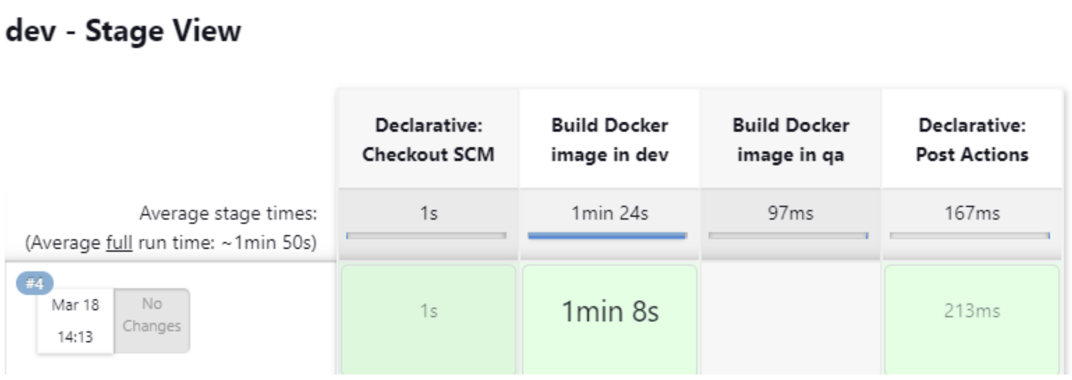
1. Create a Jenkins file to construct and upload Docker images to your Docker-hub registries. Ensure that when the branch is 'dev', the image is constructed and uploaded to the DEV Docker-hub registry. Similarly, when the branch is 'QA', it should be sent to the QA Docker-hub registry.

   Running pipeline in dev branch

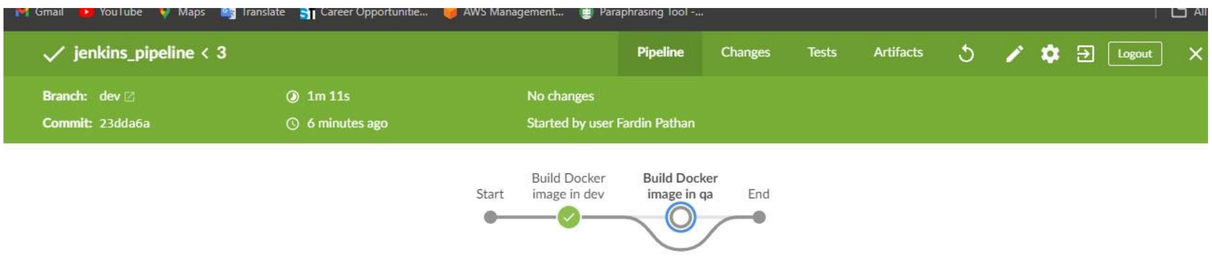
```
Started by user Fardin Pathan
 > git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-
ce6ba13e862e5508b6e64ebfef64b7bd/.git # timeout=10
Setting origin to https://github.com/Fardin31/Jenkins_Assignments_02.git
 > git config remote.origin.url https://github.com/Fardin31/Jenkins_Assignments_02.git # 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 GIT_CRED
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/dev
Seen branch in repository origin/master
Seen branch in repository origin/qa
Seen 3 remote branches
Obtained Jenkinsfile from 23dda6a3745666e331479edd1c3ab5f01dd7c18a
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/jenkins_pipeline_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_CRED
Cloning the remote Git repository
Cloning with configured refspecs honoured and without tags
Cloning repository https://github.com/Fardin31/Jenkins_Assignments_02.git
 > git init /var/jenkins_home/workspace/jenkins_pipeline_dev # timeout=10
Fetching upstream changes from https://github.com/Fardin31/Jenkins_Assignments_02.git
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
using GIT_SSH to set credentials GIT_CRED
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/Fardin31/Jenkins_Assignments_02.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Fardin31/Jenkins_Assignments_02.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision 23dda6a3745666e331479edd1c3ab5f01dd7c18a (dev)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 23dda6a3745666e331479edd1c3ab5f01dd7c18a # timeout=10
Commit message: "jekinsfile is added"
 > git rev-list --no-walk 23dda6a3745666e331479edd1c3ab5f01dd7c18a # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build Docker image in dev)
[Pipeline] script
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker build -t fardin31/dev:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 131B done
#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/nginx:alpine
#2 DONE 12.4s

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

#4 [1/3] FROM
docker.io/library/nginx:alpine@sha256:02d8d94023878cedf3e3acc55372932a9ba1478b6e2f3357786d
916c2af743ba
#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
#8 exporting layers done
#8 writing image sha256:f5864ebed9344b30cd5d36798f53ceb184a249b9aea70dfb408a9b901b1c3bde
done
#8 naming to docker.io/fardin31/dev:latest done
#8 DONE 0.0s
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withDockerRegistry
$ docker login -u fardin31 -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/jenkins_pipeline_dev@tmp/f2ab7994-94e9-4d8d-a127-
30c0212741d4/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 fardin31/dev:latest registry.hub.docker.com/fardin31/dev:latest
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker push registry.hub.docker.com/fardin31/dev:latest
The push refers to repository [registry.hub.docker.com/fardin31/dev]
74538962c069: Preparing
07214b638128: Preparing
13c52683b537: Preparing
337b7d64083b: Preparing
cdd311f34c29: Preparing
3e8ad8bcb0ac: Preparing
74b4ff8dbbd1: Preparing
c018a48a857c: Preparing
0f73163669d4: Preparing
aedc3bda2944: Preparing
74b4ff8dbbd1: Waiting
c018a48a857c: Waiting
0f73163669d4: Waiting
3e8ad8bcb0ac: Waiting
aedc3bda2944: Waiting
cdd311f34c29: Layer already exists
337b7d64083b: Layer already exists
13c52683b537: Layer already exists
74538962c069: Layer already exists
07214b638128: Layer already exists
c018a48a857c: Layer already exists
74b4ff8dbbd1: Layer already exists
3e8ad8bcb0ac: Layer already exists
aedc3bda2944: Layer already exists
0f73163669d4: Layer already exists
latest: digest: sha256:b6198eeaea18a25b820b15508547821cf70408ae41d5304643b6712c566a3333
size: 2403
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // script
[Pipeline] }
```

```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Docker image in qa)
Stage "Build Docker image in qa" 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
```



