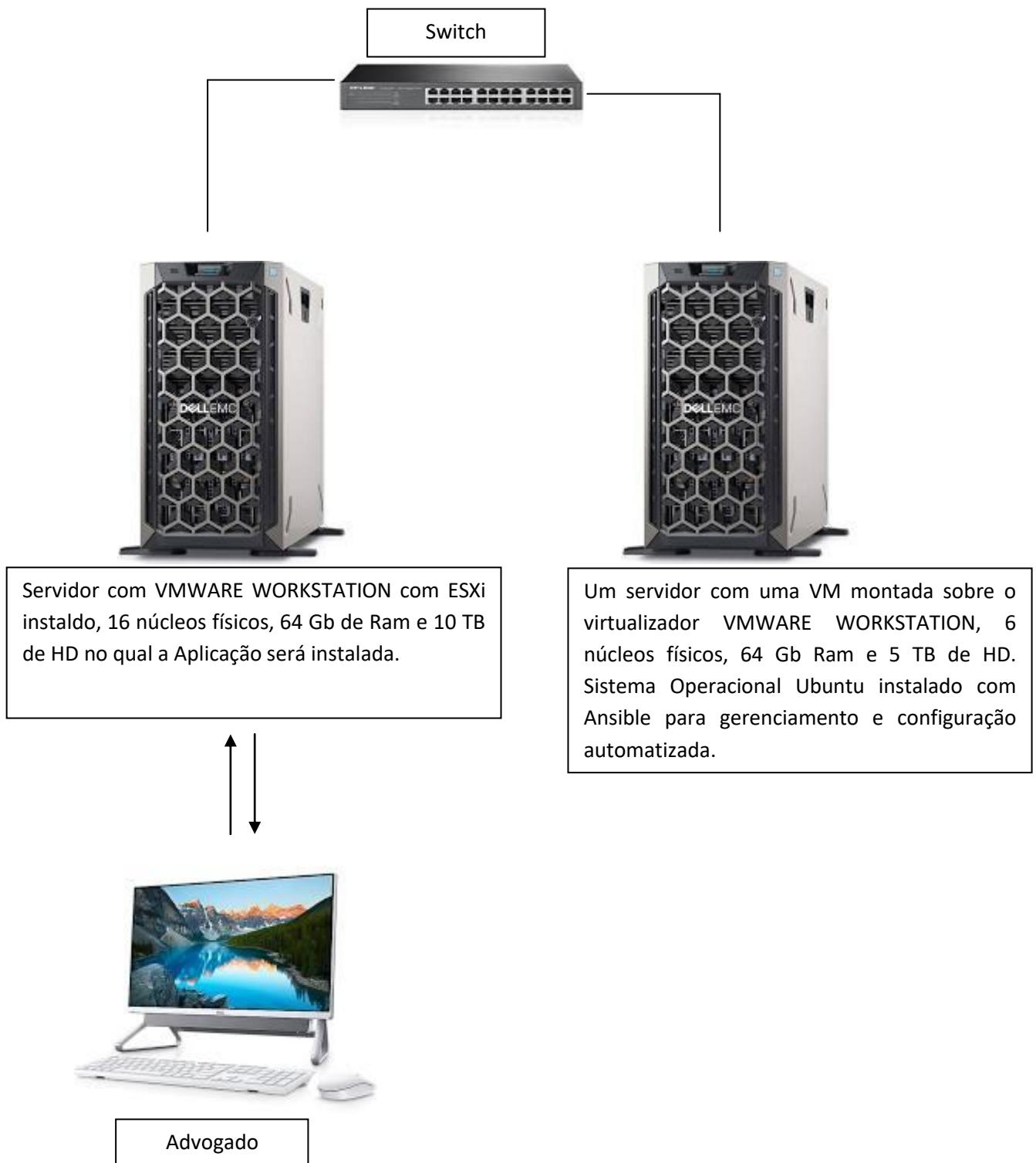
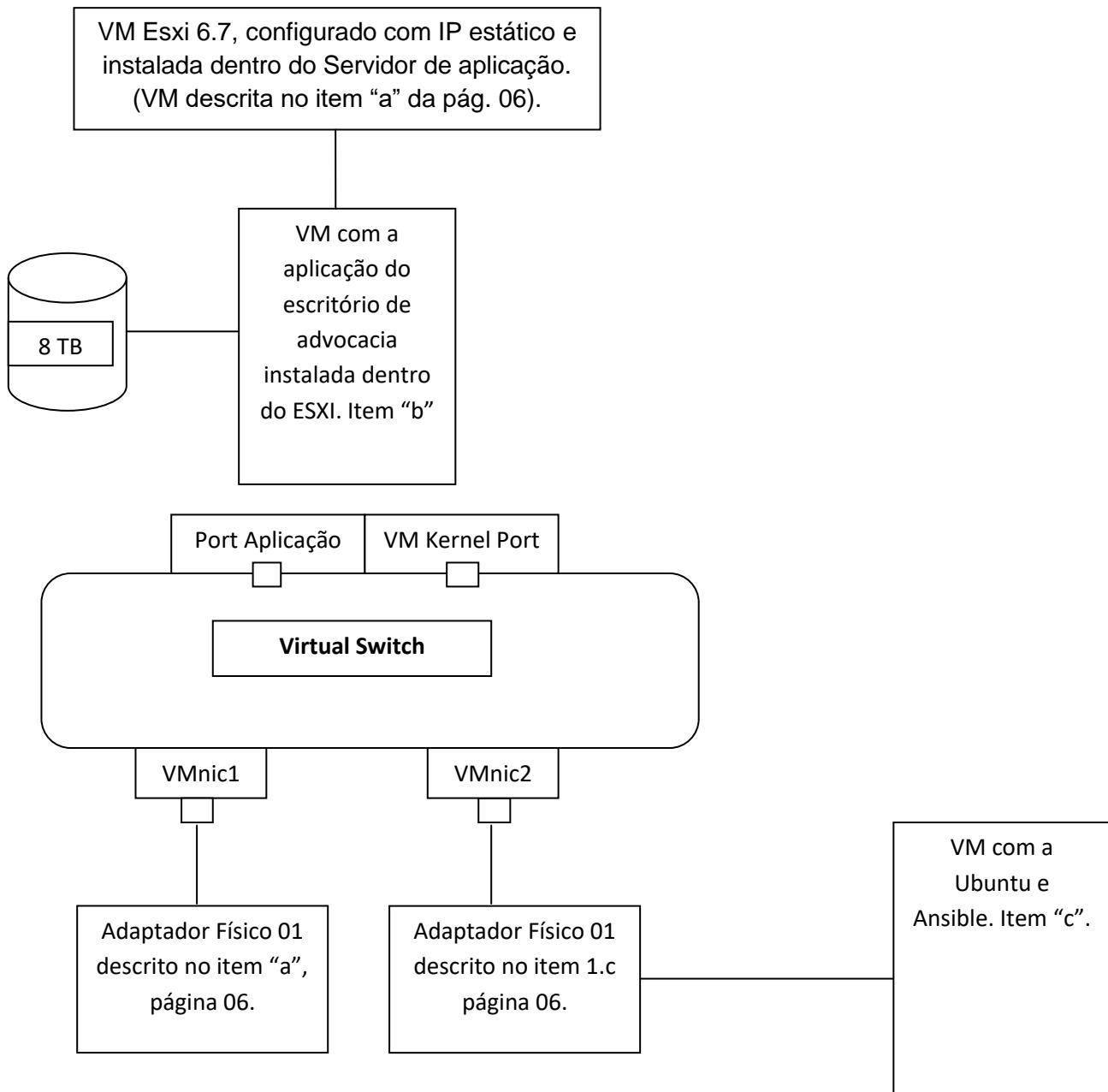


Diagrama Virtual:



a) Especificação do Servidor Físico da Aplicação:

- Intel Core I7, 64 GB de RAM, 16 vCPU e 10 TB HD.
- Sistema Operacional Windows Server 2019.
- Virtualizador VMware Workstation 16 Pro.
- VM Esxi 6.7, configurado com IP estático.
- 1 placas de rede físicas.

b) Especificação da VM instalada no Exsi:

- Sistema Operacional- Ubuntu 20.04.3 LTS, 32 GB de RAM, 8 vCPU e 8 TB HD
- Aplicação da SIQUEIRA&MENEZES com base no Wordpress versão 5.8.2 linguagem PHP versão 8.0.3 (4 de março de 2021) e banco de dados MySQL versão 8.0.21 (13 de julho de 2020).
- 1 Vnic.

c) Especificação do Servidor Físico do Ansible.

- Intel Core I7, 64 GB de RAM, 16 vCPU e 5 TB HD.
- Sistema Operacional Windows Server 2019.
- Virtualizador VMware Workstation 16 Pro.
- VM com Sistema Operacional- Ubuntu 20.04.3 LTS.
- Ansible versão 3.0.
- 1 placas de rede físicas.

c) Switche com no mínimo 8 portas.

Ferramentas de busca permitirão o usuário buscar a petição por Área de atuação (cível ou criminal) e tipo de petição.

Como nesse projeto o cliente não dispõe de muitos recursos, a infraestrutura tem que ser a mais “enxuta” possível, e para ficar dentro do orçamento, o planejamento envolve utilizar Máquinas Virtuais, Software de código fonte aberto como Ansible, Sistema Operacional Ubutun e Wordpress.

Para o servidor da aplicação será usado uma VM (Máquina Virtual) que será o nó gerenciado, e uma segunda VM que será o nó de controle contendo o Ansible.

Ansible é uma ferramenta de software de código aberto que fornece de forma fácil e simples a automação de um ou mais computadores. Ele fará toda a gestão da página do escritório de forma automática e facilitará a atualização de versões e correção de bugs.

Já o Protocolo SSH é um protocolo que garante que cliente e servidor remoto troquem informações de maneira segura e dinâmica.

O portal Web será desenvolvido na plataforma do Wordpress baseado em PHP e banco de dados MySQL.

O Wordpress é sistema livre e aberto de gestão de conteúdo de internet, é a ferramenta que será usada para “confeccionar” a página Web. Já o PHP é a linguagem de computação que será usada para criação da página web. Por último temos o MySQL, que é um sistema de código fonte aberto que gerencia todo o banco de dados.

1.3 Atualizações, Correções de Bugs e requisição de Desenvolvimento.

As atualizações, correções de bugs e requisição de Desenvolvimento serão feitos através de GitHub, o qual permitirá a hospedagem de controle de versão e, também, que os projetos de desenvolvimento sejam feitos por mais de uma pessoa e sejam compartilhados, sendo a aplicação implantada pelo o Wordpress.

Git é uma ferramenta de controle de versão de códigos, ou seja, ele é útil para gerenciar as versões de um código, principalmente quando o trabalho é desenvolvido em equipe, como a maioria dos cenários de trabalho hoje em dia. Dessa forma, o Git permite que se trabalhe por duas ou mais pessoas diferentes sem que um “danifique” o que já foi feito pelo outro profissional.

O GitHub é a mesma coisa que o Git, só que com possibilidade de deixar ele público na internet, ou seja, é uma plataforma onde o código ficará armazenado publicamente o código para que possa ser compartilhado entre as pessoas que fazem parte do projeto.

2. Capítulo II

2.1 Razões para o uso de VMs em servidores próprios e não AWS.

Para este projeto utilizaremos Maquinas Virtuais (VM).

Como se trata de um banco de dados com processo e informações de clientes, que muita vez correm em segredo de justiça, é importante para a empresa ter a segurança de gerir sua própria infraestrutura. Além disso, há previsibilidade no orçamento e o patrimônio gerado com a compra do equipamento passará a ser ativos da empresa trazendo, inclusive, benefícios fiscais.

Diferentemente da VM, o sistema AWS, embora seja mais barato, não fornece uma gestão própria de estrutura, o orçamento pode ser alterado significativamente ao longo prazo, pois pode sofrer muito com flutuação do preço principalmente devido a variação cambial, e todo o custo/gasto não são convertidos como ativo da empresa, o que impede projetos futuros de crescimento com a atração de novos sócios investidores e advogados, uma vez que na AWS a maior parte da infraestrutura como servidores, memórias, banco de dados e etc, são alugados.

2.2. Previsão Orçamentária.

- a) 1 Servidor com Intel Core I7, 64 GB de RAM, 16 vCPU, 10 TB HD e com Sistema Operacional Windows Server 2019. R\$ 38.000,00
- b) 1 Servidor com Intel Core I7, 64 GB de RAM, 16 vCPU, 5 TB HD e Sistema Operacional Windows Server 2019. R\$ 33.000,00
- c) 2 placas de rede físicas. R\$ 1.300,00
- d) 1 Switch com no mínimo 8 portas. R\$ 1.700,00
- e) Workstation 16 Pro. R\$ 880,00
- f) Sistema Operacional- Ubuntu 20.04.3 LTS. Grátis
- g) Ansible versão 3.0 Grátis.
- h) PHP versão 8.0.3 (4 de março de 2021) Grátis
- i) MySQL versão 8.0.21 (13 de julho de 2020) Grátis
- j) Wordpress versão 5.8.2 Grátis

TOTAL: R\$ 74.880,00 (setenta e quatro mil oitocentos e oitenta reais)

2.3 Planejamento e Cronograma.

Para finalização do projeto teremos o prazo de 15 dias conforme cronograma.

- 1º Dia- Compra dos equipamentos, periféricos, cabos e etc.
- 2º ao 4º Dia- Montagem do equipamento e cabeamento.
- 5º e 6º Dia- instalação dos softwares.
- 7º Dia- Montagem das VMs com o esxi e workstation
- 8º Dia- Configuração Ansible e GitHub.
- 9º ao 12º Dia- Montagem da Pagina Web no Wordpress.
- 13º ao 15º Dia – Teste de funcionamento e correção de bugs.

3. Capítulo III.

3.1 Ref. dos Downloads e versões dos componentes da solução.

- a) VMware Workstation Pro.
- b) Sistema Operacional- Ubuntu 20.04.3 LTS.
- c) Ansible versão 3.0
- d) PHP versão 8.0.3 (4 de março de 2021)
- e) MySQL versão 8.0.21 (13 de julho de 2020)
- f) Wordpress versão 5.8.2
- g) Github versão 2.31.0 (15 de março de 2021)
- h) Windows 10 Pro.
- i) Esxi 6.7

3.2 Planejamento e Implementação de Infraestrutura e Configuração de Aplicação.

Com já mencionado na introdução, esse projeto pretende criar uma pagina web de um escritório de advocacia instalado em um servidor Web que permitirá aos usuários acessarem a pagina, buscar e incluir modelos de petições.

A página será “confeccionada” utilizando o WordPress e instalada no servidor Web de forma automatizada utilizando o Ansible/docker que, através de comunicação SSH com o servidor, irá implementar a página no servidor Web.

Neste projeto, como não será possível ter várias maquina, principalmente devido a recursos técnicos e financeiros, e como o intuito é demonstrar o conceito faremos todo esse procedimento em um só computador, *all in one*, contudo, toda a infraestrutura do projeto na vida real já está demonstrada no item 1.2 deste projeto.

Na vida real o primeiro passo é instalar o Ansible em uma maquina que chamaremos aqui de maquina Master, nessa maquina faremos rotinas através do Ansible, que podem ser desde instalações de aplicações a atualizações e correções de bugs, e essas rotinas são repassadas e aplicadas a todos servidores através da comunicação SSH.

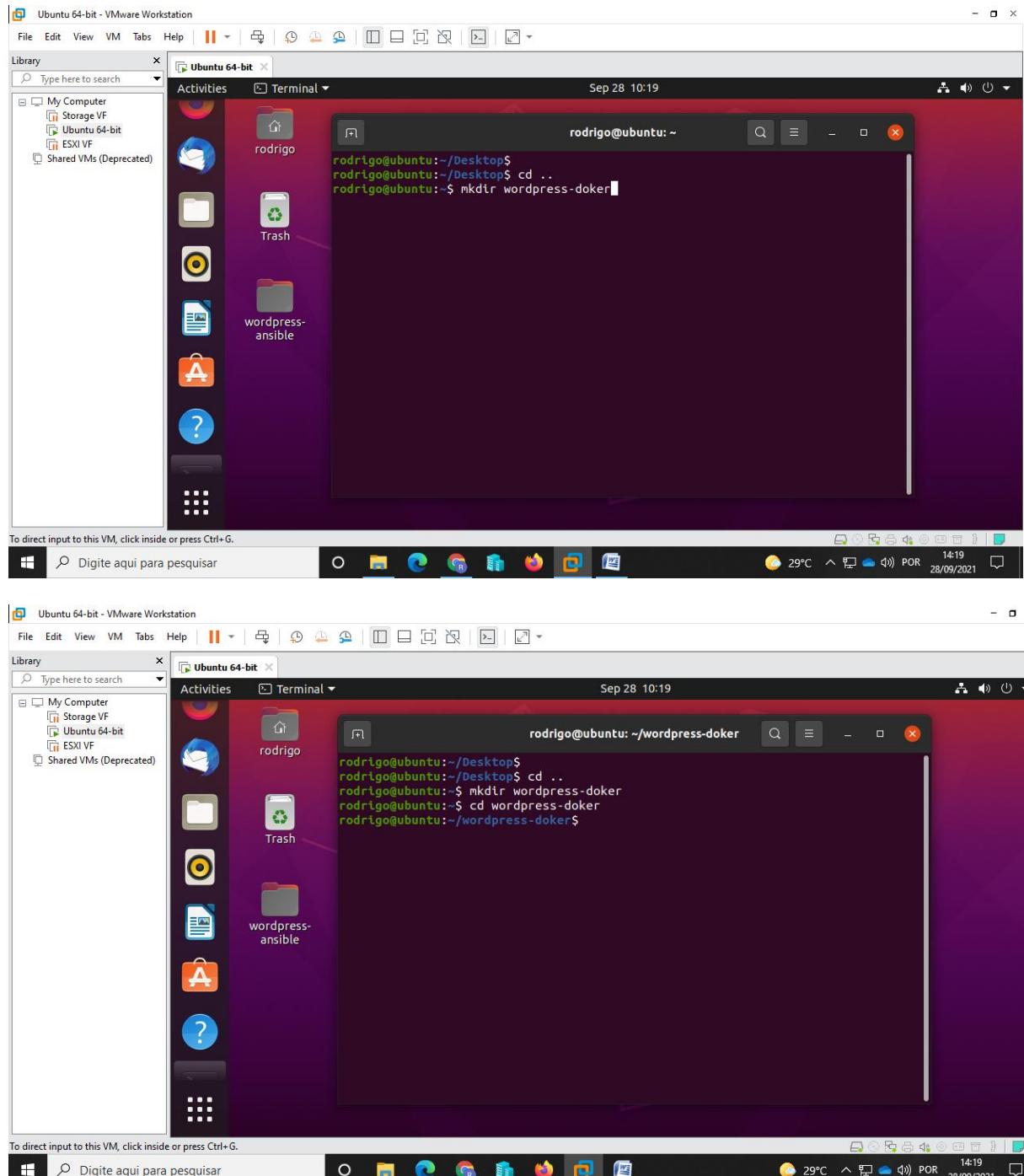
Para poder confeccionar a página, precisaremos de containers de Wordpress com Mysql e PHP. O presente trabalho pretende documentar a instalação e execução desses containers dokers.

