# ATTACK DEFENSE

by PentesterAcademy

| | |
|---|---|
| **Name** | DevOps Pipeline: Django WebApp Pipeline as Code |
| **URL** | https://www.attackdefense.com/challengedetails?cid=2068 |
| **Type** | Pipeline Basics: Web Applications |

**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 a sample Django-based web application. The pipeline is stored in form of a Jenkinsfile (Pipeline as Code) and 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 all parts: building Django project, deploying with Ansible, and dynamic testing with Selenium)
- Test server (For test deployment)

**Objective:** Run the pipeline project 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:

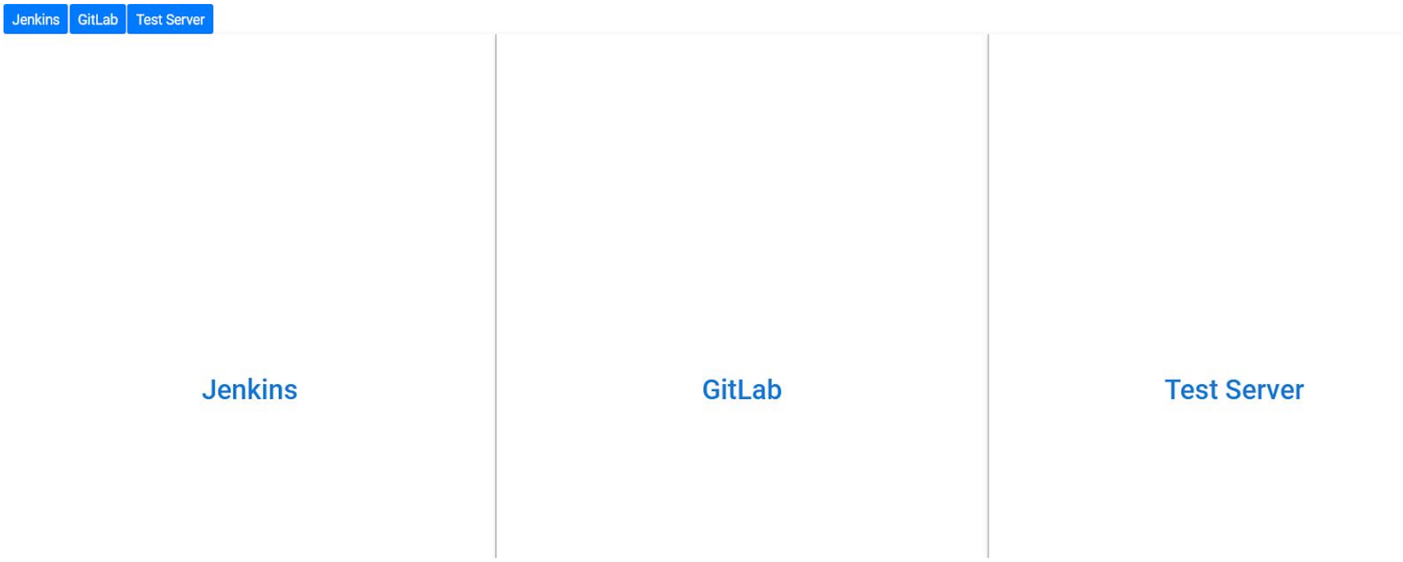
| Username | Password |
|---|---|
| admin | welcome123 |

- The test deployment server is reachable by the name "test-server"
- Test server SSH credentials:

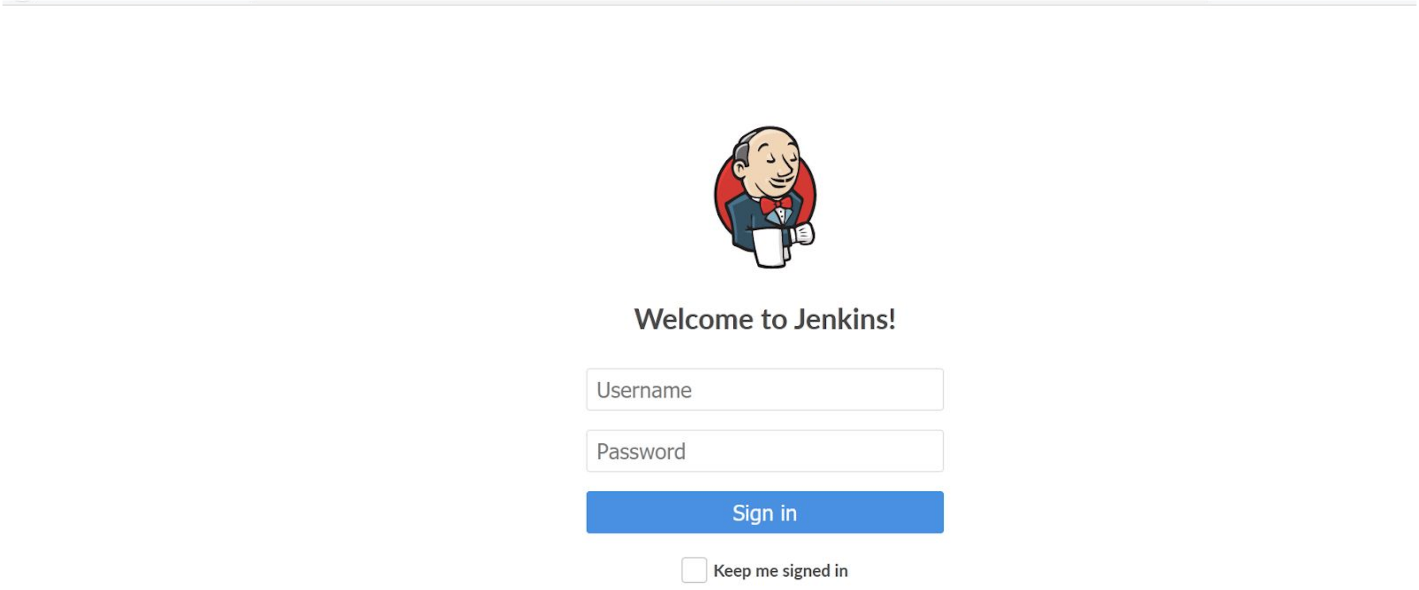
| Username | Password |
|---|---|
| tomcat | password1 |