dev - Stage View

	Declarative: Checkout SCM	Build Docker image in dev	Build Docker image in qa	Declarative: Post Actions
Average stage times: (Average full run time: ~1min 50s)	1s	1min 24s	97ms	167ms
#4 Mar 18 14:13 No Changes	1s	1min 8s		213ms



jenkins_pipeline < 3

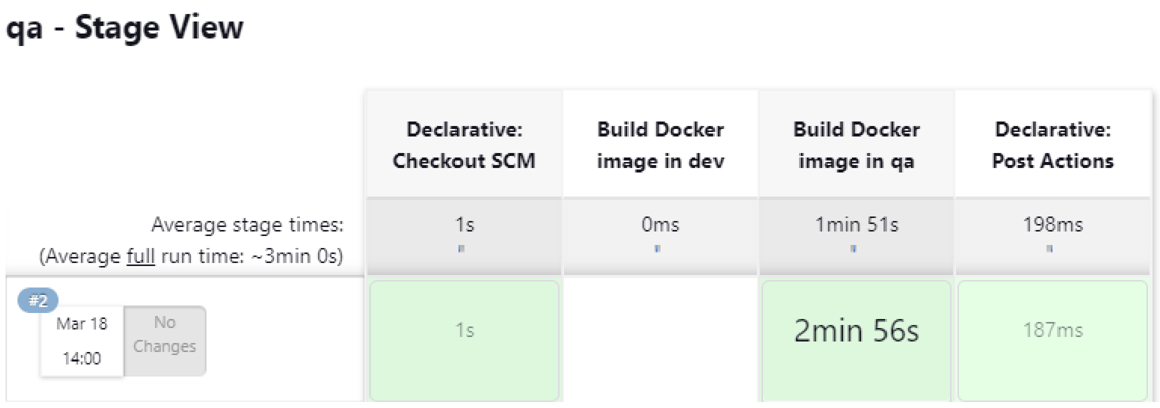
Branch: dev
Commit: 23dda6a
1m 11s
6 minutes ago
No changes
Started by user Fardin Pathan

