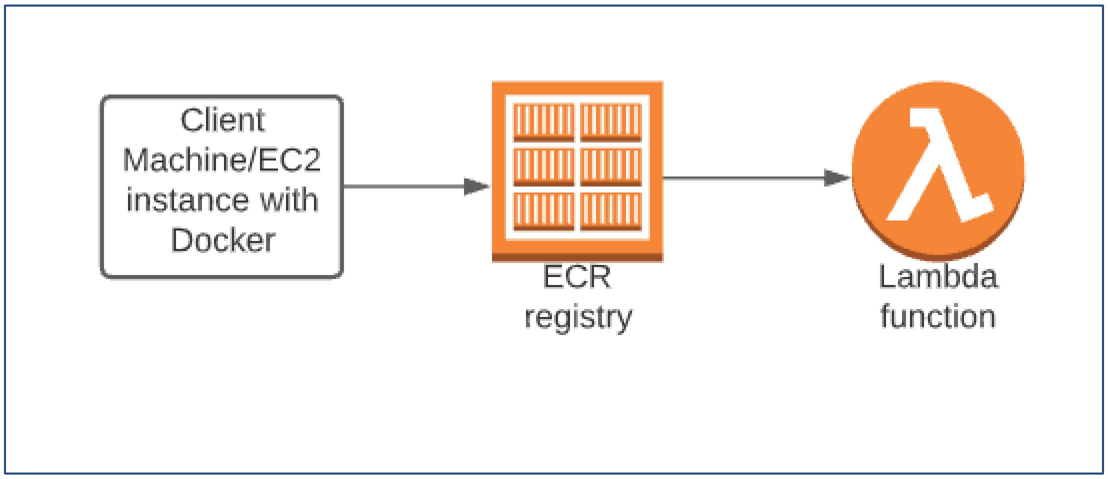
Dr. Shubhangi Kharche

AWS Project 4

Scenario

The introduction of Lambda support for OCI container images provides customers with more choices when it comes to packaging formats. Developers can now choose to take advantage of the event-driven runtime model and cost-savings advantages of AWS Lambda, while taking advantage of the predictability and control offered by a container-based development and deployment cycle.

Architecture diagram



Architecture Implementation	
1	Download the Dockerfile and the app code folder provided with this workbook
2	Package the web application as a Docker image running on Alpine with Python
3	Create an ECR repository and login to it.
4	Build the image with the downloaded dockerfile and the support files
5	Tag the image appropriately and push it into the ECR repository.
6	Create a Lambda function with the image in ECR.

Step 1 : Docker Image creation

Step number a

Step name Creation of Docker image

Instructions

1) Create an EC2 instance using the Amazon Linux 2 AMI in the default VPC.

2) Attach the role “LabInstanceProfile” to the instance created above

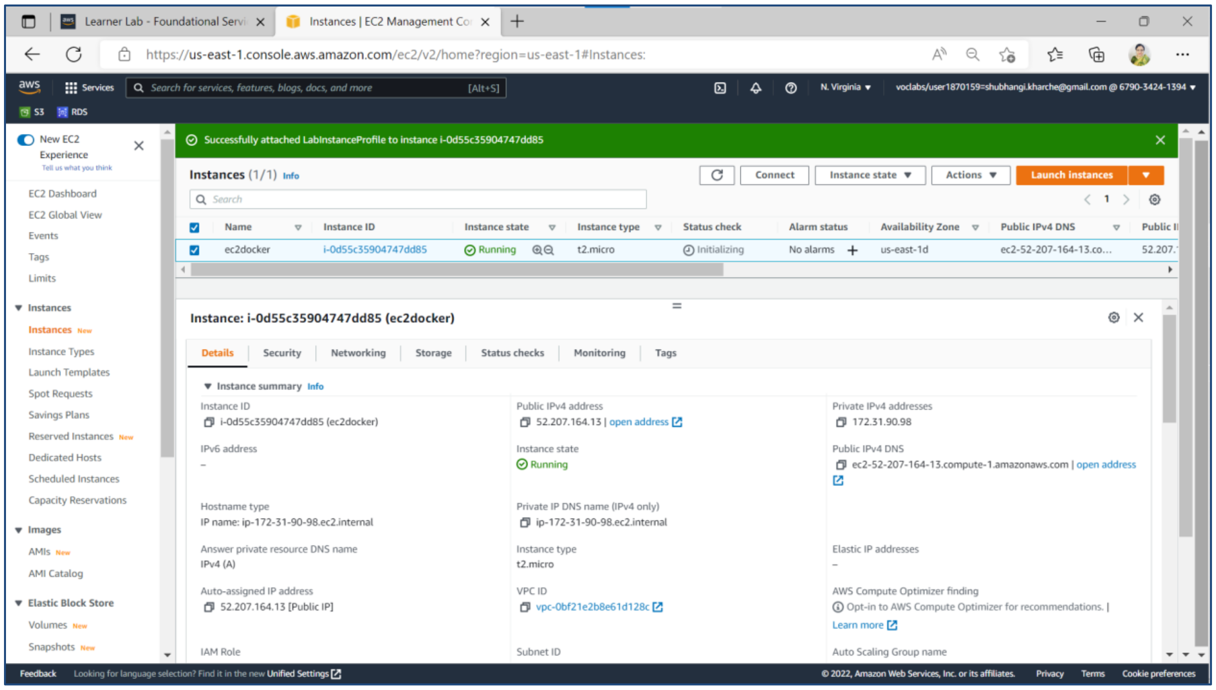
3) Download the file OCI.zip provided with this workbook and copy it to the EC2 instance using the scp command
scp -i <pem file name> ./OCI.zip ec2-user@<public IP of instance>:/home/ec2-user
(Ensure that the file OCI.zip and the pem file are in the same folder before running this command)

4) Login to the instance using SSH and run the following commands to set up the environment
sudo yum update
sudo yum install unzip
sudo unzip OCI.zip
sudo amazon-linux-extras install docker
sudo service docker start
sudo usermod -a -G docker ec2-user
(At this point, log out of the instance and log in again to ensure that the above command works. Then continue with the rest of the commands)
sudo yum install awscli -y
aws configure
Skip the access key and secret access key fields by pressing the Enter key. Enter the region as
us-east-1 and format as json

5) Run the below command to create the Docker image
docker build -t lambda_ecr .

6) Run the below command to verify the creation of the image
docker images

Expected screenshots 1)Building the Docker image 3) List of the created image



https://us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:

Successfully attached LabInstanceProfile to instance i-0d55c35904747dd85

Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public I
ec2docker	i-0d55c35904747dd85	Running	t2.micro	Initializing	No alarms	us-east-1d	ec2-52-207-164-13.co...	52.207.

Instance: i-0d55c35904747dd85 (ec2docker)

