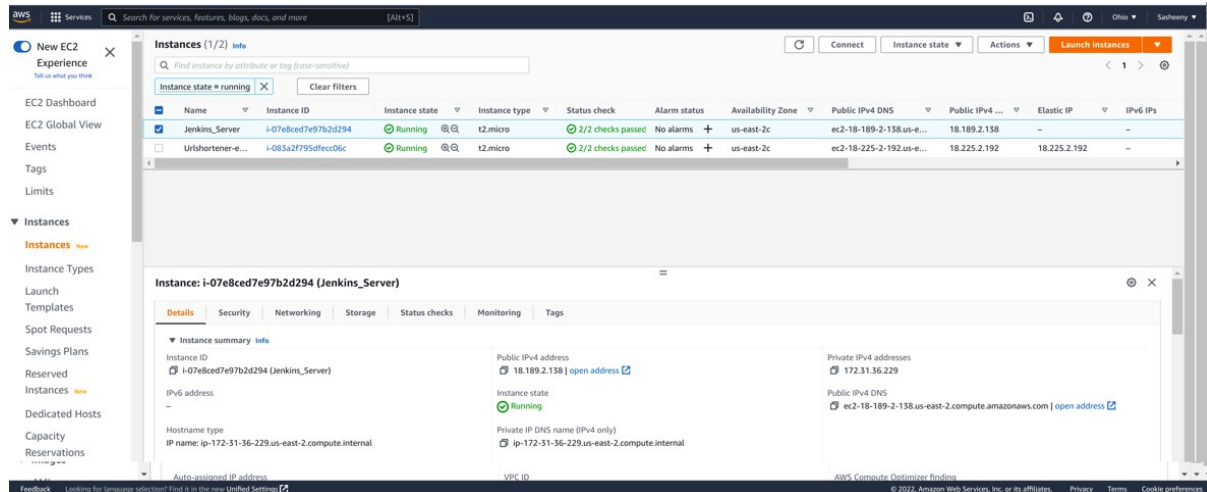


Task: Set up a Jenkins CI/CD pipeline; Deploy a Python Flask app to AWS Elastic Beanstalk

Create an Ubuntu EC2 instance in AWS that will serve as your Jenkins server.



Once the EC2 has been established, from your local server (computer) SSH into your Jenkins server, the EC2 instance you just created above.

```
ubuntu@ip-172-31-36-229: ~/kuralabs_dep... x Terminal x ubuntu@ip-172-31-36-229: ~
sasheenhubbard:~/kura/Deployments/key_pairs$ ssh -i jenkins.pem ubuntu@18.189.2.138
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-1019-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Tue Oct  4 01:08:02 UTC 2022

System load:  0.0               Processes:           103
Usage of /:   59.5% of 7.57GB   Users logged in:    0
Memory usage: 71%              IPv4 address for eth0: 172.31.36.229
Swap usage:  0%

 * Ubuntu Pro delivers the most comprehensive open source security and
   compliance features.

https://ubuntu.com/aws/pro

59 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

*** System restart required ***
Last login: Sun Oct  2 23:57:24 2022 from 47.184.114.235
ubuntu@ip-172-31-36-229:~$ ls
```

In your Jenkins server install Python3 and Jenkins. To confirm Python3 and Jenkins are installed and running enter the following command in the terminal:

python3 --version

jenkins --version

systemctl status Jenkins.service

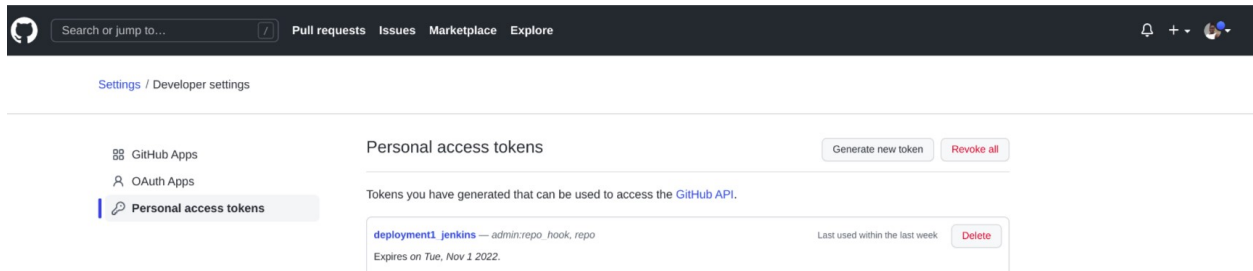
```
ubuntu@ip-172-31-36-229: ~/kuralabs_dep... x Terminal x ubuntu@ip-172-31-36-229: ~ x
ubuntu@ip-172-31-36-229:~$ python3 --version
Python 3.10.4
ubuntu@ip-172-31-36-229:~$ jenkins --version
2.346.3
ubuntu@ip-172-31-36-229:~$ systemctl status jenkins.service
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2022-10-02 23:54:16 UTC; 1 day 1h ago
     Main PID: 462 (java)
       Tasks: 40 (limit: 1143)
      Memory: 489.2M
         CPU: 3min 59.011s
        CGroup: /system.slice/jenkins.service
                └─462 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=>

Oct 03 22:49:13 ip-172-31-36-229 jenkins[462]: 2022-10-03 22:49:13.449+0000 [id=388]      INFO >
Oct 03 22:49:13 ip-172-31-36-229 jenkins[462]: 2022-10-03 22:49:13.455+0000 [id=388]      INFO >
Oct 03 23:49:13 ip-172-31-36-229 jenkins[462]: 2022-10-03 23:49:13.450+0000 [id=396]      INFO >
Oct 03 23:49:13 ip-172-31-36-229 jenkins[462]: 2022-10-03 23:49:13.455+0000 [id=396]      INFO >
Oct 03 23:54:15 ip-172-31-36-229 jenkins[462]: 2022-10-03 23:54:15.432+0000 [id=404]      INFO >
Oct 03 23:54:15 ip-172-31-36-229 jenkins[462]: 2022-10-03 23:54:15.436+0000 [id=404]      INFO >
Oct 04 00:18:20 ip-172-31-36-229 jenkins[462]: 2022-10-04 00:18:20.474+0000 [id=405]      INFO >
Oct 04 00:18:20 ip-172-31-36-229 jenkins[462]: 2022-10-04 00:18:20.477+0000 [id=405]      INFO >
Oct 04 00:49:13 ip-172-31-36-229 jenkins[462]: 2022-10-04 00:49:13.449+0000 [id=406]      INFO >
Oct 04 00:49:13 ip-172-31-36-229 jenkins[462]: 2022-10-04 00:49:13.451+0000 [id=406]      INFO >
lines 1-20/20 (END)
```

In your GitHub account go to [Settings](#) → [Developer Settings](#) → [Personal Access Token](#) and generate a new token. Be sure to copy and paste your key to a safe place you can access it.

DO NOT store your access token in GitHub!

You will use this access token to link your source code in GitHub to your Jenkins pipeline.

