kura labs

COHORT 3

Deployment #4

Kerri Smith
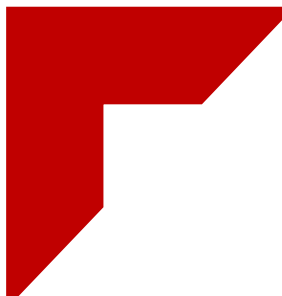
Table of Contents

Overview

This exercise demonstrated the steps for using Terraform as part of the deployment. It further extends the Pipeline from Deployment #3.

Steps

Jenkins Server setup

Existing Jenkins server from previous deployments was used

Install Terraform on Jenkins Server

Terraform was installed on the Jenkins server.

Configure credentials on Jenkins

AWS credentials were configured on Jenkins server

Create a Pipeline build in Jenkins

Pipeline build was created and run to test integration of Jenkins and Terraform. (see oldTerraform folder in repository)

This test infrastructure was created with an EC2 instance running the "url_shortener" application on "gunicorn". (see Old Pipeline diagram)
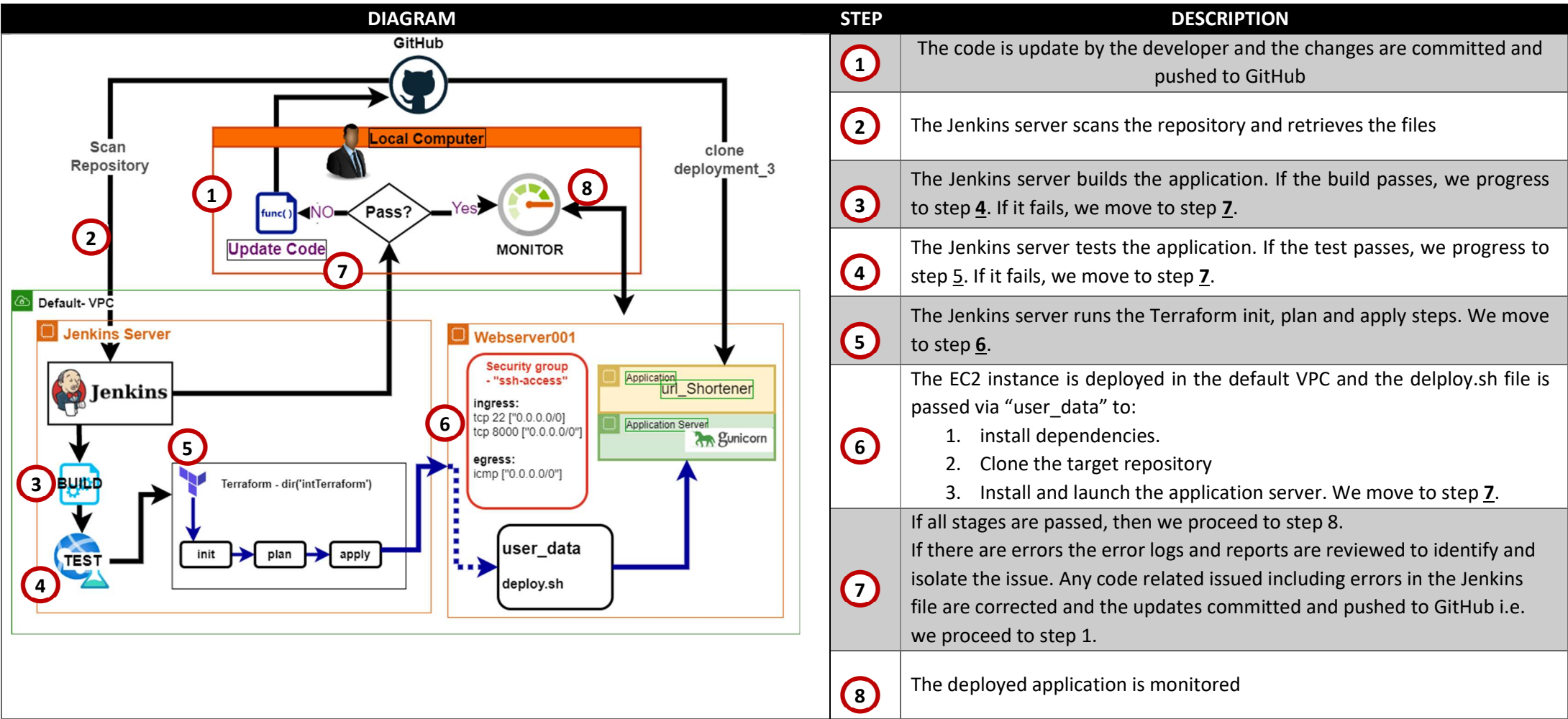
Add "Destroy" stage

A destroy stage was included in the Jenkins file to remove the infrastructure that was originally created.

