

Integração entre Jenkins, Terraform, Github e AWS

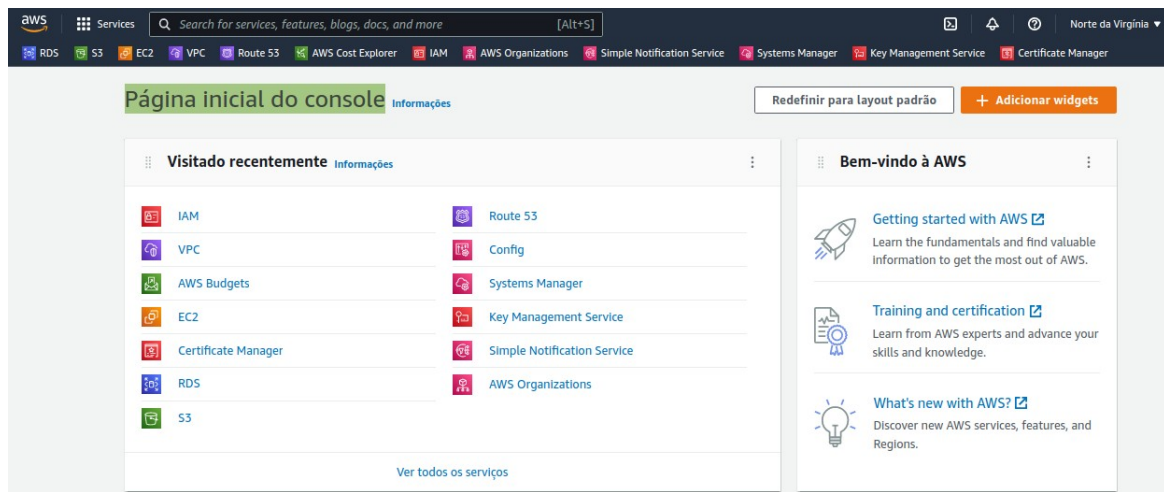
Antes de começarmos é necessário que você tenha uma conta gratuita na AWS e tenha Sua VPC criada. Vou deixar links para instrução de como fazer isso, caso você ainda não tenha feito.

[Criar Conta na AWS](#)

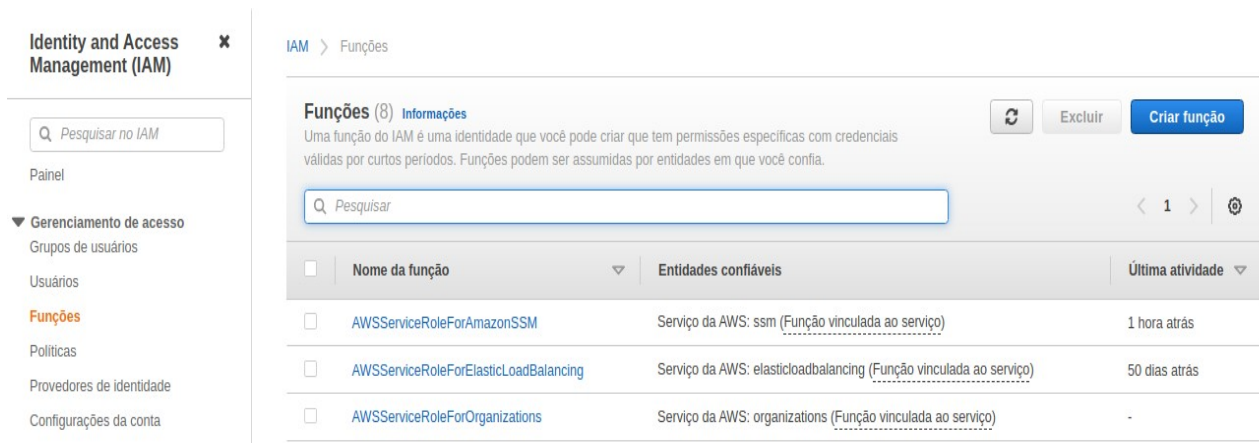
[Criar VPC](#)

Após

Acesse a Página inicial do console da sua conta e selecione o serviço de IAM.



Dentro do Painel do IAM vamos criar uma Função(Role), que será anexada a nossa instância do servidor Jenkins. Essa função terá um acesso temporário.



Clique em Criar função, marque a opção Serviço da AWS e em Caso de Uso escolha a opção EC2 e clique no botão Próximo

Selecionar entidade confiável

Tipo de entidade confiável

<input checked="" type="radio"/> Serviço da AWS Permitir que serviços da AWS, como o EC2, Lambda ou outros executem ações nessa conta.	<input type="radio"/> Conta da AWS Permitir que entidades em outras contas da AWS pertencentes a você ou a terceiros executem ações nessa conta.	<input type="radio"/> Identidade Web Permite que os usuários federados pelo provedor de identidade da Web externo especificado assumam essa função para executar ações nessa conta.
<input type="radio"/> Federação SAML 2.0 Permitir que os usuários federados com o SAML 2.0 de um diretório corporativo executem ações nessa conta.	<input type="radio"/> Política de confiança personalizada Crie uma política de confiança personalizada para permitir que outras pessoas executem ações nessa conta.	

Caso de uso

Permitir que um serviço da AWS, como o EC2, o Lambda ou outros executem ações nessa conta.

Casos de uso comuns

- EC2**
Allows EC2 instances to call AWS services on your behalf.
- Lambda**
Allows Lambda functions to call AWS services on your behalf.

Casos de uso para outros serviços da AWS:

Escolha um serviço para visualizar o caso de uso

Cancelar

Próximo

Agora vamos escolher as Políticas de Permissões para nossa instância. Didaticamente vamos escolher algumas políticas com acesso total. **Não é aconselhável reproduzir esse laboratório em um ambiente de produção.**

Adicionar permissões

Políticas de permissões (Selecionados 2/754)
Escolha uma ou mais políticas para anexar à sua nova função.

Q Filtre políticas por propriedade ou nome de política e pressione Enter 34 correspondências < 1 2 > ⚙

"admin" ✕ Limpar filtros

	Nome da política	Tipo	Descrição
<input type="checkbox"/>	AWSSSODirectoryAdministrator	Gerenci...	Administrator access for SSO Directory
<input type="checkbox"/>	CloudWatchAgentAdminPolicy	Gerenci...	Full permissions required to use AmazonCloudWatchAgent.
<input type="checkbox"/>	DatabaseAdministrator	Gerenci...	Grants full access permissions to AWS services and actions required to ...
<input type="checkbox"/>	AWSSSOMasterAccountAdministrator	Gerenci...	Provides access within AWS SSO to manage AWS Organizations mast...
<input type="checkbox"/>	AWSCloud9Administrator	Gerenci...	Provides administrator access to AWS Cloud9.
<input type="checkbox"/>	AWSSSOMemberAccountAdministrator	Gerenci...	Provides access within AWS SSO to manage AWS Organizations mem...
<input type="checkbox"/>	SystemAdministrator	Gerenci...	Grants full access permissions necessary for resources required for app...
<input checked="" type="checkbox"/>	AdministratorAccess	Gerenci...	Provides full access to AWS services and resources.

A política **AdministratorAccess**, garante acesso total a todos os recursos e serviços da AWS. Em um ambiente real a forma correta e segura é escolher políticas segmentadas dos serviços que serão utilizados. Após a escolha clique no botão Próximo. Defina um nome e descrição para a sua nova política.

Detalhes da função

Nome da função

Insira um nome significativo para identificar esta função.

JenkinsTerraformAWS

Máximo de 64 caracteres. Use caracteres alfanuméricos e '+,=,@,-,_'.

Descrição

Adicione uma breve explicação para esta função.

Allows EC2 instances to call AWS services on your behalf.

Máximo de 1000 caracteres. Use caracteres alfanuméricos e '+,=,@,-,_'.

