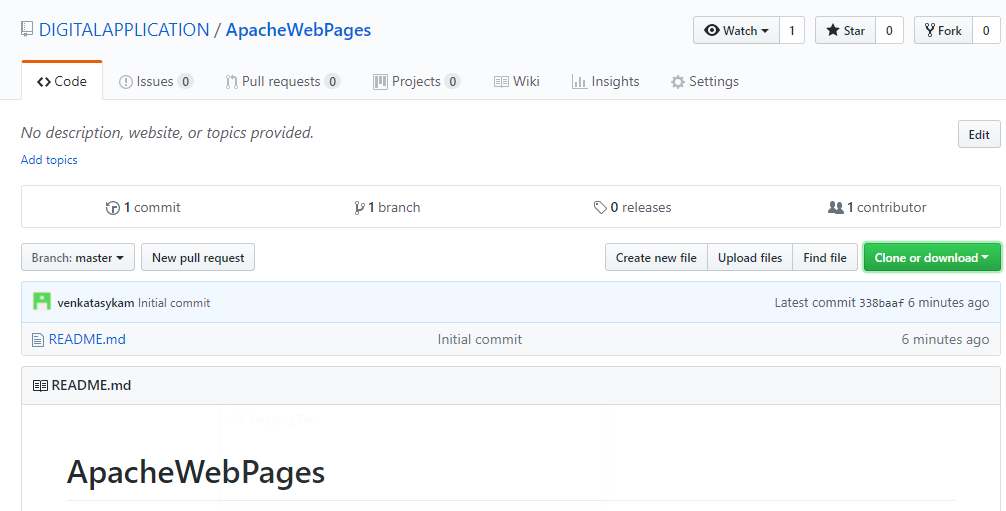
**Task details:** Jenkins job should perform the below steps.

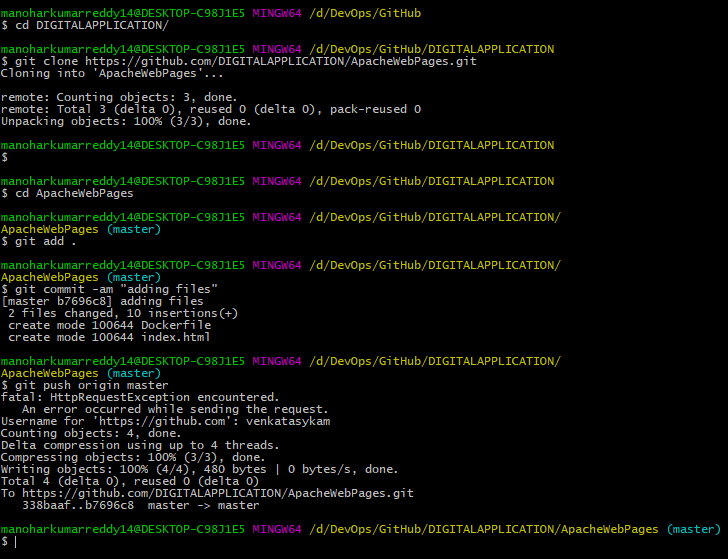
1. Checkout the source code from GitHub repo.
   1. Repo contains the original source code (here in Practice-1, we are taking an example, one web application in terms of HTML pages only).
   2. The same repo(where source code available) contains the **Dockerfile** which will create an environment for our web application as to run and test.
      1. Environment details to run & test our application: Ubuntu OS, apache web server installed on ubuntu, and then our web app will be deployed into this container/environment created by docker.
      2. Assume these are the same exact similar details as to compare with production. So that, we can test our apps which is exactly similar to the prod env. Many issues will be solved during development stage.
2. Build the project (compiling source code, packaging etc) if required. But in this example, we are using only HTML static files as our web app (just to practice & understand the docker how this will work). Build is not required for this example. In Practice-2, we will take one webapp which is required to build with maven.
   1. Job is running on slave machine.
   2. So make sure, the tools which are configured on master, the same should be installed on slave machine too i.e., maven, git, java, docker etc.
3. Create an environment/container using docker.
   1. Build the Dockerfile, it will create an image.
   2. Run the image, it will create a container.
   3. (Manually check)Access the web application in any browser once the environment is fully setup i.e., containing is up & running.

Note: Jenkins is connecting & performing all the above three steps(three different configurations), so this will be called **orchestration** tool. Establishing the connection between the different types of configurations is the process called **orchestration**. Examples are Jenkins, Ansible, chef, puppet, docker etc.