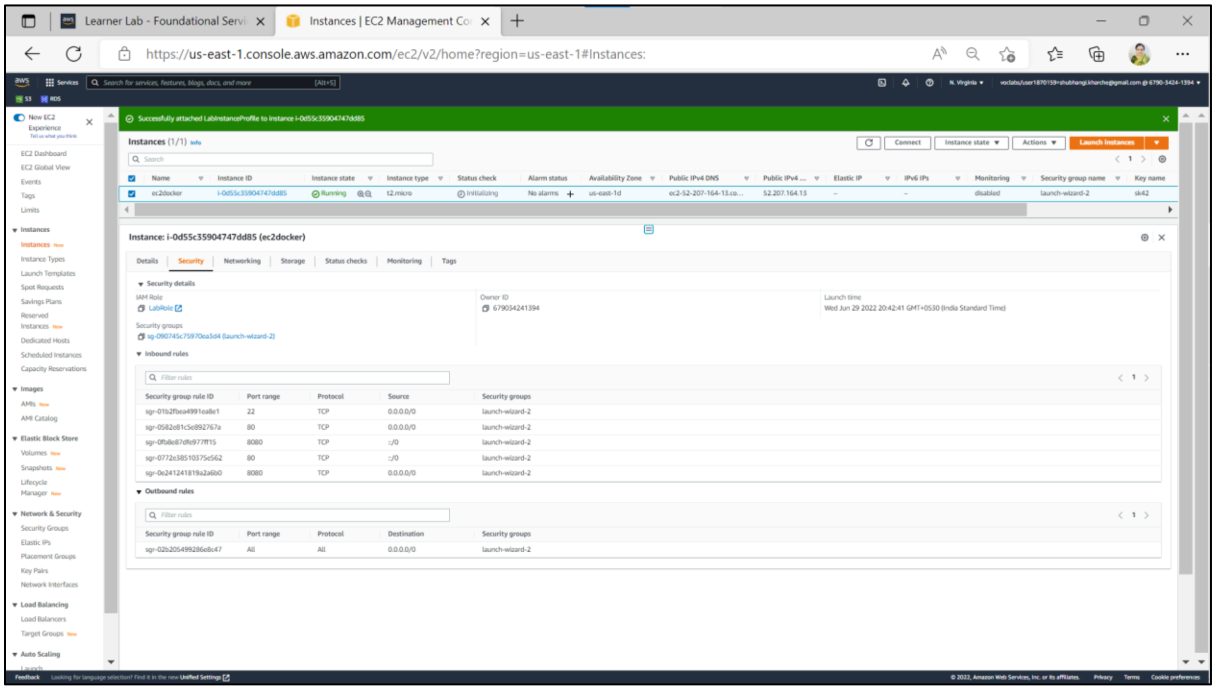
Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance summary Info

Instance ID i-0d55c35904747dd85 (ec2docker)	Public IPv4 address 52.207.164.13 | open address	Private IPv4 addresses 172.31.90.98
IPv6 address –	Instance state Running	Public IPv4 DNS ec2-52-207-164-13.compute-1.amazonaws.com | open address
Hostname type IP name: ip-172-31-90-98.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-90-98.ec2.internal	
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	Elastic IP addresses –
Auto-assigned IP address 52.207.164.13 [Public IP]	VPC ID vpc-0bf21e2b8e61d128c	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role	Subnet ID	Auto Scaling Group name

© 2022, Amazon Web Services, Inc. or its affiliates.

Fig 1.a Creation of EC2 instance (optional screenshot)



https://us-east-1.console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:

Successfully attached LabInstanceProfile to instance i-0d55c35904747dd85

Instances (1/1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name	Key name
ec2docker	i-0d55c35904747dd85	Running	t2.micro	Initializing	No alarms	us-east-1d	ec2-52-207-164-13.co...	52.207.164.13	–	–	disabled	launch-wizard-2	sk42

Instance: i-0d55c35904747dd85 (ec2docker)

Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Security details

IAM Role LabRole	Owner ID 679034241394	Launch time Wed Jun 29 2022 20:42:41 GMT+0530 (India Standard Time)
Security groups sg-090745c75970ea3d4 (launch-wizard-2)		

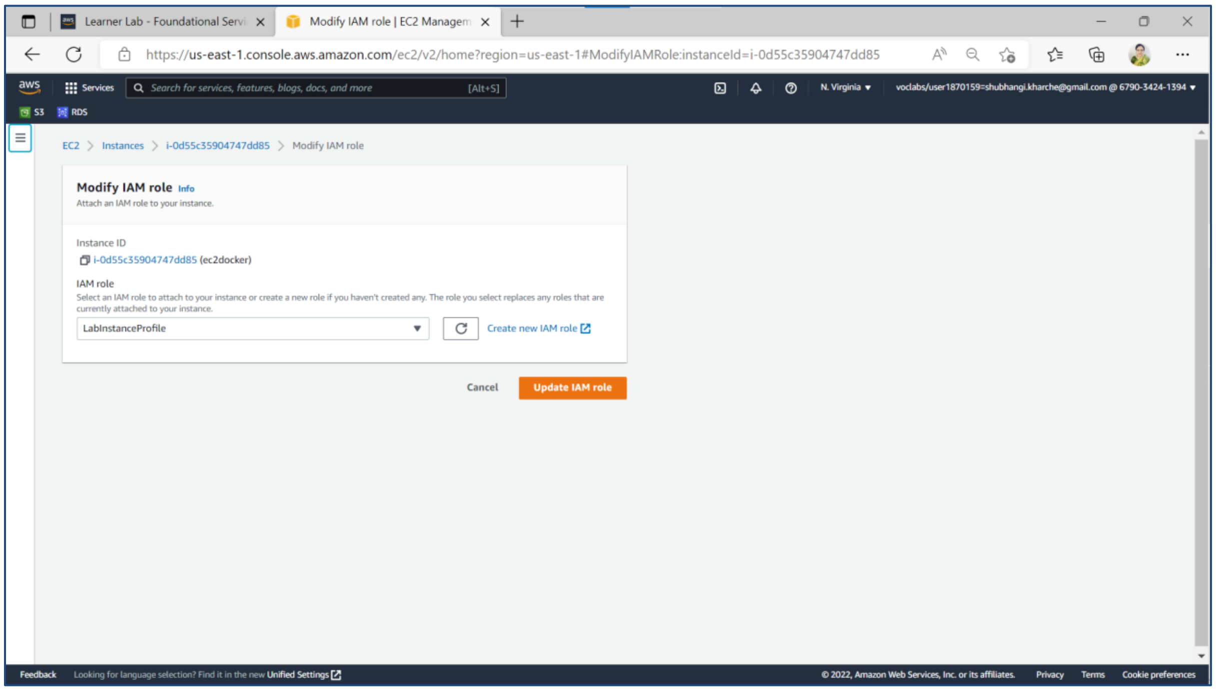
Inbound rules

Security group rule ID	Port range	Protocol	Source	Security groups
sgr-01b2fbea4991ea8e1	22	TCP	0.0.0.0/0	launch-wizard-2
sgr-0582e81c5e892767a	80	TCP	0.0.0.0/0	launch-wizard-2
sgr-0fb8e87dfe977ff15	8080	TCP	::/0	launch-wizard-2
sgr-0772e38510375e562	80	TCP	::/0	launch-wizard-2
sgr-0e241241819a2a6b0	8080	TCP	0.0.0.0/0	launch-wizard-2

Outbound rules

Security group rule ID	Port range	Protocol	Destination	Security groups
sgr-02b205499286e8c47	All	All	0.0.0.0/0	launch-wizard-2

Fig 1.b Defining security groups [SSH port 22 (from anywhere using IPv4), HTTP port 80 (from anywhere using both IPv4 and IPv6), docker host port 8080 (from anywhere using both IPv4 and IPv6)] (optional screenshot)



EC2 > Instances > i-0d55c35904747dd85 > Modify IAM role

Modify IAM role Info

Attach an IAM role to your instance.