Etapa 1: Selecionar entidades confiáveis

Editar

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Effect": "Allow",  
6       "Action": [  
7         "sts:AssumeRole"  
8       ],  
9       "Principal": {  
10        "Service": [  
11          "ec2.amazonaws.com"  
12        ]  
13      }  
14    ]  
15  }  
16 }
```

Clique no botão Criar Função. Assim que for criada ela ficará disponível para uso.

IAM > Funções

Funções (9) [Informações](#) Atualizar Excluir Criar função

Uma função do IAM é uma identidade que você pode criar que tem permissões específicas com credenciais válidas por curtos períodos. Funções podem ser assumidas por entidades em que você confia.

<input type="checkbox"/>	Nome da função	Entidades confiáveis	Última atividade
<input type="checkbox"/>	AWSServiceRoleForAmazonSSM	Serviço da AWS: ssm (Função vinculada ao serviço)	2 horas atrás
<input type="checkbox"/>	AWSServiceRoleForElasticLoadBalancing	Serviço da AWS: elasticloadbalancing (Função vinculada ao serviço)	50 dias atrás
<input type="checkbox"/>	AWSServiceRoleForOrganizations	Serviço da AWS: organizations (Função vinculada ao serviço)	-
<input type="checkbox"/>	AWSServiceRoleForRDS	Serviço da AWS: rds (Função vinculada ao serviço)	2 horas atrás
<input type="checkbox"/>	AWSServiceRoleForSupport	Serviço da AWS: support (Função vinculada ao serviço)	-
<input type="checkbox"/>	AWSServiceRoleForTrustedAdvisor	Serviço da AWS: trustedadvisor (Função vinculada ao serviço)	-
<input type="checkbox"/>	conn-tf-aws	Serviço da AWS: ec2	-
<input type="checkbox"/>	JenkinsTerraformAWS	Serviço da AWS: ec2	-

A próxima fase é criação de uma instância EC2 e instalação do Jenkins Server. Para isso vamos utilizar uma instância do tipo T2.micro que se enquadra no [Nível Gratuito AWS](#) de uso.

Antes vou deixar o código de instalação do Jenkins para ser usado no momento de criação da instância, ou seja, a instância será criada e o Jenkins instalado ao mesmo tempo.

```
### Script de Instalação do Jenkins ###
```

```
#!/bin/bash
```

```
# Update and upgrade system  
apt-get update && apt-get -y upgrade
```

```
# Install Java  
apt install -y fontconfig openjdk-11-jre
```

```
# Install Jenkins  
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | apt-key add -  
sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \  
/etc/apt/sources.list.d/jenkins.list'  
apt-get update  
apt-get install -y jenkins
```

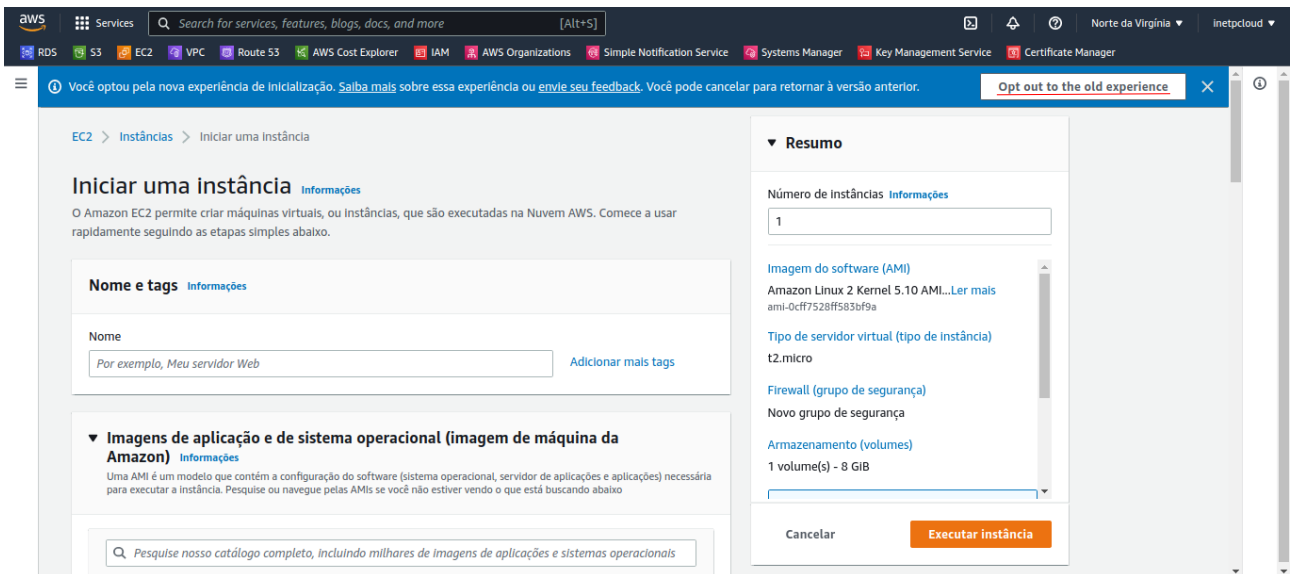
```
# Add Jenkins on startup  
systemctl enable jenkins
```

Vamos utilizar como base uma instância Ubuntu.

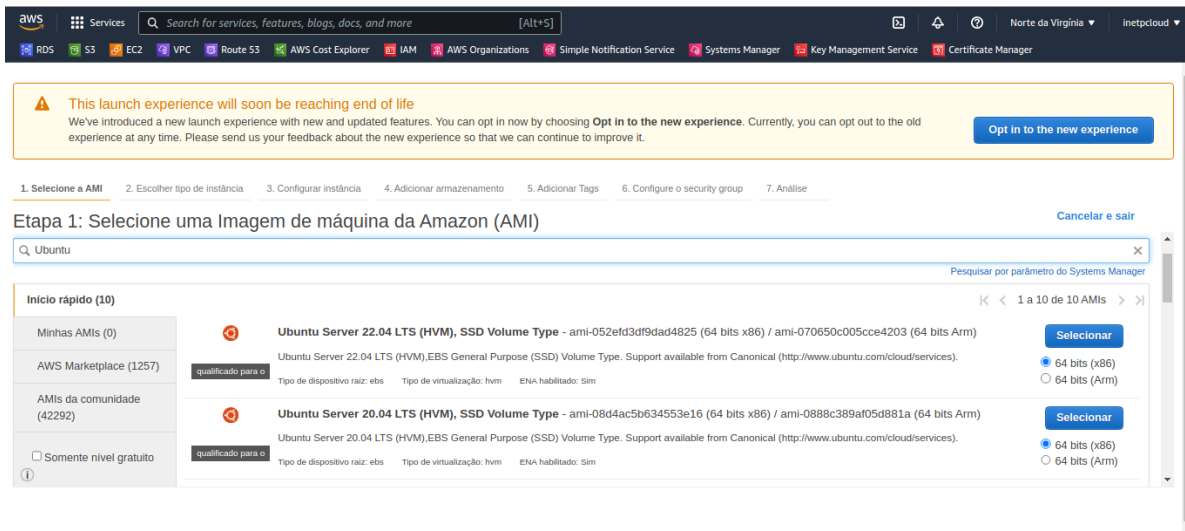
Como o Painel de Serviços da AWS muda constantemente, irei utilizar a versão antiga da console para ficar mais simples. Quem se sentir confortável com a nova versão pode usar sem problemas. Na busca escolha o serviço EC2.

Nome da zona	ID da zona
us-east-1a	use1-az1
us-east-1b	use1-az2
us-east-1c	use1-az4
us-east-1d	use1-az6
us-east-1e	use1-az3

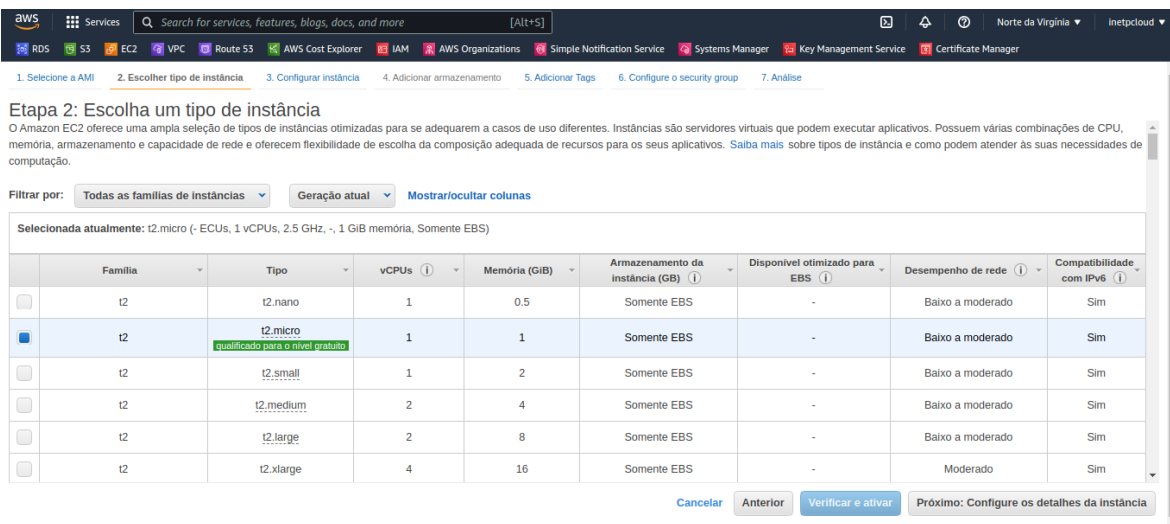
Vamos clicar no botão para Executar Instância. Para voltar para a console antiga clique no botão **Opt out the old experience.**



Dessa forma ele muda para o padrão antigo. Na barra de Pesquisa procure por Ubuntu e aperte o Enter. Temos duas opções no nível gratuito para escolha.



Selecione a primeira opção - **Ubuntu Server 22.04 LTS**. Já está marcada como padrão a opção T2.micro. Clique no botão Próximo.



Caso já tenha criado a sua VPC ela estará disponível na caixa de seleção junto com a VPC padrão. No meu cenário eu não utilizo a VPC Padrão. Também já está selecionada a sub-rede pública que a nossa instância fará parte, para que possa ter acesso a internet.

aws Services Search for services, features, blogs, docs, and more [Alt+S] Norte da Virgínia inetcloud

1. Selecionar a AMI 2. Escolher tipo de instância 3. Configurar instância 4. Adicionar armazenamento 5. Adicionar Tags 6. Configure o security group 7. Análise

Etapa 3: Configure os detalhes da instância

Configure a instância para se adequar aos seus requisitos. Você pode executar várias instâncias na mesma AMI, solicitar instâncias spot para aproveitar a vantagem de preços mais baixos, atribuir uma função de gerenciamento de acesso à instância, e outros.

Número de instâncias Executar no grupo de Auto Scaling

Opção de compra Solicitar instâncias spot

Rede Criar nova VPC
Nenhuma VPC padrão encontrada. Criar uma nova VPC padrão.

Sub-rede Criar nova sub-rede
4091 endereços IP disponíveis

Auto-assign Public IP

Atribuir IP IPv6 automaticamente

Tipo de nome do host

DNS Hostname Enable IP name IPv4 (A record) DNS requests
 Habilitar solicitações de DNS IPv4 (registro A) com base em recursos
 Habilitar solicitações de DNS IPv6 (registro AAAA) com base em recursos

Cancelar Anterior **Verificar e ativar** Próximo: Adicionar armazenamento

Role a barra até o final da página onde usaremos o script acima para que o Jenkins seja instalado. Copie e cole o script em dados de usuário. Ao final aperte o Enter. Caso o contrário o script não será executado. Depois clique no botão Próximo.

aws Services Search for services, features, blogs, docs, and more [Alt+S] Norte da Virgínia inetcloud

1. Selecionar a AMI 2. Escolher tipo de instância 3. Configurar instância 4. Adicionar armazenamento 5. Adicionar Tags 6. Configure o security group 7. Análise

Etapa 3: Configure os detalhes da instância

ADICIONAR IP ADICIONAR IP

Adicionar dispositivo

▼ Detalhes avançados

Enclave Habilitar

Metadados acessíveis

Versão de metadados

Limite de salto de resposta do token de metadados

Allow tags in metadata

Dados do usuário Como texto Como arquivo A entrada já está codificada como base64

