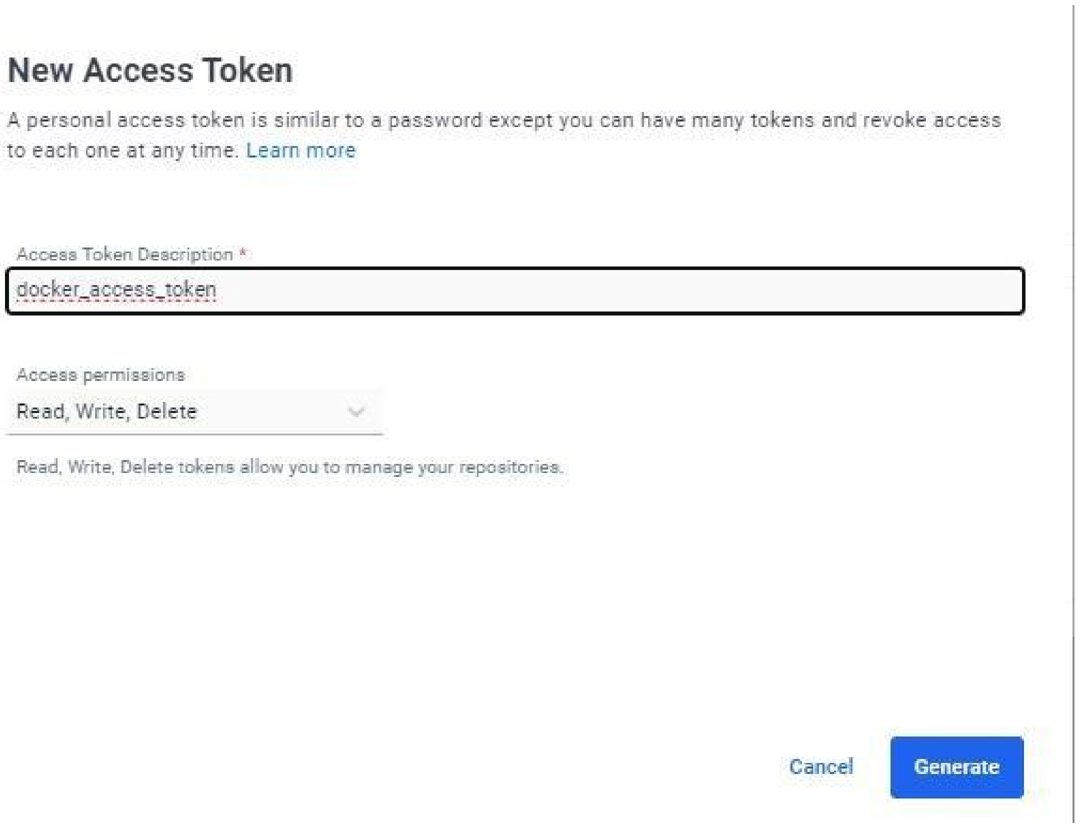
1. Create a Jenkins file to construct and upload Docker images to your Docker-hub registries. Ensure that when the branch is 'dev', the image is constructed and uploaded to the DEV Docker-hub registry. Similarly, when the branch is 'QA', it should be sent to the QA Docker-hub registry.



New Access Token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. Learn more

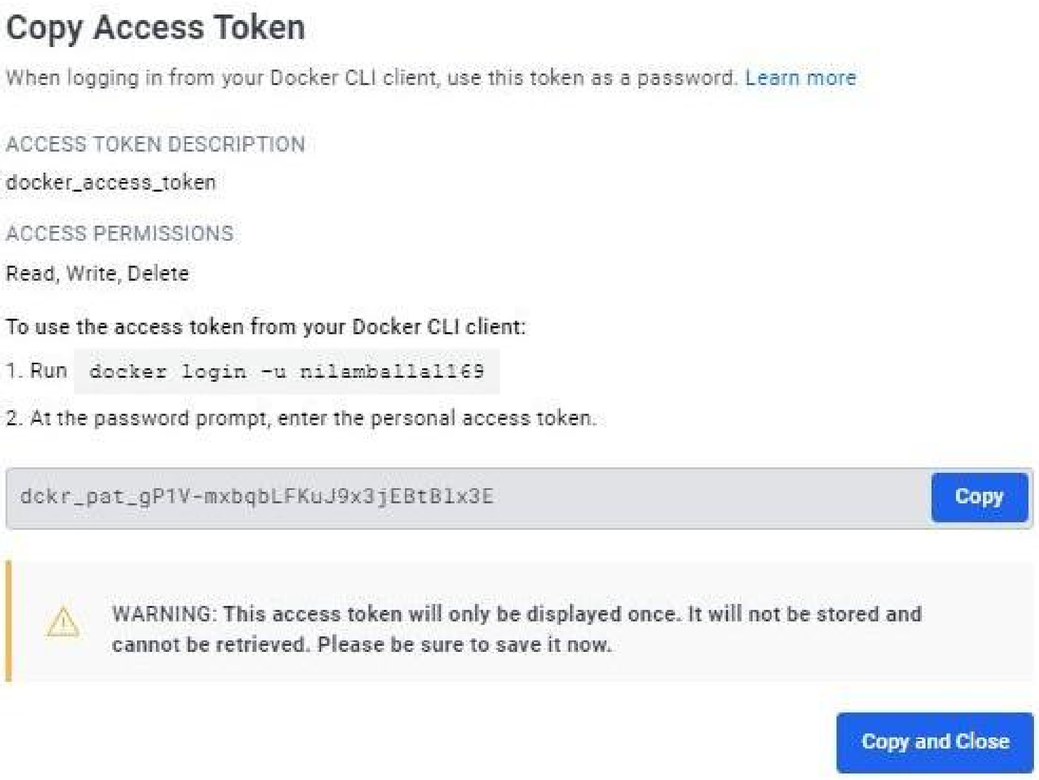
Access Token Description *

docker_access_token

Access permissions

Read, Write, Delete

Read, Write, Delete tokens allow you to manage your repositories.



Copy Access Token

When logging in from your Docker CLI client, use this token as a password. Learn more

ACCESS TOKEN DESCRIPTION

docker_access_token

ACCESS PERMISSIONS

Read, Write, Delete

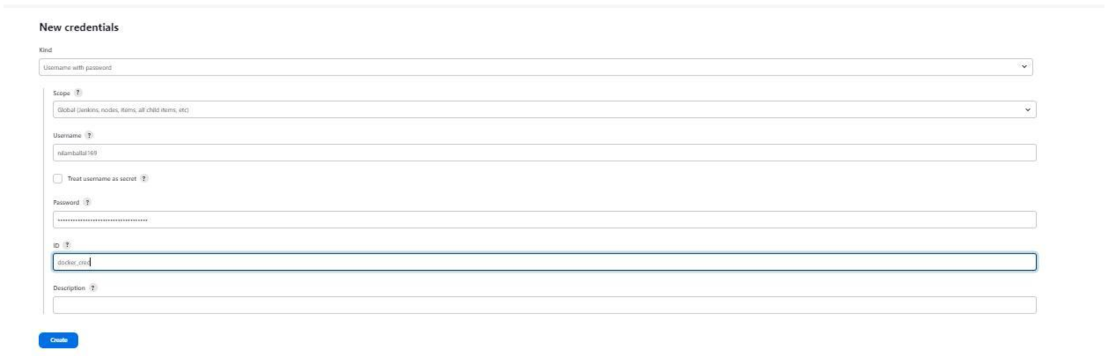
To use the access token from your Docker CLI client:

1. Run `docker login -u nilamballal169`

2. At the password prompt, enter the personal access token.

```
dckr_pat_gP1V-mxbqbLFKuJ9x3jEBtBlx3E
```

WARNING: This access token will only be displayed once. It will not be stored and cannot be retrieved. Please be sure to save it now.



New credentials

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

nilamballal169

☐ Treat username as secret ?

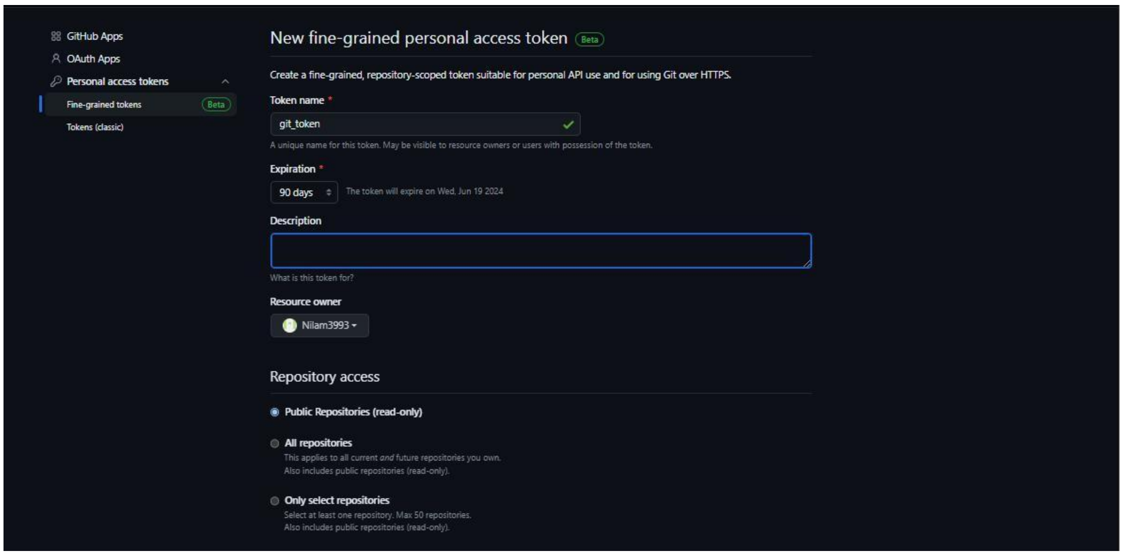
Password ?

