kura labs

Deployment #2

The goal of this deployment is to set up a CI/CD pipeline from start to finish. I used Elastic Beanstalk and customized the pipeline.

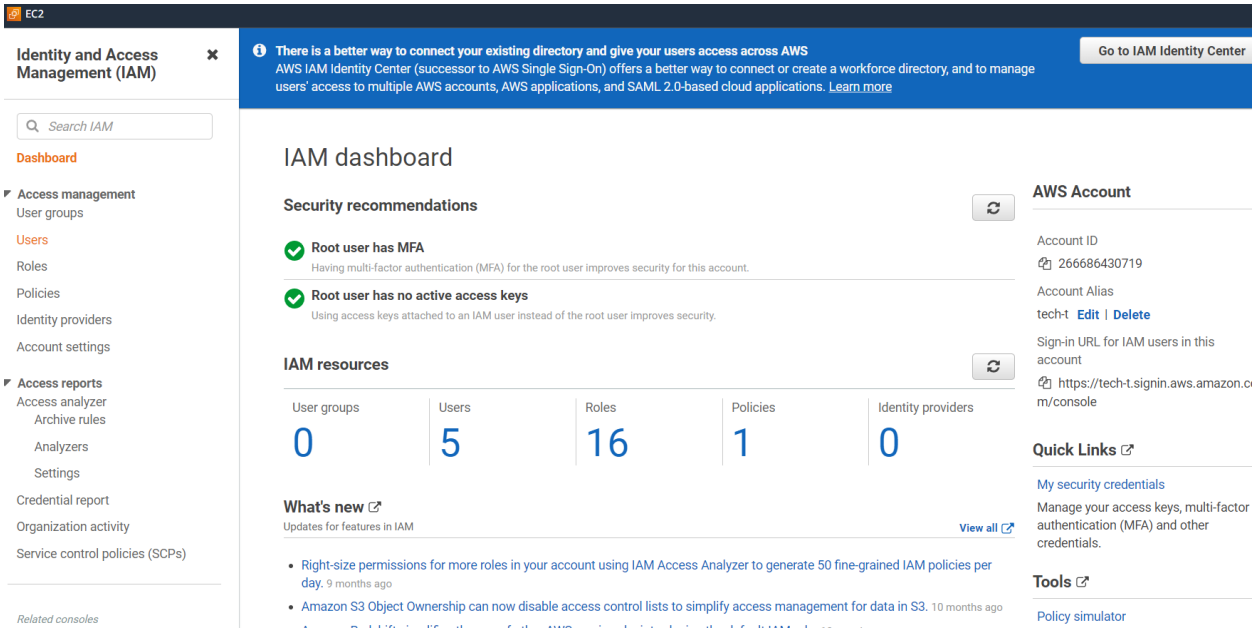
Step 1: Install Jenkins on an EC2

Step 2: Activate the Jenkins user on the EC2:

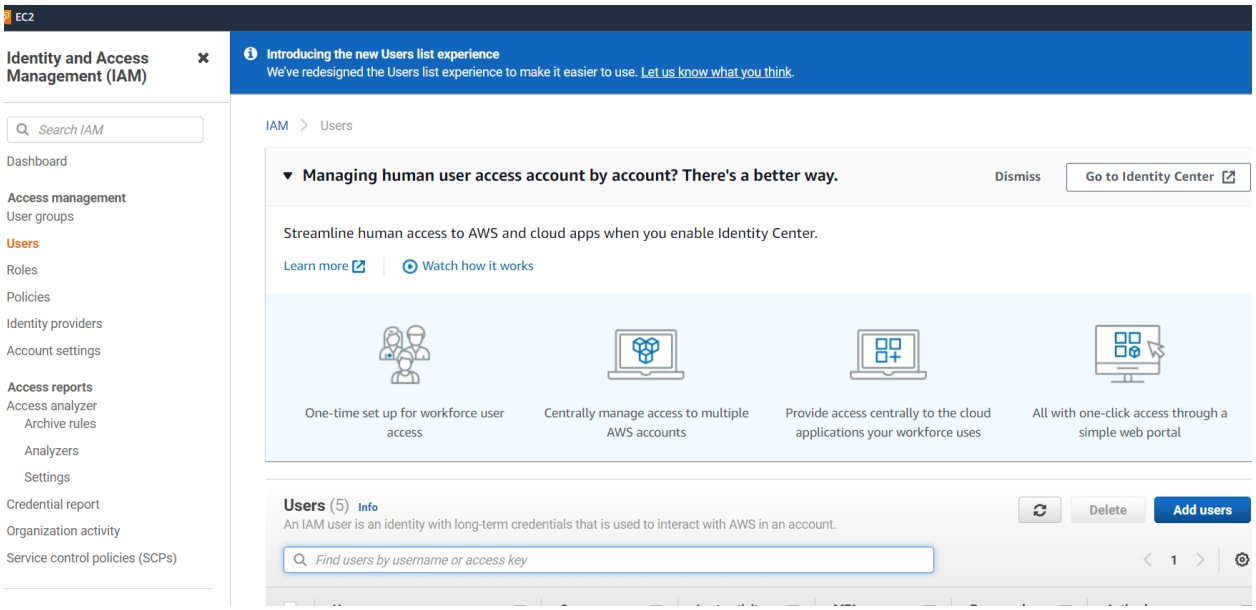
```
$sudo passwd jenkins
$sudo su - jenkins -s /bin/bash
```

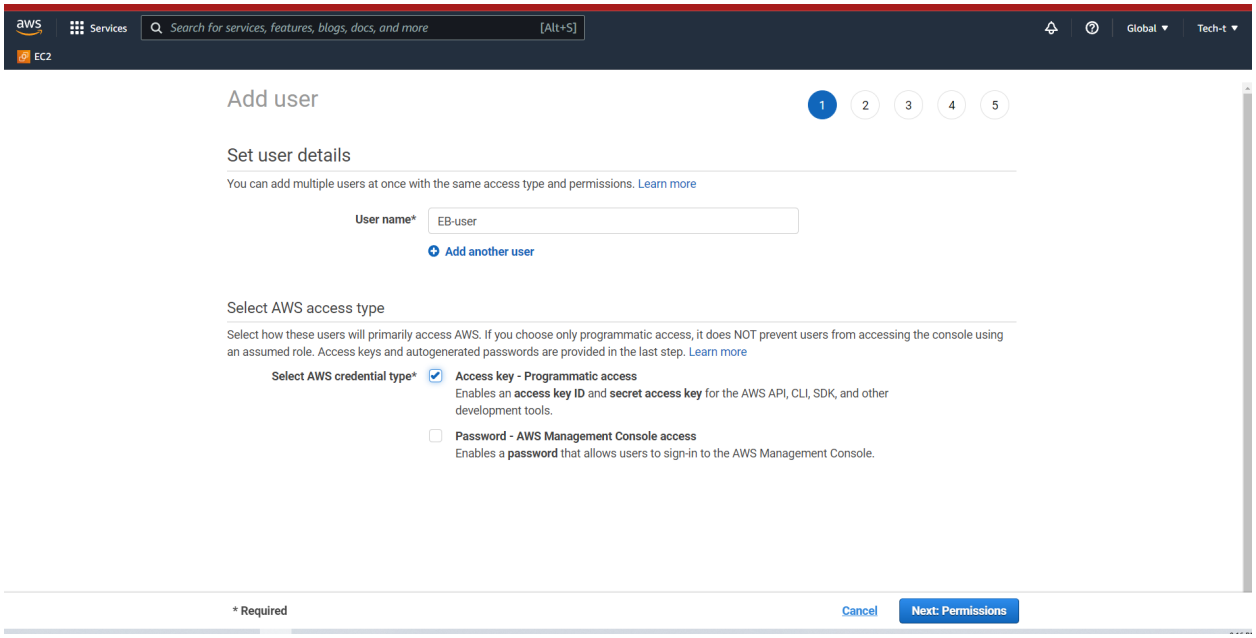
Step 3: Create a Jenkins user in my AWS account:

- I navigated to IAM in the AWS console. Next, I clicked on the Users option in the Access management.

