Costa Rica 13 / Feb / 2019

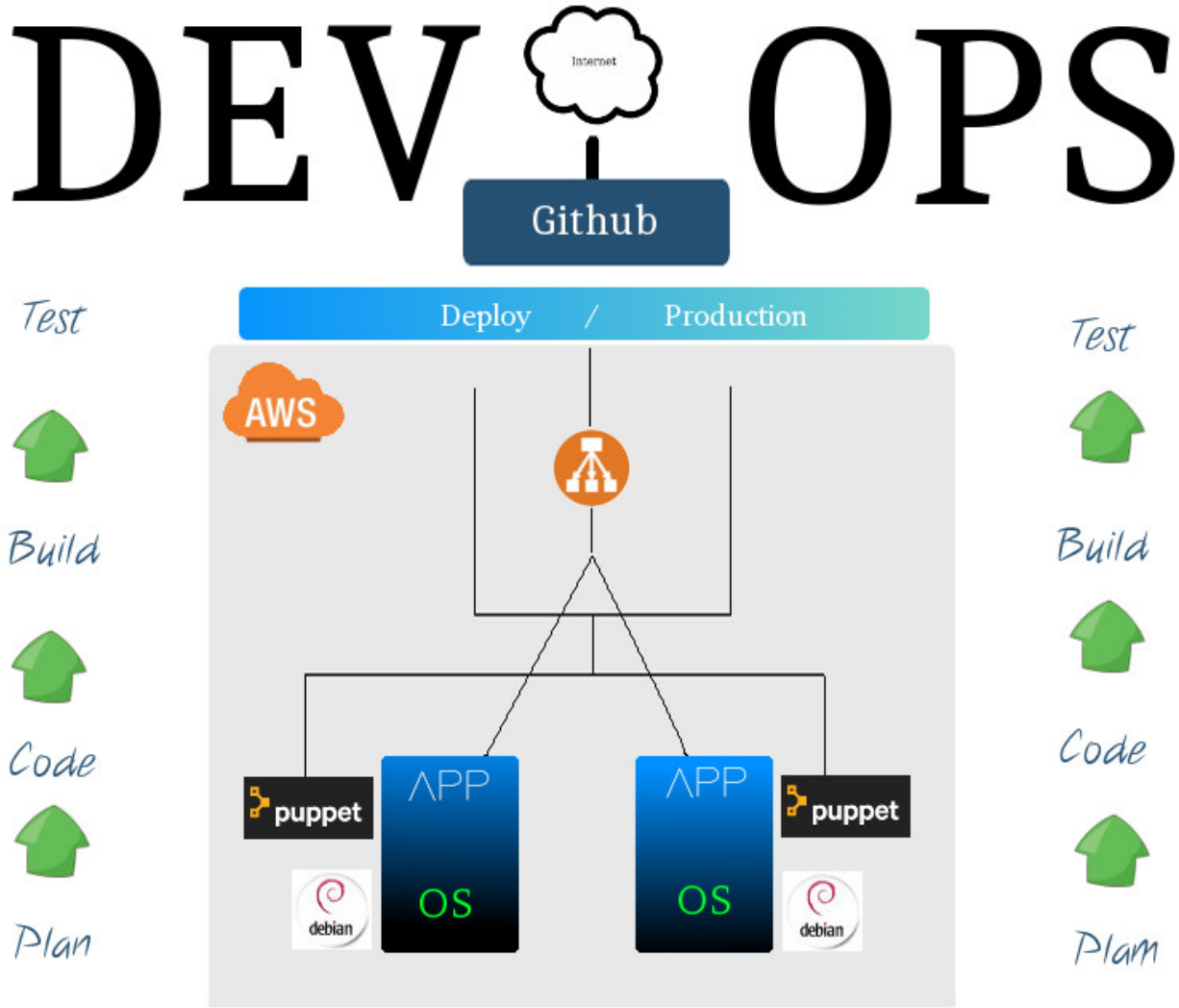
Work Report

Rafael E Rumbobs S

Acceptance Criteria:

- Architecture diagram demonstrating planned solution.
- Timeoff-management fork on local repository (Github, CodeCommit, BitBucket, GitLab, etc)
- Required infrastructure running on cloud provider of preference, provisioned using some sort of infrastructure as code solution.
- Application must be deployed using a fully automated continuous integration solution, triggered by a change in source control.
- Application must be secured from external access and the application should be serving via standard HTTP and HTTPs protocols.
- The application should be highly available and load balanced.

Solution Diagram



Components

ec2 instances
Debian GNU/Linux
Aws Classic Balancer
Puppet
Apache2
Cron Jobs
Git - github
NodeJs
Bash

- Timeoff-management fork on local repository (Github, CodeCommit, BitBucket, GitLab, etc)

Solution: please review a copy the app at the following link: https://github.com/rufus88/deploy

- Required infrastructure running on cloud provider of preference, provisioned using some sort of infrastructure as code solution.

