Ishtaar Desravines

09/01/2022

Deployment #1 Documentation

Jenkins on an EC2

1. In order to set up a Jenkins manager, I first created an AWS EC2 with ports 22, 80, and 8080 open.
   - Port numbers:
     - Port 22: open to ssh into the remote server (EC2) from my local ubuntu virtual machine.
     - Port 80: to access the web server of Jenkins.
     - Port 8080: to access the Jenkins webpage to create/log into my account from my browser.
   - Installing Jenkins:
     - First installed Java on my EC2, since Jenkins is a Java application and needs it to run.
     - Then added Jenkin's Linux Repository Signing Keys so that the Jenkins' repository can be updated on my remote server.
     - Then I added the Jenkins repository to my server.
     - Next, I updated the repository and I installed Jenkins on my EC2
     - Finally, I started Jenkins on my EC2 and checked its status.

Virtual Environment on EC2

1. In order for the application to run on my EC2, I installed python, since the application included python files.
2. I also installed a virtual environment on the EC2 so that the application can run without version conflicts from their dependencies.

Configure Jenkins:

1. I accessed the Jenkins webpage through http://EC2publicipaddress:8080 and entered in the key placed into a file on my server when installing Jenkins on my EC2.
2. I created an account and then configured a cloud on Jenkins by adding my AWS credentials. This allows me to use my EC2 as a Jenkins agent.
3. I then connected Github to the Jenkins server by creating a multi branch build and entering my Github username and personal access token generated in the Github developer settings.

After Github was connected to the Jenkins server, a build started automatically from the files that were on the Github repository. The build was successful and the tests were passed.

Deploying the application:

1. Since the flask application build was complete and passed the tests on Jenkins, I cloned the Github repo files to my EC2, compressed the files, and then deployed it to AWS Elastic Beanstalk.
2. The application had a green check mark, indicating ok health.

What I would improve:

1. Include two Jenkins servers. One manager and one agent. The agent can handle the build/ test and the manager and delegate to the agent.
2. If the builds and test pass in Jenkins, then the application files should automatically be deployed to AWS Elastic Beanstalk. Can be done through a service like AWS Codepipeline.