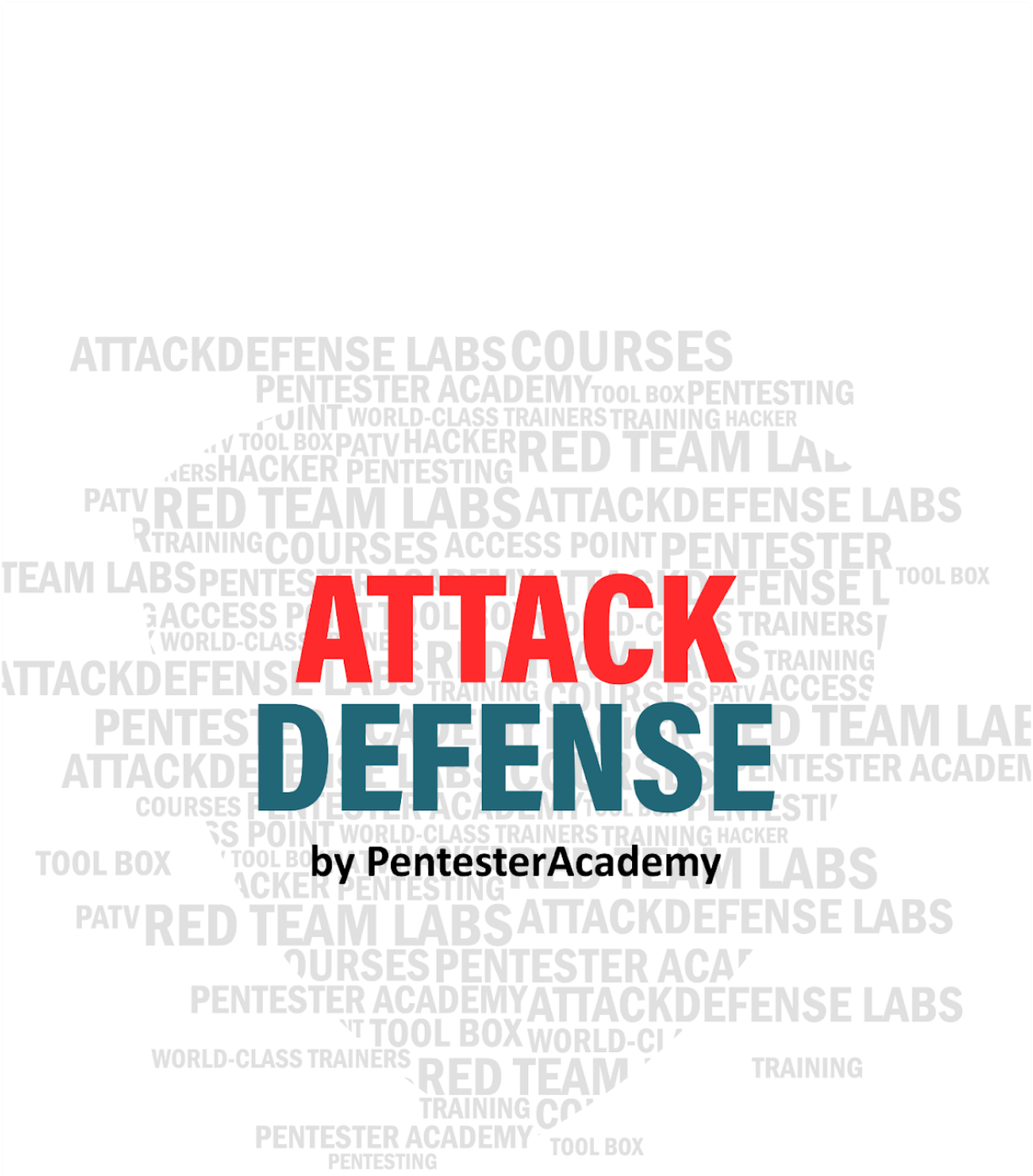

ATTACK
DEFENSE
by PentesterAcademy

Name	Django WebApp
URL	https://www.attackdefense.com/challengedetails?cid=2039
Type	DevOps Basics: Continuous Integration

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

The continuous Integration development process dictates that developers push the code into the master development repository frequently and each (or a small number of) code pushes can trigger the automated build, tests, and deploy the latest build on the test server.

Jenkins is an open-source automation server that is used widely for continuous integration.

A Jenkins instance and a Gitlab instance are provided to the user. The source code of a Django Webapp is stored on the Gitlab instance.

Objective: Create a Jenkins job to migrate and test the web application!

Instructions:

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

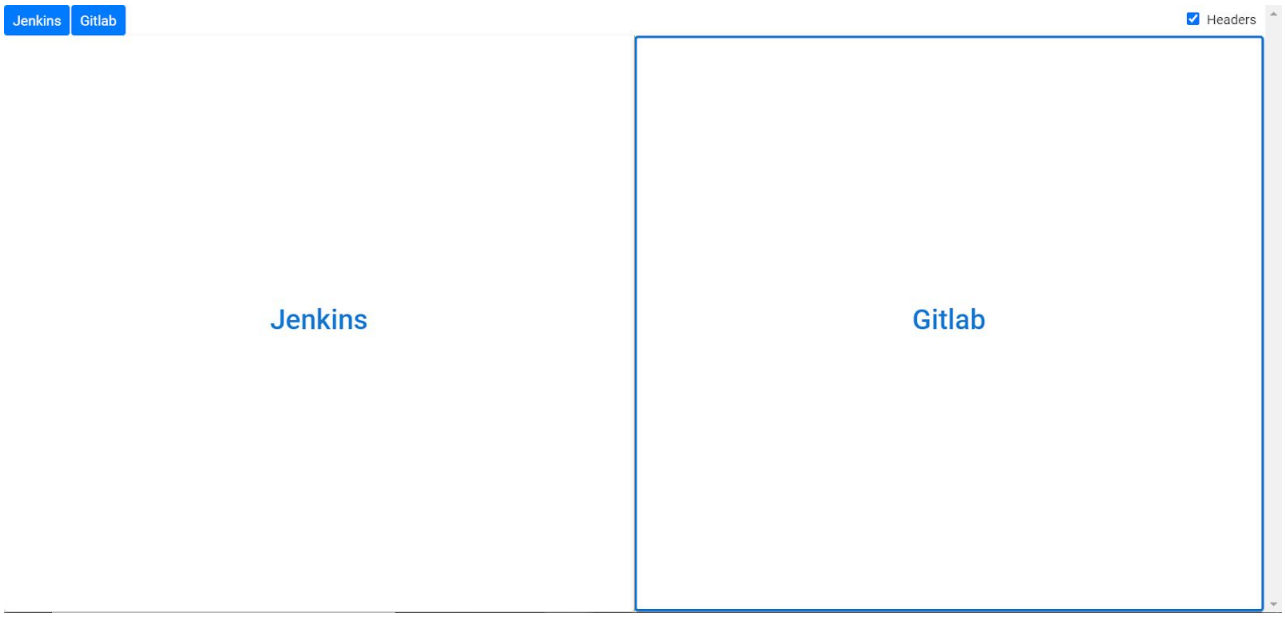
Username	Password
root	welcome123

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

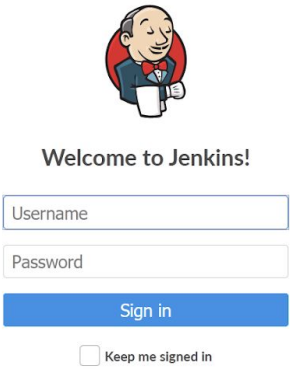
Username	Password
admin	welcome123

Lab Setup

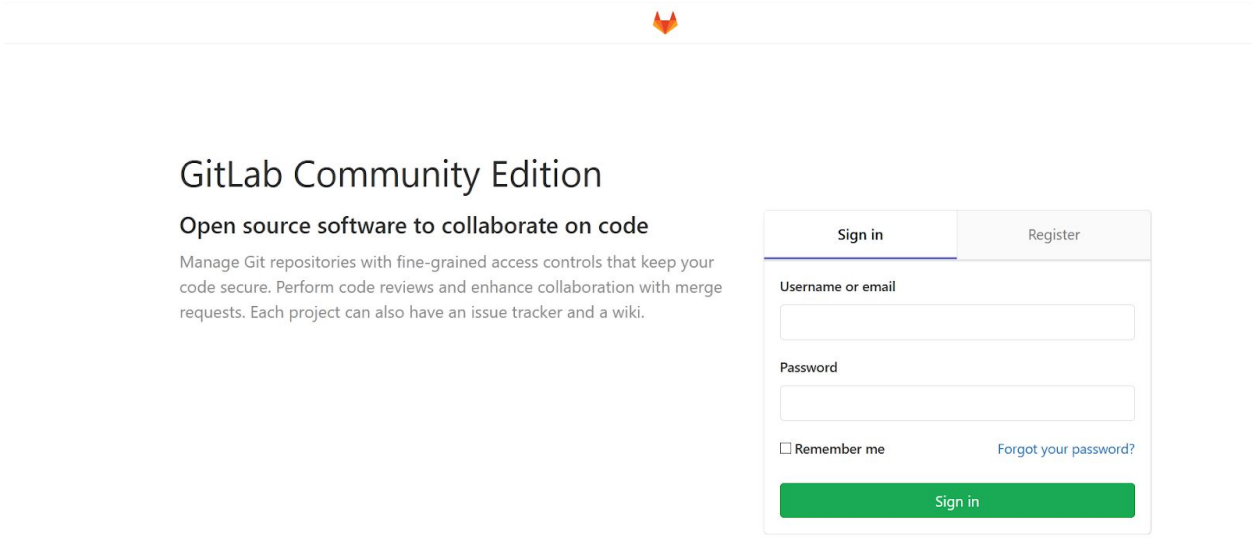
On starting the lab, the following interface will be accessible to the user.



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



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

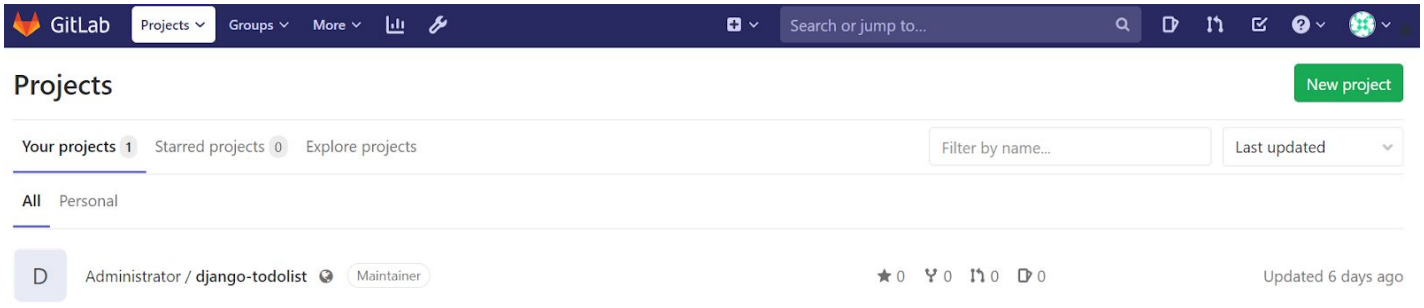


The GitLab instance takes time to function properly. Hence, for some time the following wait page might be visible. This page reloads automatically.

