# ATTACK DEFENSE

by PentesterAcademy

| Name | DevOps Pipeline: Nginx |
|---|---|
| URL | https://attackdefense.com/challengedetails?cid=2070 |
| Type | Pipeline Basics: Software |

**Important Note:** This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

## Challenge Description

DevOps practices are to combine software development (Dev) and IT operations (Ops) in order to improve the delivery process. DevOps pipelines are chained tasks and components that run in a sequence to cover different phases of software compilation, packaging, automated testing, and test deployment.

In this lab, we have a simple DevOps pipeline for Nginx. The pipeline consists of the following components (and tasks):

- Kali machine (For pulling, modifying, and pushing the code)
- GitLab server (For hosting code)
- Jenkins server (For integrating). Different phases and components used:
  - Building Binary: make
  - Test Deployment: Ansible
  - Dynamic Testing: Selenium
- Test server (For test deployment)

**Objective:** Run the pipeline and observe/understand the DevOps process!

**Instructions:**

- The GitLab server is reachable with the name 'gitlab'
- Gitlab credentials:

| Username | Password |
|---|---|
| root | welcome123 |

- The Jenkins server is reachable with the name 'jenkins'
- Jenkins credentials:

| Username | Password |
|---|---|
| admin | welcome123 |

- The test deployment server is reachable by the name "test-server"

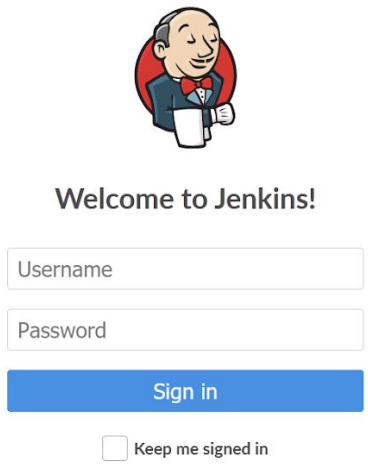
## Lab Setup

On starting the lab, the following interface will be accessible to the user.

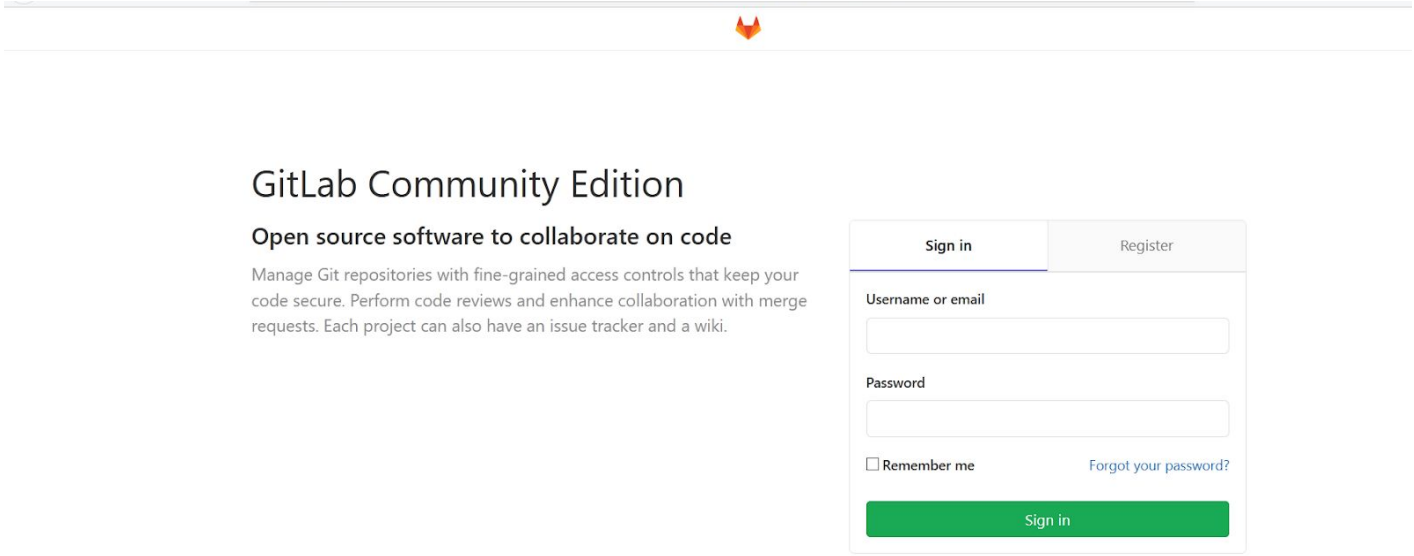

On choosing the first panel, **KALI CLI** will open in a new tab

```
root@kali-cli:~#
```