Para instalação desses containers utilizamos o Ansible, que tem o objetivo principal de automatizar a instalação dos containers no servidor.

1º PASSO.

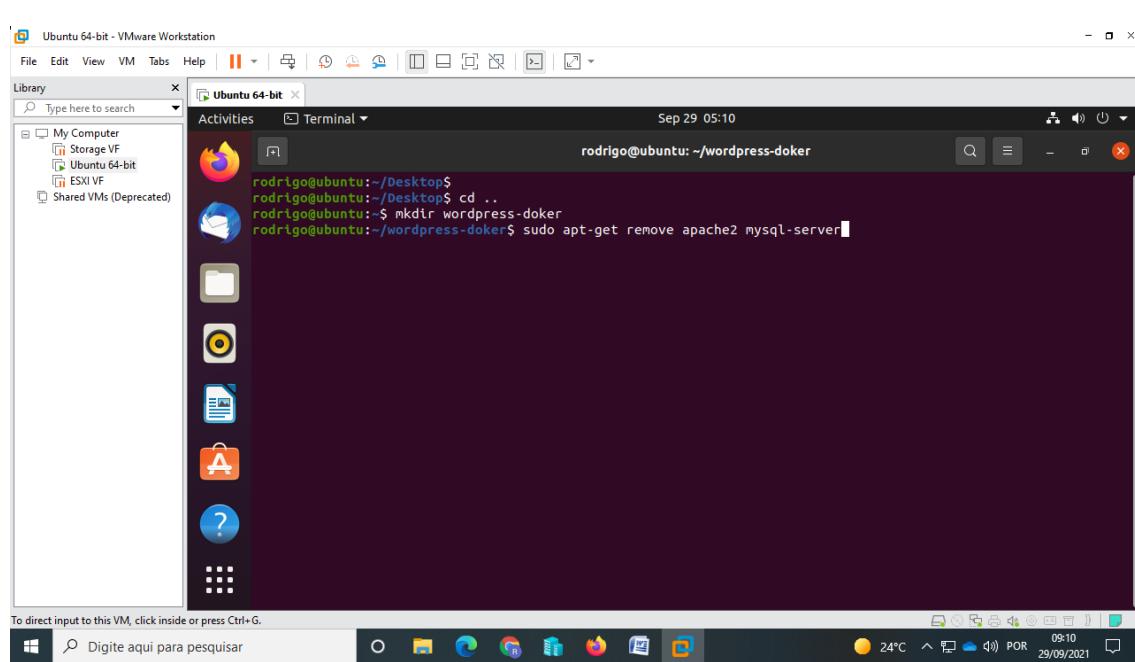
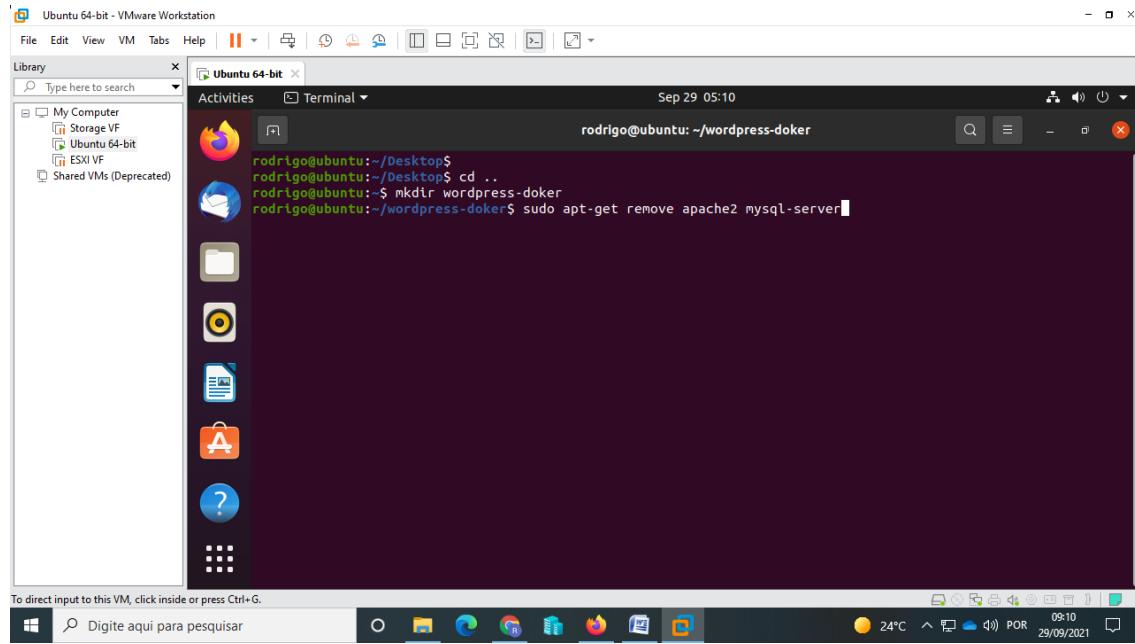
Para criação do Containers wordpress precisaremos criar uma estrutura própria e por isso o primeiro passo é cria uma “pasta” para a nova estrutura.

Para isso utilizamos o comando “mkdir wordpress-doker”.



2º PASSO.

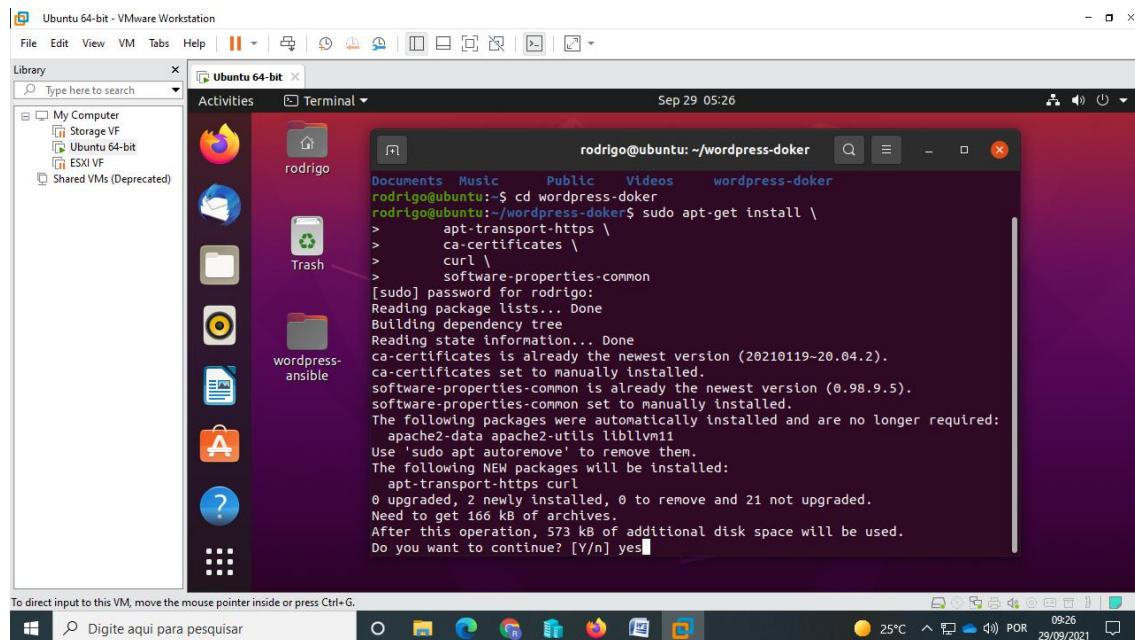
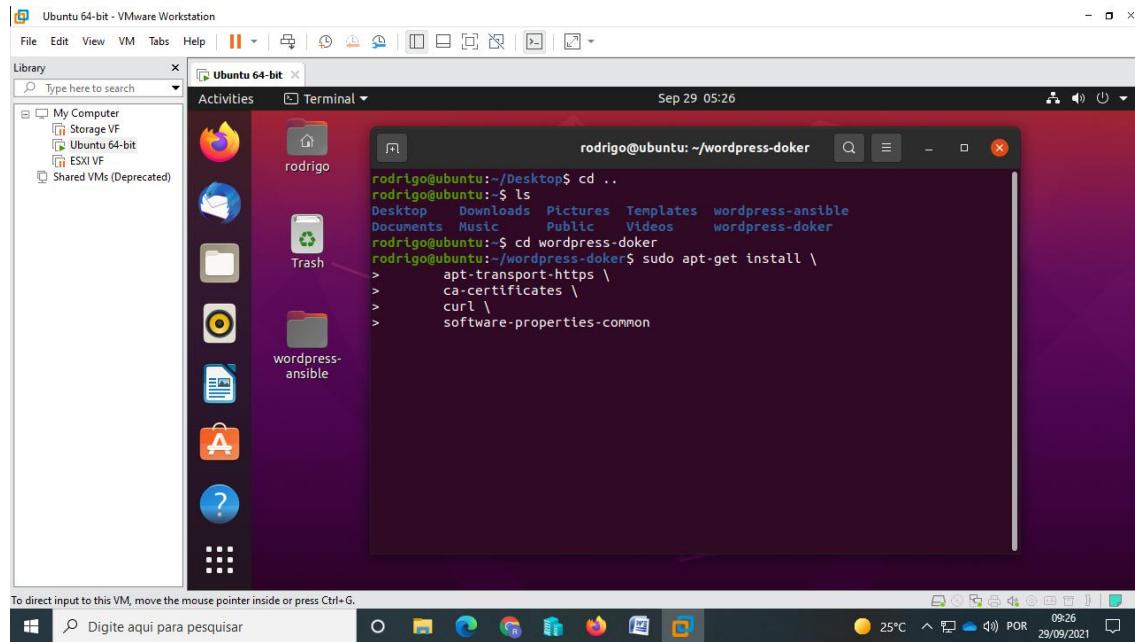
Como já temos instalado no Ansible o mysql, apache e etc, precisamos agora remover essa estrutura que criamos anteriormente, isso se faz necessário para que não tenhamos conflito de portas. Então utilizamos o comando “sudo apt-get remove apache2 mysql-server”.



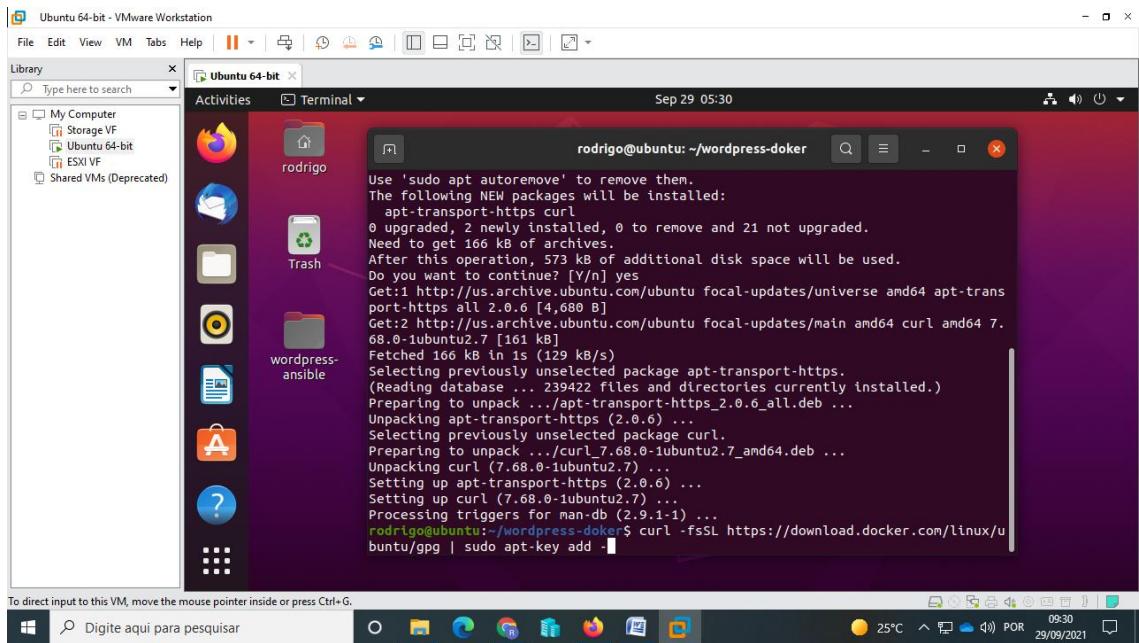
3º PASSO.

Agora começaremos a preparar o servidor para ser um servidor Doker. O próximo comando tem como objetivo instalar vários pacotes de suporte. Para isso utilizamos o comando

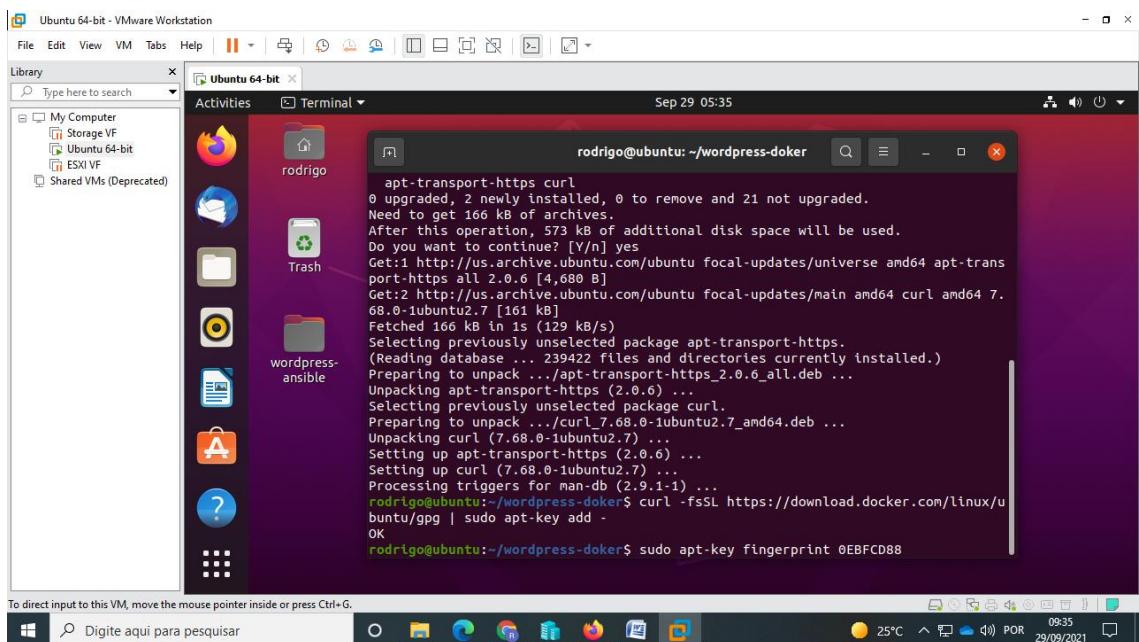
```
sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common
```

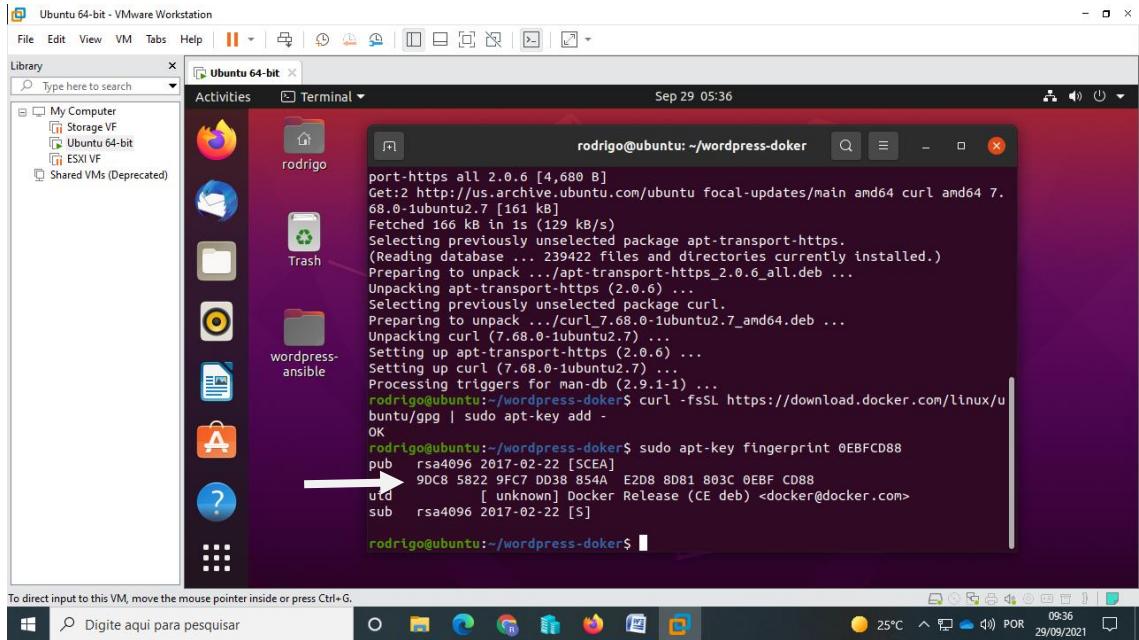


O próximo comando tem como objetivo instalar pacotes de suporte necessários para as funções de assinatura digital e download para outros pacotes que serão necessários para o nosso projeto. Este comando consta o endereço do docker na web e lá possui todos os pacotes necessários para poder rodar a aplicação, nesse caso utilizaremos o comando `curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -`



O próximo passo é gerar um “fingerprint”, é uma “impressão digital” que permite conectar com o ambiente. Para isso usamos o seguinte comando `sudo apt-key fingerprint 0EBFCD88`.

