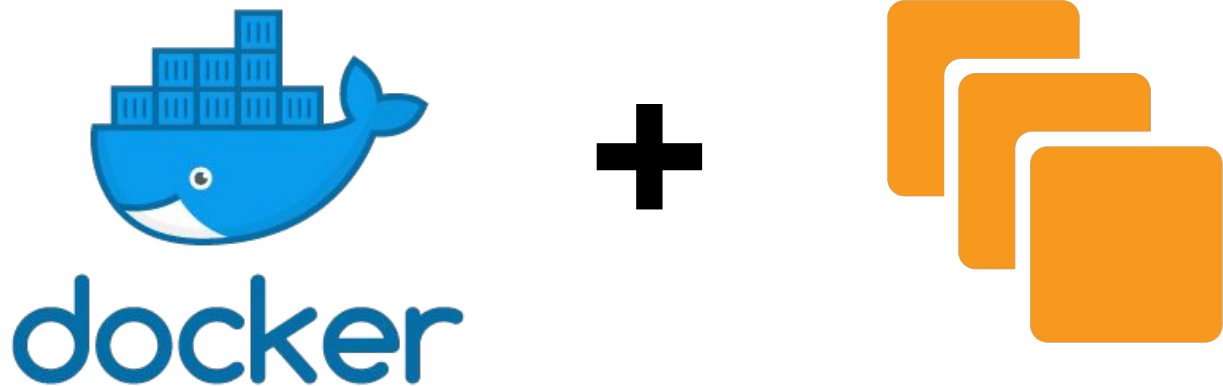


Laboratório

SO



docker **AWS**

Configurando o
ambiente

Sobre o material

Esse material foi elaborado para disciplina TT304 - Sistemas Operacionais.

Pelos monitores **Arthur G. S. Conti** e **Myrelle S. Lopes**, supervisionados pelo **Prof. Dr. André Leon S. Gradvohl**.

Esse material, tem como objetivo explicar para os alunos da disciplina como preparar o ambiente da **AWS**, para a utilização na aula!

Nesse material, vamos abordar a instalação do docker na **AWS**, do docker-compose, abrir as portas para acesso e configuração dos exemplos!

Esperamos que seja de grande ajuda e utilidade para vocês alunos e qualquer sugestão para a melhoria do material é bem-vinda!



Índice

- **Docker Essencial**
- **Docker Compose**
- **Abrindo portas**
- **Configurando o node**
- **Configurando o projeto**
- **Enviando projeto para a nuvem**



Docker Essencial

Para iniciarmos a instalação do Docker, atualize os pacotes instalados e o cache do pacote na instância:

- Os comandos a seguir devem ser executados como root (administrador), para isso usamos o sudo:

```
sudo yum update -y
```

Instale o pacote Docker Engine mais recente:

```
sudo amazon-linux-extras install docker
```

Inicie o Docker:

```
sudo service docker start
```

Para que não seja necessário utilizar o comando sudo junto aos comandos Docker adicione o ec2-user ao grupo docker:

```
sudo usermod -a -G docker ec2-user
```

Docker Essencial

Faço logout e login novamente para obter as novas permissões do docker. Você pode fazer isso fechando a janela do terminal SSH atual, ou usando o comando *logout*, e reconectando à sua instância em uma nova. Sua nova sessão SSH terá as permissões do docker apropriadas.