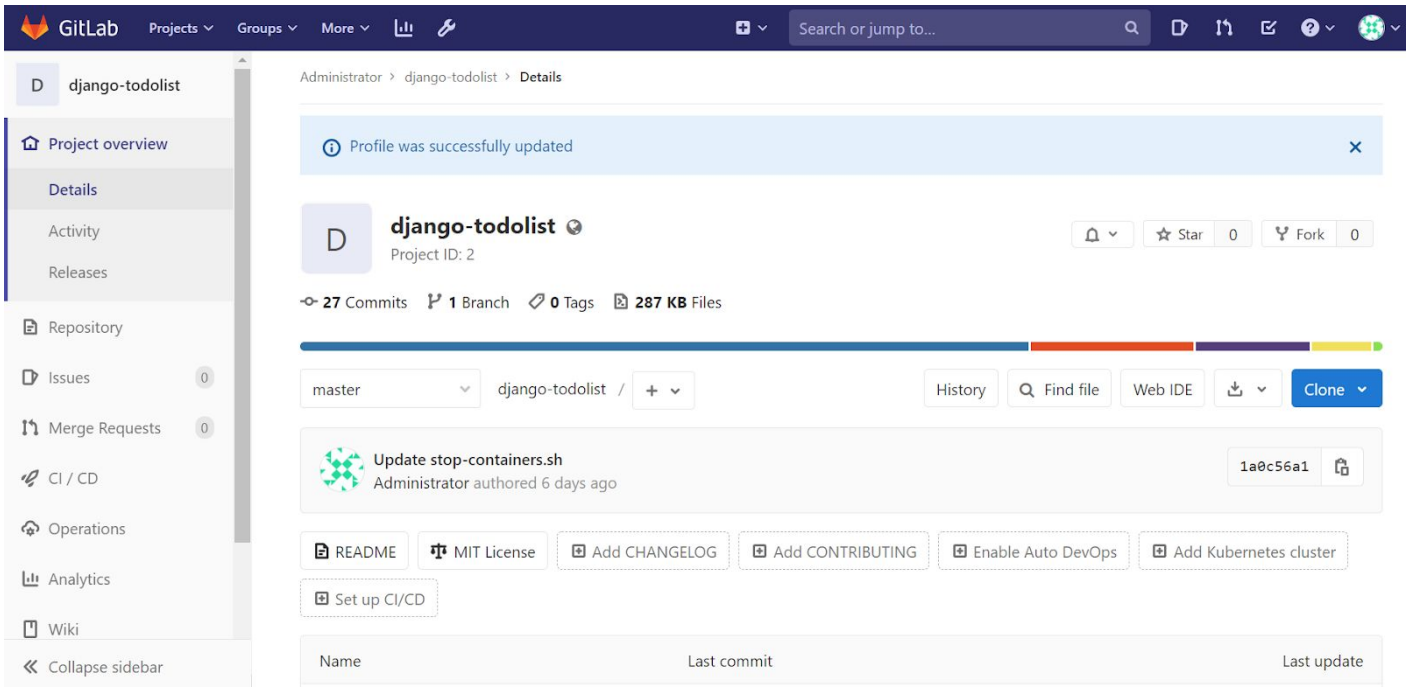
Solution

Step 1: Login into GitLab instance using the provided credentials:

Username: root
Password: welcome123

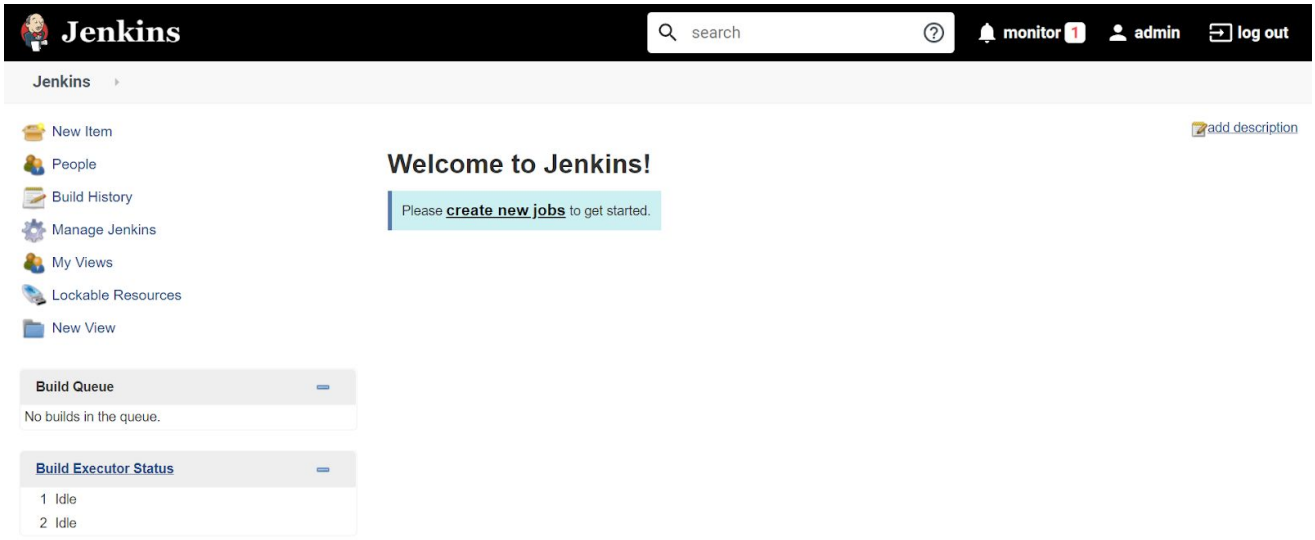


There is a repository present in the Administrator's account. Click on the repository link to open the repository page.