••••••••••••••••••••••••••••••••••••

ID ?

docker_cred

Description ?



New fine-grained personal access token Beta

Create a fine-grained, repository-scoped token suitable for personal API use and for using Git over HTTPS.

Token name *

git_token

A unique name for this token. May be visible to resource owners or users with possession of the token.

Expiration *

90 days — The token will expire on Wed, Jun 19 2024

Description

What is this token for?

Resource owner

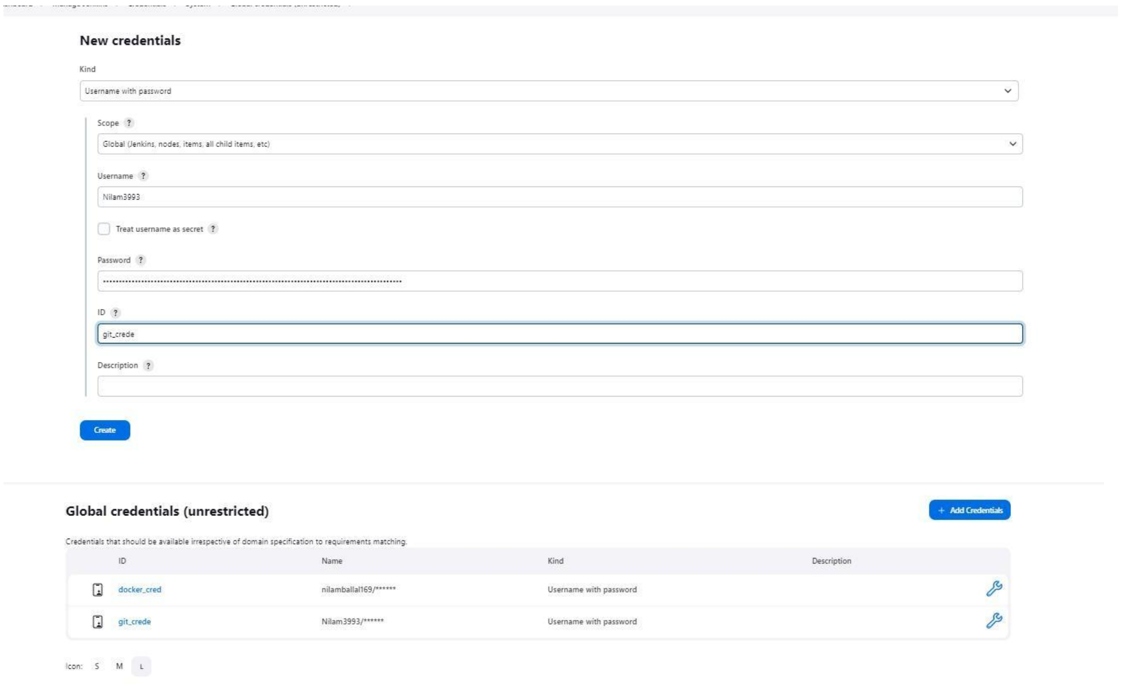
Nilam3993

Repository access

● Public Repositories (read-only)

○ All repositories
This applies to all current *and* future repositories you own.
Also includes public repositories (read-only).

○ Only select repositories
Select at least one repository. Max 50 repositories.
Also includes public repositories (read-only).



New credentials

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

Nilam3993

☐ Treat username as secret ?

Password ?

••

ID ?

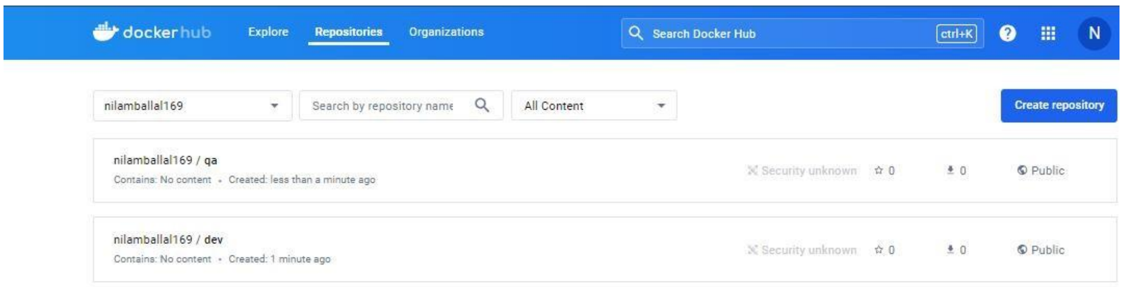
git_crede

Description ?

Global credentials (unrestricted)

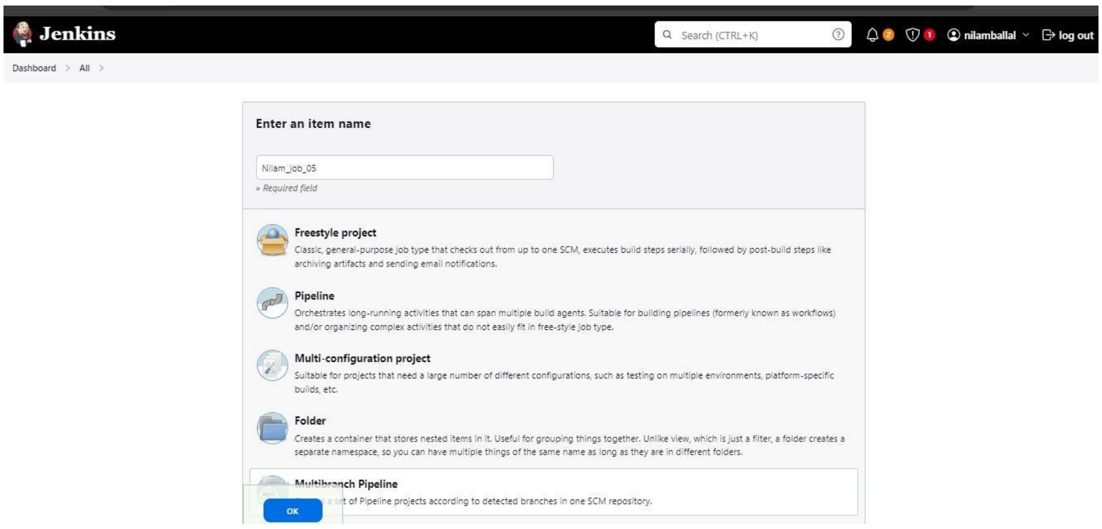
Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
docker_cred	nilamballal169/******	Username with password	
git_crede	Nilam3993/******	Username with password	



nilamballal169 / qa
Contains: No content • Created: less than a minute ago
Security unknown ☆ 0 ⬇ 0 Public

nilamballal169 / dev
Contains: No content • Created: 1 minute ago
Security unknown ☆ 0 ⬇ 0 Public



Enter an item name

Nilam_job_05

» Required field

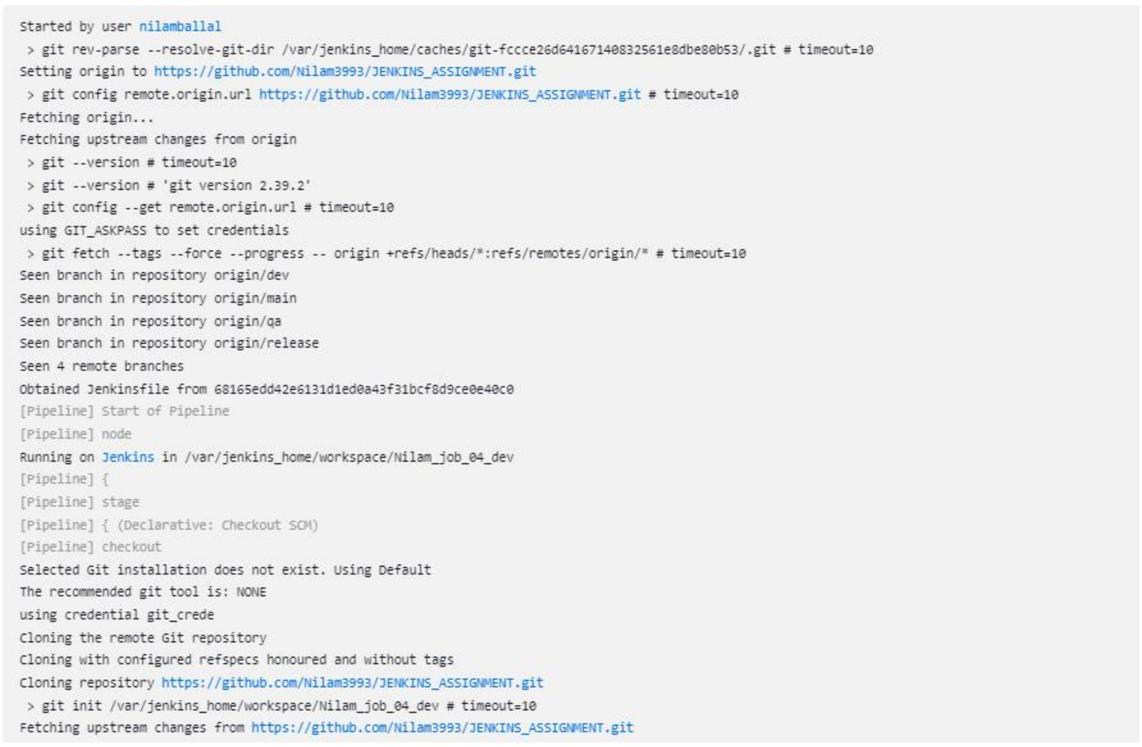
Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

