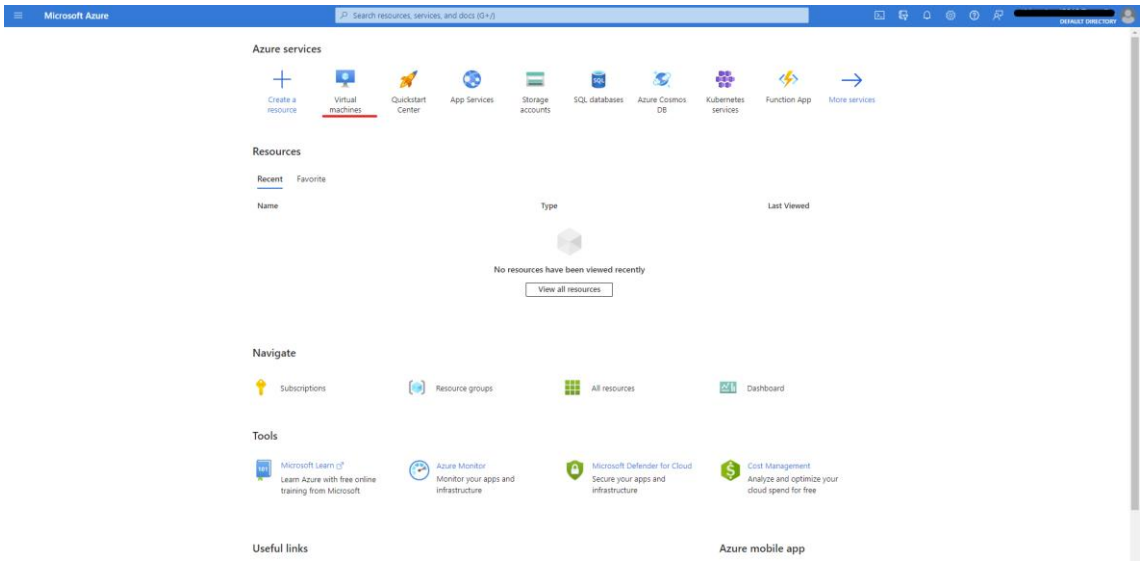
Configurando um servidor de aplicação NodeJS na Microsoft Azure

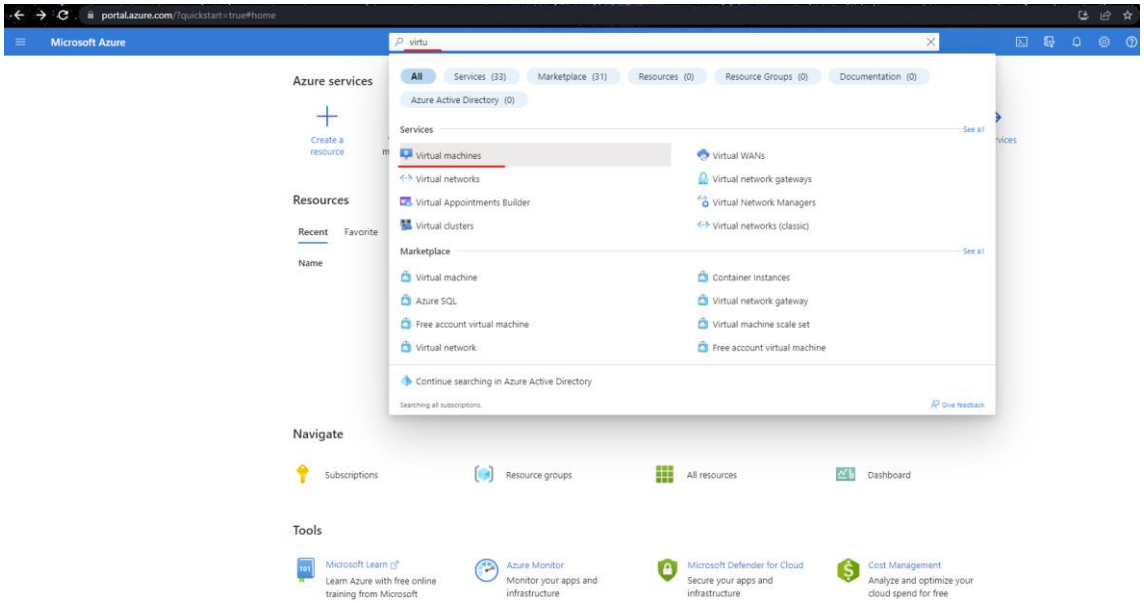
Olá, a seguinte vai ser encontrar um tutorial contendo o passo a passo de como criar e configurar um servidor de aplicação NodeJs Na Microsoft Azure.

O primeiro passo é estar criando a sua conta grátis, que pode ser efetuado pelo link a seguir: https://azure.microsoft.com/en-us/free/. (Lembrando, a conta é gratuita, entretanto vai pedir um número de cartão mesmo assim, podendo gerar cobranças dependendo o uso, então tenha cuidado).

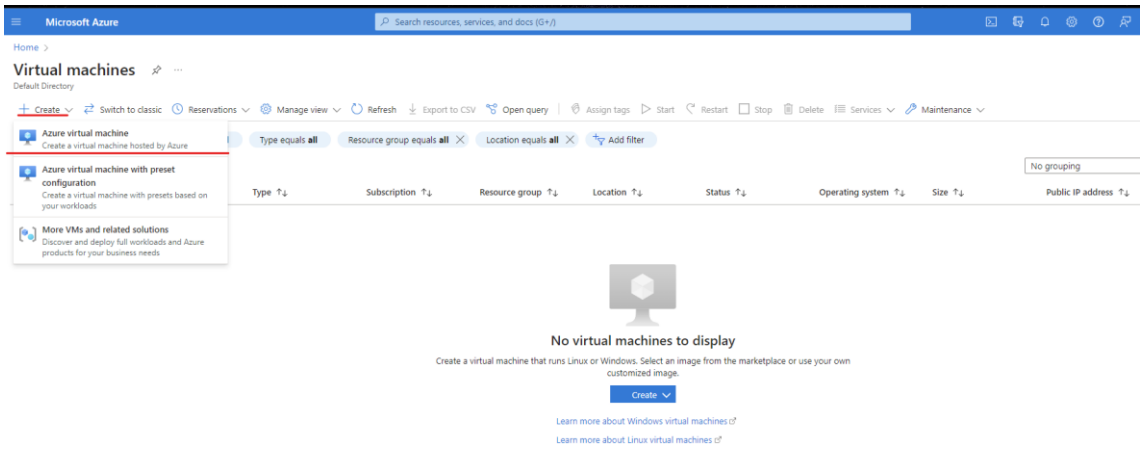
Após a criação da conta na Azure, selecione a opção Virtual Machines.



Caso não apareça como na scrrenshot acima, basta pesquisar no campo de busca:

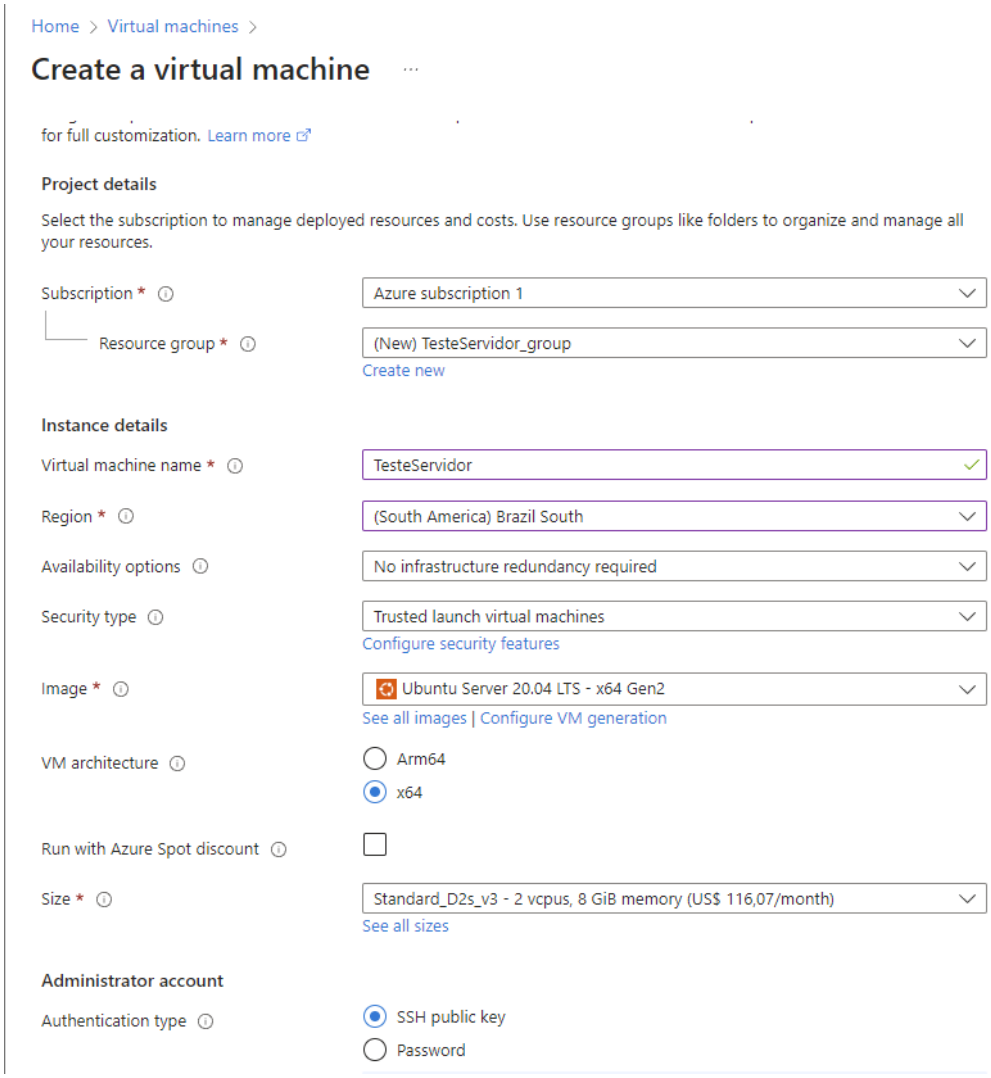


Selecione o Botão Create, o que vai abrir uma pequena tela com algumas opções:



Selecione a primeira opção, no caso Azure Virtual Machine.