Step 2: Login into Jenkins instance using the provided credentials:

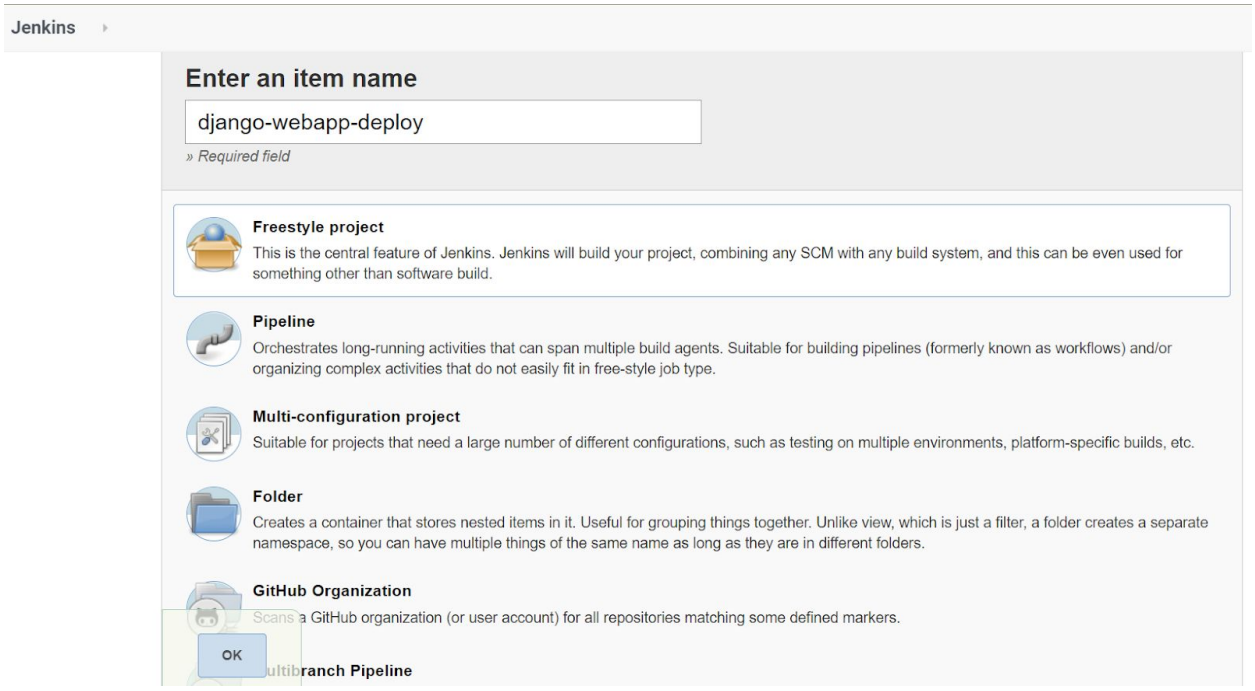
Username: admin
Password: welcome123



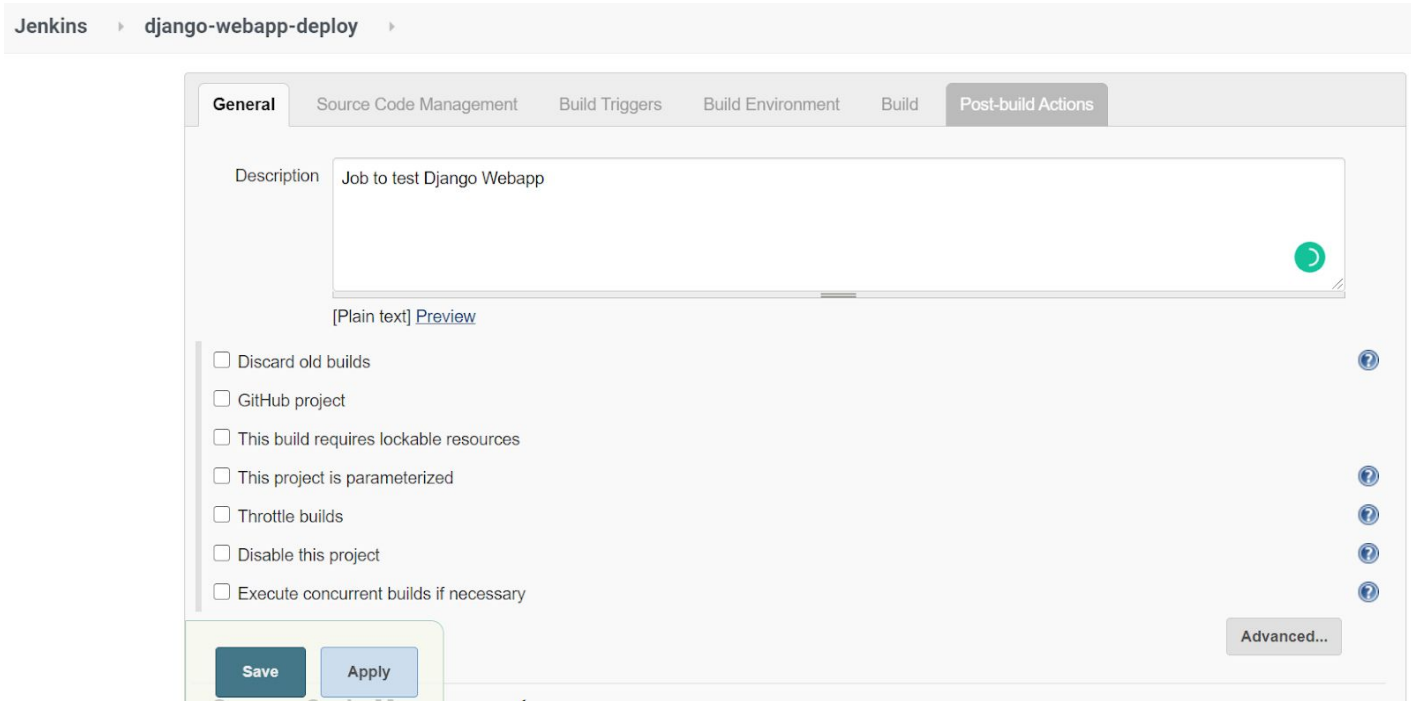
There is no job in this Jenkins server.