Você pode testar a execução do comando Docker sem o sudo:

```
docker info
```



Docker Compose

O Docker compose ainda não está disponível nos pacotes oficiais da amazon, então para isso, vamos precisar fazer a instalação “manual”, usando o [repositório oficial](#) do docker-compose no github, para isso, basta rodar o [seguinte comando](#):

```
sudo curl -L  
https://github.com/docker/compose/releases/latest/d  
ownload/docker-compose-$(uname -s)-$(uname -m)  
-o /usr/local/bin/docker-compose
```

Estamos instalando a última versão do compose, para podermos utilizar alguns recursos específicos nesse lab

Vamos alterar as permissões e permitir que o binário do docker-compose possa ser executado:

```
sudo chmod +x /usr/local/bin/docker-compose
```

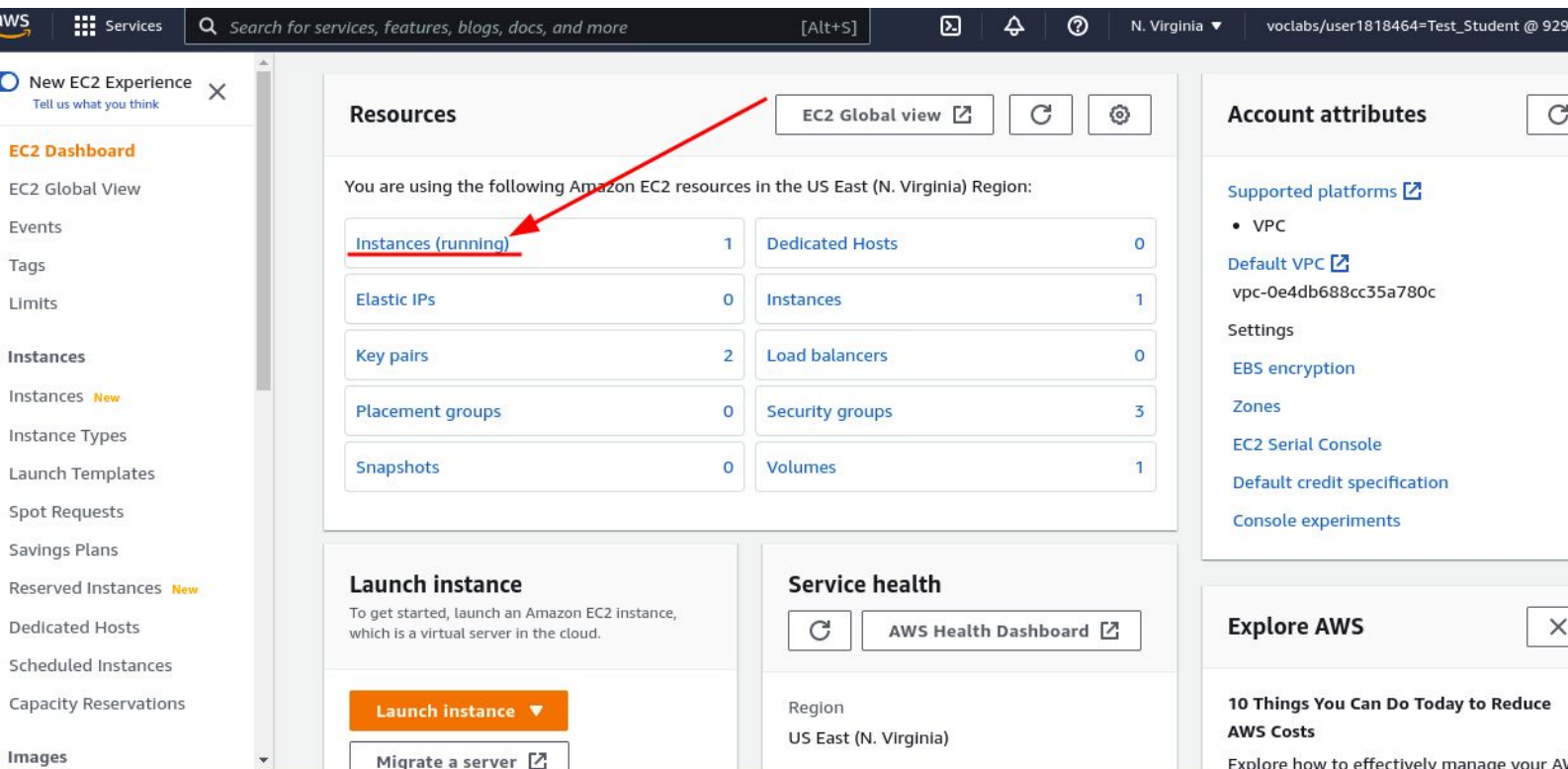
Após isso, vamos relogar na instância, como fizemos no [docker](#), e após relogar podemos ver se tudo deu certo com o comando:

```
docker-compose version
```