Solution: cloud provider AWS

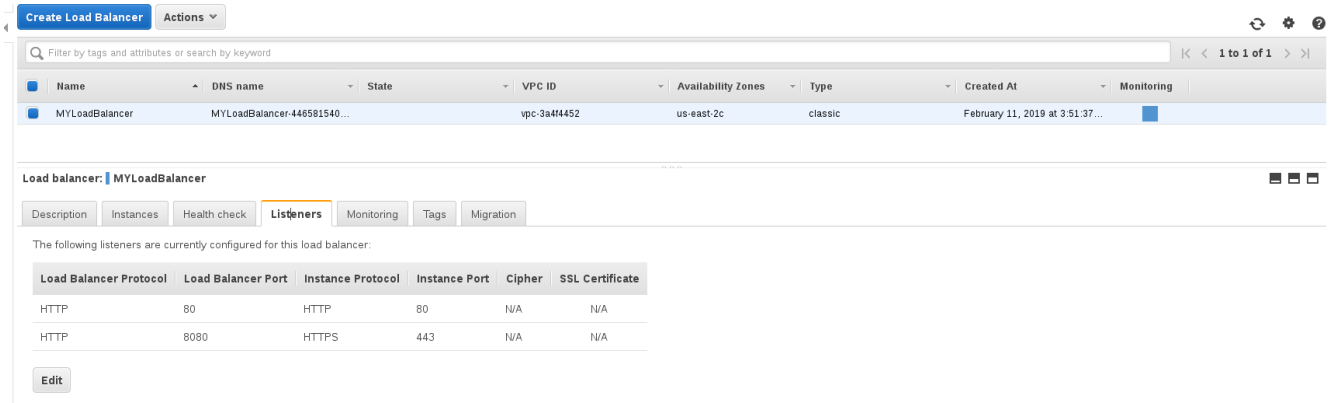
Infrastructure as code: Puppet - please review https://github.com/rufus88/production/blob/master/manifest/install.pp

- Application must be deployed using a fully automated continuous integration solution, triggered by a change in source control.

Solution: Using cron jobs and bash logic
challenge: improve code to catch all possible case scenarios
https://github.com/rufus88/production/blob/master/manifest/env.pp

- Application must be secured from external access and the application should be serving via standard HTTP and HTTPs protocols.

Solution: applications is now currently running on port 80 and 8080 as an attempt to configure SSL.



```
<VirtualHost *:443>
SSLEngine On
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
SSLCertificateFile /etc/ssl/localcerts/apache.pem
SSLCertificateKeyFile /etc/ssl/localcerts/apache.key
ProxyPreserveHost On
ProxyRequests Off
ProxyPass / http://localhost:3000/
ProxyPassReverse / http://localhost:3000/
</VirtualHost>
```

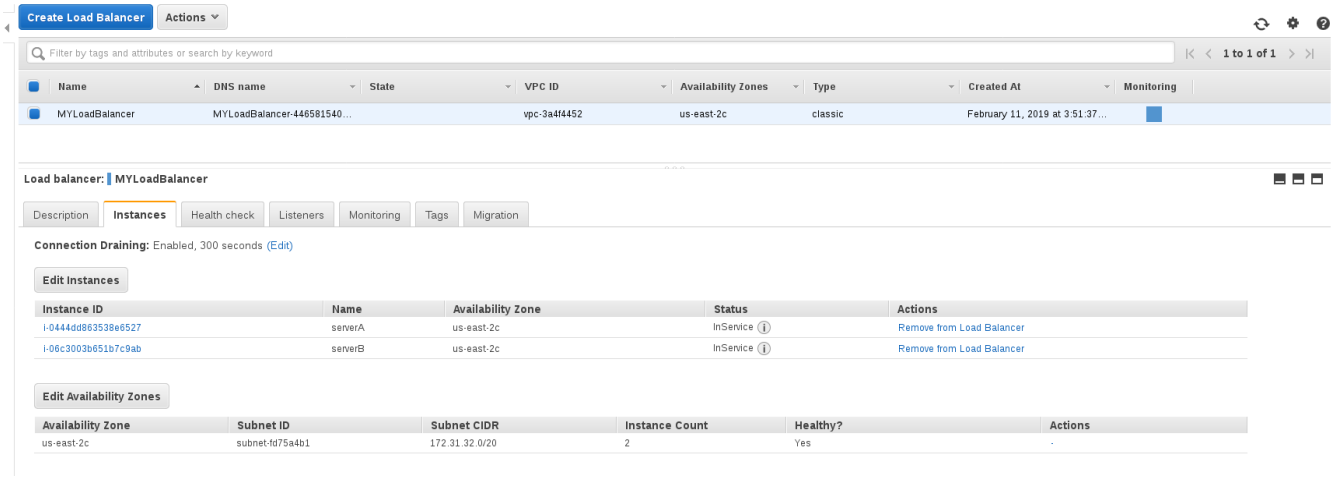
```
Nmap scan report for ec2-3-16-1-186.us-east-2.compute.amazonaws.com
(3.16.1.186)
Host is up (0.11s latency).
Not shown: 996 filtered ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
443/tcp  open  https
3000/tcp open  ppp
```

```
Nmap scan report for ec2-18-222-240-252.us-east-
2.compute.amazonaws.com (18.222.240.252)
Host is up (0.10s latency).
Not shown: 996 filtered ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
443/tcp  open  https
3000/tcp open  ppp
```

Challenge: without an FQDN for the project is hard to setup a valid SSL service as note want to state that try to use Lets Encrypt service but the DNS name of the AWS classic load balancer was not accepted.

- The application should be highly available and load balanced.

Solution: Using AWS classic load Balancer on two EC2 instances please see the following screenshot.



Challenge: application is using SQLite3 as data backend - the backup of this element should be also automated in case of emergency.