Instance ID

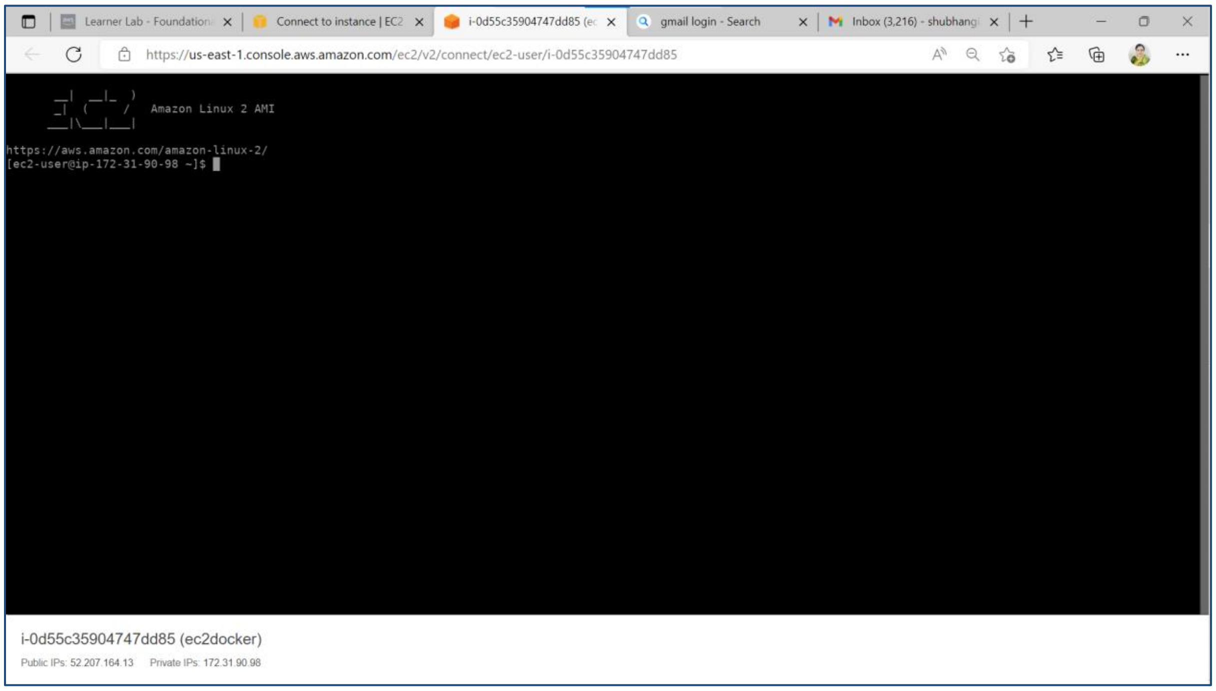
i-0d55c35904747dd85 (ec2docker)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

LabInstanceProfile

Fig 1.c Updation of IAM Role (optional screenshot)

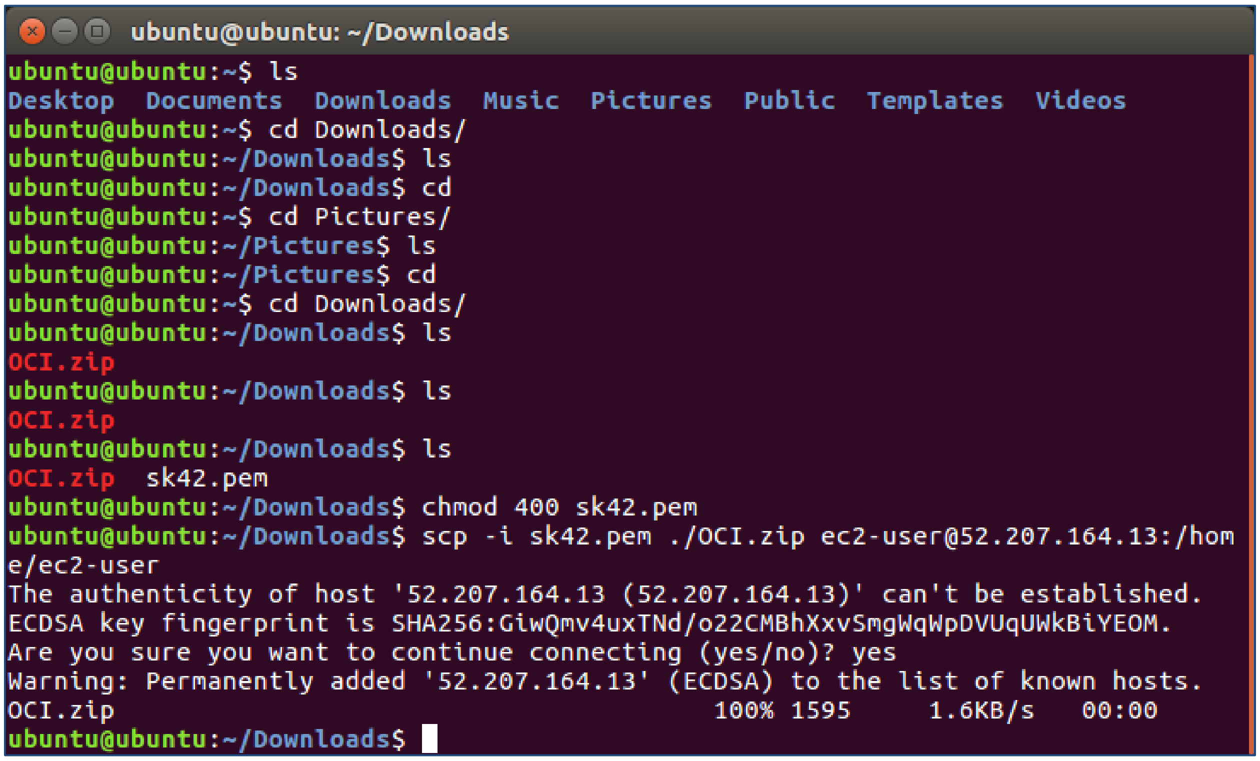


```
       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-90-98 ~]$ 
```

i-0d55c35904747dd85 (ec2docker)

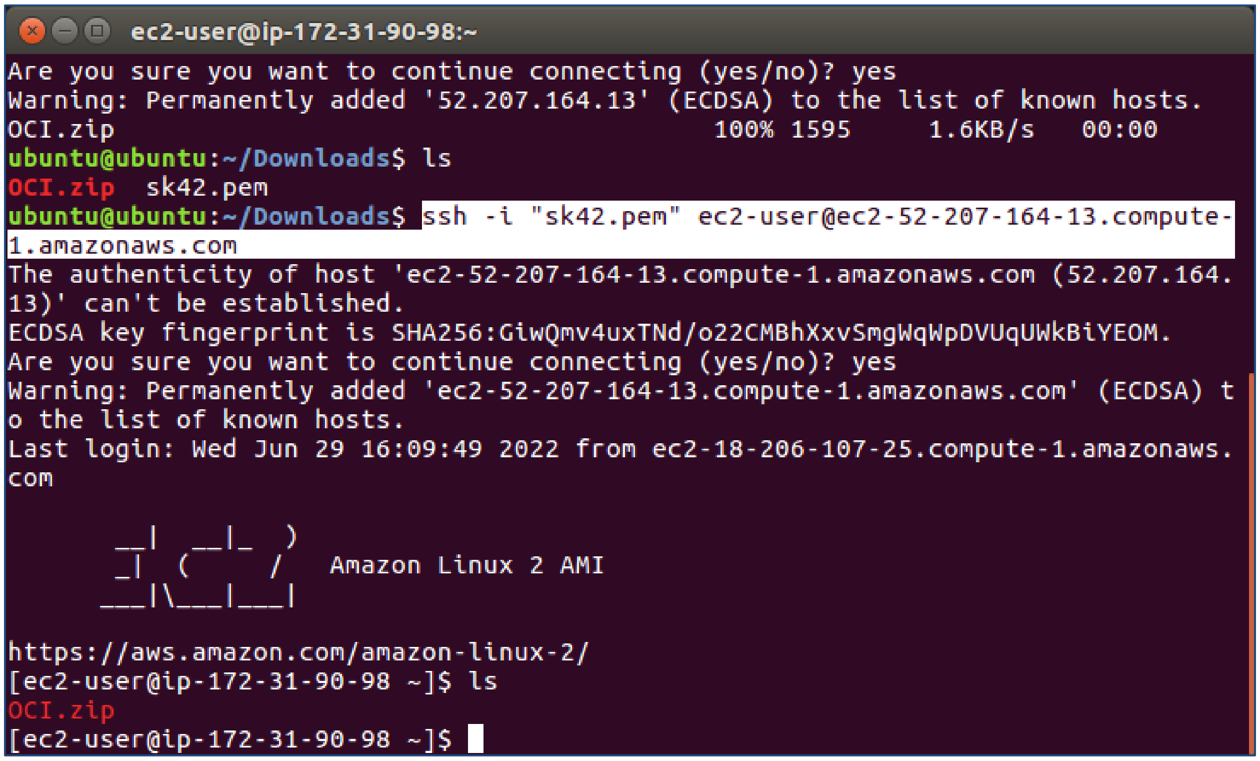
Public IPs: 52.207.164.13 Private IPs: 172.31.90.98

