**Documentation**

Please go through the whole documentation to understand the necessary information. And check **NOTE** section below before applying Terraform code.

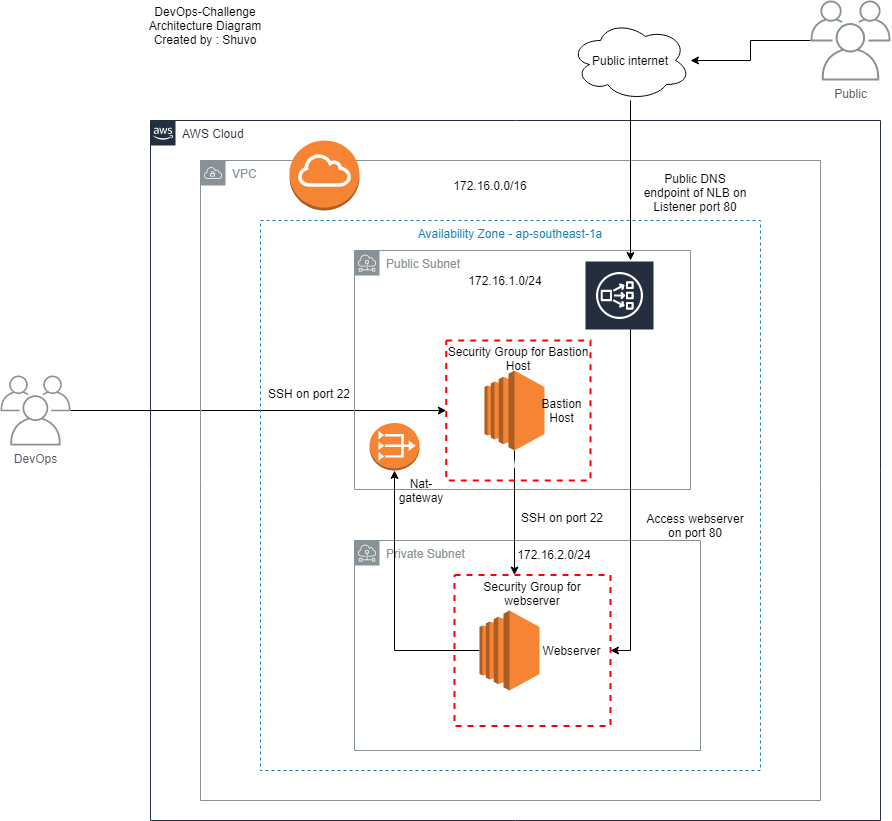
# Running in Dev Environment

1. To build the application image and run the application locally:
2. Take a V.machine with ubuntu 20.04 os.
3. Git clone <https://github.com/shakhawat-devops/project-oval-hr.git>
4. Go inside the directory project-oval-hr and there you’ll find a file named “userdata.sh”
5. Move the file out of the directory (/home/ubuntu/ - this location for user- ubuntu)
6. Give permission to the file for execution purpose -> chmod +x ./userdata.sh
7. Then run this command -> ./userdata.sh
8. This will containerize the whole application in docker, and you can check the containers running “$ docker ps” command.
9. Now if you access your machine public Ip, you will see the Laravel application is running.
10. Or, if you run “curl <http://localhost>:80” to see the application running locally.

# Running in PROD Environment

1. Provisioning the infrastructure in Terraform:
2. All the resources will be created in North Virginia region. So our default region is set to “us-east-1” and put it in variables.tf file.
3. First A custom vpc, a public subnet, private subnet and rout tables are created accordingly.
4. Then both the internet gateway and nat gateway are created in public subnet. In the route table of public subnet, a 0.0.0.0 route has been added to internet gateway and in the route table of private subnet a 0.0.0.0 route has been added to nat-gateway for accessing internet.
5. A private Ec2 instance (webserver) is created in the private subnet in the created custom vpc.
6. We are now using only one availability zone but we can use multi-AZ also.
7. Then a security group is attached to the webserver and allow traffic for 80 port for everywhere and 22 port for ssh from Bastion Host.
8. A Bastion Host will be created also in the public subnet and same availability zone to ssh into private webserver. A security group is attached with it allowing port 22 for ssh purpose.
9. Using agent forwarding mechanism we will ssh into our private webserver using bastion host.
10. Then A network load balancer is created to expose the application publicly. A target group will be created and attach the private webserver in the target group in port 80.
11. A listener will be added for port 80 in load balancer and mapped the listener to the created target group.
12. The public endpoint of NLB dns will expose the application in port 80.

