TASK -2

1. Create container image that’s has Jenkins installed using dockerfile

2. When we launch this image, it should automatically starts Jenkins service in the container.

3. Create a job chain of job1, job2, job3 and job4 using build pipeline plugin in Jenkins

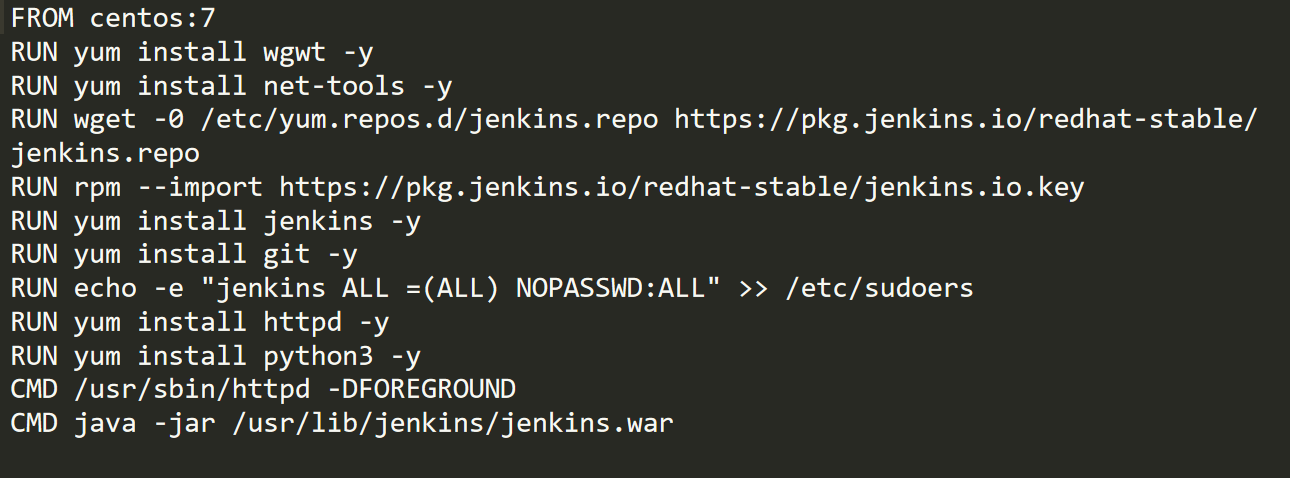
4. Job1 : Pull the Github repo automatically when some developers push repo to Github.

5. Job2 : By looking at the code or program file, Jenkins should automatically start the respective language interpreter install image container to deploy code ( eg. If code is of PHP, then Jenkins should start the container that has PHP already installed ).

6. Job3 : Test your app if it is working or not.

7. Job4 : if app is not working , then send email to developer with error messages.

8. Create One extra job job5 for monitor : If container where app is running. fails due to any reason then this job should automatically start the container again.

First we start by making a dockerfile which will help to start Jenkins when a container is run

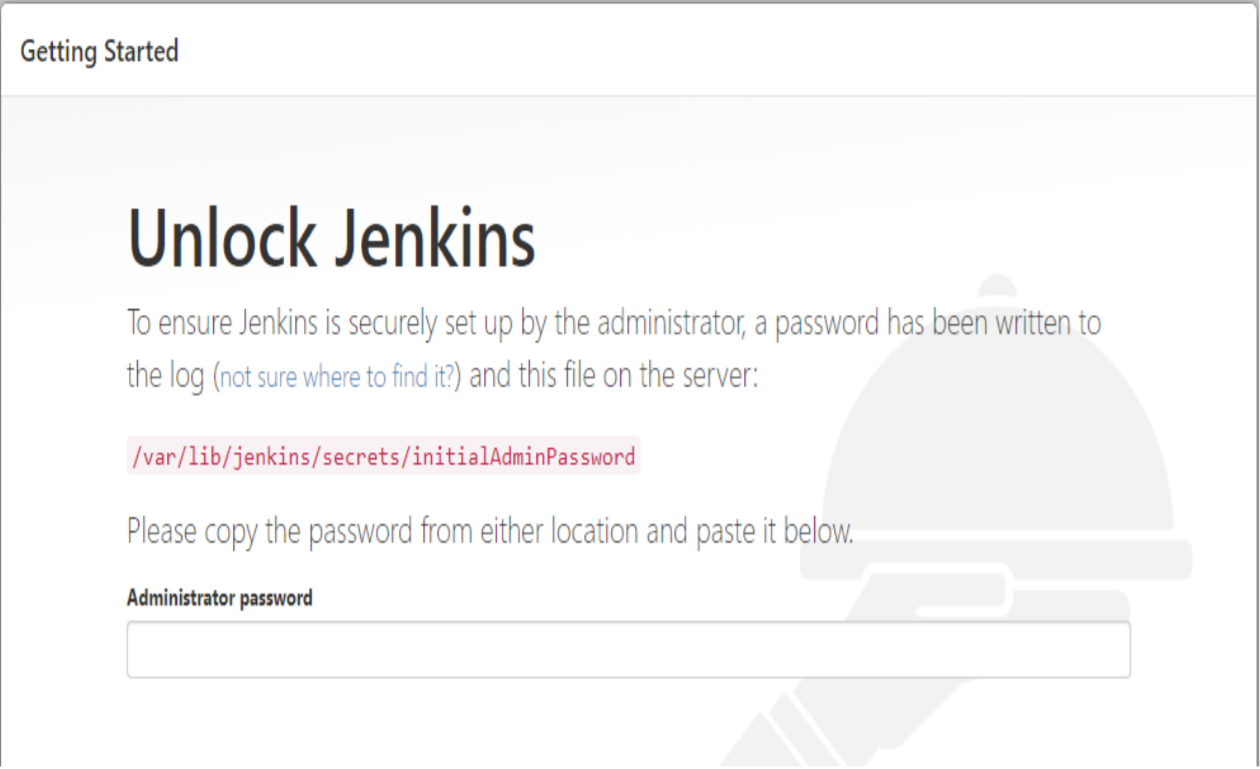
To build docker file we use following command