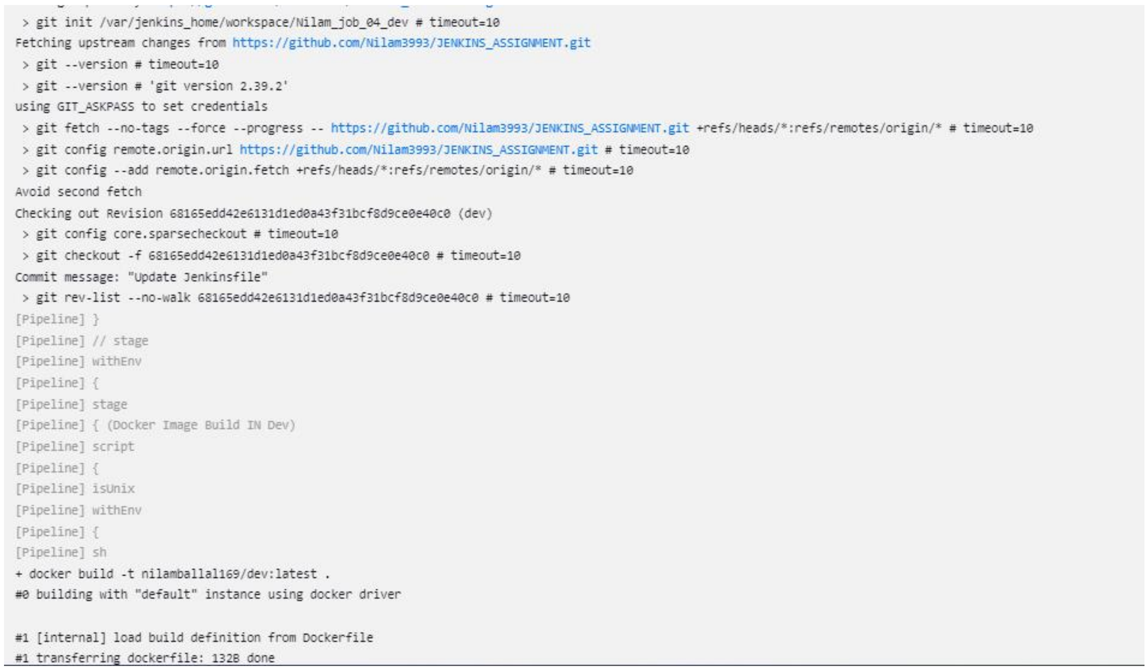
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

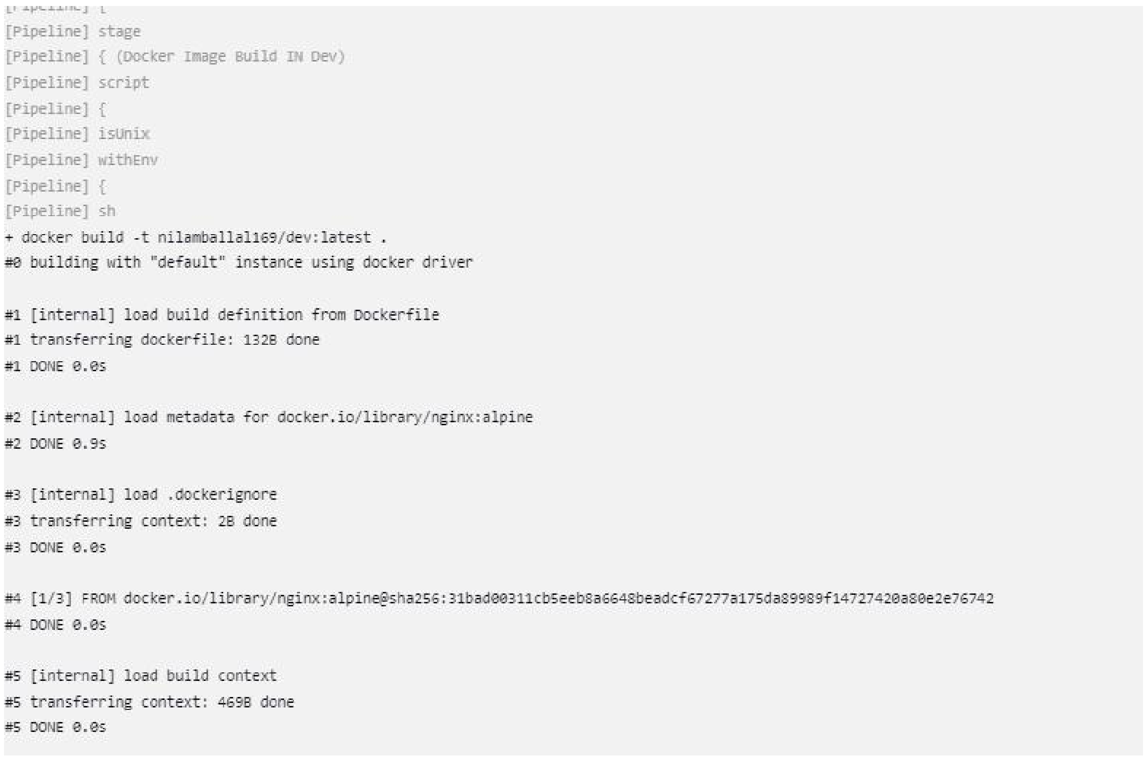
Multibranch Pipeline
a set of Pipeline projects according to detected branches in one SCM repository.



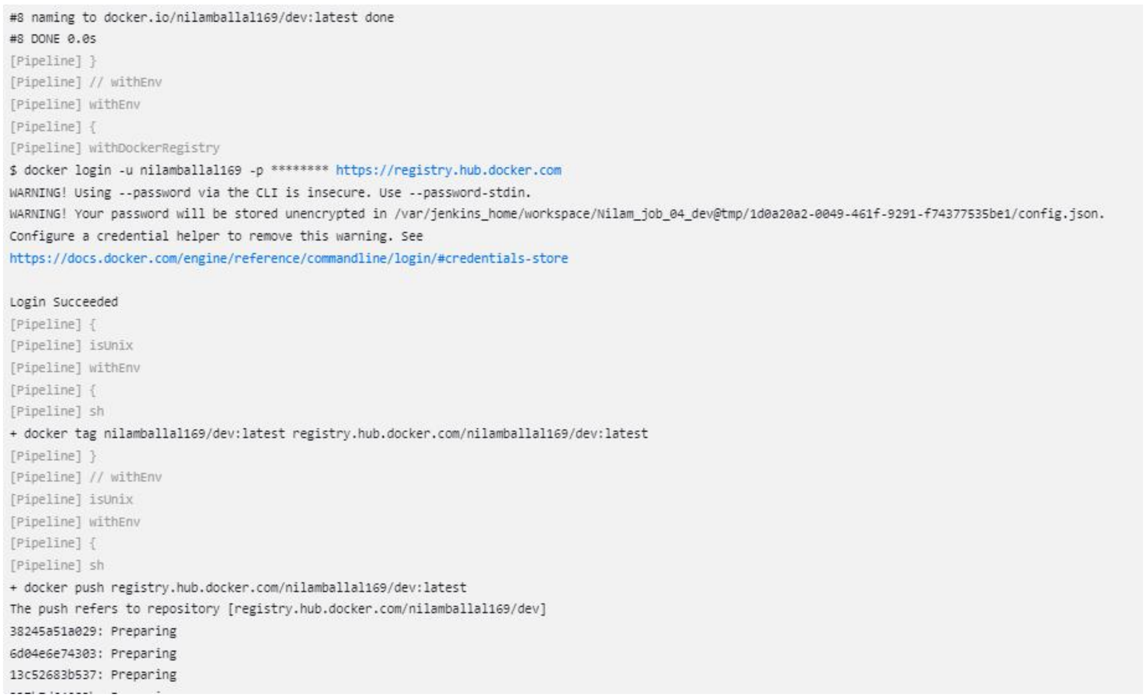
```
Started by user nilamballal
 > git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-fccce26d64167140832561e8dbe80b53/.git # timeout=10
Setting origin to https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
 > git config remote.origin.url https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git # timeout=10
Fetching origin...
Fetching upstream changes from origin
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
 > git config --get remote.origin.url # timeout=10
using GIT_ASKPASS to set credentials
 > git fetch --tags --force --progress -- origin +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/dev
Seen branch in repository origin/main
Seen branch in repository origin/qa
Seen branch in repository origin/release
Seen 4 remote branches
Obtained Jenkinsfile from 68165edd42e6131d1ed0a43f31bcf8d9ce0e40c0
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/Nilam_job_04_dev
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential git_crede
Cloning the remote Git repository
Cloning with configured refspecs honoured and without tags
Cloning repository https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
 > git init /var/jenkins_home/workspace/Nilam_job_04_dev # timeout=10
Fetching upstream changes from https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
```