Running pipeline in qa branch

```
Started by user Fardin Pathan
 > git rev-parse --resolve-git-dir /var/jenkins_home/caches/git-
ce6ba13e862e5508b6e64ebfef64b7bd/.git # timeout=10
Setting origin to https://github.com/Fardin31/Jenkins_Assignments_02.git
 > git config remote.origin.url https://github.com/Fardin31/Jenkins_Assignments_02.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 GIT_CRED
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/dev
Seen branch in repository origin/master
Seen branch in repository origin/qa
Seen 3 remote branches
Obtained Jenkinsfile from 05a6187e7d7b97aa1dd4df2625980106c1cf1dfd
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/jenkins_pipeline_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_CRED
Cloning the remote Git repository
Cloning with configured refspecs honoured and without tags
Cloning repository https://github.com/Fardin31/Jenkins_Assignments_02.git
 > git init /var/jenkins_home/workspace/jenkins_pipeline_qa # timeout=10
Fetching upstream changes from https://github.com/Fardin31/Jenkins_Assignments_02.git
 > git --version # timeout=10
 > git --version # 'git version 2.39.2'
using GIT_SSH to set credentials GIT_CRED
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/Fardin31/Jenkins_Assignments_02.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Fardin31/Jenkins_Assignments_02.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
Checking out Revision 05a6187e7d7b97aa1dd4df2625980106c1cf1dfd (qa)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 05a6187e7d7b97aa1dd4df2625980106c1cf1dfd # timeout=10
Commit message: "jenkinsfile is added"
 > git rev-list --no-walk 05a6187e7d7b97aa1dd4df2625980106c1cf1dfd # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build Docker image in dev)
```

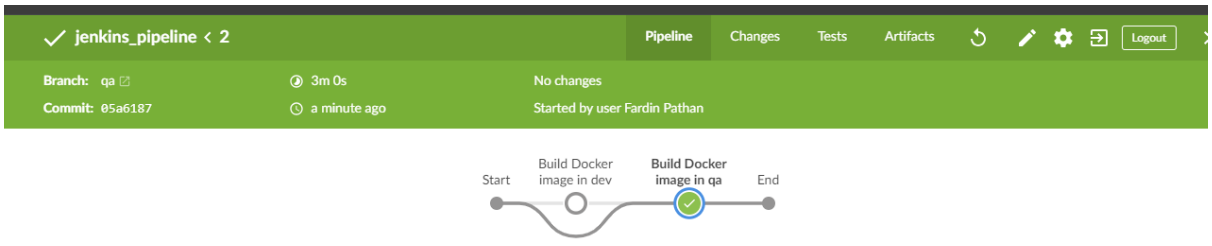
```
Stage "Build Docker image in dev" skipped due to when conditional
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build Docker image in qa)
[Pipeline] script
[Pipeline] {
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker build -t fardin31/qa:latest .
#0 building with "default" instance using docker driver

#1 [internal] load build definition from Dockerfile
#1 transferring dockerfile: 131B done
#1 DONE 0.0s

#2 [internal] load metadata for docker.io/library/nginx:alpine
#2 DONE 0.7s

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

#4 [1/3] FROM
docker.io/library/nginx:alpine@sha256:02d8d94023878cedf3e3acc55372932a9ba1478b6e2f3357786d
916c2af743ba
#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
#8 exporting layers done
#8 writing image sha256:f5864ebed9344b30cd5d36798f53ceb184a249b9aea70dfb408a9b901b1c3bde
done
#8 naming to docker.io/fardin31/qa:latest done
#8 DONE 0.0s
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withDockerRegistry
$ docker login -u fardin31 -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/jenkins_pipeline_qa@tmp/4fb0d04c-79a9-4800-8d4c-
1ebc062f3242/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 fardin31/qa:latest registry.hub.docker.com/fardin31/qa:latest
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] isUnix
[Pipeline] withEnv
[Pipeline] {
[Pipeline] sh
+ docker push registry.hub.docker.com/fardin31/qa:latest
The push refers to repository [registry.hub.docker.com/fardin31/qa]
74538962c069: Preparing
07214b638128: Preparing
13c52683b537: Preparing
337b7d64083b: Preparing
cdd311f34c29: Preparing
3e8ad8bcb0ac: Preparing
74b4ff8dbbd1: Preparing
c018a48a857c: Preparing
3e8ad8bcb0ac: Waiting
74b4ff8dbbd1: Waiting
c018a48a857c: Waiting
0f73163669d4: Preparing
aedc3bda2944: Preparing
0f73163669d4: Waiting
aedc3bda2944: Waiting
74538962c069: Mounted from fardin31/dev
07214b638128: Mounted from fardin31/dev
cdd311f34c29: Mounted from fardin31/dev
13c52683b537: Mounted from fardin31/dev
337b7d64083b: Mounted from fardin31/dev
0f73163669d4: Mounted from fardin31/dev
c018a48a857c: Mounted from fardin31/dev
3e8ad8bcb0ac: Mounted from fardin31/dev
74b4ff8dbbd1: Mounted from fardin31/dev
aedc3bda2944: Mounted from fardin31/dev
latest: digest: sha256:b6198eeaea18a25b820b15508547821cf70408ae41d5304643b6712c566a3333
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
```