```
api-get update
api-get install -y jenkins
# Add Jenkins on startup
systemctl enable jenkins
```

Cancelar Anterior **Verificar e ativar** Próximo: Adicionar armazenamento

A próxima parte é sobre o disco rígido. A opção padrão já vem configurada com 8GB que é mais que suficiente para nosso laboratório. Clique no botão Próximo.

The screenshot shows the AWS console interface for the 'Add storage' step. At the top, there's a navigation bar with the AWS logo, a search bar, and various service icons. Below that, a progress bar indicates the current step: '4. Adicionar armazenamento'. The main content area is titled 'Etapa 4: Adicionar armazenamento' and contains a table with columns for 'Tipo de volume', 'Dispositivo', 'Snapshot', 'Tamanho (GiB)', 'Tipo de volume', 'IOPS', 'Transferência (MB/s)', 'Excluir no encerramento', and 'Criptografia'. A single row is visible for the 'Root' volume, showing a size of 8 GiB and 'Finalidade geral de SSD (gp2)' type. Below the table is a 'Adicionar novo volume' button and a blue informational box. At the bottom right, there are navigation buttons: 'Cancelar', 'Anterior', 'Verificar e ativar', and 'Próximo: Adicionar Tags'.

As Tags na AWS, são muito importantes, tanto para localização de serviços e recursos criados, como também na bilhetagem de custos. As Tags usam os atributos Chave-Valor. Clique em criar tag. A chave será Name(N maiúsculo) e o Valor o nome da instância, nesse caso - Jenkins Server.

The screenshot shows the AWS console interface for the 'Add tags' step. The navigation bar and progress bar are similar to the previous step, with the current step being '5. Adicionar Tags'. The main content area is titled 'Etapa 5: Adicionar Tags' and contains a form with two input fields: 'Chave' (Key) and 'Valor' (Value). The 'Chave' field has 'Name' entered and the 'Valor' field has 'Jenkins Server' entered. There are checkboxes for applying the tag to 'Instâncias', 'Volumes', and 'Interfaces de rede'. Below the form is a 'Criar outra tag' button. At the bottom right, there are navigation buttons: 'Cancelar', 'Anterior', 'Verificar e ativar', and 'Próximo: Configure o security group'.

A próxima etapa é a criação do grupo de segurança, que vai permitir que a instancia seja acessada via SSH e também via HTTP. Por default o SSH já vem liberado para acesso. Clique em Adicionar Regra e escolha Regra Personalizada de TCP e informe a porta 8080. Dessa forma esse equipamento está liberado para acesso. Lembrando que a forma correta seria liberar apenas o seu **IP** para que pudesse se comunicar com a instancia. Clique em Verificar e Ativar.

Etapa 6: Configure o security group

Um grupo de segurança é um conjunto de regras de firewall que controla o tráfego da sua instância. Nesta página, você pode adicionar regras para permitir que tráfegos específicos cheguem até a sua instância. Por exemplo, se você quiser configurar um servidor Web e permitir que tráfego da Internet chegue até a sua instância, adicione regras que permitam acesso irrestrito às portas HTTP e HTTPS. Você pode criar um novo grupo de segurança ou selecionar um dos existentes abaixo. [Saiba mais](#) sobre grupo de segurança do Amazon EC2.

Atribuir um grupo de segurança: Criar um grupo de segurança novo
 Selecionar um grupo de segurança existente

Nome do grupo de segurança:
 Descrição:

Tipo	Protocolo	Intervalo de Portas	Origem	Descrição
SSH	TCP	22	Personaliza 0.0.0.0/0	Por exemplo SSH for Admin Desktop
Regra personalizada	TCP	8080	Personaliza 0.0.0.0, ::0	Por exemplo SSH for Admin Desktop

Aviso
 Regras com origem 0.0.0.0/0 permitem que todos os endereços IP acessem sua instância. Recomendamos configurar regras de grupo de segurança para permitir o acesso apenas de endereços IP conhecidos.

A próxima tela mostra um Review completo da instância com tudo que foi configurado. Clique no botão Executar.

Etapa 7: Review Instance Launch

Verifique os detalhes de execução da instância. Você pode voltar para editar alterações para cada seção. Clique em **Executar** para atribuir um par de chaves à sua instância e concluir o processo de execução.

Melhore a segurança da sua instância. Seu grupo de segurança, launch-wizard-1, está aberto para o mundo.
 Sua instância pode ser acessada de qualquer endereço IP. Recomendamos atualizar as regras do seu grupo de segurança para permitir o acesso apenas de endereços IP conhecidos. Você também pode abrir portas adicionais no seu grupo de segurança para facilitar o acesso ao aplicativo ou serviço que está executando, por ex., HTTP (80) para servidores Web. [Editar grupos de segurança](#)

Detalhes da AMI [Editar AMI](#)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type - ami-052efd3df9dad4825
 Ubuntu Server 22.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).
 Tipo de dispositivo raiz: ebs Tipo de virtualização: hvm

Tipo de instância [Editar tipo de instância](#)

Tipo de instância	ECUs	vCPUs	Memória (GiB)	Armazenamento da instância (GB)	Disponível otimizado para EBS	Desempenho de rede
t2.micro	-	1	1	Somente EBS	-	Low to Moderate

Grupos de segurança [Editar grupos de segurança](#)

A etapa seguinte vamos criar uma chave de acesso secreta. Escolha no Dropdown a opção - Criar um novo par de chaves. O tipo de par de chaves escolha - ED25519. Especifique um nome para a chave e depois em **Fazer download do par de chaves**. E depois em Executar Instâncias.

✕

Selecione um par de chaves existente ou crie um novo par de chaves

chaves

Um par de chaves consiste em uma **chave pública** armazenada pela AWS e um **arquivo de chave privada** que você armazena. Juntos, eles permitem que você se conecte à sua instância com segurança. Em AMIs do Windows, o arquivo de chave privada é necessário para obter a senha usada para fazer login na sua instância. Para AMIs do Linux, o arquivo de chave privada permite fazer SSH com segurança na sua instância. O Amazon EC2 oferece suporte aos tipos de par de chaves ED25519 e RSA.

Observação: O par de chaves selecionado será adicionado ao conjunto de chaves autorizado para essa instância. Saiba mais sobre [Como remover pares de chaves existentes de uma AMI pública](#).

Criar um novo par de chaves

Tipo de par de chaves

RSA
 ED25519

Nome do par de chaves

Jenkins

Fazer download do par de chaves

... Antes de continuar, você precisa fazer download do **arquivo de chave privada** (*.pem file). **Armazene-o em um local seguro e acessível**. Depois que o arquivo tiver sido criado, não será possível fazer o download novamente.

Cancelar
Executar instâncias

Por fim Clique em Exibir Instâncias.

Caso tudo tenha corrido de forma correta a instância será exibida.

The screenshot shows the AWS Management Console interface. The top navigation bar includes the AWS logo, a search bar, and various service icons. The main content area is titled 'Instâncias (1) Informações'. Below the title, there are buttons for 'Conectar', 'Estado da instância', 'Ações', and 'Executar instâncias'. A search bar is present above a table of instances. The table has columns for Name, ID de instância, Estado da inst..., Tipo de inst..., Verificação de s..., Status do al..., Zona de dispon..., and DNS IPv4 p. One instance is listed: 'Jenkins Server' with ID 'i-0682312f444586717', state 'Executando', type 't2.micro', and '2/2 verificações aj'. The left sidebar shows navigation options like 'Painel EC2', 'Visualização Global do EC2', 'Eventos', 'Tags', 'Limites', and 'Instâncias'.

Agora vamos verificar se a instalação do Jenkins foi efetuada corretamente. Clique na caixa de seleção do Jenkins Server e copie o Endereço IPV4 público.

The screenshot shows the AWS Management Console interface. At the top, there's a header for 'Instâncias (1/1) Informações'. Below it is a table with columns: Name, ID de instância, Estado da inst..., Tipo de inst..., Verificação de s..., Status do al..., Zona de dispon..., and DNS IPv4 p. The first row is 'Jenkins Server' with ID 'i-0682312f444586717', status 'Executando', type 't2.micro', and public DNS 'ec2-34-205...'. Below the table, the details for 'Instância: i-0682312f444586717 (Jenkins Server)' are shown. The 'Resumo da instância' section includes: ID de instância (i-0682312f444586717), Endereço IPv6 (-), Endereço IPv4 público (redacted with 'endereço aberto'), Estado da instância (Executando), Endereços IPv4 privados (redacted), and DNS IPv4 público (redacted with 'endereço aberto').

Cole o endereço IP no navegador com a porta 8080 dessa forma:
Exemplo: **192.168.0.1:8080**. Troque o IP do exemplo pelo da sua instância.

Dessa forma será exibido a tela abaixo:

The screenshot shows the Jenkins 'Começando' (Getting Started) screen. The title is 'Abrir o Jenkins'. The text reads: 'Para garantir que o Jenkins está configurado de forma segura pelo administrador, uma senha foi escrita no arquivo de registro (não sabe onde encontrar?) e neste arquivo no servidor:'. Below this is a code block: `/var/lib/jenkins/secrets/initialAdminPassword`. The text continues: 'Por favor copie a senha de qualquer uma das localizações e cole abaixo.' There is a text input field labeled 'Senha do administrador'. At the bottom right, there is a 'Continuar' button.

Agora precisamos da senha de acesso para o servidor Jenkins. Existem algumas formas para resgate da senha. Se estiver usando o Linux acesse a instância via SSH, caso use o Mac ou Windows utilize o Putty ou outro programa de acesso da sua preferência.

No linux acesse o diretório onde foi feito o download da chave de acesso. De a permissão de acesso na chave e faça o acesso a instância.