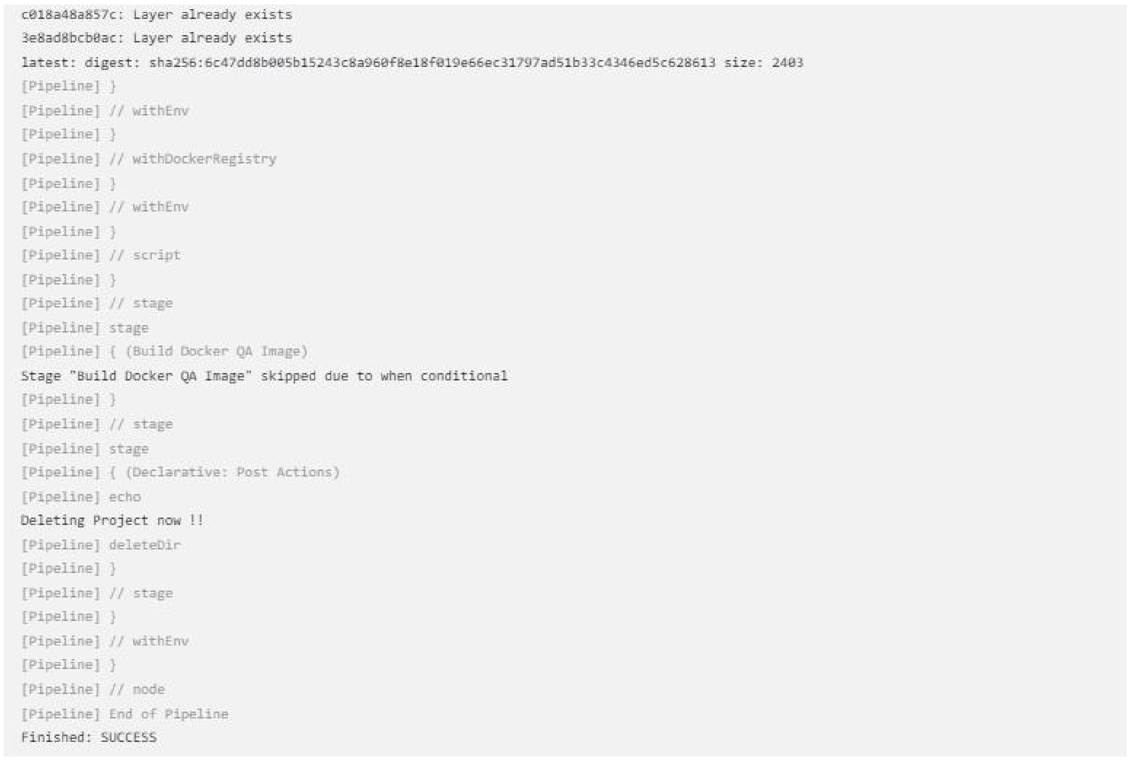
```
 > git init /var/jenkins_home/workspace/Nilam_job_04_dev # timeout=10
Fetching upstream changes from https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
using GIT_ASKPASS to set credentials
 > git fetch --no-tags --force --progress -- https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision 68165edd42e6131d1ed0a43f31bcf8d9ce0e40c0 (dev)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 68165edd42e6131d1ed0a43f31bcf8d9ce0e40c0 # timeout=10
Commit message: "Update Jenkinsfile"
 > git rev-list --no-walk 68165edd42e6131d1ed0a43f31bcf8d9ce0e40c0 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Docker Image Build IN Dev)
[Pipeline] script
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker build -t nilamballal169/dev:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 132B done
```