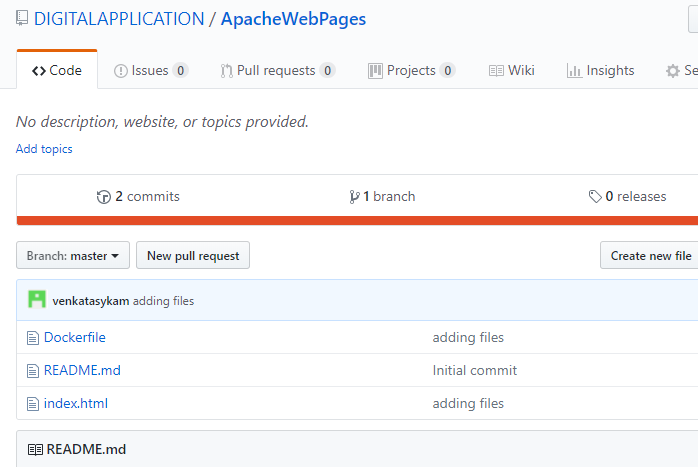
**Create a New repo in GitHub**:



Checkout the repo to local machine and then add **Dockerfile & index.html** files.



File structure after pushing the code to GitHub.



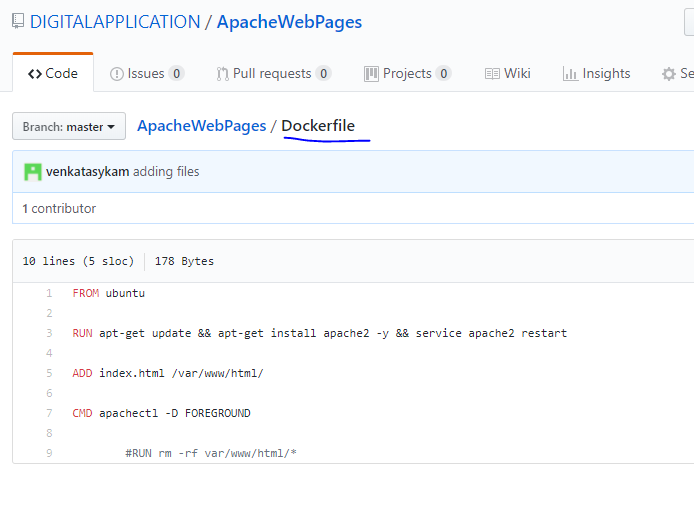
**Content of “Dockerfile”:**

FROM **ubuntu – image** 🡪 docker will download the image “ubuntu” from docker hub (docker repo) with the latest tag/version. Of we can specify the image tag as “FROM **ubuntu:12.04**”.

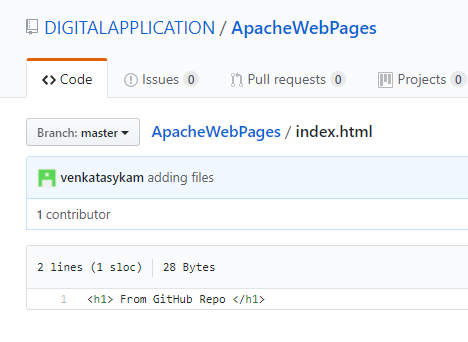
RUN 🡪 running the ubuntu commands as to update & install the required apps.

ADD 🡪 Adding the local **index.html** file to container (where ubuntu installed) in to particular path(/var/www/html/) where apache can execute the **index.html**.