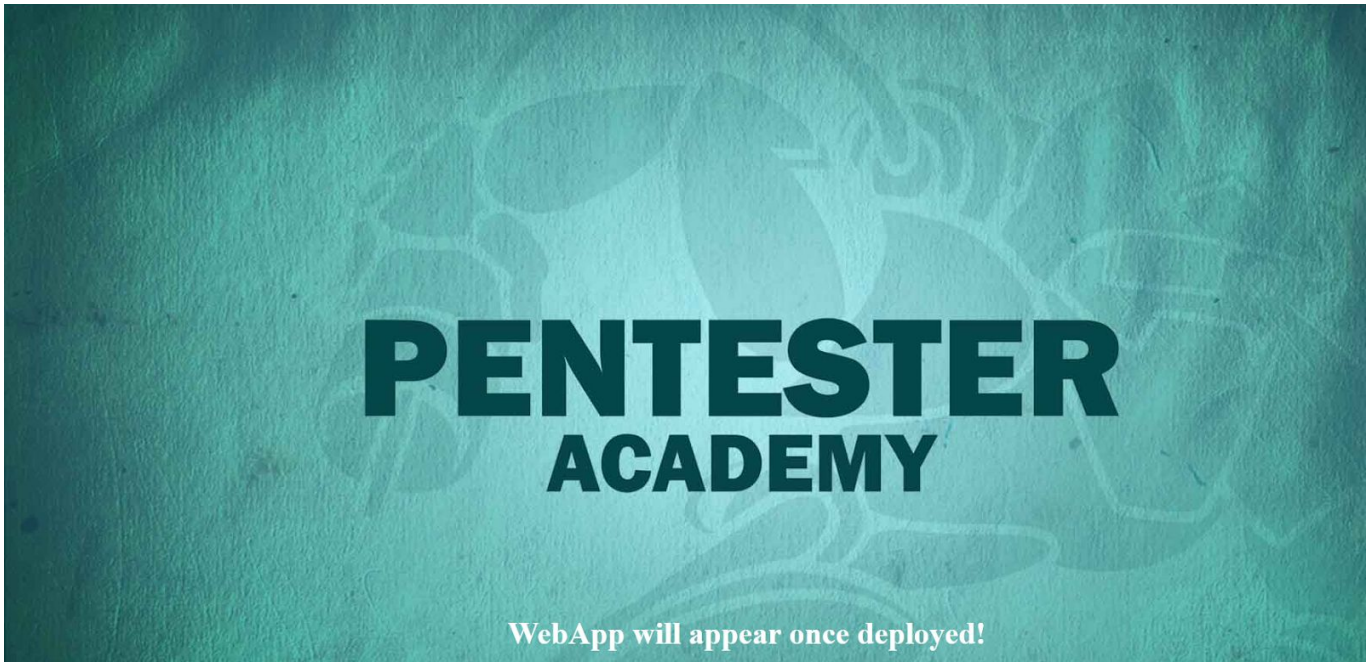
On selecting the second panel, **Jenkins** will open in a new tab




On selecting the middle panel, a web UI of **Gitlab** will open in a new tab.




And on selecting the right panel, a web UI of **Test Server** will open in a new tab.



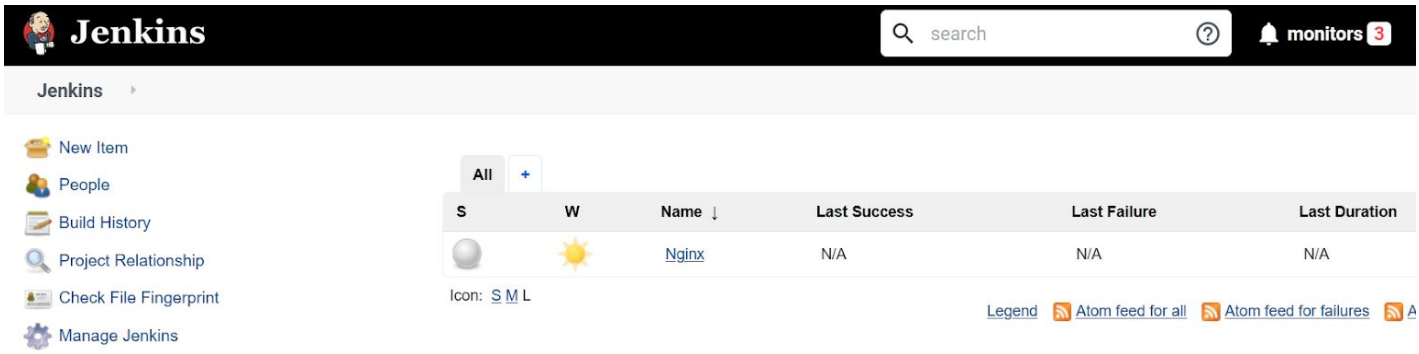

The page will reload until the test-server has started running the web service at port 80