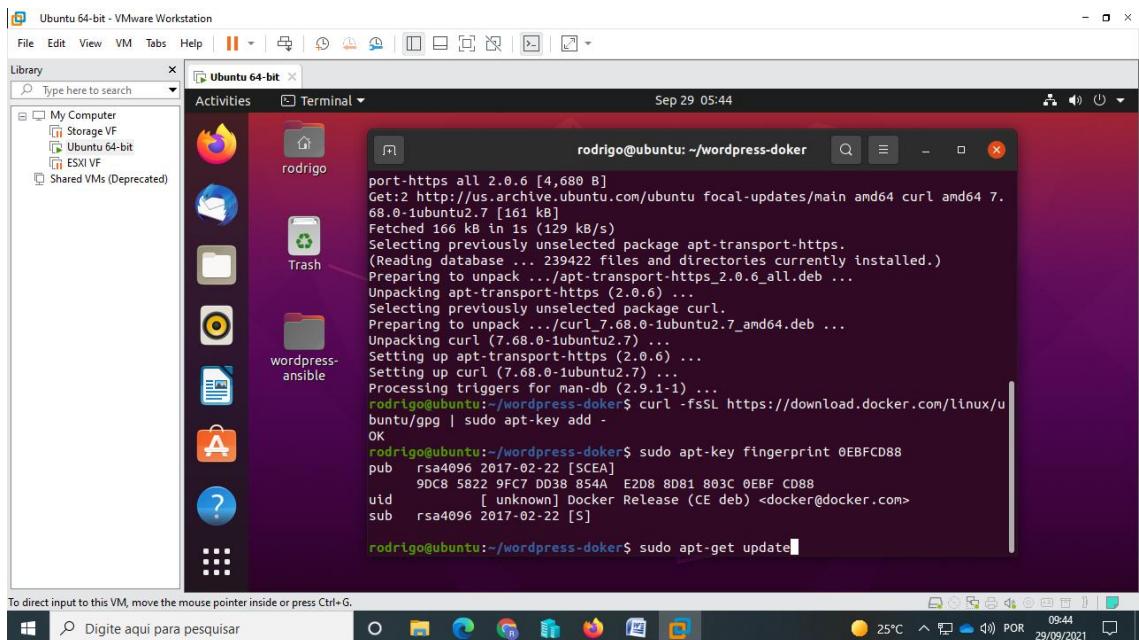


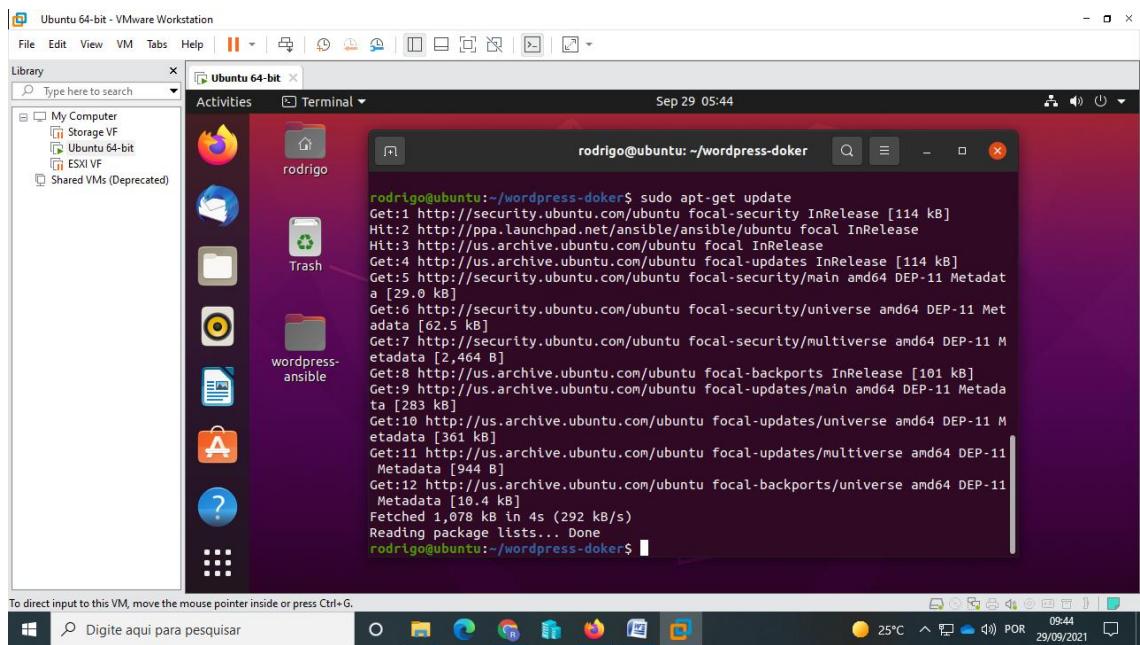


Podemos observar nessa tela a chave criada. Agora, todas as vezes que precisarmos de pegar uma imagem no repositório, não será necessário ficar autenticando.

4º PASSO.

Nesse passo iremos baixar alguns pacotes do ambiente do docker, mas antes, por precaução, iremos fazer um update no get antes de iniciar o próximo passo. Esse passo tem como objetivo acelerar o download dos pacotes do Docker, não sendo necessário baixar o que já temos no repositório.





5º PASSO.

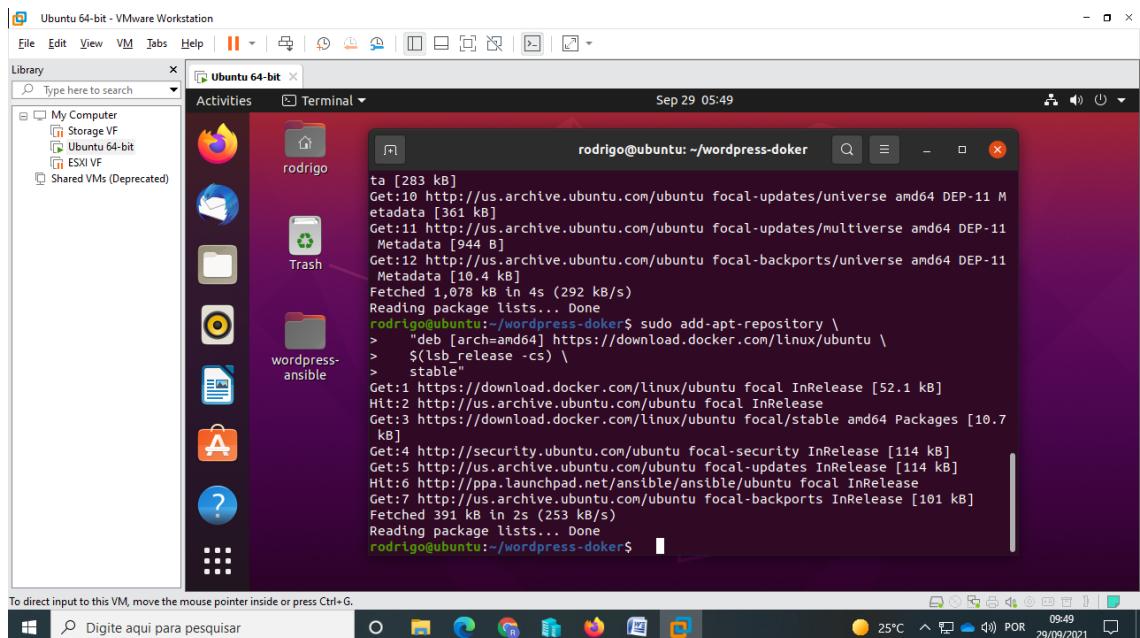
Agora precisamos adicionar ao repositório local os pacotes, nesse caso utilizamos o comando

```
sudo add-apt-repository \
```

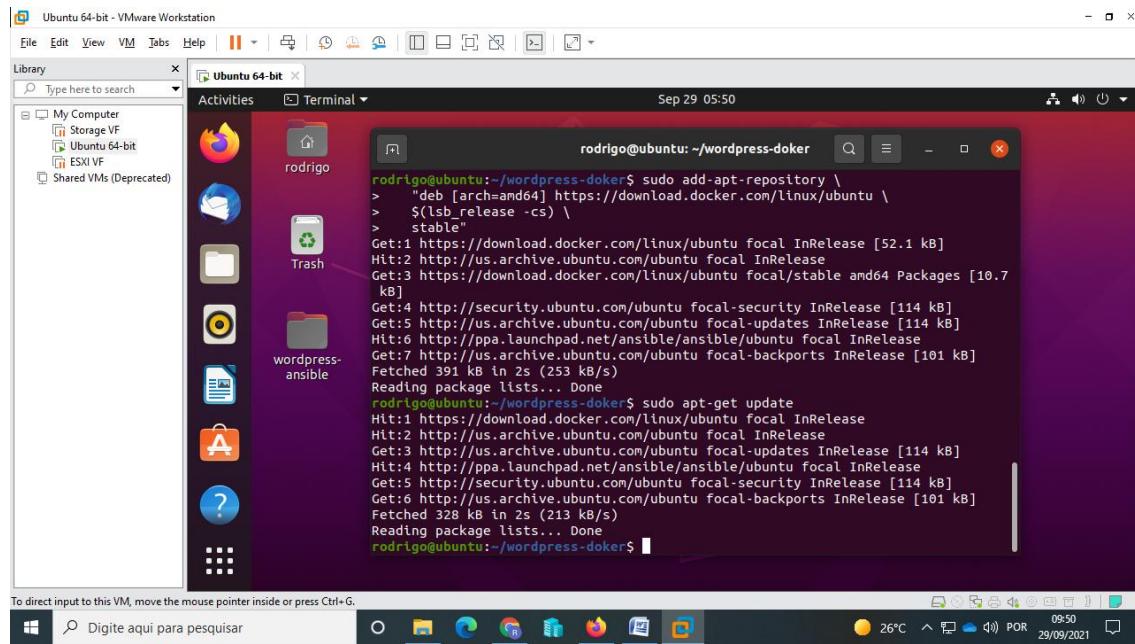
```
"deb [arch=amd64] https://download.docker.com/linux/ubuntu \
```

```
$(lsb_release -cs) \
```

```
stable"
```

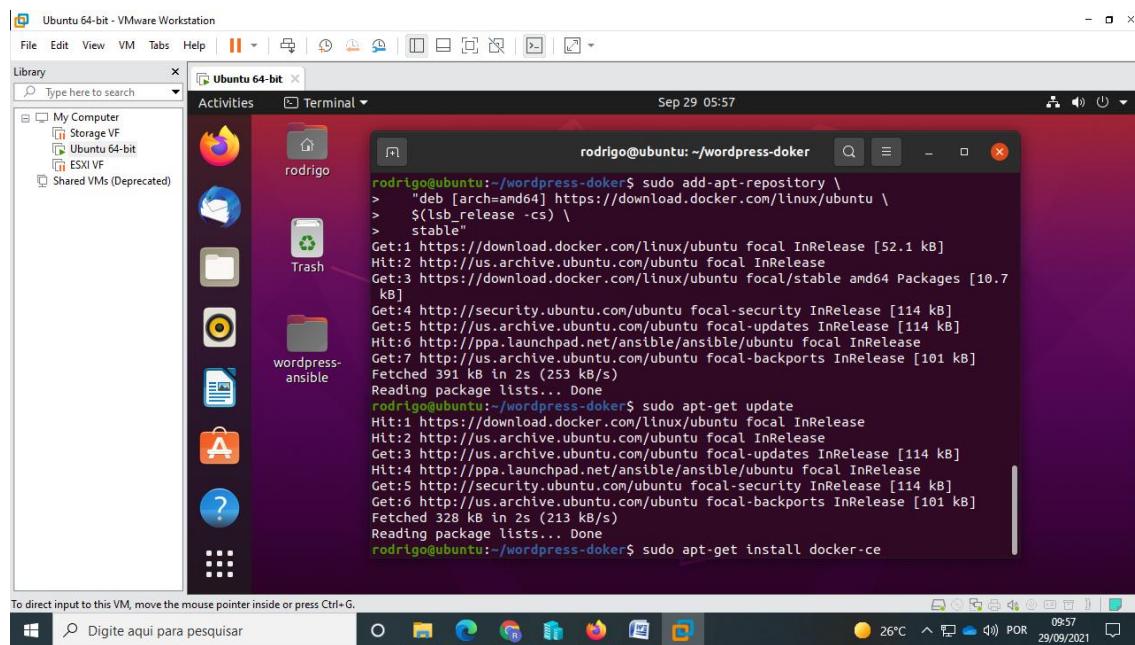


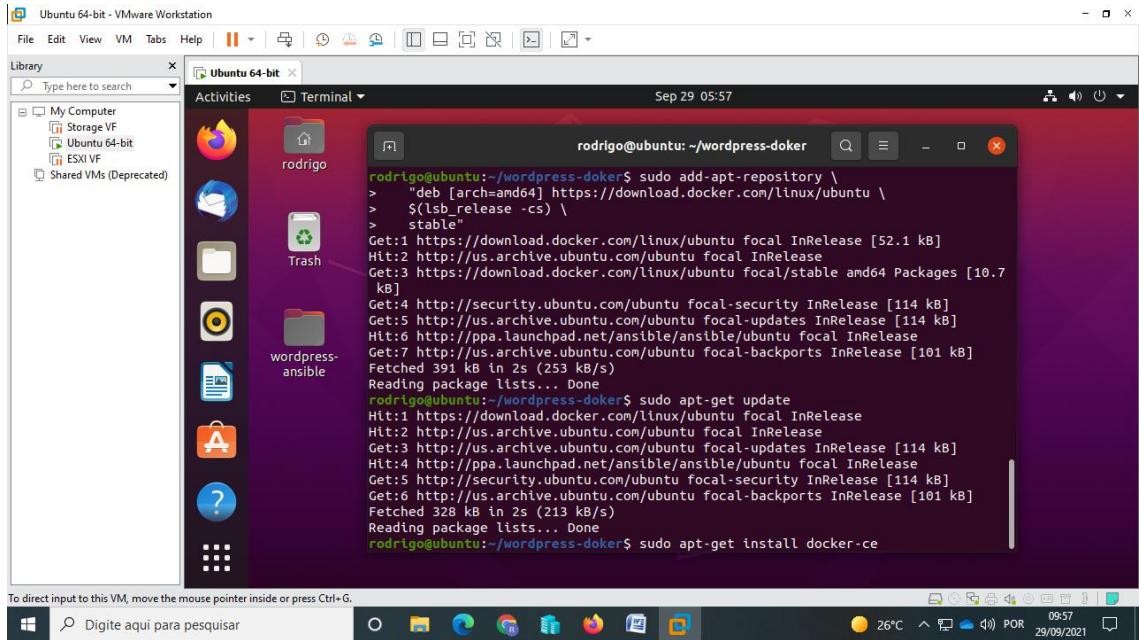
Agora basta fazer o update novamente.