docker build –t dockcontainer ws1

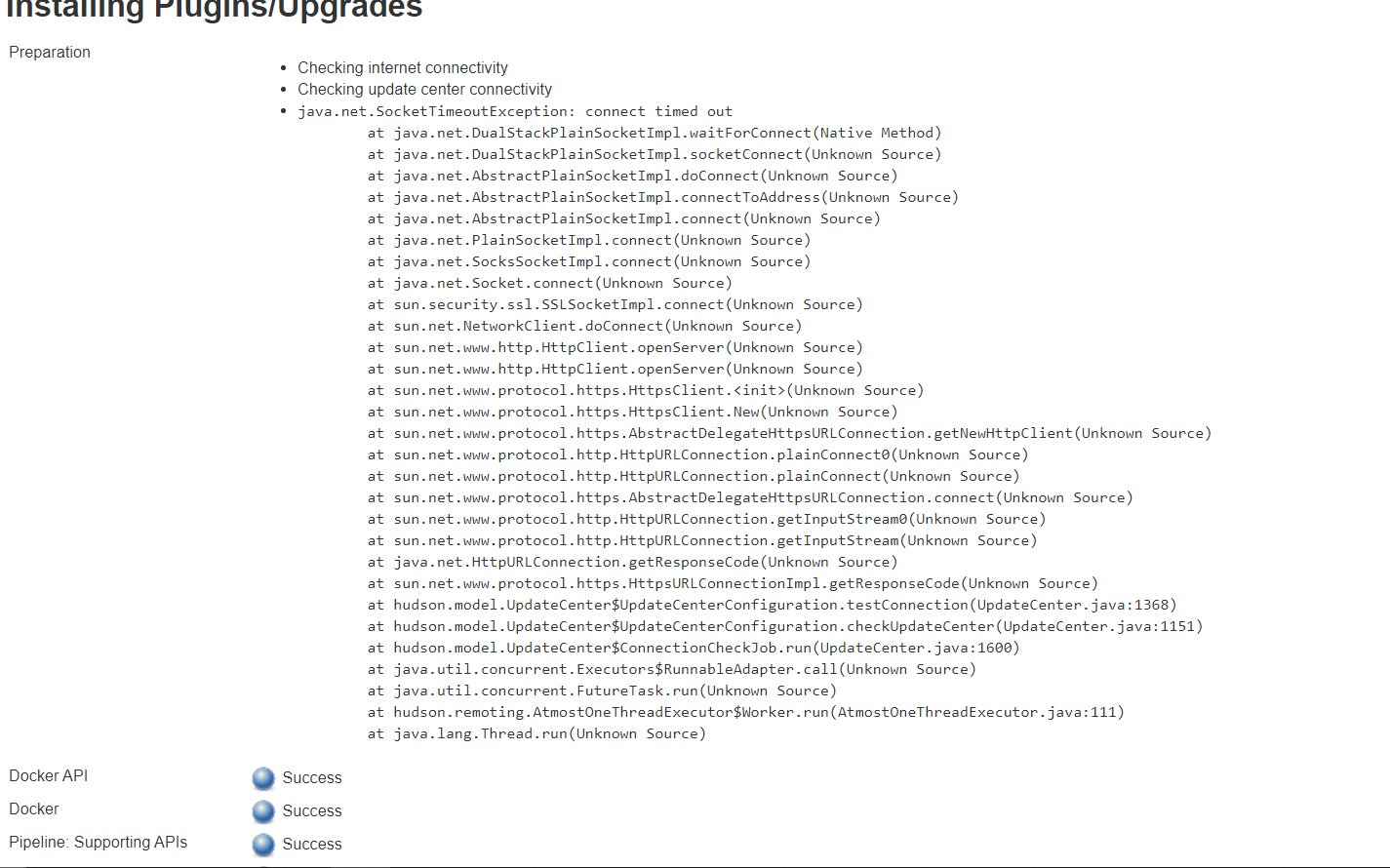
Then, we have to use another command to launch the container for automatically starting Jenkins on port no. 7777

Docker run -dit --privileged -p 7777:8080 -v /:/host dockcontainer

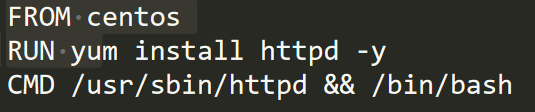
When we will start Jenkins in our browser on port 7777 then it will ask for password to us which we are required to provide it. So then we will provide it initial Admin Password.