## Solution

**Step 1:** Login into the Jenkins, The credentials are provided in the challenge description.

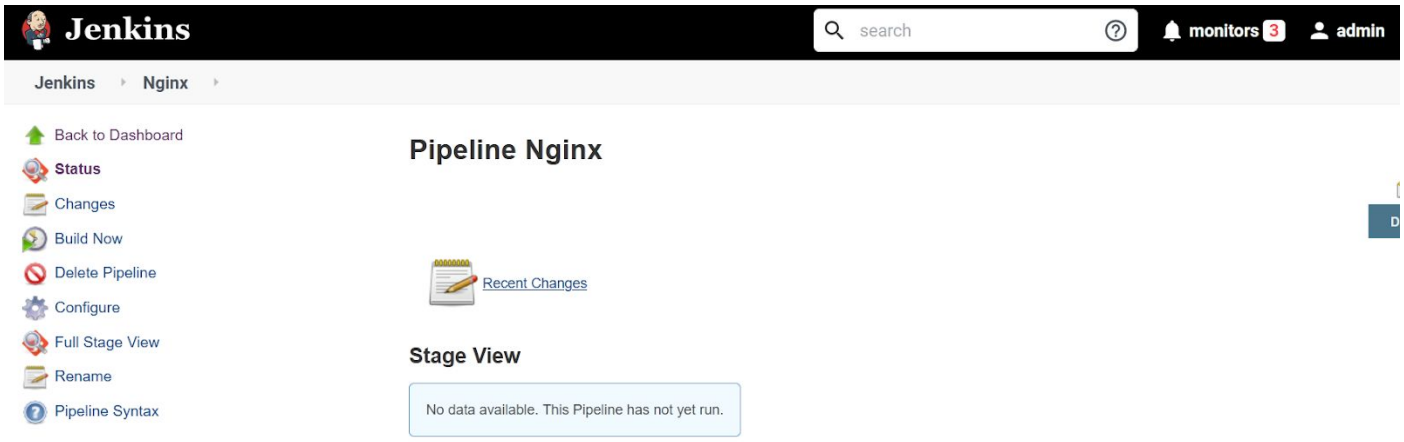
**Credentials:**

- **Username:** admin
- **Password:** welcome123



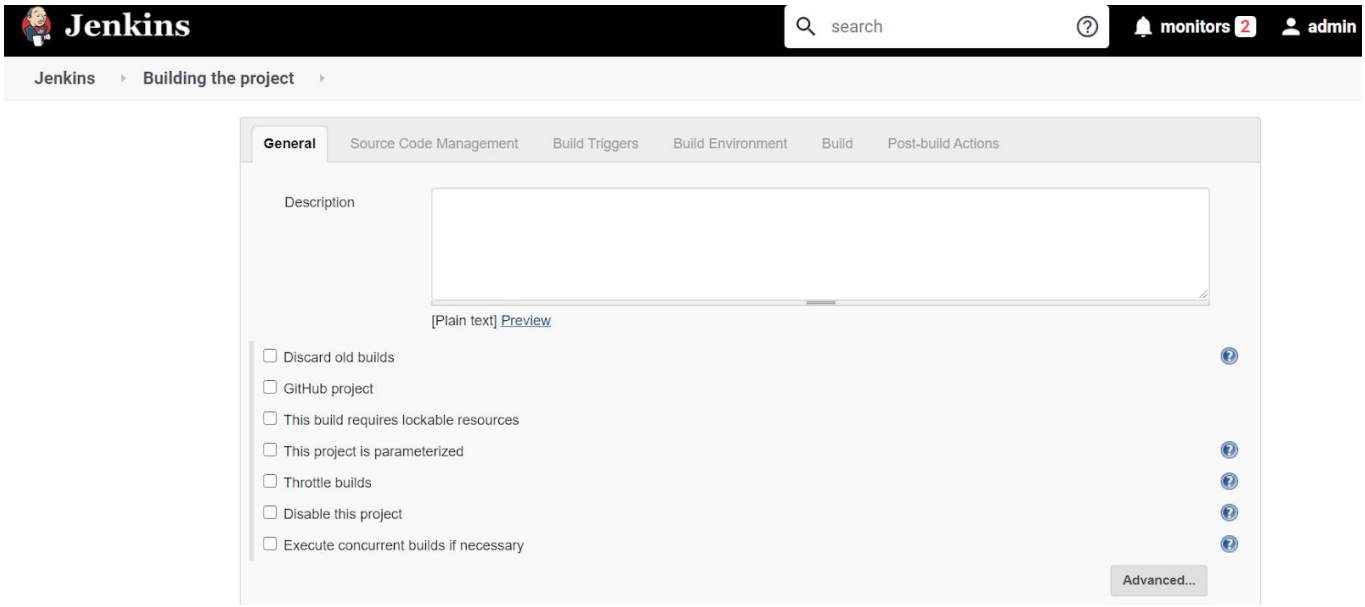

There is only one Job (Nginx) available in the Jenkins instance.