## Lab Setup

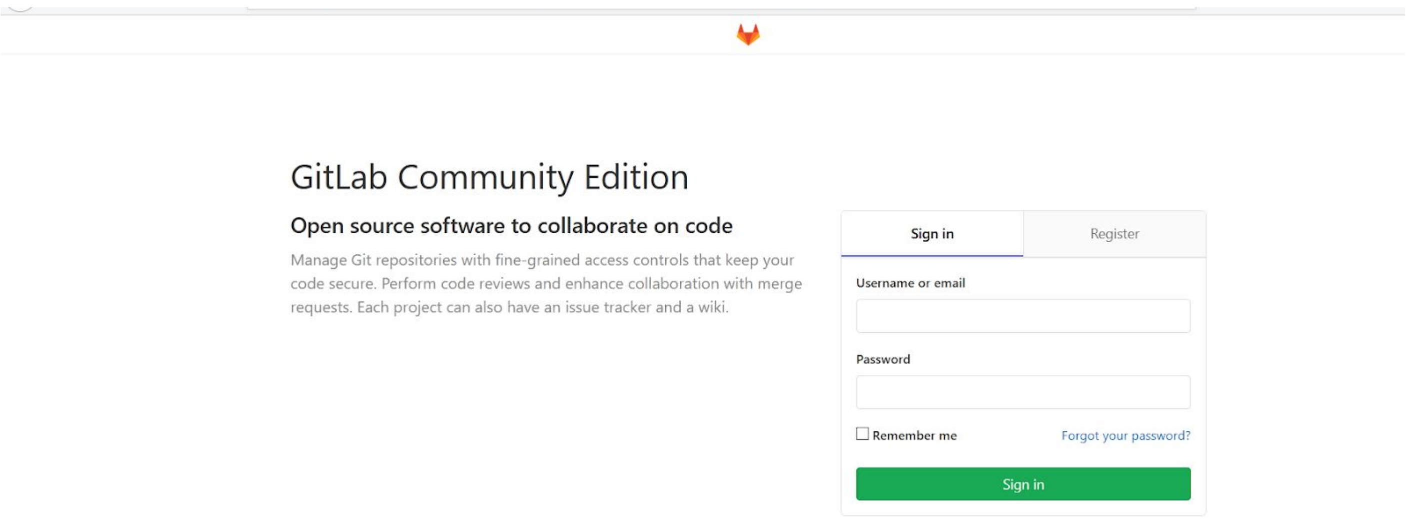
On starting the lab, the following interface will be accessible to the user.



On choosing (clicking the text in the center) left panel, **Jenkins** will open in a new tab


Welcome to Jenkins!

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



GitLab Community Edition

Open source software to collaborate on code

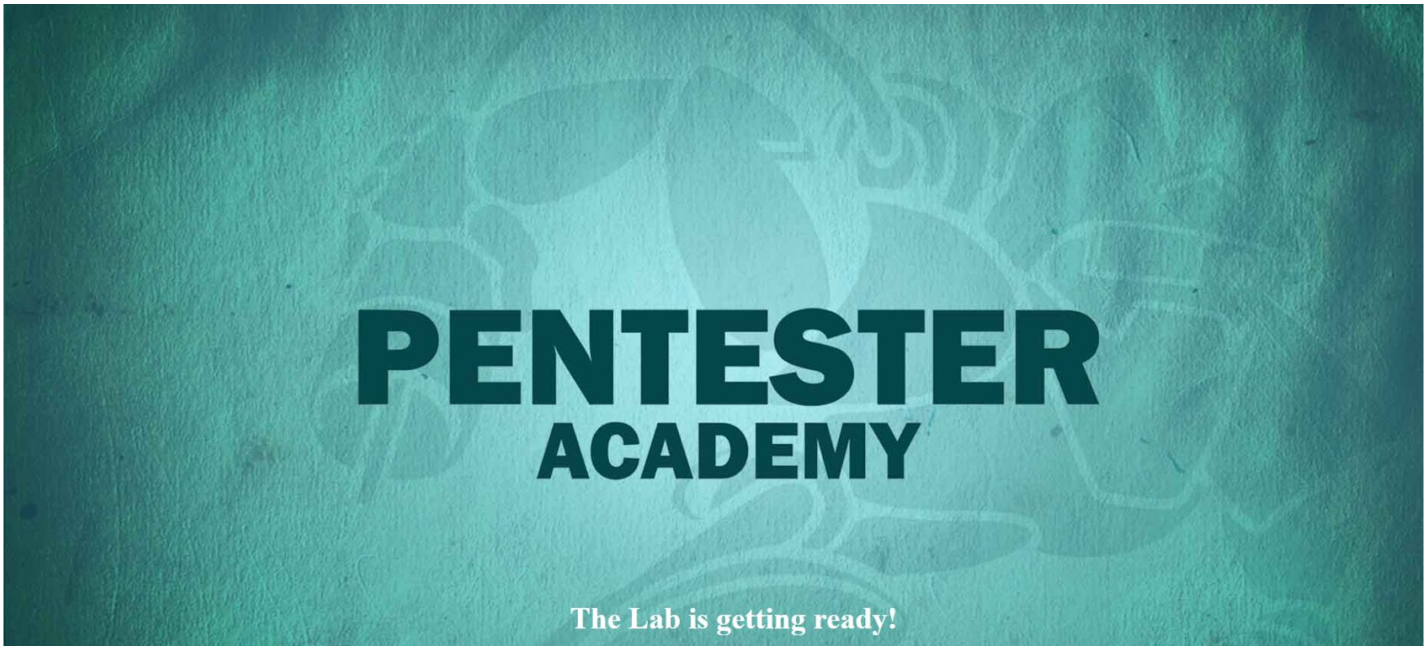
Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Username or email

