**Step 1** − Create a file called **Docker File** and edit it using **vim**. Please note that the name of the file has to be "Dockerfile" with "D" as capital.



**Step 2** − Build your Docker File using the following instructions.

#This is a sample Image

FROM ubuntu

MAINTAINER demousr@gmail.com

RUN apt-get update

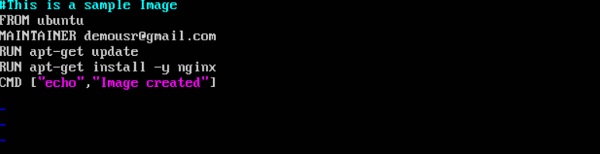
RUN apt-get install –y nginx

CMD [“echo”,”Image created”]

The following points need to be noted about the above file −

* The first line "#This is a sample Image" is a comment. You can add comments to the Docker File with the help of the **#** command
* The next line has to start with the **FROM** keyword. It tells docker, from which base image you want to base your image from. In our example, we are creating an image from the **ubuntu** image.
* The next command is the person who is going to maintain this image. Here you specify the **MAINTAINER** keyword and just mention the email ID.
* The **RUN** command is used to run instructions against the image. In our case, we first update our Ubuntu system and then install the nginx server on our **ubuntu** image.
* The last command is used to display a message to the user.

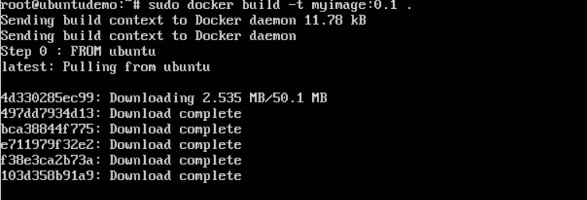
**Step 3** − Save the file. In the next chapter, we will discuss how to build the image.



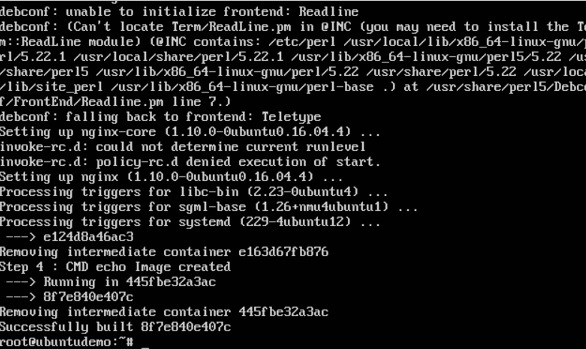
sudo docker build –t myimage:0.1

Output

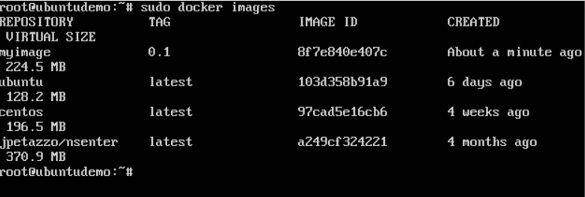
From the output, you will first see that the Ubuntu Image will be downloaded from Docker Hub, because there is no image available locally on the machine.



Finally, when the build is complete, all the necessary commands would have run on the image.



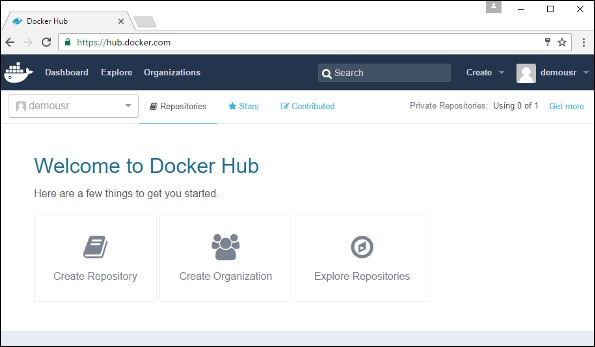
You will then see the successfully built message and the ID of the new Image. When you run the Docker **images command**, you would then be able to see your new image.



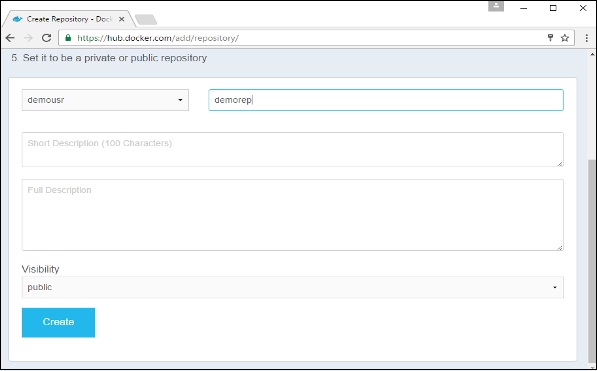
You can now build containers from your new Image.