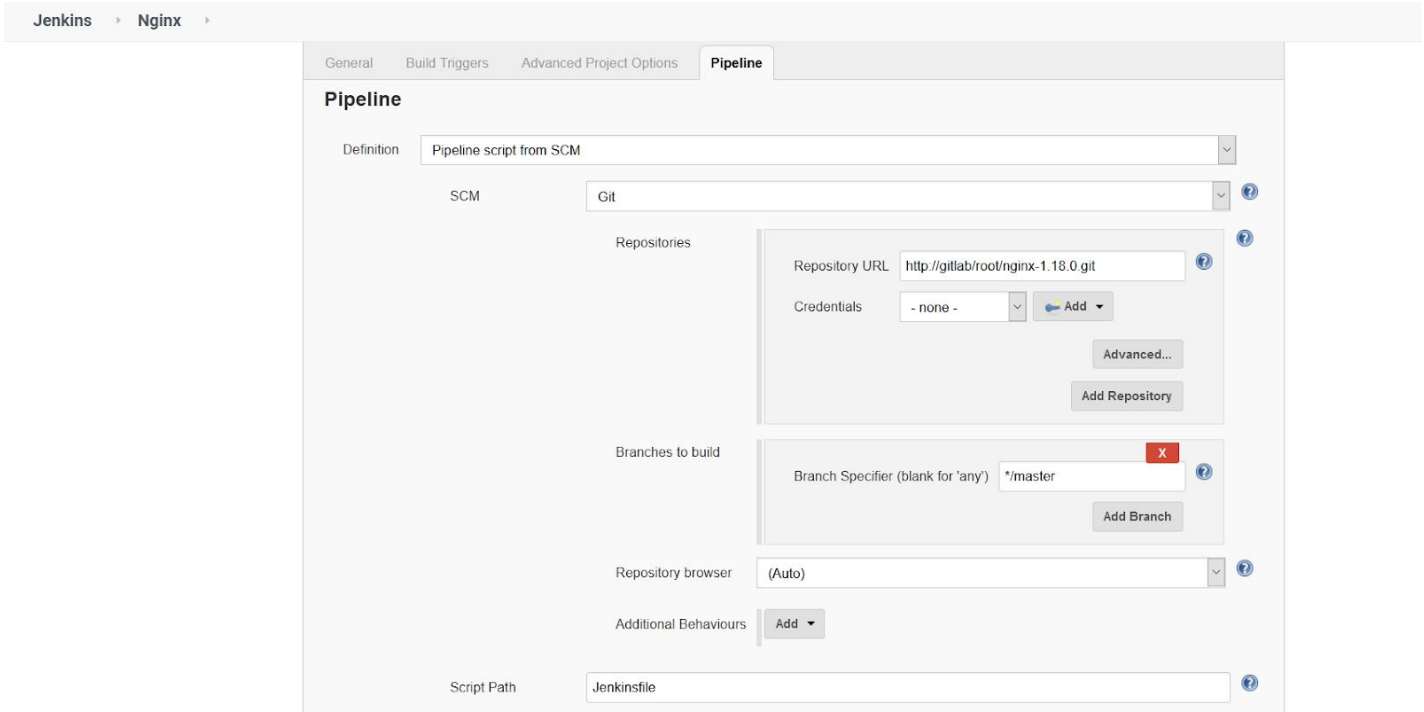
**Step 2:** Click on the “Nginx” job.



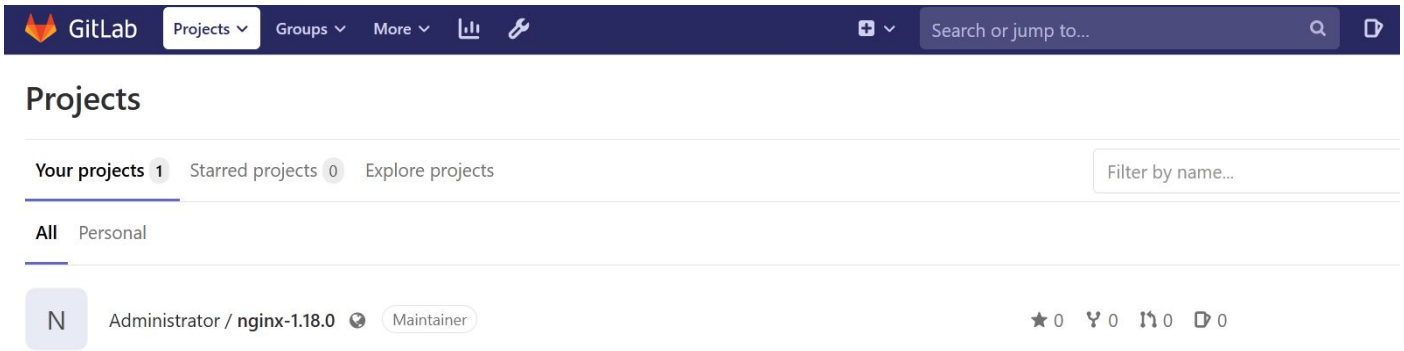
This page is for “Pipeline Nginx” job, The Pipeline is appended in front of the Job name because this is a “Pipeline” type job in which it accepts a ‘Jenkinsfile’ which has all the commands and configuration of the pipeline.