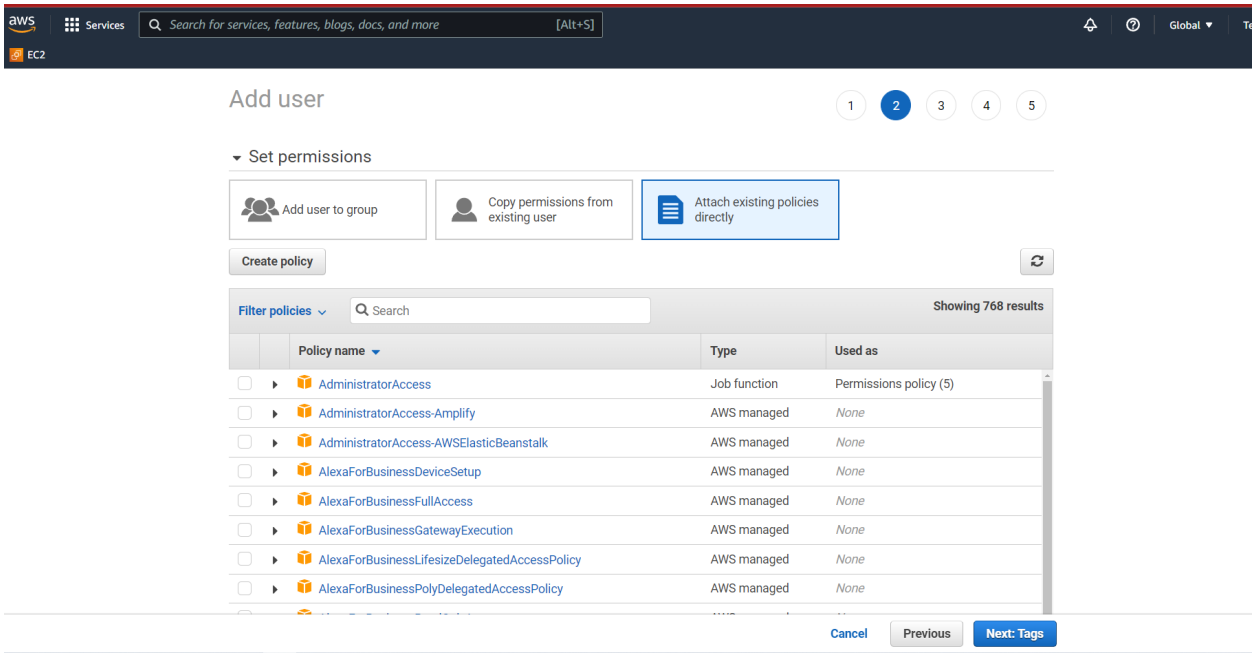


- I selected add user. The username I added was EB-user. I then selected Programmatic access and then I clicked next.

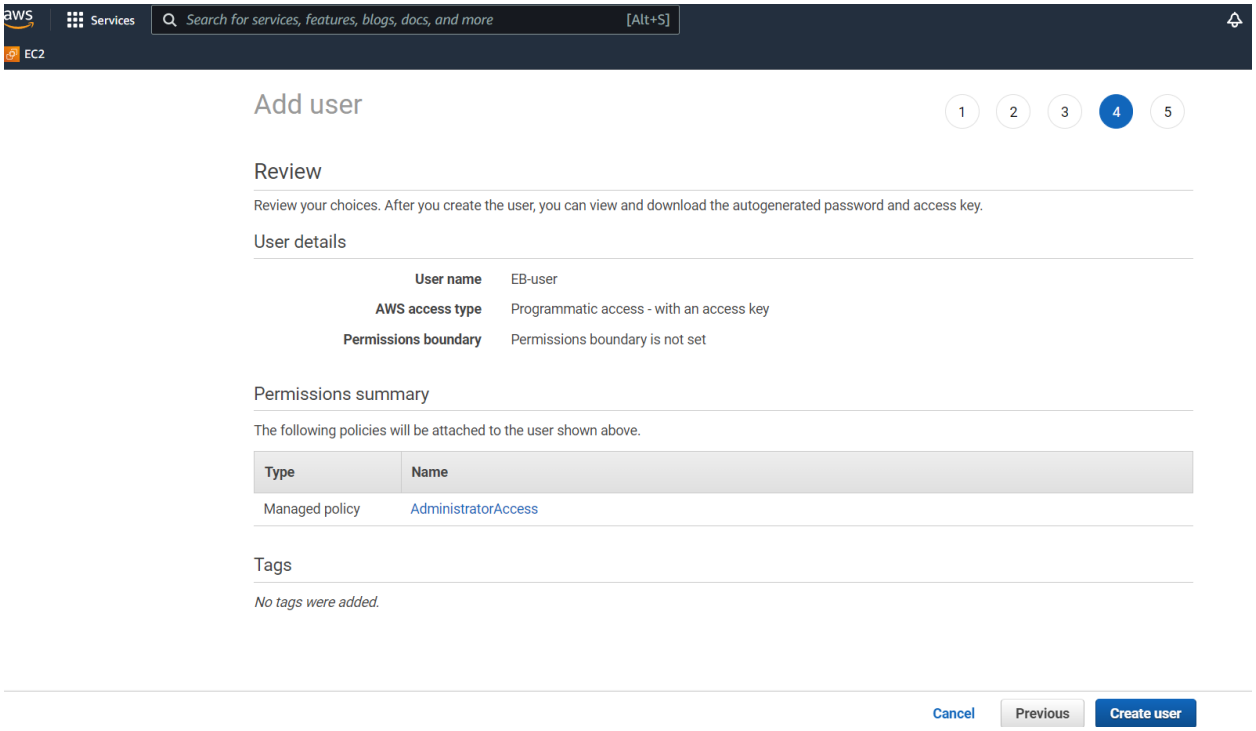


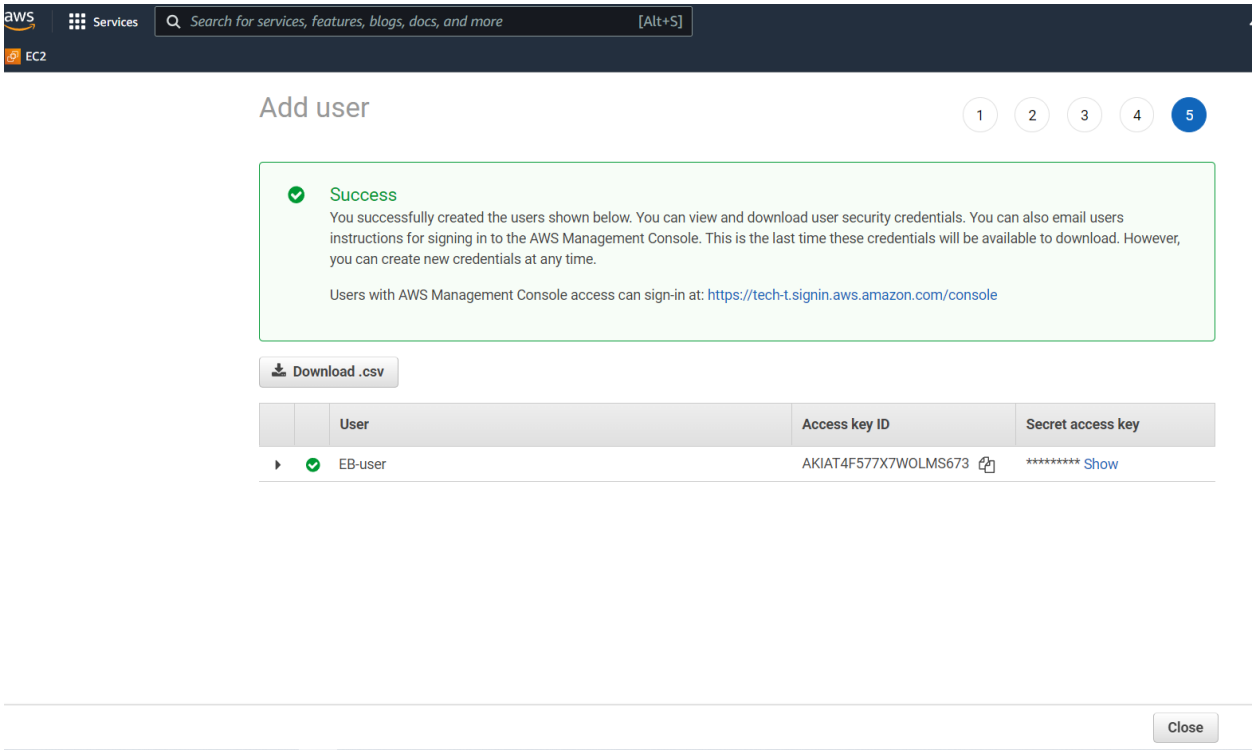


- Next, I selected “Attach existing policies directly” and select administrator access. Then I selected next for the current page as well as the next pages.



- Finally, I created the user and then copied and saved the “access key ID” and the “secret access key” to a text file on my PC.