Step 3: Click on the “create new jobs” link. A job creation wizard will run. Enter the name of the job on the first page. You are free to select any name you like. We are using “django-webapp-deploy” because this project doesn’t really need to be built but only to checked/tested that it is ready for deployment.

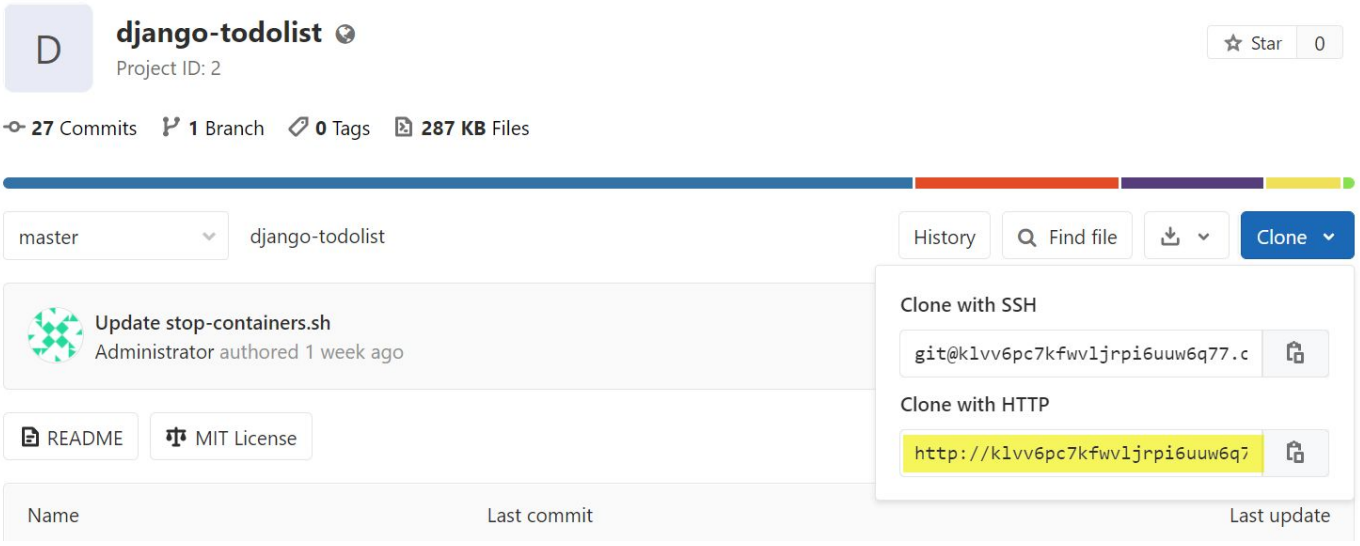
Then select the “Freestyle project” option and click the “OK” button.



On the next page, enter some description of the job.



Step 4: Open the Django project repository page on GitLab and copy the “Clone with HTTP” path.