**Step 3:** Click on the “Configure” option to check the configuration of the Job.

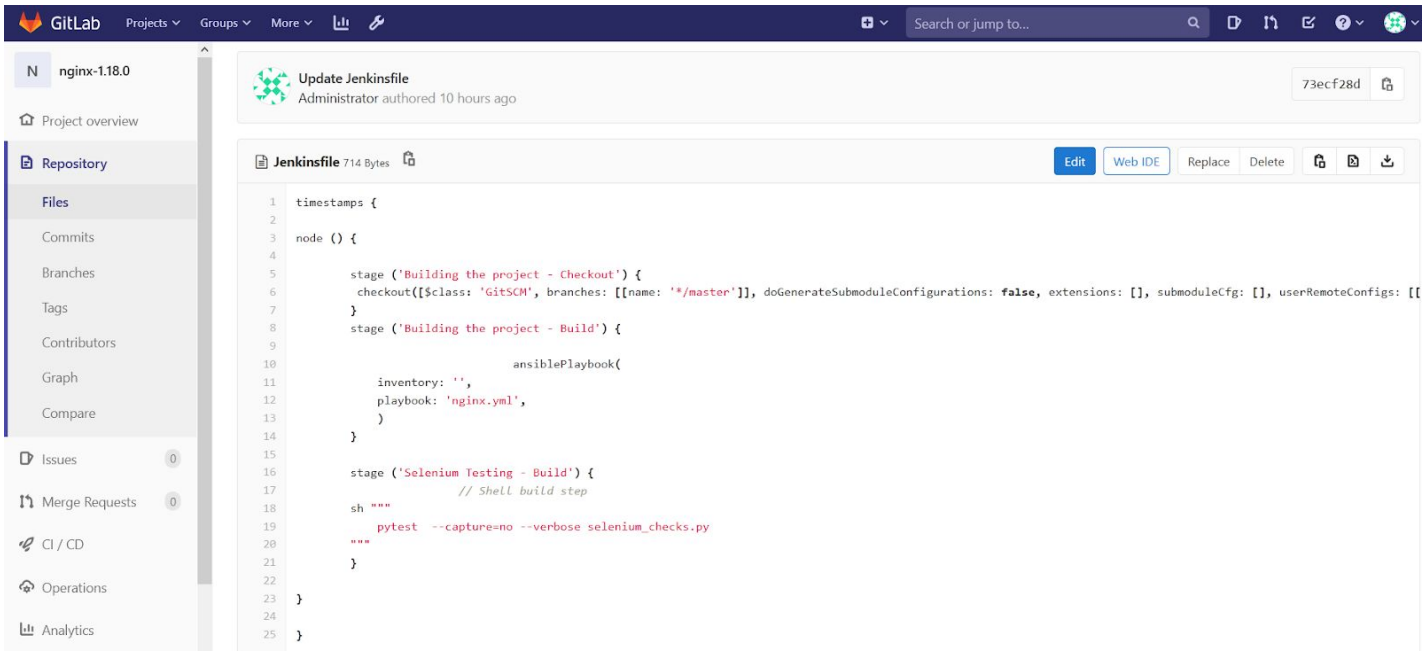


The “Pipeline” sections accept Jenkinsfile directly or a source such as Gitlab where the code and Jenkinsfile are stored for the project. The code is hosted on GitLab instance at this path “http://gitlab/root/nginx-1.18.0.git”




**Step 3:** Open the project on Gitlab and check the Jenkinsfile to build the pipeline.



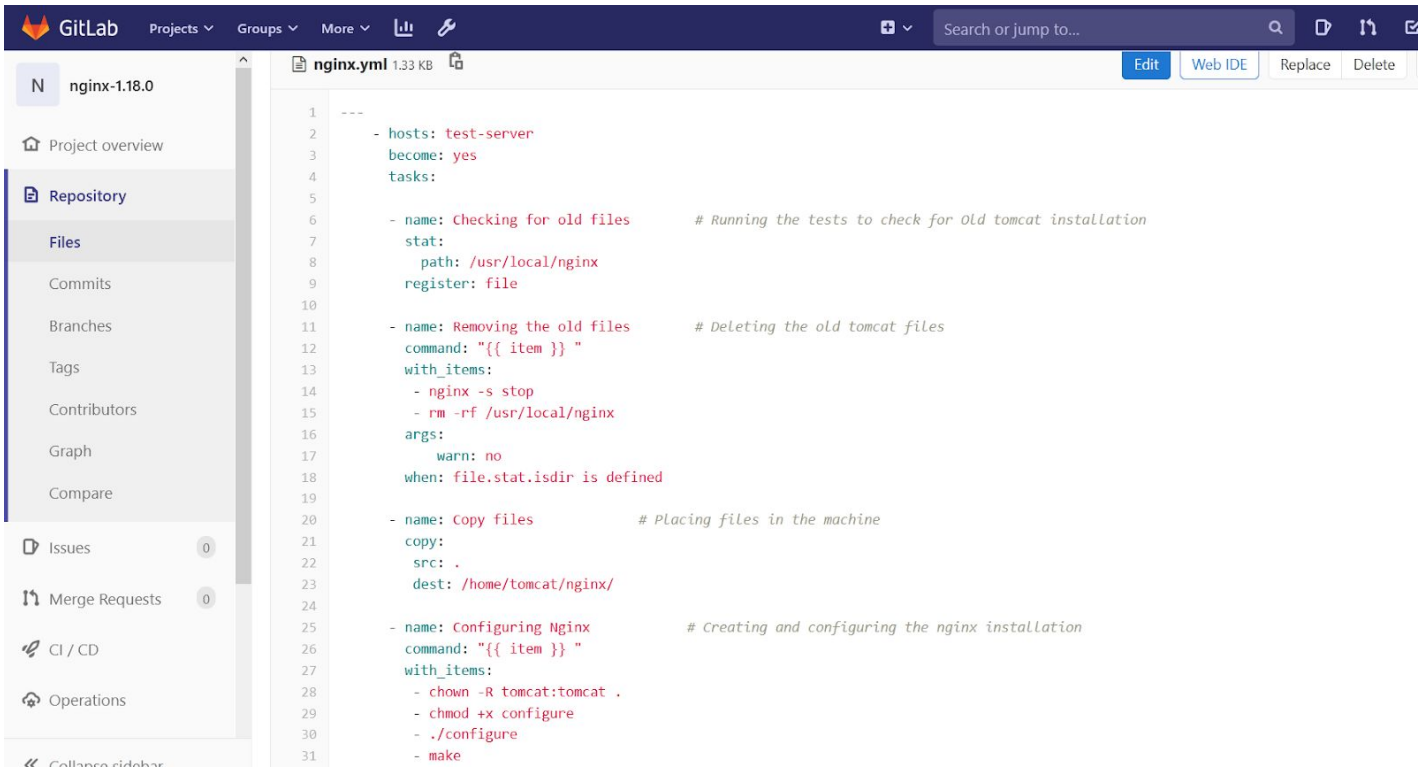

The file includes ‘stages’ that are a collection of steps, Each step performs a function which is explained below:-

