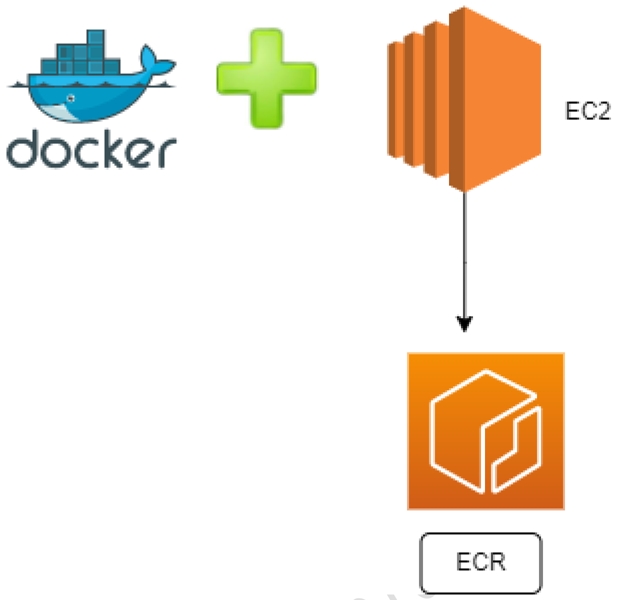
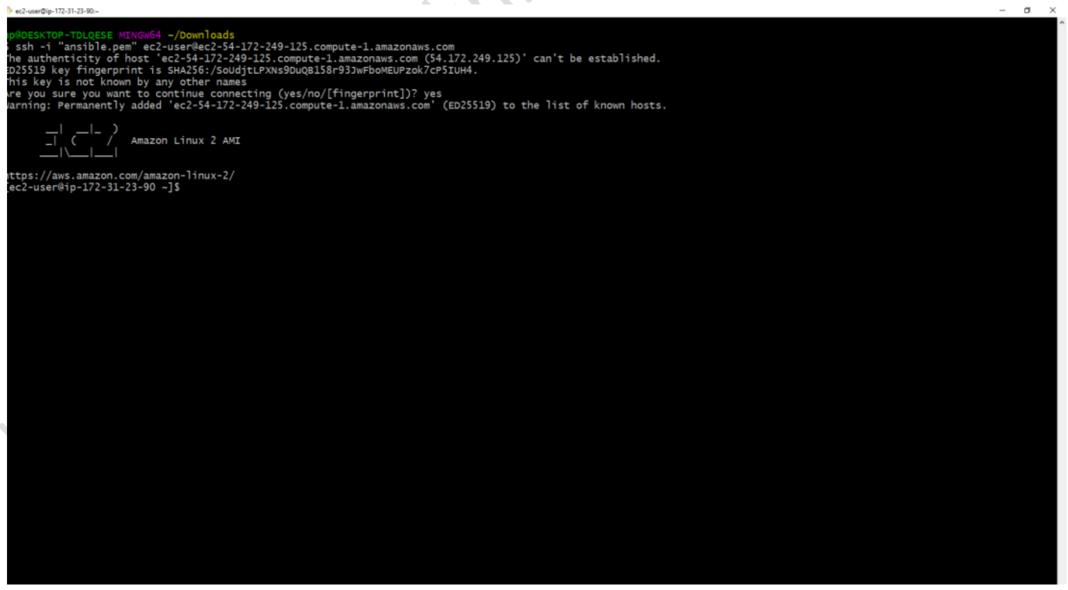
BUILD DOCKER IMAGE IN EC2 AND PUSH TO AWS ECR

https://www.linkedin.com/in/aziz-afoni-811503227/



1. Launch an EC2 Linux instance
   - Select t2.micro
   - Use default security group
2. SSH into your instance



```
ec2-user@ip-172-31-23-90:~

p@DESKTOP-TDLQESE MINGW64 ~/Downloads
 ssh -i "ansible.pem" ec2-user@ec2-54-172-249-125.compute-1.amazonaws.com
he authenticity of host 'ec2-54-172-249-125.compute-1.amazonaws.com (54.172.249.125)' can't be established.
D25519 key fingerprint is SHA256:/SoUdjtLPXNs9DuQB158r93JwFboMEUPzok7cP5IUH4.
his key is not known by any other names
re you sure you want to continue connecting (yes/no/[fingerprint])? yes
arning: Permanently added 'ec2-54-172-249-125.compute-1.amazonaws.com' (ED25519) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

ttps://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-23-90 ~]$
```