Step 4: Install AWS CLI on the Jenkins EC2 and configure:

- Using the curl command I downloaded AWS CLI files from the AWS CLI website
- I had difficulty with this step because the user was not part of the sudoers list and unzip was not installed on the jenkins user
- So I had to exit the jenkins user, and add jenkins to sudoers on the ubuntu EC2.
- Then as a Jenkins user I installed the unzip command and proceeded to configure.

```
$curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
$exit
$sudo usermod -aG sudo jenkins
sudo su - jenkins -s /bin/bash
$sudo apt install unzip
```

```
$sudo ./aws/install
$unzip awscliv2.zip

$aws --version
$aws configure
  - Set Access Key ID
  - Set Secret Access Key
  - Set region to: us-east-1
  - Set Output format: json
```

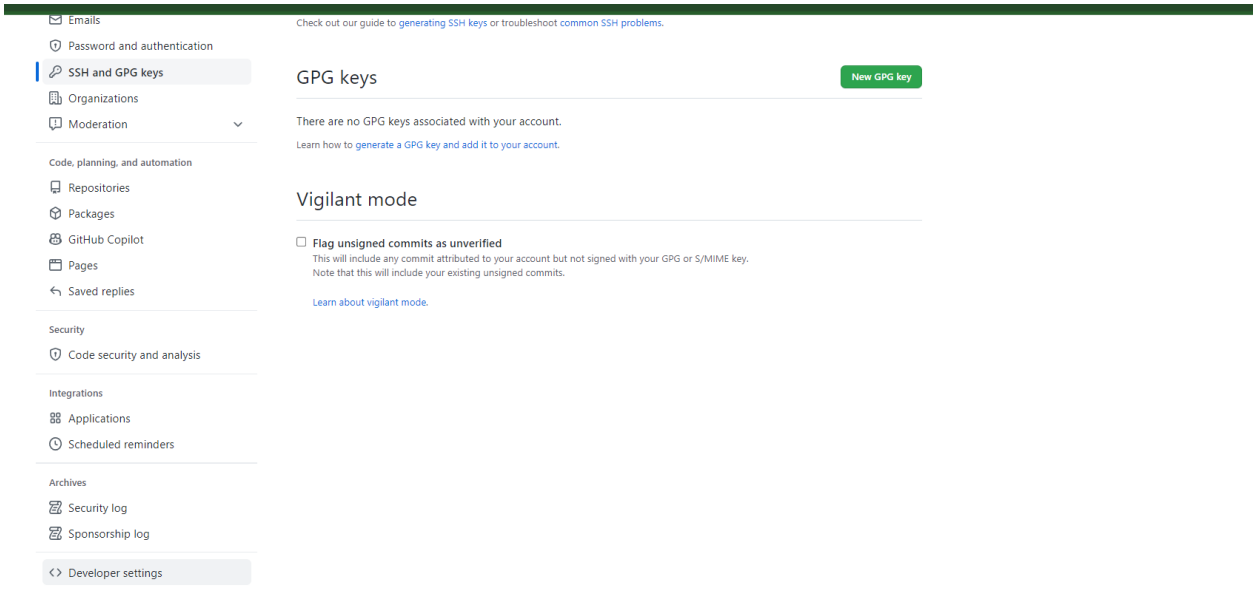
Step 5: Install EB CLI in the jenkins EC2 user:

- **I tried to install EB CLI but my jenkins user did not have the pip command**
- **So I had to install the pip command using $sudo apt install python3-pip**
- **Then, I used the below commands to install the EB CLI in the Jenkins EC2 user**

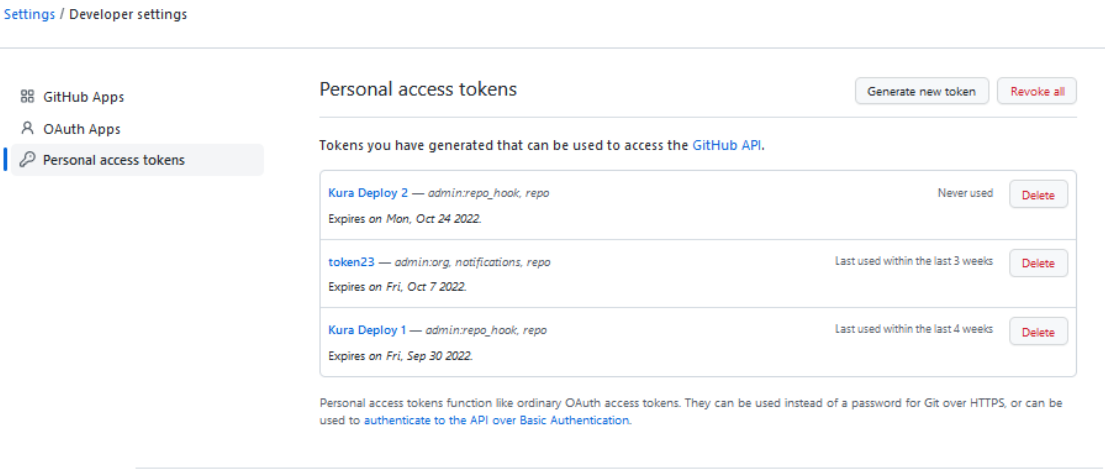
```
$pip install awsebcli --upgrade --user
$eb --version
```

Step 6: Connect GitHub to Jenkins Server:

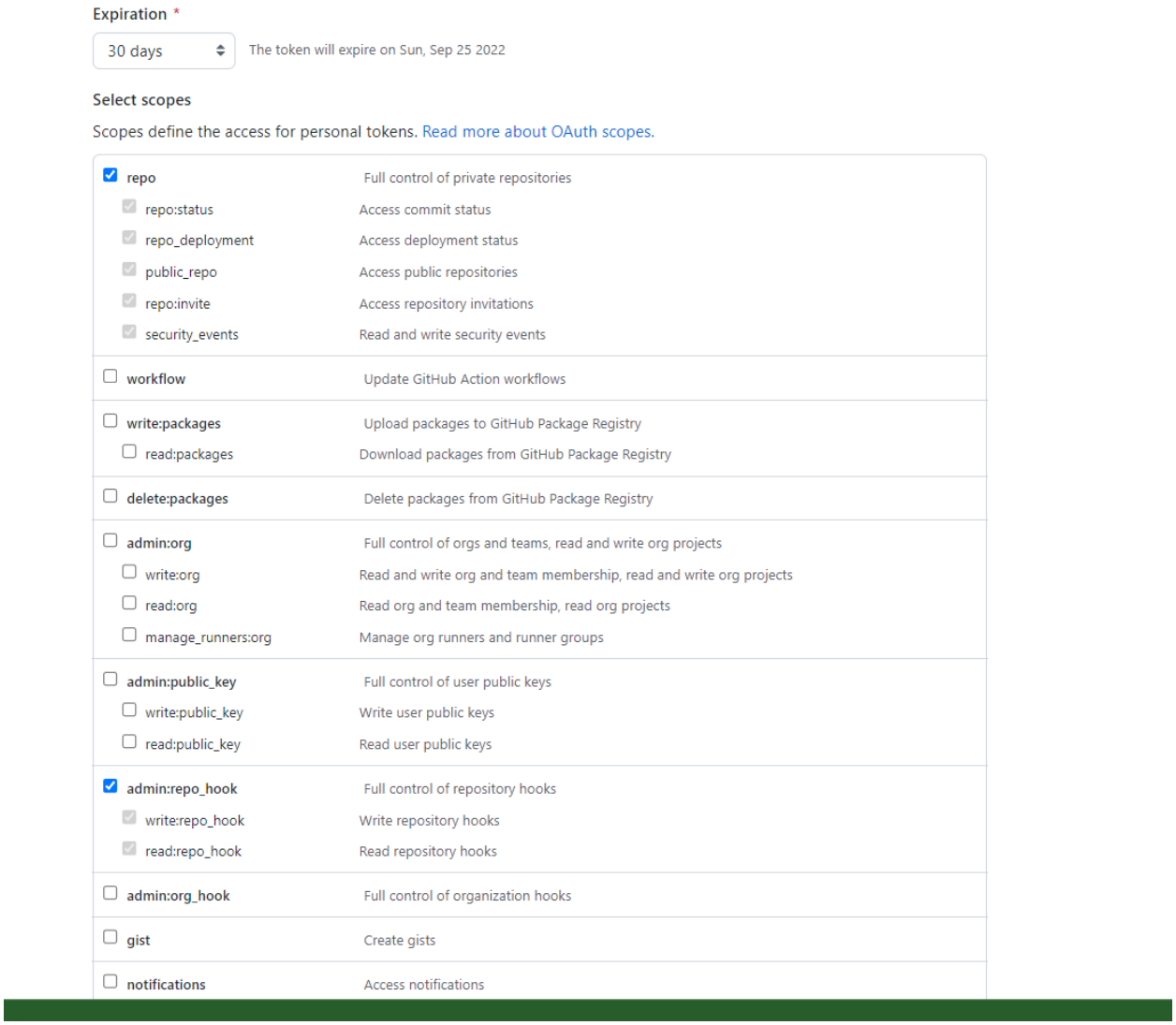
- I first forked the Deployment repo: https://github.com/kura-labs-org/kuralabs_deployment_2.git
- Next, I created an access token from GitHub:
    - I navigated to my GitHub settings, and selected developer settings



- I then selected a personal access token and generated a new token.