Abrindo portas

Como vamos subir uma api nos exemplos em salas, precisamos que essa api seja acessível de fora do container, para isso, precisamos abrir as portas dentro da AWS, e vamos fazer isso já agora, pois a atualização de quais portas estão abertas geralmente leva um tempinho!

Para fazer isso, acesse o dashboard da AWS e acesse a Dashboard do EC2 depois vá em Instances



The screenshot shows the AWS Management Console interface. On the left is a navigation sidebar with the 'EC2 Dashboard' selected. The main content area is titled 'Resources' and shows a table of EC2 resources in the 'US East (N. Virginia) Region'. A red arrow points to the 'Instances (running)' link, which is underlined. Below this, there are sections for 'Launch instance' and 'Service health'. The 'Launch instance' section has a 'Launch instance' button. The 'Service health' section shows the 'AWS Health Dashboard' and the current region 'US East (N. Virginia)'. On the right side, there are sections for 'Account attributes' and 'Explore AWS'.

Resources	
You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:	
Instances (running)	1
Elastic IPs	0
Key pairs	2
Placement groups	0
Snapshots	0
Dedicated Hosts	0
Instances	1
Load balancers	0
Security groups	3
Volumes	1

Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

[Launch instance](#) ▼

[Migrate a server](#)

Service health

[AWS Health Dashboard](#)

Region
US East (N. Virginia)

Account attributes

[Supported platforms](#)

- VPC

[Default VPC](#)
vpc-0e4db688cc35a780c

Settings

- [EBS encryption](#)
- [Zones](#)
- [EC2 Serial Console](#)
- [Default credit specification](#)
- [Console experiments](#)

Explore AWS

10 Things You Can Do Today to Reduce AWS Costs

Explore how to effectively manage your AWS

Abrindo portas

Após isso clique no Instance ID

Instances (1) Info		Refresh	Connect	Instance state ▼	Actions ▼	Launch instances	▼
Search		< 1 > ⚙					
Instance state = running X		Clear filters					
<input type="checkbox"/>	Name ▼	Instance ID	Instance state ▼	Instance type ▼	Status check	Alarm status	Availability Zo
<input type="checkbox"/>	teste docker	<u>i-08dc9e429e7da712b</u>	Running	t2.micro	2/2 checks passed	No alarms +	us-east-1c

Nessa tela temos alguns dados importantes, como nosso ipv4 público que será utilizado mais pra frente.

Instance summary for i-08dc9e429e7da712b (teste docker)

Updated less than a minute ago

Info

Refresh

Connect

Instance state ▼

Actions ▼

Instance ID

i-08dc9e429e7da712b (teste docker)

IPv6 address

–

Hostname type

IP name: ip-172-31-93-46.ec2.internal

Answer private resource DNS name

IPv4 (A)

Auto-assigned IP address

54.89.250.4 [Public IP]

IAM Role

–

Public IPv4 address

54.89.250.4 | open address

Instance state

Running

Private IP DNS name (IPv4 only)

ip-172-31-93-46.ec2.internal

Instance type

t2.micro

VPC ID

vpc-0e4db688cc35a780c

Subnet ID

subnet-04564cab4da0af280

Private IPv4 addresses

172.31.93.46

Public IPv4 DNS

ec2-54-89-250-4.compute-1.amazonaws.com

| open address

Elastic IP addresses

–

AWS Compute Optimizer finding

Opt-in to AWS Compute Optimizer for recommendations.