3. Update the installed packages and package cache on your instance.
   sudo yum update -y

4. Install the most recent Docker Engine package.

   Amazon Linux 2

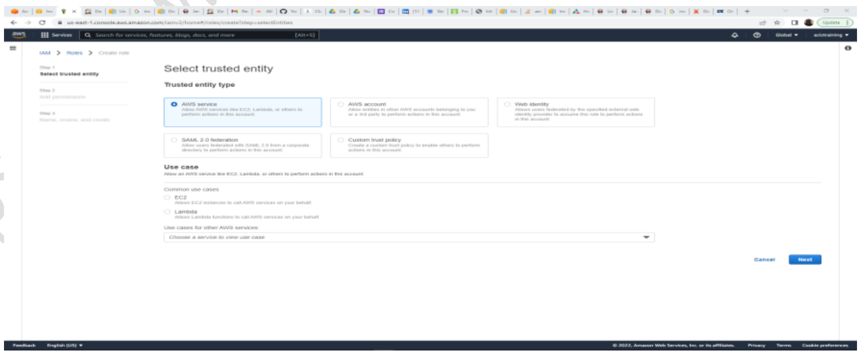
   sudo amazon-linux-extras install docker
5. Start the Docker service.
   sudo service docker start
6. Add the `ec2-user` to the `docker` group so you can execute Docker commands without using `sudo`.
   sudo usermod -a -G docker ec2-user

N: B In some cases, you may need to reboot your instance to provide permissions for the `ec2-user` to access the Docker daemon. Try rebooting your instance if you see the following error:

```
Cannot connect to the Docker daemon. Is the docker daemon running on this host?
```

7. To have your ec2 to push docker images to ECR you need to create an IAM role for ec2 and attach

8. Navigate to iam and click on roles
   - Click create role
   -



us-east-1.console.aws.amazon.com/iamv2/home#/roles/create?step=selectEntities

IAM > Roles > Create role

Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create

Select trusted entity

Trusted entity type

- AWS service
  Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account
  Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- Web identity
  Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- SAML 2.0 federation
  Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy
  Create a custom trust policy to enable others to perform actions in this account.

Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases
- EC2
  Allows EC2 instances to call AWS services on your behalf.
- Lambda
  Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:
Choose a service to view use case

Services
Search for services, features, blogs, docs, and more [Alt+S]
Global
azizraining
Cancel
Next
Feedback
English (US)
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

   - Select ec2 and click next
   - Search container and select the first role
     AmazonEC2ContainerRegistryFullAccess