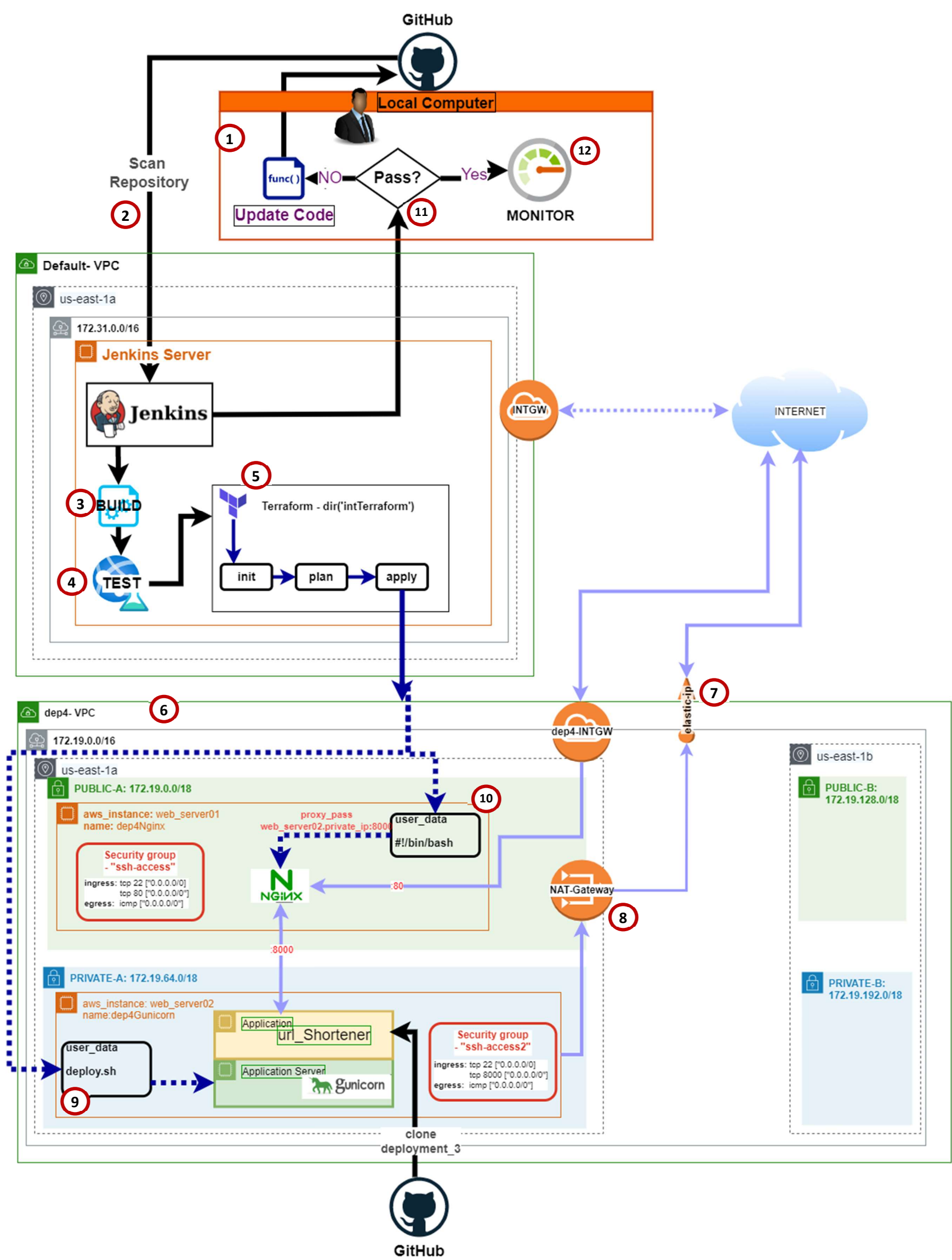### Create a new VPC and add Deployment 3