Fig 1.d Successful login to created EC2 instance (optional screenshot)



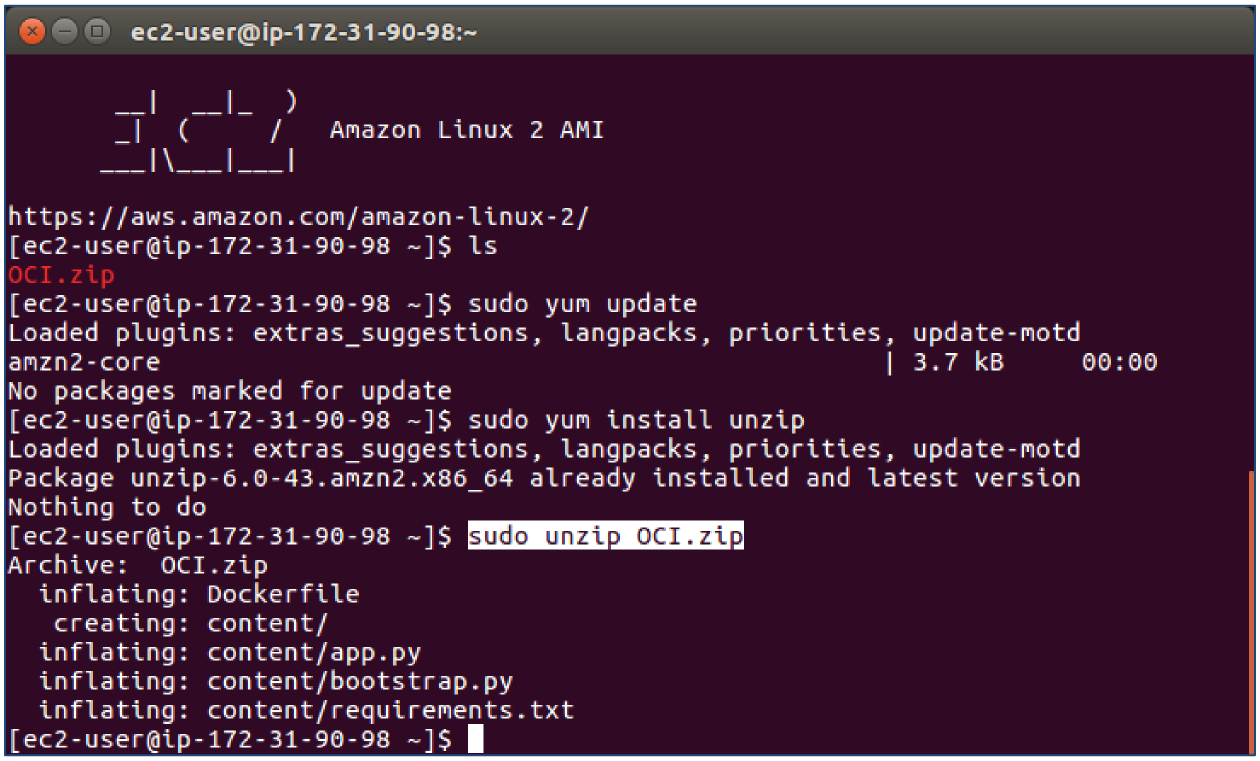
```
ubuntu@ubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
ubuntu@ubuntu:~$ cd Downloads/
ubuntu@ubuntu:~/Downloads$ ls
ubuntu@ubuntu:~/Downloads$ cd
ubuntu@ubuntu:~$ cd Pictures/
ubuntu@ubuntu:~/Pictures$ ls
ubuntu@ubuntu:~/Pictures$ cd
ubuntu@ubuntu:~$ cd Downloads/
ubuntu@ubuntu:~/Downloads$ ls
OCI.zip
ubuntu@ubuntu:~/Downloads$ ls
OCI.zip
ubuntu@ubuntu:~/Downloads$ ls
OCI.zip  sk42.pem
ubuntu@ubuntu:~/Downloads$ chmod 400 sk42.pem
ubuntu@ubuntu:~/Downloads$ scp -i sk42.pem ./OCI.zip ec2-user@52.207.164.13:/hom
e/ec2-user
The authenticity of host '52.207.164.13 (52.207.164.13)' can't be established.
ECDSA key fingerprint is SHA256:GiwQmv4uxTNd/o22CMBhXxvSmgWqWpDVUqUWkBiYEOM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.207.164.13' (ECDSA) to the list of known hosts.
OCI.zip                                       100% 1595     1.6KB/s   00:00
ubuntu@ubuntu:~/Downloads$ 
```

Fig 1.e Securely copying OCI.zip to created EC2 instance (optional screenshot) (assigned read permission to .pem file)



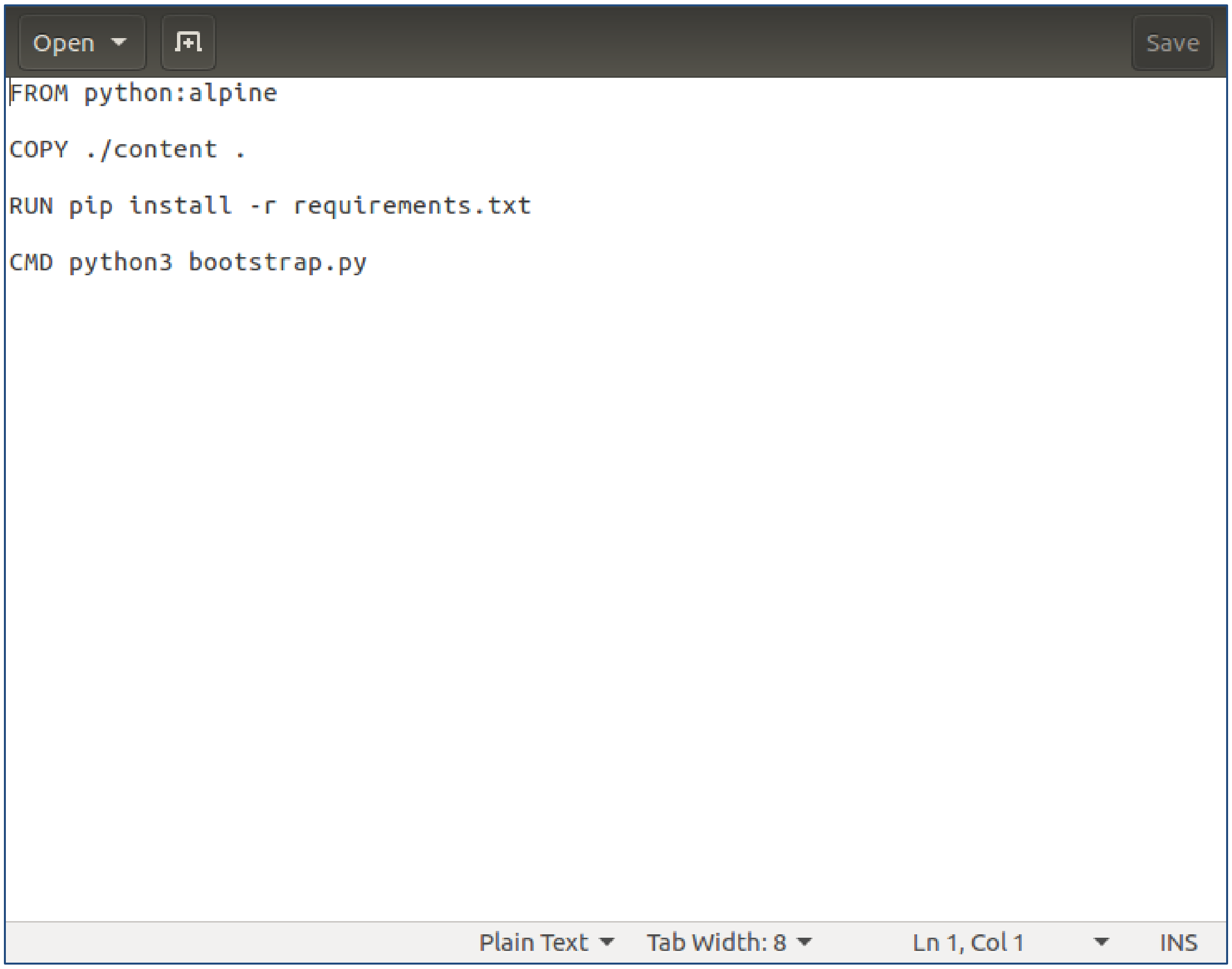
```
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '52.207.164.13' (ECDSA) to the list of known hosts.
OCI.zip                                       100% 1595     1.6KB/s   00:00
ubuntu@ubuntu:~/Downloads$ ls
OCI.zip  sk42.pem
ubuntu@ubuntu:~/Downloads$ ssh -i "sk42.pem" ec2-user@ec2-52-207-164-13.compute-
1.amazonaws.com
The authenticity of host 'ec2-52-207-164-13.compute-1.amazonaws.com (52.207.164.
13)' can't be established.
ECDSA key fingerprint is SHA256:GiwQmv4uxTNd/o22CMBhXxvSmgWqWpDVUqUWkBiYEOM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-52-207-164-13.compute-1.amazonaws.com' (ECDSA) t
o the list of known hosts.
Last login: Wed Jun 29 16:09:49 2022 from ec2-18-206-107-25.compute-1.amazonaws.
com

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-90-98 ~]$ ls
OCI.zip
[ec2-user@ip-172-31-90-98 ~]$ 
```