Installing the required Jenkins plugins

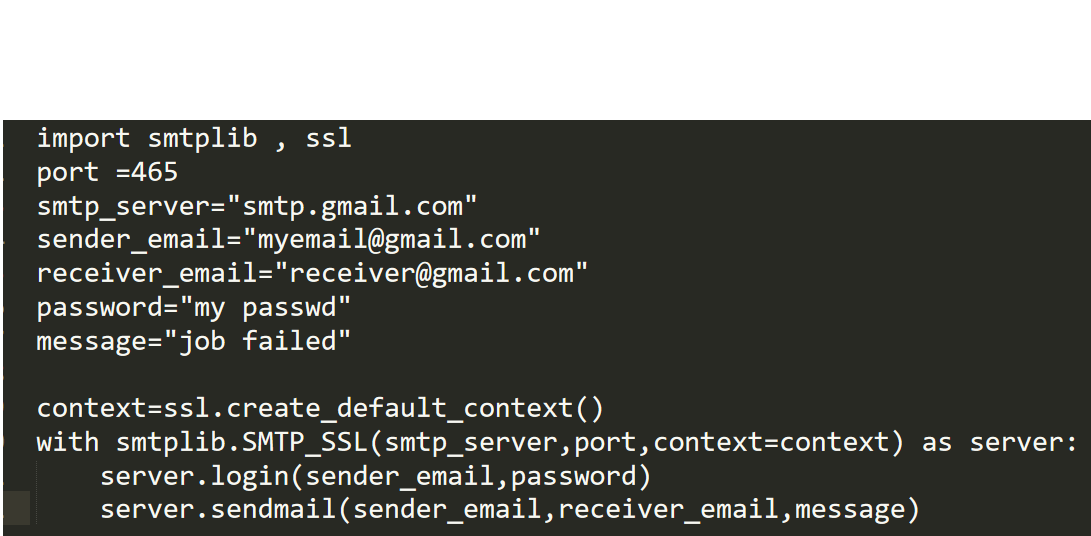


Dockerfile to run httpd

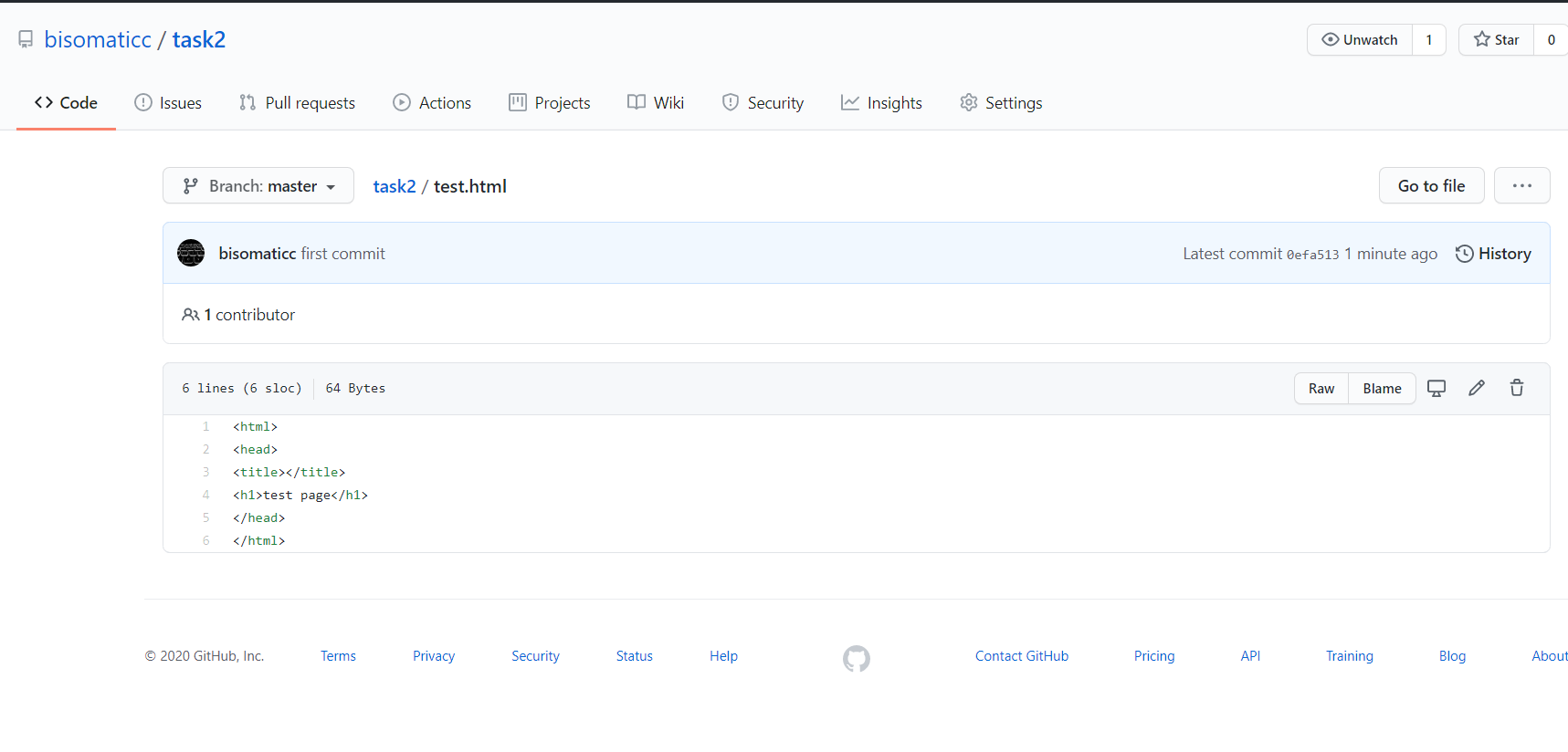


Email file

Python file to mail in case our program goes wrong