Após o passo acima, vai aparecer a seguinte tela:

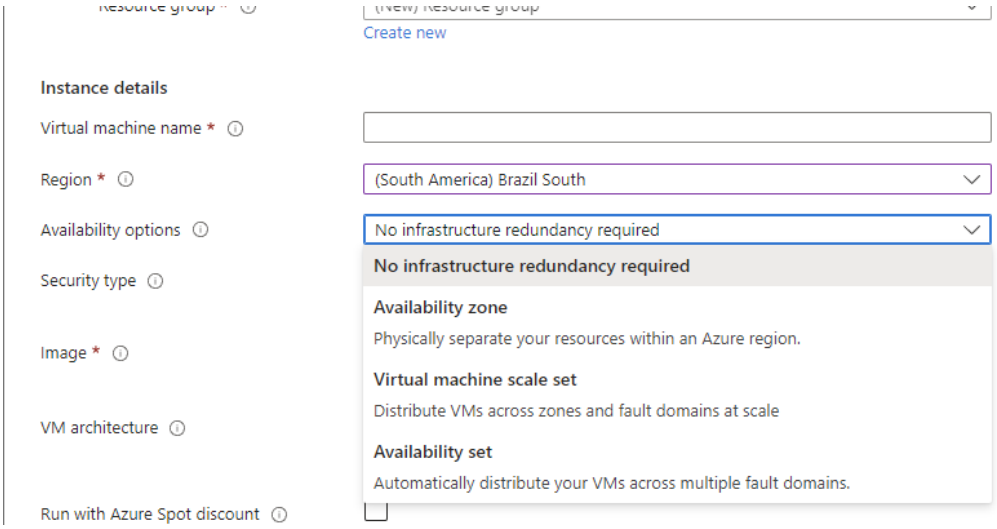


O Resource group já vem com um nome por padrão, você pode estar alterando-o para qual desejar.

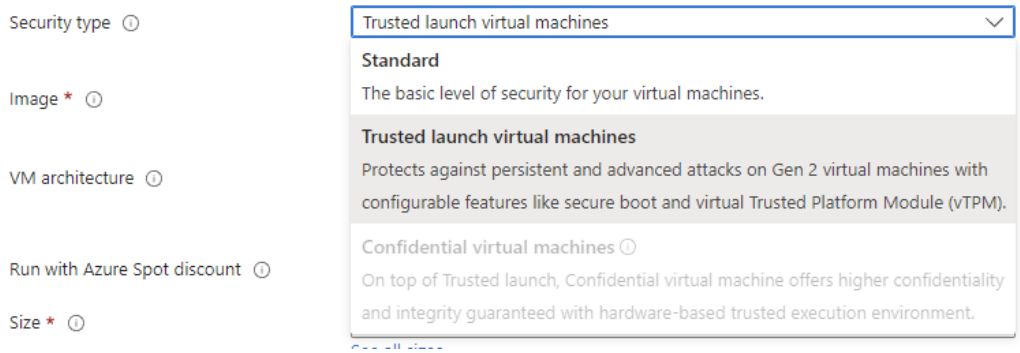
O Virtual Machine name pode ser preenchido com o nome que você preferir

Na Region, poderá escolher o que lhe melhor atender, o servidor dos Estados Unidos (US) tende a ser mais barato, porém tem a questão de latência.

Em Availability Options caso deseja que o servidor esteja replicado em diversos pais ao mesmo tempo.

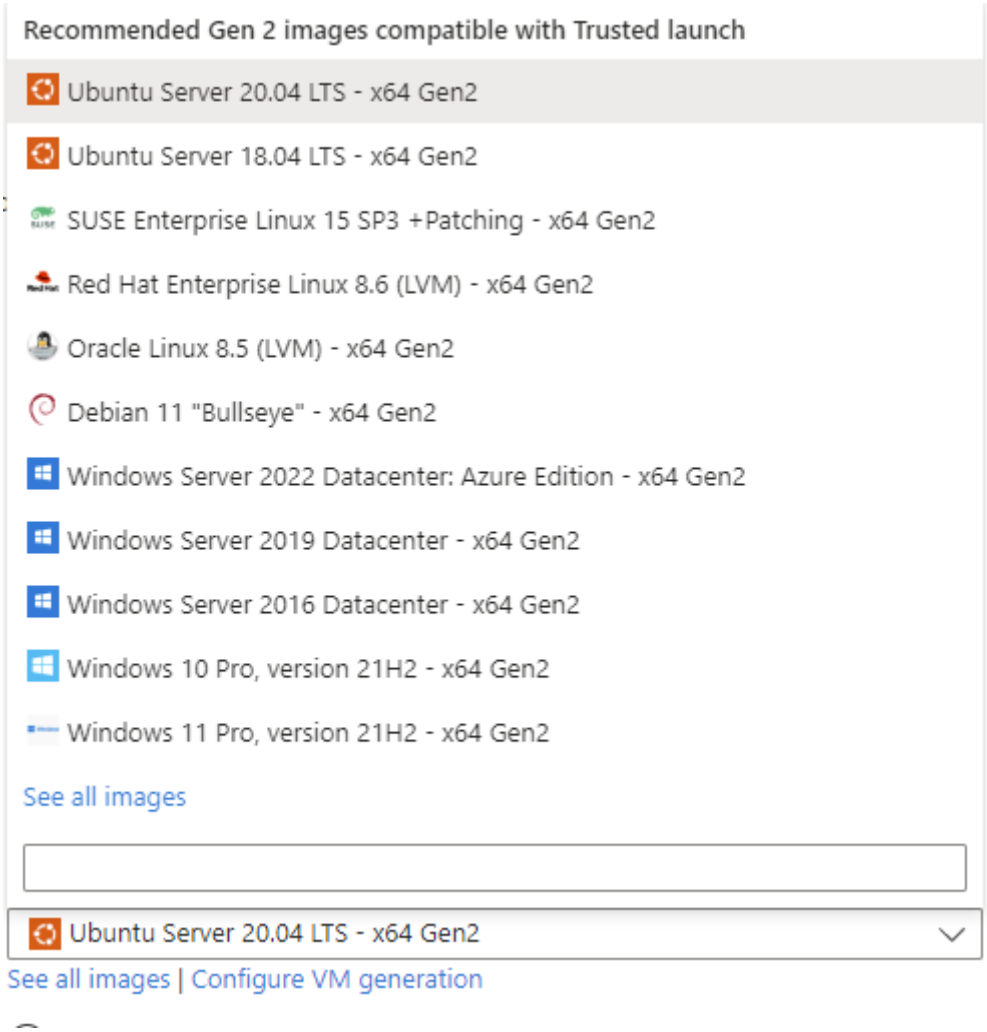


Em Security type, tem duas opções. A Standard e a Trusted launch virtual machines.



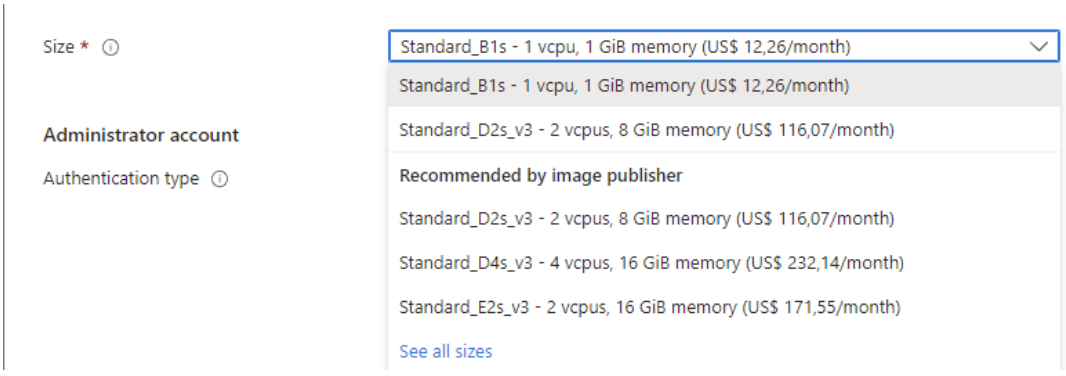
Escolha de acordo com a sua preferência.

Em Image, é por onde você escolhera o sistema operacional do servidor:

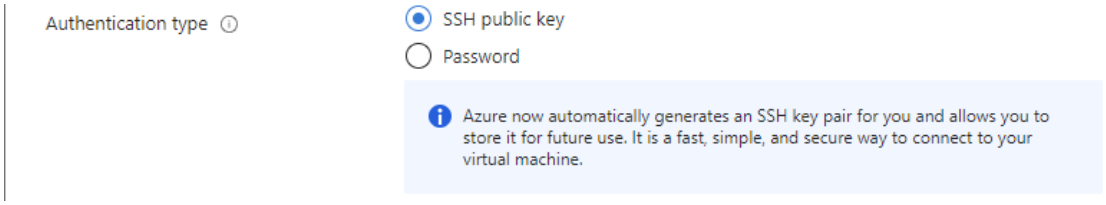


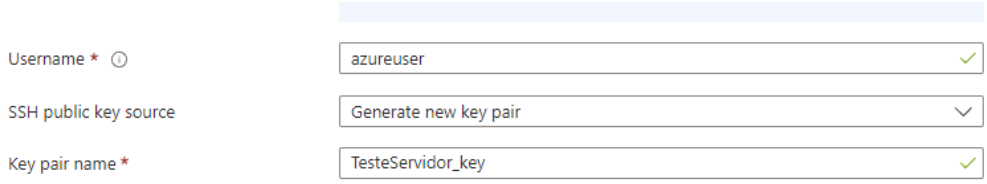
Em VM architecture tem duas opções, infelizmente no Brasil só temos acesso a x64.

Em size, é referente ao tamanho do servidor, como estou utilizando a conta grátis, optei pelo menor servidor, entretanto poderá escolher de acordo com a sua necessidade:



Em Authentication type, optei por SSH public key





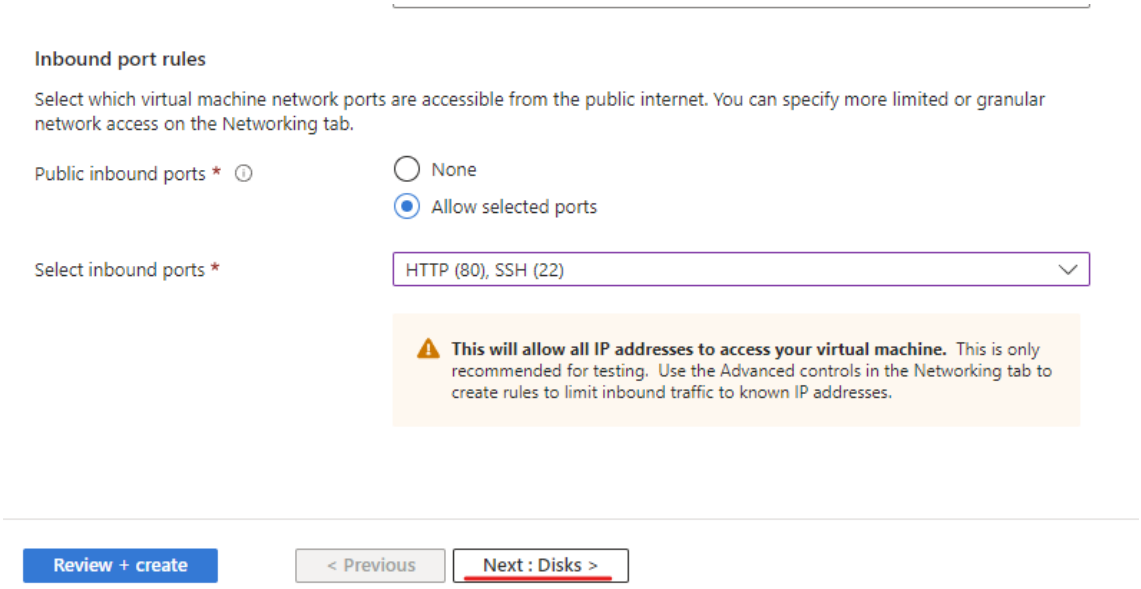
Em Username optei por deixar o padrão.