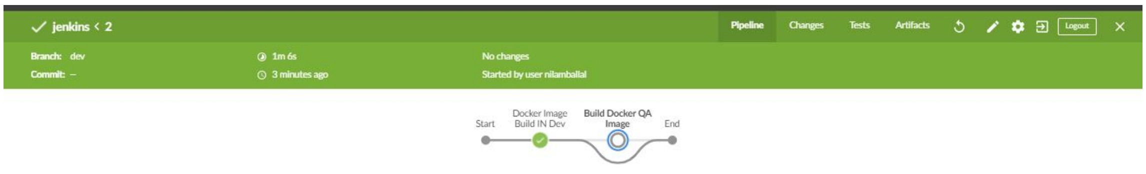
```
[Pipeline] stage
[Pipeline] { (Docker Image Build IN Dev)
[Pipeline] script
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker build -t nilamballal169/dev:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 132B done
#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/nginx:alpine
#2 DONE 0.9s

#3 [internal] load .dockerignore
#3 transferring context: 2B done
#3 DONE 0.0s

#4 [1/3] FROM docker.io/library/nginx:alpine@sha256:31bad00311cb5eeb8a6648beadcf67277a175da89989f14727420a80e2e76742
#4 DONE 0.0s

#5 [internal] load build context
#5 transferring context: 469B done
#5 DONE 0.0s
```



```
#8 naming to docker.io/nilamballal169/dev:latest done
#8 DONE 0.0s
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withDockerRegistry
$ docker login -u nilamballal169 -p ******** https://registry.hub.docker.com
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/jenkins_home/workspace/Nilam_job_04_dev@tmp/1d0a20a2-0049-461f-9291-f74377535be1/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker tag nilamballal169/dev:latest registry.hub.docker.com/nilamballal169/dev:latest
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker push registry.hub.docker.com/nilamballal169/dev:latest
The push refers to repository [registry.hub.docker.com/nilamballal169/dev]
38245a51a029: Preparing
6d04e6e74303: Preparing
13c52683b537: Preparing
```



```
c018a48a857c: Layer already exists
3e8ad8bcb0ac: Layer already exists
latest: digest: sha256:6c47dd8b005b15243c8a960f8e18f019e66ec31797ad51b33c4346ed5c628613 size: 2403
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Docker QA Image)
Stage "Build Docker QA Image" skipped due to when conditional
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Deleting Project now !!
[Pipeline] deleteDir
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



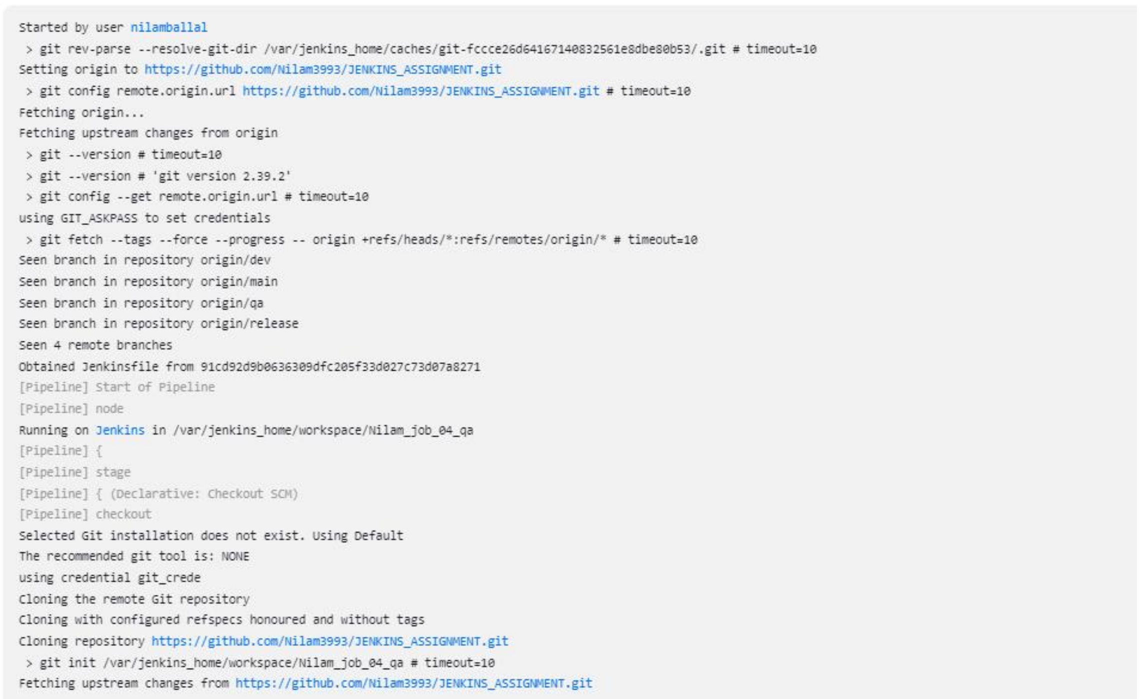
✓ jenkins < 2

Branch: dev
Commit: –

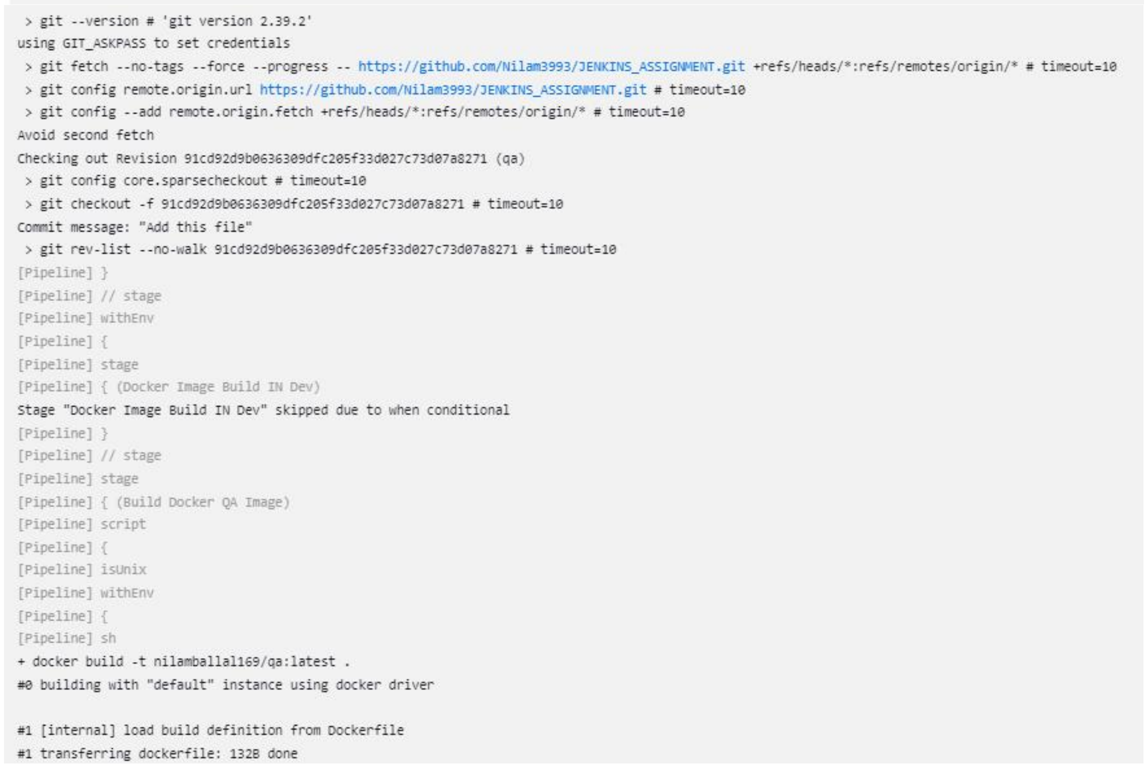
1m 6s
3 minutes ago

No changes
Started by user nilamballal

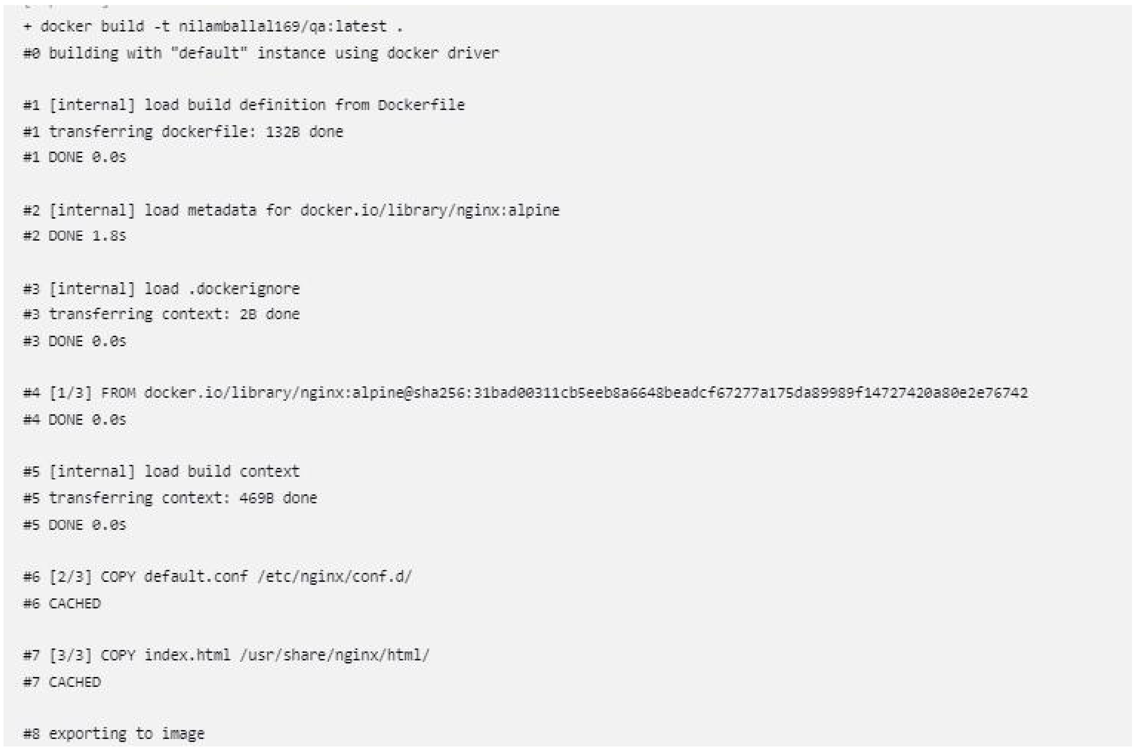
For qa



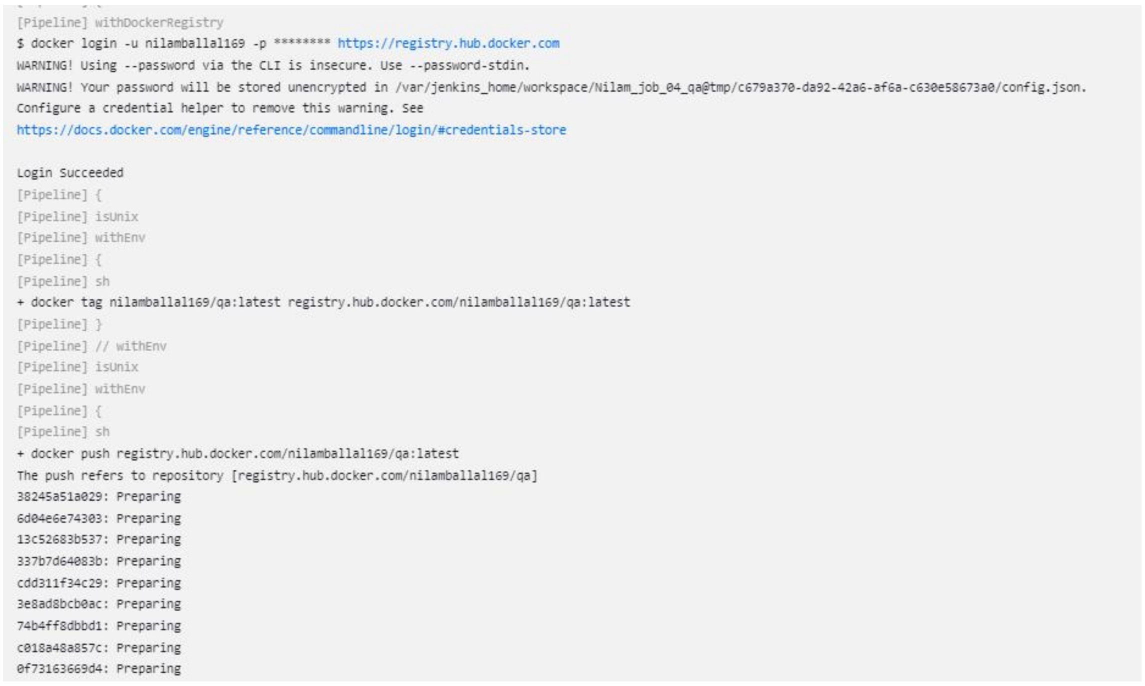
```
Started by user nilamballal
 > git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-fccce26d64167140832561e8dbe80b53/.git # timeout=10
Setting origin to https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
 > git config remote.origin.url https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git # timeout=10
Fetching origin...
Fetching upstream changes from origin
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
 > git config --get remote.origin.url # timeout=10
using GIT_ASKPASS to set credentials
 > git fetch --tags --force --progress -- origin +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/dev
Seen branch in repository origin/main
Seen branch in repository origin/qa
Seen branch in repository origin/release
Seen 4 remote branches
Obtained Jenkinsfile from 91cd92d9b0636309dfc205f33d027c73d07a8271
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/Nilam_job_04_qa
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential git_crede
Cloning the remote Git repository
Cloning with configured refspecs honoured and without tags
Cloning repository https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
 > git init /var/jenkins_home/workspace/Nilam_job_04_qa # timeout=10
Fetching upstream changes from https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git
```



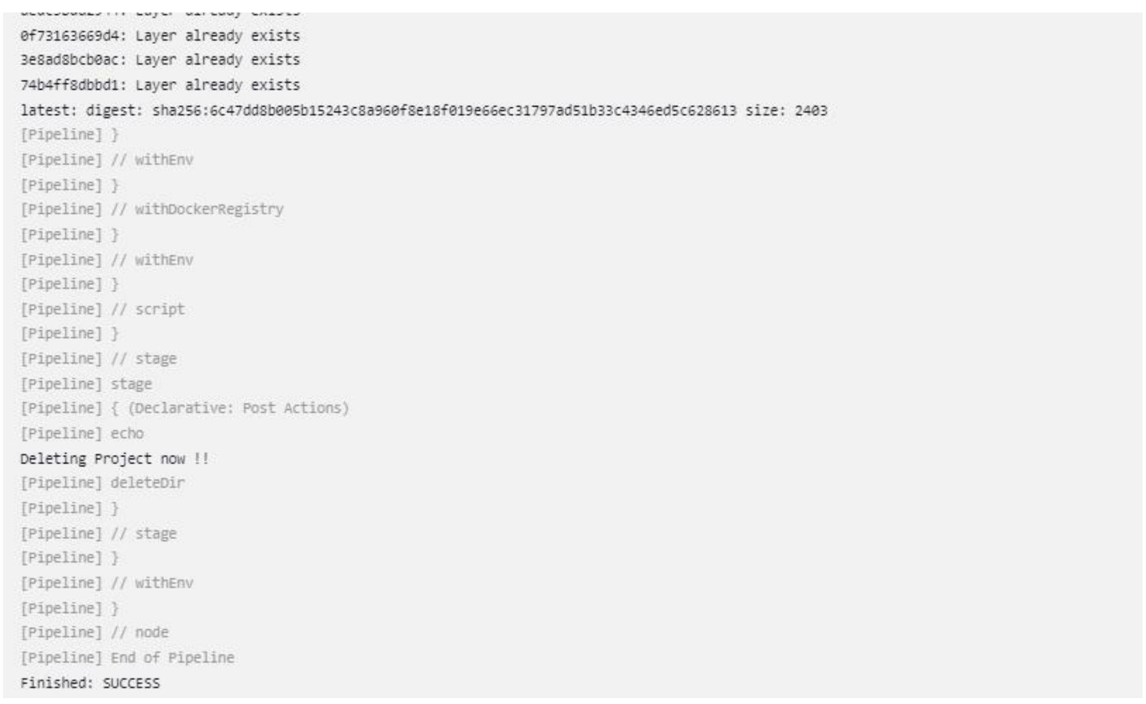
```
 > git --version # 'git version 2.39.2'
using GIT_ASKPASS to set credentials
 > git fetch --no-tags --force --progress -- https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Nilam3993/JENKINS_ASSIGNMENT.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision 91cd92d9b0636309dfc205f33d027c73d07a8271 (qa)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 91cd92d9b0636309dfc205f33d027c73d07a8271 # timeout=10
Commit message: "Add this file"
 > git rev-list --no-walk 91cd92d9b0636309dfc205f33d027c73d07a8271 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Docker Image Build IN Dev)
Stage "Docker Image Build IN Dev" skipped due to when conditional
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Docker QA Image)
[Pipeline] script
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker build -t nilamballal169/qa:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 132B done
```



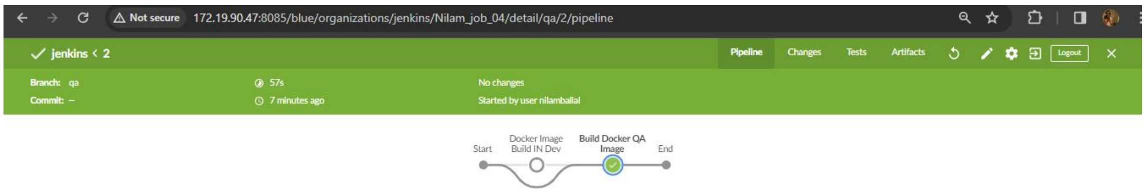
```
+ docker build -t nilamballal169/qa:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 132B done
#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/nginx:alpine
#2 DONE 1.8s