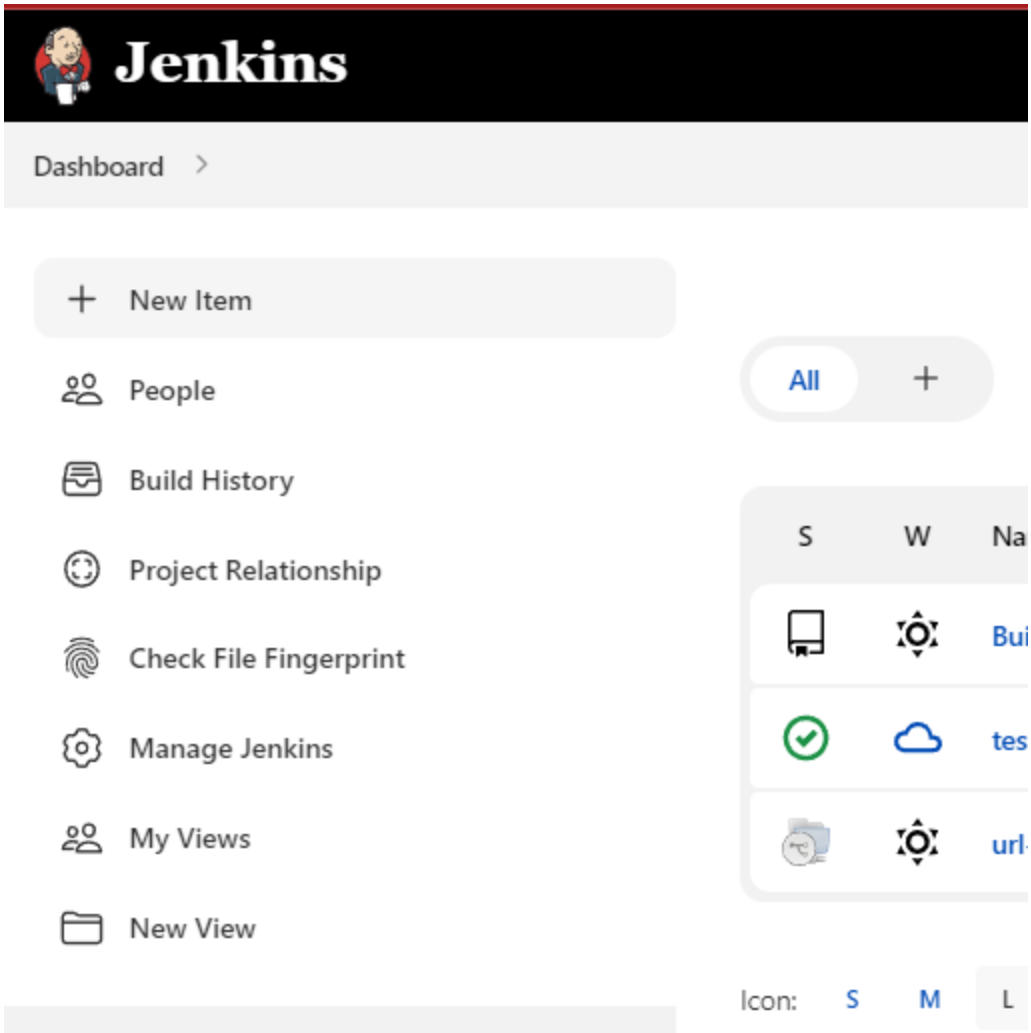


- Select the settings you see below for access token permissions.

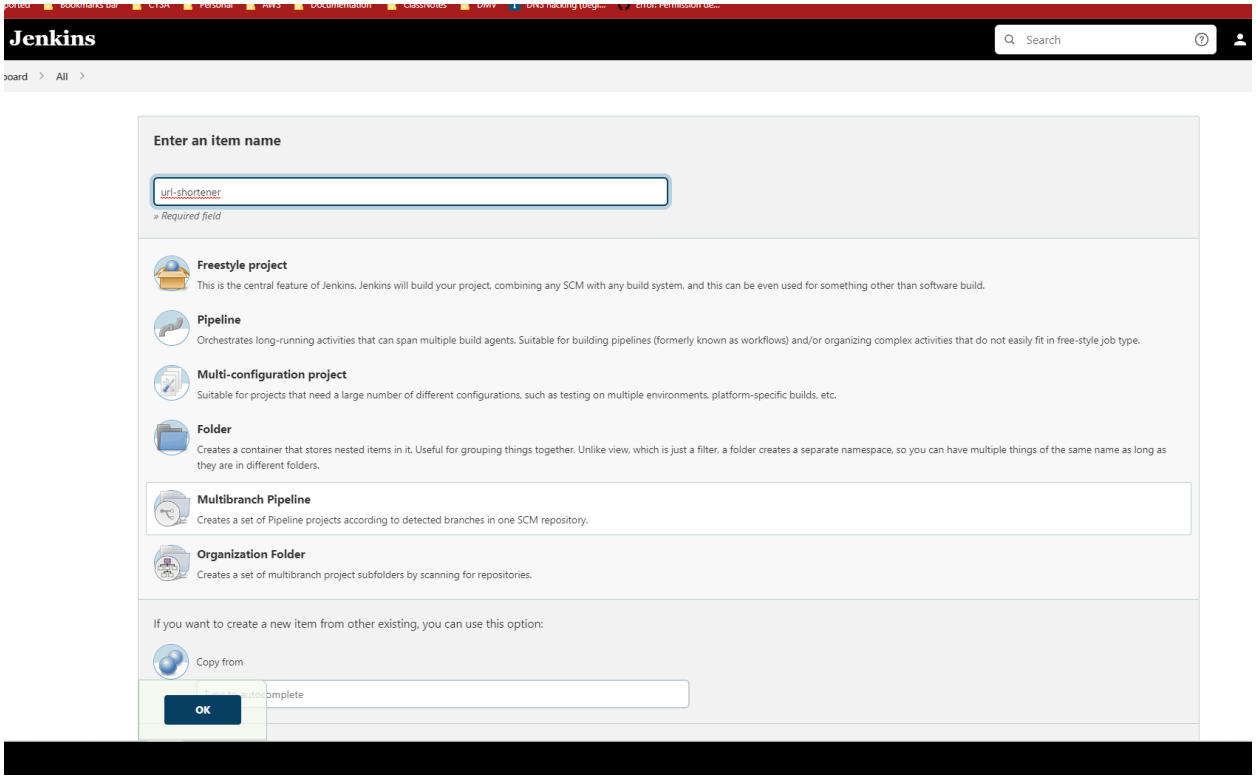


Step 7: Create a multibranch build:

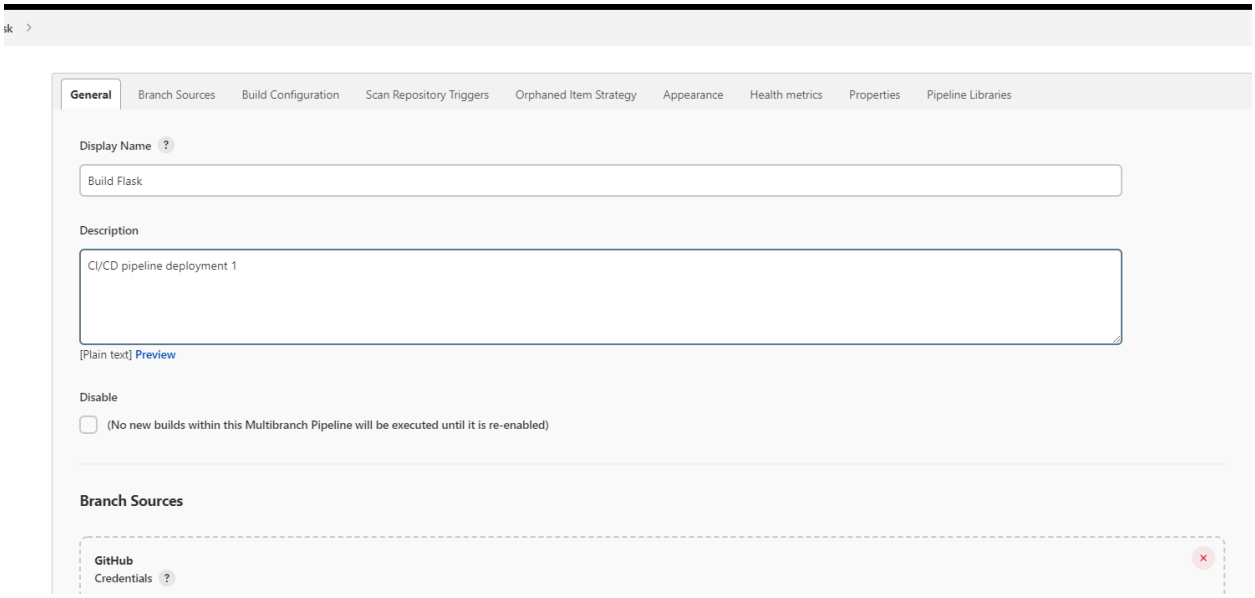
- I logged back into Jenkins and selected “New item”


Jenkins
Dashboard
New Item
People
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views
New View
All
S
W
Na
Bui
tes
url
Icon:
S
M
L
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle

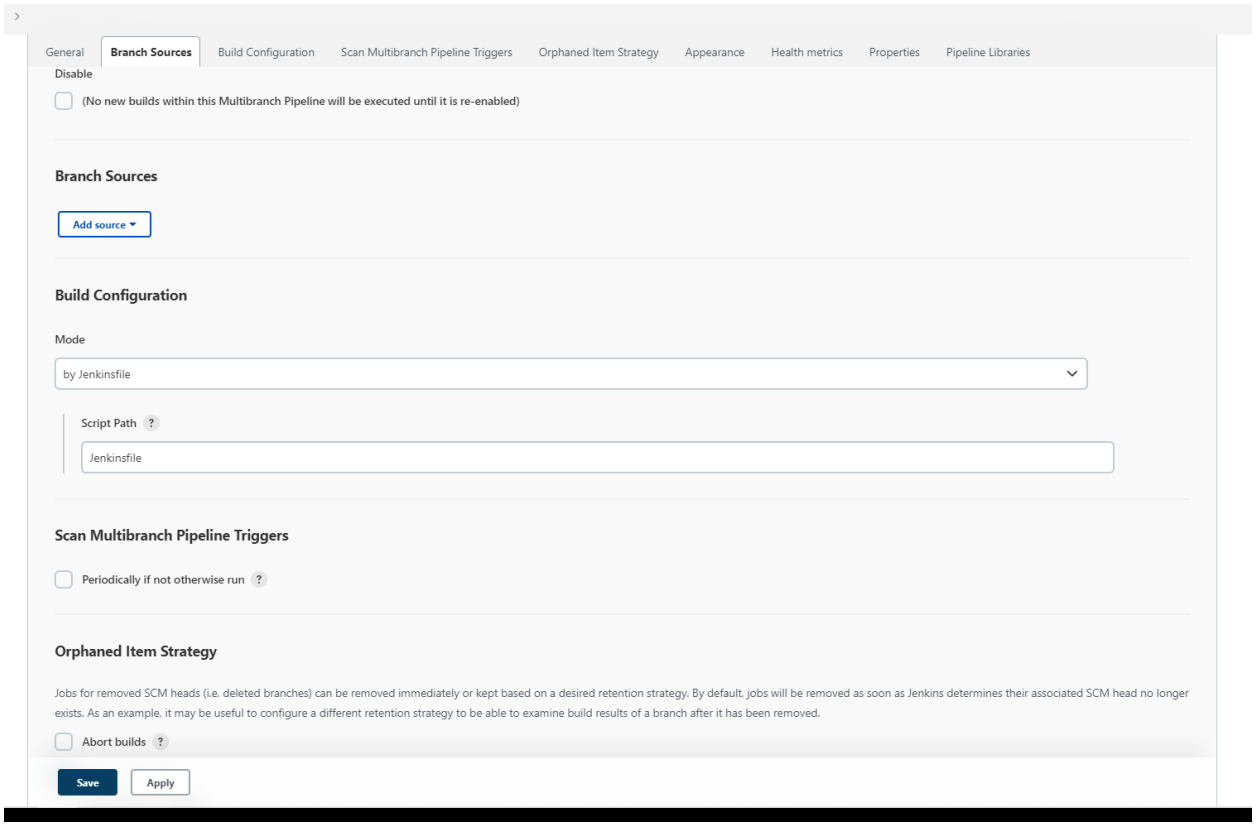
- I entered the item name as "url-shortener".
- Then I selected a multibranch pipeline.



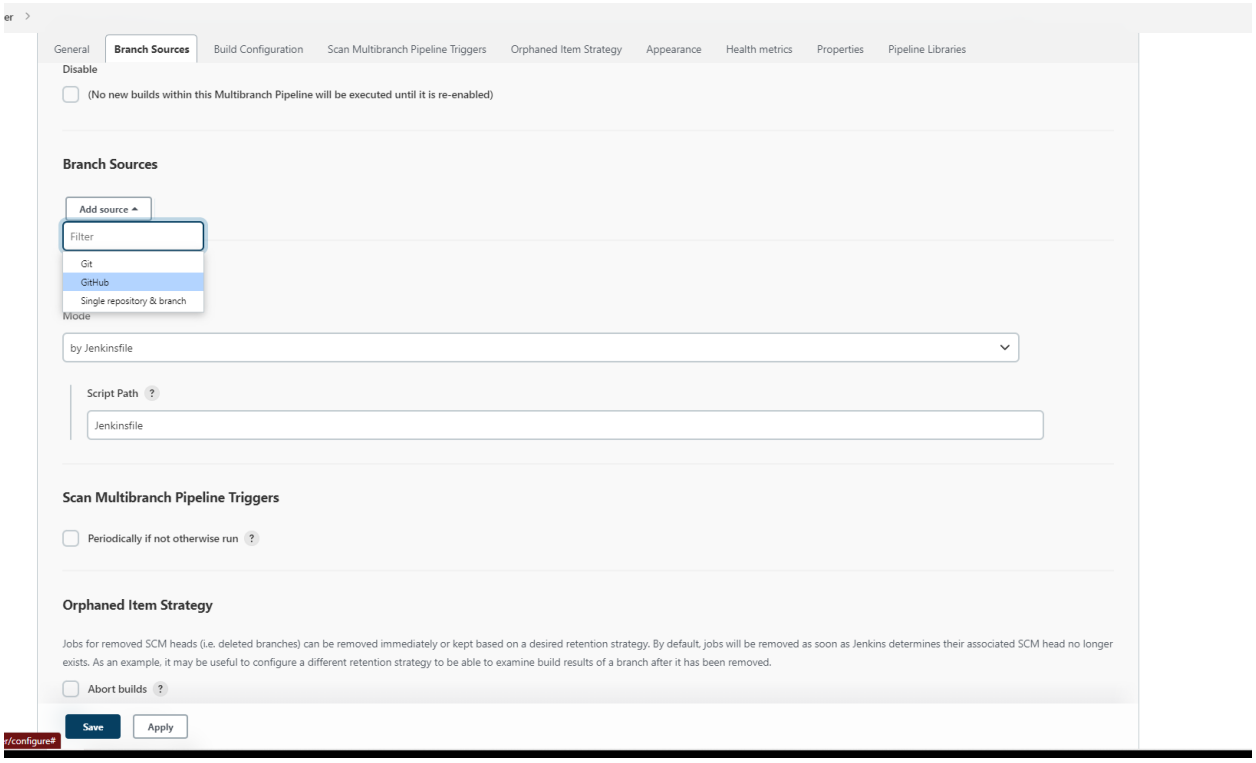
- I entered the display name as "Build Flask".
- Then I entered a brief description.