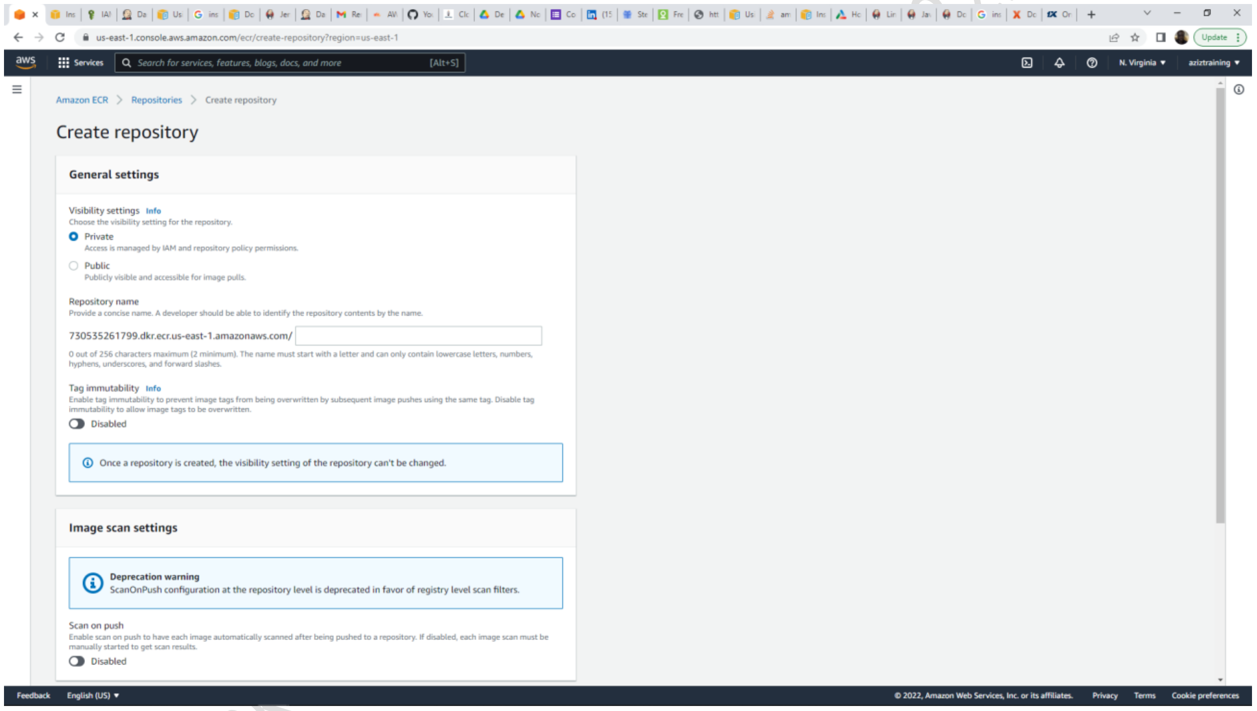
- Click next give a name to the role and click create role

- Navigate back to ec2 and select the instance

- Click on actions >> instance settings >> attach iam role

- Click the dropdown and select the role you created then apply

- Now navigate to ECR and click on create repo



us-east-1.console.aws.amazon.com/ecr/create-repository?region=us-east-1
Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
aziztraining
Amazon ECR > Repositories > Create repository
Feedback
English (US)
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Create repository

General settings

Visibility settings Info
Choose the visibility setting for the repository.
[✓ Private]
Access is managed by IAM and repository policy permissions.
Public
Publicly visible and accessible for image pulls.

Repository name
Provide a concise name. A developer should be able to identify the repository contents by the name.
730535261799.dkr.ecr.us-east-1.amazonaws.com/
0 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, hyphens, underscores, and forward slashes.

Tag immutability Info
Enable tag immutability to prevent image tags from being overwritten by subsequent image pushes using the same tag. Disable tag immutability to allow image tags to be overwritten.
Disabled

Once a repository is created, the visibility setting of the repository can't be changed.

Image scan settings

Deprecation warning
ScanOnPush configuration at the repository level is deprecated in favor of registry level scan filters.