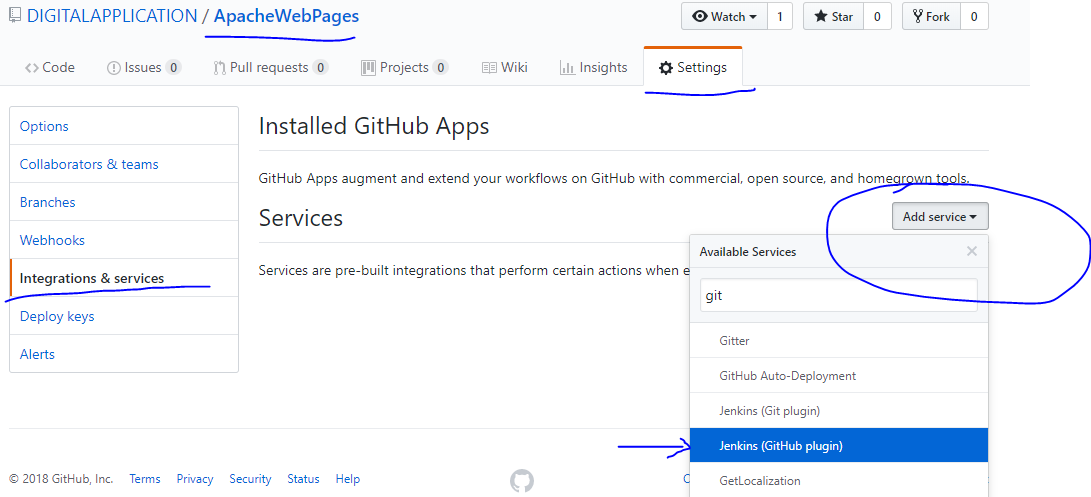
CMD 🡪 Execute the command apachectl –D FOREGROUND🡪 This means that systemd is capable of automatically restarting Apache if it does crash.

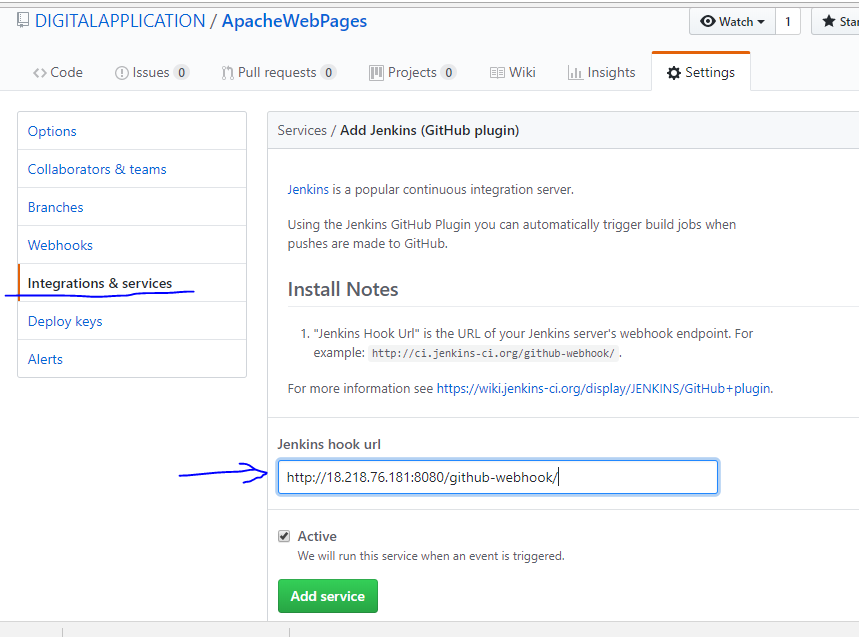


**Index.html** file content:

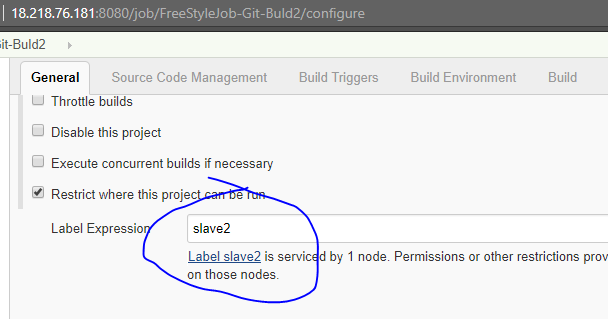


Configure the Jenkins web-hook URL in GitHub repo settings as shown below. So that, Jenkins job whcch is configured for this repo, will trigger automatically once the check-in done.