6º PASSO

Nessa etapa vamos fazer a instalação do docker. Isso permitirá a execução de containers MySQL e WordPress. O comando utilizado é `sudo apt-get install docker-ce`

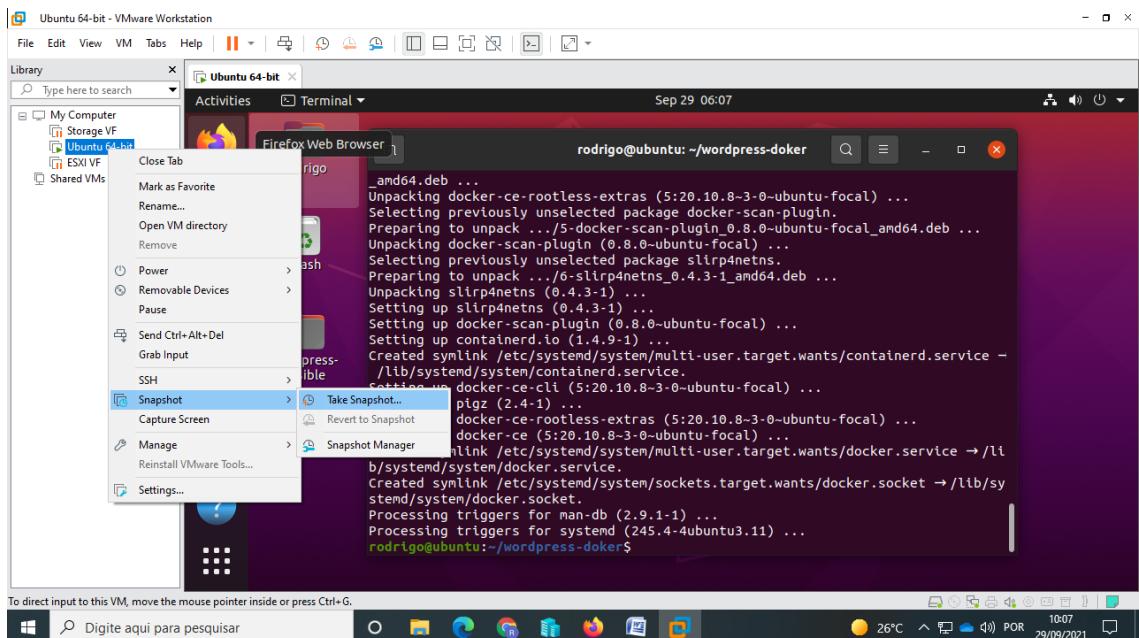




Com isso agora é possível criar containers dentro do Linux.

7º PASSO.

Agora que já temos a estrutura que nos permite criar containers vamos instalar o MySQL e Wordpress, só que também faremos a mesma instalação utilizando o Playbook. Para não termos problema de instalação e ter um procedimento mais rápido, essa etapa iremos criar um Snapshot, finalizando assim o TP04 conforme orientado em aula.

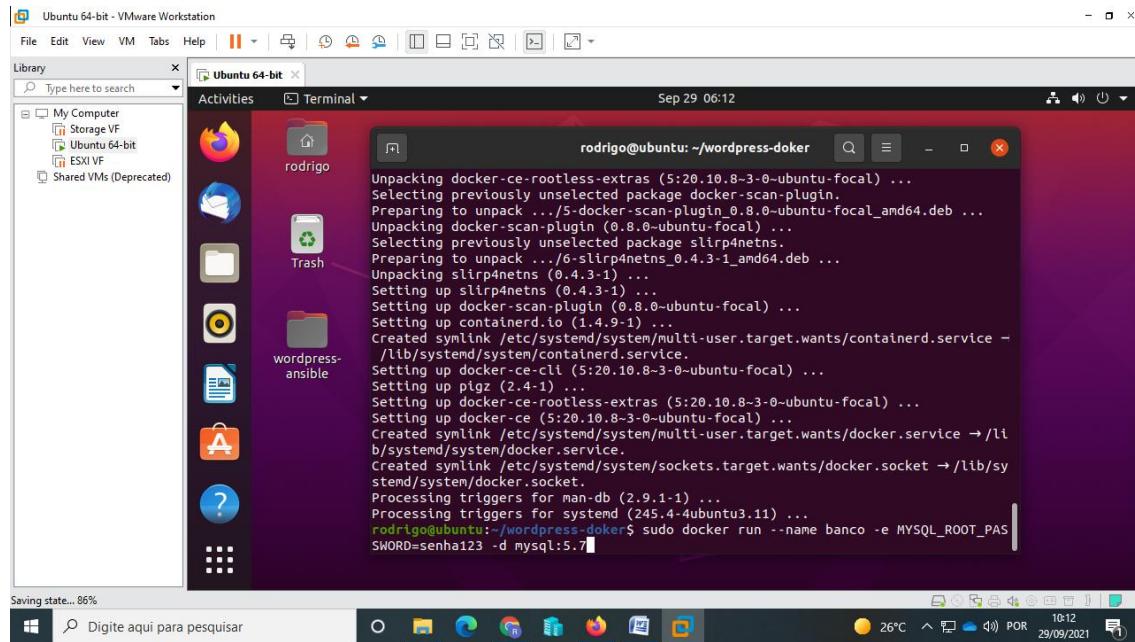


Agora é só dar um nome para essa cena, nesse caso chamei de “Ubuntu com Docker”.

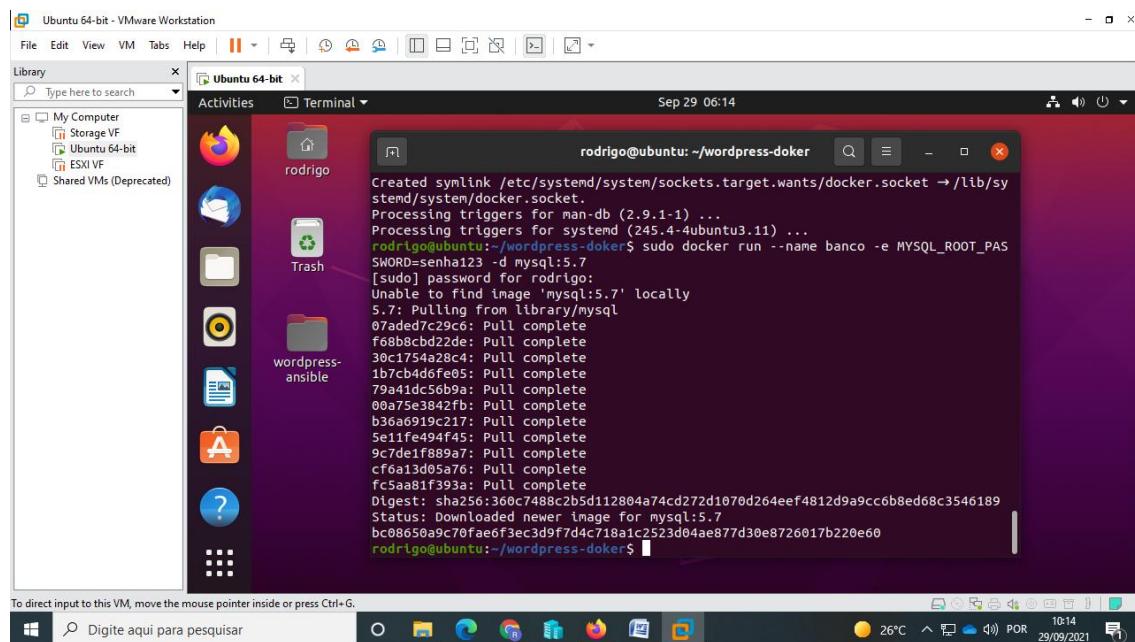
8º PASSO.

Agora sim vamos criar o container com Wordpress e MySQL.

Vamos então primeiro baixar o MySQL com o comando `sudo docker run --name banco -e MYSQL_ROOT_PASSWORD=senha123 -d mysql:5.7`

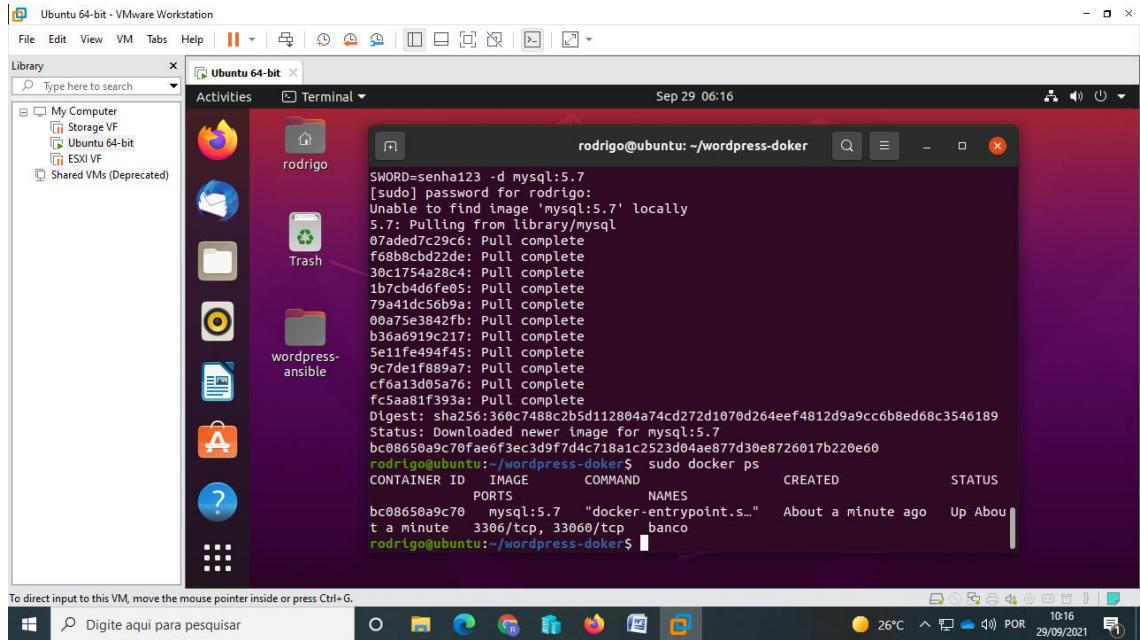


```
rodrigo@ubuntu: ~/wordpress-doker
Unpacking docker-ce-rootless-extras (5:20.10.8-3-0-ubuntu-focal) ...
Selecting previously unselected package docker-scan-plugin.
Preparing to unpack .../5-docker-scan-plugin_0.8.0-ubuntu-focal_amd64.deb ...
Unpacking docker-scan-plugin (0.8.0-ubuntu-focal) ...
Selecting previously unselected package slirp4netns ...
Preparing to unpack .../6-slirp4netns_0.4.3-1_amd64.deb ...
Unpacking slirp4netns (0.4.3-1) ...
Setting up slirp4netns (0.4.3-1) ...
Setting up docker-scan-plugin (0.8.0-ubuntu-focal) ...
Setting up containedr.io (1.4.9-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up docker-ce-cll (5:20.10.8-3-0-ubuntu-focal) ...
Setting up pigz (2.4-1) ...
Setting up docker-ce-rootless-extras (5:20.10.8-3-0-ubuntu-focal) ...
Setting up docker-ce (5:20.10.8-3-0-ubuntu-focal) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for systemd (245.4-4ubuntu3.11) ...
rodrigo@ubuntu:~/wordpress-doker$ sudo docker run --name banco -e MYSQL_ROOT_PASSWORD=senha123 -d mysql:5.7
```



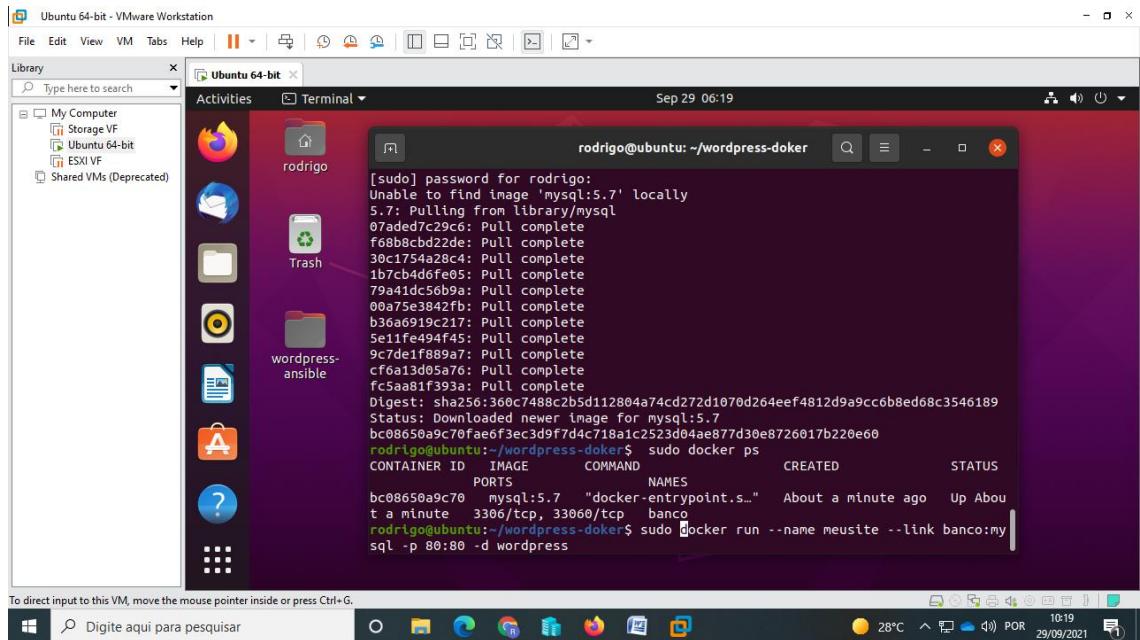
```
rodrigo@ubuntu: ~/wordpress-doker
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for systemd (245.4-4ubuntu3.11) ...
rodrigo@ubuntu:~/wordpress-doker$ sudo docker run --name banco -e MYSQL_ROOT_PASSWORD=senha123 -d mysql:5.7
[sudo] password for rodrigo:
Unable to find image 'mysql:5.7' locally
5.7: Pulling from library/mysql
07aded7c29c6: Pull complete
f68bbcbdd2de: Pull complete
30c1754a284c: Pull complete
1b7cb4d6fe05: Pull complete
79a41dc56b9a: Pull complete
00a75e3842fb: Pull complete
b36a6919c217: Pull complete
5e11fe494f45: Pull complete
9c7de1f889a7: Pull complete
cf6a13d05a76: Pull complete
fc5aa81f393a: Pull complete
Digest: sha256:360c7488c2b5d112804a74cd272d1070d264eef4812d9a9cc6b8ed68c3546189
Status: Downloaded newer image for mysql:5.7
bc08650a9c70fae0f3ec3d9fd74c718a1c2523d04ae877d30e8726017b220e60
rodrigo@ubuntu:~/wordpress-doker$
```

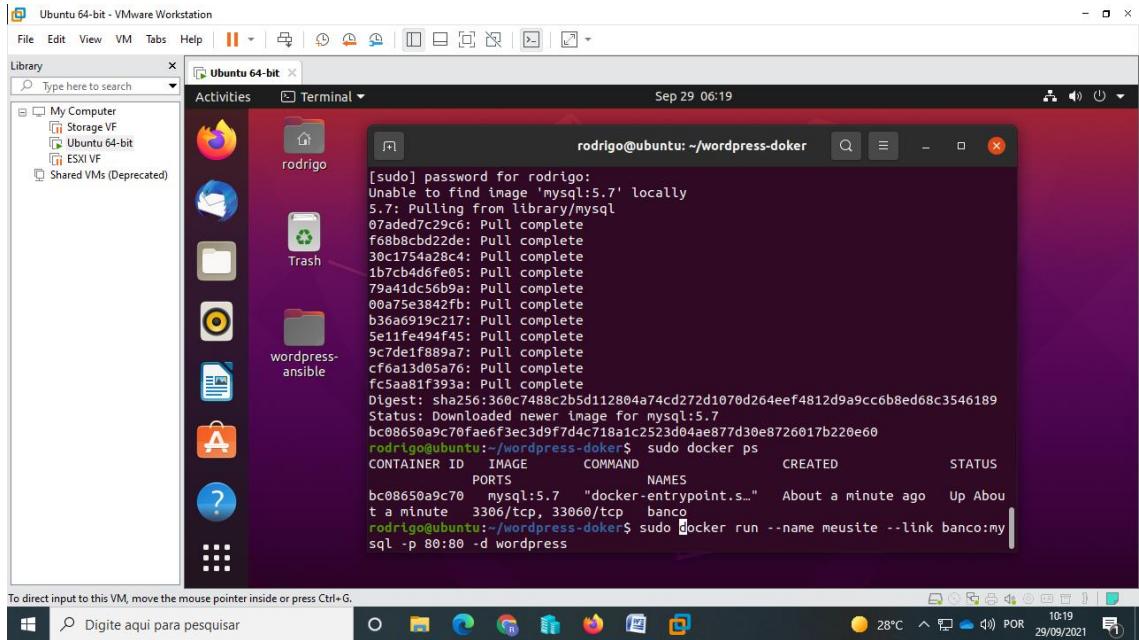
Agora precisamos de saber se o container está em execução, para isso usamos o comando `sudo docker ps`



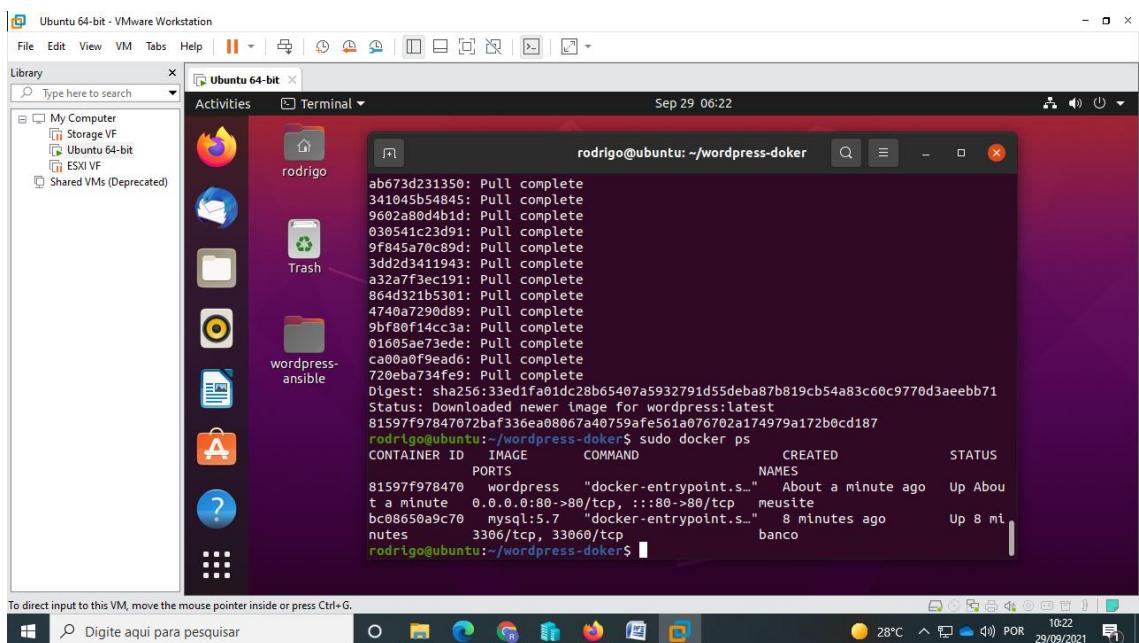
Podemos verificar que ele foi criado corretamente. Agora vamos criar o segundo container vinculado a esse, que é o wordpress, usando o comando `docker run --name meusite --link banco:mysql -p 80:80 -d wordpress`

Esse comando faz o link com o MySQL.