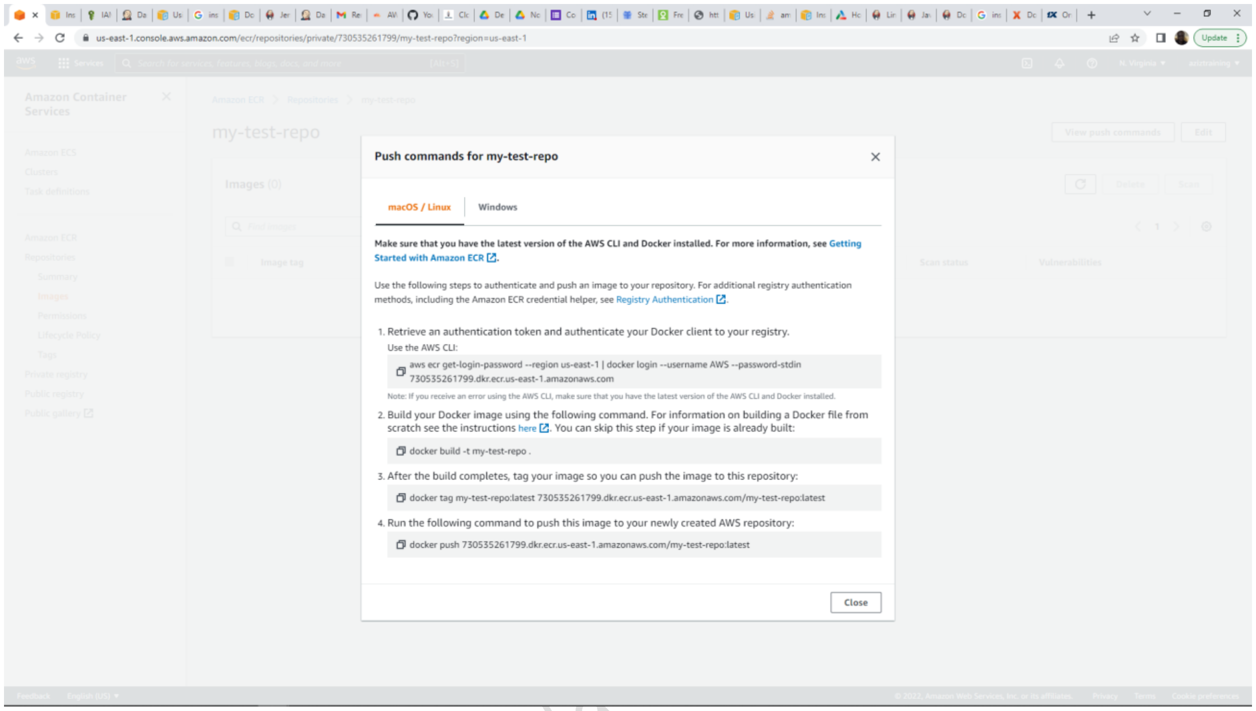
Scan on push
Enable scan on push to have each image automatically scanned after being pushed to a repository. If disabled, each image scan must be manually started to get scan results.
Disabled

-

- Provide the repository name for this example I am using my-test-repo

- Leave everything default and click on create repo

- Now navigate back to repositories and click the repo name then con the top right click view push commands



Push commands for my-test-repo

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see Getting Started with Amazon ECR.

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see Registry Authentication.

1. Retrieve an authentication token and authenticate your Docker client to your registry.
   Use the AWS CLI:
   ```
   aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 730535261799.dkr.ecr.us-east-1.amazonaws.com
   ```
   Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:
   ```
   docker build -t my-test-repo .
   ```
3. After the build completes, tag your image so you can push the image to this repository:
   ```
   docker tag my-test-repo:latest 730535261799.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest
   ```
4. Run the following command to push this image to your newly created AWS repository:
   ```
   docker push 730535261799.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest
   ```

- Copy the first command and go back to your ec2 terminal

  aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com

  where xxxxxxxx is account id

- You should see the following

```
WARNING! Your password will be stored unencrypted in /home/ec2-user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

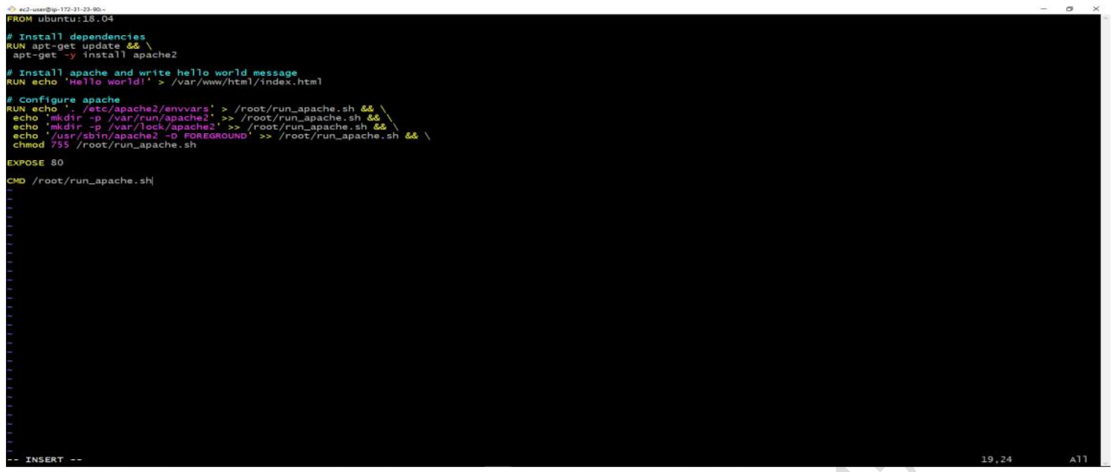
- Now let us create a simple Dockerfile