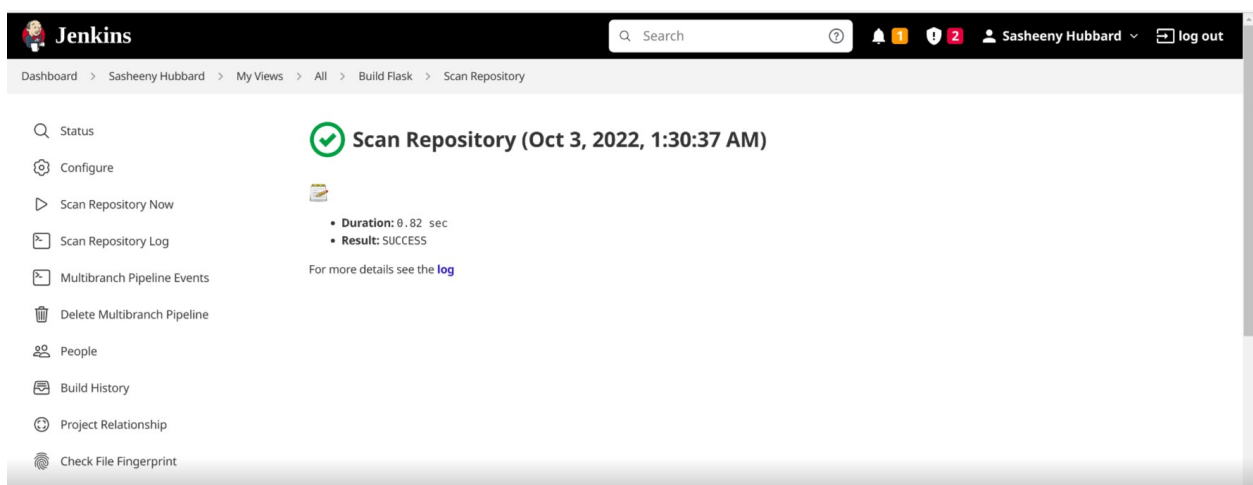


In order Jenkins to have access to the code when setting up your access code you have to select *repo* and *admin:repo_hook*(webhook). These options give Jenkins the ability to control of repo hooks and of your private repositories also know as your source code files, in this instance.

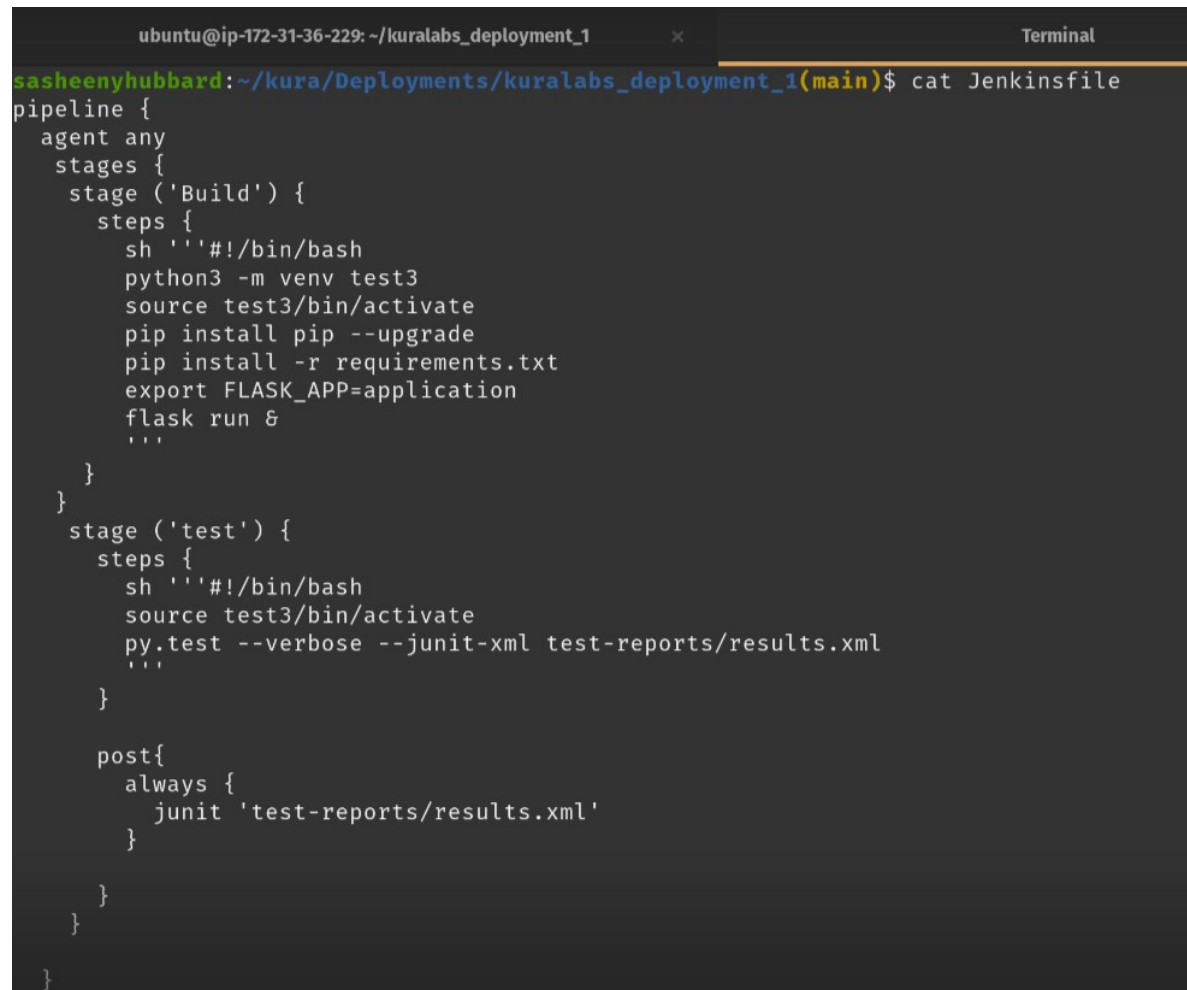
Create a multibranch pipeline in Jenkins and use the GitHub Branch Source plugin to create a new project based on your source code repository. When adding your GitHub credentials select Jenkins and enter your GitHub username and your GitHub access token as the password. Next, enter your repo URL address and validate to ensure the link is readable. **(Make sure you don't have any extra characters in the address because you will get an error and will not be able to proceed to the next stage.)**