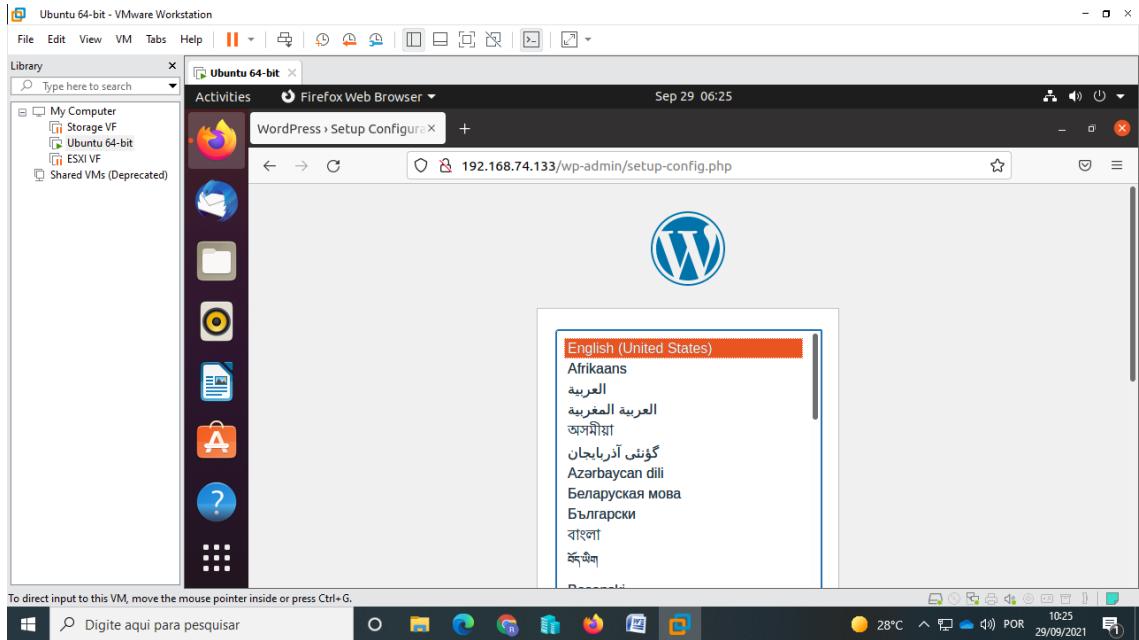


Agora podemos usar o comando e ver que já temos dois containers criados



Agora podemos verificar que já temos o Wordpress disponível.

Para verificar se já está em funcionamento, basta digitar o ip do maquina no navegador web.

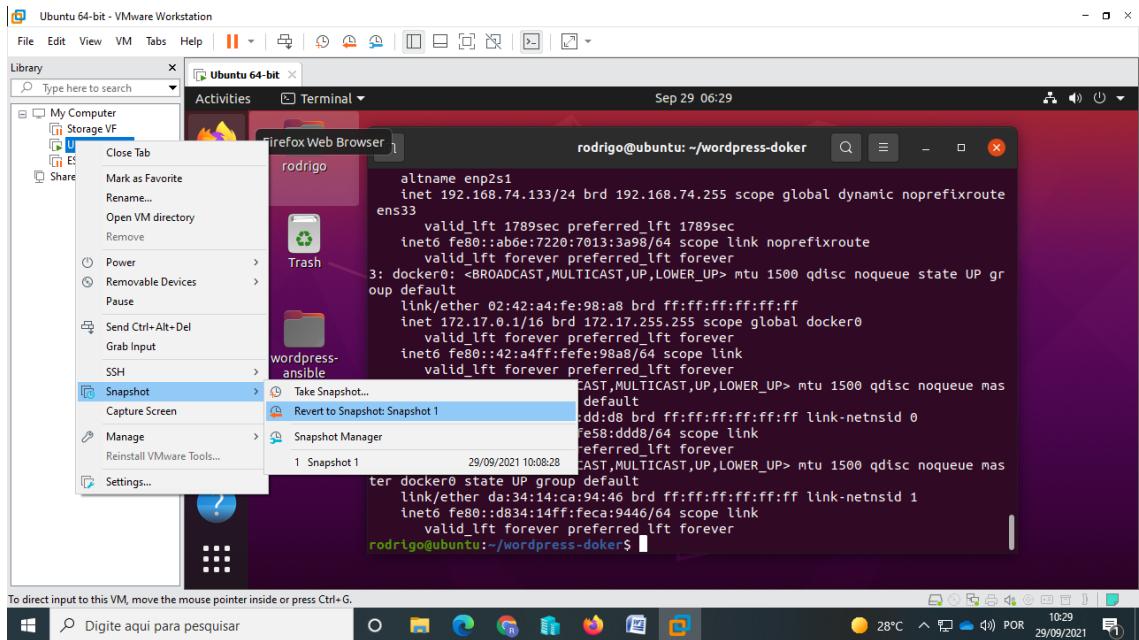


Como podemos verificar, foi instalado corretamente todos os dockers que instalamos.

9º PASSO.

Agora é fazer novamente tudo que foi feito, só que no Ansible. Mas antes precisamos retonar no snapshot criado.

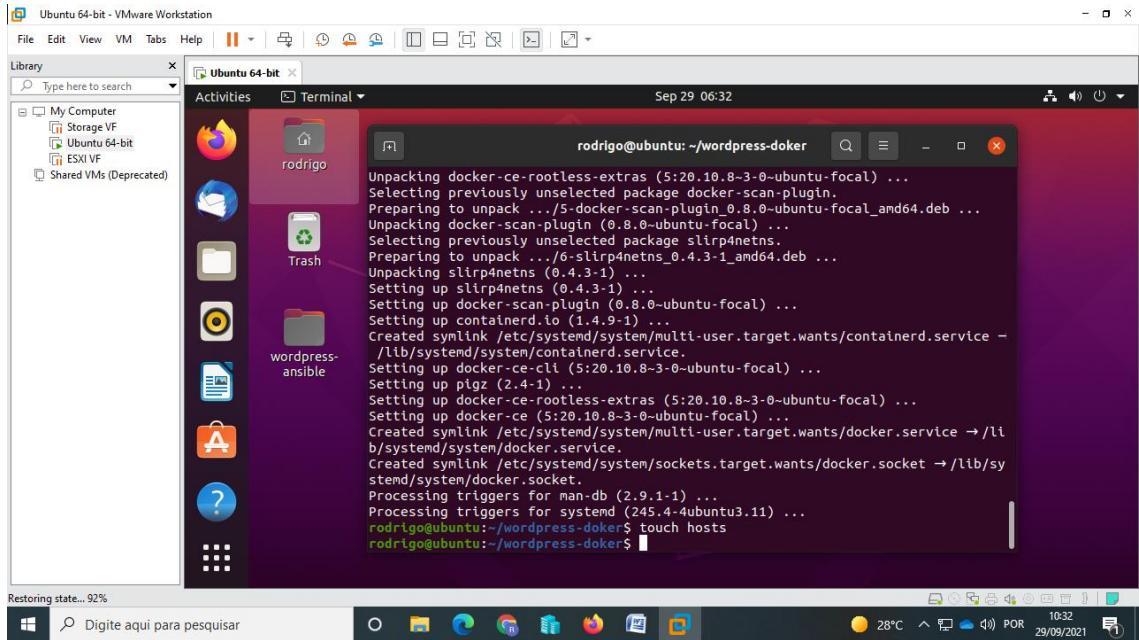
Para isso basta acessar a aba lateral da VM e reverter o Snapshot.



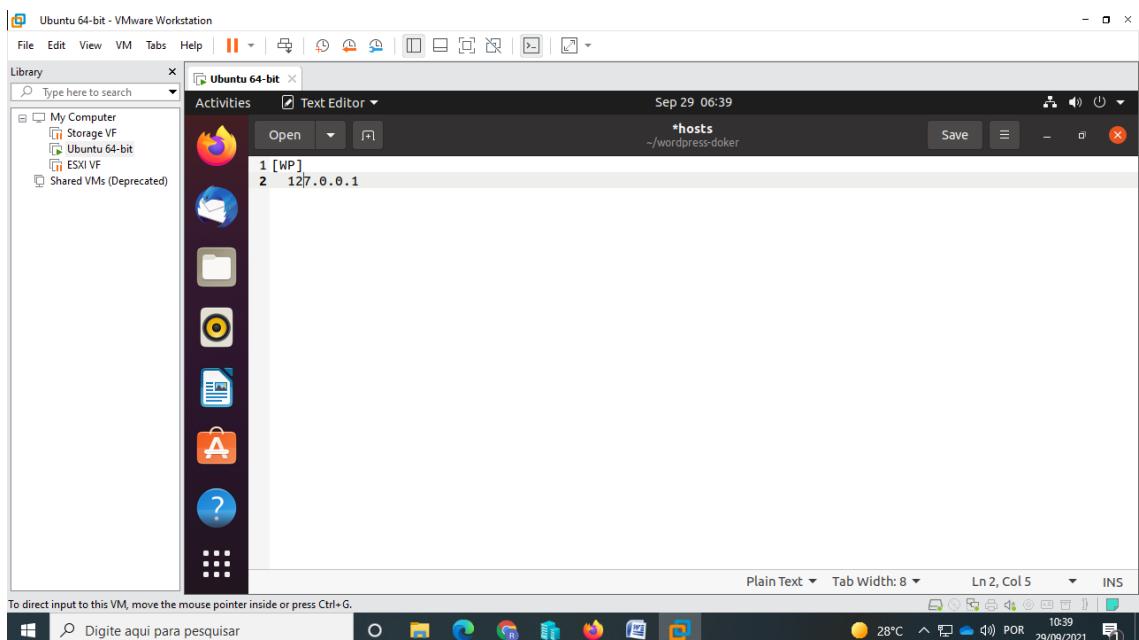
10º PASSO.

Agora vamos criar dentro da pasta Wordpress-docker um novo Playbook.

A primeira coisa a se fazer é criar um arquivo host.



E agora é configurar o local host com a função regit hosts. Nesse caso dei o nome de WP e coloquei o Ip host. Lembrando sempre que aqui seria a lista dos servidores que faríamos a instalação do wordpress no mundo real.



Agora faremos a mesma coisa no palybook, utilizando os comandos touch playbook.yml e regit playbook.yml e posteriormente montaremos nosso playbook.

```

1 ---
2 - hosts: WP
3   remote_user: rodrigo
4   become: yes
5   tasks:
6     - name: "Executa o container MySQL"
7       docker_container:
8         name: banco
9         image: mysql:5.7
10        env:
11          MYSQL_ROOT_PASSWORD: senha123
12     - name: "Executa o container WordPress"
13       docker_container:
14         name: wordpress
15         image: wordpress
16         links:
17           - banco:mysql
18         ports:
19           - "80:80"

```

Agora, precisamos instalar dois componentes de docker para ansible que não estão automaticamente instalado. Para isso precisamos usar o comando “`sudo apt install python3-pip`”

```

rodrigo@ubuntu:~/wordpress-doker$ gedit hosts
rodrigo@ubuntu:~/wordpress-doker$ gedit hosts
rodrigo@ubuntu:~/wordpress-doker$ touch palybook.yml
rodrigo@ubuntu:~/wordpress-doker$ redit palybook.yml

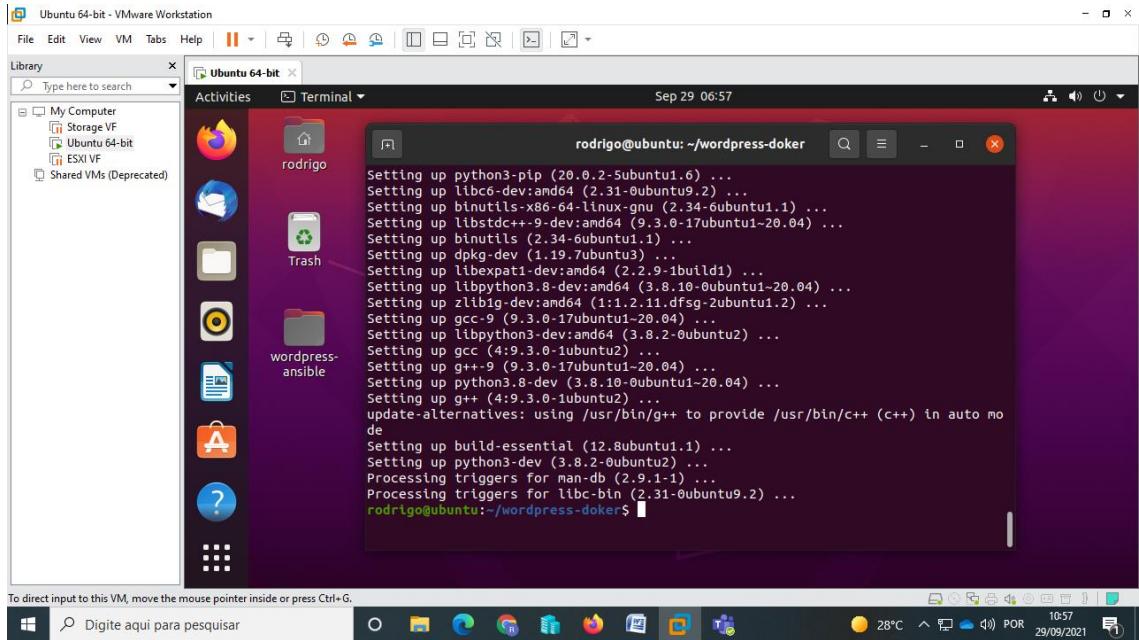
Command 'redit' not found, did you mean:

  command 'gedit' from snap gedit (3.36.2+git14.c07b37c55)
  command 'ledit' from deb ledit (2.04~4build1)
  command 'redet' from deb redet (8.26-1.3)
  command 'xedit' from deb x11-apps (7.7+8)
  command 'gedit' from deb gedit (3.36.2-0ubuntu1)
  command 'nedit' from deb nedit (1:5.7-2)
  command 'redir' from deb redit (3.2-1)
  command 'jedit' from deb jedit (5.5.0+dfsg-1)
  command 'edit' from deb mime-support (3.64ubuntu1)

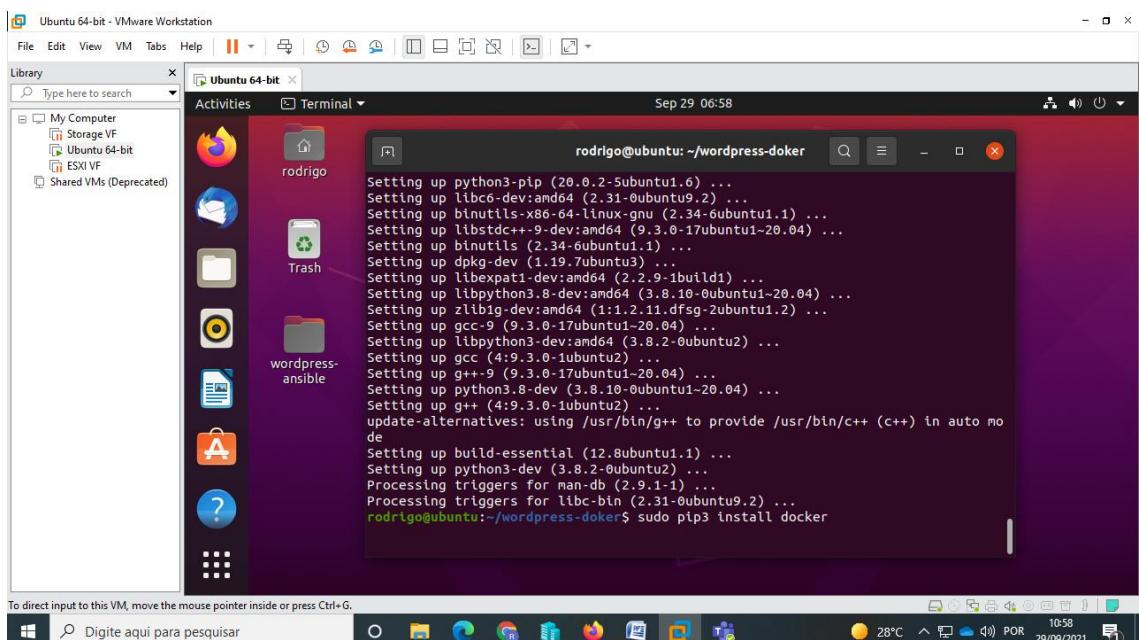
See 'snap info <snapname>' for additional versions.

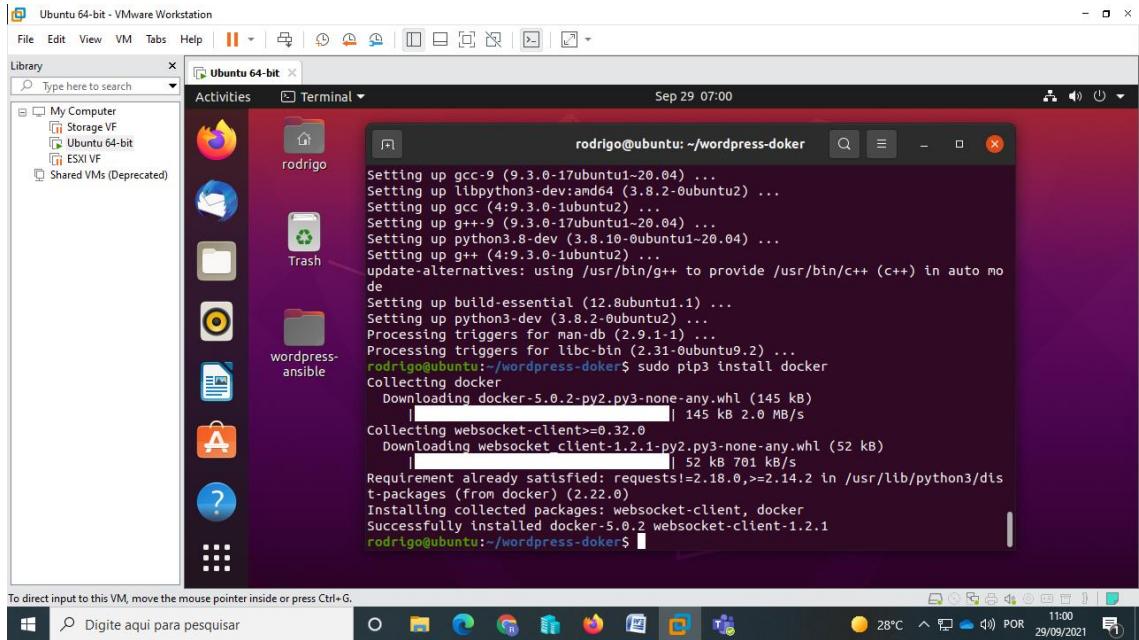
rodrigo@ubuntu:~/wordpress-doker$ gedit palybook.yml
rodrigo@ubuntu:~/wordpress-doker$ sudo apt install python3-pip