```
 FROM ubuntu:18.04

# Install dependencies
RUN apt-get update && \
 apt-get -y install apache2

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo '. /etc/apache2/envvars' > /root/run_apache.sh && \
 echo 'mkdir -p /var/run/apache2' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/apache2' >> /root/run_apache.sh && \
 echo '/usr/sbin/apache2 -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

         CMD /root/run_apache.sh
```

- This Dockerfile uses the Ubuntu 18.04 image. The `RUN` instructions update the package caches, install some software packages for the web server, and then write the "Hello World!" content to the web server's document root.
  The `EXPOSE` instruction exposes port 80 on the container, and the `CMD` instruction starts the web server.

- In your terminal create a Dockerfile using
  Touch Dockerfile
- Open the Dockerfile using vi editor and paste the above dockerfile specifications

-



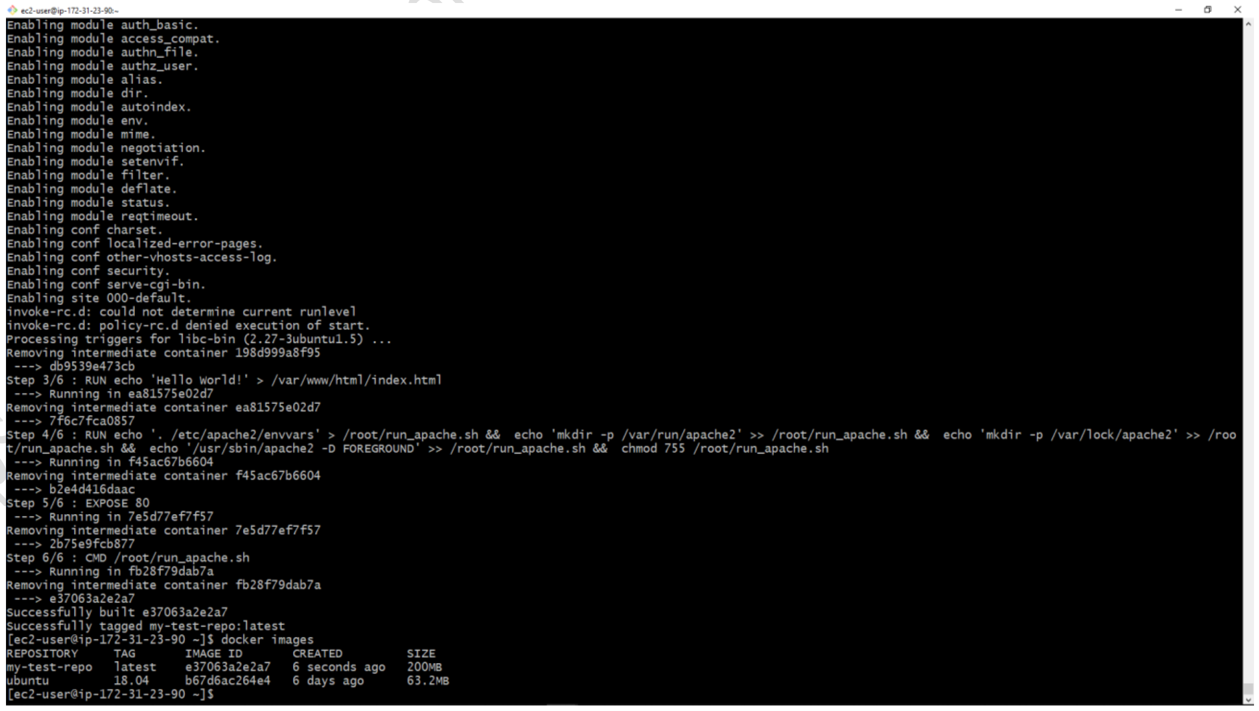
```
FROM ubuntu:18.04