#3 [internal] load .dockerignore
#3 transferring context: 2B done
#3 DONE 0.0s

#4 [1/3] FROM docker.io/library/nginx:alpine@sha256:31bad00311cb5eeb8a6648beadcf67277a175da89989f14727420a80e2e76742
#4 DONE 0.0s

#5 [internal] load build context
#5 transferring context: 469B done
#5 DONE 0.0s

#6 [2/3] COPY default.conf /etc/nginx/conf.d/
#6 CACHED

#7 [3/3] COPY index.html /usr/share/nginx/html/
#7 CACHED

#8 exporting to image
```



```
[Pipeline] withDockerRegistry
$ docker login -u nilamballal169 -p ******** https://registry.hub.docker.com
WARNING! Using --password via the CLI is insecure. Use --password-stdin.
WARNING! Your password will be stored unencrypted in /var/jenkins_home/workspace/Nilam_job_04_qa@tmp/c679a370-da92-42a6-af6a-c630e58673a0/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker tag nilamballal169/qa:latest registry.hub.docker.com/nilamballal169/qa:latest
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker push registry.hub.docker.com/nilamballal169/qa:latest
The push refers to repository [registry.hub.docker.com/nilamballal169/qa]
38245a51a029: Preparing
6d04e6e74303: Preparing
13c52683b537: Preparing
337b7d64083b: Preparing
cdd311f34c29: Preparing
3e8ad8bcb0ac: Preparing
74b4ff8dbbd1: Preparing
c018a48a857c: Preparing
0f73163669d4: Preparing
```



```
0f73163669d4: Layer already exists
3e8ad8bcb0ac: Layer already exists
74b4ff8dbbd1: Layer already exists
latest: digest: sha256:6c47dd8b005b15243c8a960f8e18f019e66ec31797ad51b33c4346ed5c628613 size: 2403
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Deleting Project now !!
[Pipeline] deleteDir
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```



Not secure 172.19.90.47:8085/blue/organizations/jenkins/Nilam_job_04/detail/qa/2/pipeline

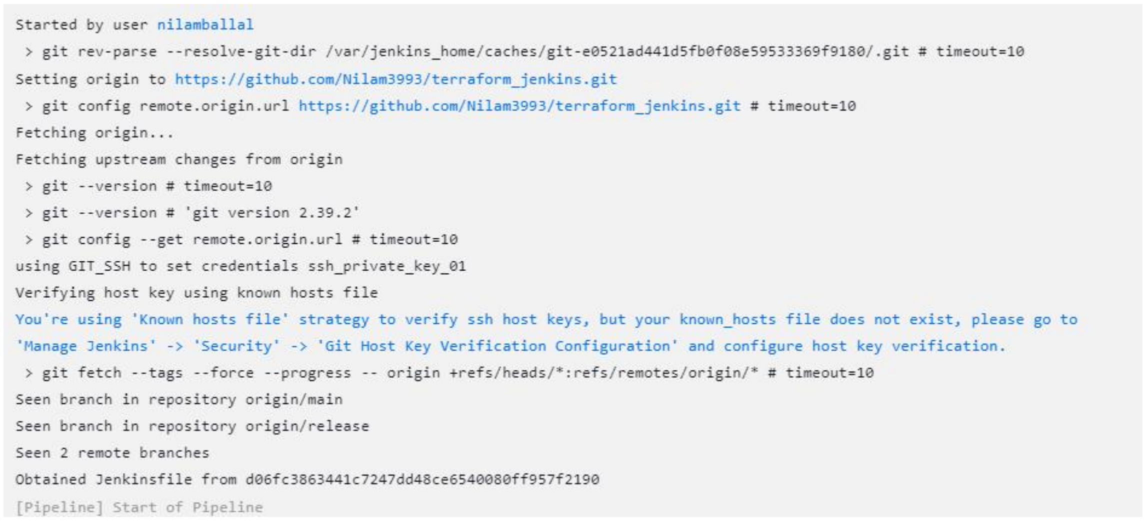
jenkins < 2

Branch: qa
Commit: –

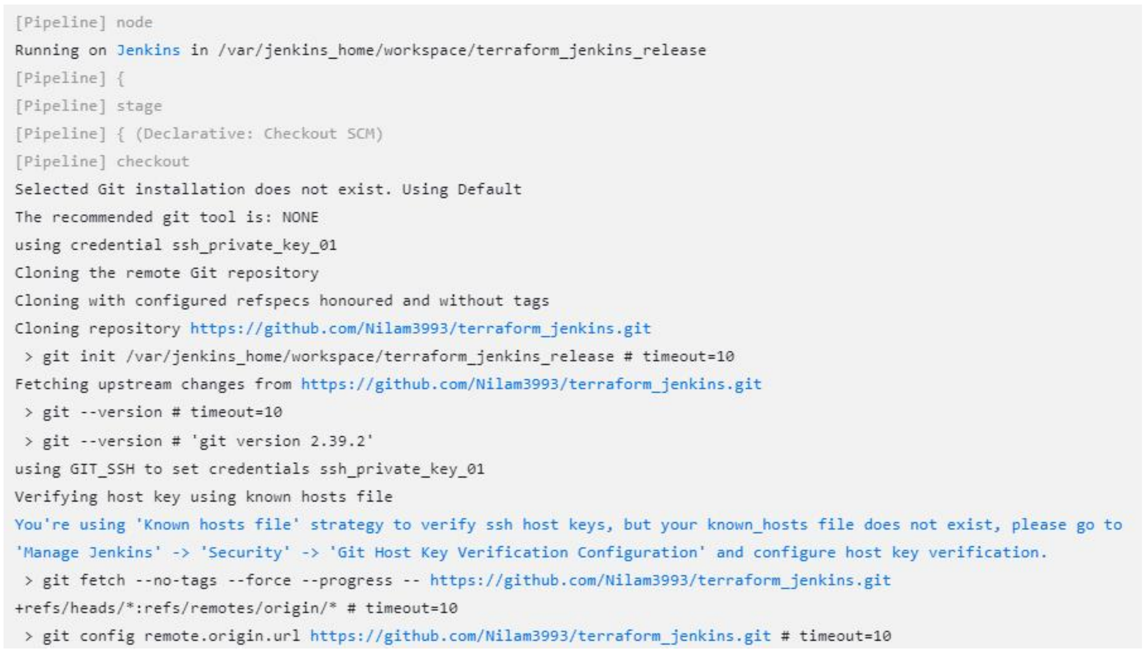
57s
7 minutes ago

No changes
Started by user nilamballal

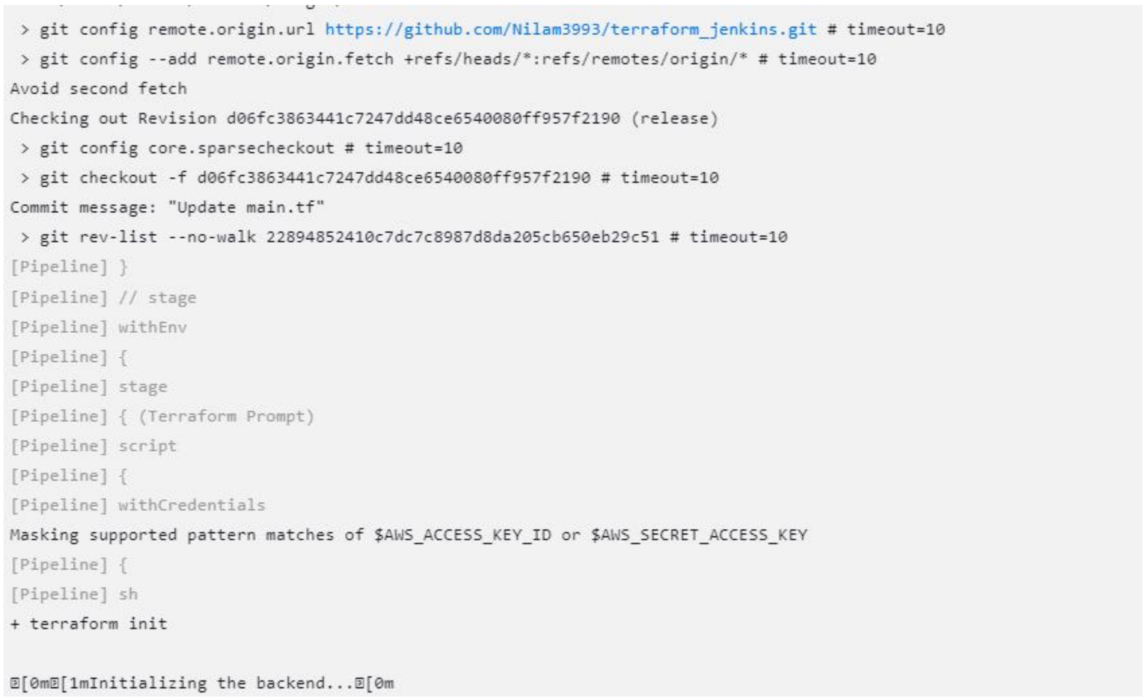
2. Design a Jenkins file to execute any Terraform code, prompting the user for two inputs: Terraform apply and Terraform destroy. Depending on the provided inputs, execute the corresponding Terraform command accordingly.