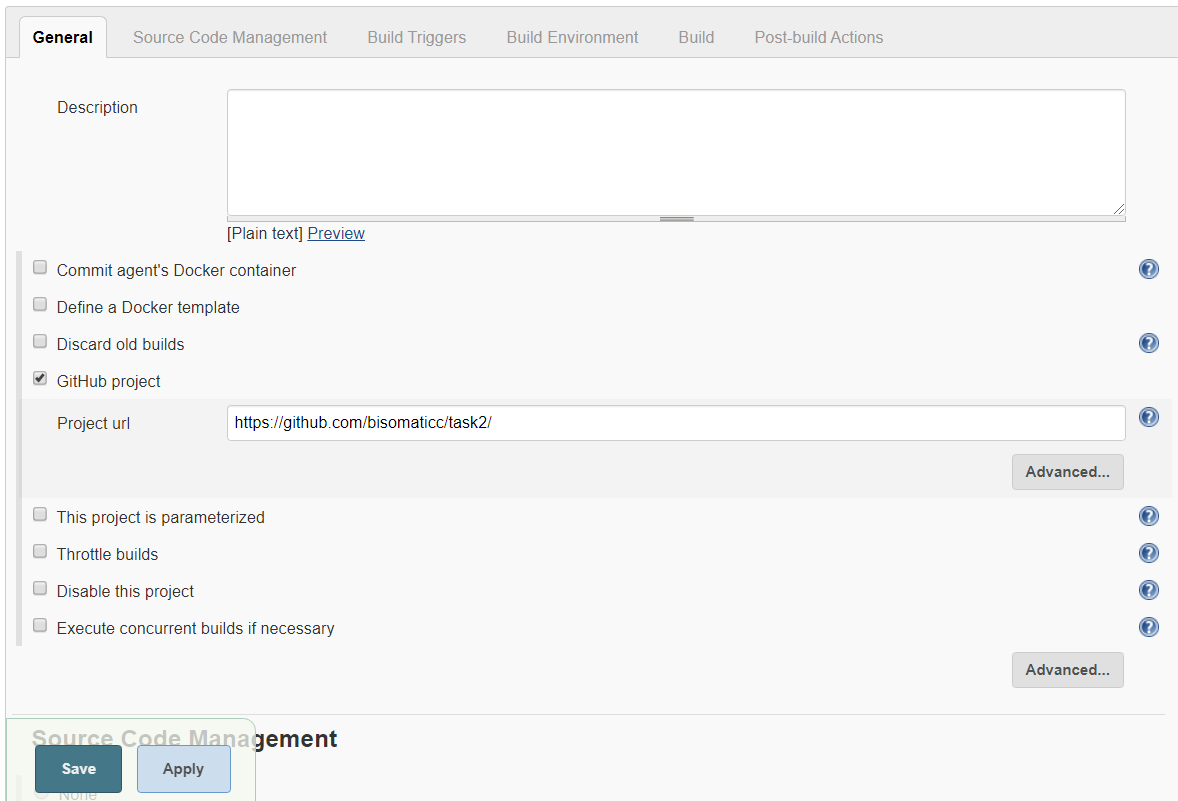
Html page pushed to repo for testing

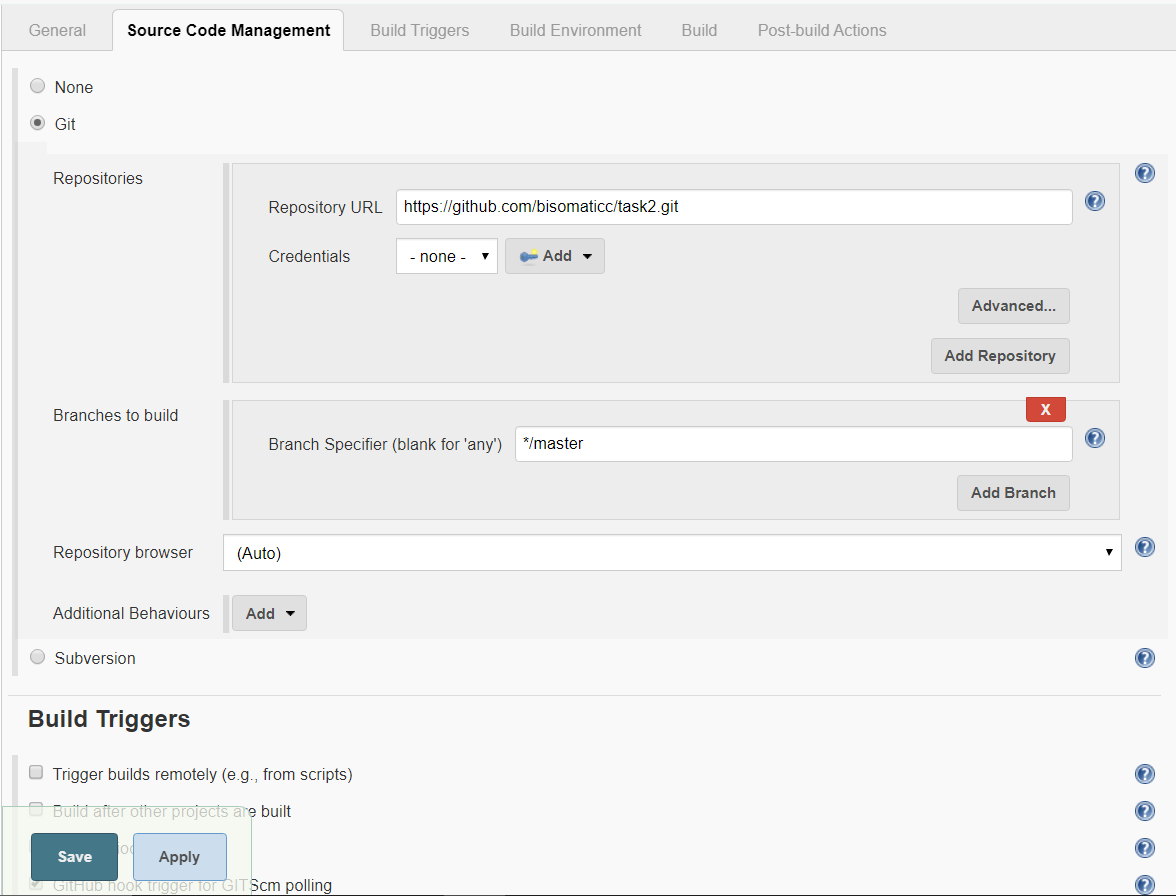


On Jenkins

Job1

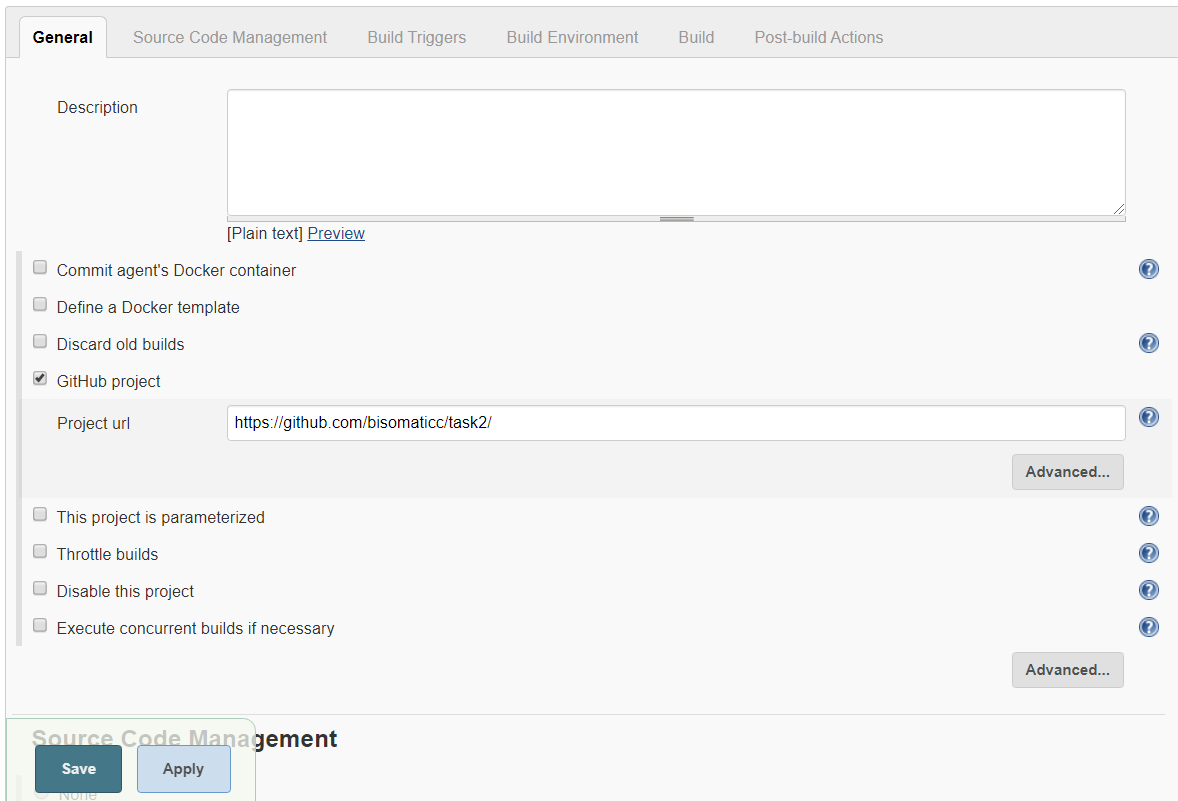
This will trigger task 2 on github

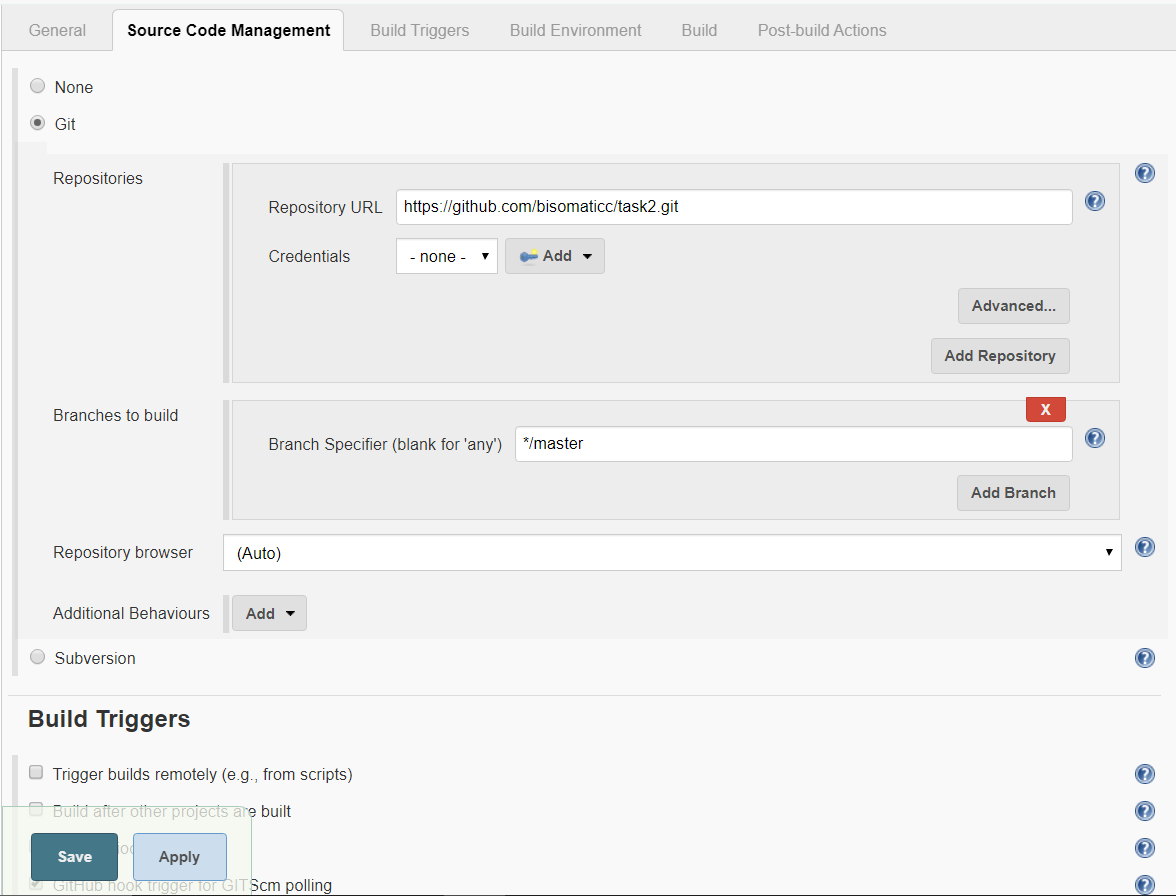