Em SSH public key source coloquei para gerar uma nova chave. Entretanto existe outras opções, utilize a que lhe servir melhor.

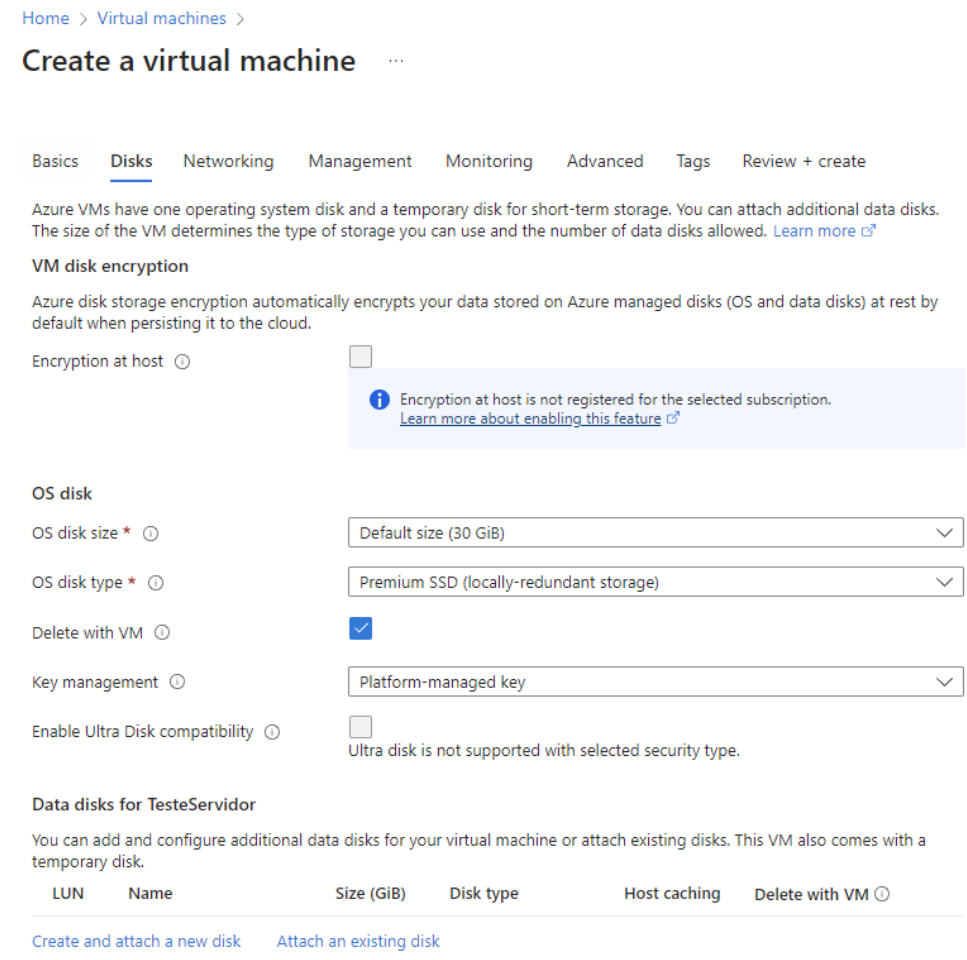
Em Key pair name: Optei por deixar o nome que já veio por padrão.

Nas inbound ports, deixei a 22 e a 80 habilitadas:

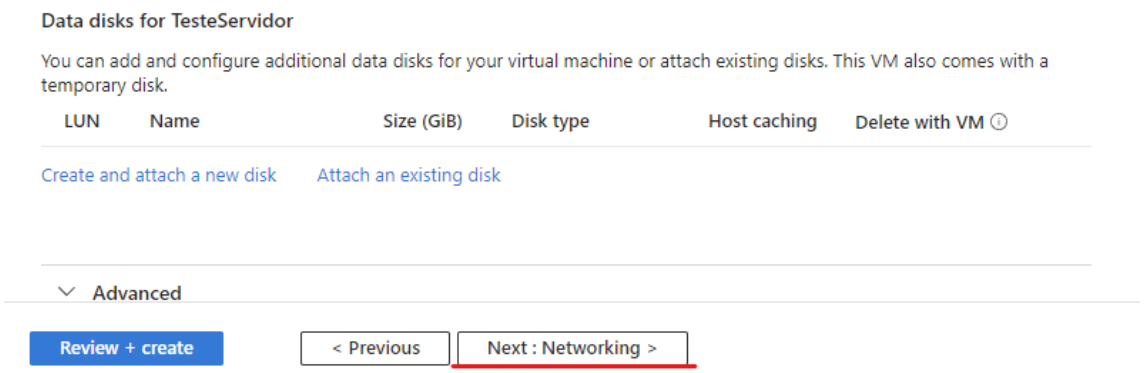
Após concluir essa etapa da configuração, selecione o seguinte botão para dar prosseguimento com a configuração:



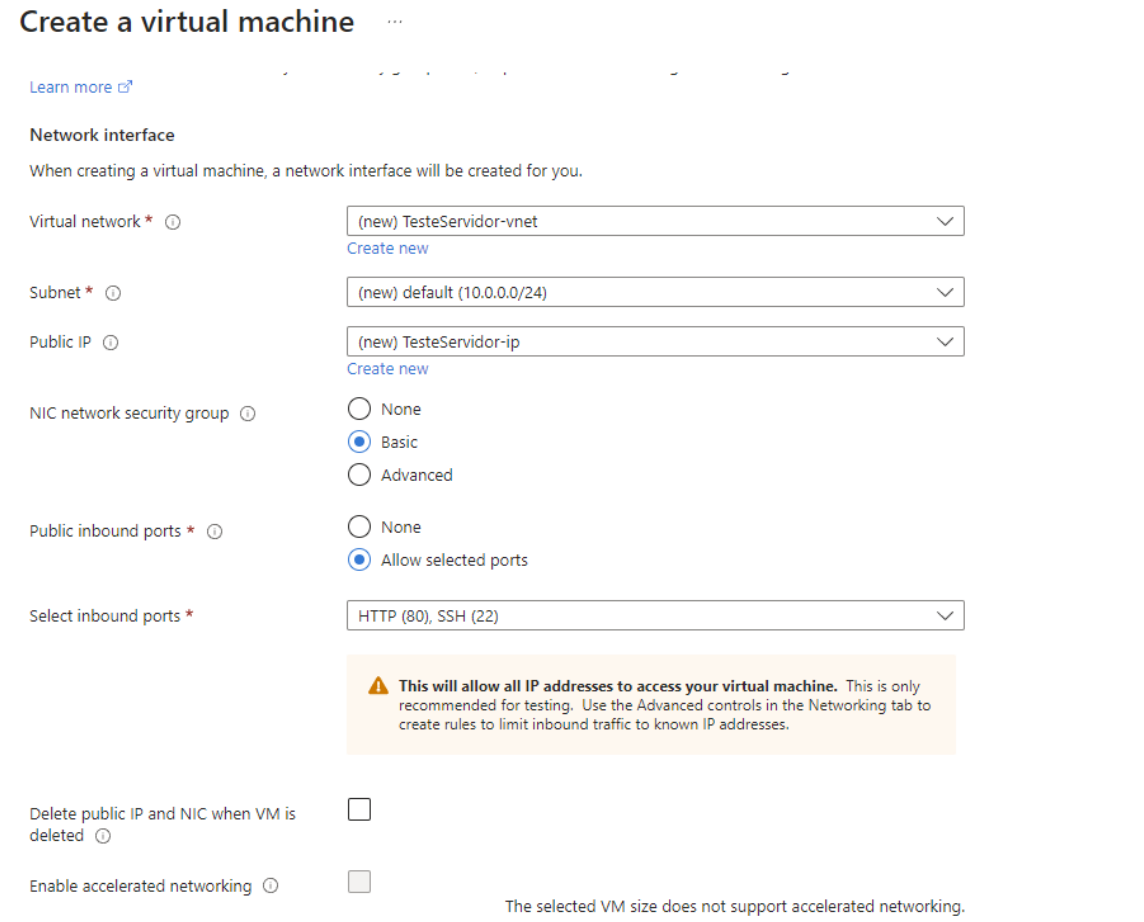
Em disks, deixei o que já estava por padrão:



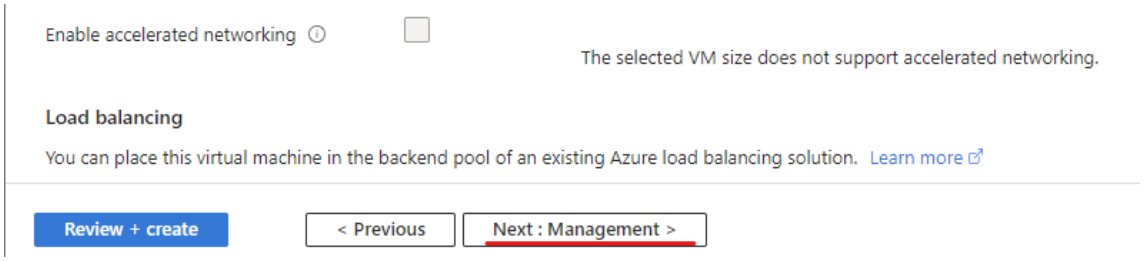
Após, selecione o botão Next: Networking:



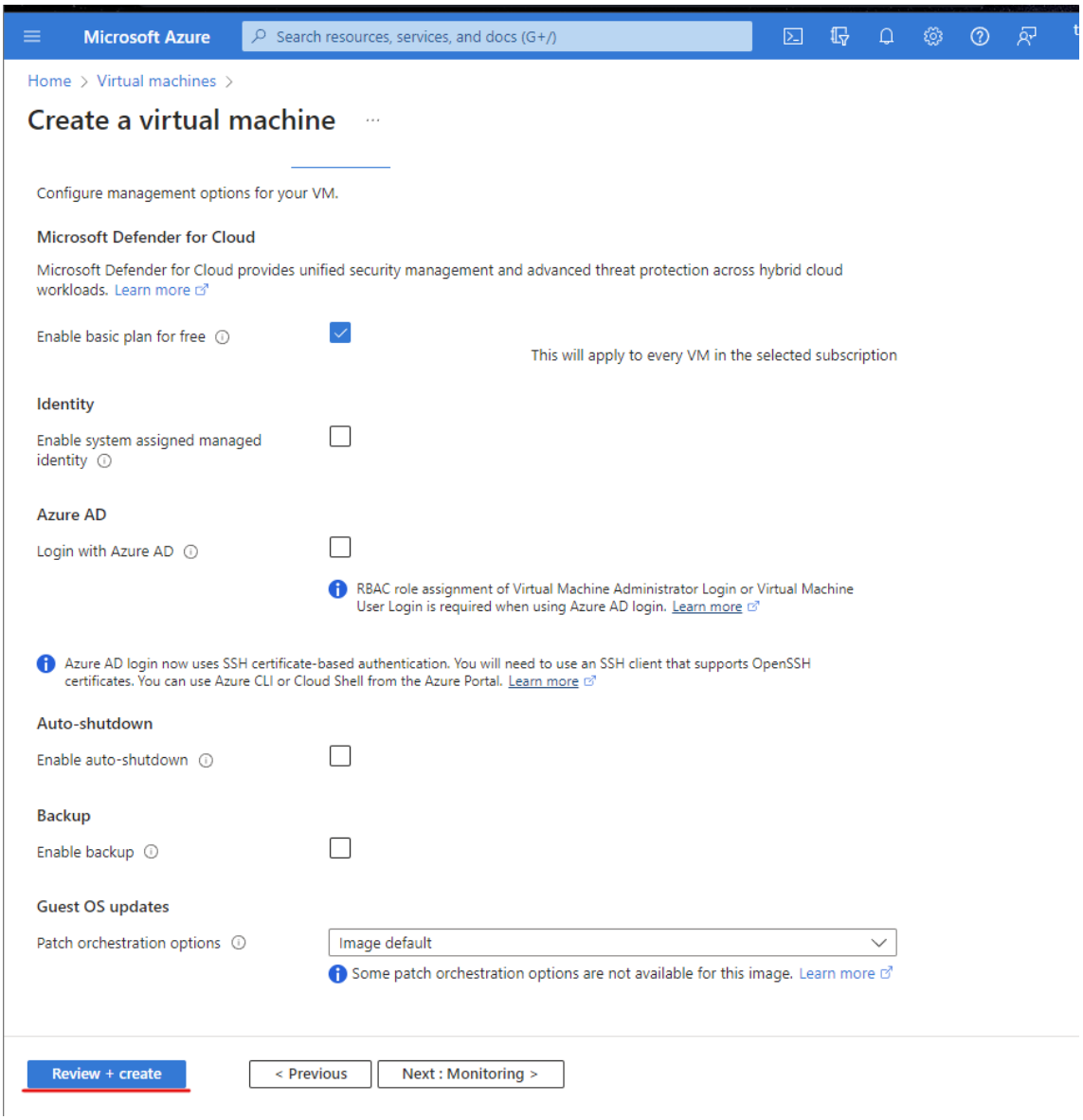
O qual vai criar uma nova virtual network, não será necessário alterar nada:



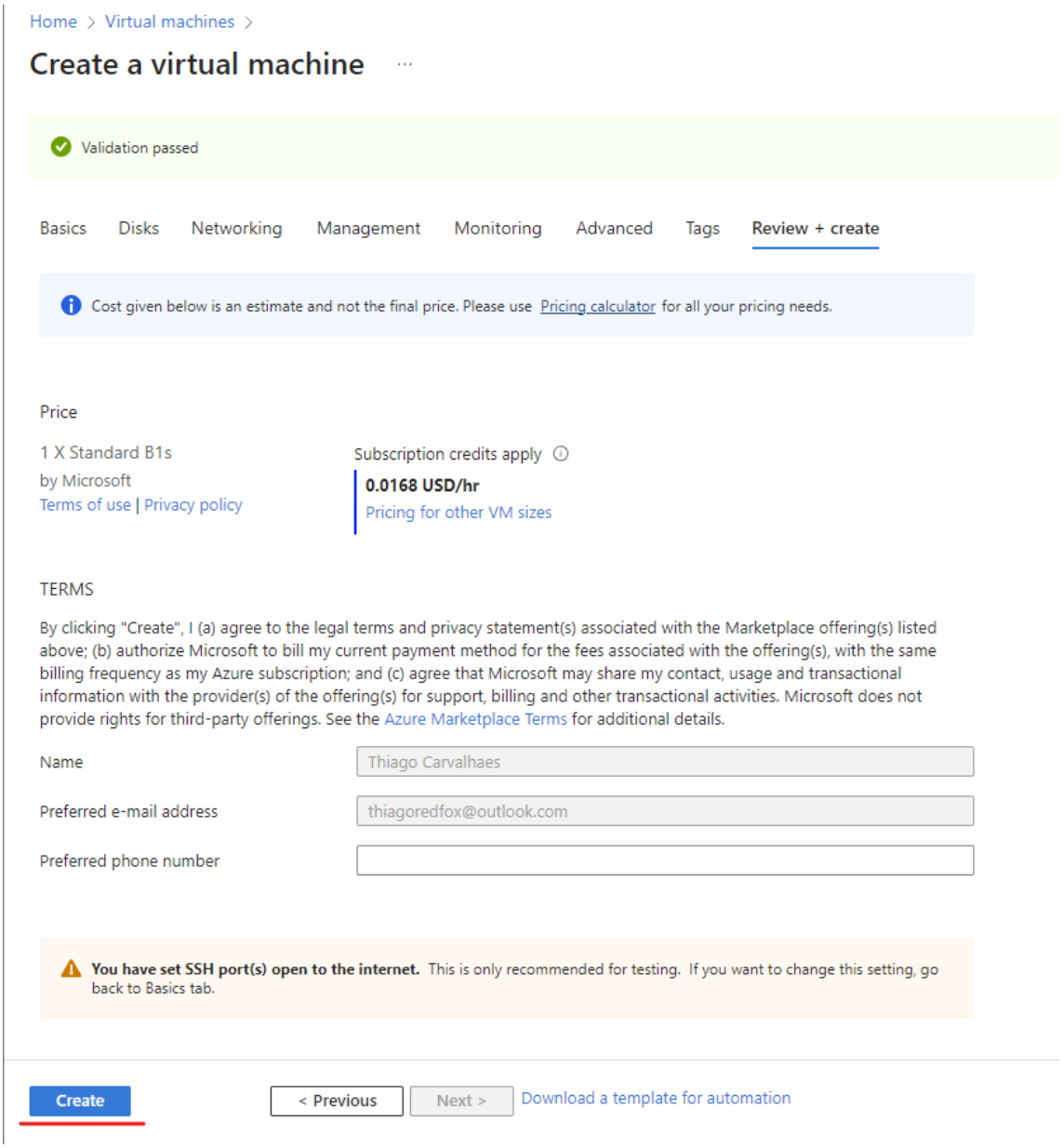
Seleciono o botão next: Monitoring



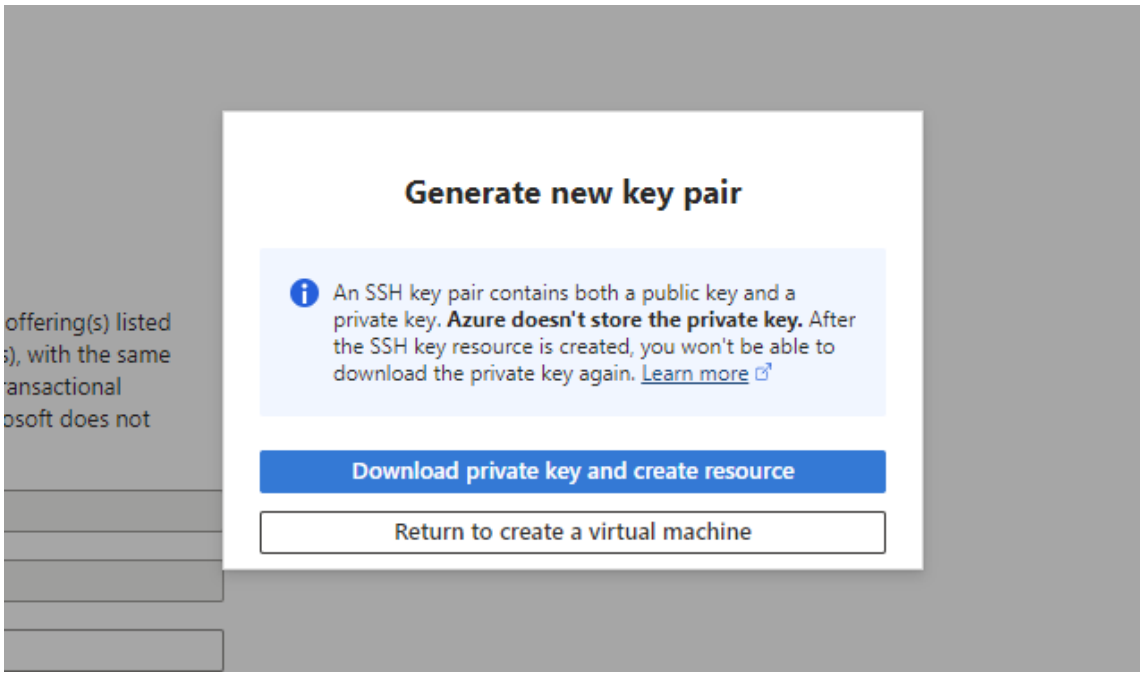
Em Managemente, selecione o botão review + create



Vai abri a seguinte tela:

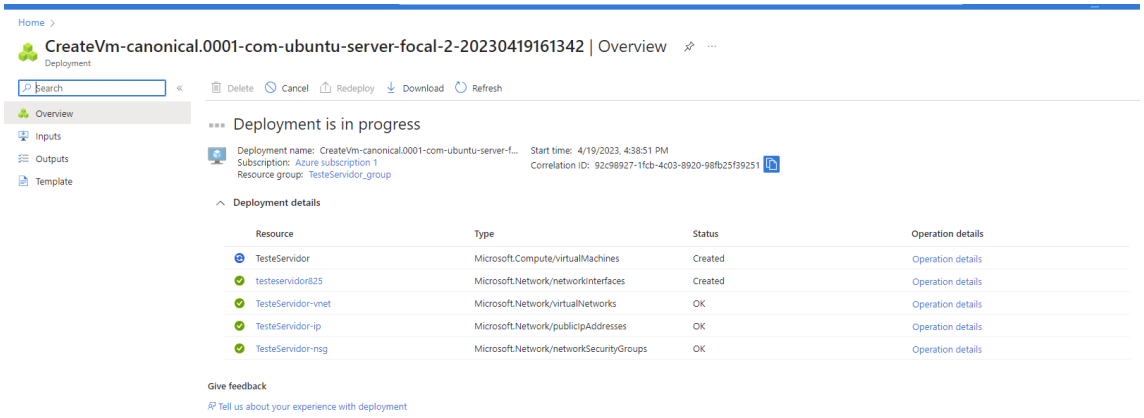


Selecione o botão create, e efetua o download da sua chave privada:

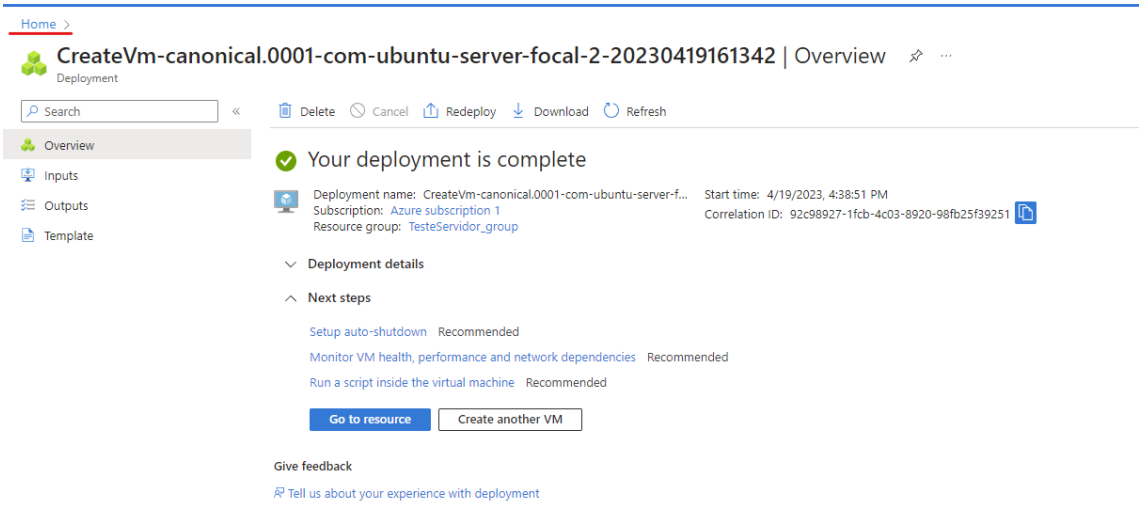


O qual vai efetuar o download de um arquivo .pem.