Fig 1.f Successful SSH login to created EC2 instance from ubuntu terminal (optional screenshot)



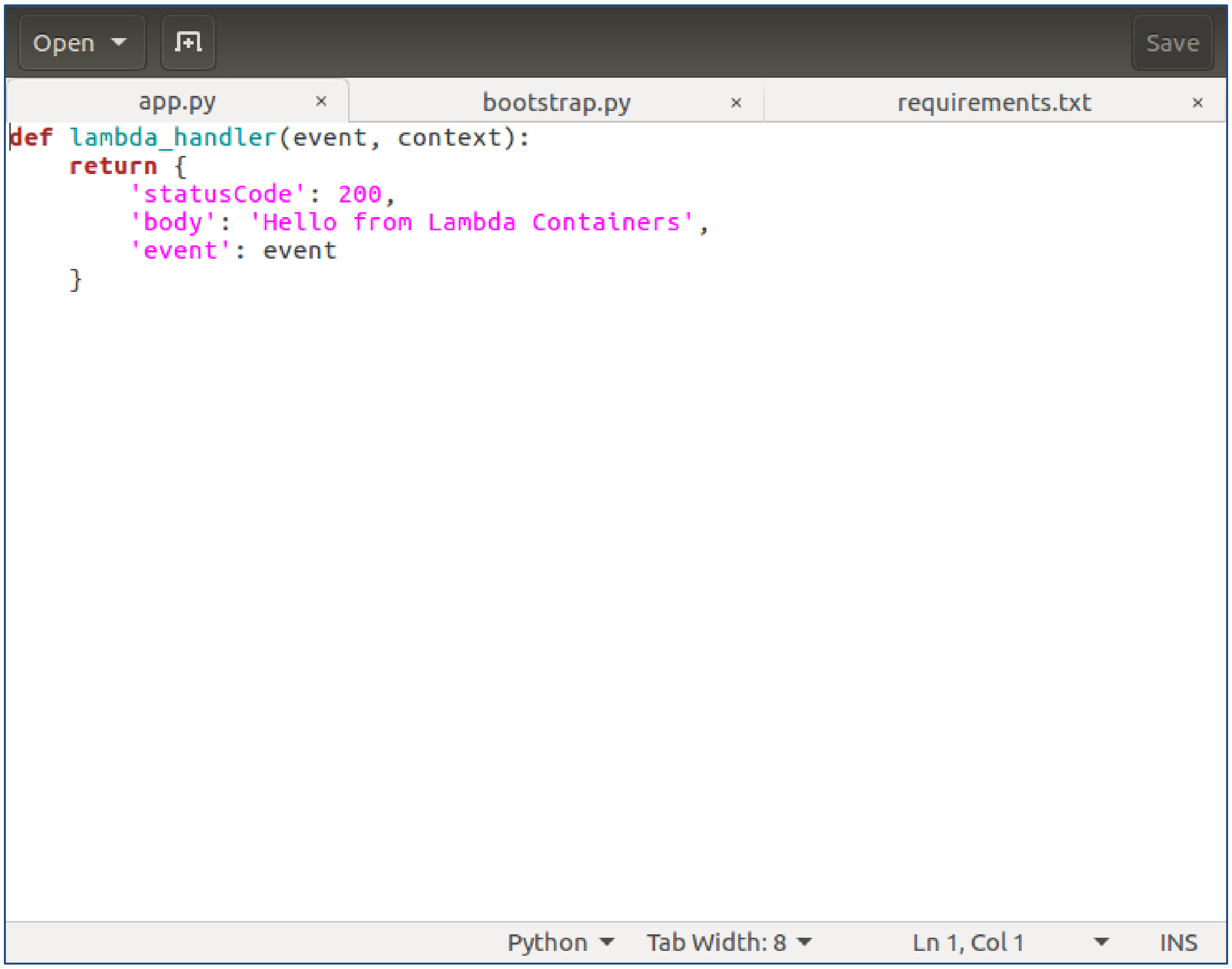
```
       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-90-98 ~]$ ls
OCI.zip
[ec2-user@ip-172-31-90-98 ~]$ sudo yum update
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                | 3.7 kB     00:00
No packages marked for update
[ec2-user@ip-172-31-90-98 ~]$ sudo yum install unzip
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package unzip-6.0-43.amzn2.x86_64 already installed and latest version
Nothing to do
[ec2-user@ip-172-31-90-98 ~]$ sudo unzip OCI.zip
Archive:  OCI.zip
  inflating: Dockerfile
   creating: content/
  inflating: content/app.py
  inflating: content/bootstrap.py
  inflating: content/requirements.txt
[ec2-user@ip-172-31-90-98 ~]$
```

Fig 1.g Unzipping OCI.zip (optional screenshot)