```
rogerio@AKUMANOMI: ~/Documentos/DevOps
Arquivo Editar Ver Pesquisar Terminal Ajuda
(base) rogerio@AKUMANOMI:~/Documentos/DevOps$ chmod 600 Jenkins.pem
(base) rogerio@AKUMANOMI:~/Documentos/DevOps$ ssh -i Jenkins.pem ubuntu@IP da sua instancia
```

Aperte a tecla Enter e confirme a chave com o Yes. A seguir você já estará conectado em sua instancia. O IP que aparece é o IP Interno da instância.

```
ubuntu@ip- [redacted] :~$
```

Agora com o comando cat vamos recuperar a senha do Jenkins.

```
ubuntu@ip- [redacted] :~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
1f0e7d3d895443088cbd30799c966977
ubuntu@ip- [redacted] :~$
```

Copie o hash e cole na pagina do Jenkins

Começando

Abrir o Jenkins

Para garantir que o Jenkins está configurado de forma segura pelo administrador, uma senha foi escrita no arquivo de registro (**não sabe onde encontrar?**) e neste arquivo no servidor:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Por favor copie a senha de qualquer uma das localizações e cole abaixo.

Senha do administrador

Continuar

Clique em Continuar para iniciar a instalação e escolha a opção selecionada.

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.346.2

A instalação irá prosseguir, configurando todos os plugins básicos necessários.

Getting Started

Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	✓ Workspace Cleanup	✓ Ant	⚙ Gradle
⚙ Pipeline	⚙ GitHub Branch Source	⚙ Pipeline: GitHub Groovy Libraries	⚙ Pipeline: Stage View
⚙ Git	⚙ SSH Build Agents	⚙ Matrix Authorization Strategy	⚙ PAM Authentication
⚙ LDAP	⚙ Email Extension	✓ Mailer	

