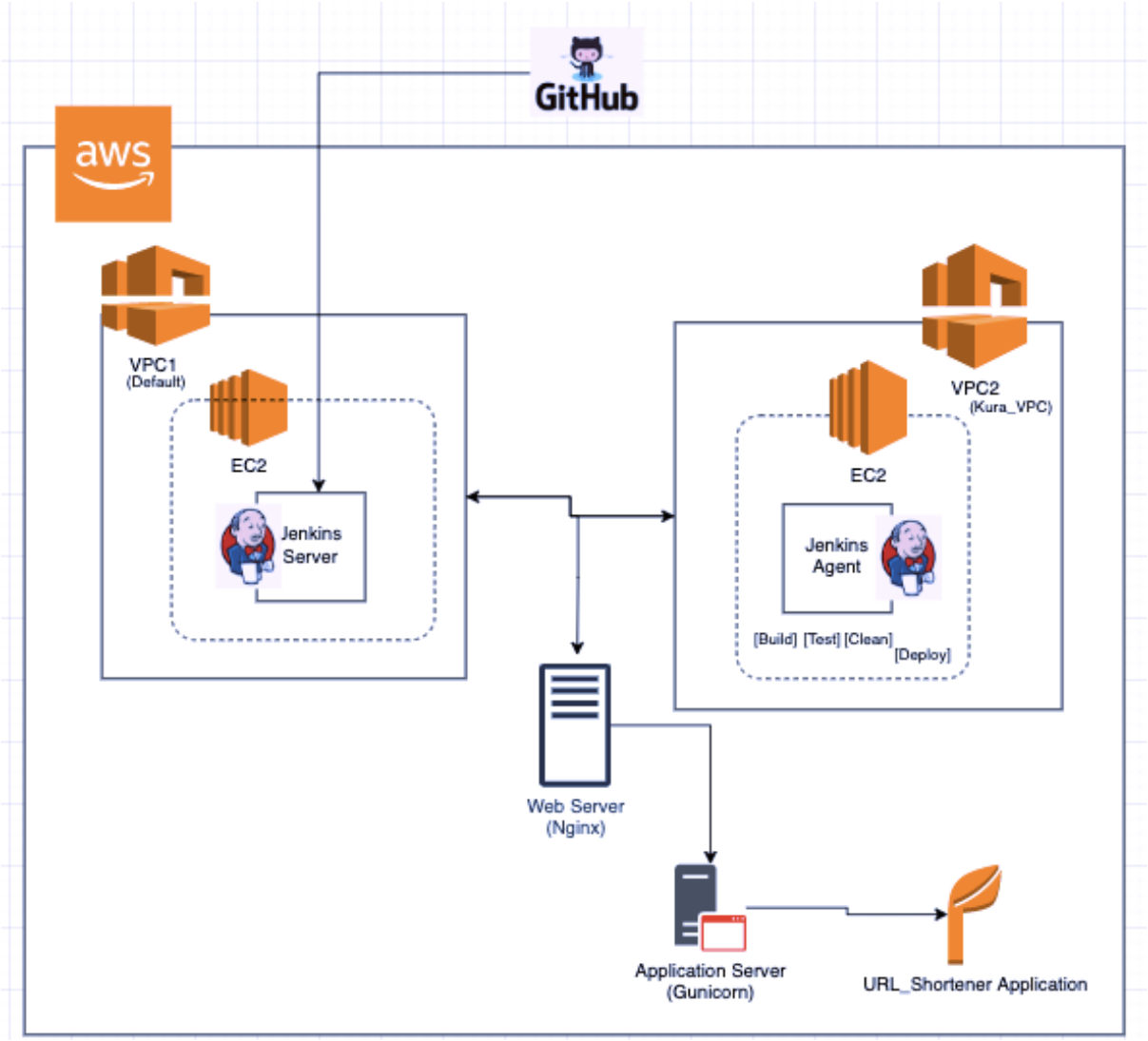
DEPLOYMENT #3



Purpose

This deployment creates a pipeline by connecting a Jenkins agent in a customized VPC to a Jenkins server in another (default) VPC to deploy a web application all within the AWS platform using EC2s.

Tools/services and software stacks used:

- GitHub repository
- AWS VPCs
- AWS EC2s
- Jenkins (server and agent)
- Web server (nginx)
- Application server (gunicorn)

GitHub

- The code to build the url-shortner web application was stored in a GitHub repository.

- The source repository was first forked to enable a copy of the original repository to reside in my GitHub repository.
- Jenkins connected to my GitHub repository to initiate the build process

AWS VPCs

- Two VPCs were created just for the purpose of demonstrating the process of housing the Jenkins server and the Jenkins agent in different public subnets. Using different availability zones in one VPC would be the more realistic topology.
- VPC1 (default) housed the EC2 that ran the Jenkins server and VPC2(Kura-VPC) housed the EC2 that ran the Jenkins agent.

AWS EC2s

- Two EC2 instances were launched, one in each VPC.
- One EC2 ran the Jenkins server with port 80 (HTTP), port 22 (SSH) and port 8080 (Jenkins) open for inbound traffic.
- The other EC2 ran the Jenkins agent with port 80 (HTTP), port 22 (SSH) and port 5000 (TCP/IP) open for inbound traffic.
- Python3-pip, python3-10-venv, default-jre and nginx packages were installed in the virtual environments to update them with the necessary libraries and dependencies.

Jenkins

- The Jenkins server was set up on VPC1. A multibranch pipeline build was selected with GitHub being the designated branch source. The server would automatically discover, manage, and execute pipelines for branches which contain a Jenkinsfile in source control.
- A Jenkins agent was configured and connected to Jenkins server to only build jobs with the label/name **awsDeploy**
- Jenkins agent ran the build, test, clean and deploy tests of the url-shortner app successfully.

Web server

- Webserver nginx, accepts request, takes care of domain logic, and handles http connections requests rerouted to listen on port 5000 instead of port 80.

Application server

- Application server gunicorn, a web server gateway interface (WSGI) ensures that the web server and the python web application **URL shortener** can talk to each other.

Issues

- Jenkins agent not active outside of build - Initially, despite a success build, test, clean and deploy in Jenkins agent, the URL shortener application failed to launch using the "IP address:5000".
    - Patch: Add port 5000 to the last line of the code below.

```
stage ('Deploy') {
agent{label 'awsDeploy'}
    steps {
        keepRunning {
            sh '''#!/bin/bash
            pip install -r requirements.txt
            pip install gunicorn
            python3 -m gunicorn -w 4 application:app -b 0.0.0.0:5000 --daemon
            '''
```

- Despite a success build, there was a flag that the commit status could not be updated even though the GitHub credential were verified and repo:status scope was correctly configured.
    - Patch: Still being researched.

```
Requirement already satisfied: setuptools>=3.0 in ./test3/lib/python3.10/site-packages (from gunicorn)
(59.6.0)
Installing collected packages: gunicorn
Successfully installed gunicorn-20.1.0
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline

Could not update commit status, please check if your scan credentials belong to a member of the organization
or a collaborator of the repository and repo:status scope is selected


GitHub has been notified of this commit's build result

Finished: SUCCESS
```