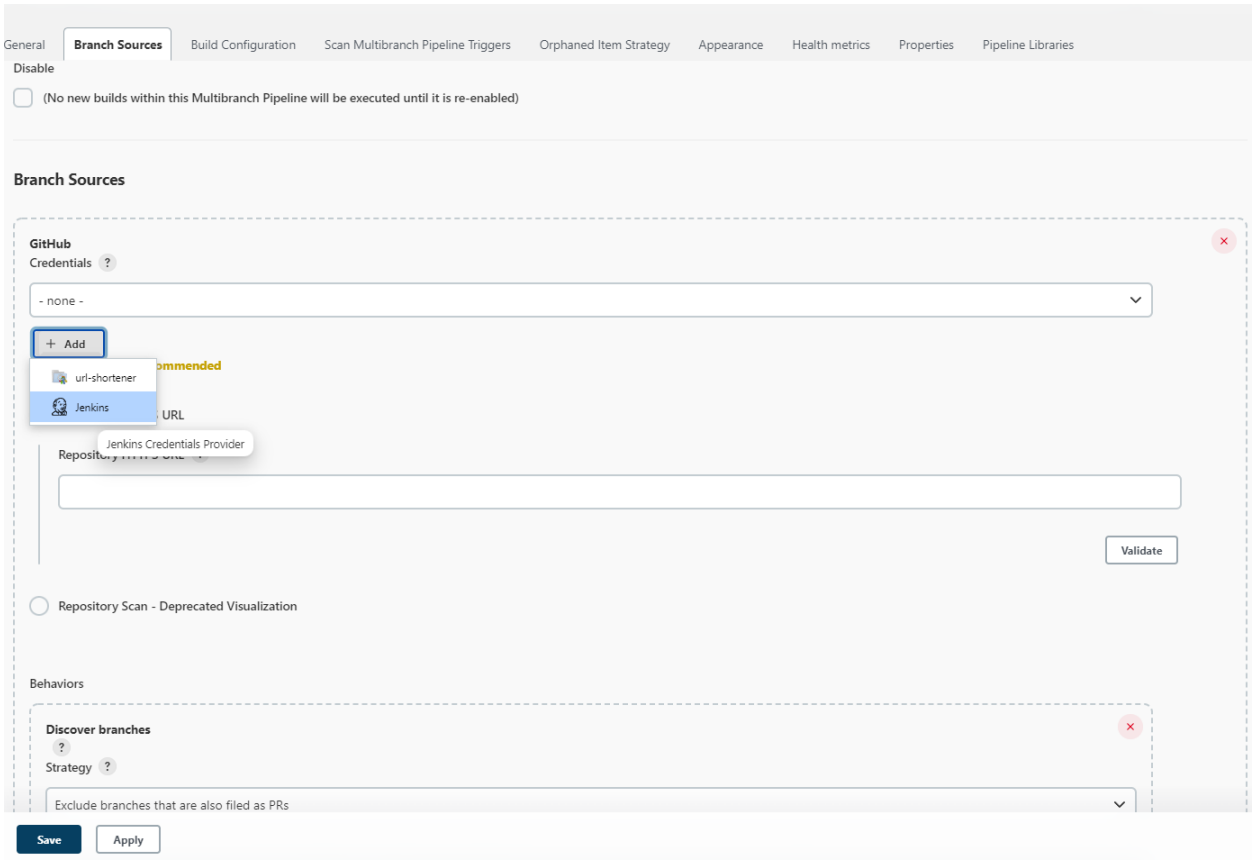
- I added a Branch source by selecting Add source and then selecting GitHub.



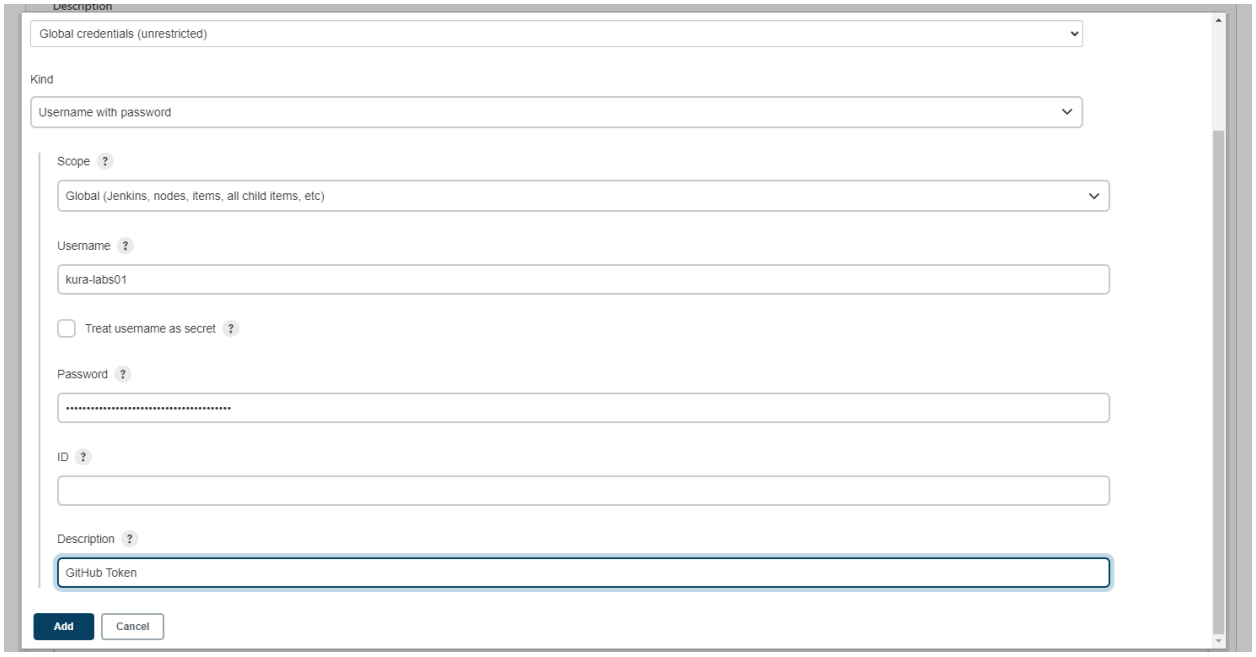
- I then selected the Add button and selected GitHub.



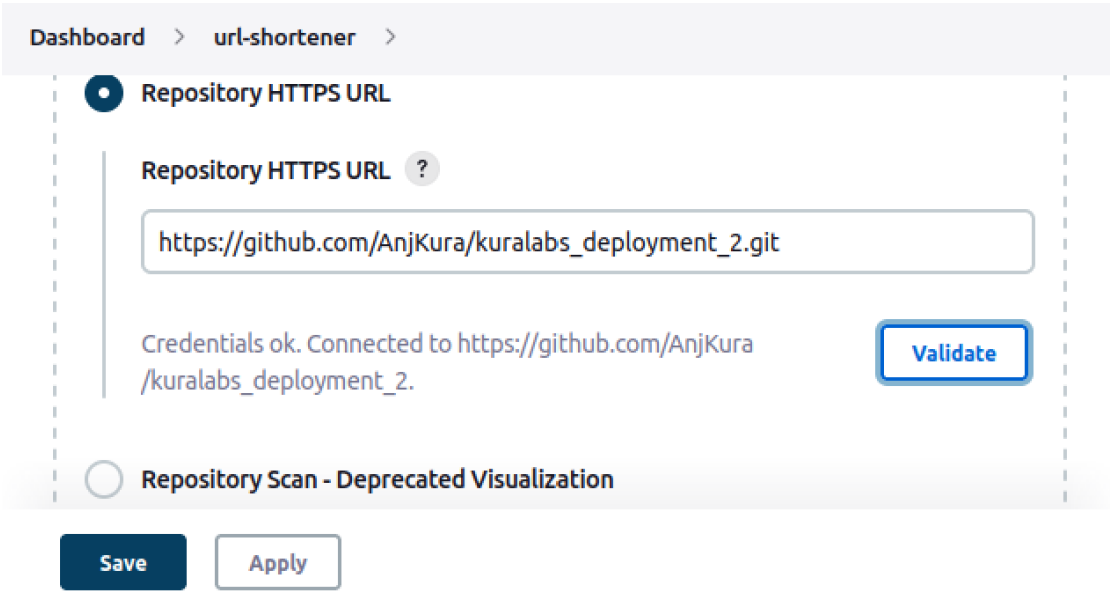
- I clicked on Add and then selected Jenkins