At this point a build should've been triggered.

If you've completed all the steps correctly, you should see a **SUCCESS** status otherwise you'll see **FAILURE**.



The image below shows the contents of the Jenkinsfile, the steps of the pipeline. It details the stages of the pipeline; the Build, the Test and the Post.

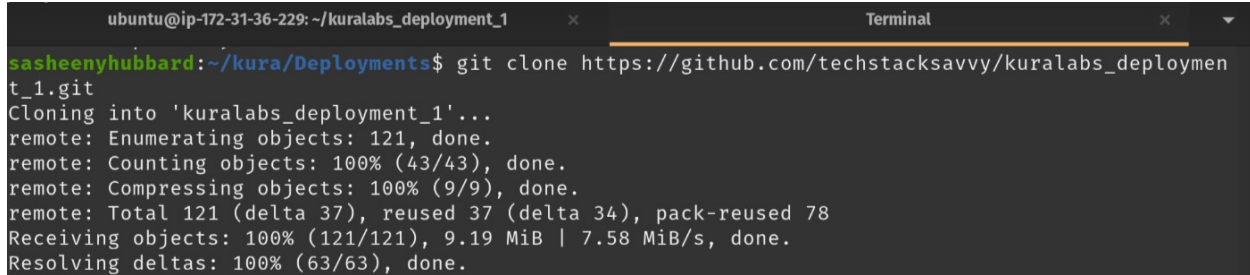
A terminal window titled 'Terminal' with a tab 'ubuntu@ip-172-31-36-229: ~/kuralabs_deployment_1'. The prompt is 'sasheenhubbard:~/kura/Deployments/kuralabs_deployment_1(main)\$'. The command 'cat Jenkinsfile' is executed, displaying the following Jenkinsfile content:

```
pipeline {
  agent any
  stages {
    stage ('Build') {
      steps {
        sh '''#!/bin/bash
        python3 -m venv test3
        source test3/bin/activate
        pip install pip --upgrade
        pip install -r requirements.txt
        export FLASK_APP=application
        flask run &
        '''
      }
    }
    stage ('test') {
      steps {
        sh '''#!/bin/bash
        source test3/bin/activate
        py.test --verbose --junit-xml test-reports/results.xml
        '''
      }
    }
    post{
      always {
        junit 'test-reports/results.xml'
      }
    }
  }
}
```

Now we'll proceed to deploy our app to AWS Elastic Beanstalk!

