Implantação do XCLIN e configuração do servidor

Criando o usuário não root

Crie um usuário de nome xy190 com o comando abaixo:

```
useradd -s /bin/bash -aG sudo -m xy190
```

Será criada uma pasta em home. Isto é, /home/xy190

Altere o password **sudo** para **xy190**:

```
su -
passwd xy190
```

Crie agora a pasta **.ssh** e o arquivo **authorized_keys** na pasta do usuário **xy190**. Esse arquivo deve conter as chaves públicas para conexão via ssh sem senha de root.

```
su - xy190

mkdir .ssh
cd .ssh
touch authorized_keys
```

Configuração do ssh

Criar as chaves privadas e públicas através do **putty** utilizando o comando abaixo:

```
ssh-keygen
```

Por padrão, será criada a chave **id_rsa** que é a chave privada e a chave **id_rsa.pub** que é a chave pública.

Transferir via **sftp** o arquivo **id_rsa.pub** para o servidor e adicionar ao arquivo **authorized_keys** conforme baixo:

```
cat id_rsa.pub > .ssh/authorized_keys
```

Isto supondo que o arquivo **id_rsa.pub** foi transferido para o diretório /**home/xy190** e você entrou neste diretório.

Agora é hora de configurar o arquivo **/etc/ssh/sshd_config**. Edite este arquivo e configure como a seguir:

```
Port 37814
#PermitRootLogin
PublicKeyAuthentication yes
```

No final do arquivo adicione a seguinte linha:

```
AllowUsers xy190
```

Reinicie o **ssh** com o seguinte comando:

```
systemctl restart sshd
```

Instalação e configuração do JDK

Para instalação do jdk, pode ser utilizado o comando wget para fazer download do site da oracle. Exemplo como segue:

```
wget https://download.oracle.com/java/20/latest/jdk-20_linux-x64_bin.tar.gz
```

Extrair o conteúdo para a pasta /opt que, neste caso, fica assim:

```
/opt/jdk-20.0.1
```

Instalação e configuração do tomcat

Baixar com o comando **wget** como segue:

```
wget https://dlcdn.apache.org/tomcat/tomcat-10/v10.1.10/bin/apache-tomcat-10.1.10.tar.gz
```

Extrair o conteúdo para pasta **/opt** que, neste caso, deve fica assim:

```
/opt/tomcat
```

Após isto, é necessário criar um usuário para o **tomcat** com os seguintes comandos:

```
sudo useradd -s /bin/false -d /opt/tomcat tomcat
sudo chown -R tomcat:tomcat /opt/tomcat
sudo chmod u+x /opt/tomcat/bin
```

Com a opção -s /bin/false, ninguém poderá entrar na conta criada!

Após isto, editar o arquivo **server.xml** de configuração do **tomcat** para desabilitar o shutdown via script **shutdown.sh**.

Altere a seguinte linha no arquivo **server.xml**:

```
<Server port=”8005” shutdown=”SHUTDOWN”>
```

Para:

```
<Server port=”-1” shutdown=”SHUTDOWN”>
```

Configurar o serviço do tomcat:

Crie o arquivo **tomcat.service** como abaixo:

touch /etc/systemd/system/tomcat.service

Edite o arquivo com o conteúdo parecido com o seguinte:

```
[Unit]
Description=Tomcat servlet container
After=syslog.target network.target

[Service]
Type=forking

User=tomcat
Group=tomcat

Environment="JAVA_HOME=/opt/jdk-20.0.1"
Environment="JAVA_OPTS=-Djava.security.egd=file:///dev/urandom"

Environment="CATALINA_BASE=/opt/tomcat"
Environment="CATALINA_HOME=/opt/tomcat"
Environment="CATALINA_PID=/opt/tomcat/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1024M -server -XX:
+UseParallelGC”

ExecStart=/opt/tomcat/bin/startup.sh
ExecStop=/bin/kill -15 $MAINPID

[Install]
WantedBy=multi-user.target
```

Manipule o serviço como segue:

```
sudo systemctl start tomcat
sudo systemctl stop tomcat
sudo systemctl restart tomcat
sudo systemctl status tomcat
```

Instalação e configuração do apache

Instalar com o seguinte comando:

```
sudo apt install apache2
```

Instale o certbot para configuração do SSL via Let’s Encrypt

```
sudo apt install certbot python3-certbot-apache
```

Feito isto, configurar o VirtualHost como segue:

```
<VirtualHost *:80>
    ServerAdmin italoherbert@outlook.com
```

```
        ServerName xclin.com.br
        ServerAlias www.xclin.com.br
        DocumentRoot /var/www/xclin.com.br

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log
</VirtualHost>
```

Crie o diretório **/var/www/xclin.com.br/** e crie um arquivo **html** neste diretório com nome **index.html** e um conteúdo de sua preferência para testar o acesso.