The intTerraform files were modified to create a new VPC and deploy the repository for Deployment 3 onto this infrastructure. (see New Pipeline diagram)

Old Pipeline

DIAGRAM	STEP	DESCRIPTION
	1	The code is update by the developer and the changes are committed and pushed to GitHub
	2	The Jenkins server scans the repository and retrieves the files
	3	The Jenkins server builds the application. If the build passes, we progress to step **4**. If it fails, we move to step **7**.
	4	The Jenkins server tests the application. If the test passes, we progress to step **5**. If it fails, we move to step **7**.
	5	The Jenkins server runs the Terraform init, plan and apply steps. We move to step **6**.
	6	The EC2 instance is deployed in the default VPC and the delploy.sh file is passed via "user_data" to: 1. install dependencies. 2. Clone the target repository 3. Install and launch the application server. We move to step **7**.
	7	If all stages are passed, then we proceed to step 8. If there are errors the error logs and reports are reviewed to identify and isolate the issue. Any code related issued including errors in the Jenkins file are corrected and the updates committed and pushed to GitHub i.e. we proceed to step 1.
	8	The deployed application is monitored

New Pipeline

The deployment environment used two Amazon Virtual Private Clouds (Amazon VPCs): The default VPC created with the AWS account and a custom VPC called "dep4-VPC". The custom VPC consists of four subnets (2 public and 2 private) split across two availability zones. An EC2 was deployed as the application server in one private subnet. Another EC2 was deployed in a public subnet to run a proxy server.

STEP	DESCRIPTION
1	The code is update by the developer and the changes are committed and pushed to GitHub
2	The Jenkins server scans the repository and retrieves the files
3	The Jenkins server builds the application. If the build passes, we progress to step **4**. If it fails, we move to step **7**.
4	The Jenkins server tests the application. If the test passes, we progress to step 5. If it fails, we move to step **7**.
5	The Jenkins server runs the Terraform init, plan and apply steps. We move to step **6**.
6	The VPC named dep4-VPC is created. This VPC consists of: 1. 2 availability zones a. us-east-1a b. us-east-1b 2. Four subnets a. PUBLIC-A:172.19.0.0/18 b. PRIVATE-A:172.19.64.0/18 c. PUBLIC-B:172.19.128.0/18 d. PRIVATE-B:172.19.192.0/18 3. Internet Gateway 4. Private route table 5. Public route table 6. 2 Security groups a. For public web server : ssh-access allowing tcp 22 and 80 b. For private application server: ssh-access2 allowing tcp 22 and 8000
7	An elastic ip address is created
8	The NAT gateway is created and the elastic ip is associated to it

STEP	DESCRIPTION

9	The EC2 designated to be the application server is created in subnet PRIVATE-A after the NAT gateway is fully provisioned. This is necessary for the EC2 to be able to access the internet and install all the dependencies when it is created.
	
	This is enforced by including the following line in the terraform file *(see line 16 in instances.tf)*:
	
	```depends_on = [aws_nat_gateway.nat_gateway_prob]```
	
	The deploy.sh referenced in the user_data field *(see line 10 in instances.tf)* installs the required dependencies, clones the repository and launches the web application and associates with the default tcp port 8000
	
	```user_data = "${file("deploy.sh")}"```

10	The EC2 designated to be the proxy server is created in subnet PUBLIC-A after the application server has it's private ip assigned. This is necessary for the proxy EC2 to be configured to point to the application server.
	
	This is enforced by including the following line in the terraform file *(see line 52 in instances.tf)*:
	
	```depends_on = [aws_instance.web_server02]```
	
	Bash commands are passed via user_data to:
	1. install the required dependencies *(see line 29 to 39 in instances.tf)*
	2. install nginx *(see line 38 in instances.tf)*
	3. configure the "*/etc/nginx/sites-enabled/default*" file so that the nginx server acts as a reverse proxy that points to the application server's private ip address *(see line 40 to 42 in instances.tf)*
	
	```sed -i "s/# First attempt to serve request as file, then/proxy_pass http:\/\/${aws_instance.web_server02.private_ip}:8000;/" /etc/nginx/sites-enabled/default```
	
	This variable passes the private ip address of the application server into the script
	