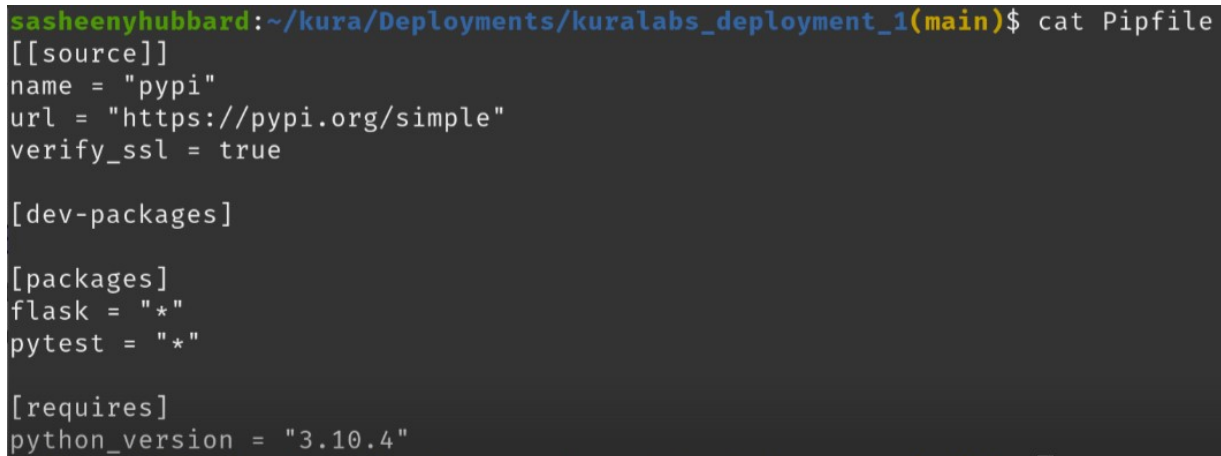
First things first, use the following app to git clone your app source code:

git clone <enter git repo URL address here>

A terminal window titled 'Terminal' with a tab 'ubuntu@ip-172-31-36-229: ~/kuralabs_deployment_1'. The prompt is 'sasheenyhubbard:~/kura/Deployments\$'. The command 'git clone https://github.com/techstacksavvy/kuralabs_deploymen' is entered. The output shows the cloning process: 'Cloning into 'kuralabs_deployment_1'...', 'remote: Enumerating objects: 121, done.', 'remote: Counting objects: 100% (43/43), done.', 'remote: Compressing objects: 100% (9/9), done.', 'remote: Total 121 (delta 37), reused 37 (delta 34), pack-reused 78', 'Receiving objects: 100% (121/121), 9.19 MiB | 7.58 MiB/s, done.', and 'Resolving deltas: 100% (63/63), done.'

```
ubuntu@ip-172-31-36-229: ~/kuralabs_deployment_1
sasheenyhubbard:~/kura/Deployments$ git clone https://github.com/techstacksavvy/kuralabs_deploymen
t_1.git
Cloning into 'kuralabs_deployment_1'...
remote: Enumerating objects: 121, done.
remote: Counting objects: 100% (43/43), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 121 (delta 37), reused 37 (delta 34), pack-reused 78
Receiving objects: 100% (121/121), 9.19 MiB | 7.58 MiB/s, done.
Resolving deltas: 100% (63/63), done.
```

Then cd into the cloned project directory. View the contents of the Pipfile, a file used by Pipenv virtual environment to manage project dependencies. If you look at image below, we can see that the Flask and Pytest packages are installed and that Python3 version 3.10.4 is required to run the Flask application.

A terminal window showing the contents of a file named 'Pipfile'. The prompt is 'sasheenyhubbard:~/kura/Deployments/kuralabs_deployment_1(main)\$'. The command 'cat Pipfile' is entered. The output shows the contents of the Pipfile: '[[source]]', 'name = "pypi"', 'url = "https://pypi.org/simple"', 'verify_ssl = true', '[dev-packages]', '[packages]', 'flask = "*"', 'pytest = "*"', '[requires]', 'python_version = "3.10.4"'.

```
sasheenyhubbard:~/kura/Deployments/kuralabs_deployment_1(main)$ cat Pipfile
[[source]]
name = "pypi"
url = "https://pypi.org/simple"
verify_ssl = true

[dev-packages]

[packages]
flask = "*"
pytest = "*"

[requires]
python_version = "3.10.4"
```

We need to compress the files from our repo in order to deploy it to Elastic Beanstalk. Use the following command to compress the files:

git archive -v -o <app_file_name.zip> --format=zip HEAD

```
ubuntu@ip-172-31-36-229: ~/kuralabs_deployment_1
Terminal
sasheenyhubbard:~/kura/Deployments/kuralabs_deployment_1(main)$ git archive -v -o application.zip
--format=zip HEAD
Deployment-1_Assignment (1).pdf
Jenkinsfile
Pipfile
Pipfile.lock
README.md
application.py
requirements.txt
static/
static/.DS_Store
static/._.DS_Store
static/._bootstrap.min.css
static/._bootstrap.min.js
static/._jquery-3.3.1.slim.min.js
static/._popper.min.js
static/._user_files
static/bootstrap.min.css
static/bootstrap.min.js
static/jquery-3.3.1.slim.min.js
static/popper.min.js
static/user_files/
static/user_files/.DS_Store
static/user_files/._.DS_Store
```