The following steps explain how you can upload an image to public repository.

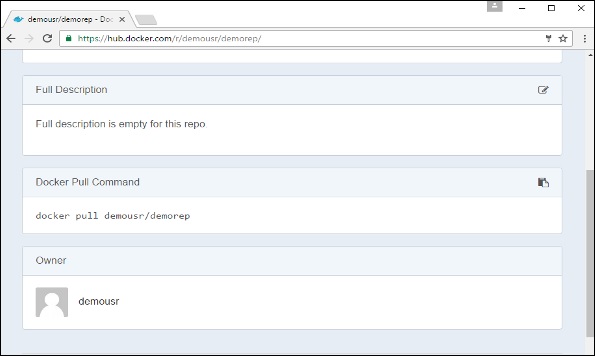
**Step 1** − Log into Docker Hub and create your repository. This is the repository where your image will be stored. Go to <https://hub.docker.com/>and log in with your credentials.



**Step 2** − Click the button "Create Repository" on the above screen and create a repository with the name **demorep**. Make sure that the visibility of the repository is public.



Once the repository is created, make a note of the **pull** command which is attached to the repository.



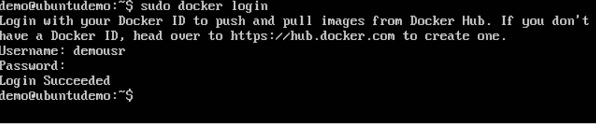
The **pull** command which will be used in our repository is as follows −

docker pull demousr/demorep

**Step 3** − Now go back to the Docker Host. Here we need to tag our **myimage**to the new repository created in Docker Hub. We can do this via the Docker **tag command**.

We will learn more about this **tag command** later in this chapter.

**Step 4** − Issue the Docker login command to login into the Docker Hub repository from the command prompt. The Docker login command will prompt you for the username and password to the Docker Hub repository.



**Step 5** − Once the image has been tagged, it’s now time to push the image to the Docker Hub repository. We can do this via the Docker **push** command. We will learn more about this command later in this chapter.

## **docker tag**

This method allows one to tag an image to the relevant repository.

### Syntax

docker tag imageID Repositoryname

### Options

* **imageID** − This is the ImageID which needs to be tagged to the repository.
* **Repositoryname** − This is the repository name to which the ImageID needs to be tagged to.

### Return Value

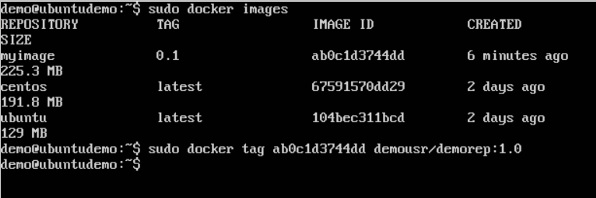
None

### Example

sudo docker tag ab0c1d3744dd demousr/demorep:1.0

### Output

A sample output of the above example is given below.



## **docker push**

This method allows one to push images to the Docker Hub.

### Syntax

docker push Repositoryname

### Options

* **Repositoryname** − This is the repository name which needs to be pushed to the Docker Hub.

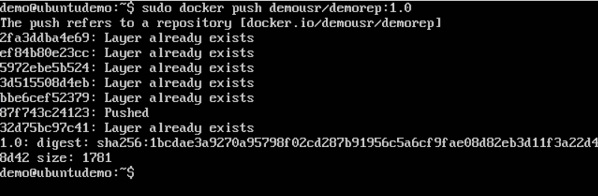
### Return Value

The long ID of the repository pushed to Docker Hub.

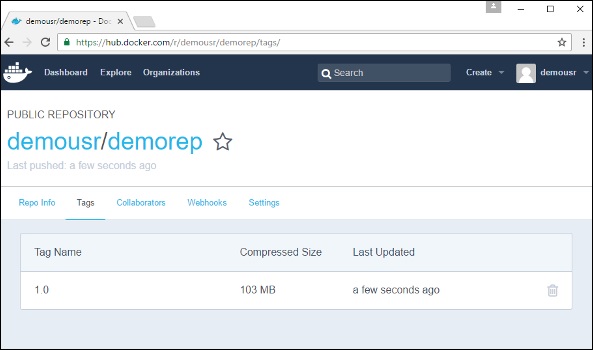
### Example

sudo docker push demousr/demorep:1.0

### Output



If you go back to the Docker Hub page and go to your repository, you will see the tag name in the repository.



Now let’s try to pull the repository we uploaded onto our Docker host. Let’s first delete the images, **myimage:0.1** and **demousr/demorep:1.0**, from the local Docker host. Let’s use the Docker **pull command** to pull the repository from the Docker Hub.



From the above screenshot, you can see that the Docker **pull** command has taken our new repository from the Docker Hub and placed it on our machine.