**Jenkinsfile Stages:**

- **Building the project - Checkout:** In this stage, the git repository will be checked for any updates or commits. If commits are found in the repository then the new files will be fetched from remote repository.
- **Building the project - Build:** In this stage, the ansible will initiate the installation of Nginx on the remote server (test-server).
- **Selenium Testing - Build:** In this stage, the Jenkins will start checks on the newly deployed server to verify if the installation was successful or not.

**Note:** The code for ansible (nginx.yml) and selenium testing (selenium_checks.py) are stored in the Gitlab repository itself.

**Step 4:** Check the Ansible configuration of the Nginx installation. The nginx.yml can be found in the root directory of the project.

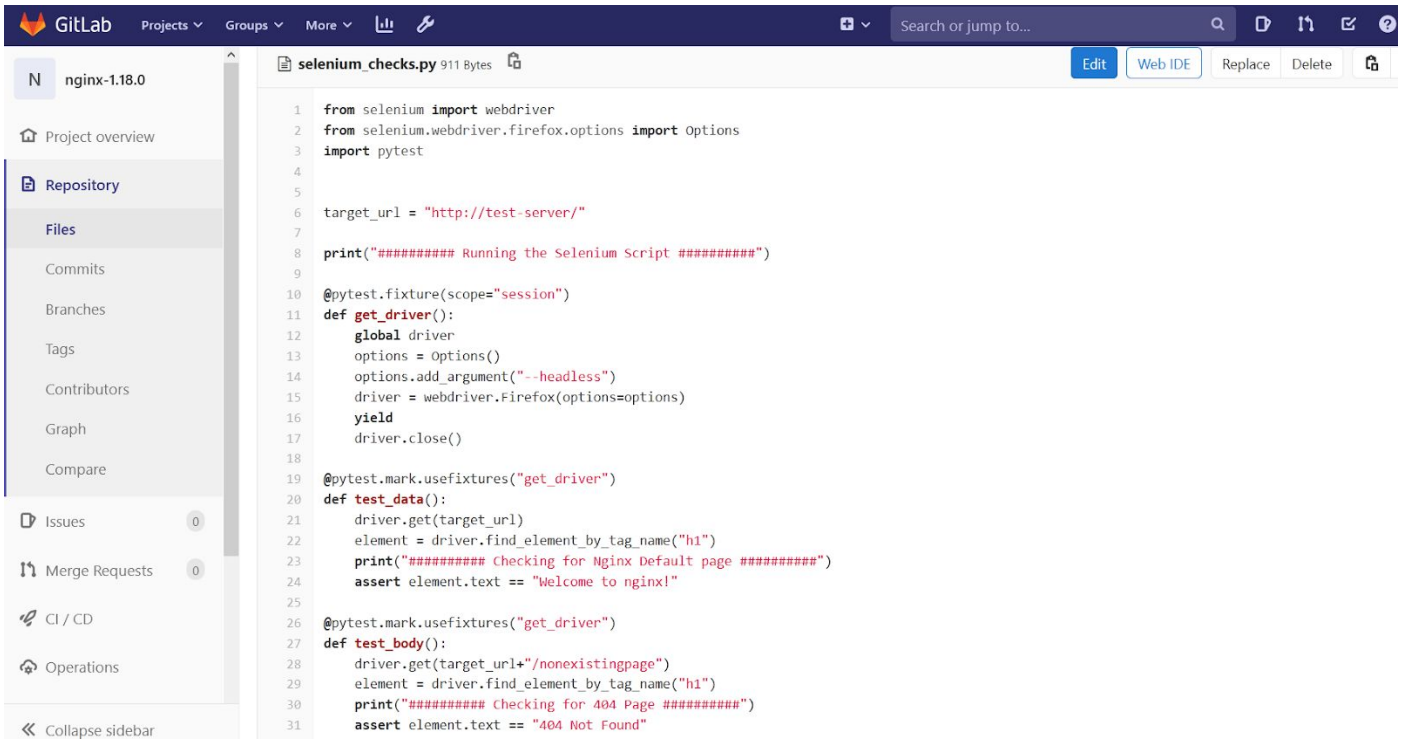



In the Ansible configuration provided above performs the following tasks:

**Tasks:**

- Check for the old Nginx installation on the server
- Remove the old installation from test-server
- Copy the new files into the server
- Install the Nginx into the remote server (test-server)
- Start the Nginx server.