Password

☐ Remember me

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



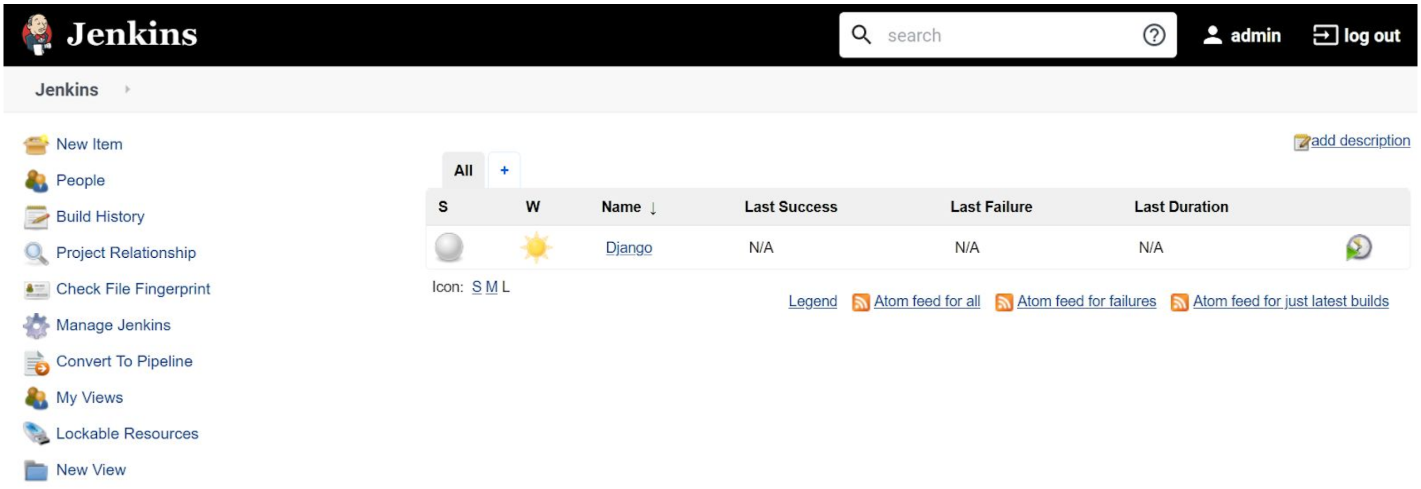
The page will reload until the test-server has started running the web service at port 8080

## Solution

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

**Credentials:**

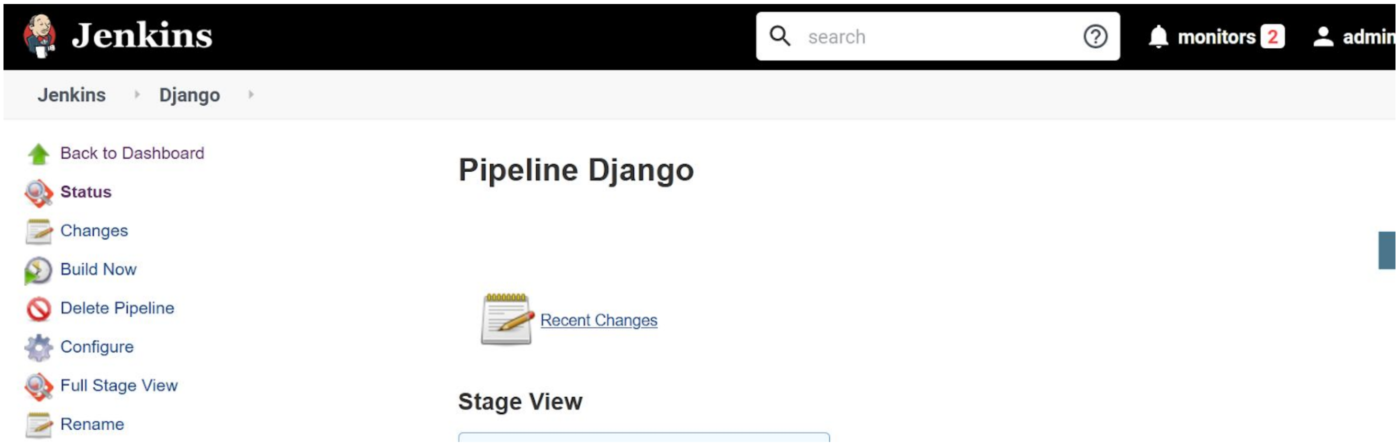
- **Username:** admin
- **Password:** welcome123



| S | W | Name ↓ | Last Success | Last Failure | Last Duration |
|---|---|---|---|---|---|
| ○ | ☀ | Django | N/A | N/A | N/A |

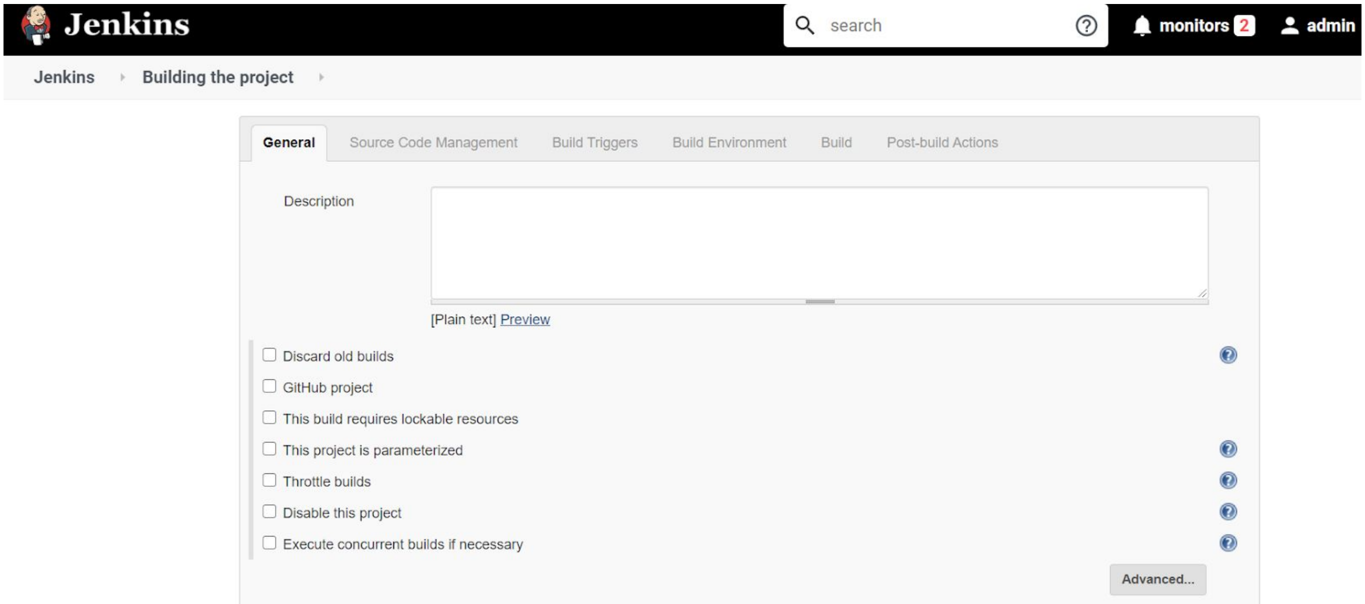
There is only one job present in the Jenkins Interface, We will take one job at a time to study.