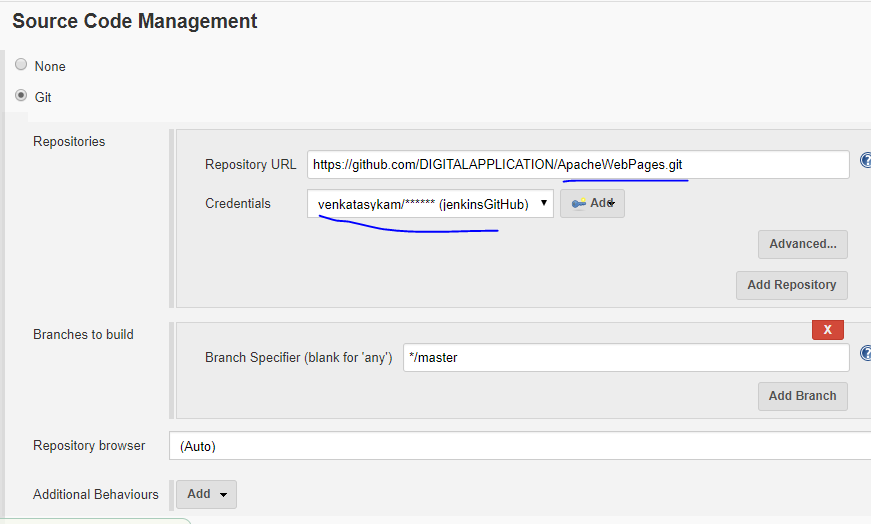


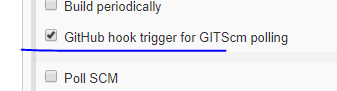


**Go to Jenkins and create a job:** Configure the job with restricted slave where docker installed. (docker did not installed where Jenkins exists. if you want, you can install docker on master Jenkins machine.).

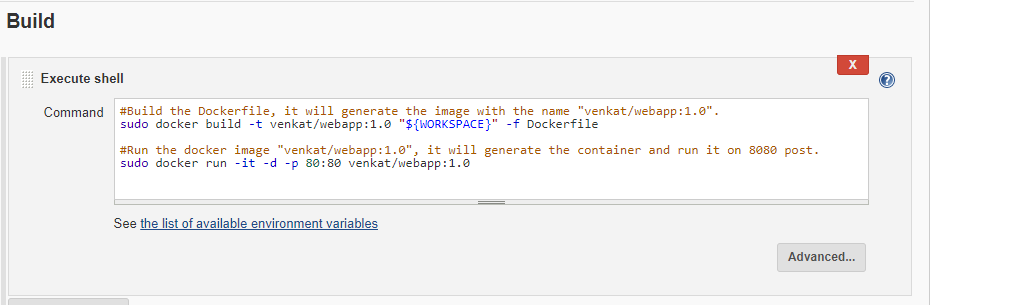


**Configure GitHub URL & credentials** to check-out the source code from GitHub repo:





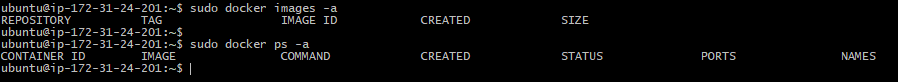
Build the docker file🡪it will generate image on out local machine🡪run the image🡪generate the container & run it on 8080 port.



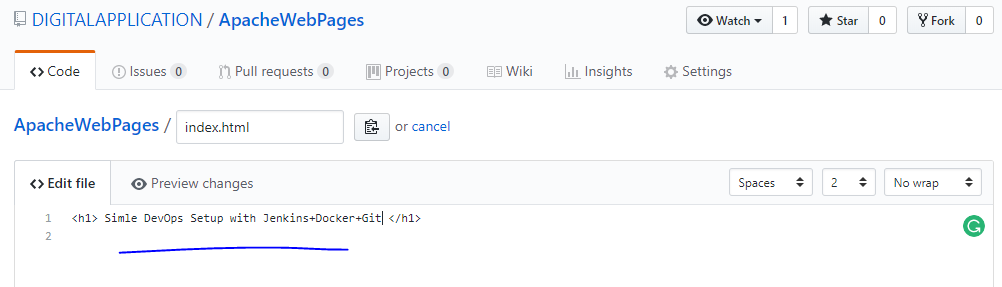
Save the Jenkins job configuration.

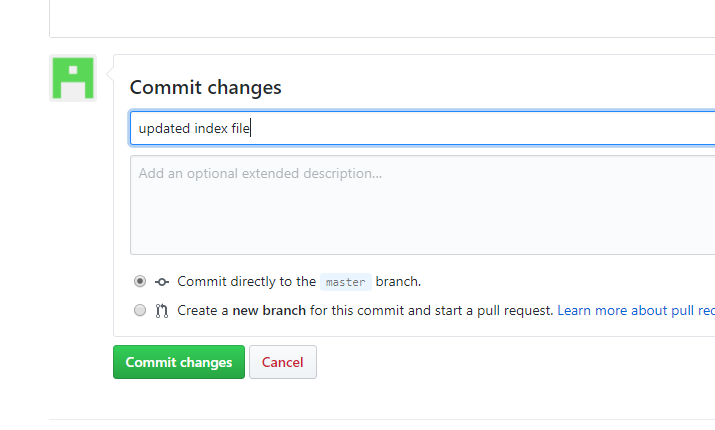
Now make some changes in “index.html” page as to trigger Jenkins job automatically.