You can use the ls command to confirm that the compressed file has been created as seen below in RED.

```
sasheenyhubbard:~/kura/Deployments/kuralabs_deployment_1(main)$ ls -l
total 18780
-rwxrwxr-x 1 sasheenyhubbard sasheenyhubbard 2118 Oct 2 20:43 application.py
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 18176926 Oct 2 20:58 application.zip
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 1003285 Oct 2 20:43 'Deployment-1_Assignment (1).pdf'
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 615 Oct 2 20:43 Jenkinsfile
-rwxrwxr-x 1 sasheenyhubbard sasheenyhubbard 166 Oct 2 20:43 Pipfile
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 8575 Oct 2 20:43 Pipfile.lock
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 52 Oct 2 20:43 README.md
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 571 Oct 2 20:43 requirements.txt
drwxrwxr-x 3 sasheenyhubbard sasheenyhubbard 4096 Oct 2 20:43 static
drwxrwxr-x 2 sasheenyhubbard sasheenyhubbard 4096 Oct 2 20:43 templates
-rw-rw-r-- 1 sasheenyhubbard sasheenyhubbard 115 Oct 2 20:43 test_app.py
-rwxrwxr-x 1 sasheenyhubbard sasheenyhubbard 246 Oct 2 20:43 urls.json
```

Open Elastic Beanstalk in your AWS console and create a new Web-server environment. After entering the following configurations:

Application name: url-shortner

Environment name: Urlshortner-env

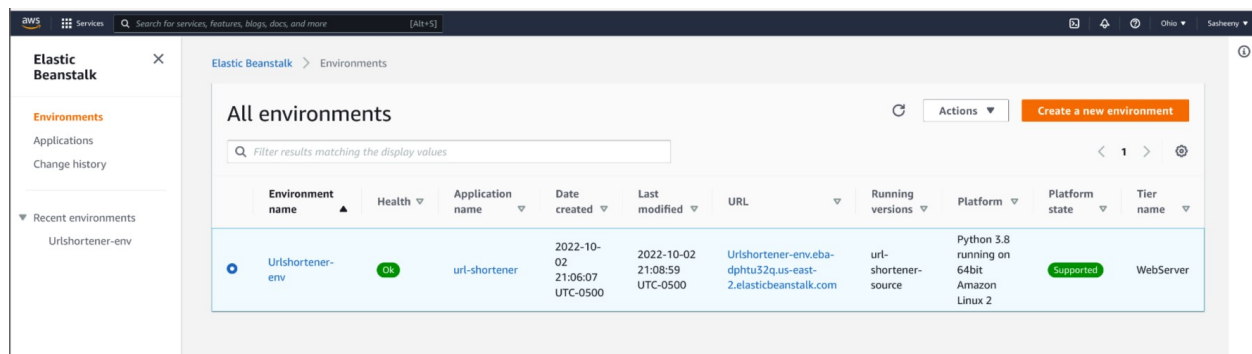
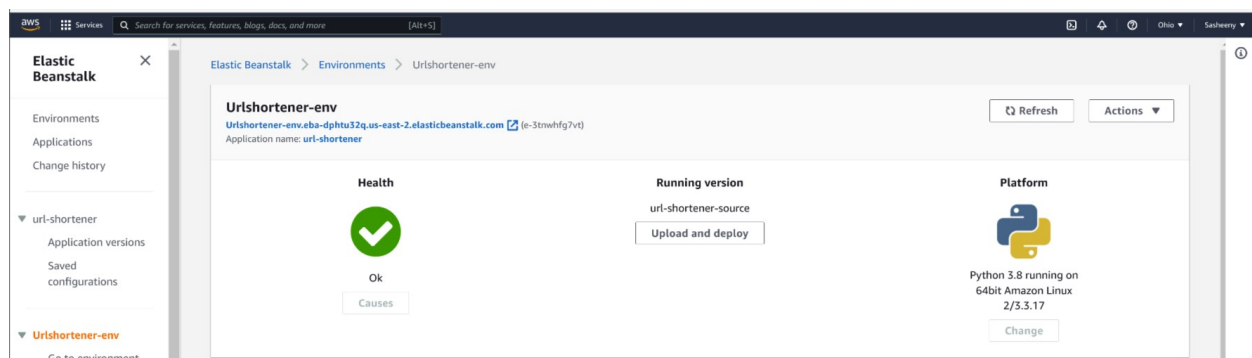
Platform: Python

Platform branch: 3.8

Platform version: 3.3.16

Application code: Upload your code, local file(your zip file)

Once created check the health of your environment.