# Install dependencies
RUN apt-get update && \
 apt-get -y install apache2

# Install apache and write hello world message
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure apache
RUN echo '. /etc/apache2/envvars' > /root/run_apache.sh && \
 echo 'mkdir -p /var/run/apache2' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/apache2' >> /root/run_apache.sh && \
 echo '/usr/sbin/apache2 -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
~
-- INSERT --                                                    19,24         All
```

- Save and quit using
  :wq

- Now run the docker build command from the push commands listed in ECR
  docker build -t my-test-repo .

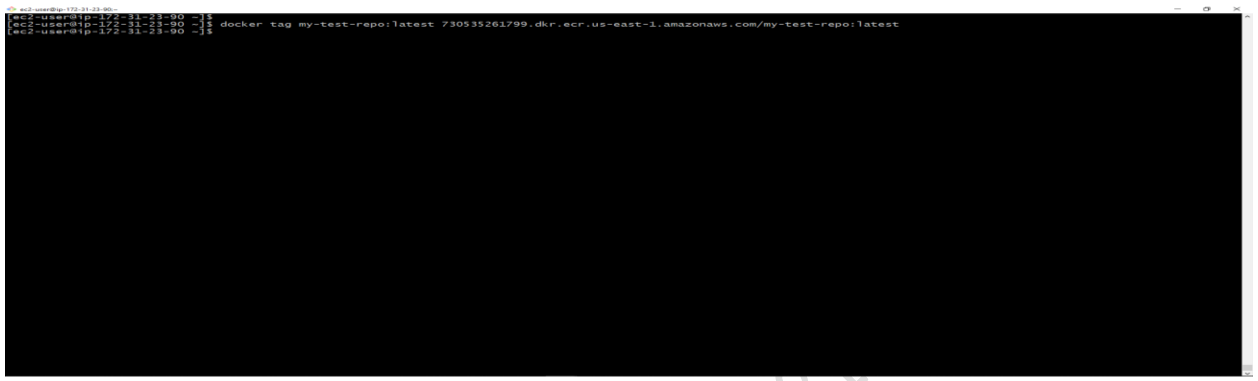
  this command builds a docker image named my-test-repo from the docker file

- Now run docker images to see if the image has been created



