# Tutorial para criação e configuração do fmdev em uma VM Ubuntu

O tutorial tem como base o Ubuntu 20.04 instalado sob uma VM do Oracle VirtualBox.

- Oracle VirtualBox: https://www.virtualbox.org/wiki/Downloads
- Ubuntu 20.04: https://releases.ubuntu.com/focal/ubuntu-20.04.6-desktop-amd64.iso

**NOTA**: Após instalado o sistema operacional **não** deve ser feita atualização dos pacotes com `apt-get upgrade` ou `apt-get dist-upgrade`.

## Criação da VM

Antes de criar a VM é necessário verificar se o VirtualBox fez a instalação da placa de rede `Host-only Adapter`.

- Abrir o menu `File`, `Tools` opção `Network Manager` ou pelo atalho `Ctrl+H`
  - verificar se já há alguma rede, que geralmente tem o nome `vboxnet0`
  - Caso não exista rede, clique no botão `Create`
    - Deverá ser criada automaticamente uma placa de rede com configurações similares a IP `192.168.56.1/24`

Criar uma nova VM

- Menu `Machine`, opção `New...` ou pelo atalho `Ctrl+N`

Abaixo estão listadas apenas as configurações que foram modificadas, o restante segue o padrão.

- Name: `Ubuntu 20.04 - UPE - fmdev`
- ISO Image: `<not selected>`
- Base Memory: `2176 MB`
- Processors: `Máximo que sua CPU permite, até o limite verde`
- Enable EFI, caso seu computador comporte, deixe marcado, caso contrário o próprio VirtualBox desabilita
- Create a Virtual Disk Now: `120,00 GB`
  - Aqui não se preocupe com o espaço em disco, pois o disco virtual cresce gradualmente à medida que a VM requisita

Clique em `Finish` e com o botão direito sobre a nova VM selecione a opção `Settings...`.

- Seção `General`, aba `Advanced`, `Shared Clipboard`: `Bidirectional`
- Seção `General`, aba `Advanced`, `Drag'n Drop`: `Bidirectional`
- Seção `System`, aba `Motherboard`, `Boot Order`: Deixar marcado apenas `Hard Disk`
- Seção `System`, aba `Motherboard`, `Pointing Device`: `PS/2 Mouse`
- Seção `System`, aba `Processor`, `Extended Features`: Deixar marcado todas as opções que sua CPU permitir

- Seção `System`, aba `Acceleration`, `Hardware Virtualization`: Deixar marcado `Enable Nested Pagging`, caso sua CPU permita
- Seção `Display`, `Video Memory`: `128 MB`
- Seção `Storage`, no controlador da unidade de CD deve-se marcar `Use Host I/O Cache`
    - Na unidade de CD, clicar no ícone azul, representando um CD, e na opção `Choose a disk file...` para selecionar a imagem ISO do Ubuntu 20.04 que já deve te rsido baixada desde o início do tutorial
- Seção `Storage`, no controlador da unidade de disco rígido deve-se marcar `Use Host I/O Cache`
    - Na unidade de disco rígido deve-se marcar `Solid-state Drive`
- Seção `Network`, aba `Adapter 1`
    - Em `Advanced` deixar desmarcado `Cable Connected`
- Seção `Network`, aba `Adapter 2`, marcar `Enable Network Adapter`
    - Em `Attached to:` selecionar `Host-only Adapter`
    - Em `Name:` selecionar a placa de rede criada no início do processo, geralmente `vboxnet0`
    - Em `Advanced` deixar desmarcado `Cable Connected`

Clique em `OK`, mande iniciar a VM, deverá ser iniciado automaticamente pela imagem ISO.

## Instalação da VM

*ATENÇÃO*: Lembrar que a placa de rede virtual deve ficar desconectada

Quando abrir a tela de `Install`, selecionar a opção `Install Ubuntu` e deixar o idioma em `English`

- Escolher o teclado `Portuguese (Brazil)`
- Selecionar `Normal Installation` e marcar `Install third-party...`
- Deixar o padrão `Erase all disk and install Ubuntu` na seção de formatação do disco
- Selecionar o `timezone` para `Recife`
- Definir `Your name:` como `fmdev`
- Definir `your computer's name:` como `vmfmdev`
- Definir `Pick a username:`, `Choose a password:` e `Confirm your password:` como `fmdev`
- Se desejar, marque a opção `Log in automatically`

O processo de instalação deverá ocorrer normalmente, siga o que pede em tela. Ao final do processo a máquina virtual será reiniciada.