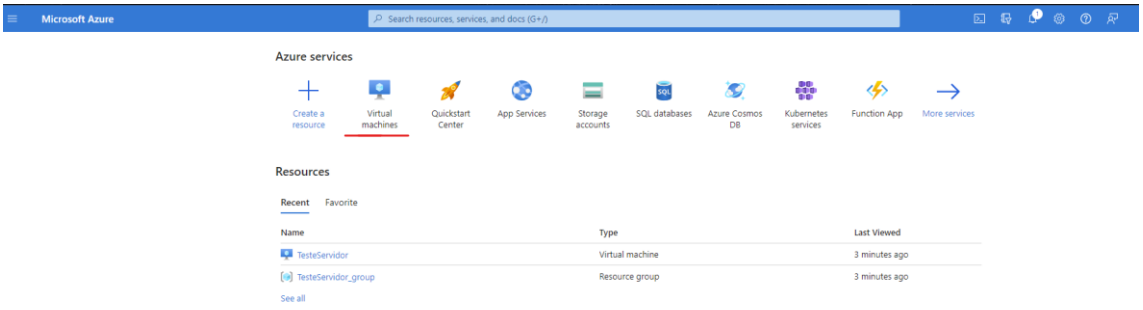
E vai iniciar o processo de criar o servidor:



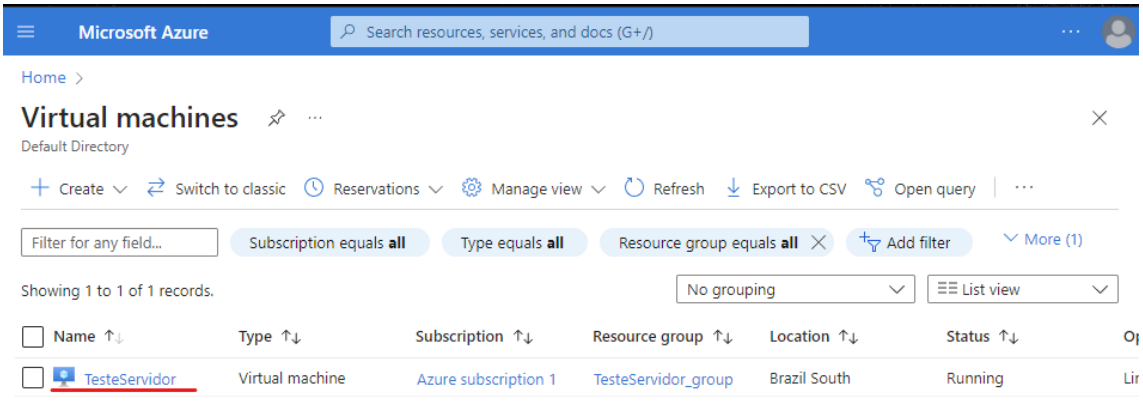
Após a conclusão da criação do servidor, selecione o botão home:



Virtual machines:

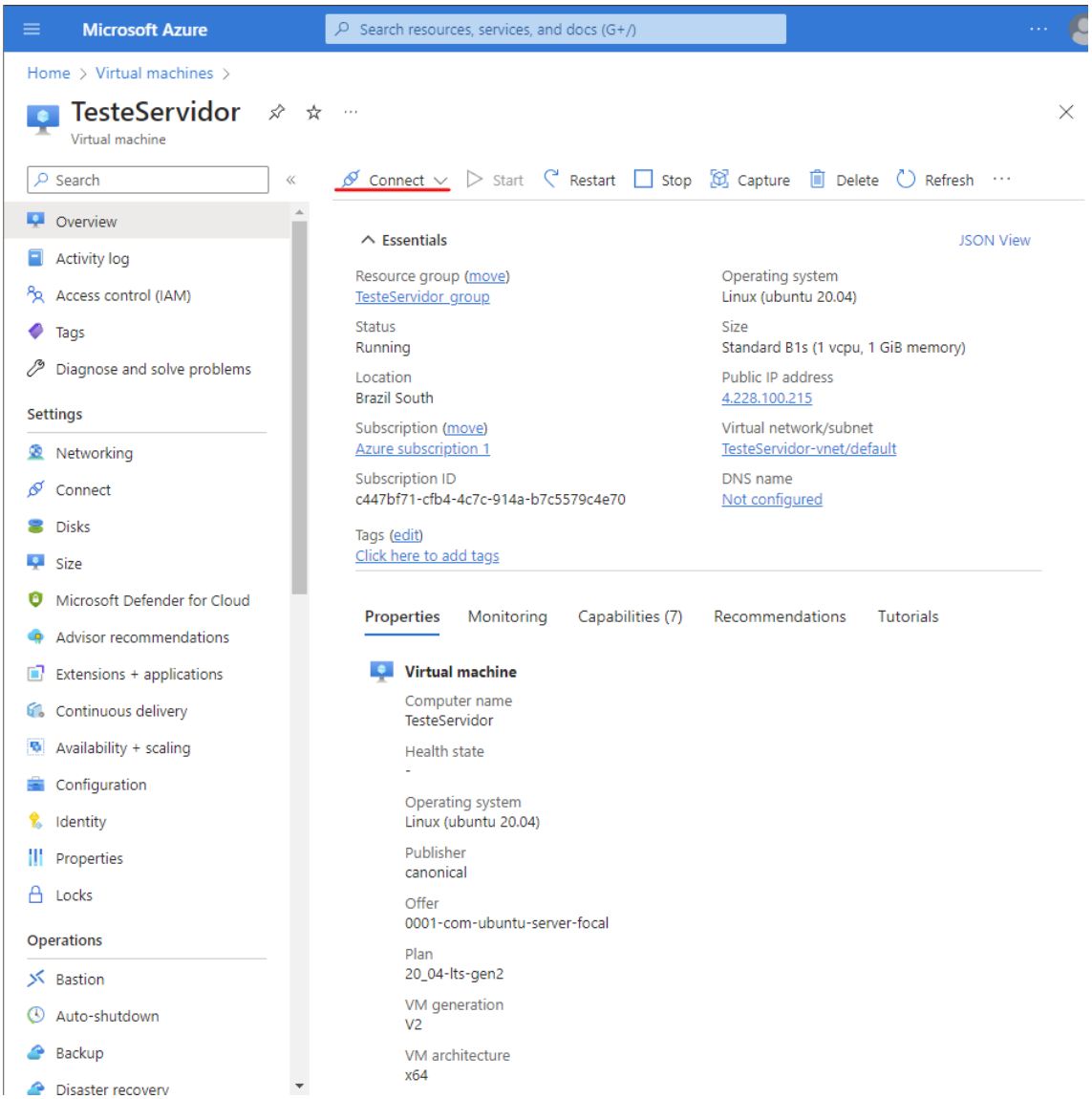


E vai aparecer o(s) servidor(es) que foi(foram) criado(s).

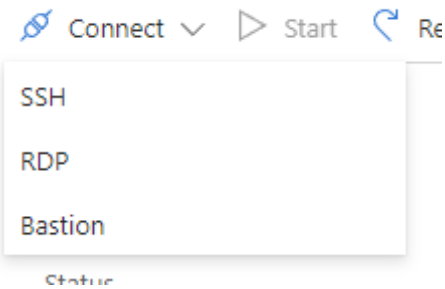


No meu caso, o TesteServidor.

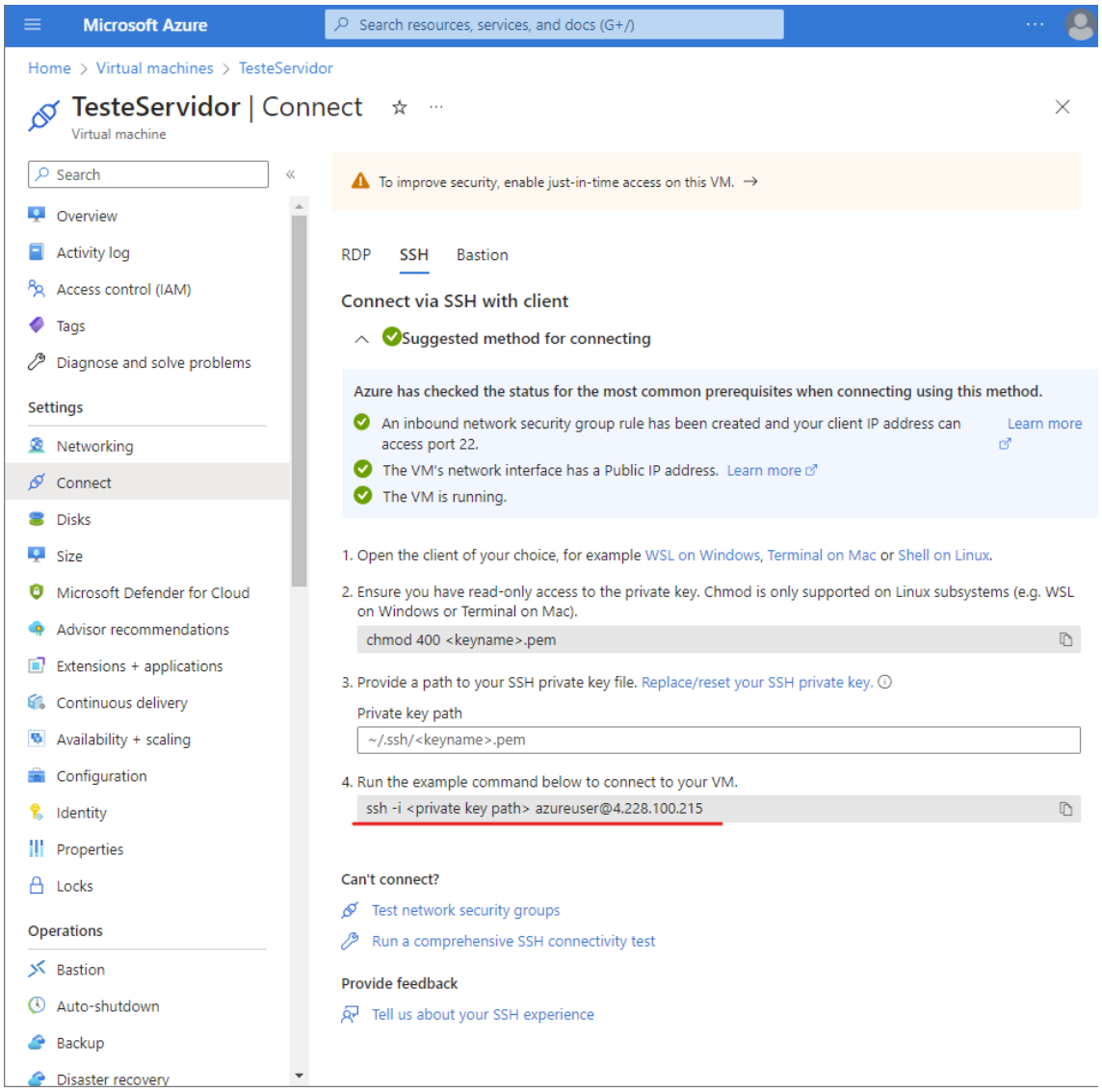
Ao selecionar o servidor vai aparecer a tela a seguir com os dados referente ao servidor:



Selecione a opção Connect, a qual vai apresentar as seguintes opções:

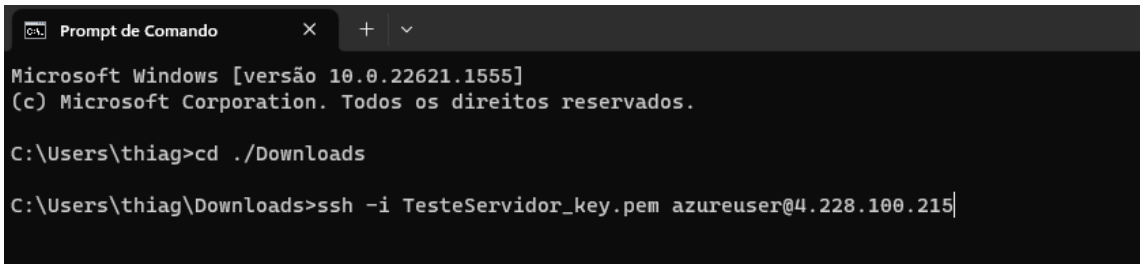


Selecione a opção SSH, e vai te apresentar informações de conexão, juntamente com um comando:



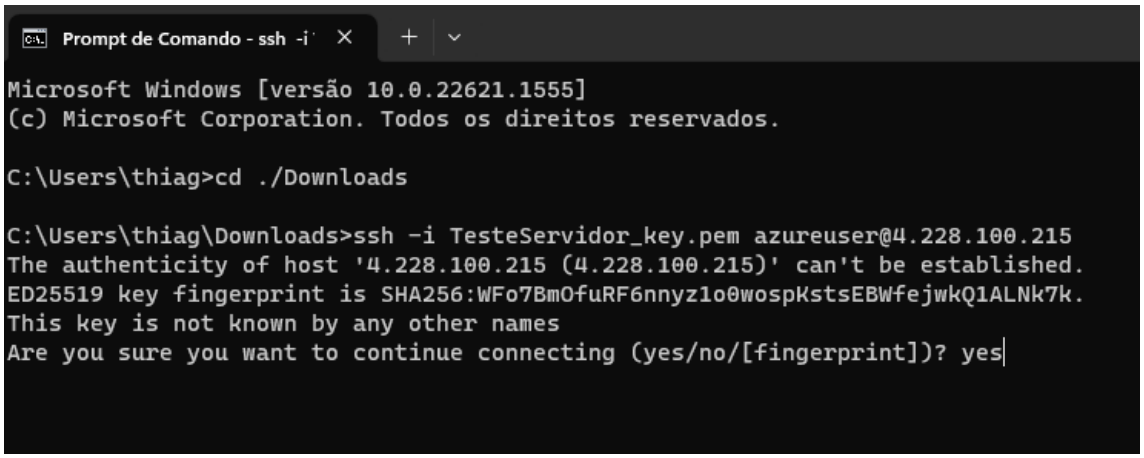
O qual será necessário informar a chave privada que foi gerada anteriormente.

Após isso, abra o prompt de comando de sua máquina e busque pelo diretório em que foi salvo a chave



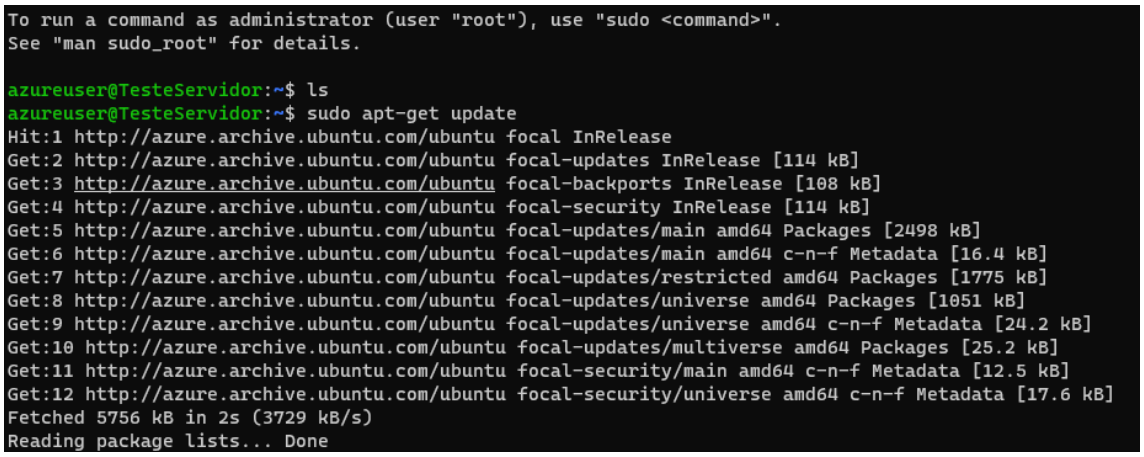
No meu caso eu deixei em Downloads.

Após pressionar a tecla enter, vai aparecer a seguinte pergunta:



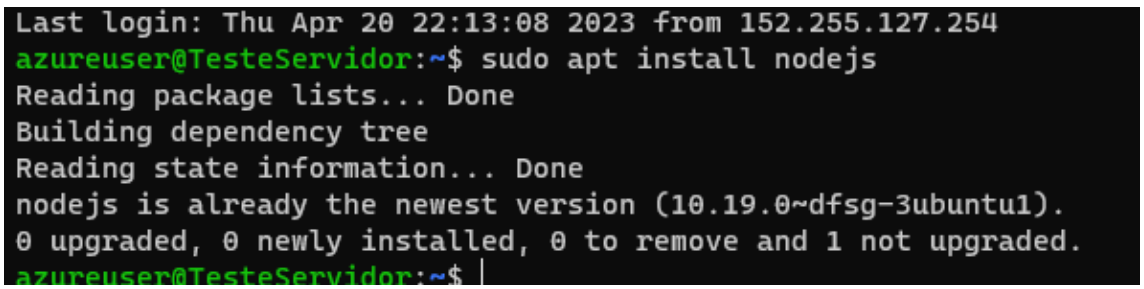
Responda como “yes” e vai conectar no servidor:

Ao conectar no servidor, rode o seguinte comando para atualizar:



sudo apt-get update

E o comando sudo apt install nodejs para instalar o Node.js:



Também será necessário estar instalando o Apache 2, com o seguinte comando:

sudo apt-get install apache2

```
azureuser@TesteServidor:~$ sudo apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-
  liblua5.2-0 ssl-cert
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
The following NEW packages will be installed:
```

Localize o diretório /var/www com o seguinte comando:

cd /var/www/

```
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
azureuser@TesteServidor:~$ cd /var/www/
azureuser@TesteServidor:/var/www$
```

E crie uma pasta para o projeto em Node:

sudo mkdir apinodejs

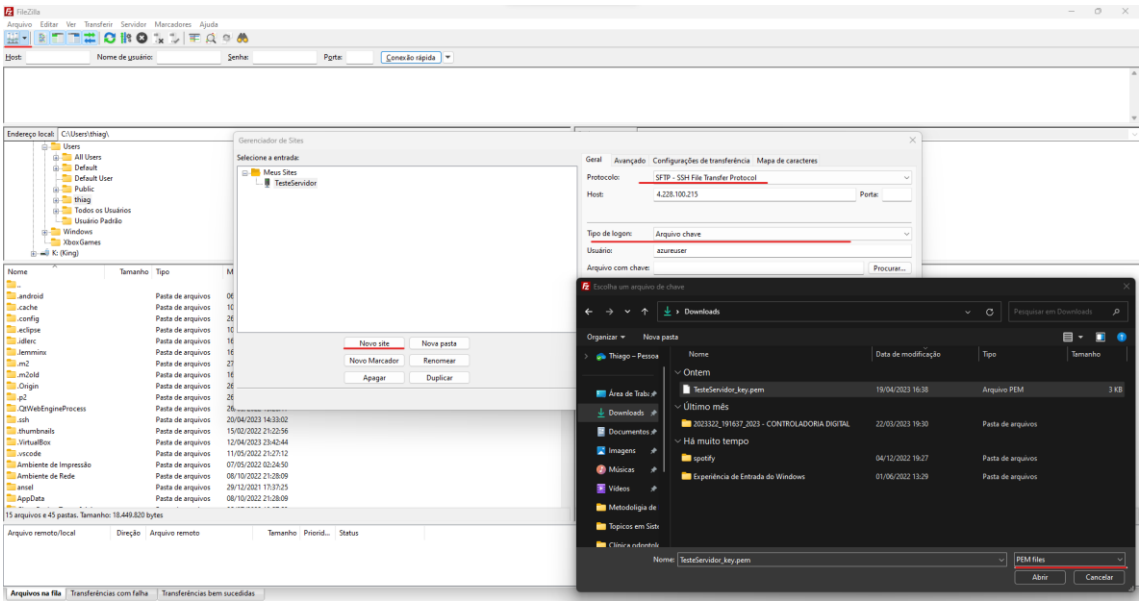
```
azureuser@TesteServidor:/var/www$ sudo mkdir apinodejs
azureuser@TesteServidor:/var/www$ ls -l
total 8
drwxr-xr-x 2 root root 4096 Apr 25 18:32 apinodejs
drwxr-xr-x 2 root root 4096 Apr 25 18:26 html
```