```

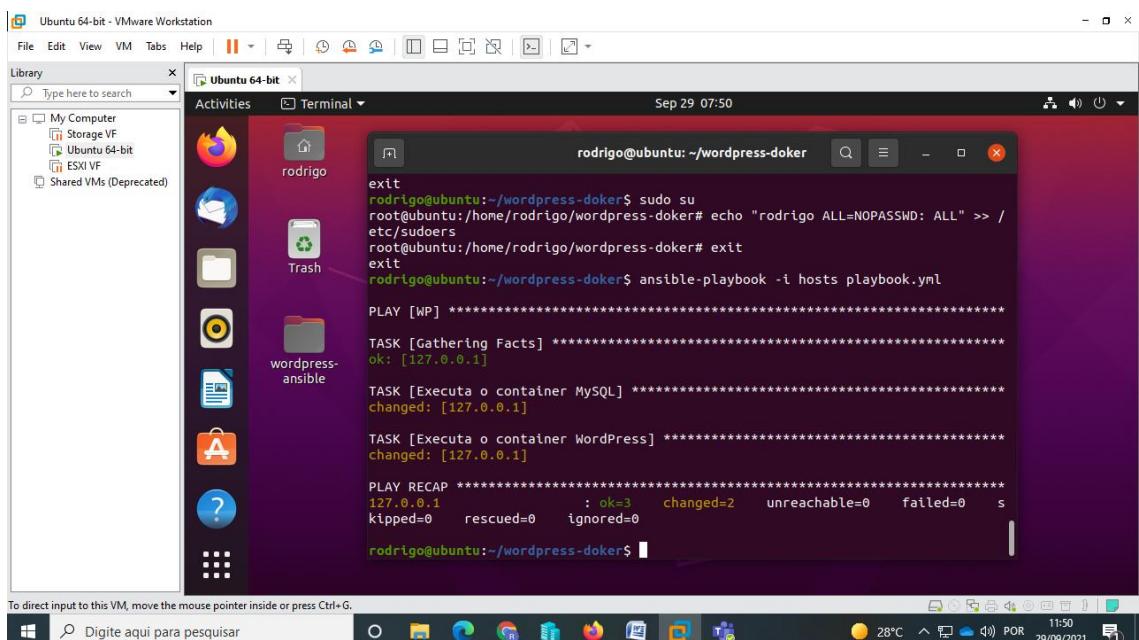


O Segundo comando é “sudo pip3 install docker”.

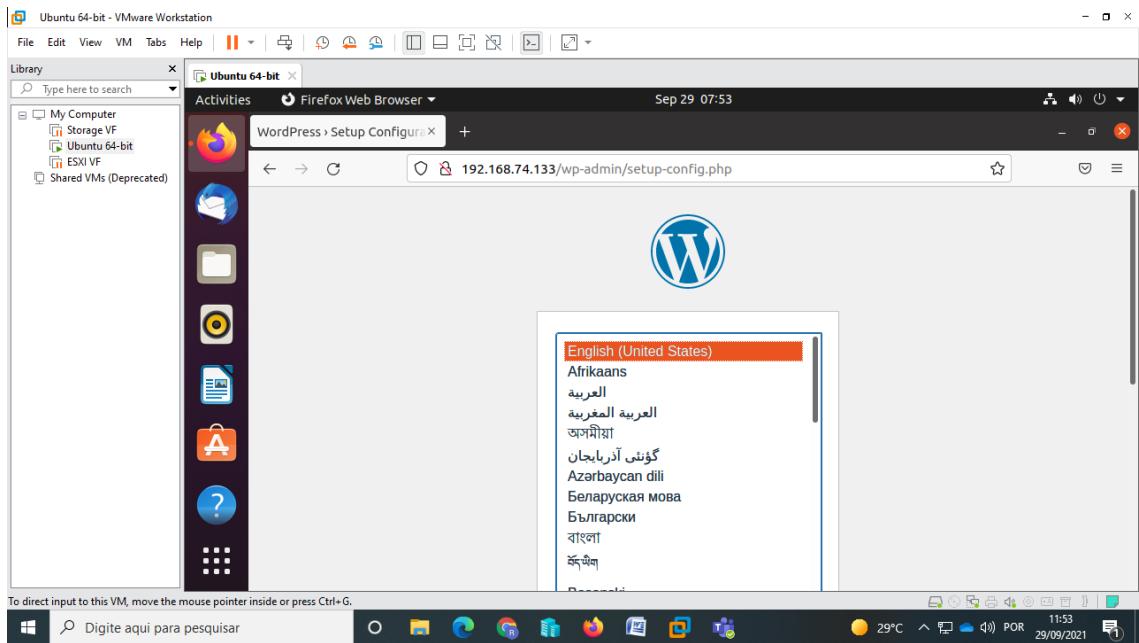




Agora vamos rodar o playbook para conferir seu tudo deu certo.



Pronto, agora está funcionando nosso playbook, mas para ter certeza basta entrar no ip local.



Para finalizar nosso trabalho, vamos disponibilizar o endereço para acesso ao repositório no github do docker criado.

https://github.com/rodrigooliveiramenezes/pb_docker

3.3 Procedimento de Criação de Repositório Público Utilizando GitHub.

Na no capítulo 1, item 2, na página 05 deste projeto (1.2 Atualizações, Correções de Bugs e requisição de Desenvolvimeto) explicamos como as atualizações, correções de bugs e requisição de Desenvolvimento serão feitos, agora demonstraremos como criar um repositório público no GitHub.

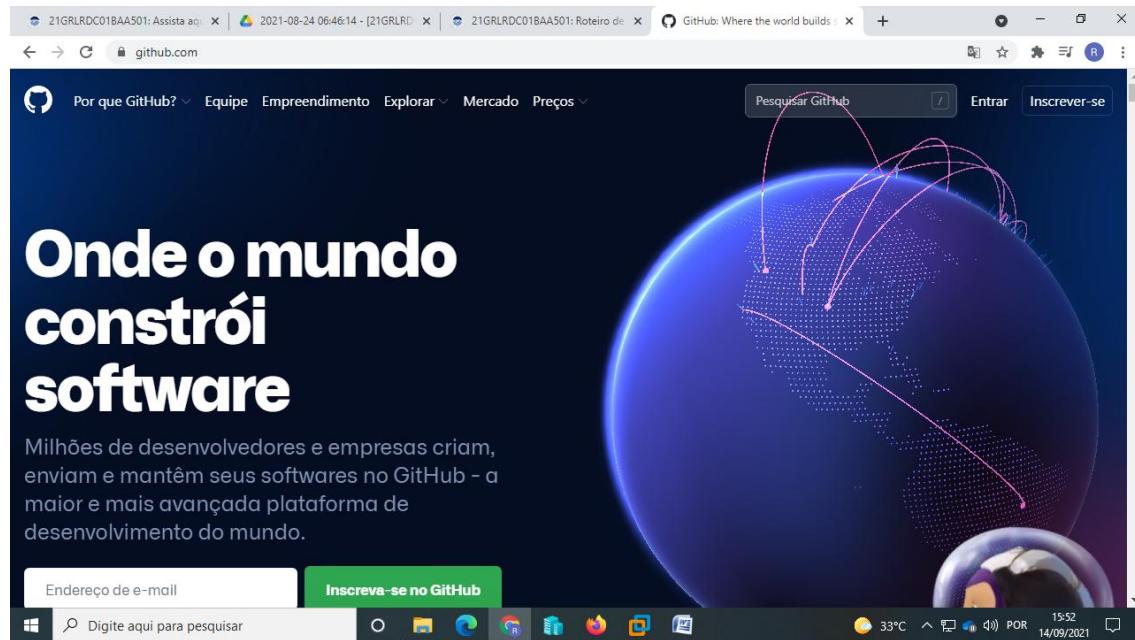
Inicialmente precisamos saber o que é um Git. Git é uma ferramenta de controle de versão de códigos, ou seja, ele é útil para gerenciar as versões de um código, principalmente quando o trabalho é desenvolvido em equipe, como a maioria dos cenários de trabalho hoje em dia. Dessa forma, o Git permite que se trabalhe por duas ou mais pessoas diferentes sem que um “danifique” o que já foi feito pelo outro profissional.

O GitHub é a mesma coisa que o Git, só que com possibilidade de deixar ele público na internet, ou seja, é uma plataforma onde o código ficará armazenado publicamente o código para que possa ser compartilhado entre as pessoas que fazem parte do projeto.

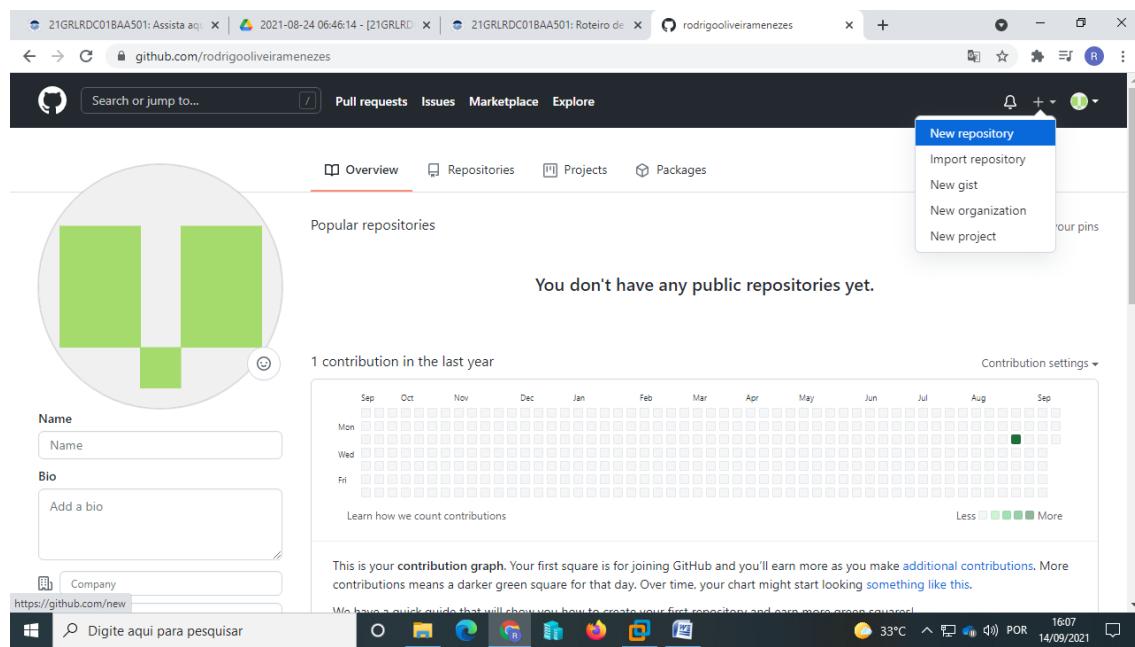
Nessa etapa então iremos criar um repositório público no GitHub, vamos lá.

Primeiro Passo.

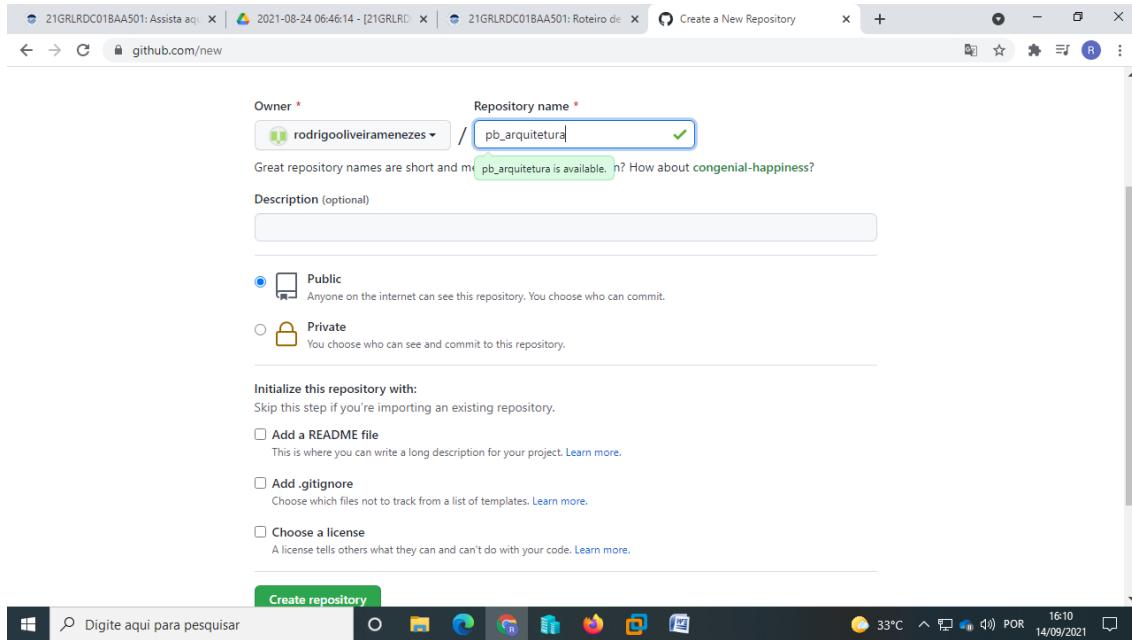
Acesse seu servidor web, digite “github.com” e crie sua conta.



Após criar a conta iremos fazer o primeiro repositório, para tanto, basta ir no canto superior direto da página e clicar no símbolo de + e depois clicar em “new repository”.



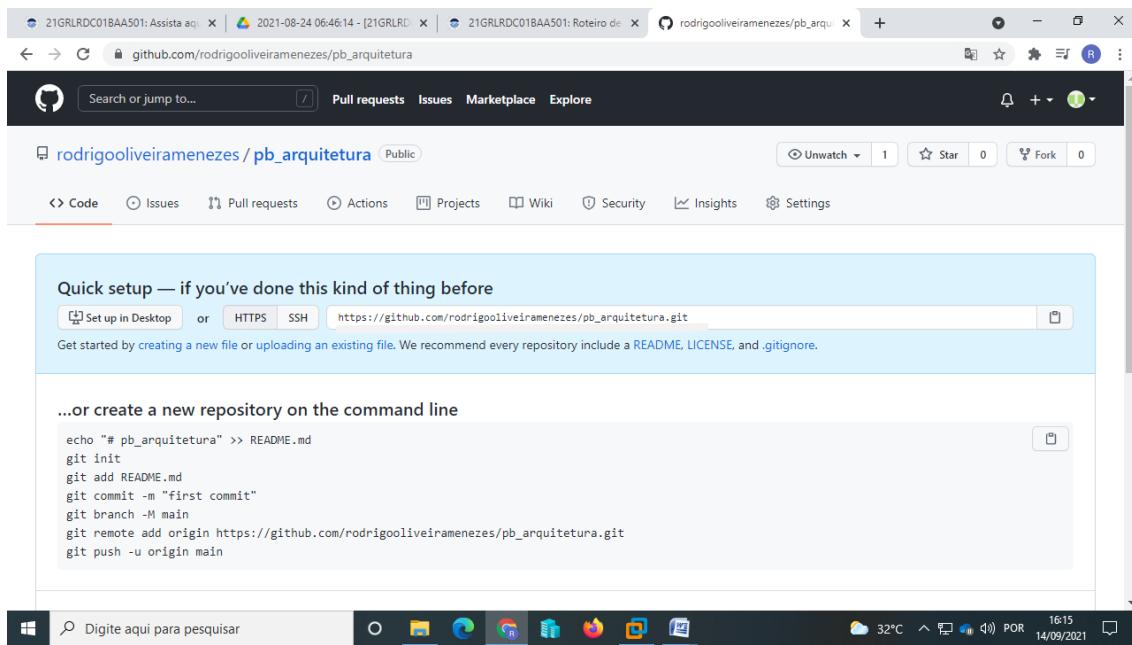
Na próxima tela de um nome ao seu repositório, nesse caso nomeie como “pb_arquitetura”, e o mais importante, a caixa “public” deve ser marcada para que ansible posso pegar o nosso código.