### Primeira inicialização

Faça login normalmente na interface gráfica com o usuário `fmdev`

- Na janela da VM do VirtualBOX
- Menu `Devices`, e na opção `Install Guest Additions CD Image`
- Seguir os processos que irão aparecer dentro da VM
    - Atente que ao final do processo irá aparecer uma mensagem no terminal similar a `Press return...`
- Reiniciar a VM

Após reiniciar a máquina virtual, reative as conexões de rede:

- Na janela da VM do VirtualBOX
- Menu `Devices`, e na opção `Network` marque as duas conexões de rede para que se conectem

### Configuração do sudo

**ATENÇÃO**: Só faça isso na máquina virtual e não numa máquina de trabalho, pois abre brecha de segurança.

```
sudo su

echo '

fmdev ALL=(ALL:ALL) NOPASSWD:ALL

' >> /etc/sudoers
```

- Feche e reabra o terminal para habilitar as modificações

### Configuração de alias

- Deverá ser feito com o usuário normal

```
echo "
#
alias l='ls -alh --color=auto'
" >> ~/.bashrc
```

- Deverá ser feito com o usuário root

```
sudo su

echo "
#
alias l='ls -alh --color=auto'
" >> ~/.bashrc
```

### Modificar configuração de memória da VM

```
sudo su

echo "
###
# Fonte: https://www.kernel.org/doc/Documentation/sysctl/vm.txt
# Reduz o uso de SWAP
vm.swappiness=20
```

```
# Melhora a gestão de cache
vm.vfs_cache_pressure=60

# Clean caches
## Note: Using vm.drop_caches can cause a deadlock if the system is under heavy
memory and I/O load!!!
## vm.drop_caches=1 - In order to clear PageCache only
## vm.drop_caches=2 - In order to clear dentries (Also called as Directory Cache)
and inodes
## vm.drop_caches=3 - In order to clear PageCache, dentries and inodes
vm.drop_caches=3
" >> /etc/sysctl.conf

sudo sysctl -p

sudo init 6
```

## Instalação de pacotes principais do Ubuntu

```
sudo apt-get update -y

sudo apt-get install -y axel build-essential checkinstall curl git git-extras gufw
libbz2-dev libc6-dev libffi-dev libgdbm-dev liblzma-dev libncursesw5-dev
libreadline-gplv2-dev libsqlite3-dev libssl-dev net-tools nginx postgresql
software-properties-common tk-dev tzdata wget yarn zlib1g-dev
```

## Instalação e configuração do PostgreSQL

- A configuração a seguir irá permitir acesso ao postgresql em todas as placas de rede e IPs:

```
sudo su

echo "
listen_addresses = '*'
" >> /etc/postgresql/12/main/postgresql.conf

echo "
host    all    all        0.0.0.0/0        md5
" >> /etc/postgresql/12/main/pg_hba.conf
```

- Habilitar e iniciar o serviço para conectar ao SGBD e criar base de dados e usuário

```
sudo systemctl enable --now postgresql

sudo systemctl restart postgresql
```

```
sudo su
su postgres
psql
\c postgres
CREATE DATABASE fmdev
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    TABLESPACE = pg_default
    IS_TEMPLATE = False;

CREATE ROLE root WITH
    SUPERUSER
    CREATEDB
    CREATEROLE
    INHERIT
    LOGIN
    password 'root';

GRANT ALL PRIVILEGES ON DATABASE fmdev TO root;

COMMIT;

\q

exit

exit
```

### Instalação do PGAdmin4

```
sudo su

sudo curl -fsS https://www.pgadmin.org/static/packages_pgadmin_org.pub | sudo gpg
--dearmor -o /usr/share/keyrings/packages-pgadmin-org.gpg

sudo sh -c 'echo "deb [signed-by=/usr/share/keyrings/packages-pgadmin-org.gpg]
https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) pgadmin4
main" > /etc/apt/sources.list.d/pgadmin4.list'

sudo apt-get update -y && sudo apt-get install -y pgadmin4-desktop
```

- Abrir o PGAdmin4 pelo Dash do Ubuntu e configure a conexão para localhost com usuário e senha root

## Instalação de pacotes extras do Ubuntu

```
sudo su
```

```
curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | sudo apt-key add -

echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee
/etc/apt/sources.list.d/yarn.list

sudo apt update -y && sudo apt install -y yarn
```

## Instalação do NodeJS

**NOTA**: Devido às características do fmdev, a instalação do NodeJS deverá ser feita para o usuário root, e não para o usuário comum.

```
sudo su

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.4/install.sh | bash
```