Libera a permissão total a pasta criada:

```
azureuser@TesteServidor:/var/www$ sudo chmod 777 apinodejs/
azureuser@TesteServidor:/var/www$ ls -l
total 8
drwxrwxrwx 2 root root 4096 Apr 25 18:32 apinodejs
```

sudo chmod 777 apinodejs/

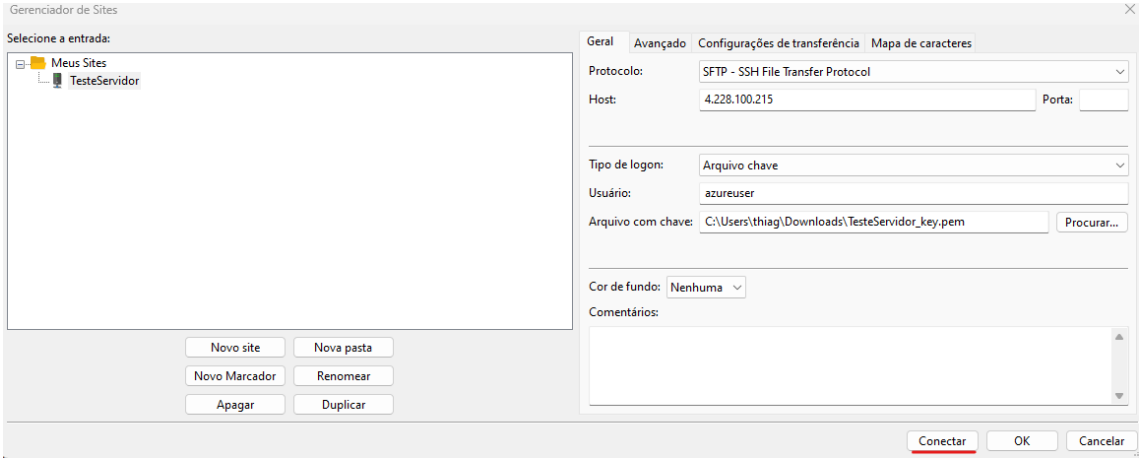
Para estar subindo os arquivos que vou utilizar na aplicação, vou estar utilizando o aplicativo FileZilla:



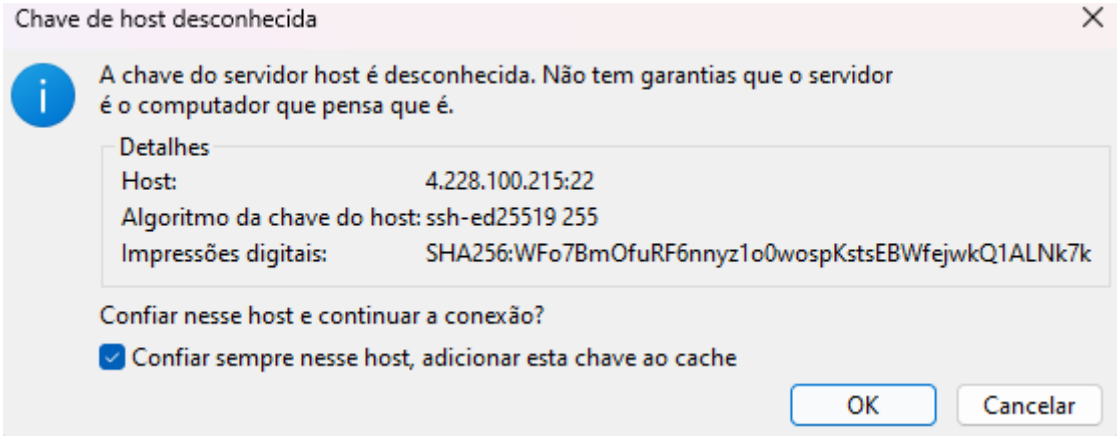
No FileZilla eu seleciono o botão superior esquerdo (destacado em vermelho) e clico e, novo site e o nomeio. Após nomear eu coloco as informações do servidor.

No Protocolo eu coloco o SFTP e em Tipo de Logon eu seleciono o tipo Arquivo de chave e localizo a chave que foi gerada a alguns passos atrás (Lembrando que no meu caso o arquivo da chave é com a extensão .pem, então no explorador de arquivos eu tive que colocar PEM files).

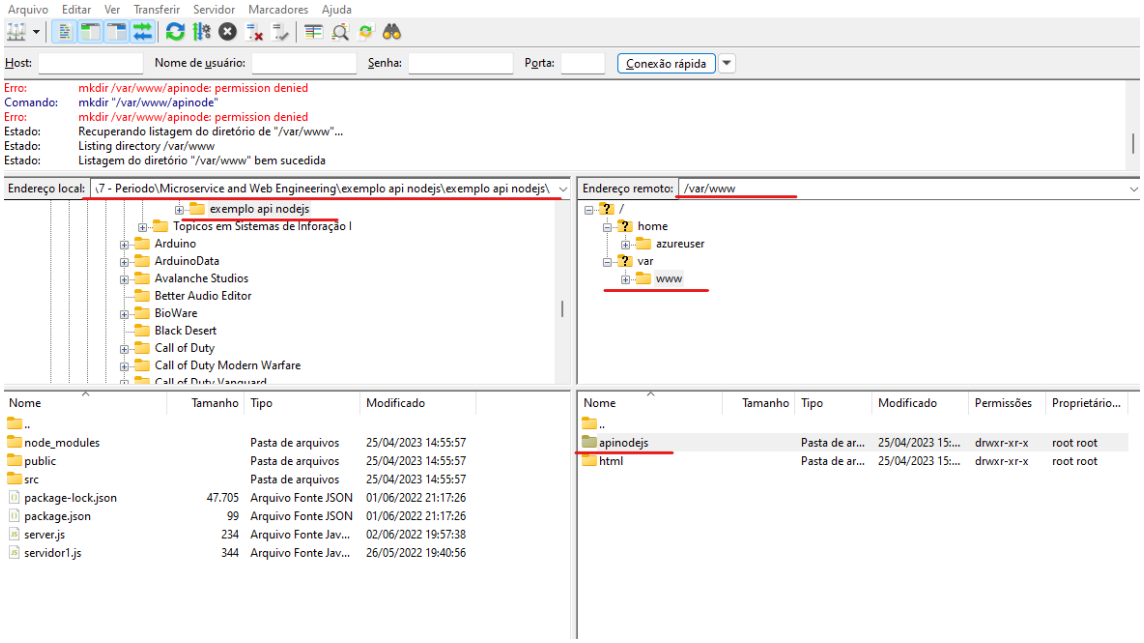
Após informar todos os dados do servidor eu clico em conectar:



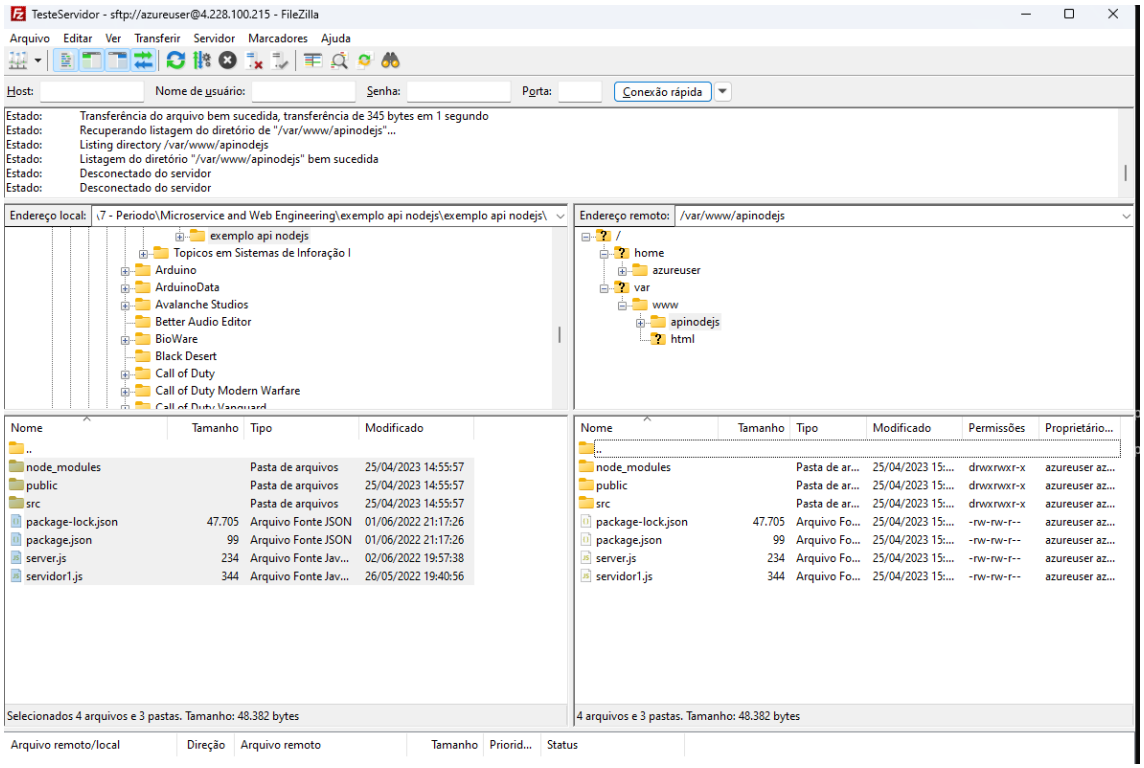
E marco a opção para confira sempre nesse host:



Localize a esquerda o Endereço local de onde está a aplicação que eu desejo subir para o servidor, selecione a pasta e na parte de baixo seleciono os arquivos:

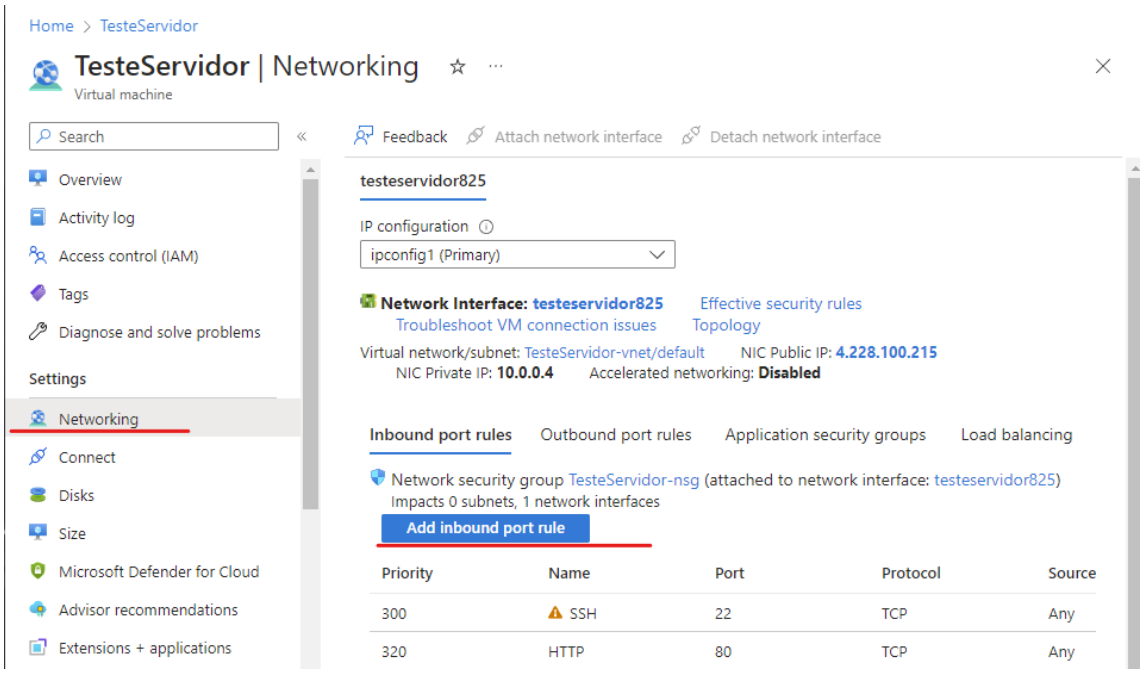


Na parte da direita do sistema, abra a pasta que foi criada anteriormente e arraste os arquivos de sua aplicação para ela.