qa - Stage View

	Declarative: Checkout SCM	Build Docker image in dev	Build Docker image in qa	Declarative: Post Actions
Average stage times: (Average full run time: ~3min 0s)	1s	0ms	1min 51s	198ms
#2 Mar 18 14:00 No Changes	1s		2min 56s	187ms

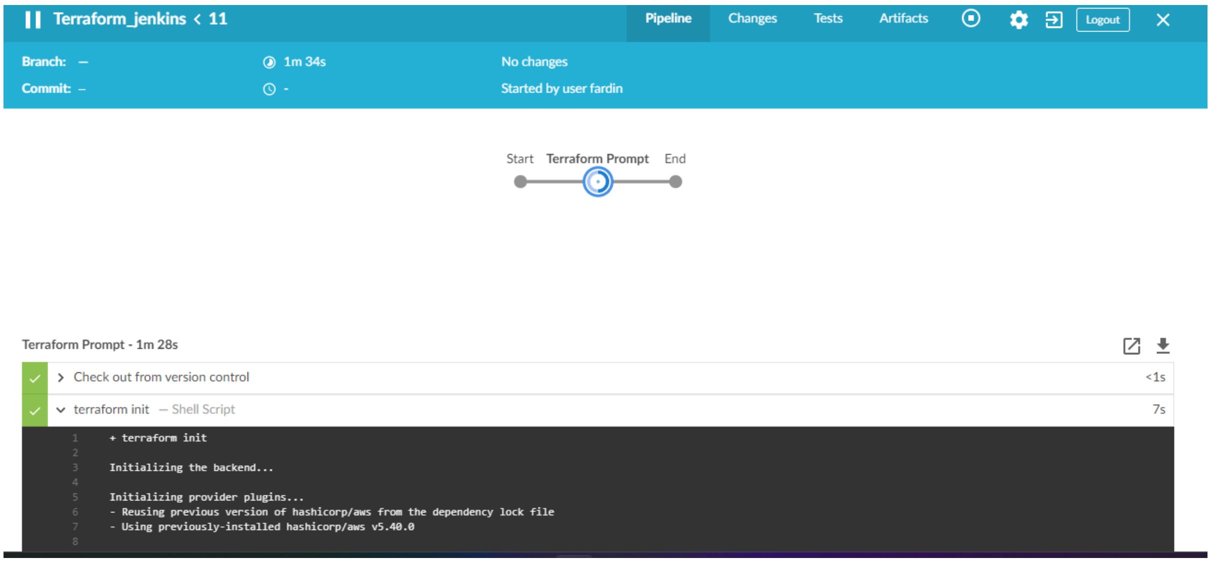


jenkins_pipeline < 2

Branch: qa
Commit: 05a6187
3m 0s
a minute ago
No changes
Started by user Fardin Pathan

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.

Using Apply Option



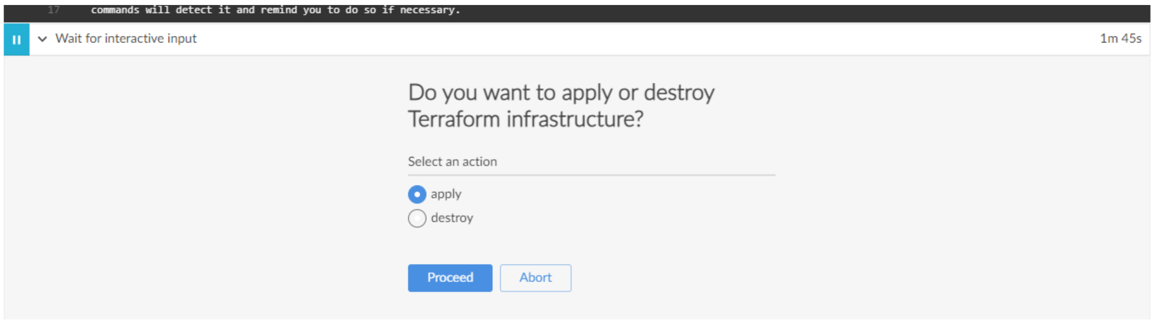
Branch: –
Commit: –
1m 34s
No changes
Started by user fardin