Before that, make sure there is no docker container running on 8080 port: Better case, remove all existing images & containers.

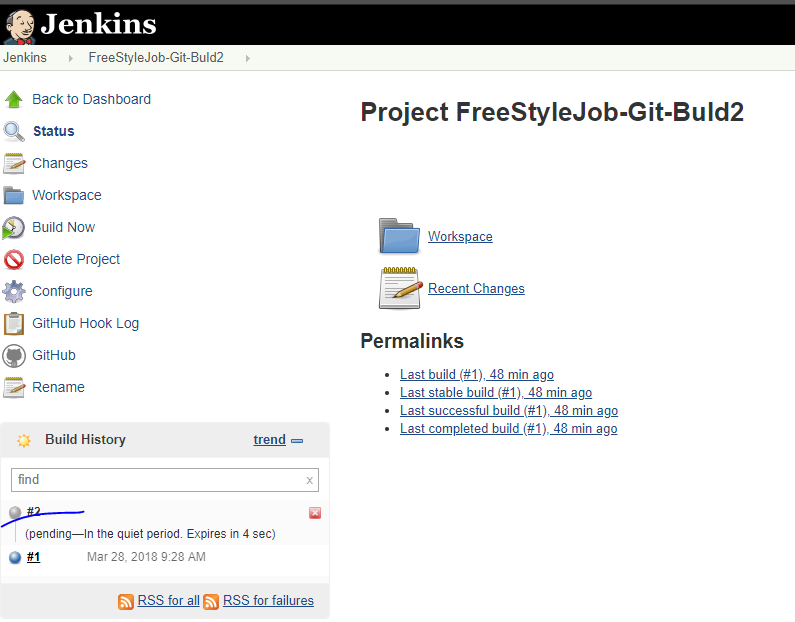


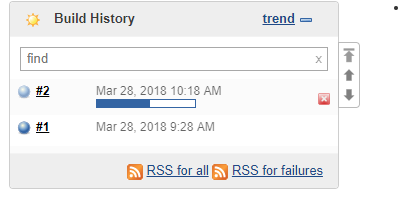
Updating the **Index.html** file over the GitHub web & commit the changes:





**Job triggered automatically** in seconds:





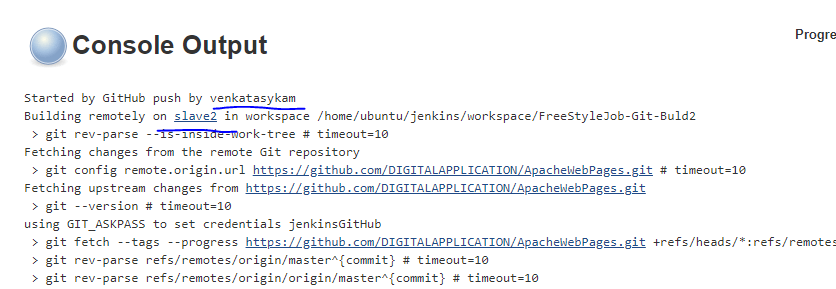
Job running on the selected slave. Here is the user name is GitHub’s username who pushed the changes. This is not the Jenkins username. Observe each step how docker is executing Dockerfile.

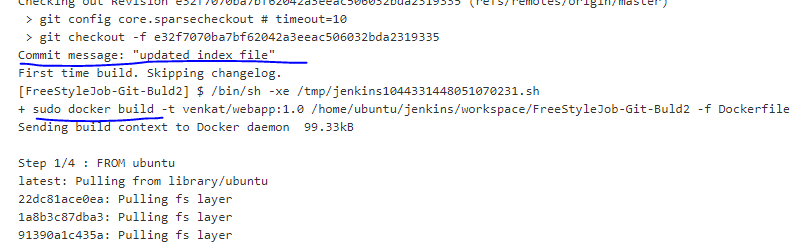
Step 1/4: FROM ubuntu

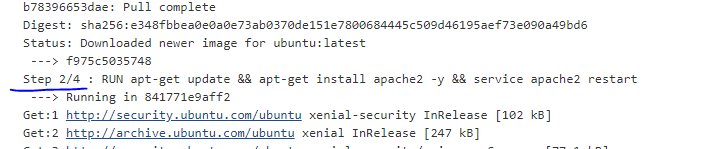
Step 2/4: RUN apt-get update && apt-get install apache2 -y && service apache2 restart