- Feche e reabra o terminal para habilitar as modificações

```
sudo su

nvm install 13.9.0

nvm use 13.9.0
```

## Baixar e configurar o fmdev

- Pode usar a versão inicial, e rezar que funcione na VM

```
sudo su

cd && git clone https://github.com/prof-alexandre-maciel/fmdev.git
```

- Ou usar a versão modificada que já está testada e funciona na VM

```
sudo su

cd && git clone https://github.com/upe-ppgec/fmdev.git
```

### Frontend

- Copiar o arquivo .env do frontend

```

```

```
sudo su

cd /root/fmdev/frontend && cp .env.template .env
```

- Verificar se o conteúdo do arquivo .env criado corresponde ao conteúdo abaixo:

```
REACT_APP_SMARTLOOK=
REACT_APP_HOST=http://localhost:5000
```

- Compilar o frontend
    - **DEMORA UM TIQUIN**

```
sudo su

sudo rm -rf /root/fmdev/frontend/build

cd ~/fmdev/frontend

yarn install

export NODE_OPTIONS='--max_old_space_size=3072'

npx browserslist@latest --update-db

yarn build

export NODE_OPTIONS='--max_old_space_size=512'
```

## Habilitar e configurar o Nginx

```
sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/default.old

sudo su

echo 'server {
    listen 80 default_server;
    listen [::]:80 default_server;
    root /var/www/html;
    index index.html index.htm index.nginx-debian.html;

    server_name _;

    location / {
        try_files $uri $uri/ =404;
        add_header Cache-Control "no-cache";
    }
```

```
    location /static {
        expires 1y;
        add_header Cache-Control "public";
    }

    location /api {
        include proxy_params;
        proxy_pass http://127.0.0.1:5000;
    }
}
' > /etc/nginx/sites-available/default

sudo systemctl enable --now nginx

sudo systemctl restart nginx
```

- Abrir e testar no navegador http://localhost
- Será exibida a página padrão do Nginx

### Copiar arquivos do frontend para o NGinx

```
sudo rm -rf /var/www/html/*

sudo cp -Rfv /root/fmdev/frontend/build/* /var/www/html

sudo systemctl restart nginx
```

## Passos do backend com o Python 3.7

- Compilação do Python 3.7.6
    - **DEMORA UM TIQUIN**

```
sudo su

cd /usr/src && sudo wget https://www.python.org/ftp/python/3.7.6/Python-3.7.6.tgz

cd /usr/src && sudo tar xzf Python-3.7.6.tgz

cd /usr/src/Python-3.7.6 && sudo ./configure --enable-optimizations

cd /usr/src/Python-3.7.6 && sudo make altinstall
```

### Verificar a versão

```
sudo python3.7 -V
```

## Configuração do serviço fmdev

- Criação e instalação dos pacotes de requisitos no virtual env
    - **DEMORA UM TIQUIN**

```
sudo su

cd /root/fmdev/backend && python3.7 -m venv venv

cd /root/fmdev/backend && source venv/bin/activate

cd /root/fmdev/backend && pip install -r requirements.txt

cd /root/fmdev/backend && pip install flask flask_migrate flask_script python-
dotenv

exit
```

- Criar o arquivo `/root/fmdev/start.sh`

```
sudo nano /root/fmdev/start.sh
```

- Verificar se já existe, ou adicionar as linhas exatamente conforme a seguir:

```
#!/bin/bash
export PATH=${PATH}:/root/fmdev/backend/venv/bin
cd /root/fmdev/backend & \
    source /root/fmdev/backend/venv/bin/activate & \
    gunicorn --chdir /root/fmdev/backend -b 0.0.0.0:5000
"run:create_app('config')"
```

- Tornar o arquivo executável:

```
sudo chmod +x /root/fmdev/start.sh
```

- Copiar o arquivo .env do backend

```
sudo su

cd /root/fmdev/backend && cp .env.template .env
```

- Verificar se o conteúdo do arquivo .env criado corresponde ao conteúdo abaixo:

```
DB_HOST=localhost
DB_USER=root
DB_PORT=5432
DB_PWD=root
DB_NAME=fmdev
```

- Criar o arquivo de serviço Linux para o fmdev:

```
sudo su

echo '[Unit]
Description=FMDEV Service
After=network.target

[Service]
User=root
WorkingDirectory=/root/fmdev/backend
Environment="PATH=/root/fmdev/backend/venv/bin"
ExecStart=/root/fmdev/start.sh
Restart=always

[Install]
WantedBy=multi-user.target
' > /etc/systemd/system/fmdev.service
```

```
sudo systemctl daemon-reload && sudo systemctl enable --now fmdev

sudo systemctl status fmdev
```