OBS: Se preferir pode colocar uma descrição para o repositório que está sendo criado.

Em seguida clik em “Create repository” e a tela abaixo aparecerá.

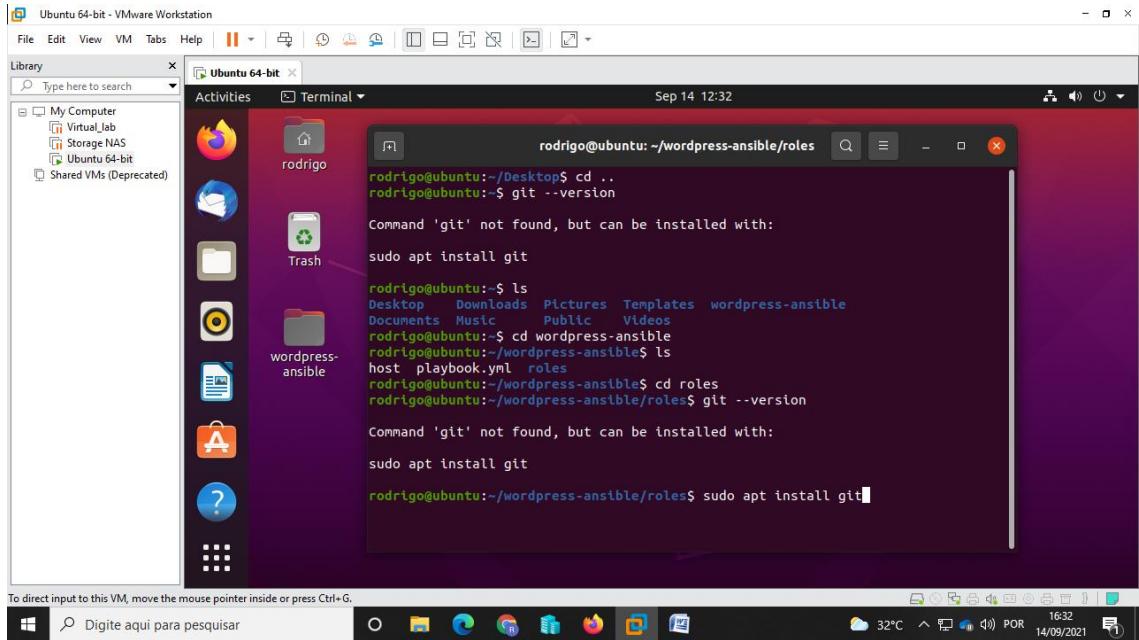
Nessa tela temos o endereço que passaremos para o Ansible, que nesse caso é o https://github.com/rodrigo oliveiramenezes/pb_arquitetura.git



Passo Dois: Instalando o Git no Ansible.

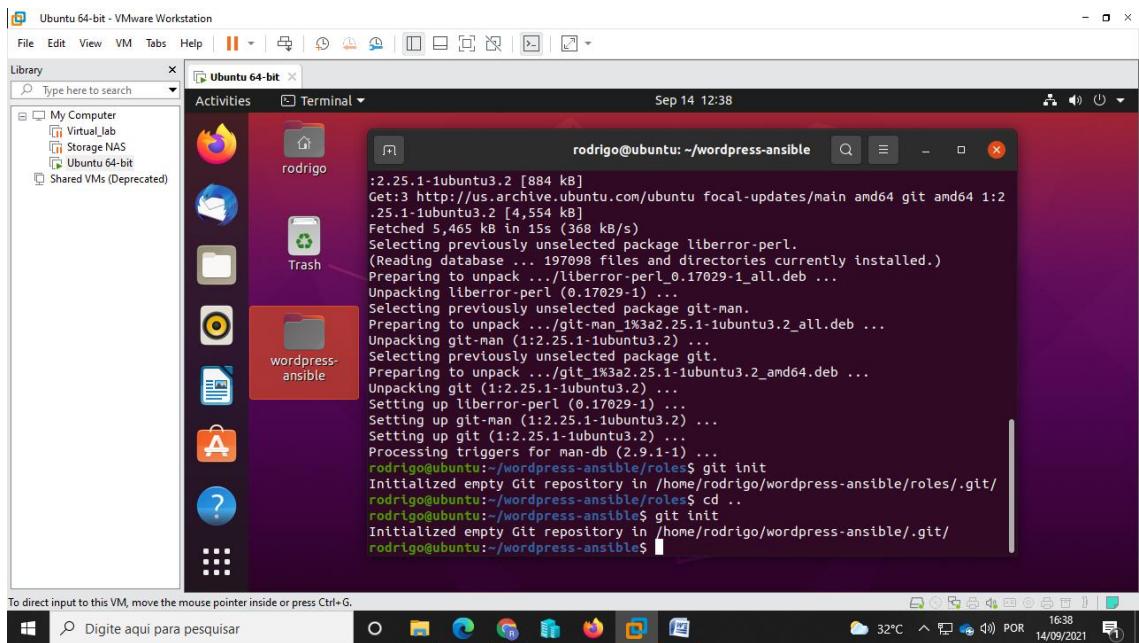
Agora que a conta foi criada e que temos o endereço do repositório, precisamos instalar o git no ansible.

Então entramos no ansible e digitamos “sudo apt install git” e teclamos o Enter.



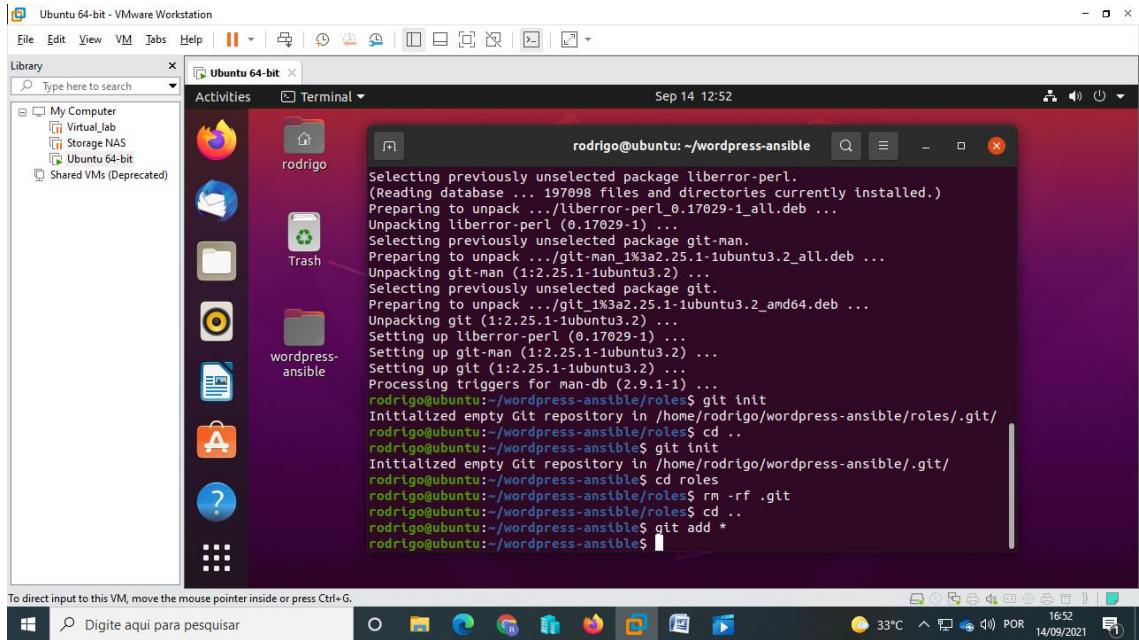
Passo três: Criando a estrutura

Depois de instalada digitamos “git init” para iniciar o Git e criamos um repositório vazio chamado “.git”.



A próxima etapa é colocar toda a estrutura do wordpres-ansible, Host, Playbook e a pasta Roles, dentro da git que foi criada, para, posteriormente, ser exportado para o github.

Para pode “copiar” toda a estrutura, basta digitar o comando git add *.



Passo quatro: Criando o primeiro Commit

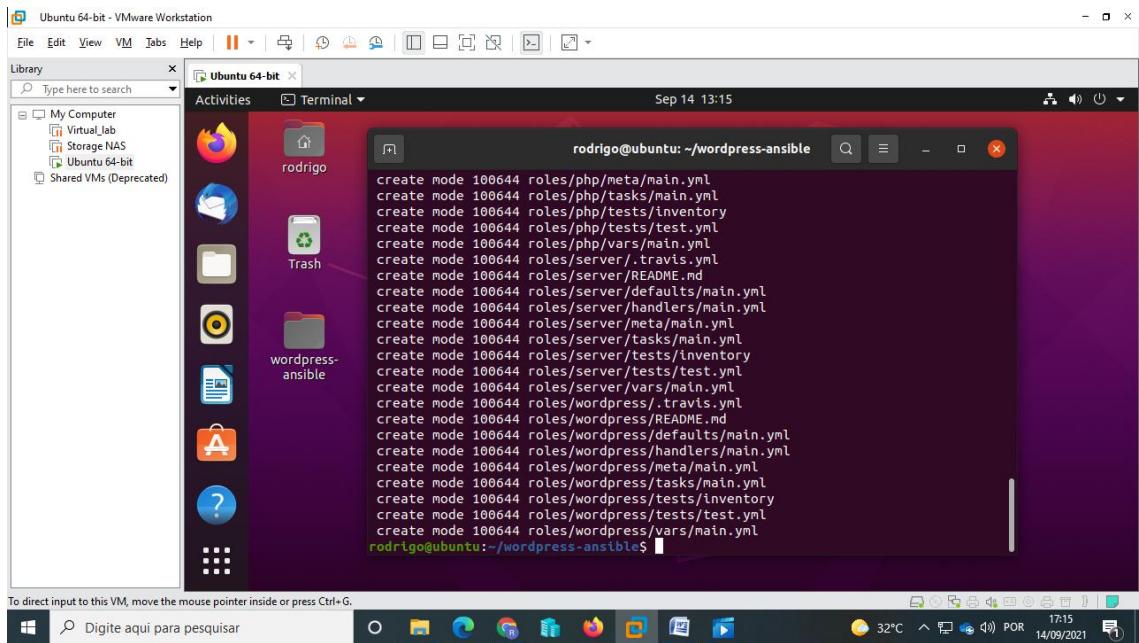
Para criar o primeiro Commit a nova versão do Ansible exige e-mail e o usuário da conta do GitHub onde será criado o Commit. Para realizar a tarefa usamos o comando:

```
git config --global user.email "digite o e-mail"
```

```
git config --global user.name "usuario"
```

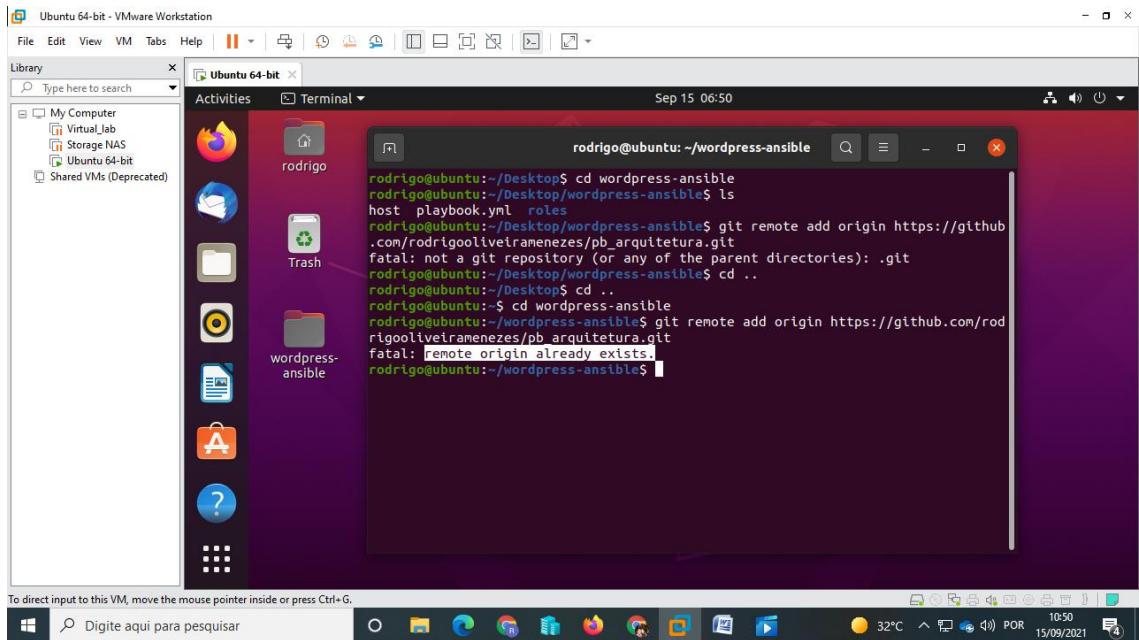
Agora damos o comando para pode fazer o Commit que é:

```
git commit -m "importando playbook WordPress"
```



O próximo passo é apontar para o repositório. Nessa etapa é necessário utilizar o link gerado no GitHub ao montar o repositório.

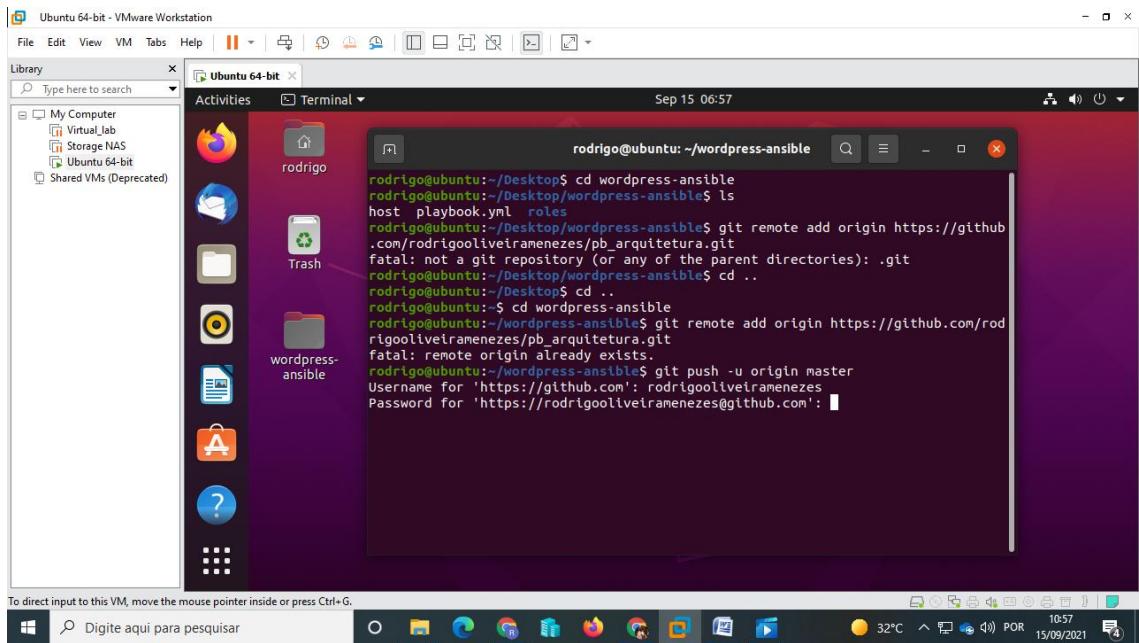
Então temos que entrar com o comando: git remote add origin https://github.com/rodrigooliveiramenezes/pb_arquitetura.git



Agora podemos dar o comando Push para enviar o código para o GitHub.

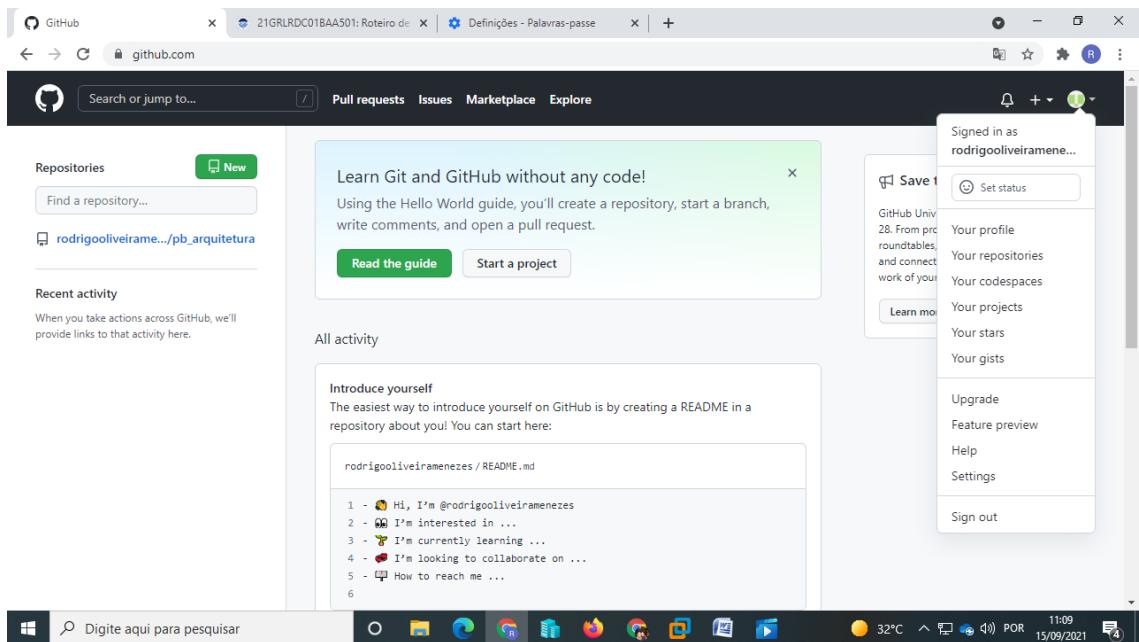
O comando é: git push -u origin master

Assim que teclar a tecla Enter, será solicitado o Username e a senha.