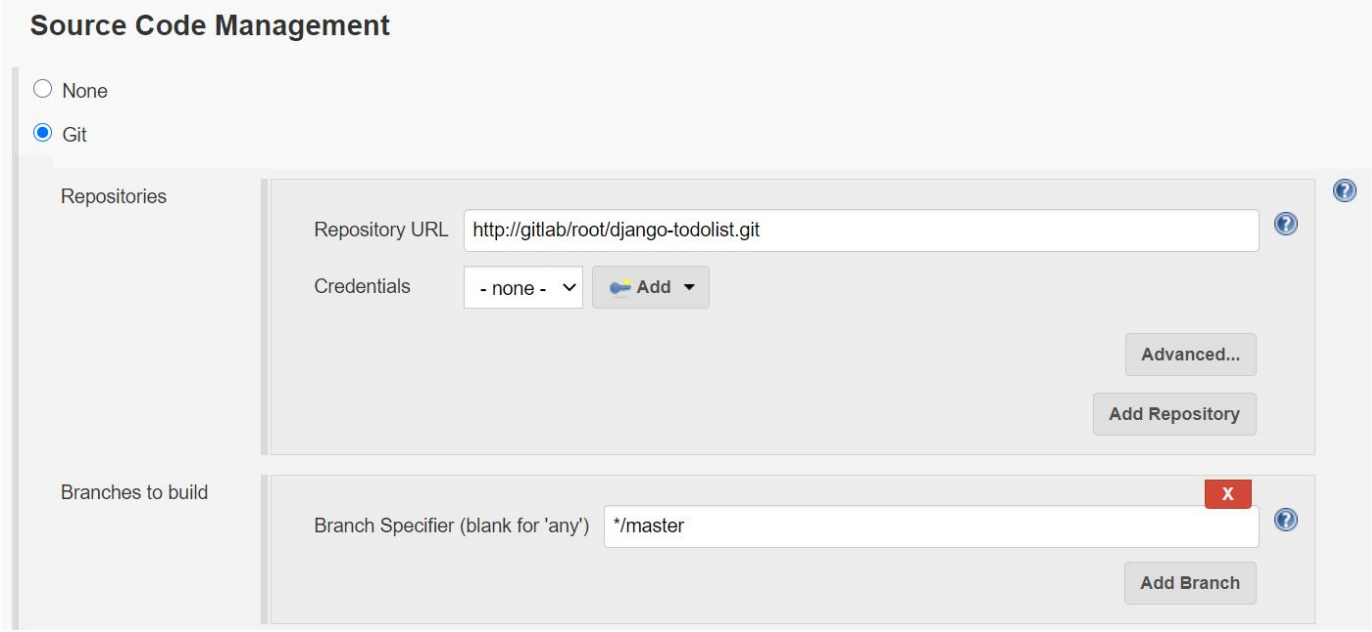


Step 5: Select “Git” in the Source Code Management section and paste this path in the “Repository URL” field.

The copied URL will look something like this:
http://<random_server_URL>/root/django-todolist.git

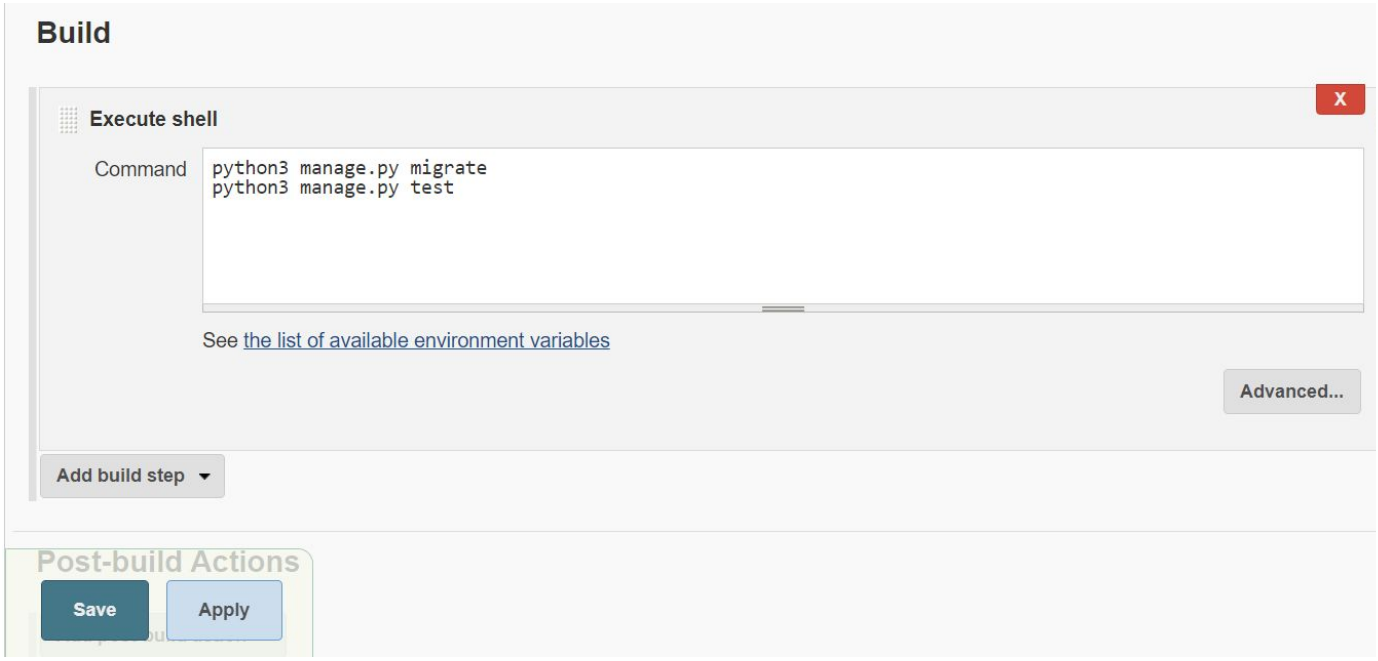
This URL is to make sure that the GitLab files can be accessed through web UI. However, to clone it to the Jenkins server, change it to:

http://gitlab/root/django-todolist.git

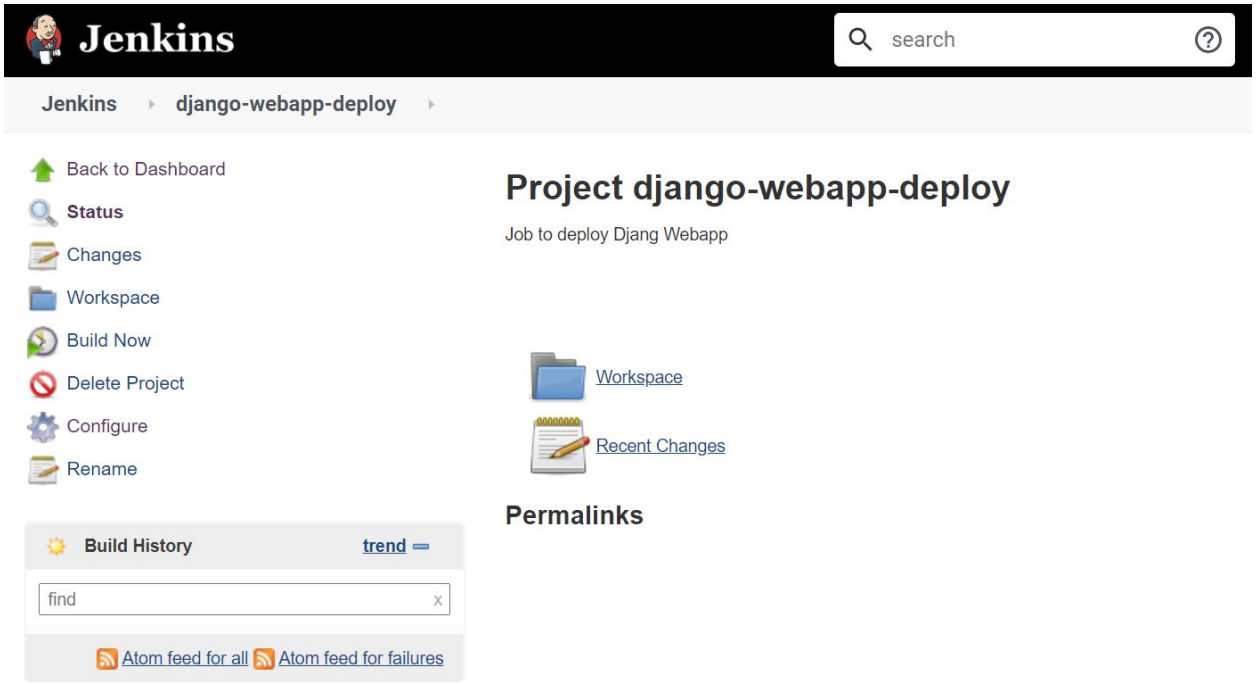


Step 6: In the build section, select “Execute shell”. Run migrate and test on the webapp. This is to test if the webapp is ready to be deployed. Paste the following commands in the build step:

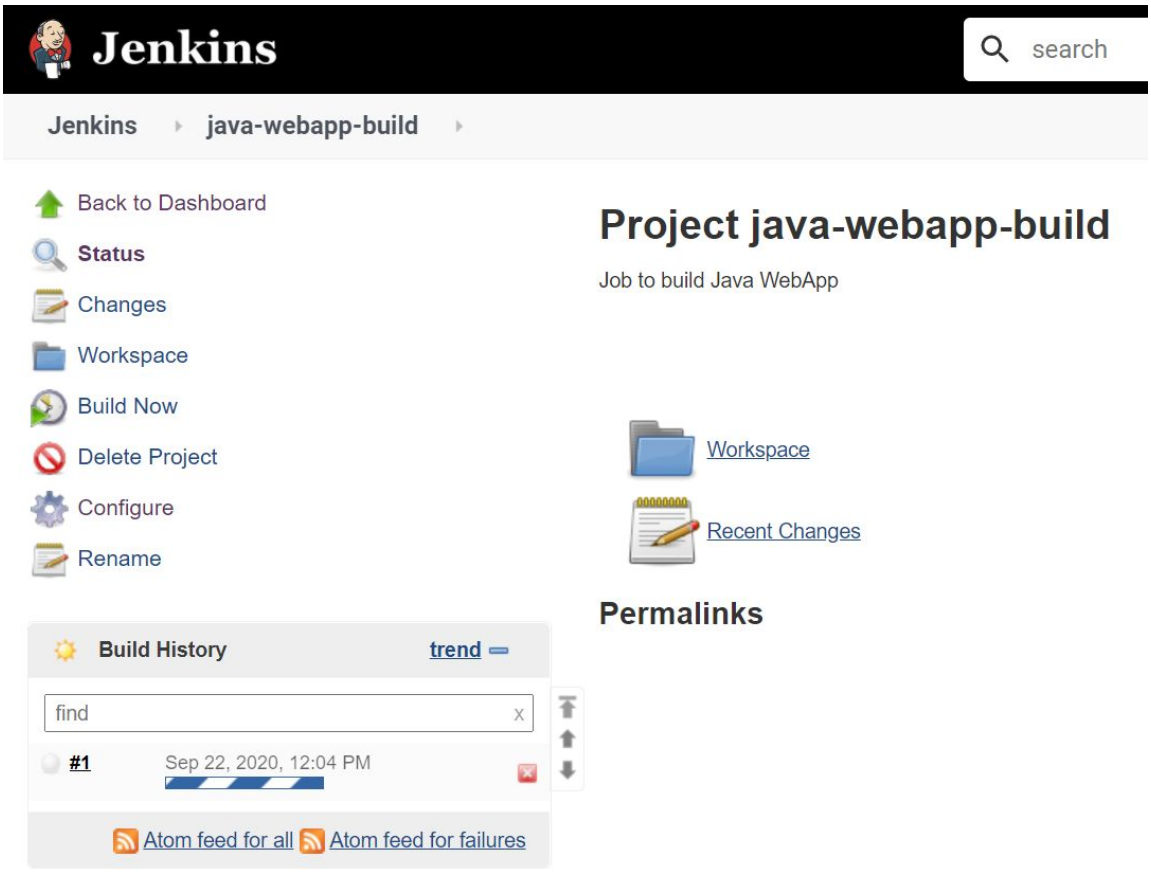
Commands
python3 manage.py migrate
python3 manage.py test