Terraform Prompt - 1m 28s

Check out from version control <1s

terraform init — Shell Script 7s

```
+ terraform init

Initializing the backend...

Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.40.0

```



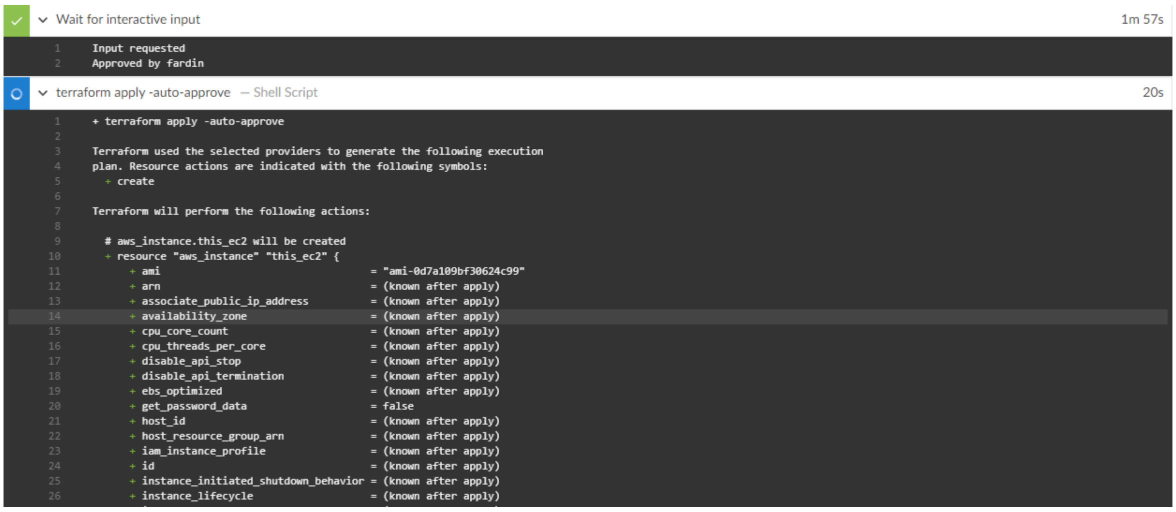
```
commands will detect it and remind you to do so if necessary.
```

Wait for interactive input 1m 45s

Do you want to apply or destroy Terraform infrastructure?

Select an action

[✓ apply]
destroy



Wait for interactive input 1m 57s

```
Input requested
Approved by fardin
```

terraform apply -auto-approve — Shell Script 20s

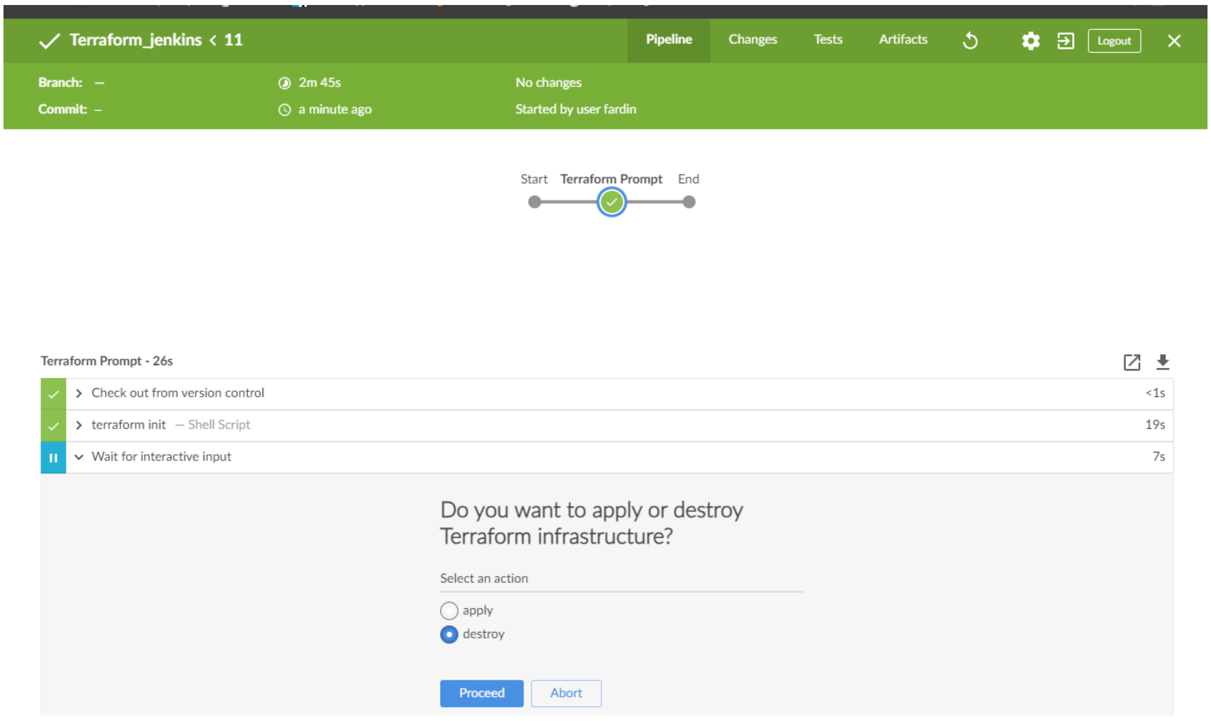
```
+ terraform apply -auto-approve

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.this_ec2 will be created
  + resource "aws_instance" "this_ec2" {
      + ami                                  = "ami-0d7a109bf30624c99"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_lifecycle                   = (known after apply)
```