Agora configure o certificado com o seguinte comando:

```
certbot –apache -d xclin.com.br -d www.xclin.com.br
```

Agora reinicie o apache e acesse a página via https

Configurando a integração tomcat/apache

Será utilizado o módulo **mod_proxy** para esta configuração

Por padrão o mod_proxy já vem instalado com o apache, bastando apanas habilitá-lo como segue:

```
sudo a2enmod proxy
sudo a2enmod proxy_http
```

Feito isto, configure o Proxy e ProxyReverse no arquivo de configuração do VirtualHost

```
#SSLProxyEngine on
ProxyPass /api http://localhost:8080/api
ProxyPassReverse /api http://localhost:8080/api
```

Atenção: O arquivo de **host** após configurado o certificado **SSL** é: **xclin.com.br-le-ssl.conf**

Agora configurar no **tomcat** o arquivo **server.xml**. Localize a seguinte linha:

```
<Connector port=”8080” …
```

Adicione as opções **proxyName** e **proxyPort** como segue:

```
<Connector port=”8080” …  proxyName=”localhost” proxyPort=”80” />
```

Instalando o postgresql

Primeiramente, instalar o postgresql 13 com os seguintes comandos:

```
sudo apt update
sudo apt install postgresql-13
```

Após a instalação, entrar com usuário do sistema que foi criado com a instalação: O usuário **postgres**. Então, faça como segue:

```
sudo passwd postgres
```

Crie uma senha e acesse a conta do usuário postgres:

```
su – postgres
```

Agora utilize o comando psql para entrar no cliente do postgresql:

```
psql
```

Agora você pode administrar o banco de dados através desse procedimento.

Instalação da aplicação xclin

Crie e configure o **banco de dados xclin** e **usuário xclin** como segue. Claro, antes acessar o **cliente postgresql** via comando **psql**.

```
create database xclin;

create user xclin with password '3950Vm08';

alter database xclin owner to xclin;

grant all on database xclin to xclin;
grant all on all tables in schema public to xclin;
grant all on all sequences in schema public to xclin;
```

É necessário também a criação da extensão “unaccent” que é utilizada para retirar acentos com uso da função lower_unaccent definida abaixo:

```
create or replace function lower_unaccent( varchar ) returns varchar as $$
begin
   return lower(unaccent($1));
end;
$$ language plpgsql;
```

A função unaccent pode ser habilitada com o seguinte comando:

```
create extension unaccent;
```

Após isto, transfira o backup de banco de dados de nome **xclin.sql** para pasta (por exemplo) **resources** e então entre no **psql** e execute:

```
psql xclin ou \c xclin

\i /home/xy190/recursos/xclin.sql
```

```
grant all on all tables in schema public to xclin;
grant all on all sequences in schema public to xclin;
```

Configure o serviço do tomcat com as seguintes variáveis de ambiente:

```
Environment=”XCLIN_DB_URL=jdbc:postgresql://localhost:5432/xclin"
Environment="XCLIN_DB_USERNAME=xclin"
Environment=”XCLIN_DB_PASSWORD=3950Vm08”
```

Após configurado o banco de dados e as variáveis de ambientes, também, do banco de dados, faça o seguinte:

Instale o unzip com o seguinte comando:

```
sudo apt install unzip
```

Transfira via **sftp** o arquivo **war** do xclin para pasta do usuário e, então, crie o **script** a seguir com nome **deploy-backend.sh**

```
cd /opt/tomcat/webapps
rm -r ROOT
unzip /home/xy190/xclin.war -d ROOT
chown -R tomcat:tomcat ROOT
systemctl restart tomcat
```

Não esqueça de tornar o arquivo deploy.sh executável com o seguinte comando:

```
sudo chmod +x deploy-backend.sh
```

Execute o script:

```
sudo ./deploy.sh
```

Agora transfira os arquivos do frontend angular via sftp para a pasta: /home/xy190/xclin.com.br/

Feito isto, crie o script deploy-frontend.sh

```
vim deploy-frontend.sh
```

Esse arquivo deve ter o seguinte conteúdo:

```
cd /home/xy190/xclin.com.br

if [ `pwd` = '/home/xy190/xclin.com.br' ]; then
        rm -r /var/www/xclin.com.br/*
        cp -r * /var/www/xclin.com.br/

        systemctl restart apache2
fi
```

Não esqueça de conceder permissão de execução do arquivo criado com o seguinte comando:

```
sudo chmod +x deploy-frontend.sh
```

Execute o script:

```
sudo ./deploy-frontend.sh
```

Instalando o certificado do serviço do IBGE

Agora falta apenas adicionar o certificado digital do serviço de localização de cidades e Ufs do ibge. Faça conforme abaixo para baixar o certificado:

```
openssl s_client -connect servicodados.ibge.gov.br:443
```

Serão mostrados vários dados, inclusive o certificado. Então, selecione o certificado e copie o conteúdo para um arquivo. Pode ser:

```
trust.pem
```

Então adicione o certificado ao arquivo cacerts do jdk conforme o seguinte:

```
/opt/jdk-20.0.1/bin/keytool -import -trustcacerts -noprompt -storepass changeit -alias servicodados.ibge.com.br -file trust.pem -keystore /opt/jdk-20.0.1/lib/security/cacerts
```

Instalando a fonte

É necessário também instalar uma fonte no linux para funcionar os relatórios. Então, faça como segue:

```
sudo apt install fontconfig
sudo fc-cache –force
```

Pronto! Feito isto, tudo deve estar configurado e, com os dois serviços dos servidores tomcat e apache em execução, pode testar o sistema pelo link abaixo:

https://www.xclin.com.br