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



This page is for “Pipeline Django” 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 2:** Click on the “Configure” option to check the configuration of the Job.



Description

[Plain text] Preview

□ Discard old builds

□ GitHub project

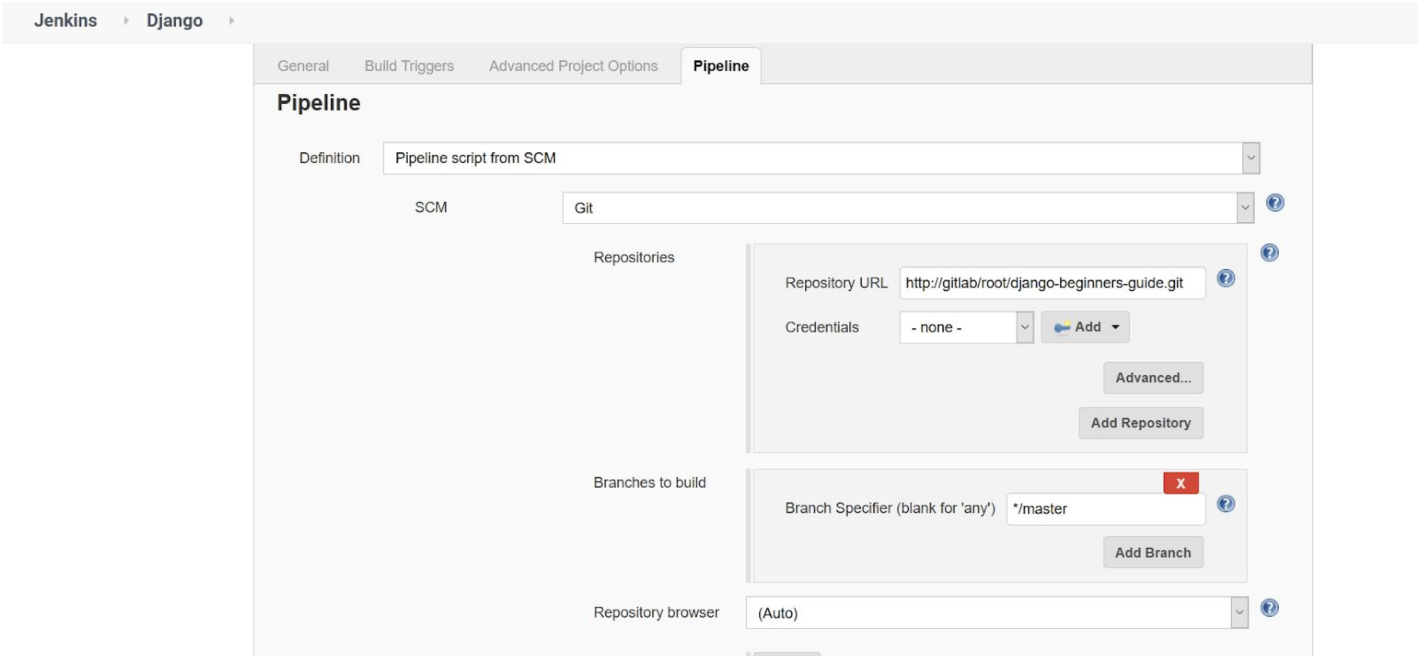
□ This build requires lockable resources

□ This project is parameterized

□ Throttle builds

□ Disable this project

□ Execute concurrent builds if necessary



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: http://gitlab/root/django-beginners-guide.git

Credentials: - none -

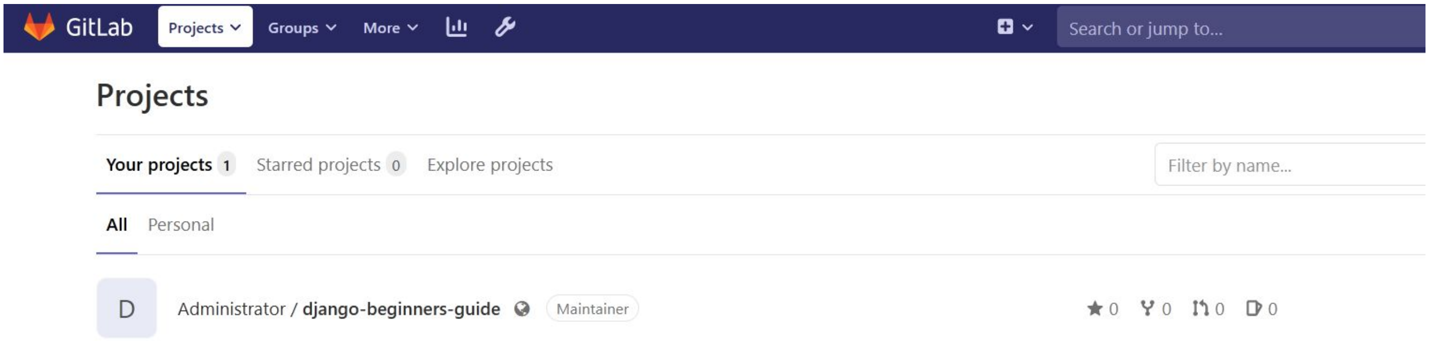
Branches to build

Branch Specifier (blank for 'any'): */master

Repository browser: (Auto)

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/django-beginners-guide.git”