| Learn more

Auto Scaling Group name

–

Details

Security

Networking

Storage

Status checks

Monitoring

Tags

Abrindo portas

Para dar sequência, clicamos na aba security

The screenshot shows the AWS Management Console interface. The left sidebar contains navigation links for EC2 Dashboard, EC2 Global View, Events, Tags, Limits, Instances, and Images. The main content area displays the 'Instance summary for i-08dc9e429e7da712b (teste docker)'. The instance is in a 'Running' state. A red arrow points to the 'Security' tab in the bottom navigation bar.

Instance ID	Public IPv4 address	Private IPv4 addresses
i-08dc9e429e7da712b (teste docker)	54.89.250.4 open address	172.31.93.46

Instance state	Private IP DNS name (IPv4 only)	Public IPv4 DNS
Running	ip-172-31-93-46.ec2.internal	ec2-54-89-250-4.compute-1.amazonaws.com open address

Hostname type	Instance type	Elastic IP addresses
IP name: ip-172-31-93-46.ec2.internal	t2.micro	-

Answer private resource DNS name	VPC ID	AWS Compute Optimizer finding
IPv4 (A)	vpc-0e4db688cc35a780c	Opt-in to AWS Compute Optimizer for recommendations. Learn more

Auto-assigned IP address	Subnet ID	Auto Scaling Group name
54.89.250.4 [Public IP]	subnet-04564cab4da0af280	-

IAM Role
-

Nessa tela, visualizamos todos os perfis de segurança e podemos editar suas regras

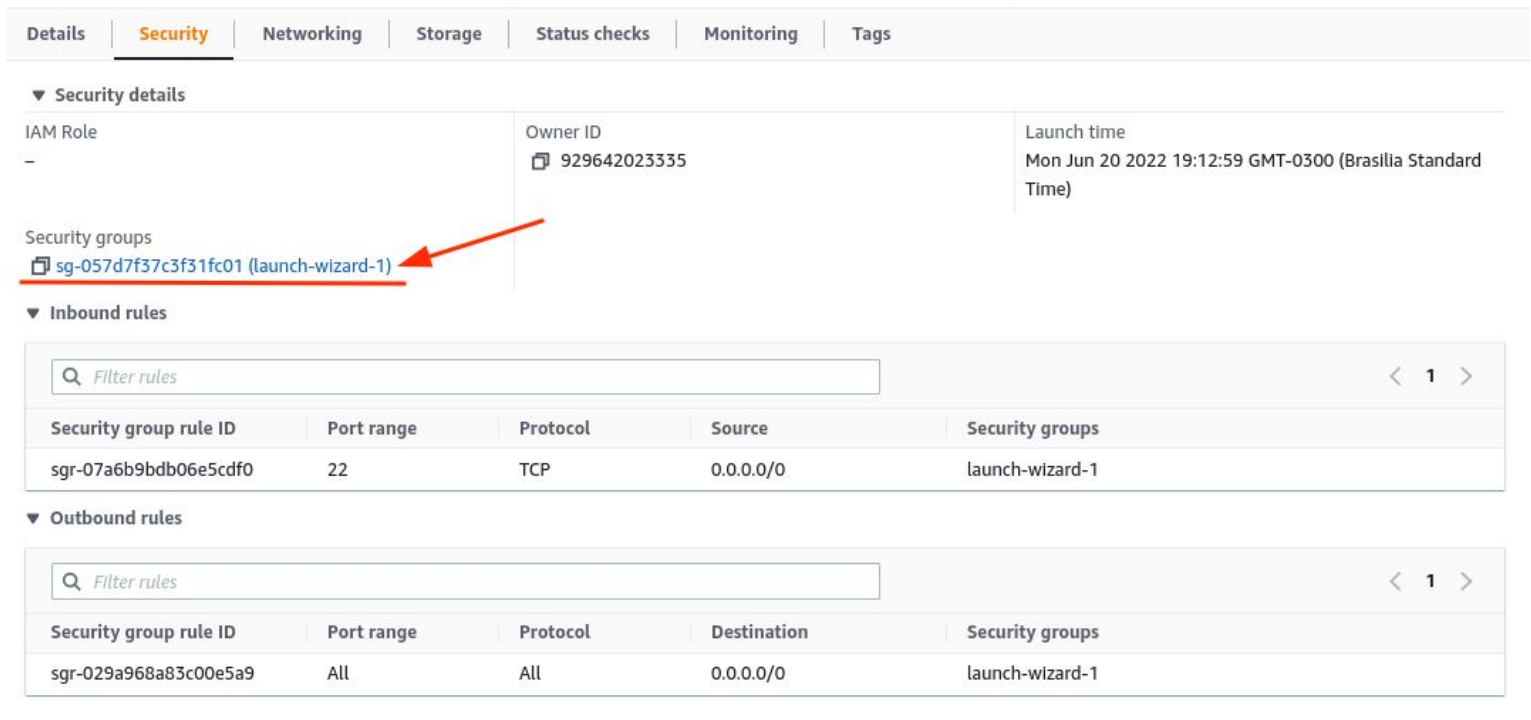
The screenshot shows the 'Security' tab of the instance summary. It displays the 'Security details' section, including the IAM Role, Subnet ID, and Auto Scaling Group name. The 'Inbound rules' and 'Outbound rules' sections are highlighted with red and blue boxes respectively.