```
Ant
** Durable Task
** Pipeline: Nodes and Processes
** Oracle Java SE Development Kit Installer
** Command Agent Launcher
** bouncycastle API
** JavaScript GUI Lib: ACE Editor bundle
** Pipeline: SCM Step
** Pipeline: Groovy
** Pipeline: Job
** Jakarta Activation API
** Jakarta Mail API
** Apache HttpComponents Client 4.x API
Mailer
** Pipeline: Basic Steps
Gradle
** - required dependency
```

Jenkins 2.346.2

Configure os seus dados de acesso, clique em Save e Continue.

Getting Started

Criar o primeiro usuário administrativo

Nome de usuário:

Senha:

Confirmar a senha:

Jenkins 2.346.2 [Skip and continue as admin](#) [Save and Continue](#)

Copie e salve a URL de acesso ao Jenkins, clique em Save e Finish.

Getting Started

Configuração da instancia

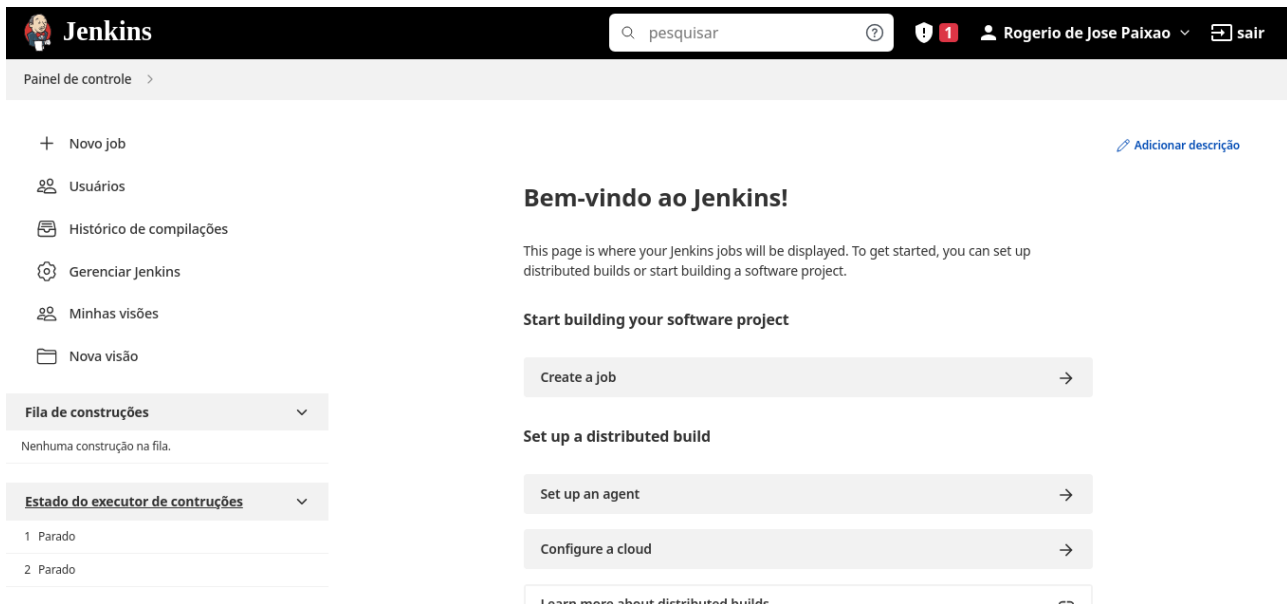
URL do Jenkins:

A URL do Jenkins é usada para prover a URL raiz para links absolutos para vários recursos do Jenkins. Isto significa que este valor é requerido para a operação apropriada de muitas funcionalidades do Jenkins incluindo notificações por e-mail, atualização de estado de PR e a variável de ambiente BUILD_URL provida pelos passos de construção.

O valor proposto padrão mostrado é **ainda não salvo** e é gerado da solicitação atual, se possível. A melhor prática é configurar este valor para a URL que espera-se que os usuários utilizem. Isto evita confusão quando compartilhando ou vendo links.

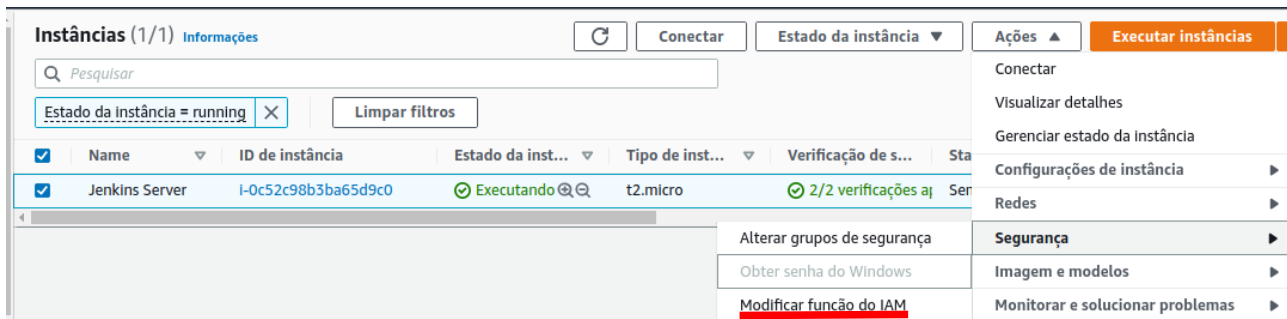
Jenkins 2.346.2 [Not now](#) [Save and Finish](#)

Tela Principal do Jenkins. E se você chegou até aqui já é um vencedor.



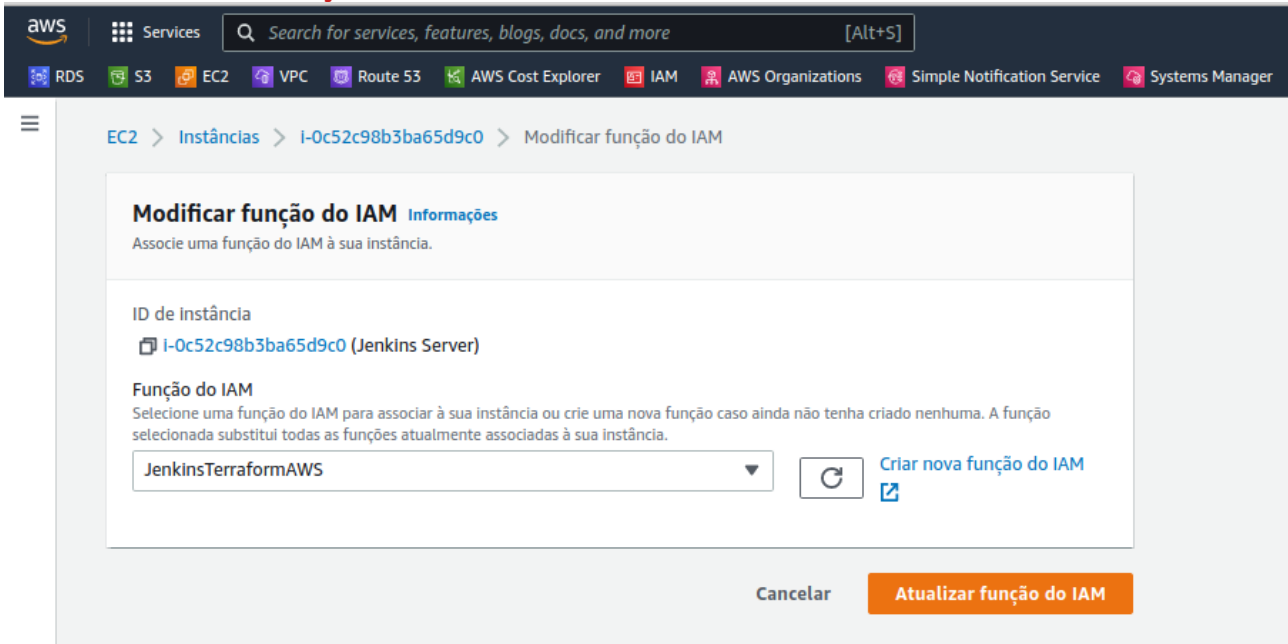
Bom terminada a instalação do servidor Jenkins ainda falta configurar uma opção em nossa instância. Alguém tem alguma opinião?

Falta adicionar a Função que criamos para que o Jenkins possa acessar a sua conta AWS e criar a sua infraestrutura via Pipeline. Vejamos a seguir.



Acesse a opção Modificar função do IAM e escolha a função criada.

Clique na seta para baixo na caixa de seleção e escolha a sua função e clique no botão **Atualizar função do IAM**



Agora precisamos fazer a instalação do [Terraform](#) em nossa instância.

No meu cenário com uma instância Ubuntu seria dessa forma.