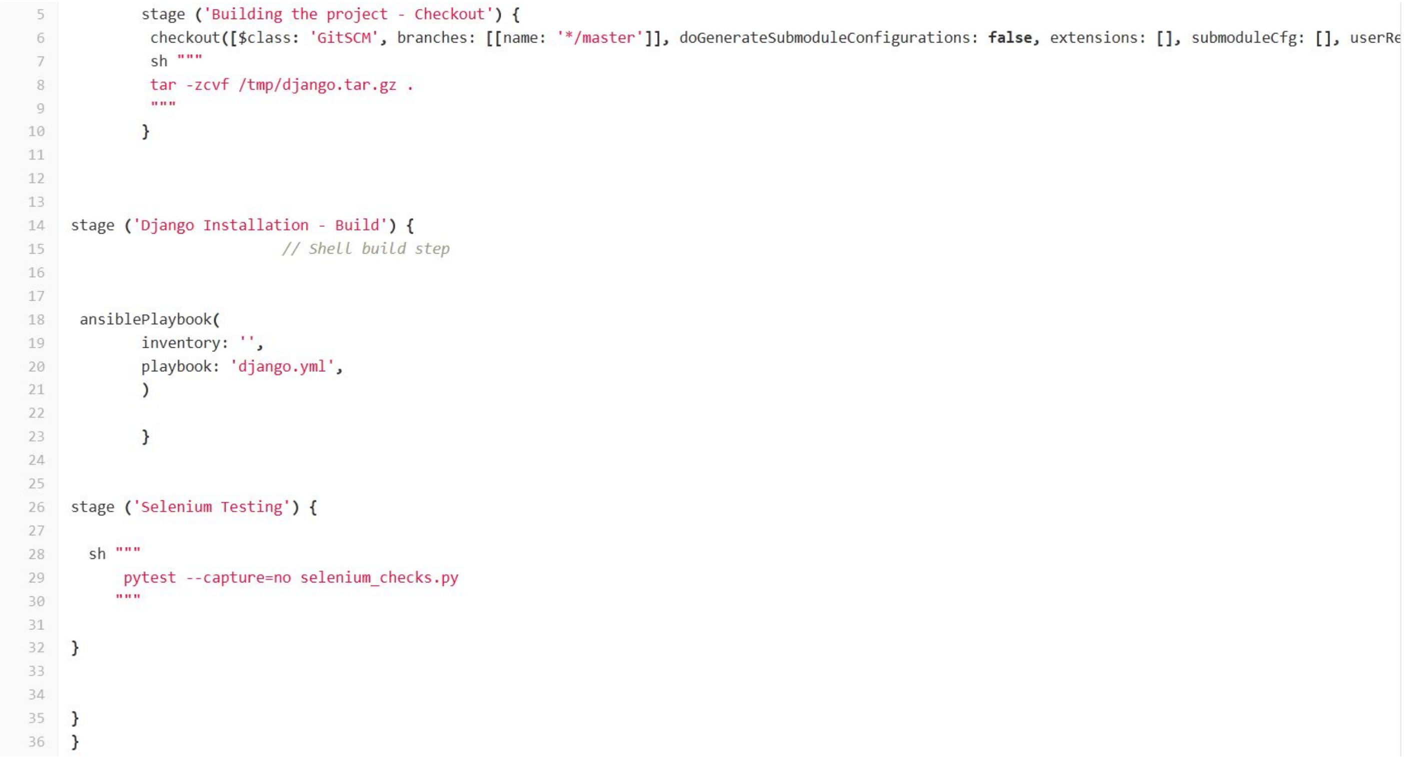
Projects

Your projects 1 | Starred projects 0 | Explore projects | Filter by name...

All | Personal

D Administrator / django-beginners-guide Maintainer ★ 0 0 0 0

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



```
        stage ('Building the project - Checkout') {
         checkout([$class: 'GitSCM', branches: [[name: '*/master']], doGenerateSubmoduleConfigurations: false, extensions: [], submoduleCfg: [], userRe
         sh """
         tar -zcvf /tmp/django.tar.gz .
         """
        }



stage ('Django Installation - Build') {
                        // Shell build step


 ansiblePlaybook(
        inventory: '',
        playbook: 'django.yml',
        )

        }


stage ('Selenium Testing') {

  sh """
      pytest --capture=no selenium_checks.py
     """

}


}
}
```

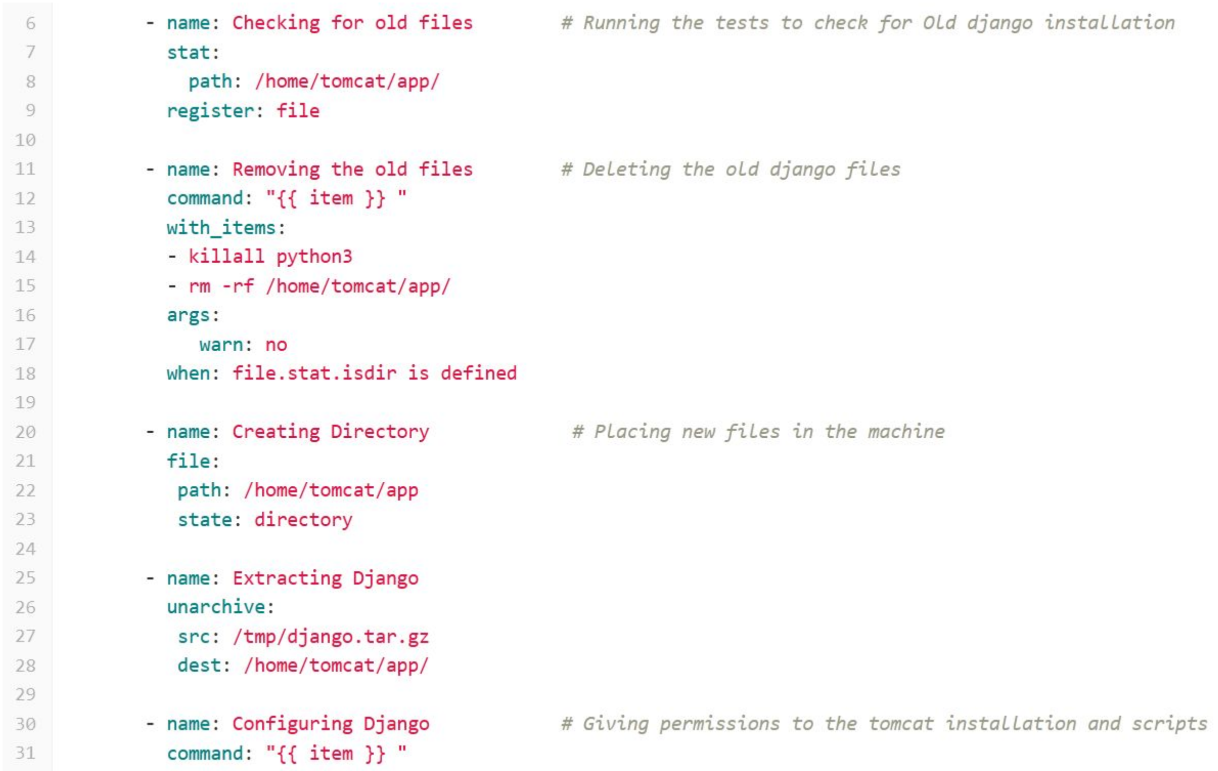
The file includes 'stages' that are a collection of steps, Each step performs a function which is explained below:-