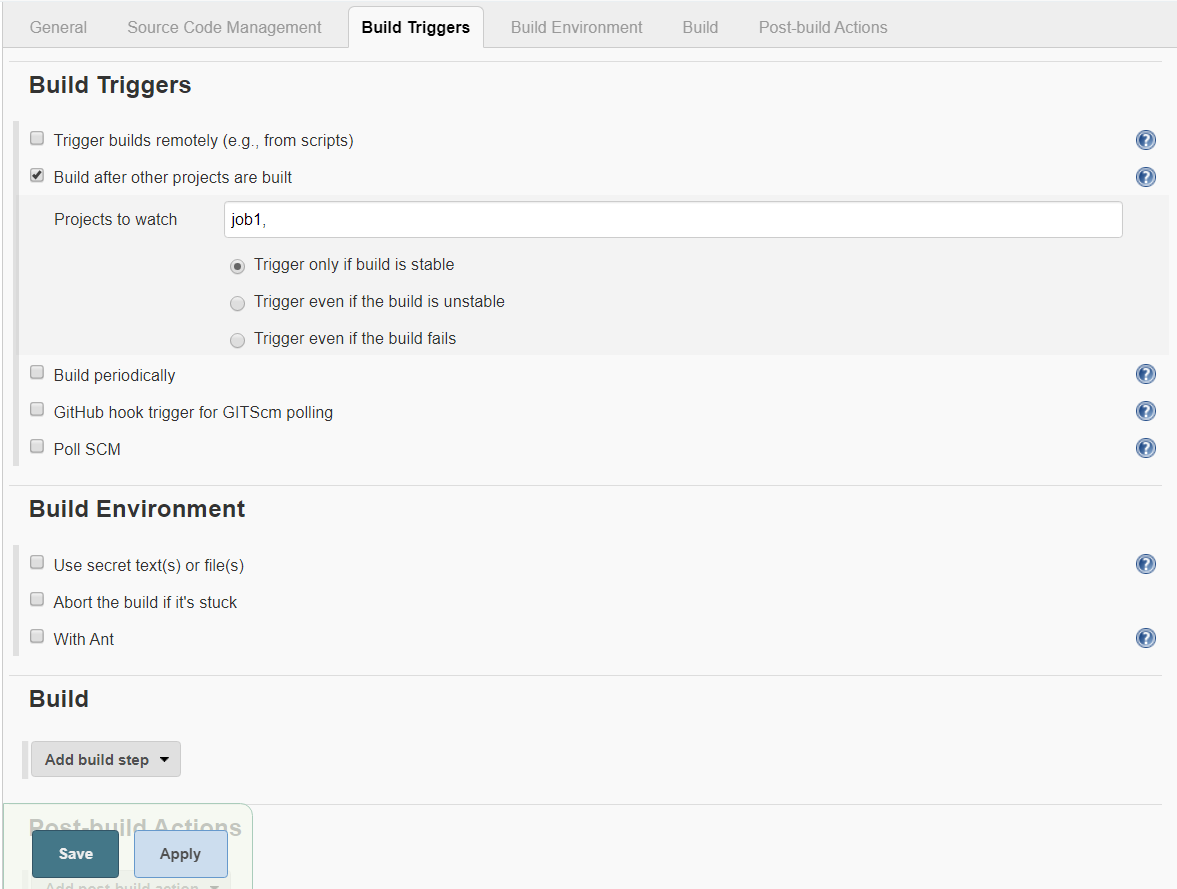


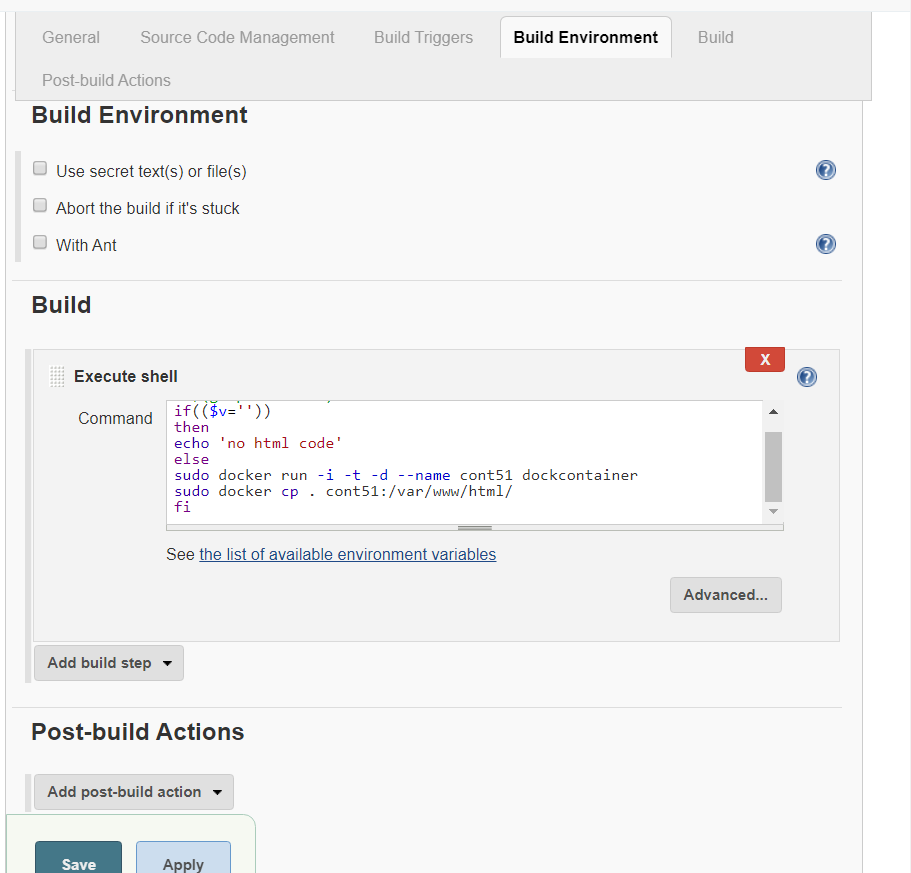
Job2

This job will check if files in our repo are html page or not if yes it will launch a new container else will return no html code in it