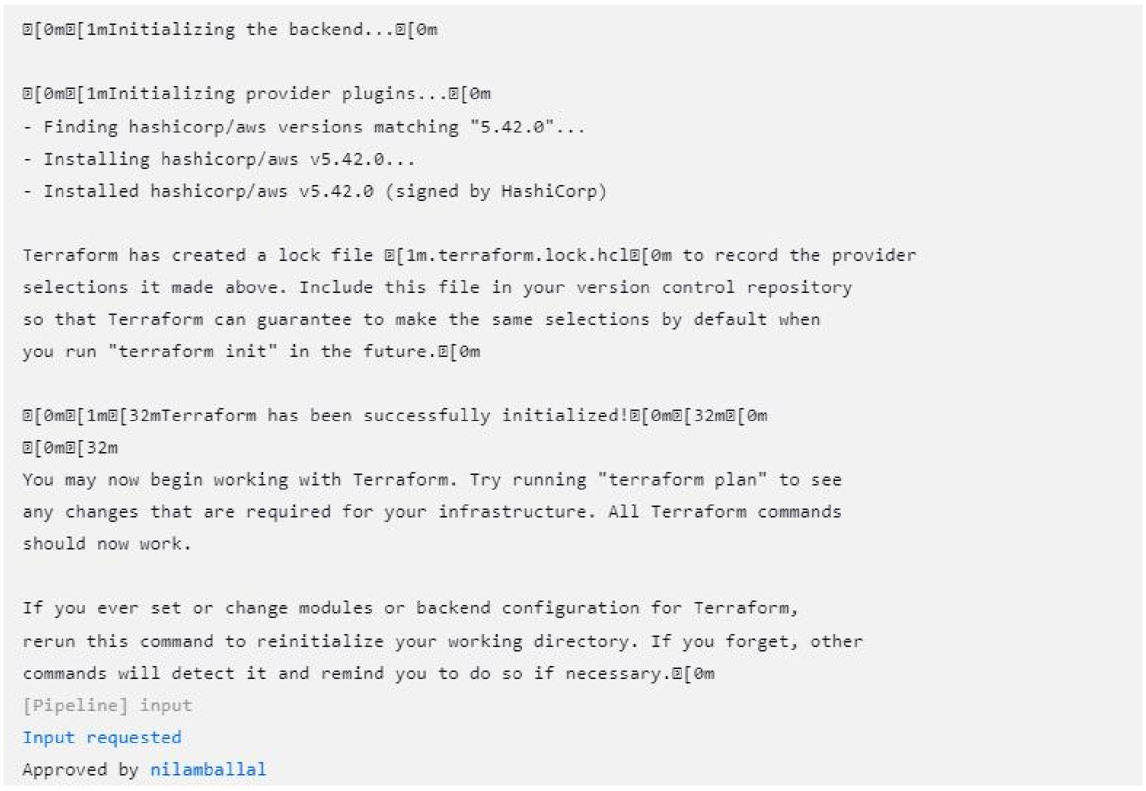
```
Started by user nilamballal
 > git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-e0521ad441d5fb0f08e59533369f9180/.git # timeout=10
Setting origin to https://github.com/Nilam3993/terraform_jenkins.git
 > git config remote.origin.url https://github.com/Nilam3993/terraform_jenkins.git # timeout=10
Fetching origin...
Fetching upstream changes from origin
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
 > git config --get remote.origin.url # timeout=10
using GIT_SSH to set credentials ssh_private_key_01
Verifying host key using known hosts file
You're using 'Known hosts file' strategy to verify ssh host keys, but your known_hosts file does not exist, please go to
'Manage Jenkins' -> 'Security' -> 'Git Host Key Verification Configuration' and configure host key verification.
 > git fetch --tags --force --progress -- origin +refs/heads/*:refs/remotes/origin/* # timeout=10
Seen branch in repository origin/main
Seen branch in repository origin/release
Seen 2 remote branches
Obtained Jenkinsfile from d06fc3863441c7247dd48ce6540080ff957f2190
[Pipeline] Start of Pipeline
```



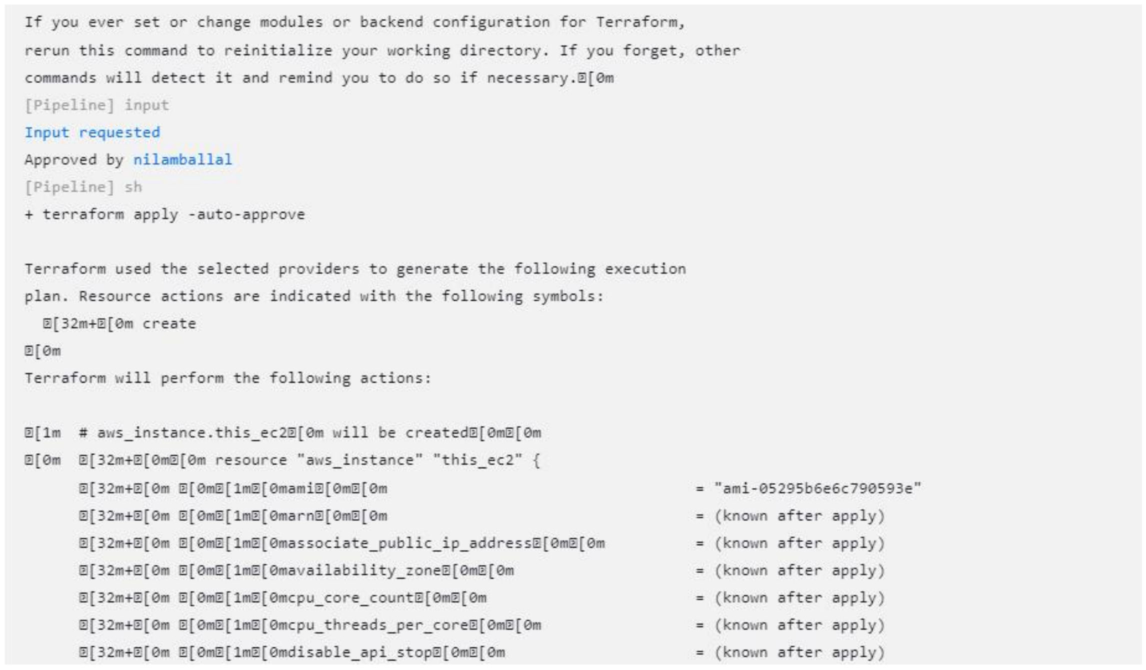
```
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/terraform_jenkins_release
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential ssh_private_key_01
Cloning the remote Git repository
Cloning with configured refspecs honoured and without tags
Cloning repository https://github.com/Nilam3993/terraform_jenkins.git
 > git init /var/jenkins_home/workspace/terraform_jenkins_release # timeout=10
Fetching upstream changes from https://github.com/Nilam3993/terraform_jenkins.git
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
using GIT_SSH to set credentials ssh_private_key_01
Verifying host key using known hosts file
You're using 'Known hosts file' strategy to verify ssh host keys, but your known_hosts file does not exist, please go to
'Manage Jenkins' -> 'Security' -> 'Git Host Key Verification Configuration' and configure host key verification.
 > git fetch --no-tags --force --progress -- https://github.com/Nilam3993/terraform_jenkins.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Nilam3993/terraform_jenkins.git # timeout=10
```



```
 > git config remote.origin.url https://github.com/Nilam3993/terraform_jenkins.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision d06fc3863441c7247dd48ce6540080ff957f2190 (release)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f d06fc3863441c7247dd48ce6540080ff957f2190 # timeout=10
Commit message: "Update main.tf"
 > git rev-list --no-walk 22894852410c7dc7c8987d8da205cb650eb29c51 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Terraform Prompt)
[Pipeline] script
[Pipeline] {
[Pipeline] withCredentials
Masking supported pattern matches of $AWS_ACCESS_KEY_ID or $AWS_SECRET_ACCESS_KEY
[Pipeline] {
[Pipeline] sh
+ terraform init

⊡[0m⊡[1mInitializing the backend...⊡[0m
```



