HARVARD
Extension School

Week 13
CI/CD and Final Project

Agenda

- **Week 13:** Group project (1/3)
  - Final Project Discussion.
  - Git Hooks
  - Testing Criteria for website success or fail
  - Assignments 5 and 6.

Final Project

- You are developing an open source project on public github.
- You want to package your project requirements in a single container.
- You don't want to spend money on a development and staging environment.
- You don’t want to put your cloudFormation templates in the public repository.
- You seperate your sensitive data and template in separate repository from your open source repository.

Final Project cont.

CI/CD techniques enable DevOps team to deploy software rapidly, repeatedly, and reliably.

In this project, we will build pipeline to serve a website that needs a seamless, end-to-end continuous delivery and deployment workflow: from source code, to build, to deployment, to software delivery. In our project we will build:

- A Continuous Integration (CI) pipeline to support test, configuration and deployment of a simple website.
- We will also practice Continuous delivery where code changes are automatically built, tested, and prepared for a release to production. We will deploy all code changes to a testing environment, if succeeds then deploy to staging and finally to production environment. This process makes the entire software release process automated, where we will build an artifact that has undergone a standardized test process, and deploying our website revisions to production environment automatically, without explicit approval from a developer.

Environments we have are:

1. Testing: hosted in AWS
2. Staging: hosted in AWS
3. Production: hosted in Google Cloud

Final Project cont.

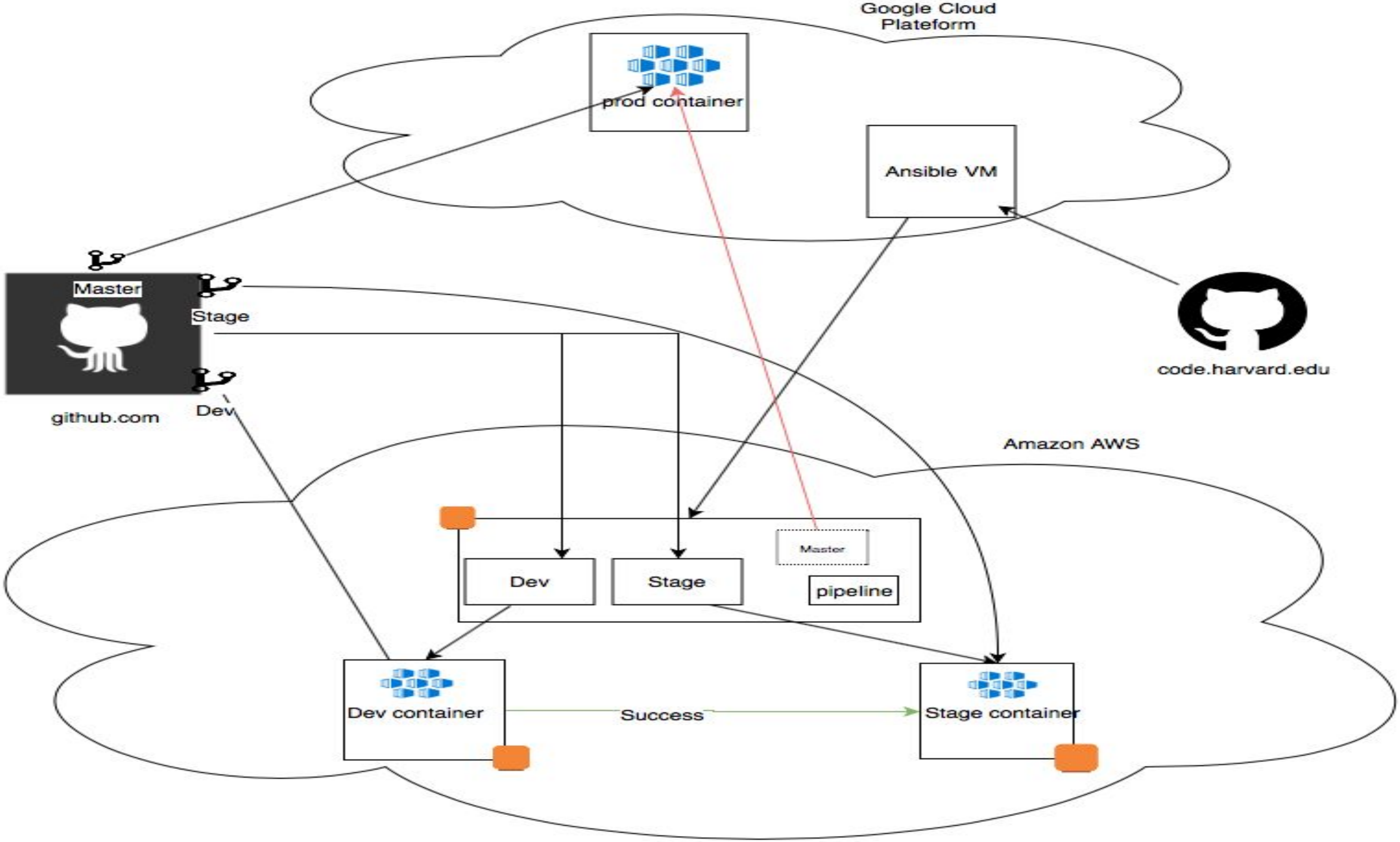
Two Git repositories, public github and enterprise private code.harvard.edu:

1. code.harvard.edu (enterprise github) repo with one branch to accommodate:
    a. Branches
        i. Master
    b. Files:
        i. Ansible configuration files
        ii. CloudFormation files
2. Public Github to perform version control for web site files and images:
    a. Branches
        i. Dev
        ii. Staging
        iii. Master
    b. Files
        i. Website (index.html)
        ii. Dockerfile (if needed)

Final Project cont.

Our pipeline will consist of:

1. Ansible: to build and configure Jenkins server
2. Jenkins to get triggered by public github using webhooks and perform testing, staging and deployment
3. AWS CloudFormation: to setup and configure the testing and staging web servers in docker containers in AWS.
4. Google CLOUD Instance Template to setup and configure the PRODUCTION web server in a docker container in Google Cloud.


Google Cloud
Plateform
prod container
Ansible VM
Master
Stage
github.com
Dev
code.harvard.edu
Amazon AWS
Master
Dev
Stage
pipeline
Dev container
Success
Stage container

Git webhooks

- To integrate GitHub (and/or GitHub Enterprise) with Jenkins.
- To trigger Jenkins jobs from GitHub (and/or GitHub Enterprise) from a “GitHub API User” (GitHub User to interact with Jenkins).
- Git hooks are event-based: When you run certain git commands, the software will check the hooks directory within the git repository to see if there is an associated script to run.

Steps to install jenkins github plugin

1. Install Jenkins github plugin. Step 1: Click on the Manage Jenkins button on your Jenkins dashboard:
   a. Click on Manage Plugins
   b. In the Plugins Page, select the GIT Plugin, Click on Install without restart.
   c. Select the option Download now and Install after restart button. In which plugin is installed after restart
   d. You will be shown a "No updates available" message if you already have the Git plugin installed.
   e. Once the plugins have been installed, go to Manage Jenkins on your Jenkins dashboard. You will see your plugins listed among the rest.
   f. Go to Manage Jenkins > Configure System
   g. In the Github section choose Add Github Server
   h. Add credentials (your Github credentials) and click Save

Steps to configure webhooks

1. Go to github repository that you are integrating with Jenkins. You need to have admin rights for that repo.
2. Click Settings
3. Select webhooks
4. Add webhook
5. Payload url is jenkins_url_with_http_or_https/github-webhook/
6. Content type: application/json
7. Just push event if you only want to trigger jenkins jobs by branch pushes.
8. Check Active
9. Click update
10. Recent deliveries should have green check icon. If that is not the case, click on latest delivery that will contain HTTP request/response.

Jenkins Stage Tests

A python program that

1) The site is up
2) HTML parser succeed
3) The content is what you expected

Python HTML Parser

https://docs.python.org/3/library/html.parser.html

Assignments 5 & 6 Questions and guidelines

1. It is an assignment/document
2. Read the whole thing once and twice
3. Visualize and write down what you understood
4. Try to visualize the solution
5. Look at the solution and commands provided in the assignment
6. Does it align with your understanding of the assignment?