3. Started by user fardin
4. Obtained Jenkinsfile from git https://github.com/Fardin31/Terraform_with_jenkins.git
5. [Pipeline] Start of Pipeline
6. [Pipeline] node
7. Running on Jenkins in /var/lib/jenkins/workspace/Terraform_jenkins
8. [Pipeline] {
9. [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_CRED
Cloning the remote Git repository
Cloning repository https://github.com/Fardin31/Terraform_with_jenkins.git
 > git init /var/lib/jenkins/workspace/Terraform_jenkins # timeout=10
Fetching upstream changes from https://github.com/Fardin31/Terraform_with_jenkins.git
 > git --version # timeout=10
 > git --version # 'git version 2.34.1'
using GIT_SSH to set credentials
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 -- https://github.com/Fardin31/Terraform_with_jenkins.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Fardin31/Terraform_with_jenkins.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision d1c86388759bfcd56e6e609451e01ea3518db24e (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f d1c86388759bfcd56e6e609451e01ea3518db24e # timeout=10
Commit message: "jenkinsfile has been added"
 > git rev-list --no-walk 1d0e656ea28e614ae188a1129b9b89c4f7e84bd9 # 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 provider plugins... [0m
- Finding hashicorp/aws versions matching "5.40.0"...
- Installing hashicorp/aws v5.40.0...
- Installed hashicorp/aws v5.40.0 (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 fardin
[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    [32m+  [0m  [0m resource "aws_instance" "this_ec2" {
      [32m+  [0m  [0m ami                          = "ami-0d7a109bf30624c99"
      [32m+  [0m  [0m arn                          = (known after apply)
      [32m+  [0m  [0m associate_public_ip_address       = (known after apply)
      [32m+  [0m  [0m availability_zone               = (known after apply)
      [32m+  [0m  [0m cpu_core_count                 = (known after apply)
      [32m+  [0m  [0m cpu_threads_per_core             = (known after apply)
      [32m+  [0m  [0m disable_api_stop                = (known after apply)
      [32m+  [0m  [0m disable_api_termination           = (known after apply)
      [32m+  [0m  [0m ebs_optimized                  = (known after apply)
      [32m+  [0m  [0m get_password_data                = false
      [32m+  [0m  [0m host_id                      = (known after apply)
      [32m+  [0m  [0m host_resource_group_arn           = (known after apply)
      [32m+  [0m  [0m iam_instance_profile             = (known after apply)
      [32m+  [0m  [0m id                         = (known after apply)
      [32m+  [0m  [0m instance_initiated_shutdown_behavior = (known after apply)
      [32m+  [0m  [0m instance_lifecycle               = (known after apply)
      [32m+  [0m  [0m instance_state                 = (known after apply)
      [32m+  [0m  [0m instance_type                  = "t2.micro"
      [32m+  [0m  [0m ipv6_address_count               = (known after apply)
      [32m+  [0m  [0m ipv6_addresses                 = (known after apply)
      [32m+  [0m  [0m key_name                      = "fardin12"
      [32m+  [0m  [0m monitoring                    = (known after apply)
      [32m+  [0m  [0m outpost_arn                   = (known after apply)
      [32m+  [0m  [0m password_data                  = (known after apply)
      [32m+  [0m  [0m placement_group                 = (known after apply)
      [32m+  [0m  [0m placement_partition_number         = (known after apply)
      [32m+  [0m  [0m primary_network_interface_id        = (known after apply)
      [32m+  [0m  [0m private_dns                   = (known after apply)
      [32m+  [0m  [0m private_ip                    = (known after apply)
      [32m+  [0m  [0m public_dns                    = (known after apply)
      [32m+  [0m  [0m public_ip                     = (known after apply)
      [32m+  [0m  [0m secondary_private_ips            = (known after apply)
      [32m+  [0m  [0m security_groups                 = (known after apply)
      [32m+  [0m  [0m source_dest_check               = true
      [32m+  [0m  [0m spot_instance_request_id          = (known after apply)
      [32m+  [0m  [0m subnet_id                     = (known after apply)
      [32m+  [0m  [0m tags_all                      = (known after apply)
      [32m+  [0m  [0m tenancy                      = (known after apply)
      [32m+  [0m  [0m user_data                     = (known after apply)
      [32m+  [0m  [0m user_data_base64                 = (known after apply)
      [32m+  [0m  [0m user_data_replace_on_change       = false
      [32m+  [0m  [0m vpc_security_group_ids            = (known after apply)
    }