Security group rule ID	Port range	Protocol	Source	Security groups
sgr-07a6b9bdb06e5cdf0	22	TCP	0.0.0.0/0	launch-wizard-1

Security group rule ID	Port range	Protocol	Destination	Security groups
sgr-029a968a83c00e5a9	All	All	0.0.0.0/0	launch-wizard-1

Abrindo portas

Para alterarmos as regras e assim abrir as portas necessárias, clicamos no security group



Details | **Security** | Networking | Storage | Status checks | Monitoring | Tags

▼ Security details

IAM Role
-

Owner ID
929642023335

Launch time
Mon Jun 20 2022 19:12:59 GMT-0300 (Brasilia Standard Time)

Security groups
[sg-057d7f37c3f31fc01 \(launch-wizard-1\)](#)

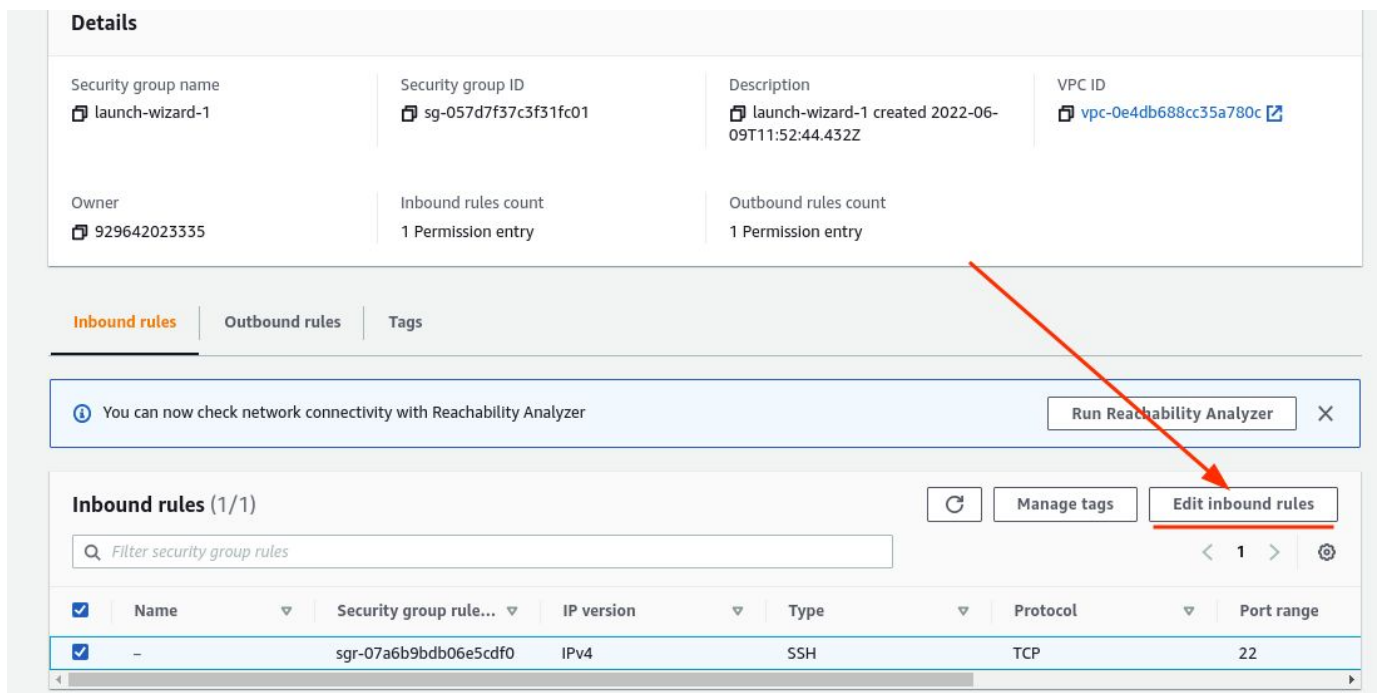
▼ Inbound rules

Security group rule ID	Port range	Protocol	Source	Security groups
sgr-07a6b9bdb06e5cdf0	22	TCP	0.0.0.0/0	launch-wizard-1

▼ Outbound rules

Security group rule ID	Port range	Protocol	Destination	Security groups
sgr-029a968a83c00e5a9	All	All	0.0.0.0/0	launch-wizard-1

Nessa tela, teremos os detalhes do nosso Security Group, vamos em baixo em “Edit inbound rules”



Details

Security group name
launch-wizard-1

Security group ID
sg-057d7f37c3f31fc01

Description
launch-wizard-1 created 2022-06-09T11:52:44.432Z