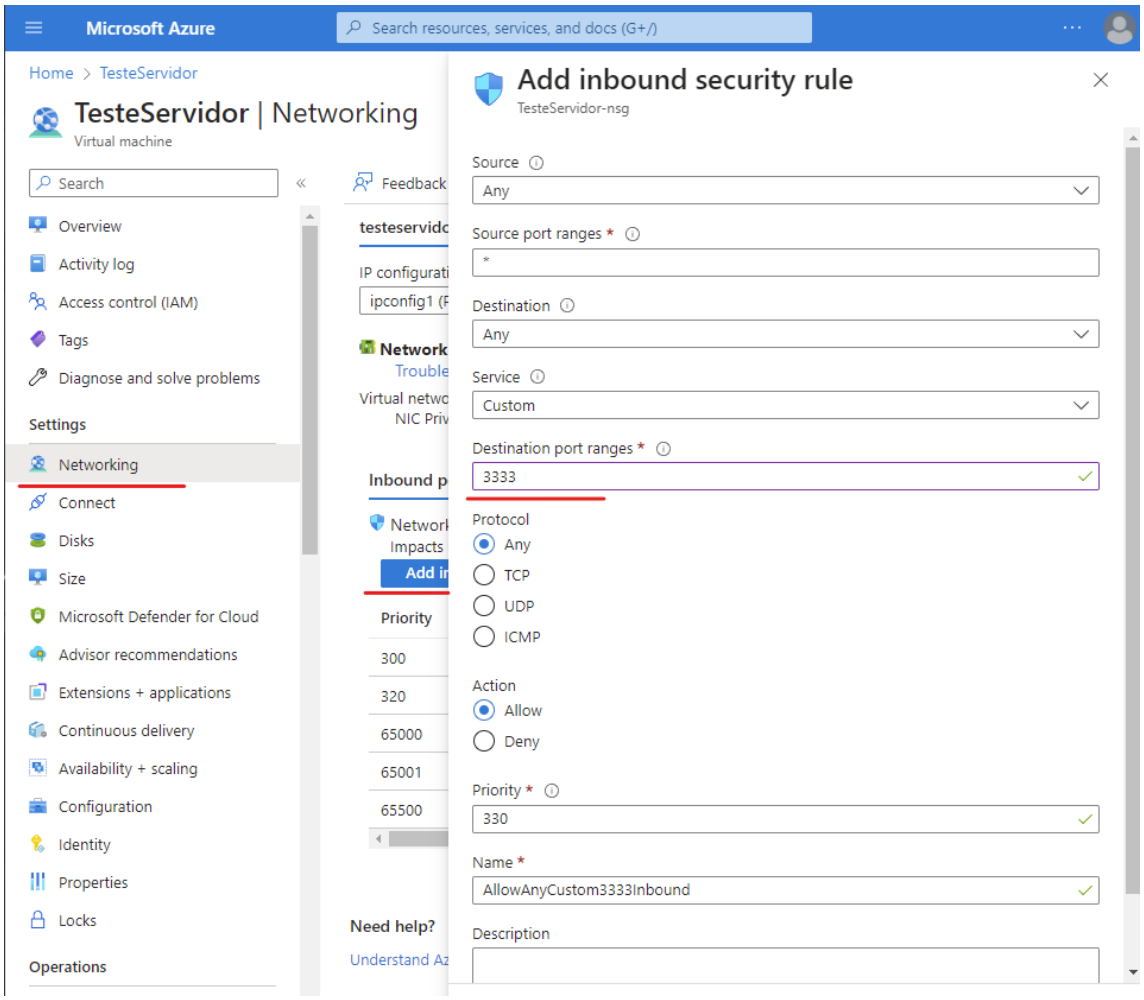
OBS:

NO meu caso tive que habilitar a porta 3333 do servidor:

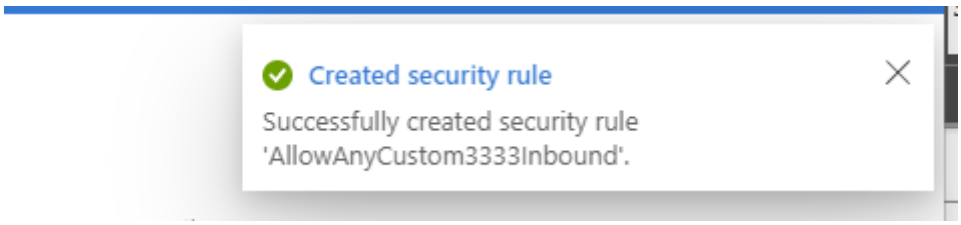


Neste caso, precisei ir às configurações do servidor no site da Azure.

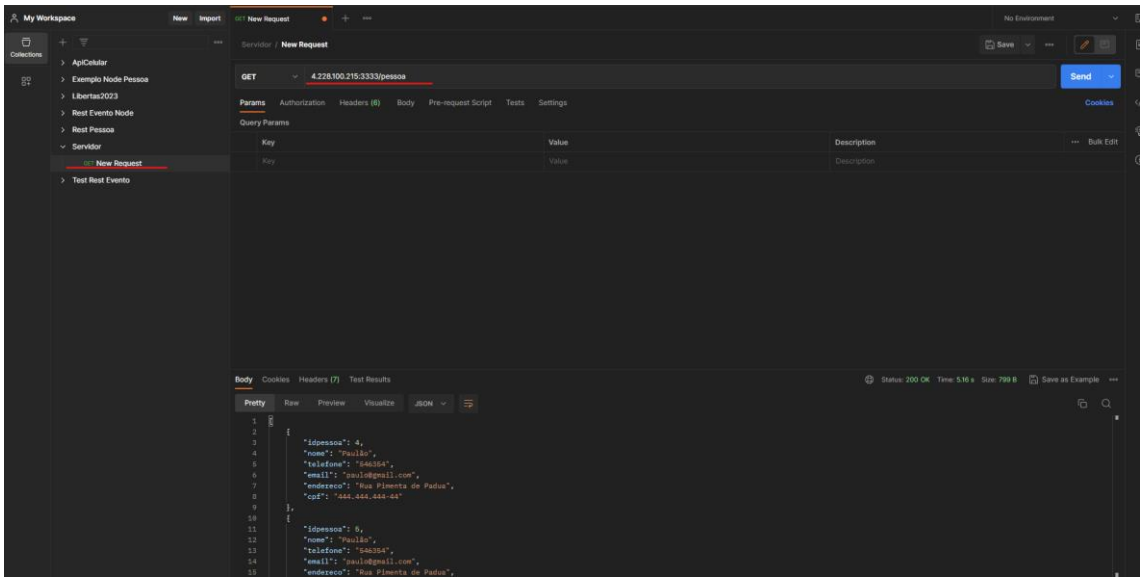
Em Networking, selecione o botão Add inbound port rule



Informe a senha e selecione o botão Add abaixo, assim que concluir vai aparecer essa mensagem de confirmação:

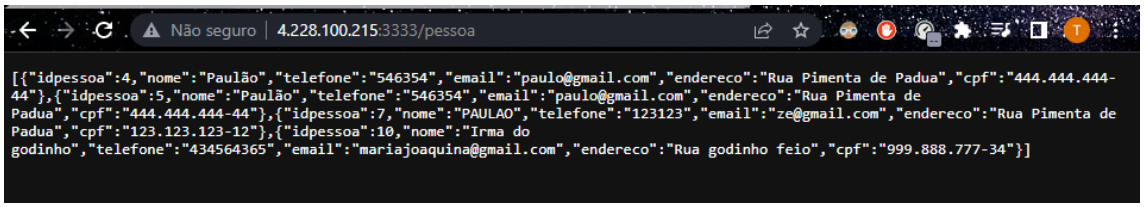


Para testar vou utilizar o aplicativo Postman:



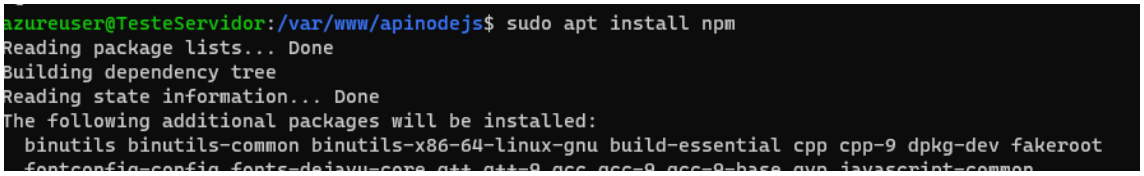
Ao verificar no Postman, foi possível visualizar as informações que já estava inserida no bando de dados a fim de testes e através do método get de minha aplicação, foi possível verificar que está totalmente funcional.

Caso não queira utilizar o Postman, pode estar utilizando o seu próprio navegador:

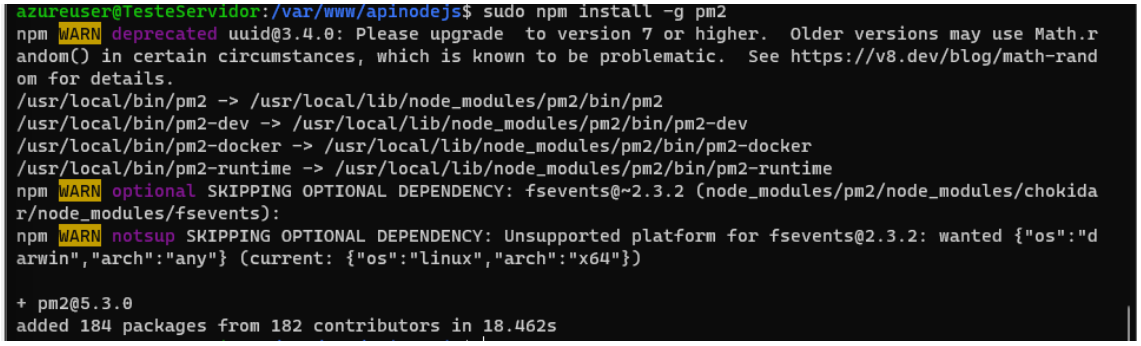


O seguinte passa é para efetuar as instalações necessárias para estar habilitando o autostart no caso de reinicialização da máquina:

sudo apt install npm



sudo npm install -g pm2



E execute o comando a seguir para iniciar o serviço:

pm2 start server.js

Com esse comando, já inicia o serviço e já está rodando a aplicação, no meu caso na porta 3333.

Com isso concluímos esse tutorial, uma vez que já temos nosso servidor e nossa aplicação online.