```
$ wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg
$ echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" |
sudo tee /etc/apt/sources.list.d/hashicorp.list
$ sudo apt update && sudo apt install terraform
```

Após a instalação verificar a versão do Terraform com o comando
\$ terraform --version. No meu caso teve essa saída.

```
Terraform v1.2.5
on linux_amd64
```

Com isso podemos partir para a próxima etapa. Criar um pipeline que irá regatar nosso projeto terraform no repositório do GitHub e criar a infraestrutura na AWS.

Vou disponibilizar o link do meu repositório que contem um projeto relativamente simples que serviu de aprendizado.

Link - <https://github.com/rjpaixao/tf-jk-iac-aws>

Abaixo vou deixar o modelo do pipeline que utilizei. É um código simples que faz o checkout no repositório, inicia o terraform e solicita que seja escolhida qual a ação a ser tomada.

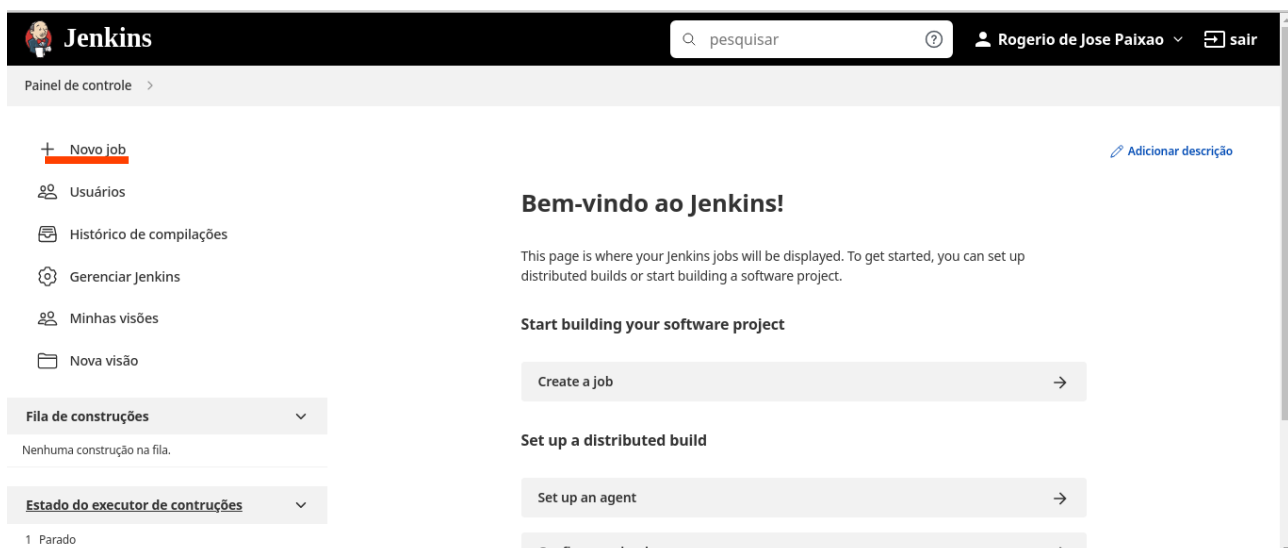
Exemplo de Pipeline

```
pipeline {
  agent any

  stages {
    stage('checkout') {
      steps {
        checkout([$class: 'GitSCM', branches: [[name: '*/main']], extensions: [],
userRemoteConfigs: [[url: 'https://github.com/seu_usuario/seu_repositorio']]])
      }
    }
    stage ("terraform init") {
      steps {
        sh ('terraform init')
      }
    }

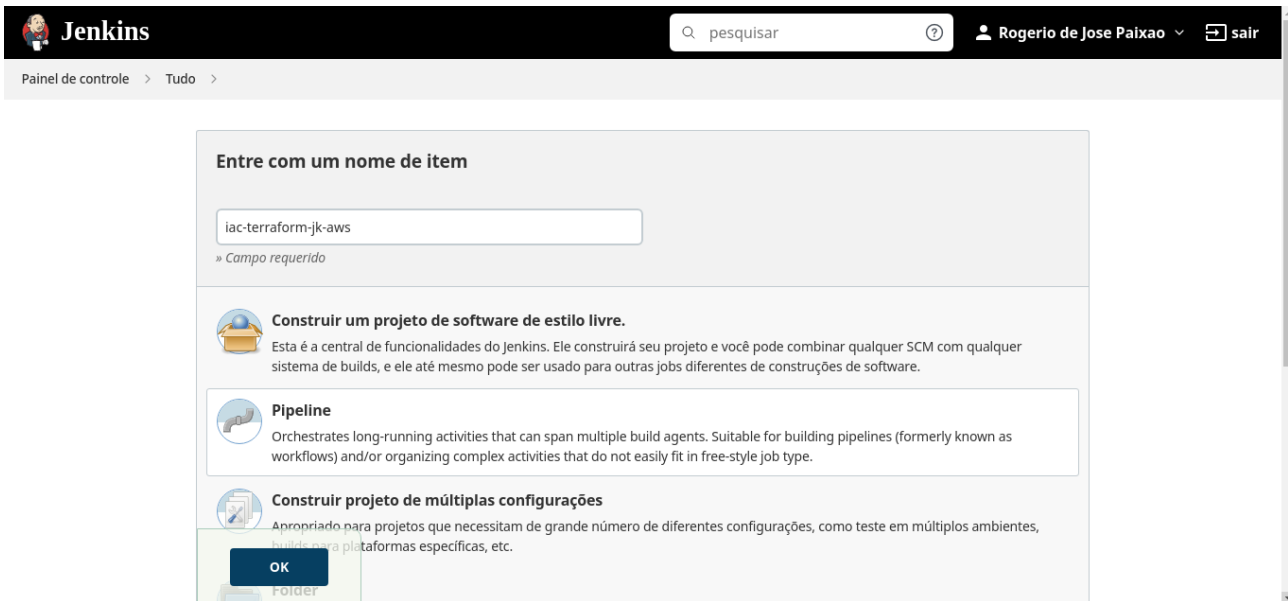
    stage ("terraform Action") {
      steps {
        echo "Terraform action is --> ${action}"
        sh ('terraform ${action} --auto-approve')
      }
    }
  }
}
```

O próximo passo é criar o nosso job no Jenkins que ira criar a nossa infraestrutura na nuvem AWS. Para isso acesse a sua console do Jenkins

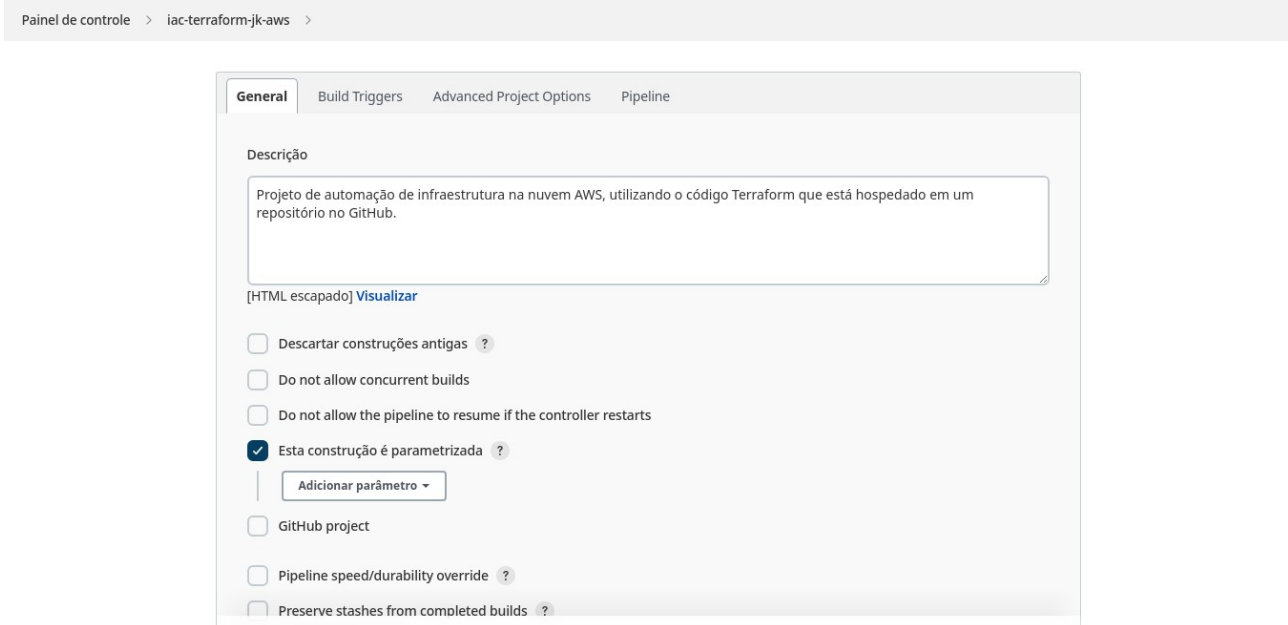


The screenshot shows the Jenkins web interface. At the top, there's a navigation bar with the Jenkins logo, a search bar containing 'pesquisar', and a user profile for 'Rogério de Jose Paixao' with a 'sair' (logout) button. Below the navigation bar is a 'Painel de controle' (Control Panel) section with a list of links: '+ Novo job', 'Usuários', 'Histórico de compilações', 'Gerenciar Jenkins', 'Minhas visões', and 'Nova visão'. On the right side of the dashboard, there's a 'Bem-vindo ao Jenkins!' (Welcome to Jenkins!) message, followed by instructions: 'This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.' Below this, there are three main action buttons: 'Create a job', 'Set up a distributed build', and 'Configure a cloud'. On the left side, there are two dropdown menus: 'Fila de construções' (Build Queue) showing 'Nenhuma construção na fila.' (No builds in queue) and 'Estado do executor de construções' (Build Executor Status) showing '1 Parado' (1 Stopped).

Clique em criar um novo job, defina um nome para ele e escolha a opção Pipeline e clique no Botão OK.



Informe uma descrição para o seu projeto, marque a opção Essa construção é parametrizada e na caixa de seleção marque a opção "Escolha"



Defina o nome como **“action”**, e as Escolhas como **“apply & destroy”**.
Esse serão os comandos executados pelo Terraform para criar ou destruir a nossa infraestrutura.

General Build Triggers Advanced Project Options Pipeline

Esta construção é parametrizada ?

Escolha ?

Nome ?

action

Escolhas ?

apply
destroy

Descrição ?

[HTML escapado] [Visualizar](#)

Agora acesse a guia Pipeline. Em “Definition” deixe selecionado Pipeline script e ao lado escolha a opção “Hello World”, para gerar um código de exemplo. Altere o stage pra (‘checkout’), essa verificação será feita no seu repositório no Github. Apague também o comando echo. Depois clique em Pipeline Syntax para gerar o código de checkout, para que fique conforme imagem abaixo.

General Build Triggers Advanced Project Options **Pipeline**

Pipeline script

Script ?