  [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: Still creating... [30s elapsed]  [0m  [0m
  [0m  [1maws_instance.this_ec2: Creation complete after 37s [id=i-
0c69417c19081d5dd]  [0m
  [0m  [1m  [32m
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
  [0m
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] deleteDir
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Using Destroy Option



Branch: —
Commit: —
2m 45s
a minute ago
No changes
Started by user fardin

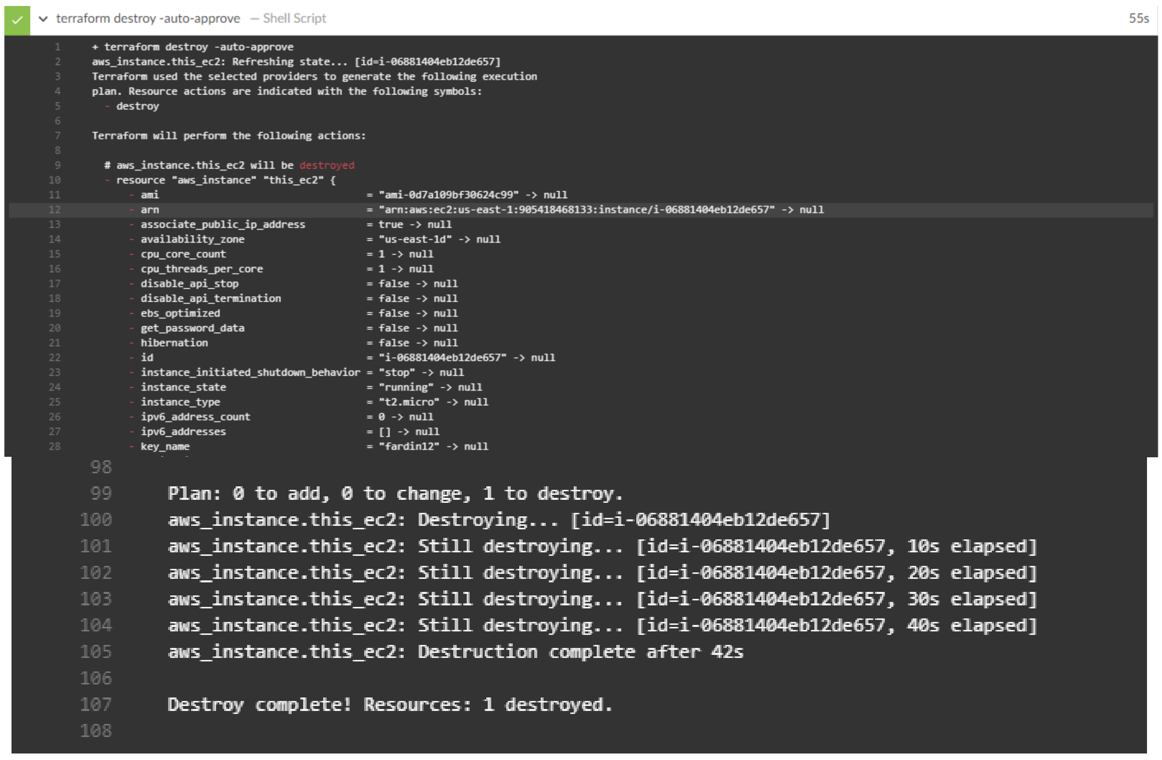
Terraform Prompt - 26s

✓ Check out from version control	<1s
✓ terraform init — Shell Script	19s
Wait for interactive input	7s

Do you want to apply or destroy Terraform infrastructure?