**Here is the Architecture diagram:**



1. In the user data section of ec2 instance all the necessary scripts of userdata.sh file has been provided by Terraform. During the launch of webserver, it fetches the code from github and install the dependencies in machine. Then it packages the Laravel application into a Docker image initiating docker build. Using Docker compose multi-container docker application have been defined and run. Docker-compose.yml file configure the application services.

Here we will get three Docker container where our services i.e app(on port 9000), db(on port 3306) and webserver(on port 80) are running.

# Running the terraform codes

1. For running the terraform code clone the below repository in your local machine

git clone <https://github.com/shakhawat-devops/project-oval-hr-terraform-codes.git>

1. Then initialize the repository by running: terraform init
2. Then see how many resources are going to be created by terraform using: terraform plan
3. Then create the resources in AWS: terraform apply
4. Put yes when its prompted.
5. You will see the the PHP-Laravel Application deployed to AWS and you can access via copying the endpoint of the NLB and hitting to the browser

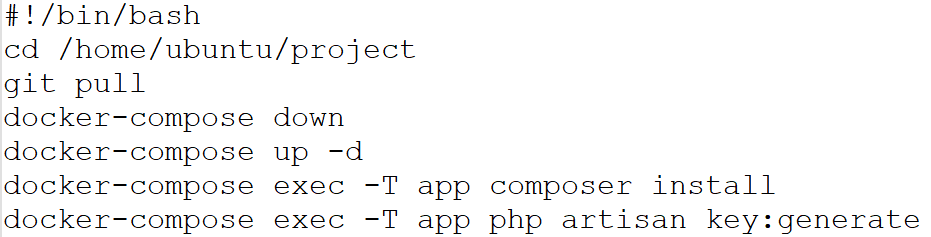
# Creating Continuous Delivery

1. For Deploying to the production environment I have used Jenkins as the CI server and from there the updated code will be deployed to the webserver. I have installed Jenkins in the Bastion host and added the .pem file in the credentials of the Jenkins and added the SSH remote server in SSH sites in Jenkins. Then I have created a pipeline which pulls the updated code from the Github repository and deploys it to the Webserver. My pipeline flow is this  
   Dev pushes the updated code to Github > Jenkins creates a build > which runs a script in the Production server.

Please add the 8080 port in the security group of the Bastion host to access the Jenkins UI

Also note that to run scripts in remote server over SSH, the SSH plugin needs to be installed in Jenkins.

1. I have created a script in the production server the production deployment. The contents are as follows. SSH into the production server from the bastion host and create a scripts.sh file and add the below lines there.

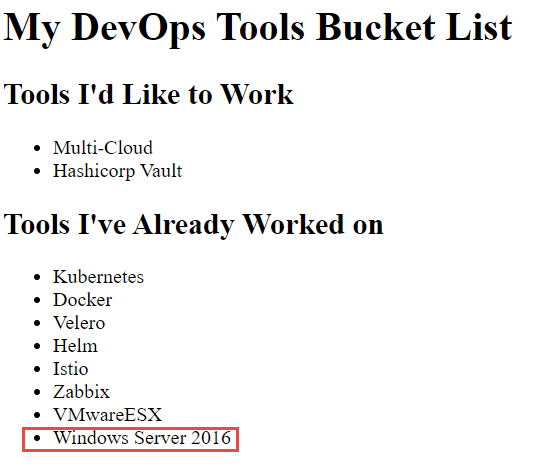


1. When Developers push the updated code to the Github the pipeline will be called and it will take a git pull and deploy the updated code to the production environment.

First Deploy:



Second Deploy(CICD):



**Note:**

**1. Please create a key pair name “app-prod” manually in your aws console before applying Terraform script or, change the key pair name in terraform script in ec2 instance create section. Change both for webserver and Bastion Host. Otherwise script will give 400 error. I have mentioned this limitation of mine below.**

**2.** **It will take some time to create docker container and run the application though the NLB becomes active. So please reload and check if you don’t get the Laravel home page.**   
  
To see the page please go to the link:

http://demo-nlb-tf-bc43c961992289e2.elb.us-east-1.amazonaws.com/

**My Limitations:**

1. I know how to generate key pair and use it in aws to create ec2 instance. But as there are no options to download the key from console after that, so I don’t know how to download it through terraform locally. So, I created a keypair manually in console and then used it to create ec2 instance and access into it.
2. I couldn’t submit the YAMLs to deploy into Kubernetes due lack of time. I will submit it gradually to the repository
3. The default security group of VPC has been attached with the created ec2 instances, I couldn’t detach it during launching servers. So as a result, for the Bastion Host all the traffic ports are open. This is a vulnerable case. But I can fix this issue manually.

Thank you.