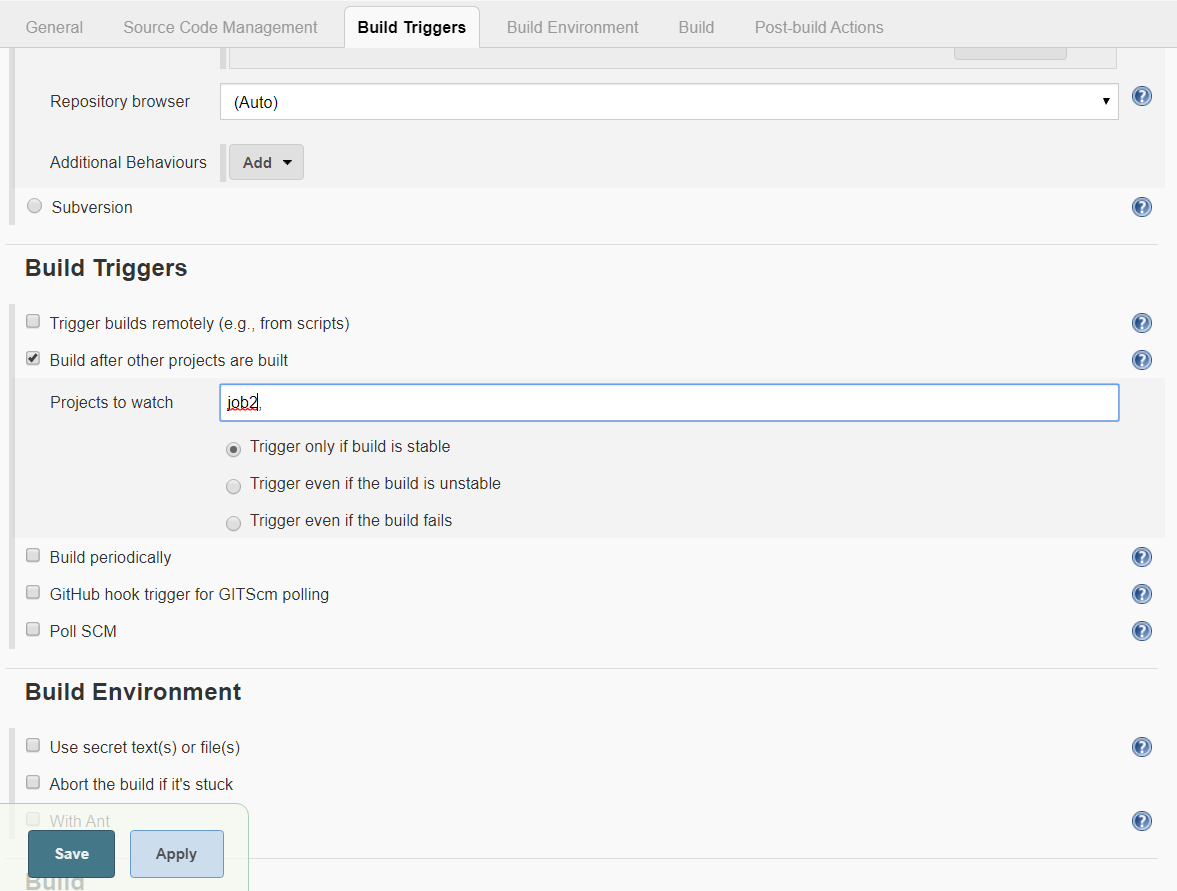


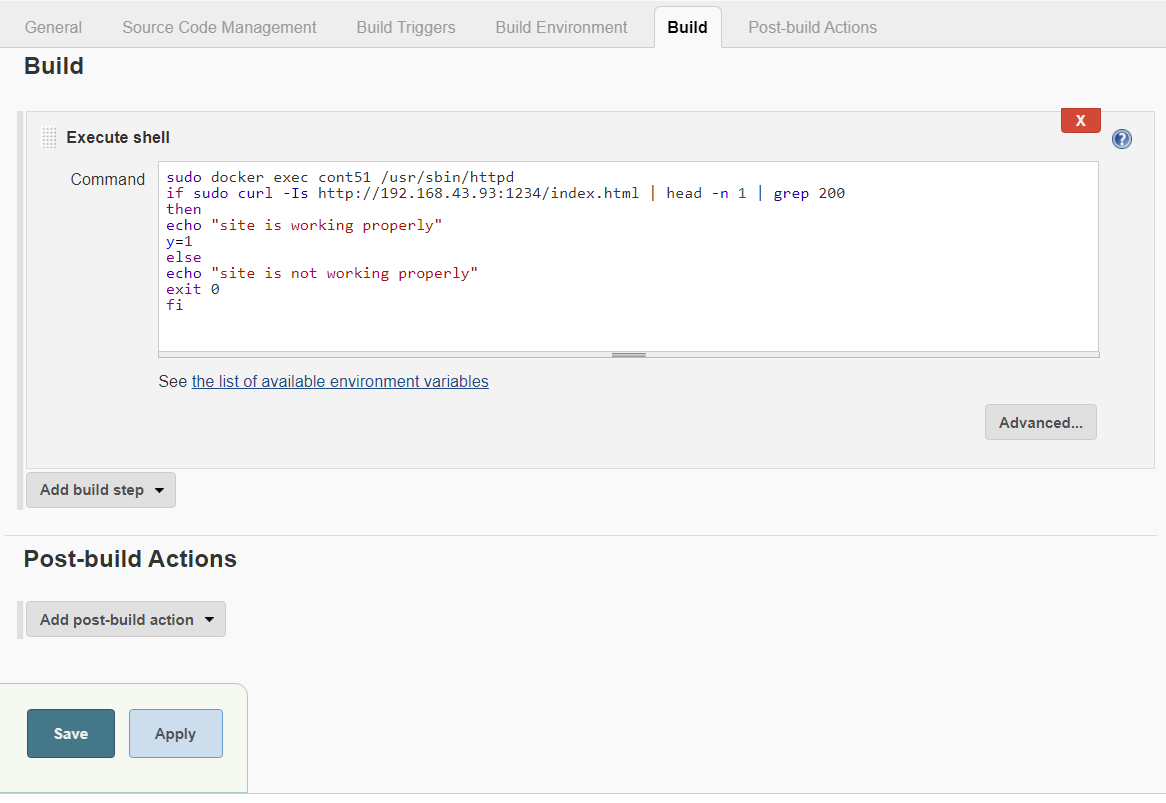




Job3

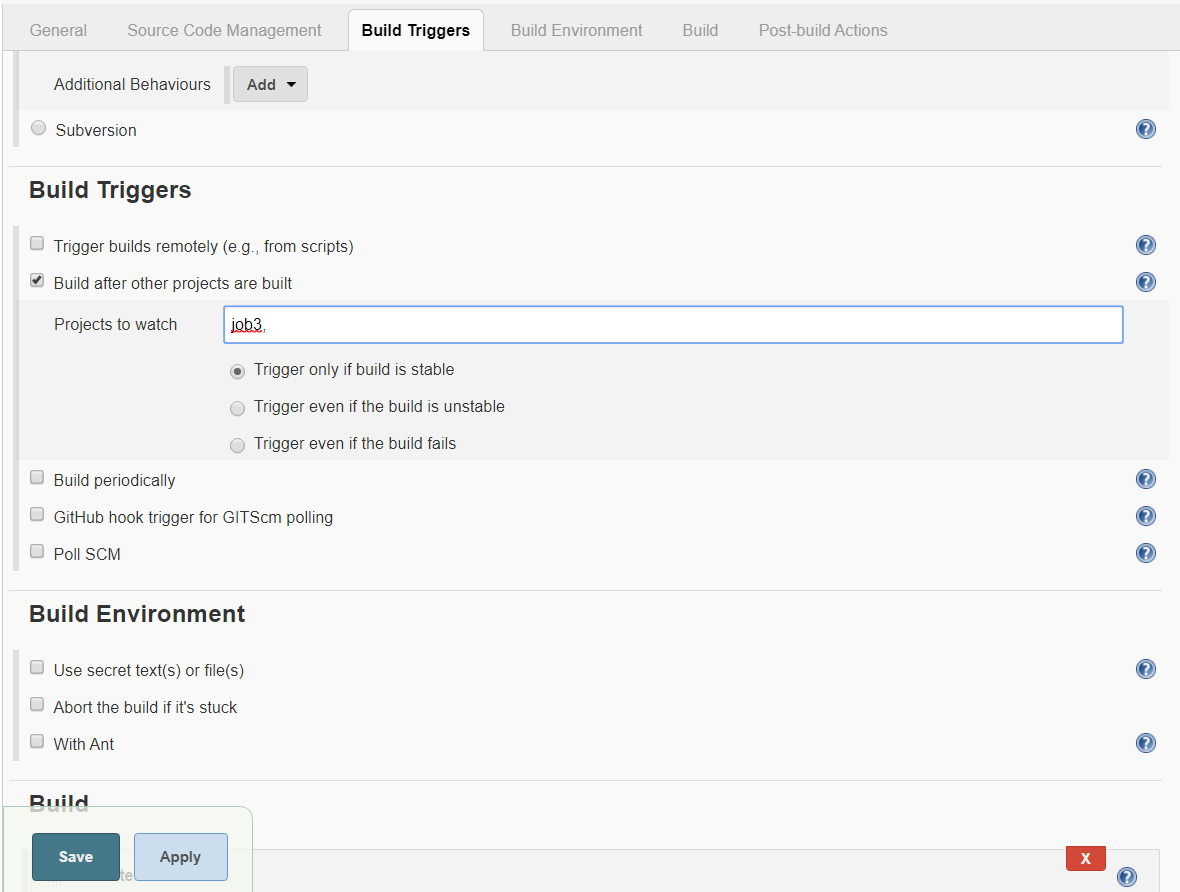
This job is to check if our website is working correctly or not if not it will