- Verificar serviços de rede rodando com o comando:

```
sudo su

sudo netstat -tunalp
```

- Os serviços de rede que nos interessam devem estar similares ao mostrado abaixo (postgres, python3.7 e nginx):

```
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 127.0.0.1:5432          0.0.0.0:*               LISTEN
904/postgres
```

```
tcp        0      0 0.0.0.0:5000            0.0.0.0:*               LISTEN
859/python3.7
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
916/nginx: master p
tcp6       0      0 :::80                   :::*                    LISTEN
916/nginx: master p
udp        0      0 127.0.0.1:54528         127.0.0.1:54528         ESTABLISHED
904/postgres
```

### Configurar as migrations do banco

- Tem que rodar exatamente conforme abaixo:

```
sudo su

cd /root/fmdev/backend && source venv/bin/activate

python3.7 migrate.py db stamp head

python3.7 migrate.py db migrate

python3.7 migrate.py db upgrade

exit
```

## Popular dados na base fmdev

- Deverá ser feito pelo PGAdmin4.

### Importar dados gerais

- Como os arquivos estão na pasta do root, é necessário copiá-los e dar permissão

```
sudo su

sudo cp /root/fmdev/backend/scripts/main.sql /home/fmdev/Desktop

sudo chown fmdev:fmdev /home/fmdev/Desktop/main.sql
```

- No PGAdmin4, clicar com o botão direito sobre a base de dados `fmdev`, opção `Query Tool`
- Clicar na pasta preta para abrir, ou atalho `Alt+O`
- Abrir o arquivo de dados sql `/home/fmdev/Desktop/main.sql`
- Remover a última linha do arquivo, linha 155
- Escrever no lugar: `commit ;`
- Selecionar todo o conteúdo do arquivo com o atalho `Ctrl+A`

- Executar como arquivo sql clicando no botão de play, ou apertar a tecla `F5`
- Irá aparecer um balão verde de sucesso

### Importar dados do moodle

- Como os arquivos estão na pasta do root, é necessário copiá-los e dar permissão

```
sudo su

sudo cp /root/fmdev/base_fmdev.csv /home/fmdev/Desktop

sudo chown fmdev:fmdev /home/fmdev/Desktop/base_fmdev.csv
```

- No PGAdmin4, clicar com o botão direito sobre a base de dados `fmdev`, opção `Refresh...`
- Expandir a base de dados `fmdev`, com dois cliques sobre o nome da base
- Expandir `Schemas`, expandir `public`, expandir `tables`
- Clicar com o botão direito sobre a tabela `moodle`, opção `Import/Export Data...`
- Selecionar o arquivo `/home/fmdev/Desktop/base_fmdev.csv`
- Na aba `options`, ativar `HEADERS`
- Na aba `options`, em `DELIMIMTER` selecionar `;`
- Clicar em `Ok`, irá aparecer dois balões verdes de sucesso
- Pode fechar o PGAdmin4

## Habilitar e configurar o ufw

```
sudo systemctl enable --now ufw

sudo ufw enable

sudo ufw allow 'Nginx Full'
sudo ufw allow 5432
sudo ufw allow 5000

sudo ufw status verbose
```

## Configurar o run level target do Ubuntu para somente texto

```
sudo systemctl enable multi-user.target

sudo systemctl set-default multi-user.target

sudo init 6
```

**ATENÇÃO**: A partir daqui o Ubuntu irá reiniciar no modo texto, sem interface gráfica, reduzindo assim o consumo de recursos.

Se eventualmente desejar entrar no modo gráfico, rode no terminal:

```
sudo init 5
```

## Aplicaçao do fmdev

A partir da VM é possível abrir o navegador em `https://localhost`

**NOTA**: Como a VM agora possui duas placas de rede, sendo uma a `Host-only`, é possível acessar os recursos da VM a partir da máquina real `Host`.

A partir da máquina real `Host` será possível se conectar aos serviços do PostgreSQL, NGinx e da API em Python. Para tanto verifique qual IP está alocado para a placa `Host-only`, com o comando:

```
ifconfig
```

A placa de rede virtual deverá ter um IP na forma `192.168.56.0/24`, use esse endereço encontrado para abrir no navegador da máquina real `Host`. Exemplo: `https://192.168.56.101`