```
Enabling module auth_basic.
Enabling module access_compat.
Enabling module authn_file.
Enabling module authz_user.
Enabling module alias.
Enabling module dir.
Enabling module autoindex.
Enabling module env.
Enabling module mime.
Enabling module negotiation.
Enabling module setenvif.
Enabling module filter.
Enabling module deflate.
Enabling module status.
Enabling module reqtimeout.
Enabling conf charset.
Enabling conf localized-error-pages.
Enabling conf other-vhosts-access-log.
Enabling conf security.
Enabling conf serve-cgi-bin.
Enabling site 000-default.
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of start.
Processing triggers for libc-bin (2.27-3ubuntu1.5) ...
Removing intermediate container 198d999a8f95
 ---> db9539e473cb
Step 3/6 : RUN echo 'Hello World!' > /var/www/html/index.html
 ---> Running in ea81575e02d7
Removing intermediate container ea81575e02d7
 ---> 7f6c7fca0857
Step 4/6 : RUN echo '. /etc/apache2/envvars' > /root/run_apache.sh &&  echo 'mkdir -p /var/run/apache2' >> /root/run_apache.sh &&  echo 'mkdir -p /var/lock/apache2' >> /roo
t/run_apache.sh &&  echo '/usr/sbin/apache2 -D FOREGROUND' >> /root/run_apache.sh &&  chmod 755 /root/run_apache.sh
 ---> Running in f45ac67b6604
Removing intermediate container f45ac67b6604
 ---> b2e4d416daac
Step 5/6 : EXPOSE 80
 ---> Running in 7e5d77ef7f57
Removing intermediate container 7e5d77ef7f57
 ---> 2b75e9fcb877
Step 6/6 : CMD /root/run_apache.sh
 ---> Running in fb28f79dab7a
Removing intermediate container fb28f79dab7a
 ---> e37063a2e2a7
Successfully built e37063a2e2a7
Successfully tagged my-test-repo:latest
[ec2-user@ip-172-31-23-90 ~]$ docker images
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
my-test-repo   latest    e37063a2e2a7   6 seconds ago   200MB
ubuntu         18.04     b67d6ac264e4   6 days ago      63.2MB
[ec2-user@ip-172-31-23-90 ~]$
```

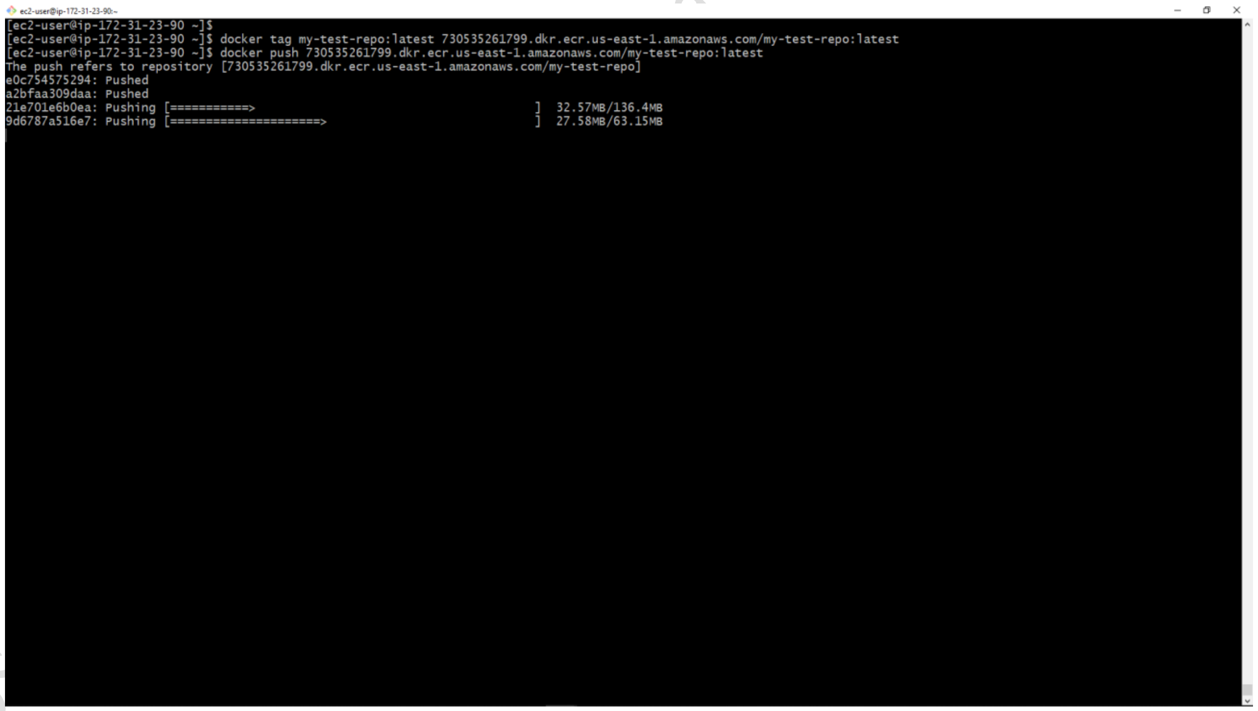
- Now tag the image using the third command from the docker push commands in ECR docker tag my-test-repo:latest xxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest

-


```
[ec2-user@ip-172-31-23-90 ~]$
[ec2-user@ip-172-31-23-90 ~]$ docker tag my-test-repo:latest 730535261799.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest
[ec2-user@ip-172-31-23-90 ~]$
```

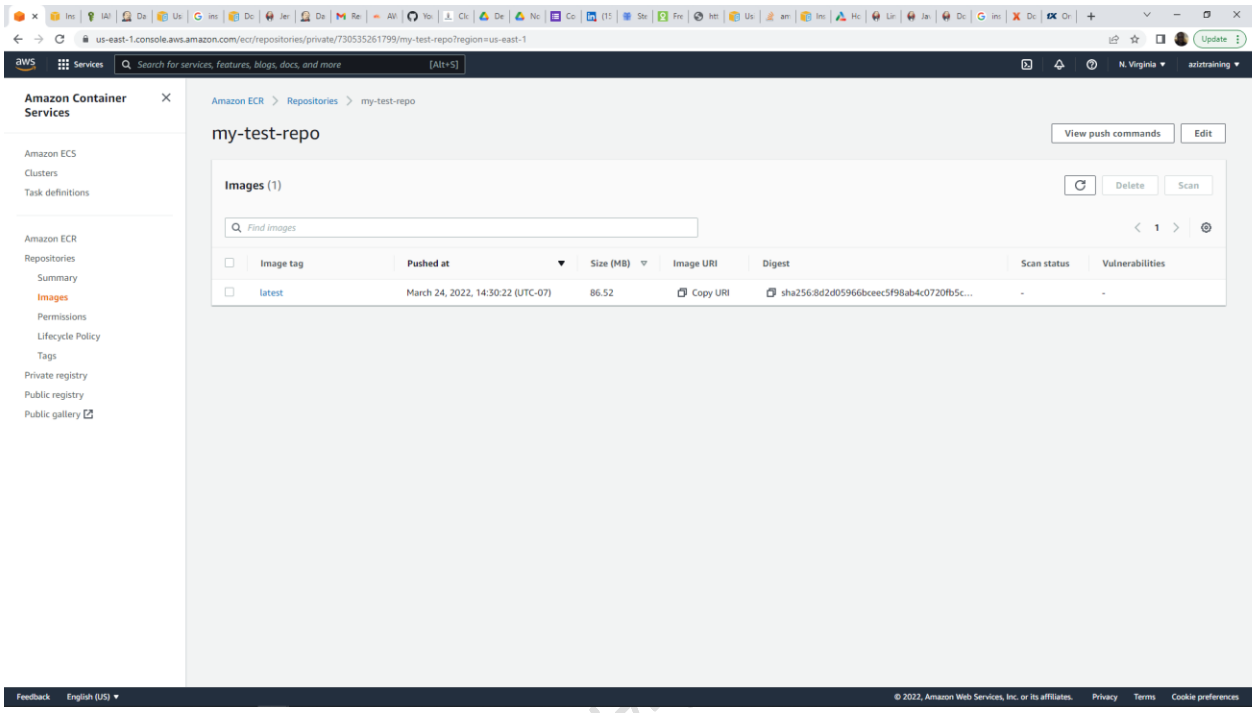
- Now push the image using the fourth command from the docker push commands

  docker push xxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest



```
[ec2-user@ip-172-31-23-90 ~]$
[ec2-user@ip-172-31-23-90 ~]$ docker tag my-test-repo:latest 730535261799.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest
[ec2-user@ip-172-31-23-90 ~]$ docker push 730535261799.dkr.ecr.us-east-1.amazonaws.com/my-test-repo:latest
The push refers to repository [730535261799.dkr.ecr.us-east-1.amazonaws.com/my-test-repo]
e0c754575294: Pushed
a2bfaa309daa: Pushed
21e701e6b0ea: Pushing [===========>                                       ]  32.57MB/136.4MB
9d6787a516e7: Pushing [=====================>                             ]  27.58MB/63.15MB
```

- Go back to the repo and open the repo you should have your mage ready



us-east-1.console.aws.amazon.com/ecr/repositories/private/730535261799/my-test-repo?region=us-east-1

my-test-repo

Images (1)

Image tag	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
latest	March 24, 2022, 14:30:22 (UTC-07)	86.52	Copy URI	sha256:8d2d05966bceec5f98ab4c0720fb5c...	-	-

-

Thank you for following this tutorial in the next tutorial we will run the image on AWS ECS