**Jenkinsfile Stages:**

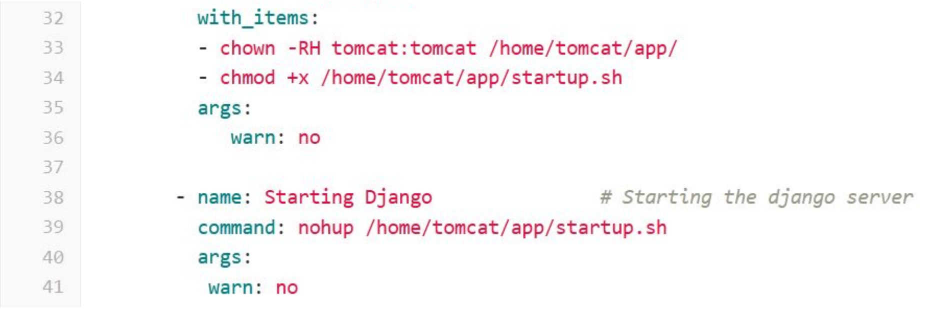
- **Building the project - Checkout:** In this stage, the git repository will be checked for any updates or commits. If commits are found in the repository then the new files will be fetched from the remote repository.
- **Django Installation - Build:** In this stage, the ansible will initiate the installation of Django 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 (django.yml) and selenium testing (selenium_checks.py) are stored in the Gitlab repository itself.

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



```
        - name: Checking for old files        # Running the tests to check for Old django installation
          stat:
            path: /home/tomcat/app/
          register: file

        - name: Removing the old files        # Deleting the old django files
          command: "{{ item }} "
          with_items:
          - killall python3
          - rm -rf /home/tomcat/app/
          args:
             warn: no
          when: file.stat.isdir is defined

        - name: Creating Directory             # Placing new files in the machine
          file:
           path: /home/tomcat/app
           state: directory

        - name: Extracting Django
          unarchive:
           src: /tmp/django.tar.gz
           dest: /home/tomcat/app/

        - name: Configuring Django            # Giving permissions to the tomcat installation and scripts
          command: "{{ item }} "
```



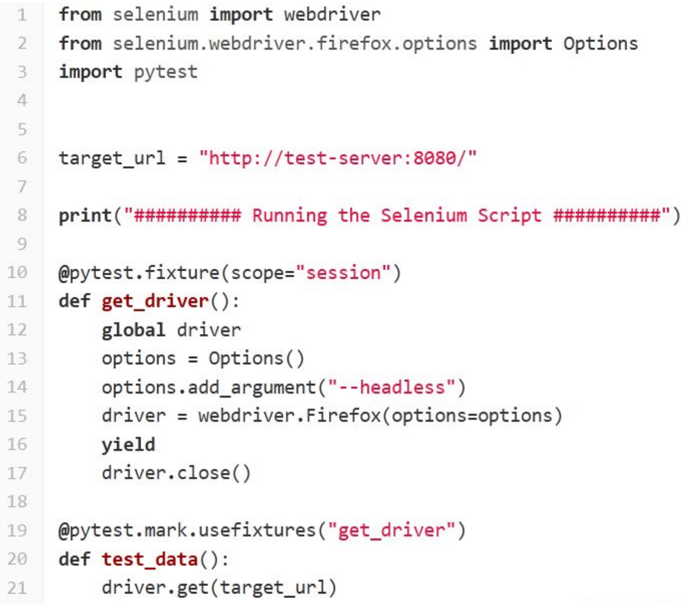
```
          with_items:
          - chown -RH tomcat:tomcat /home/tomcat/app/
          - chmod +x /home/tomcat/app/startup.sh
          args:
             warn: no

        - name: Starting Django               # Starting the django server
          command: nohup /home/tomcat/app/startup.sh
          args:
           warn: no
```

In the Ansible configuration provided above performs the following tasks:

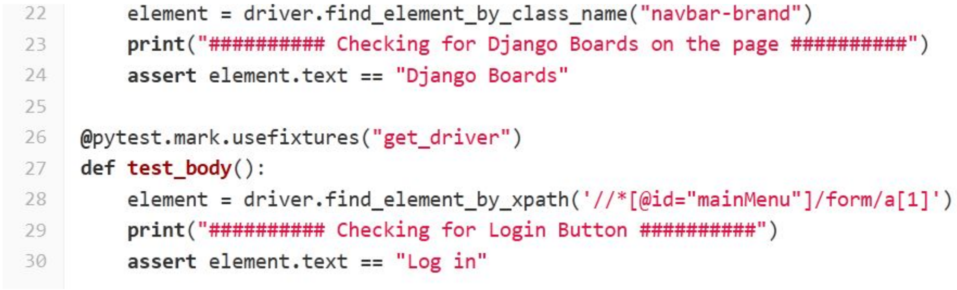
**Tasks:**

- Check for the old Django installation on the server
- Remove the old installation from test-server
- Creating the required directories
- Extracting the Django files
- Assigning the permissions to Django directory
- Starting the Django server