exit else will update value of y to 1

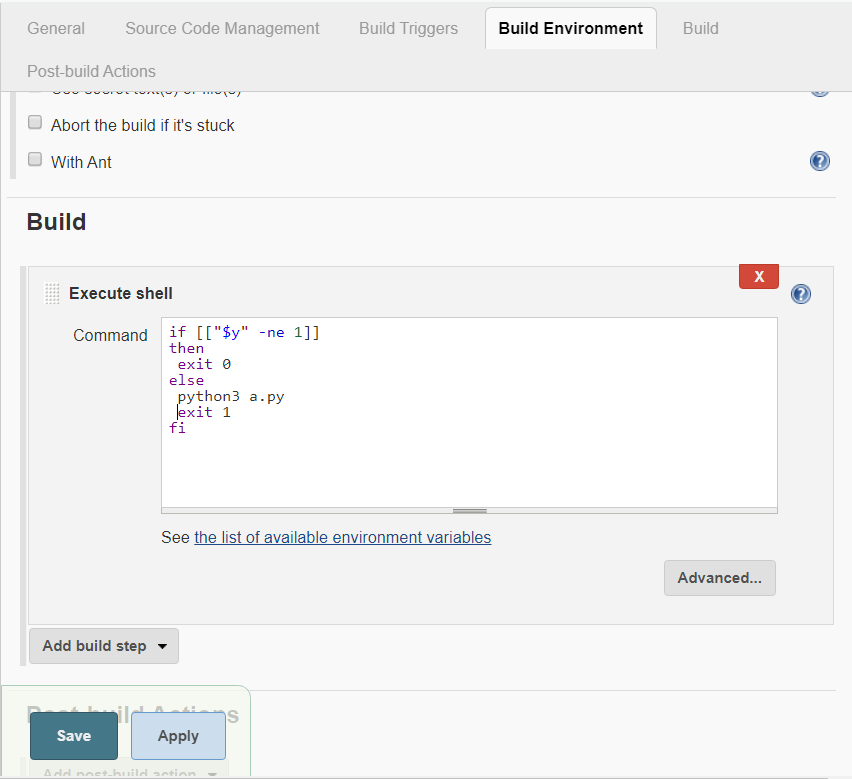




Job4

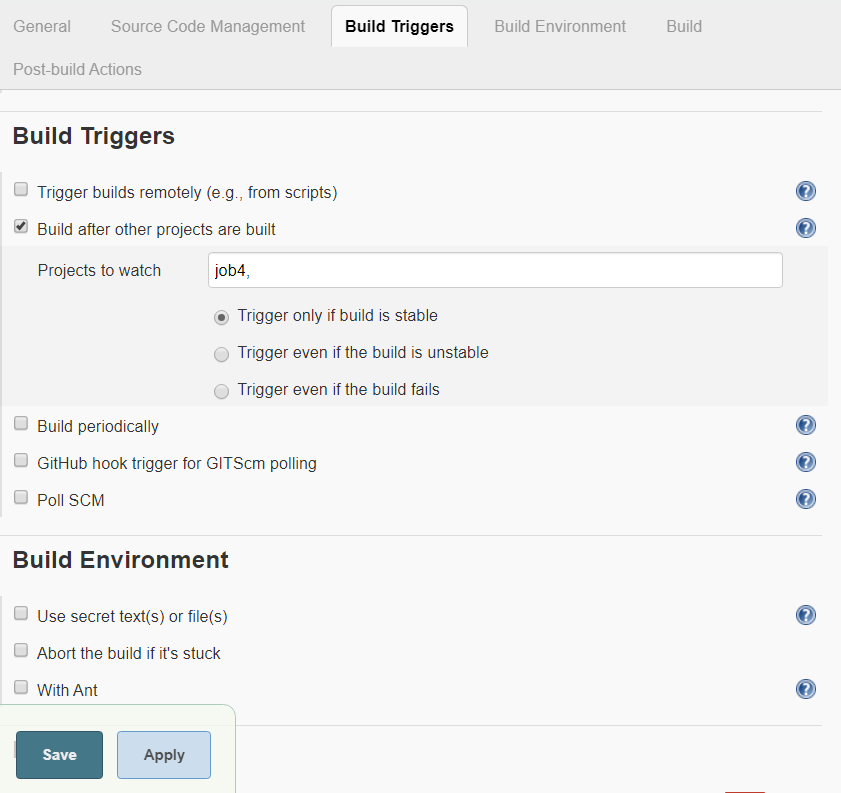
This will check the value of y if its 1 that means our webpage was working correctly