```
1 pipeline {
2   agent any
3
4   stages {
5     stage('checkout') {
6       steps {
7
8       }
9     }
10  }
11 }
12 }
```

Use Groovy Sandbox ?

[Pipeline Syntax](#)

Depois de clicar em Pipeline Syntax, escolha a opção Checkout em Steps. Em Repository URL - coloque o link do seu repositório, conforme mostrado abaixo. Não é necessário credencial pois o repositório é público.

Pipeline Syntax

Steps

Sample Step

checkout: Check out from version control

checkout ?

SCM

Git

Repositories ?

Repository URL ?

https://github.com/rjpaixao/tf-jk-iac-aws

Credentials ?

- none -

+ Add

Mais abaixo troque o nome da branch para a que você está usando, no meu caso é a "main" e clique no botão Generate Pipeline Script. Copie o código e volte para a tela de construção do Pipeline.

Pipeline Syntax

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

Add Branch

Navegar no repositório ?

(Auto)

Additional Behaviours

Adicionar

Include in polling? ?

Include in changelog? ?

Generate Pipeline Script

```
checkout([$class: 'GitSCM', branches: [[name: */main]], extensions: [], userRemoteConfigs: [[url: 'https://github.com/rjpaixao/tf-jk-iac-aws']]])
```

Pipeline

Definition

Pipeline script

Script ?