**Step 5:** Check the Selenium tests. The tests will be used to determine if the Django website is successfully up and running or not.



```python
from selenium import webdriver
from selenium.webdriver.firefox.options import Options
import pytest


target_url = "http://test-server:8080/"

print("########## Running the Selenium Script ##########")

@pytest.fixture(scope="session")
def get_driver():
    global driver
    options = Options()
    options.add_argument("--headless")
    driver = webdriver.Firefox(options=options)
    yield
    driver.close()

@pytest.mark.usefixtures("get_driver")
def test_data():
    driver.get(target_url)
```



```
    element = driver.find_element_by_class_name("navbar-brand")
    print("########## Checking for Django Boards on the page ##########")
    assert element.text == "Django Boards"

@pytest.mark.usefixtures("get_driver")
def test_body():
    element = driver.find_element_by_xpath('//*[@id="mainMenu"]/form/a[1]')
    print("########## Checking for Login Button ##########")
    assert element.text == "Log in"
```

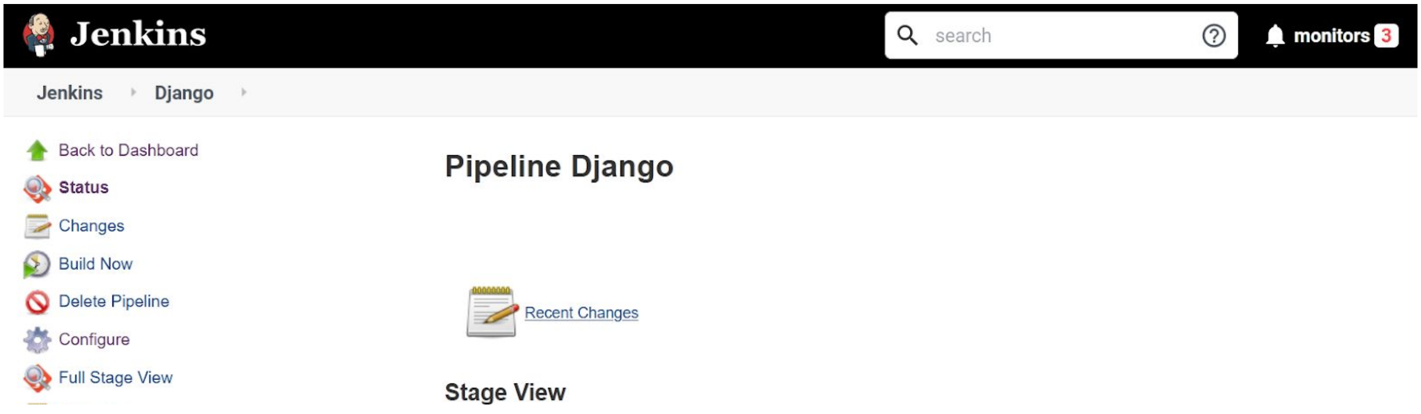
**Tasks:**

- Open the index page and find “Django Boards” in the source code.
- Check for the Login button in the page

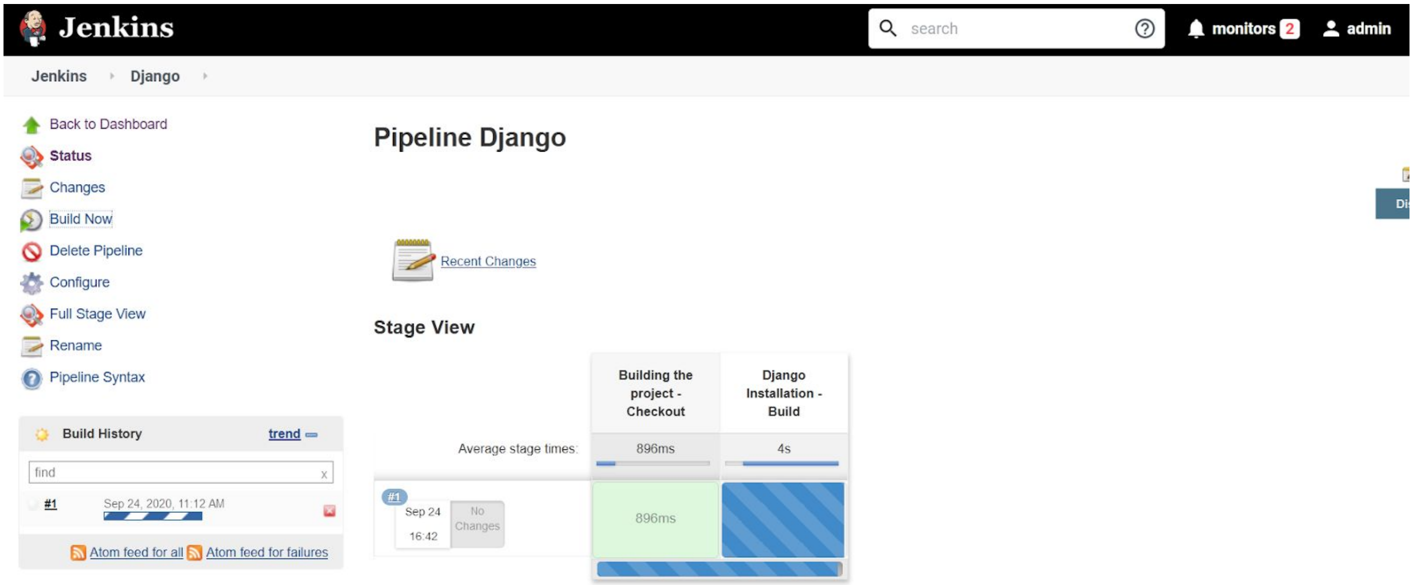
Performing these tests on the target machine will ensure the website is up and running on the test-server

**Pipeline Execution**

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



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



Jenkins  search  monitors 2  admin

Jenkins › Django ›

- Back to Dashboard
- Status
- Changes
- Build Now
- Delete Pipeline
- Configure
- Full Stage View
- Rename
- Pipeline Syntax