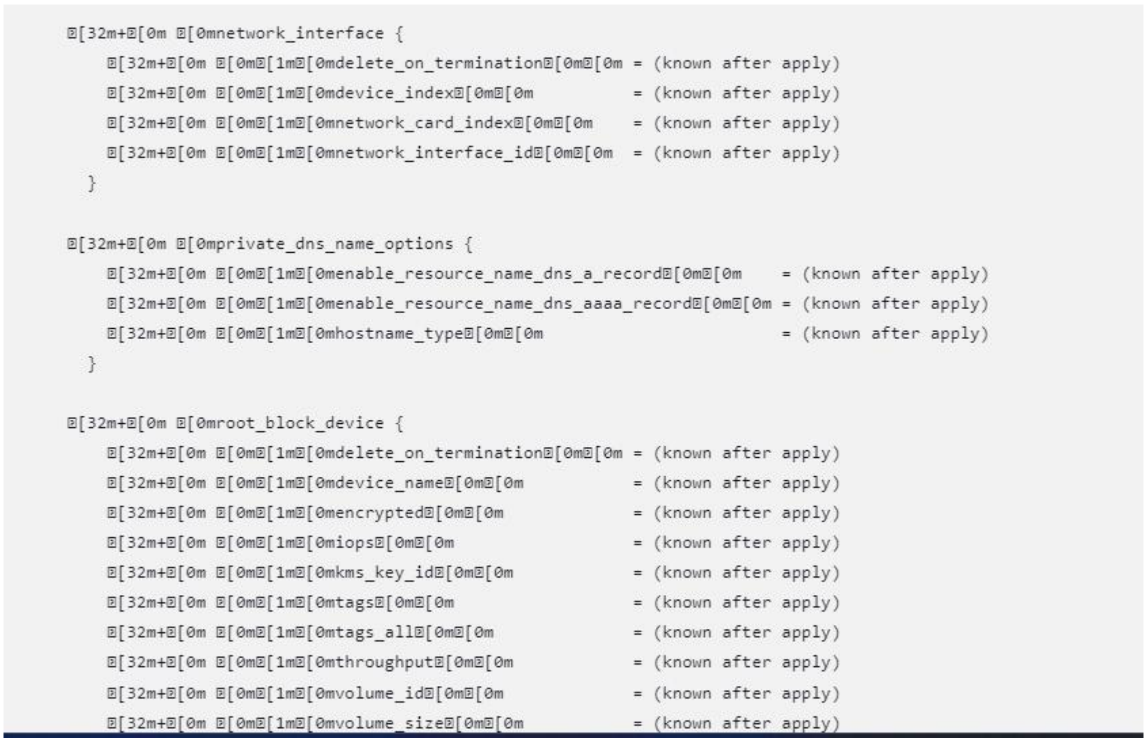
```
⍰[0m⍰[1mInitializing the backend...⍰[0m

⍰[0m⍰[1mInitializing provider plugins...⍰[0m
- Finding hashicorp/aws versions matching "5.42.0"...
- Installing hashicorp/aws v5.42.0...
- Installed hashicorp/aws v5.42.0 (signed by HashiCorp)

Terraform has created a lock file ⍰[1m.terraform.lock.hcl⍰[0m to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.⍰[0m

⍰[0m⍰[1m⍰[32mTerraform has been successfully initialized!⍰[0m⍰[32m⍰[0m
⍰[0m⍰[32m
You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.⍰[0m
[Pipeline] input
Input requested
Approved by nilamballal
```



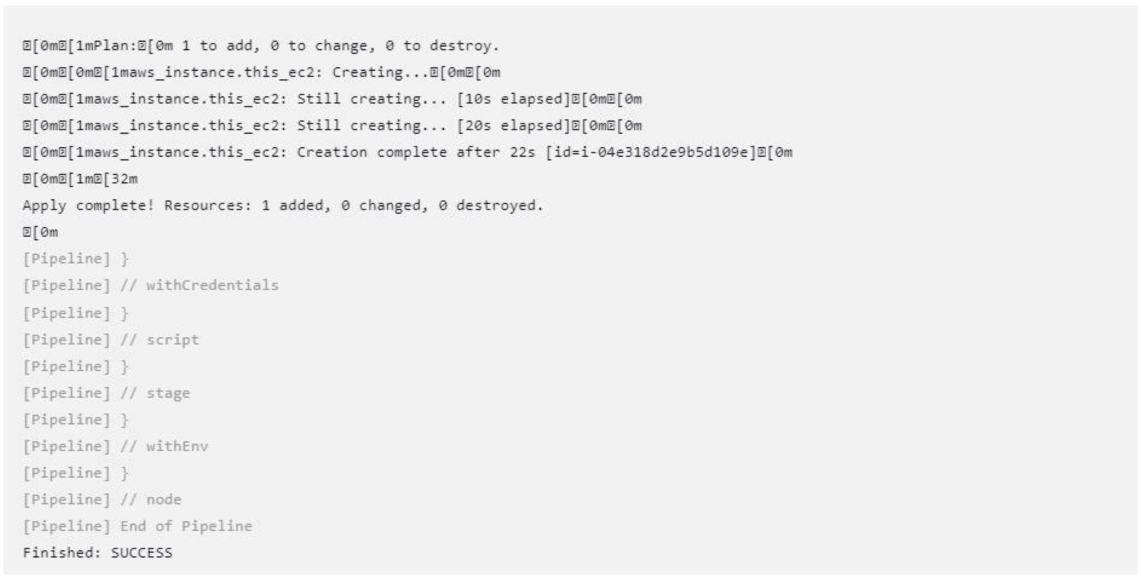
```
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.⍰[0m
[Pipeline] input
Input requested
Approved by nilamballal
[Pipeline] sh
+ terraform apply -auto-approve

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  ⍰[32m+⍰[0m create
⍰[0m
Terraform will perform the following actions:

⍰[1m  # aws_instance.this_ec2⍰[0m will be created⍰[0m⍰[0m
⍰[0m  ⍰[32m+⍰[0m⍰[0m resource "aws_instance" "this_ec2" {
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0mami⍰[0m⍰[0m                                   = "ami-05295b6e6c790593e"
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0marn⍰[0m⍰[0m                                   = (known after apply)
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0massociate_public_ip_address⍰[0m⍰[0m           = (known after apply)
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0mavailability_zone⍰[0m⍰[0m                     = (known after apply)
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0mcpu_core_count⍰[0m⍰[0m                        = (known after apply)
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0mcpu_threads_per_core⍰[0m⍰[0m                  = (known after apply)
      ⍰[32m+⍰[0m ⍰[0m⍰[1m⍰[0mdisable_api_stop⍰[0m⍰[0m                      = (known after apply)
```

```
⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0muser_data_replace_on_change⊠[0m⊠[0m          = false
⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mvpc_security_group_ids⊠[0m⊠[0m               = (known after apply)

⊠[32m+⊠[0m ⊠[0mcapacity_reservation_specification {
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mcapacity_reservation_preference⊠[0m⊠[0m = (known after apply)

    ⊠[32m+⊠[0m ⊠[0mcapacity_reservation_target {
        ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mcapacity_reservation_id⊠[0m⊠[0m                 = (known after apply)
        ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mcapacity_reservation_resource_group_arn⊠[0m⊠[0m = (known after apply)
      }
  }

⊠[32m+⊠[0m ⊠[0mcpu_options {
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mamd_sev_snp⊠[0m⊠[0m      = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mcore_count⊠[0m⊠[0m       = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mthreads_per_core⊠[0m⊠[0m = (known after apply)
  }

⊠[32m+⊠[0m ⊠[0mebs_block_device {
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mdelete_on_termination⊠[0m⊠[0m = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mdevice_name⊠[0m⊠[0m           = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mencrypted⊠[0m⊠[0m             = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0miops⊠[0m⊠[0m                  = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mkms_key_id⊠[0m⊠[0m            = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0msnapshot_id⊠[0m⊠[0m           = (known after apply)
    ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mtags⊠[0m⊠[0m                  = (known after apply)
```

```
 ⊠[32m+⊠[0m ⊠[0menclave_options {
     ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0menabled⊠[0m⊠[0m = (known after apply)
   }

 ⊠[32m+⊠[0m ⊠[0mephemeral_block_device {
     ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mdevice_name⊠[0m⊠[0m  = (known after apply)
     ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mno_device⊠[0m⊠[0m    = (known after apply)
     ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mvirtual_name⊠[0m⊠[0m = (known after apply)
   }

 ⊠[32m+⊠[0m ⊠[0minstance_market_options {
     ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mmarket_type⊠[0m⊠[0m = (known after apply)

     ⊠[32m+⊠[0m ⊠[0mspot_options {
         ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0minstance_interruption_behavior⊠[0m⊠[0m = (known after apply)
         ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mmax_price⊠[0m⊠[0m                      = (known after apply)
         ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mspot_instance_type⊠[0m⊠[0m             = (known after apply)
         ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mvalid_until⊠[0m⊠[0m                    = (known after apply)
       }
   }

 ⊠[32m+⊠[0m ⊠[0mmaintenance_options {
     ⊠[32m+⊠[0m ⊠[0m⊠[1m⊠[0mauto_recovery⊠[0m⊠[0m = (known after apply)
   }
```



```
    [32m+[0m [0mnetwork_interface {
        [32m+[0m [0m[1m[0mdelete_on_termination[0m[0m = (known after apply)
        [32m+[0m [0m[1m[0mdevice_index[0m[0m          = (known after apply)
        [32m+[0m [0m[1m[0mnetwork_card_index[0m[0m    = (known after apply)
        [32m+[0m [0m[1m[0mnetwork_interface_id[0m[0m  = (known after apply)
      }

    [32m+[0m [0mprivate_dns_name_options {
        [32m+[0m [0m[1m[0menable_resource_name_dns_a_record[0m[0m    = (known after apply)
        [32m+[0m [0m[1m[0menable_resource_name_dns_aaaa_record[0m[0m = (known after apply)
        [32m+[0m [0m[1m[0mhostname_type[0m[0m                        = (known after apply)
      }

    [32m+[0m [0mroot_block_device {
        [32m+[0m [0m[1m[0mdelete_on_termination[0m[0m = (known after apply)
        [32m+[0m [0m[1m[0mdevice_name[0m[0m           = (known after apply)
        [32m+[0m [0m[1m[0mencrypted[0m[0m             = (known after apply)
        [32m+[0m [0m[1m[0miops[0m[0m                  = (known after apply)
        [32m+[0m [0m[1m[0mkms_key_id[0m[0m            = (known after apply)
        [32m+[0m [0m[1m[0mtags[0m[0m                  = (known after apply)
        [32m+[0m [0m[1m[0mtags_all[0m[0m              = (known after apply)
        [32m+[0m [0m[1m[0mthroughput[0m[0m            = (known after apply)
        [32m+[0m [0m[1m[0mvolume_id[0m[0m             = (known after apply)
        [32m+[0m [0m[1m[0mvolume_size[0m[0m           = (known after apply)
```



```
[0m[1mPlan:[0m 1 to add, 0 to change, 0 to destroy.
[0m[0m[1maws_instance.this_ec2: Creating...[0m[0m
[0m[1maws_instance.this_ec2: Still creating... [10s elapsed][0m[0m
[0m[1maws_instance.this_ec2: Still creating... [20s elapsed][0m[0m
[0m[1maws_instance.this_ec2: Creation complete after 22s [id=i-04e318d2e9b5d109e][0m
[0m[1m[32m
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
[0m
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```