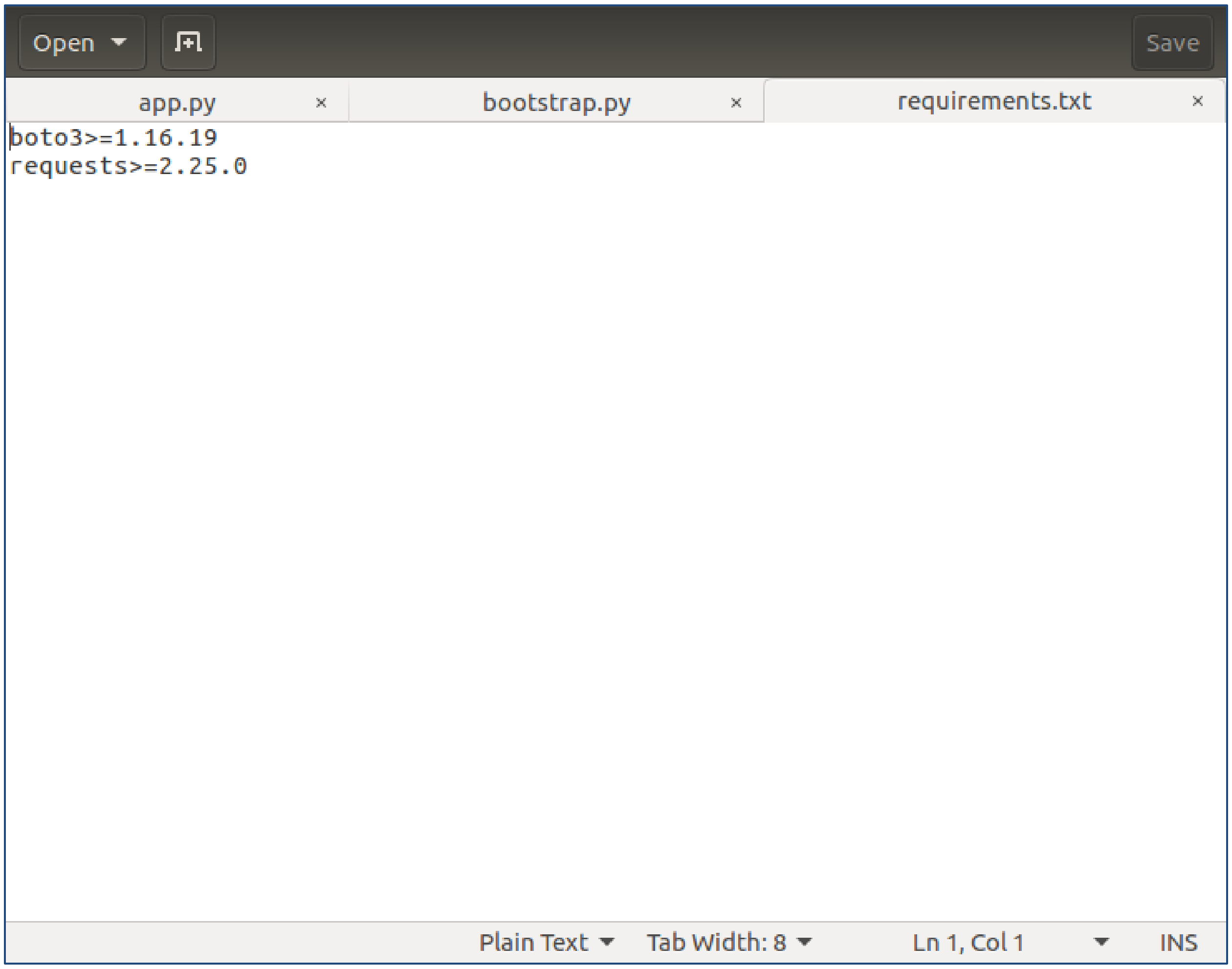
```
FROM python:alpine

COPY ./content .

RUN pip install -r requirements.txt

CMD python3 bootstrap.py
```

Fig 1.h Contents of Dockerfile (optional screenshot)



```python
def lambda_handler(event, context):
    return {
        'statusCode': 200,
        'body': 'Hello from Lambda Containers',
        'event': event
    }
```

Fig 1.i Contents of app.py file (optional screenshot)



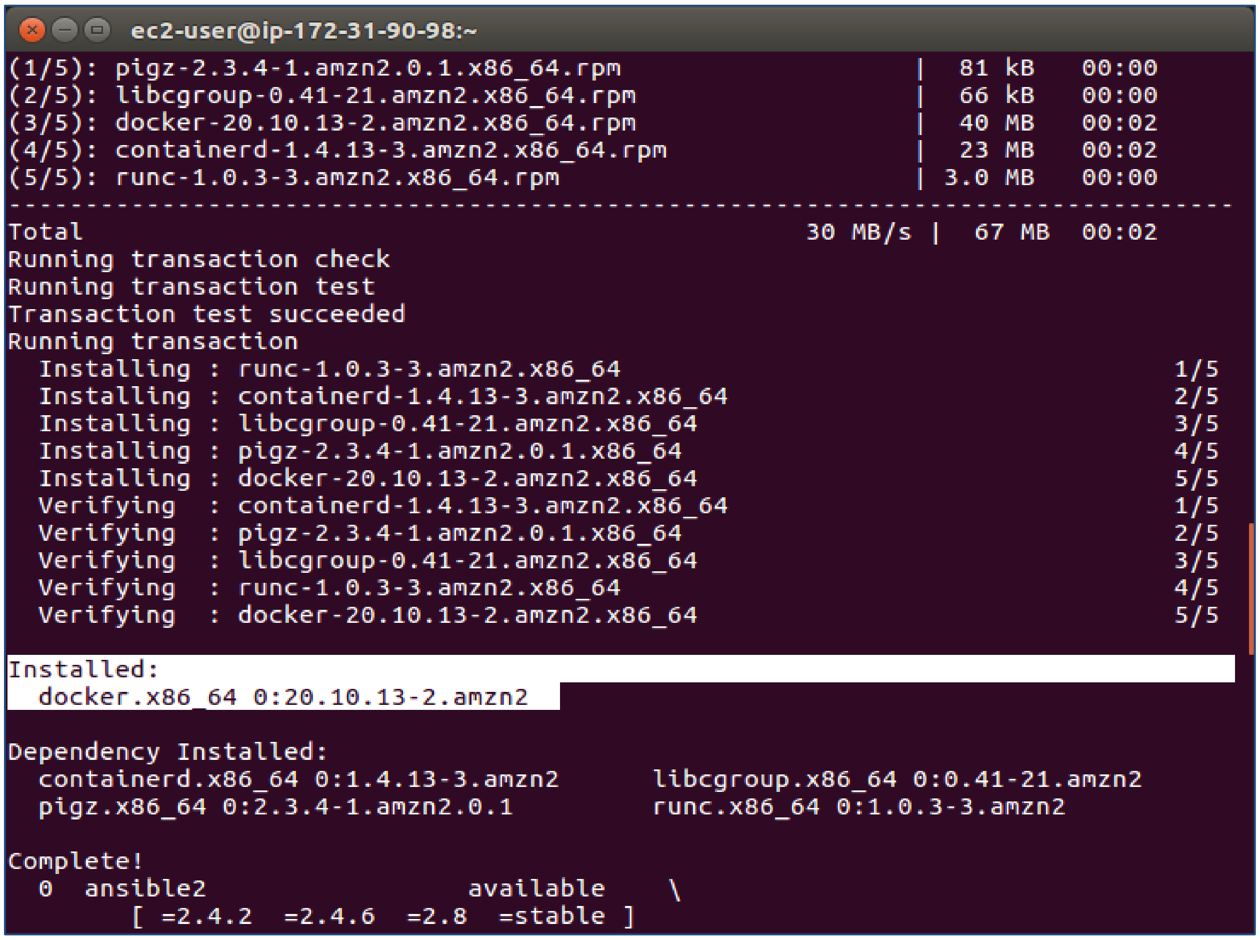
```
boto3>=1.16.19
requests>=2.25.0
```

Fig 1.j Contents of requirements.txt file (optional screenshot)

```
import os
import requests
import sys
import traceback

def run_loop():
        aws_lambda_runtime_api = os.environ['AWS_LAMBDA_RUNTIME_API']

        import app

        while True:
                request_id = None
                try:
                        invocation_response = requests.get(f'http://{aws_lambda_runtime_api}/2018-06-
01/runtime/invocation/next')

                        request_id = invocation_response.headers['Lambda-Runtime-Aws-Request-Id']
                        invoked_function_arn = invocation_response.headers['Lambda-Runtime-Invoked-Function-
Arn']
                        trace_id = invocation_response.headers['Lambda-Runtime-Trace-Id']
                        os.environ['_X_AMZN_TRACE_ID'] = trace_id

                        context = {
                                'request_id': request_id,
                                'invoked_function_arn': invoked_function_arn,
                                'trace_id': trace_id
                        }

                        event = invocation_response.json()

                        response_url = f'http://{aws_lambda_runtime_api}/2018-06-
01/runtime/invocation/{request_id}/response'

                        result = app.lambda_handler(event, context)

                        sys.stdout.flush()

                        requests.post(response_url, json=result)

                except:
                         if request_id != None:
                                try:
                                        exc_type, exc_value, exc_traceback = sys.exc_info()
                                        exception_message = {
```