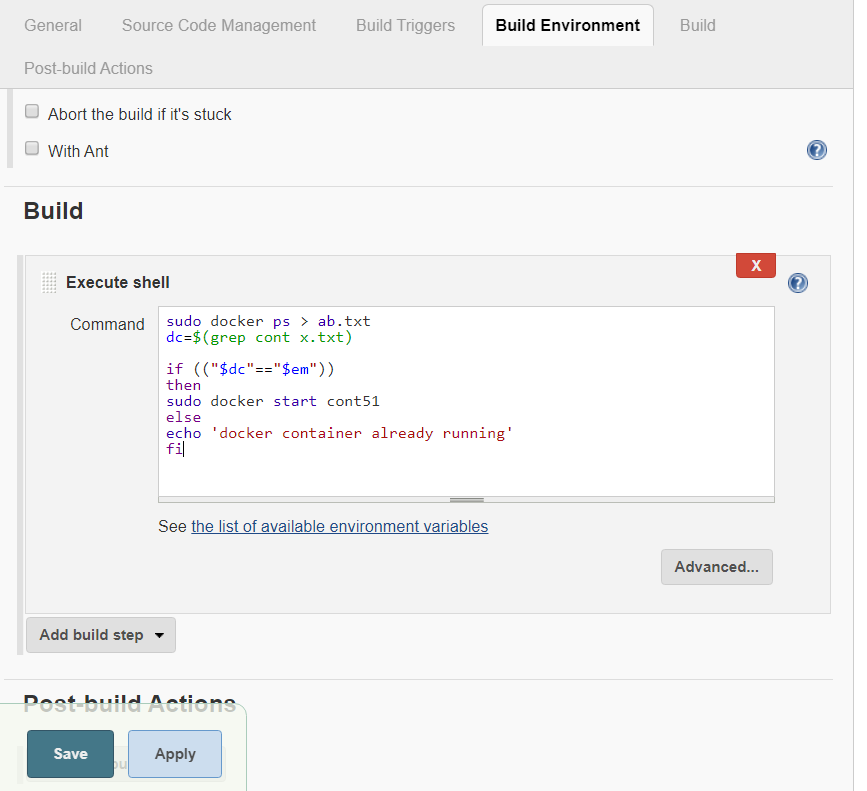


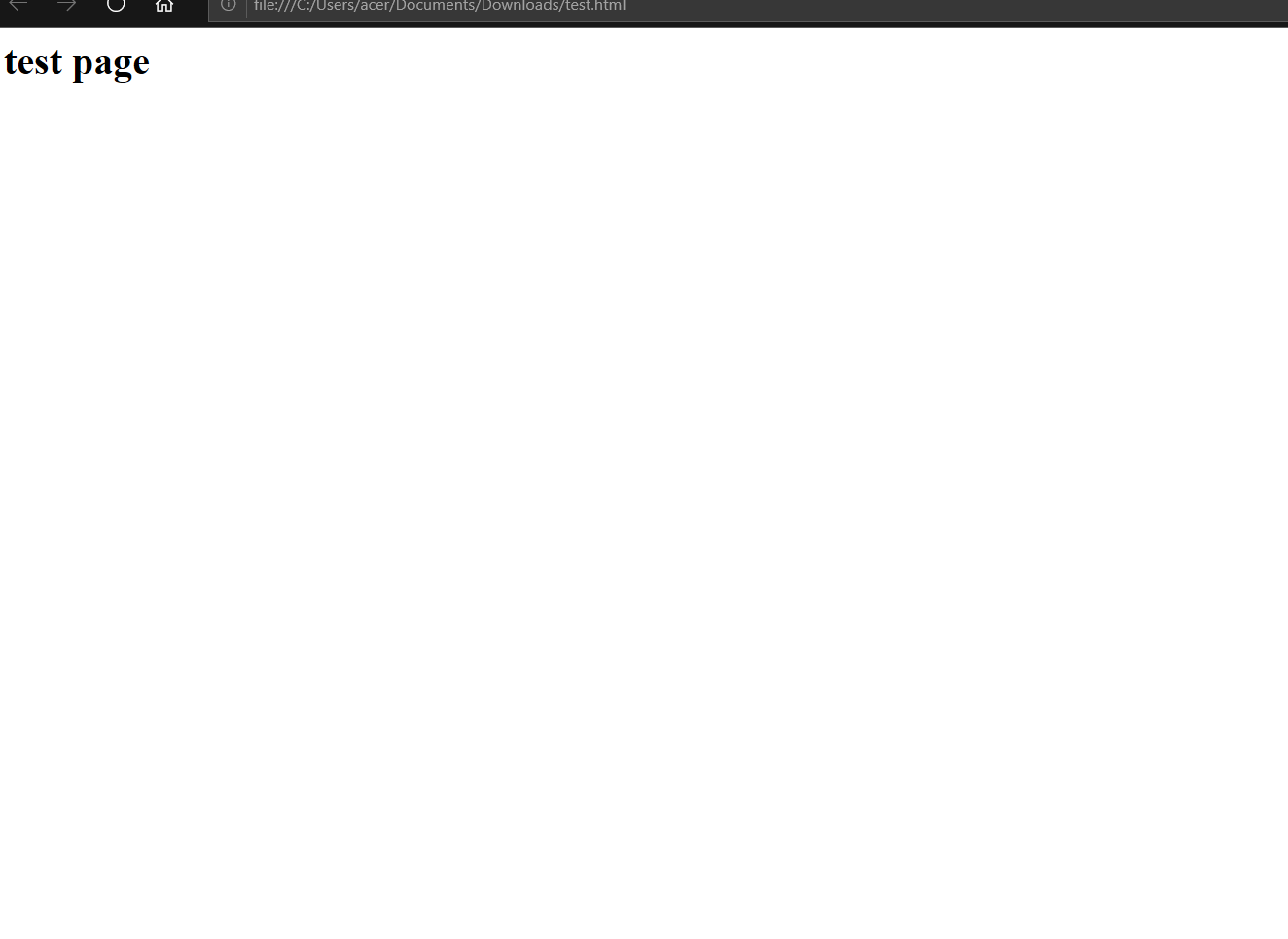


Job5

This job will check if our container is working properly if not it will be launched again else will output message of it running







Finally at the end we will have to build pipeline which will show our jobs running properly above images show our website working properly