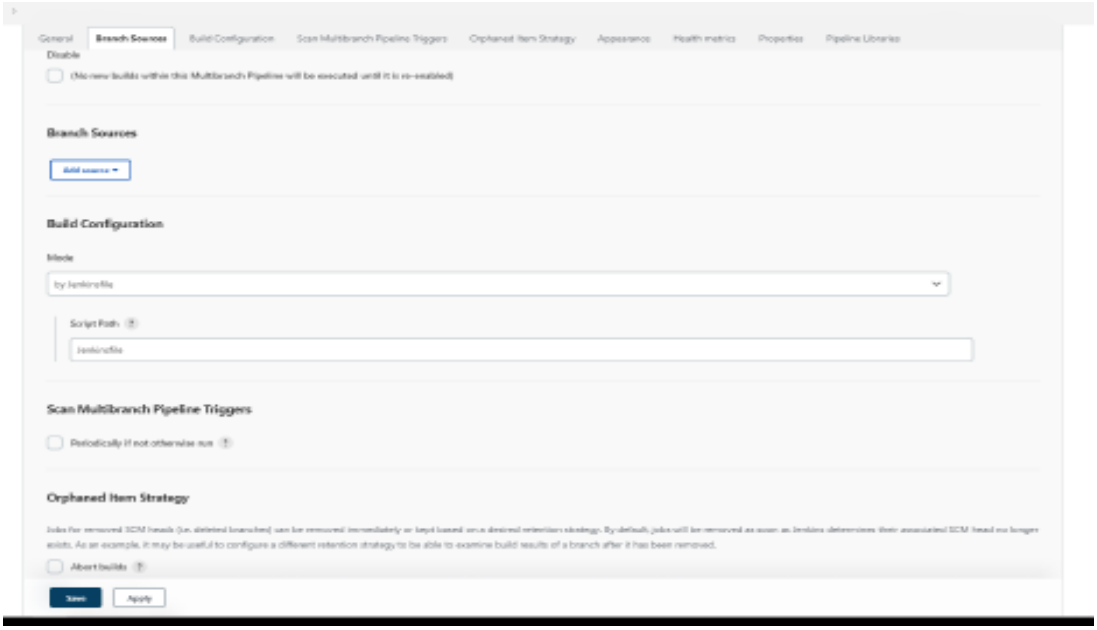
- Under username I entered my GitHub username "AnjKura"
- Under password I entered my token from GitHub.



- I entered the URL to my repository "https://github.com/AnjKura/kuralabs_deployment_2.git".
- Then, I validated by selecting validate.



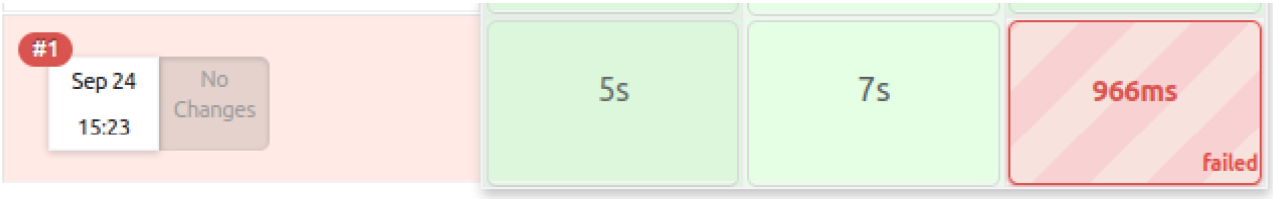
- I made sure that the Build Configuration says Jenkinsfile.



- I selected Apply and then Save.

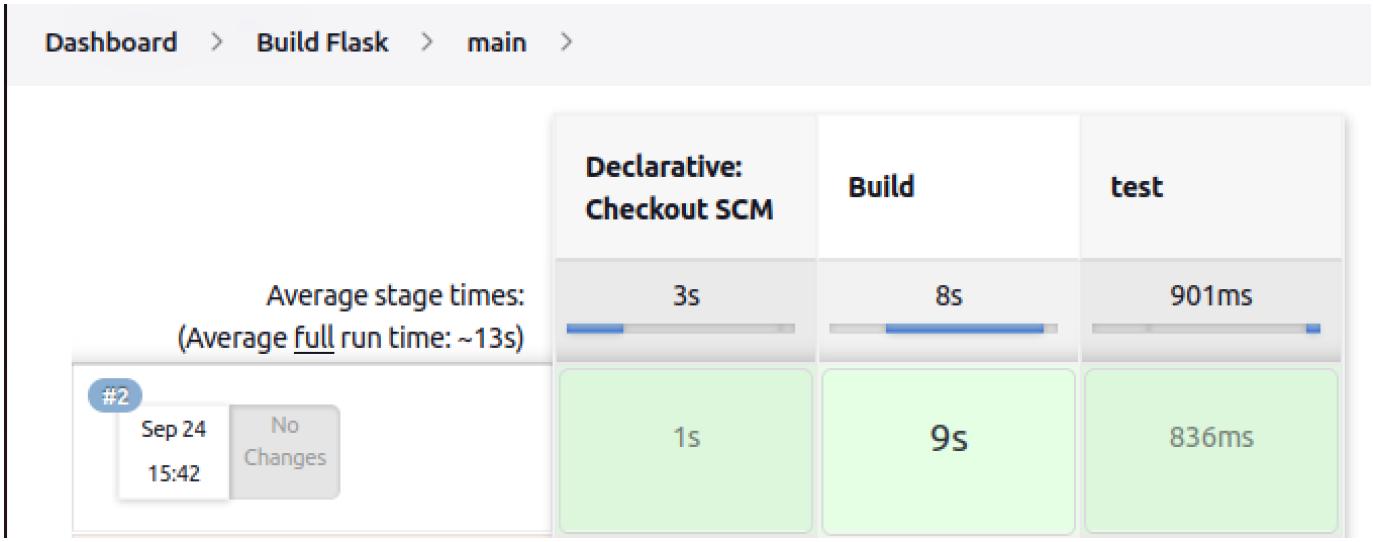
- I did not see a Build happening right away, so I selected Scan Repository Now

- After I scanned the repository, I found out that my first build failed.



- I checked the logs and found that I needed to install python on my EC2.

- After I installed python, I ran the Build again and the Build was successful.



Step 8: Deploy the application from Elastic Beanstalk CLI:

- I used sudo to initialize the jenkins user again.
- I changed to directories in order to enter the workspace where the url-shortener_main is located
- I ran $pip install awsebcli --upgrade --user
- Then, I ran export PATH="/var/lib/jenkins/.local/bin: $PATH"

-

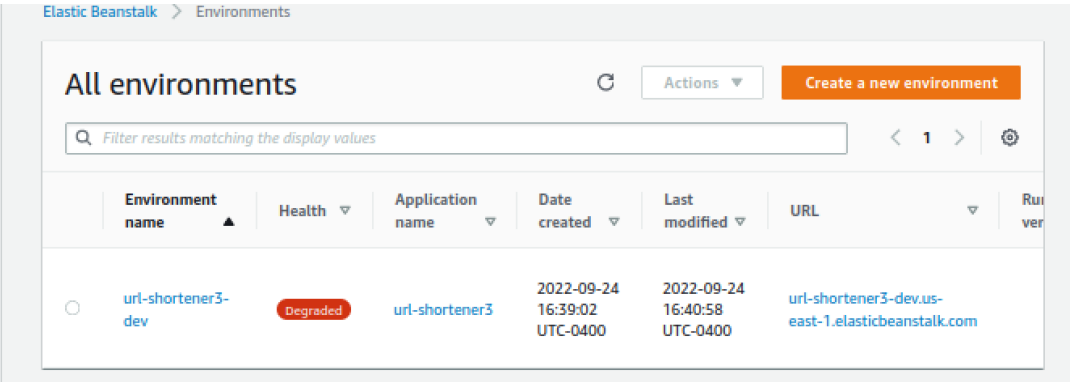
```
$sudo su - jenkins -s /bin/bash
$cd workspace/url-shortener_main/
$eb init
```

- Select: us-east-1
- Press enter
- Select application to use: url-shortener3
- Select: Python
- Select: (The latest version of python available)
- Select: N (for CodeCommit)
- Set up SSH for instance: Yes
- Select keypair: I selected the keypair ssh2

```
$eb create
```

- I entered the default for the next 3 questions by hitting enter
- My environment name was url-shortener3-dev

- Spot Fleet: No
- I waited for the environment to be made.
- When I checked the environment on elastic beanstalk it was degraded.