VPC ID
vpc-0e4db688cc35a780c

Owner
929642023335

Inbound rules count
1 Permission entry

Outbound rules count
1 Permission entry

Inbound rules | Outbound rules | Tags

You can now check network connectivity with Reachability Analyzer [Run Reachability Analyzer](#)

Inbound rules (1/1) [Manage tags](#) [Edit inbound rules](#)

<input checked="" type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range
<input checked="" type="checkbox"/>	-	sgr-07a6b9bdb06e5cdf0	IPv4	SSH	TCP	22

Abrindo portas

Para o nosso exemplo, precisaremos abrir 3 portas:

- MYSQL/Aurora porta 3306
- HTTP porta 80
- Custom TCP: 3333

A porta Mysql e a HTTP estão pré configurada pela amazon, basta dizer que queremos abrir elas e colocar o source como "Anywhere-ipv4".

para fazer isso clicamos em "add rule" e depois configuramos as portas

The screenshot shows the 'Inbound rules' configuration page in the AWS IAM console. The page has a table of existing rules and an 'Add rule' button. Three red circles with arrows indicate the steps to add a new rule:

1. Click the 'Add rule' button.
2. Select 'MYSQL/Aurora' in the 'Type' dropdown menu.
3. Select 'Anywhere-IPv4' in the 'Source' dropdown menu.