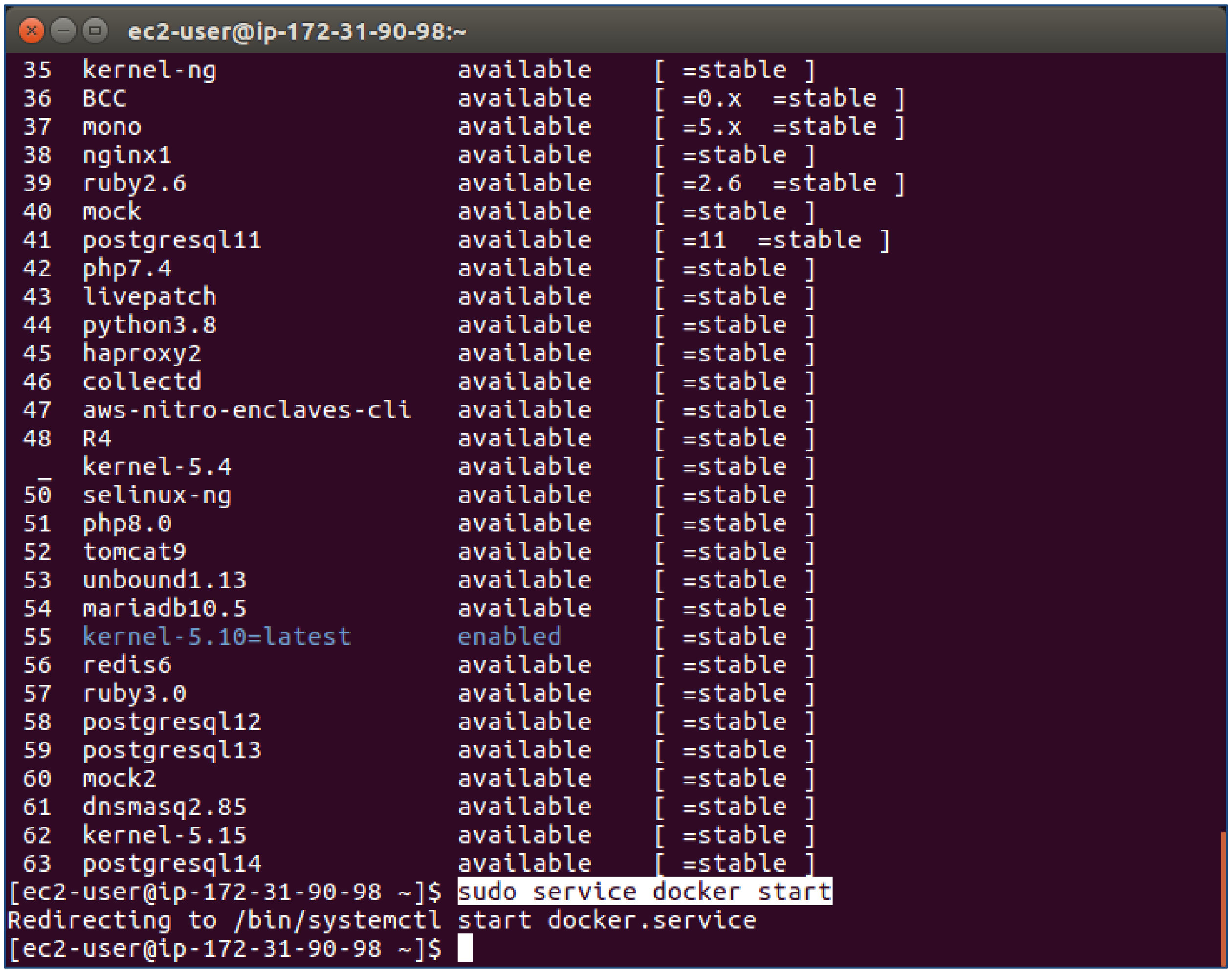
```
                'errorType': exc_type.__name__,
                'errorMessage': str(exc_value),
                'stackTrace': traceback.format_exception(exc_type,exc_value, exc_traceback)
            }

            error_url = f'http://{aws_lambda_runtime_api}/2018-06-
01/runtime/invocation/{request_id}/error'
            sys.stdout.flush()

            requests.post(error_url, json=exception_message)
        except:
            pass

run_loop()
```

Fig 1.k Contents of bootstrap.py file (optional screenshot)



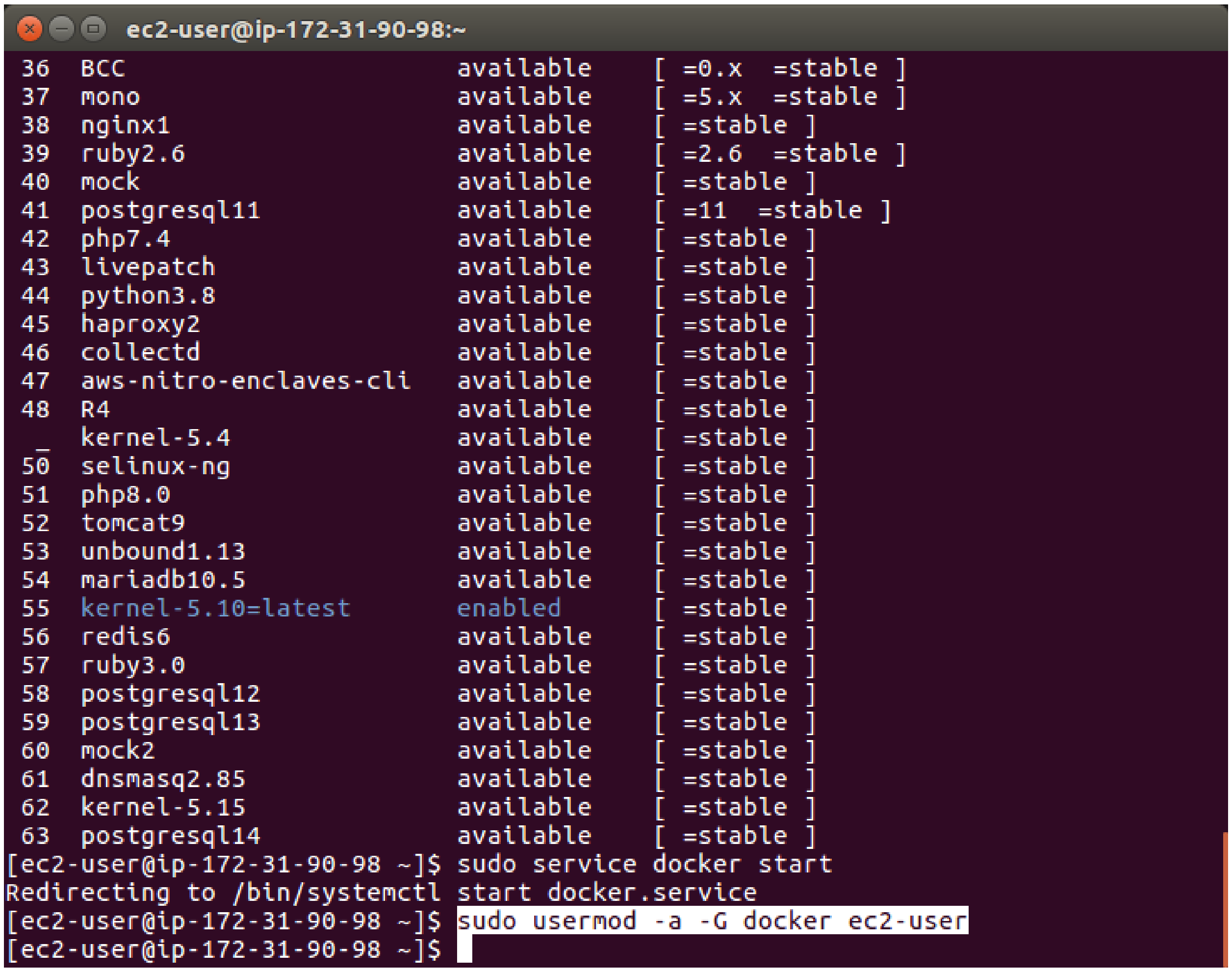
```
ec2-user@ip-172-31-90-98:~
(1/5): pigz-2.3.4-1.amzn2.0.1.x86_64.rpm                  |  81 kB   00:00
(2/5): libcgroup-0.41-21.amzn2.x86_64.rpm                 |  66 kB   00:00
(3/5): docker-20.10.13-2.amzn2.x86_64.rpm                 |  40 MB   00:02
(4/5): containerd-1.4.13-3.amzn2.x86_64.rpm               |  23 MB   00:02
(5/5): runc-1.0.3-3.amzn2.x86_64.rpm                      | 3.0 MB   00:00
--------------------------------------------------------------------------------
Total                                              30 MB/s |  67 MB   00:02
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : runc-1.0.3-3.amzn2.x86_64                                   1/5
  Installing : containerd-1.4.13-3.amzn2.x86_64                            2/5
  Installing : libcgroup-0.41-21.amzn2.x86_64                              3/5
  Installing : pigz-2.3.4-1.amzn2.0.1.x86_64                               4/5
  Installing : docker-20.10.13-2.amzn2.x86_64                              5/5
  Verifying  : containerd-1.4.13-3.amzn2.x86_64                            1/5
  Verifying  : pigz-2.3.4-1.amzn2.0.1.x86_64                               2/5
  Verifying  : libcgroup-0.41-21.amzn2.x86_64                              3/5
  Verifying  : runc-1.0.3-3.amzn2.x86_64                                   4/5
  Verifying  : docker-20.10.13-2.amzn2.x86_64                              5/5

Installed:
  docker.x86_64 0:20.10.13-2.amzn2

Dependency Installed:
  containerd.x86_64 0:1.4.13-3.amzn2      libcgroup.x86_64 0:0.41-21.amzn2
  pigz.x86_64 0:2.3.4-1.amzn2.0.1         runc.x86_64 0:1.0.3-3.amzn2

Complete!
  0  ansible2                 available    \
        [ =2.4.2  =2.4.6  =2.8  =stable ]
```