	4. restart the nginx server

11	If all stages are passed, then we proceed to step 12. If there are errors the error logs and reports are reviewed to identify and isolate the issue. Any code related issued including errors in the Jenkins file are corrected and the updates committed and pushed to GitHub i.e. we proceed to step 1.
12	The deployed application is monitored

kerri.p.h.smith@gmail.com

Screen Capture

Old Pipeline

Status			
Changes			
Build Now			
View Configuration			
Full Stage View			
GitHub			
Pipeline Syntax			
Build History trend ⌃			
Atom feed for all Atom feed for failures			

Branch main

Full project name: Deployment 4/main

Stage View

	Declarative: Checkout SCM	Build	test	Init	Plan	Apply
Average stage times:	3s	12s	1s	9s	12s	43s
#1 Oct 22 17:12 No Changes	3s	12s	1s	9s	12s	43s

Permalinks

- Last build (#1), 20 sec ago

0.5 ─────────

trend ⌃
for failures

Stage View

	Declarative: Checkout SCM	Build	test	Init	Plan	Apply	Destroy
Average stage times: (Average full run time: ~1min 43s)	2s	8s	1s	9s	15s	28s	1min 6s
#3 Oct 22 17:24 2 commits	1s	5s	1s	9s	17s	12s	1min 6s
#2 Oct 22 17:20 No Changes							
#1 Oct 22 17:12 No Changes	3s	12s	1s	9s	12s	43s	

URL Shortener Dep3_2022   API   New URL

Website	File
Short Name	Short Name
_____	_____
Website URL	Website URL
_____	Choose File No file chosen
Shorten	Shorten

New Pipeline

Stage View

	Declarative: Checkout SCM	Build	test	Init	Plan	Apply	Destroy
Average stage times: (Average full run time: ~2min 19s)	2s	7s	1s	9s	12s	1min 12s	53s
#5 Nov 08 21:24 — 4 commits	2s	7s	1s	9s	10s	3min 15s	
#4 Oct 27 22:01 — No Changes	1s	5s	1s	10s	10s	39s	41s
#3 Oct 22 17:24 — 2 commits	1s	5s	1s	9s	17s	12s	1min 6s
#2 Oct 22 17:20 — No Changes							
#1 Oct 22 17:12 — No Changes	3s	12s	1s	9s	12s	43s	

Latest Test Result (no failures)

Instances (1/2) Info

Connect | Instance state ▼ | Actions ▼ | Launch instances ▼

Find instance by attribute or tag (case-sensitive)

VPC ID = vpc-0413b8e28cef125d6 ✕ | Clear filters

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
☐	dep4Gunicorn	i-0c71908a48f1925f9	⊘ Running	t2.micro	⊘ 2/2 checks passed	No alarms +	us-east-1a	–	–
☑	dep4Nginx	i-0dfdb8c29042c65f8	⊘ Running	t2.micro	⊘ 2/2 checks passed	No alarms +	us-east-1a	ec2-44-200-252-34.co...	44.200.252.34

Not secure | 44.200.252.34

Apps | Home | AWS Management... | BitRix24 | KURA | Python PAGES | BASH | API | POLITICS_BIG_DATA | FINAL PROJECT | Algorithms | TERRAFORM | GitHub | Docker

URL Shortener Dep3_2022   API | New URL

Website	File
Short Name	Short Name
Website URL	Website URL
	Choose File No file chosen
Shorten	Shorten

kerri.p.h.smith@gmail.com

board > Deployment 4 > main >

Pipeline Syntax

Build History   trend ⌃

Atom feed for all   Atom feed for failures

Stage View

	Declarative: Checkout SCM	Build	test	Init	Plan	Apply	Destroy
Average stage times: (Average full run time: ~2min 22s)	1s	7s	1s	9s	12s	59s	1min 14s
#6 Nov 08 21:41 — 1 commit	1s	5s	1s	8s	12s	7s	1min 57s
#5 Nov 08 21:24 — 4 commits	2s	7s	1s	9s	10s	3min 15s	
#4 Oct 27 22:01 — No Changes	1s	5s	1s	10s	10s	39s	41s
#3 Oct 22 17:24 — 2 commits	1s	5s	1s	9s	17s	12s	1min 6s
#2 Oct 22 17:20 — No Changes							
#1 Oct 22 17:12 — No Changes	3s	12s	1s	9s	12s	43s	