```
1 pipeline {
2   agent any
3
4   stages {
5     stage('chekcout') {
6       steps {
7         checkout([$class: 'GitSCM', branches: [[name: '*/main']], extensions: [], userRemoteConfigs
8       }
9     }
10  }
11 }
12 }
13 }
```

Salvar Aplicar

Não esqueça de salvar o seu projeto constantemente. Agora vamos configurar os steps para iniciar o Terraform e para aplicar os parâmetros que escolhemos.

Definition

Pipeline script

Script ?

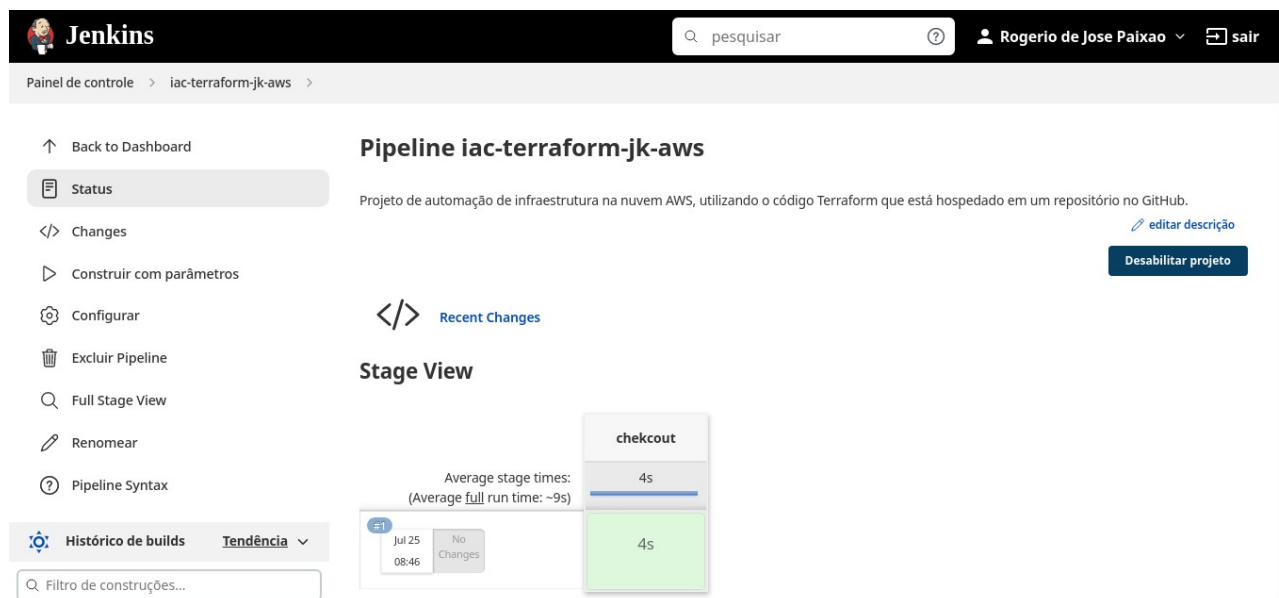
```
8   }
9   }
10  stage ("terraform init") {
11    steps {
12      sh ('terraform init')
13    }
14  }
15
16  stage ("terraform Action") {
17    steps {
18      echo "Terraform action is --> ${action}"
19      sh ('terraform ${action} --auto-approve')
20    }
21  }
22 }
23 }
24 }
```

try sample Pipeline...

O Stage do Terraform Action utiliza os parâmetros ("**apply**" ou "**destroy**") na variável **\${action}**, dependendo da sua escolha no momento de criação do Job. Terminado o script salve o seu trabalho. A condição ('--auto-approve'), faz com que o código seja executado imediatamente, sem que seja necessária aprovação prévia.

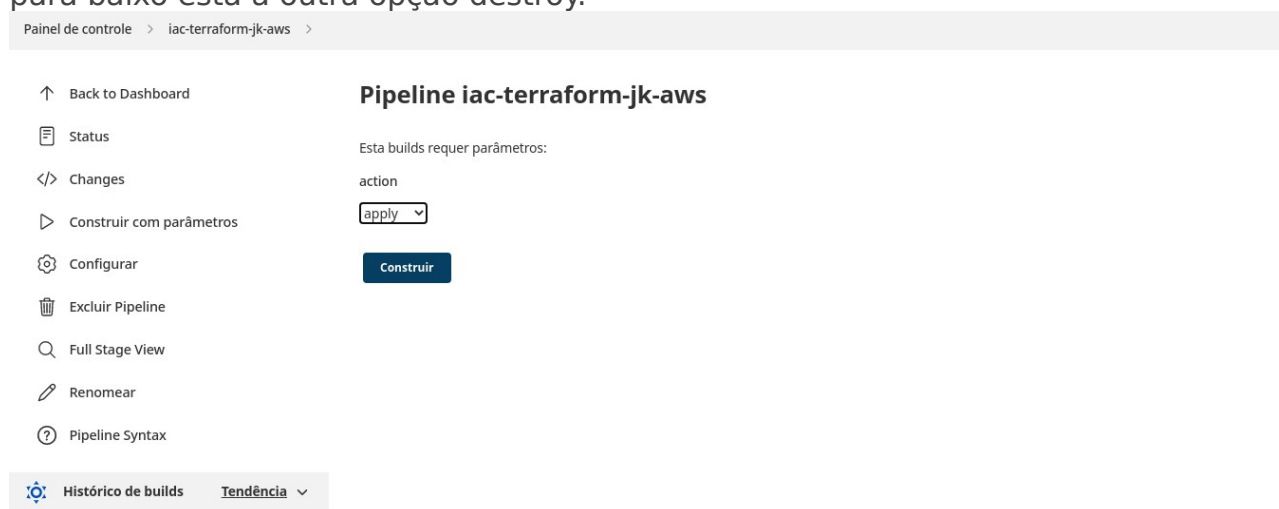
Lembrando, esse cenário é apenas para aprendizado, não sendo aconselhável a reprodução em um ambiente de produção.

Voltando ao painel de controle do Jenkins, vamos acessar a opção construir com parâmetros o nosso job e verificar a opções disponíveis



The screenshot shows the Jenkins interface for the pipeline 'iac-terraform-jk-aws'. The left sidebar contains navigation options: Back to Dashboard, Status (selected), Changes, Construir com parâmetros, Configurar, Excluir Pipeline, Full Stage View, Renomear, and Pipeline Syntax. Below these are 'Histórico de builds' and 'Tendência'. The main content area displays the pipeline name and a description: 'Projeto de automação de infraestrutura na nuvem AWS, utilizando o código Terraform que está hospedado em um repositório no GitHub.' It includes links for 'editar descrição' and a 'Desabilitar projeto' button. The 'Recent Changes' section shows a change on Jul 25 at 08:46 with 'No Changes'. The 'Stage View' section shows a bar chart for the 'checkout' stage with an average time of 4s. A filter box for builds is at the bottom left.

Nosso job já vem com o parâmetro apply selecionado por default. Clicando na seta para baixo está a outra opção destroy.



The screenshot shows the Jenkins interface for the pipeline 'iac-terraform-jk-aws' in the build configuration view. The left sidebar is identical to the previous screenshot. The main content area displays the pipeline name and a warning: 'Esta builds requer parâmetros:'. Below this, the 'action' dropdown menu is set to 'apply'. A 'Construir' button is visible at the bottom.

Agora só nos resta fazer o teste. Clique construir par iniciar a construção do sua infraestrutura. Para verificar o andamento do processo clique no numero do Build, abaixo a esquerda e depois na saída do console

The screenshot shows the Jenkins interface. On the left, the 'Histórico de builds' (Build History) section is visible, showing three builds: #3 (25 de jul de 2022 13:03), #2 (25 de jul de 2022 12:54), and #1 (25 de jul de 2022 11:46). On the right, the 'Painel de controle' (Control Panel) for build #2 is shown, with a menu containing: 'Back to Project', 'Status', 'Changes', 'Console Output' (highlighted with a red underline), 'Edit Build Information', 'Apagar a construção {0}', and 'Parâmetros'.

Se tudo funcionou terá uma saída parecida com essa

Saída do console

```
Started by user Rogério de Jose Paixao
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/iac-terraform-jk-aws
[Pipeline] {
[Pipeline] stage
[Pipeline] { (checkout)
[Pipeline] checkout
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/iac-terraform-jk-aws/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/rjpaixao/tf-jk-iac-aws # timeout=10
Fetching upstream changes from https://github.com/rjpaixao/tf-jk-iac-aws
> git --version # timeout=10
> git --version # 'git version 2.34.1'
> git fetch --tags --force --progress -- https://github.com/rjpaixao/tf-jk-iac-aws
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
```

Dependendo do tamanho a infraestrutura que você criou a saída será extensa. No meu caso foram criados 23 novos recursos.

```
Apply complete! Resources: 23 added, 0 changed, 0 destroyed.
[][0m
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```


Meu Código do Terraform

(Arquivo main.tf)

```
terraform {
  required_providers {
    aws = {
      source = "hashicorp/aws"
      version = "4.19.0"
    }
  }
}

# Configure the AWS Provider
provider "aws" {
  region = var.aws_region
}

# Configuração do Módulo VPC

module "vpc" {
  source = "terraform-aws-modules/vpc/aws"
  version = "3.14.2"

  name          = var.vpc_name
  cidr          = var.vpc_cidr
  azs           = var.vpc_azs
  private_subnets = var.vpc_private_subnets
  public_subnets = var.vpc_public_subnets
  enable_nat_gateway = var.vpc_enable_nat_gateway
  tags          = var.vpc_tags
}

# Criação da Instância EC2

module "ec2-instance" {
  source = "terraform-aws-modules/ec2-instance/aws"
  version = "4.0.0"

  for_each = toset(["one", "two", "three"])

  name = "instance-${each.key}"

  ami          = "ami-08d4ac5b634553e16"
  instance_type = "t2.micro"
  key_name     = "DevOps"
  monitoring   = true
  vpc_security_group_ids = [module.vpc.default_security_group_id]
  subnet_id    = module.vpc.public_subnets[0]

  tags = {
    Terraform = "true"
    Environment = "dev"
  }
}
```

Arquivo (variables.tf)

```
variable "aws_region" {
  description = "AWS Region"
  default     = "us-east-1"
  type       = string
}

# Input Variables Definitions

variable "vpc_name" {
  description = "Name of VPC"
  default     = "Curso"
  type       = string
}

variable "vpc_cidr" {
  description = "CIDR Block For VPC"
  default     = "10.0.0.0/16"
  type       = string
}

variable "vpc_azs" {
  description = "Availability Zones For VPC"
  default     = ["us-east-1a", "us-east-1b", "us-east-1c", "us-east-1d"]
  type       = list(string)
}

variable "vpc_private_subnets" {
  description = "Private Subnet For VPC"
  default     = ["10.0.1.0/24", "10.0.2.0/24"]
  type       = list(string)
}

variable "vpc_public_subnets" {
  description = "Public Subnet For VPC"
  default     = ["10.0.101.0/24", "10.0.102.0/24"]
  type       = list(string)
}

variable "vpc_enable_nat_gateway" {
  description = "Enable Nat Gateway For VPC"
  default     = true
  type       = bool
}

variable "vpc_tags" {
  description = "Tags To Apply To Resources Created By VPC Module"
  default = {
    Terraform = "True"
    Environment = "Dev"
  }
  type       = map(string)
}
```

Com isso terminamos esse passo a passo simples para construir uma infraestrutura na nuvem AWS utilizando o Jenkins, Terraform e Github.

Espero ter contribuído com o aprendizado de vocês, pois eu também estou aprendendo.

Aceito todos os feedbacks, positivos e negativos, pois fazem parte do processo de aprendizado. Até a próxima

Contato - rogerio.j.paixao@gmail.com

Linkedin - <https://www.linkedin.com/in/rjpaixao/>