Build History  trend

find

#1  Sep 24, 2020, 11:12 AM

Atom feed for all  Atom feed for failures

## Pipeline Django

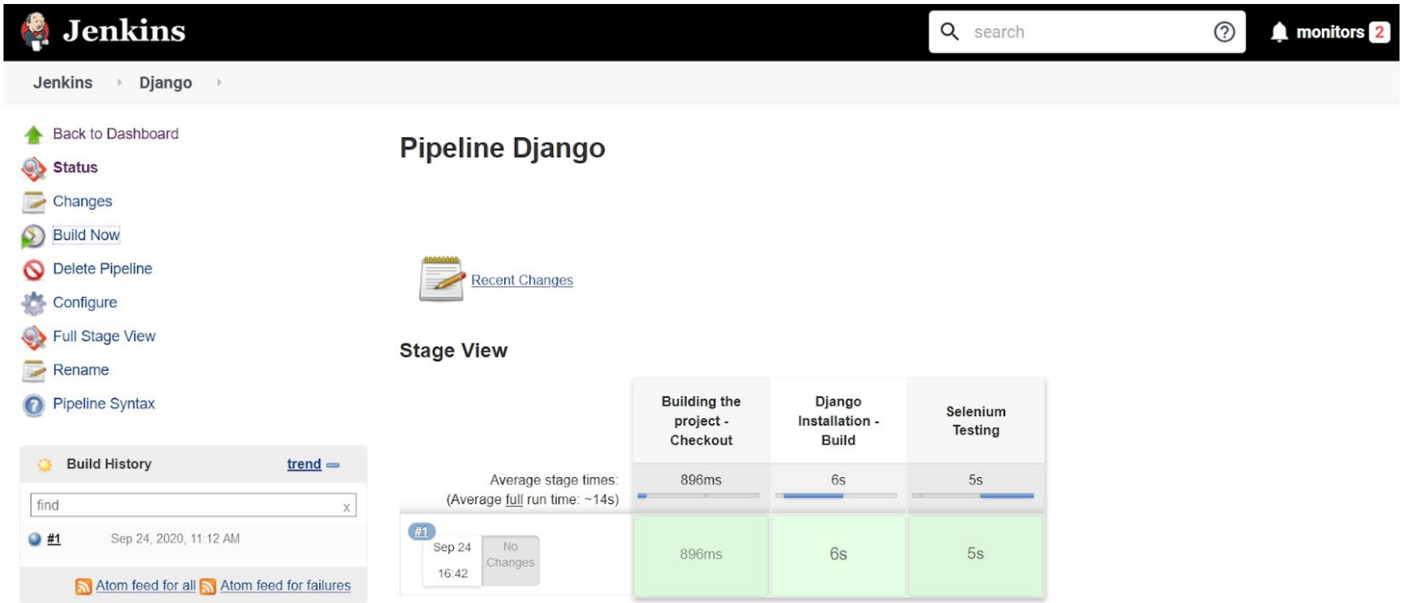
Recent Changes

### Stage View

| | Building the project - Checkout | Django Installation - Build |
|---|---|---|
| Average stage times: | 896ms | 4s |
| #1 Sep 24 16:42 No Changes | 896ms | |

### Permalinks

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



Jenkins  search  monitors 2

Jenkins › Django ›

- Back to Dashboard
- Status
- Changes
- Build Now
- Delete Pipeline
- Configure
- Full Stage View
- Rename
- Pipeline Syntax

Build History  trend

find

#1  Sep 24, 2020, 11:12 AM

Atom feed for all  Atom feed for failures

## Pipeline Django

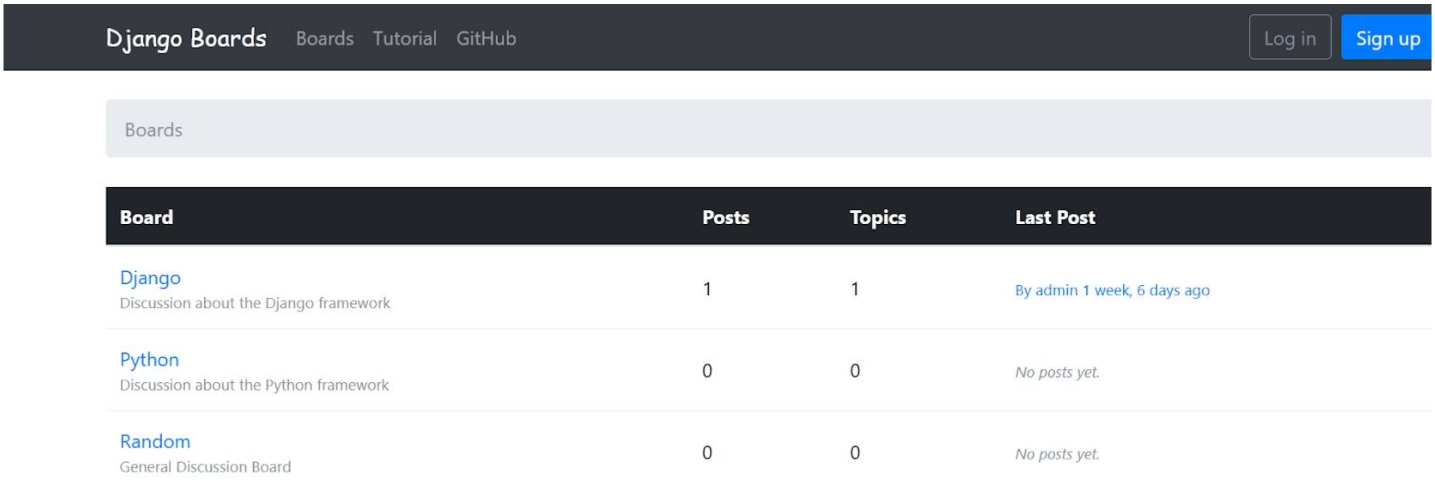
Recent Changes

### Stage View

| | Building the project - Checkout | Django Installation - Build | Selenium Testing |
|---|---|---|---|
| Average stage times:<br>(Average full run time: ~14s) | 896ms | 6s | 5s |
| #1 Sep 24 16:42 No Changes | 896ms | 6s | 5s |

The pipeline completed the execution successfully.

**Step 3:** Navigate to the deployed website.



| Board | Posts | Topics | Last Post |
|---|---|---|---|
| Django<br>Discussion about the Django framework | 1 | 1 | By admin 1 week, 6 days ago |
| Python<br>Discussion about the Python framework | 0 | 0 | *No posts yet.* |
| Random<br>General Discussion Board | 0 | 0 | *No posts yet.* |

The Django website is successfully up and running on the target-server.

## Learning

Working of a simple DevOps pipeline consisting of different components.