Step 3/4: ADD index.html /var/www/html/

Step 4/4: CMD apachectl -D FOREGROUND

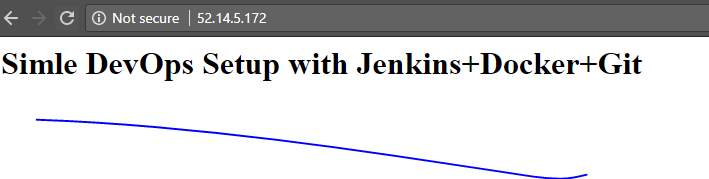






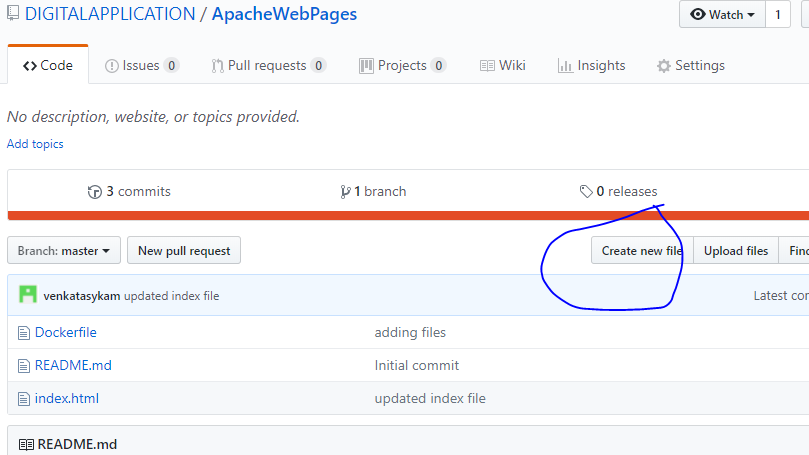


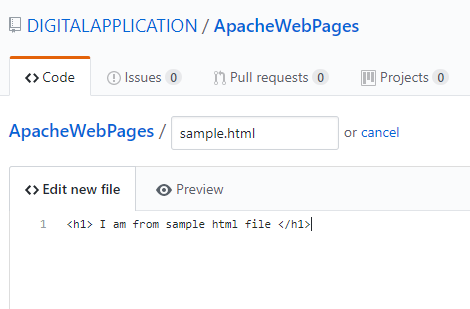
Run the application:

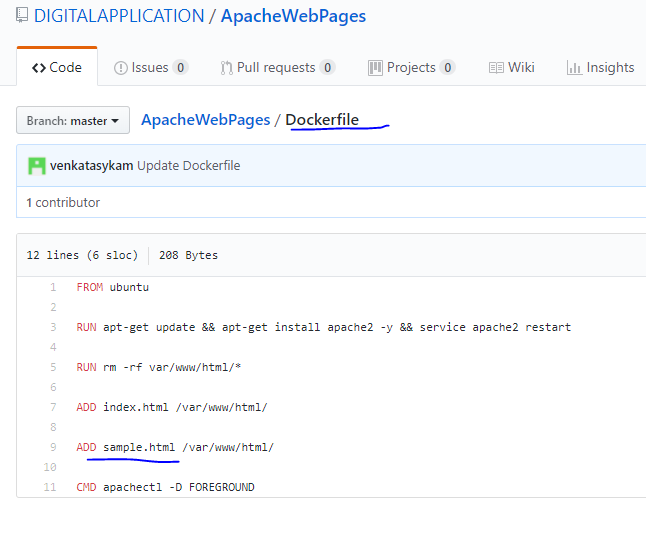


**Let’s do another exercise:**

Add a new html file with the name **sample.html**











**Output**: **Index.html** is the default file which will be detected by apache server. <http://52.14.5.172> & <http://52.14.5.172/index.html> both will run the default html file i.e., index.html.





To run the sample html file, specify the name: <http://52.14.5.172/sample.html>