Step 9 I attempted to add a deployment stage to the pipeline in the Jenkinsfile:

```
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 {
```

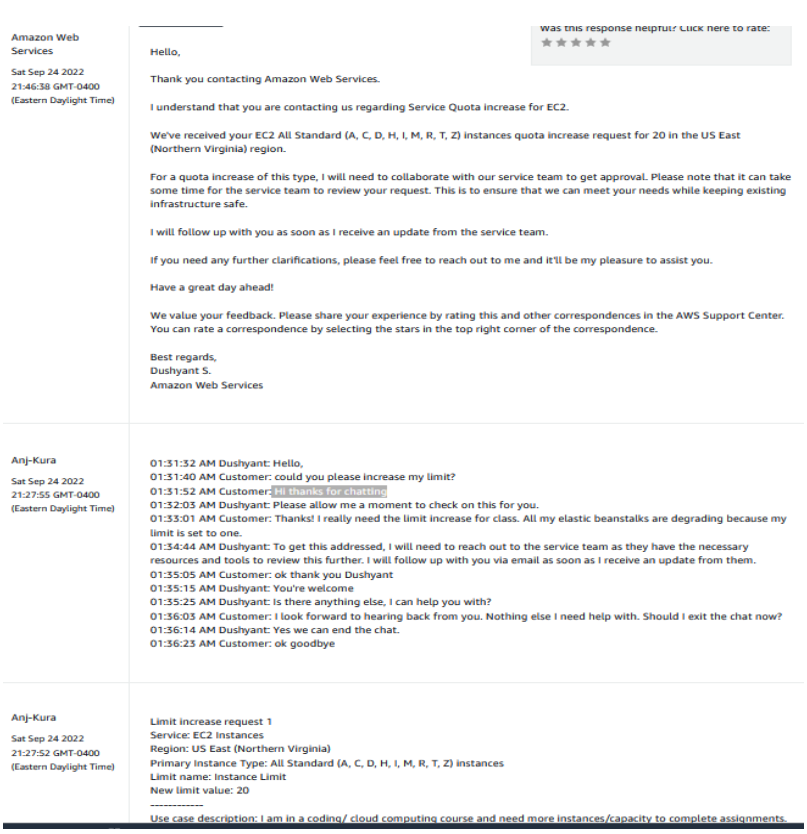
```
          junit 'test-reports/results.xml'
        }

      }
    }
    stage ('Deploy') {
      steps {
        sh '/var/lib/jenkins/.local/bin/eb deploy url-shortener3-dev5'

      }
    }
  }
}
```

- I could not successfully deploy the application because my url shortener environment was degraded.
- I received the following error message: "Environment health has transitioned from Pending to Degraded. Auto Scaling activity failed 35 seconds ago with error: You have requested more vCPU capacity than your current vCPU limit of 1 allows for the instance bucket that the specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit. Launching EC2 instance failed. At 2022-09-25T01:08:00Z a user request update of AutoScalingGroup constraints to min: 1, max: 4, desired: 1 changing the desired capacity from 0 to 1. At 2022-09-25T01:08:09Z an instance was started in response to a difference between desired and actual capacity, increasing the capacity from 0 to 1. Initialization in progress (running for 87 seconds). There are no instances. Auto Scaling group desired capacity is set to 1."

- Richard Deodutt from class worked with me for over 2 hours to try to find the reason for this capacity issue .
- We found that since my vCPU capacity is only one, I was not able to run elastic beanstalk and Jenkins EC2 at the same time.
- I contacted AWS support to request assistance for increasing my instance quota. I am awaiting Amazon's response.



Amazon Web Services

Sat Sep 24 2022 21:46:38 GMT-0400 (Eastern Daylight Time)

Was this response helpful? Click here to rate:

Hello,

Thank you contacting Amazon Web Services.

I understand that you are contacting us regarding Service Quota increase for EC2.

We've received your EC2 All Standard (A, C, D, H, I, M, R, T, Z) instances quota increase request for 20 in the US East (Northern Virginia) region.

For a quota increase of this type, I will need to collaborate with our service team to get approval. Please note that it can take some time for the service team to review your request. This is to ensure that we can meet your needs while keeping existing infrastructure safe.

I will follow up with you as soon as I receive an update from the service team.

If you need any further clarifications, please feel free to reach out to me and it'll be my pleasure to assist you.

Have a great day ahead!

We value your feedback. Please share your experience by rating this and other correspondences in the AWS Support Center. You can rate a correspondence by selecting the stars in the top right corner of the correspondence.

Best regards,
Dushyant S.
Amazon Web Services

Anj-Kura

Sat Sep 24 2022 21:27:55 GMT-0400 (Eastern Daylight Time)

01:31:32 AM Dushyant: Hello,
01:31:40 AM Customer: could you please increase my limit?
01:31:52 AM Customer: Hi thanks for chatting
01:32:03 AM Dushyant: Please allow me a moment to check on this for you.
01:33:01 AM Customer: Thanks! I really need the limit increase for class. All my elastic beanstalks are degrading because my limit is set to one.
01:34:44 AM Dushyant: To get this addressed, I will need to reach out to the service team as they have the necessary resources and tools to review this further. I will follow up with you via email as soon as I receive an update from them.
01:35:05 AM Customer: ok thank you Dushyant
01:35:15 AM Dushyant: You're welcome
01:35:25 AM Dushyant: Is there anything else, I can help you with?
01:36:03 AM Customer: I look forward to hearing back from you. Nothing else I need help with. Should I exit the chat now?
01:36:14 AM Dushyant: Yes we can end the chat.
01:36:23 AM Customer: ok goodbye

Anj-Kura

Sat Sep 24 2022 21:27:52 GMT-0400 (Eastern Daylight Time)

Limit increase request 1
Service: EC2 Instances
Region: US East (Northern Virginia)
Primary Instance Type: All Standard (A, C, D, H, I, M, R, T, Z) instances
Limit name: Instance Limit
New limit value: 20

Use case description: I am in a coding/ cloud computing course and need more instances/capacity to complete assignments.

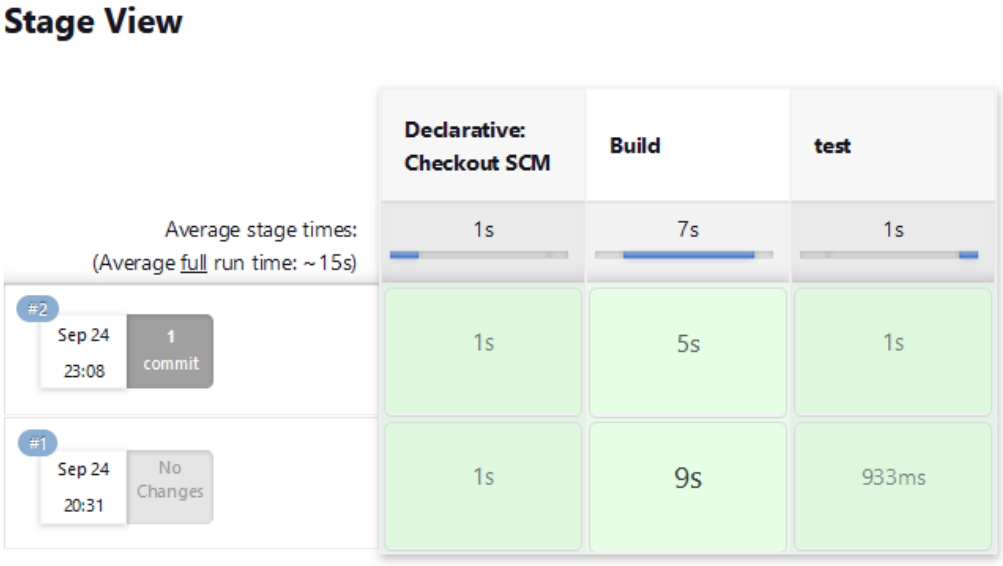
Step 10: I added an extra test to the pipeline.

- I added the below code to the test_app.py file

```
# def test_fast():
#   b = "Bobby"
#   new_greeting = greet(b)
#   assert new_greeting == "Good Evening Bobby"

def test_home_page():
    response = app.test_client().get('/')
    assert response.status_code == 200
```

- I built the pipeline again and the test ran successfully.



Stage View

	Declarative: Checkout SCM	Build	test
Average stage times: (Average full run time: ~15s)	1s	7s	1s
#2 Sep 24 23:08 — 1 commit	1s	5s	1s
#1 Sep 24 20:31 — No Changes	1s	9s	933ms

What could have been done differently:

- To install the eb command I could have added the path to the bash.rc file so that eb would have been added to my path. Since I only installed the eb on the Jenkins user, it needed to be reinstalled every time I stopped and restarted the instance.

- To prevent my initial build from failing, I should have installed python on my EC2, especially because the url-shortener is a python application.

- I could have also made sure that I had adequate vCPU capacity before I began the deployment.