Once the environment is created copy and paste the generated URL code into a web browser.

The screenshot shows a web browser with multiple tabs. The active tab is 'URL Shortener' at the address 'urlshortener-env.eba-dphtu32q.us-east-2.elasticbeanstalk.com'. The page has a header with the title 'URL Shortener' and a 'New URL' button. The main content area contains two side-by-side forms: 'Website' and 'File'. Both forms have a 'Short Name' input field and a 'Website URL' input field. The 'File' form also includes a 'Choose File' button and the text 'No file chosen'. Both forms have a blue 'Shorten' button at the bottom.

URL Shortener API New URL

Website

Short Name

Website URL

Shorten

File

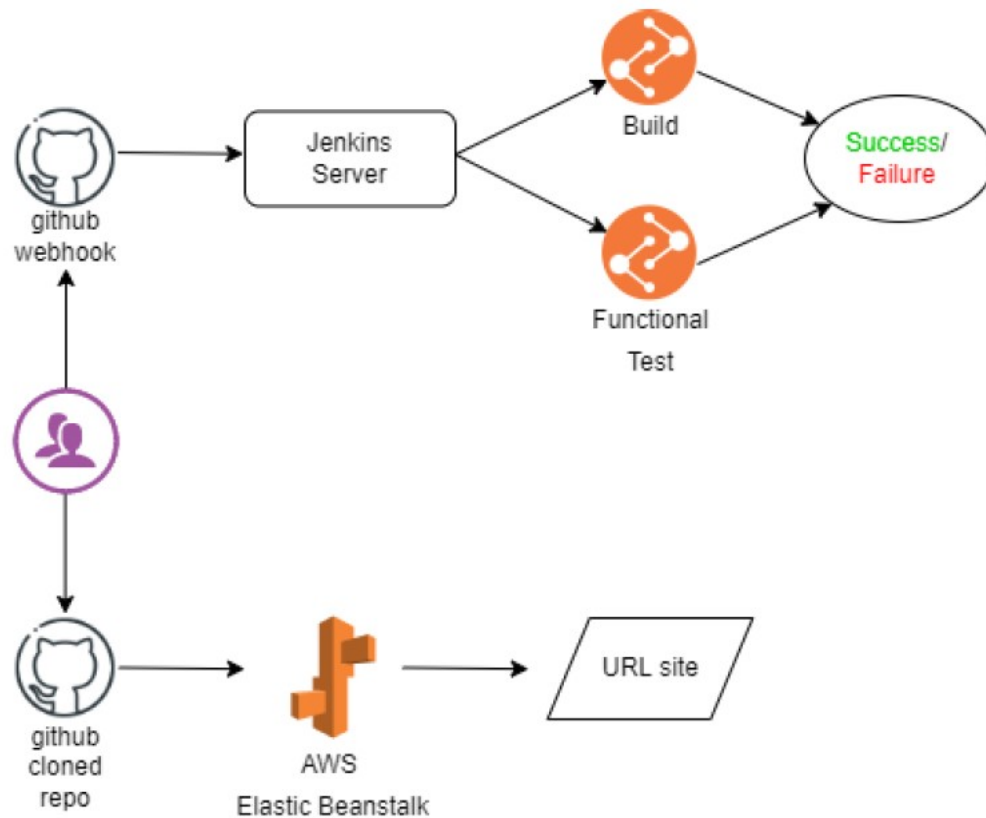
Short Name

Website URL

No file chosen

Shorten

Diagram of the Pipeline



Recommendations for Improvement

- Add an AWS SNS to notify client/user in the event of an error/failure in Build Test
- Add a database to store artifacts