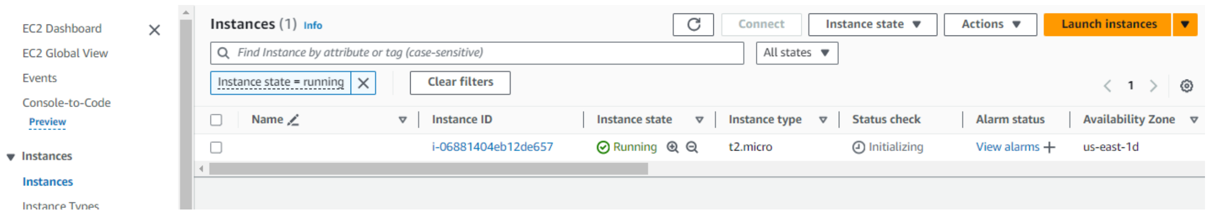
Select an action

apply [✓ destroy]



terraform destroy -auto-approve — Shell Script 55s

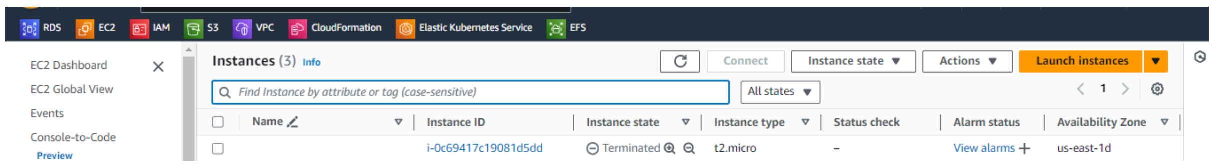
```
+ terraform destroy -auto-approve
aws_instance.this_ec2: Refreshing state... [id=i-06881404eb12de657]
Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  - destroy

Terraform will perform the following actions:

  # aws_instance.this_ec2 will be destroyed
  - resource "aws_instance" "this_ec2" {
      - ami                                  = "ami-0d7a109bf30624c99" -> null
      - arn                                  = "arn:aws:ec2:us-east-1:905418468133:instance/i-06881404eb12de657" -> null
      - associate_public_ip_address          = true -> null
      - availability_zone                    = "us-east-1d" -> null
      - cpu_core_count                       = 1 -> null
      - cpu_threads_per_core                 = 1 -> null
      - disable_api_stop                     = false -> null
      - disable_api_termination              = false -> null
      - ebs_optimized                        = false -> null
      - get_password_data                    = false -> null
      - hibernation                          = false -> null
      - id                                   = "i-06881404eb12de657" -> null
      - instance_initiated_shutdown_behavior = "stop" -> null
      - instance_state                       = "running" -> null
      - instance_type                        = "t2.micro" -> null
      - ipv6_address_count                   = 0 -> null
      - ipv6_addresses                       = [] -> null
      - key_name                             = "fardin12" -> null

Plan: 0 to add, 0 to change, 1 to destroy.
aws_instance.this_ec2: Destroying... [id=i-06881404eb12de657]
aws_instance.this_ec2: Still destroying... [id=i-06881404eb12de657, 10s elapsed]
aws_instance.this_ec2: Still destroying... [id=i-06881404eb12de657, 20s elapsed]
aws_instance.this_ec2: Still destroying... [id=i-06881404eb12de657, 30s elapsed]
aws_instance.this_ec2: Still destroying... [id=i-06881404eb12de657, 40s elapsed]
aws_instance.this_ec2: Destruction complete after 42s

Destroy complete! Resources: 1 destroyed.
```



Instances (1) Info

Instance state = running

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☐		i-06881404eb12de657	Running	t2.micro	Initializing	View alarms +	us-east-1d



Instances (3) Info

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
☐		i-0c69417c19081d5dd	Terminated	t2.micro	–	View alarms +	us-east-1d