**Step 5:** Check the Selenium tests. The tests will be used to determine if the Nginx is successfully installed or not.




**Tasks:**

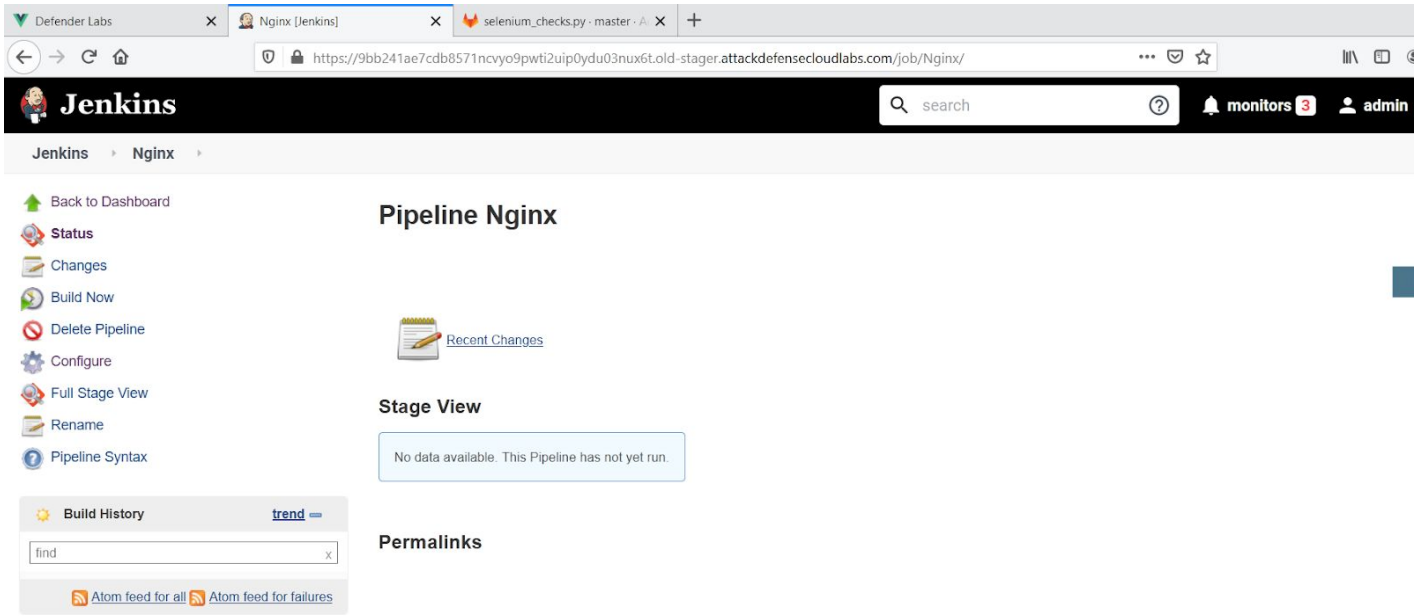
- Open the index file and find “welcome to nginx!” in the source code.
- Open any non-existing page (i.e nonexistingpage) and find the string “404 not Found” which comes in the Nginx errors when the file does not exist.

.

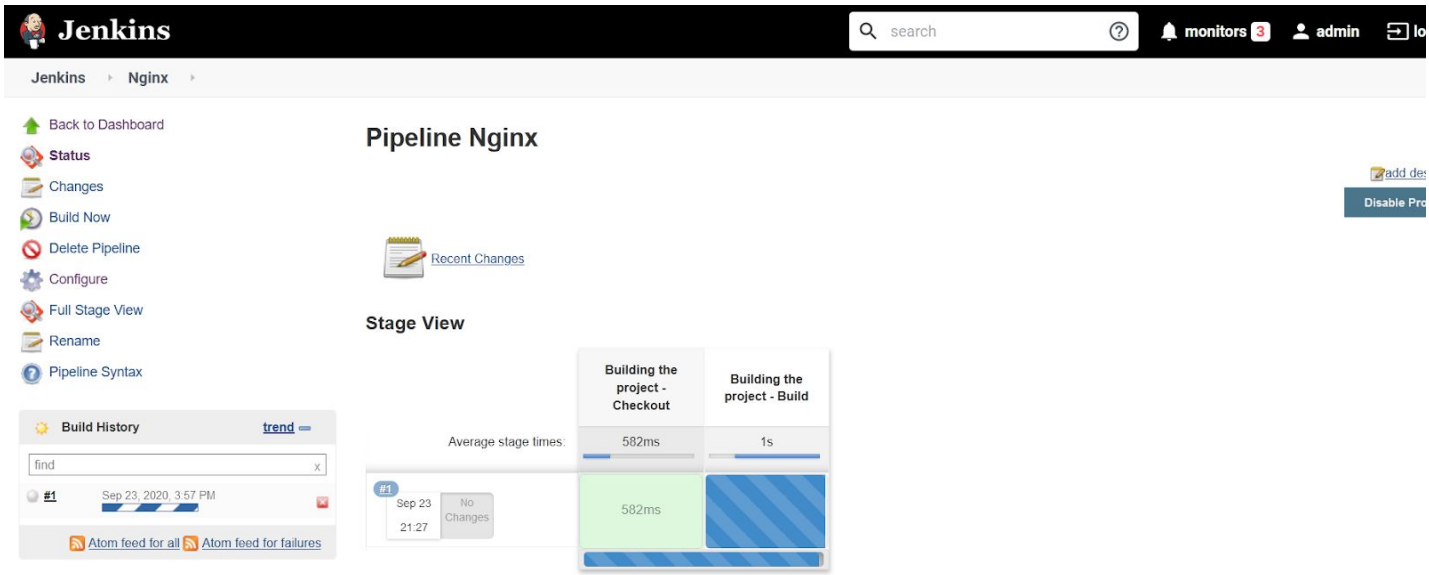
Performing these tests on the target machine will ensure the Server is up and running on the test-server