The table shows the following rules:

Security group rule ID	Type	Protocol	Port range	Source	Description - optional	Info
sgr-07a6b9bdb06e5cdf0	SSH	TCP	22	Custom	0.0.0.0/0	Delete
-	MYSQL/Aurora	TCP	3306	Anywhere-IPv4	0.0.0.0/0	Delete

Como podemos ver, essa porta já vem pré configurada, precisamos apenas liberar, porém a porta 3333, precisamos configurar, basta fazer da seguinte forma:

Abrindo portas

Inbound rules [Info](#)

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info		
sgr-07a6b9bdb06e5cdf0	SSH	TCP	22	Custom	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		
-	MySQL/Aurora	TCP	3306	Anywh...	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		
-	HTTP	TCP	80	Anywh...	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		
-	Custom TCP	TCP	3333	Anywh...	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		

1 2 3 4

Após isso, clicamos em “Save rules” e as portas já estarão abertas

Inbound rules [Info](#)

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info		
sgr-07a6b9bdb06e5cdf0	SSH	TCP	22	Custom	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		
-	MySQL/Aurora	TCP	3306	Anywh...	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		
-	HTTP	TCP	80	Anywh...	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		
-	Custom TCP	TCP	3333	Anywh...	<input type="text" value="Q"/>	<input type="text"/>	<input type="button" value="Delete"/>
					<input type="text" value="0.0.0.0/0"/>		

Configurando o node

Para o nosso exemplo prático, precisaremos compilar nossa interface, ela está desenvolvida em Vuejs, para isso precisamos do node instalado na máquina, nesse laboratório, usaremos o node instalado pelo asdf, e para isso precisamos executar os instaladores tanto do asdf quanto do node!

Para isso baixem os scripts e deem de execução para eles usando os seguintes comandos:

```
chmod +x ./installASDF  
chmod +x ./installNode
```

Na sequência, executem o installASDF usando `./installASDF`

Feche e abra novamente o terminal, e execute o installNode:

```
./installNode
```

Pronto o node já está instalado e pronto para ser usado!

Configurando o projeto

Para o nosso exemplo prático, utilizaremos uma api, um banco e uma tela de aplicação, para essa tela, precisamos alterar o endereço para as requisições e fazer o build final!

Para isso, copiamos dentro da pasta do projeto o arquivo `".env.example"` e renomeamos para `".env"`, podemos fazer isso através do comando `cp`

```
cp ./.env.example ./.env
```

Após isso, podemos abrir o `.env` com o nano, e onde está `"baseUrl"` colocamos nosso ipv4 público da Amazon.

Após isso, podemos fazer o build do projeto, para isso rodamos na pasta raiz do projeto, frontend-vue, o seguinte comando:

```
npm install  
npm run build
```

Após isso, nosso projeto já está configurado e pronto para rodar

Enviando projeto para a nuvem

Para enviarmos nosso projeto para nuvem existem diversas formas, hoje, nós usaremos o scp que é um comando para copiar arquivos de forma remota.

Antes de enviarmos, vamos deletar a node_modules da frontend-vue, usamos o comando:

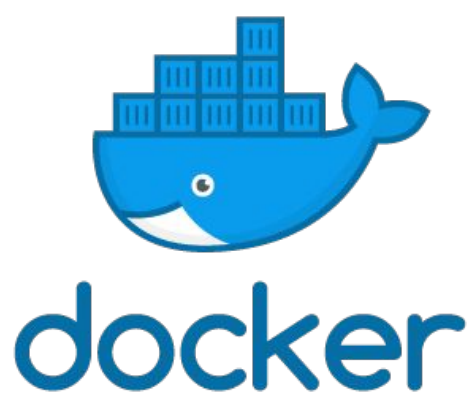
```
rm -r node_modules
```

Para isso usamos o seguinte comando:

```
scp -i ~/chave.pem -r passadoProjeto  
ec2-user@seu-ip:~/pastaDestino
```

Dessa forma nosso projeto, já está na AWS e podemos executar ele.

Decorative geometric shapes consisting of two overlapping triangles, one orange and one blue, located at the bottom of the page.



+



docker **AWS**