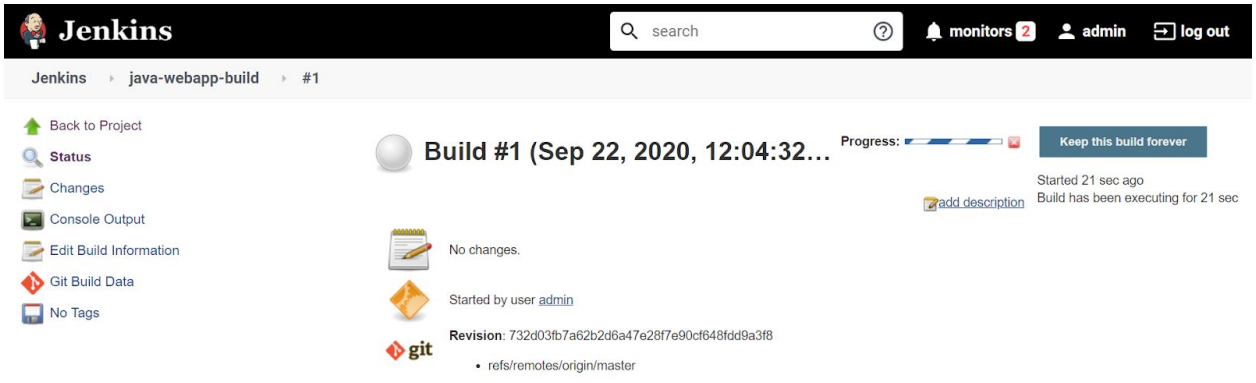
On clicking the “Save” button, the job will be saved and the page will redirect to the Job page.



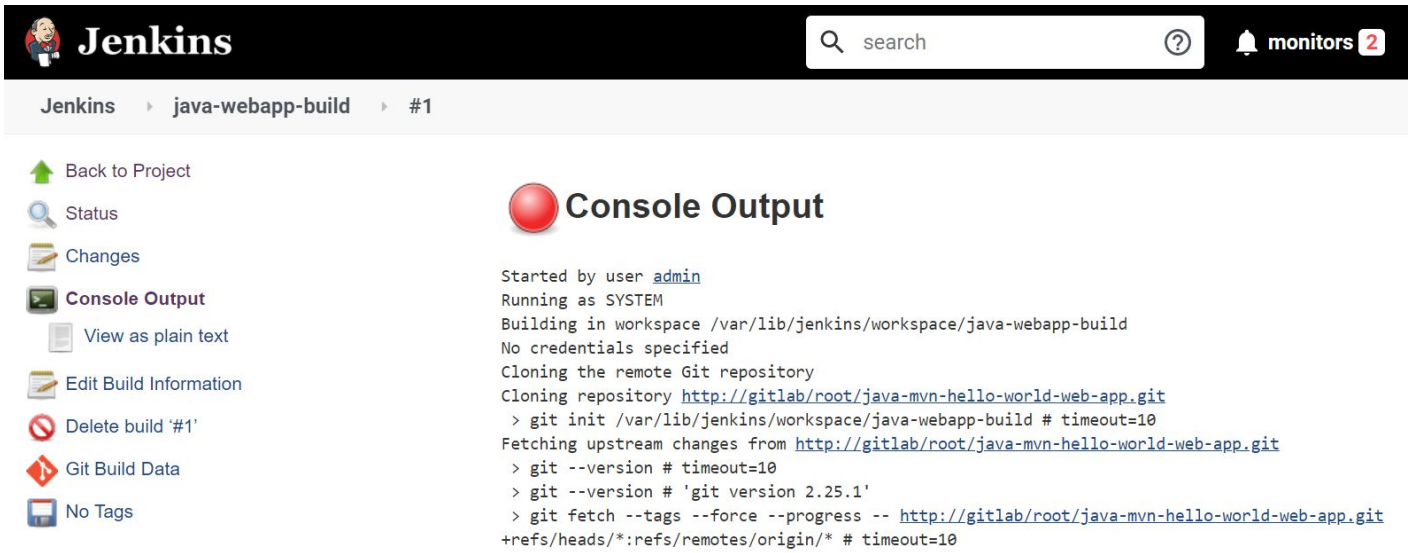
Step 8: Click “Build now” from the left-hand menu to fire the job.



An in-progress will appear under the “Build History”. Click on the number #1 to visit the build page.



Step 9: Click on “Console Output” to view the build logs.




```
[django-webapp-deploy] $ /bin/sh -xe /tmp/jenkins4620753108026315603.sh
+ python3 manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, lists, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
```

```
+ python3 manage.py test
Creating test database for alias 'default'...
.....................................................
----------------------------------------------------------------------
Ran 54 tests in 6.098s

OK
Destroying test database for alias 'default'...
System check identified no issues (0 silenced).
Finished: SUCCESS
```

The job completed successfully after the successful execution of migrate and test commands.

Learning

Testing the Django web app using Jenkins.