**Pipeline Execution**

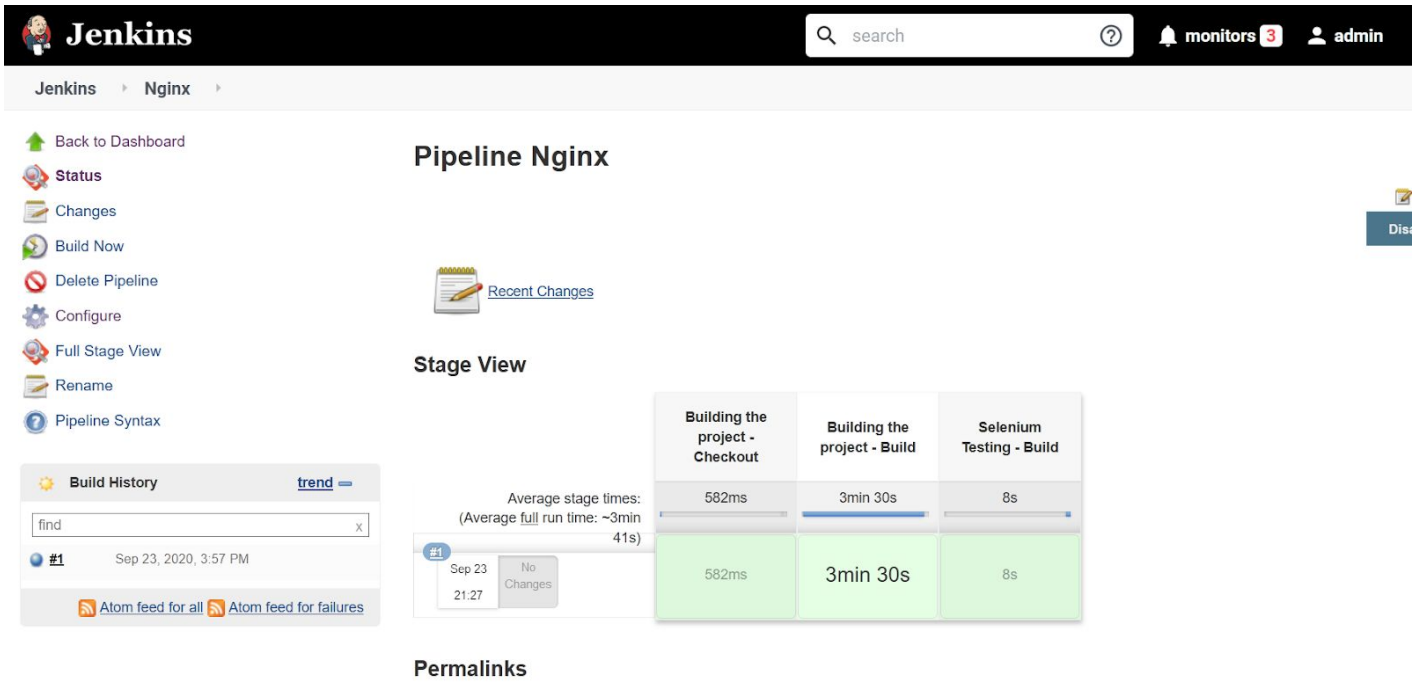
**Step 1:** Navigate to the Pipeline tab.




**Step 2:** Click on the "Build Now" button to start the Pipeline.

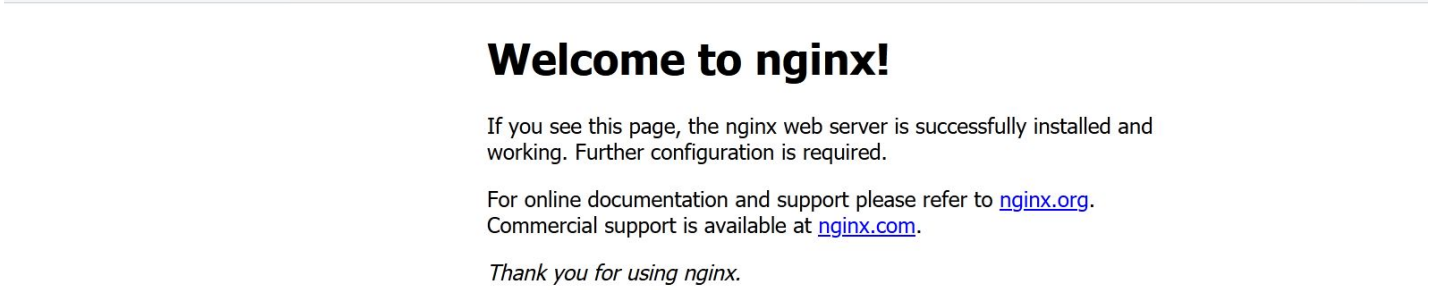



The page will automatically update and show the latest build information about the test-server.




The pipeline completed the execution successfully.

**Step 3:** Check the Test server.




The default page of Nginx is displayed on the test-server which means the installation of Nginx was successful.

## Learning

Working of a simple DevOps pipeline consisting of different components.