Nesse momento para entrar com a senha precisaremos gerar um Token copiar ele e colar no lugar da senha.

Para criar o Token basta entrar no GitHub, clicar no bolinha no canto superior a esquerda.



Quando abria a aba basta clicar em Settings e na próxima pagina click na opção “Develop Settings”

The screenshot shows the GitHub Profile settings page. On the left, there is a sidebar with various options: Security log, Security & analysis, Sponsorship log, Emails, Notifications, Scheduled reminders, SSH and GPG keys, Repositories, Packages, Organizations, Saved replies, Applications, Developer settings, Moderation settings (which is selected), and Blocked users. The main area contains fields for Bio, URL, Twitter username, Company, and Location. A note at the bottom states: "All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our [privacy statement](#) to learn more about how we use this information." The status bar at the bottom right shows it's 11:12, 32°C, POR, and the date is 15/09/2021.

Na próxima pagina clique em “Personal access token”

The screenshot shows the GitHub Apps settings page. The sidebar on the left has options: GitHub Apps (selected), OAuth Apps, and Personal access tokens. The main content area is titled "GitHub Apps" and includes a "New GitHub App" button. A note says: "Want to build something that integrates with and extends GitHub? [Register a new GitHub App](#) to get started developing on the GitHub API. You can also read more about building GitHub Apps in our [developer documentation](#)." At the bottom, there are links for Contact GitHub, Pricing, API, Training, Blog, and About.



Depois clique em “Generate a personal access token”

The screenshot shows the GitHub developer settings page. The 'Personal access tokens' tab is selected. A note explains that personal access tokens function like OAuth tokens and can be used instead of a password for Git over HTTPS or for API authentication. At the top right, there is a 'Generate new token' button.



Na próxima pagina, de um nome ao token, e nesse caso vai se chamar token_Lab, escolha o tempo para expirar e marque todos os quadrinhos e depois clique em gerar o token no fim da pagina.

The screenshot shows the 'New personal access token' creation form on GitHub. The 'Personal access tokens' tab is selected. The form includes fields for 'Note' (containing 'Token lab'), 'Expiration' (set to '30 days'), and a 'Select scopes' section. In the 'Select scopes' section, multiple checkboxes are checked, including 'repo', 'workflow', 'write:packages', and 'delete:packages'. Below the checkboxes, descriptions for each scope are provided.

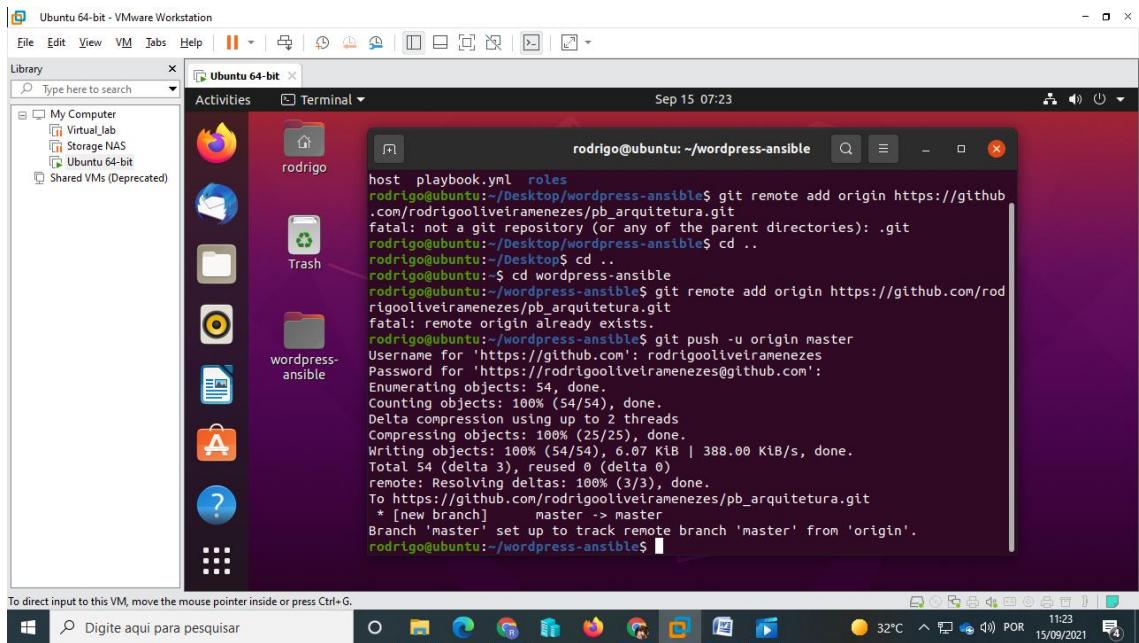
Scope	Description
repo	Full control of private repositories
repo:status	Access commit status
repo_deployment	Access deployment status
public_repo	Access public repositories
repo:invite	Access repository invitations
security_events	Read and write security events
workflow	Update GitHub Action workflows
write:packages	Upload packages to GitHub Package Registry
read:packages	Download packages from GitHub Package Registry
delete:packages	Delete packages from GitHub Package Registry

Na próxima pagina clique para copiar o token

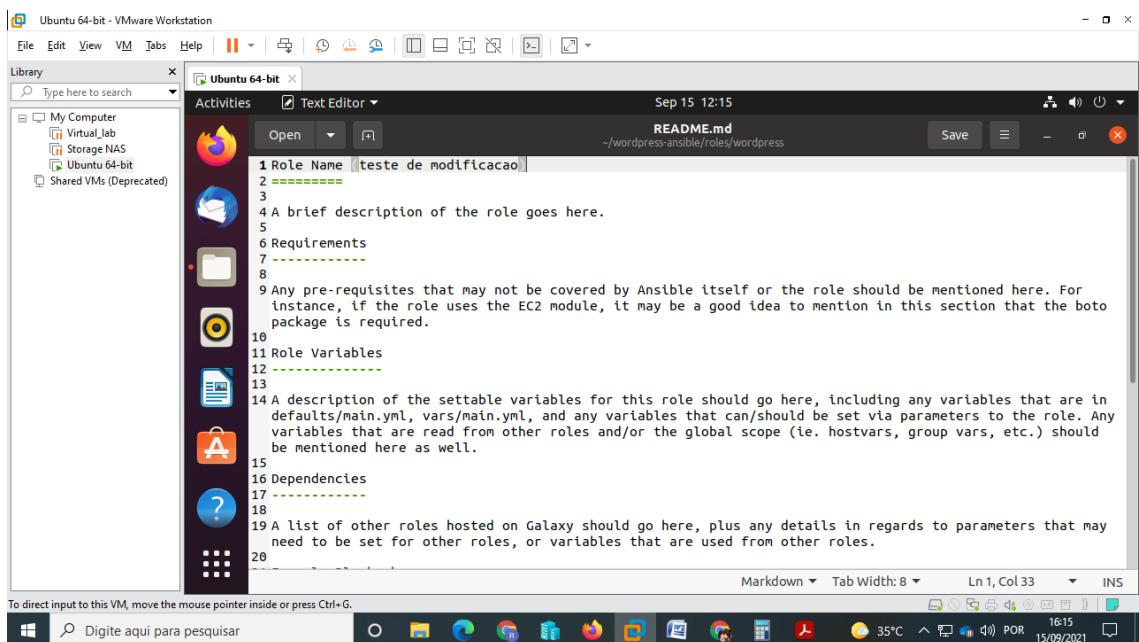
The screenshot shows a browser window with multiple tabs open. The active tab is 'Personal Access Tokens' on GitHub. The URL is github.com/settings/tokens. The page displays a list of tokens under the heading 'Personal access tokens'. One token is listed: 'ghp_4YeL1KPKNHKaKTzKVPj9RZqYAEAAI3xh9ex'. A note below the list says, 'Make sure to copy your personal access token now. You won't be able to see it again!'. At the top right, there are 'Generate new token' and 'Revoke all' buttons. The GitHub navigation bar at the top includes 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. Below the navigation bar, a message states, 'Some of the scopes you've selected are included in other scopes. Only the minimum set of necessary scopes has been saved.' A small 'X' button is to the right of this message.

E agora basta colar o token na linha de comando. Observe que não aparecerá nada.

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is 'Terminal' and the date is 'Sep 15 07:21'. The user is running the command 'git remote add origin https://github.com/rodrigoooliveiramenezes/pb_arquitetura.git'. The output shows an error: 'fatal: not a git repository (or any of the parent directories): .git'. The desktop environment includes a dock with various icons and a system tray at the bottom showing the date and time as '11:20 15/09/2021'.

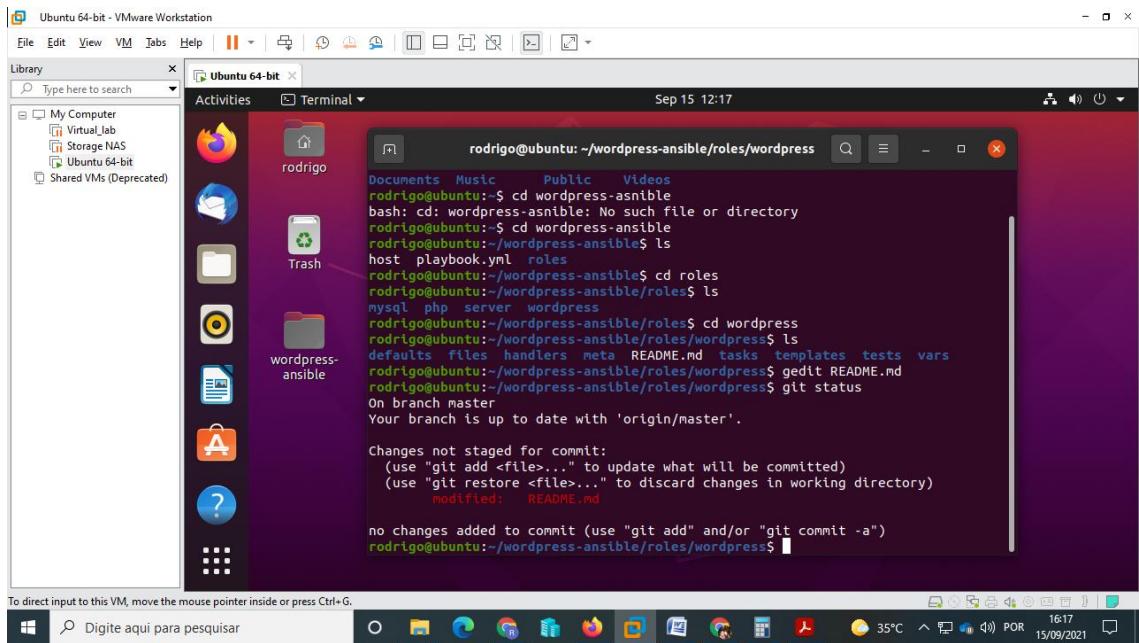


Agora vamos fazer um teste de alteração, para isso vamos entrar na pasta wordpress que esta dentro da roles e vamos modificar o arquivo README.md e colocaremos na primeira linha colocaremos em frente a “Role name” o texto “(teste de modificação)”.

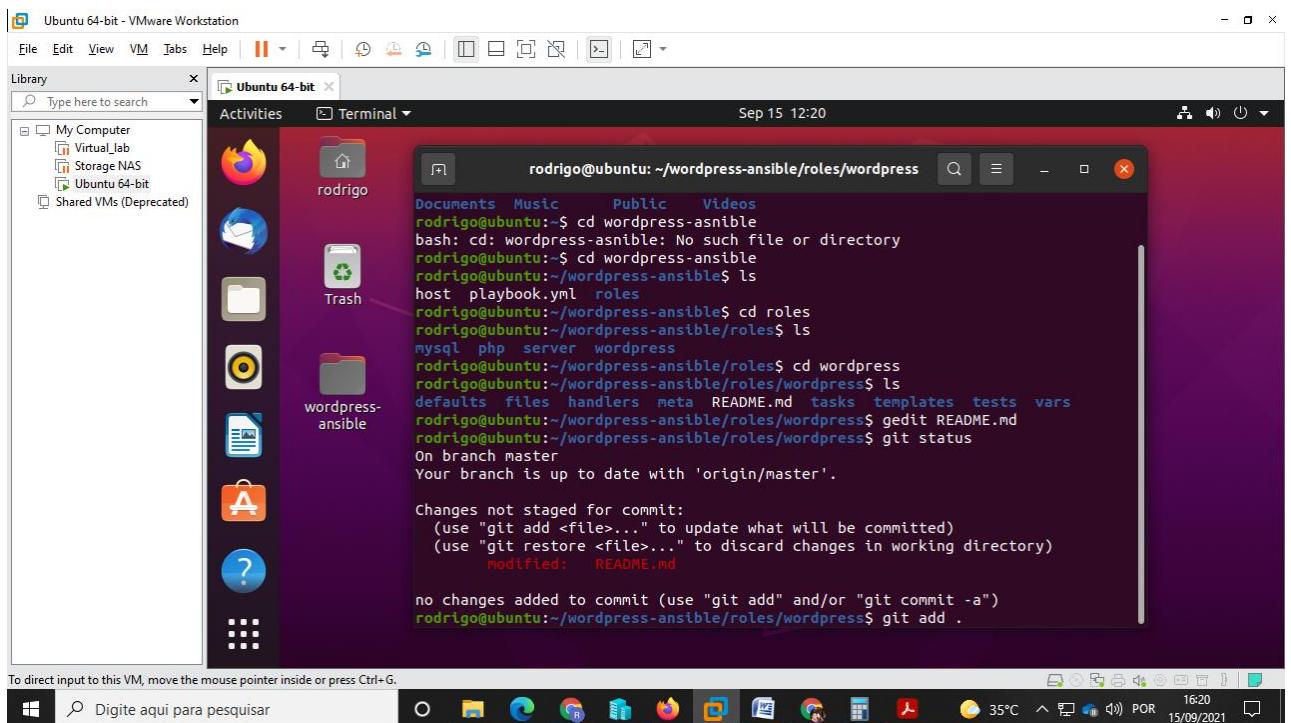


Agora basta salvar e fechar o gedit.

Só digitar o git.status e aparecerá o arquivo README.md em vermelho mostrando que foi modificado.



Próximo passo é só digitar “git add .” para poder adicionar as modificações.



Agora é só verificar se foi modificada digitando git status.

```

rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ cd wordpress
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ ls
defaults files handlers meta README.md tasks templates tests vars
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ gedit README.md
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git add .
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$
```

Nesse caso podemos observar que o arquivo README.md deixou de ser vermelho para ficar verde, o que mostra que foi modificado com sucesso.

Proximo passo é só “subir” a modificação para o Github com o comando “git push origin máster”

Será solicitado login e o token, basta colocar teclar enter

```

rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git add .
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   README.md

rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$ git push origin master
Username for 'https://github.com': rodrigooliveiramezes
Password for 'https://rodrigooliveiramezes@github.com':
Everything up-to-date
rodrigo@ubuntu:~/wordpress-ansible/roles/wordpress$
```

Modificação foi alterada

4. Capítulo III.

4.1 Conclusão

O presente projeto teve como principal objetivo documentar todo o procedimento da criação de uma página web para o escritório SIQUEIRA&MENEZES atendendo todas as necessidades solicitadas pelo cliente.

Os prazos para execução do projeto foram coerentes conforme planejamento traçado no item 2.1.

Restou comprovado que o projeto inicial descrito no diagrama contido no item 1.2 possuir recurso suficiente para colocar a solução em funcionamento, pois foi validado e demonstrado toda a funcionalidade da solução através da montagem de um ambiente alternativo de teste com requisitos mínimos e sem custo, apenas para atestar os requisitos funcionais, sem a necessidade de se realizar nenhum teste de desempenho.

Embora o diagrama contido no item 1.2 atenda as necessidades atuais do cliente, algumas melhorias podem ser realizadas, tornado o ambiente ainda mais seguro e menos suscetível a falhas. E para garantir o correto funcionamento do portal bem como sua disponibilidade e escalabilidade, uma estrutura ainda mais robusta seria necessário, conforme descrito abaixo:

TO BE:

- a) 2 x Servidores com mínimo Intel Core I7, 64 GB de RAM, 16 vCPU e 10 TB HD.
- b) 1 Storage NAS com Intel Core I7, 32 GB de RAM, 16 vCPU 4 Discos de 4TB organizados em volume RAID6
- c) 1 Switch 12 portas padrão Ethernet 1000
- d) 1 Roteador conectado à internet

Para acompanhamento e disponibilidade de todo o projeto, o mesmo encontra-se disponível em https://github.com/rodrigooliveiramenezes/pb_arquitetura