Fig 1.l Successful installation of docker (optional screenshot)



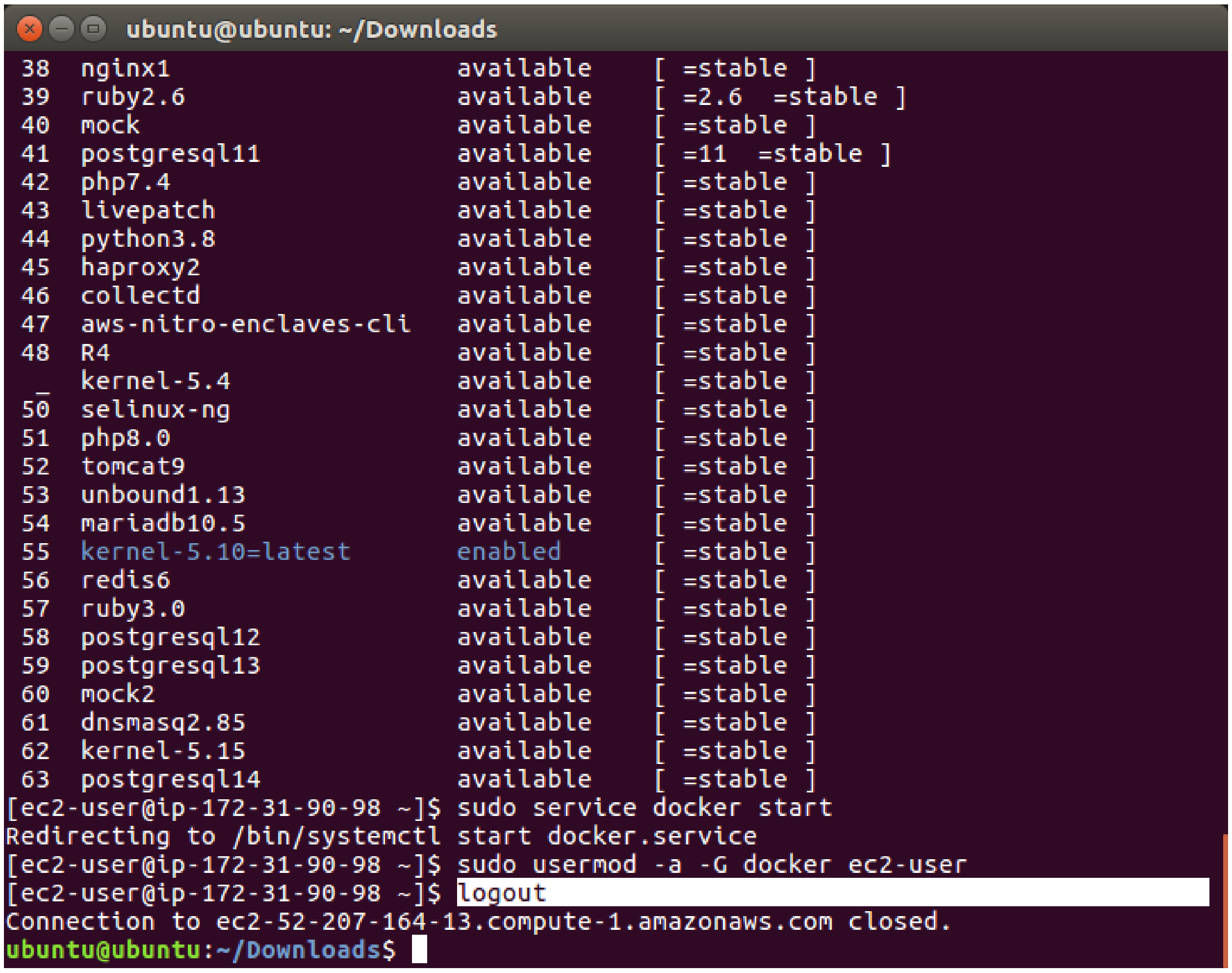
```
ec2-user@ip-172-31-90-98:~
 35  kernel-ng                available    [ =stable ]
 36  BCC                      available    [ =0.x  =stable ]
 37  mono                     available    [ =5.x  =stable ]
 38  nginx1                   available    [ =stable ]
 39  ruby2.6                  available    [ =2.6  =stable ]
 40  mock                     available    [ =stable ]
 41  postgresql11             available    [ =11  =stable ]
 42  php7.4                   available    [ =stable ]
 43  livepatch                available    [ =stable ]
 44  python3.8                available    [ =stable ]
 45  haproxy2                 available    [ =stable ]
 46  collectd                 available    [ =stable ]
 47  aws-nitro-enclaves-cli   available    [ =stable ]
 48  R4                       available    [ =stable ]
  _  kernel-5.4               available    [ =stable ]
 50  selinux-ng               available    [ =stable ]
 51  php8.0                   available    [ =stable ]
 52  tomcat9                  available    [ =stable ]
 53  unbound1.13              available    [ =stable ]
 54  mariadb10.5              available    [ =stable ]
 55  kernel-5.10=latest       enabled      [ =stable ]
 56  redis6                   available    [ =stable ]
 57  ruby3.0                  available    [ =stable ]
 58  postgresql12             available    [ =stable ]
 59  postgresql13             available    [ =stable ]
 60  mock2                    available    [ =stable ]
 61  dnsmasq2.85              available    [ =stable ]
 62  kernel-5.15              available    [ =stable ]
 63  postgresql14             available    [ =stable ]
[ec2-user@ip-172-31-90-98 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-90-98 ~]$
```

Fig 1.m docker service start (optional screenshot)



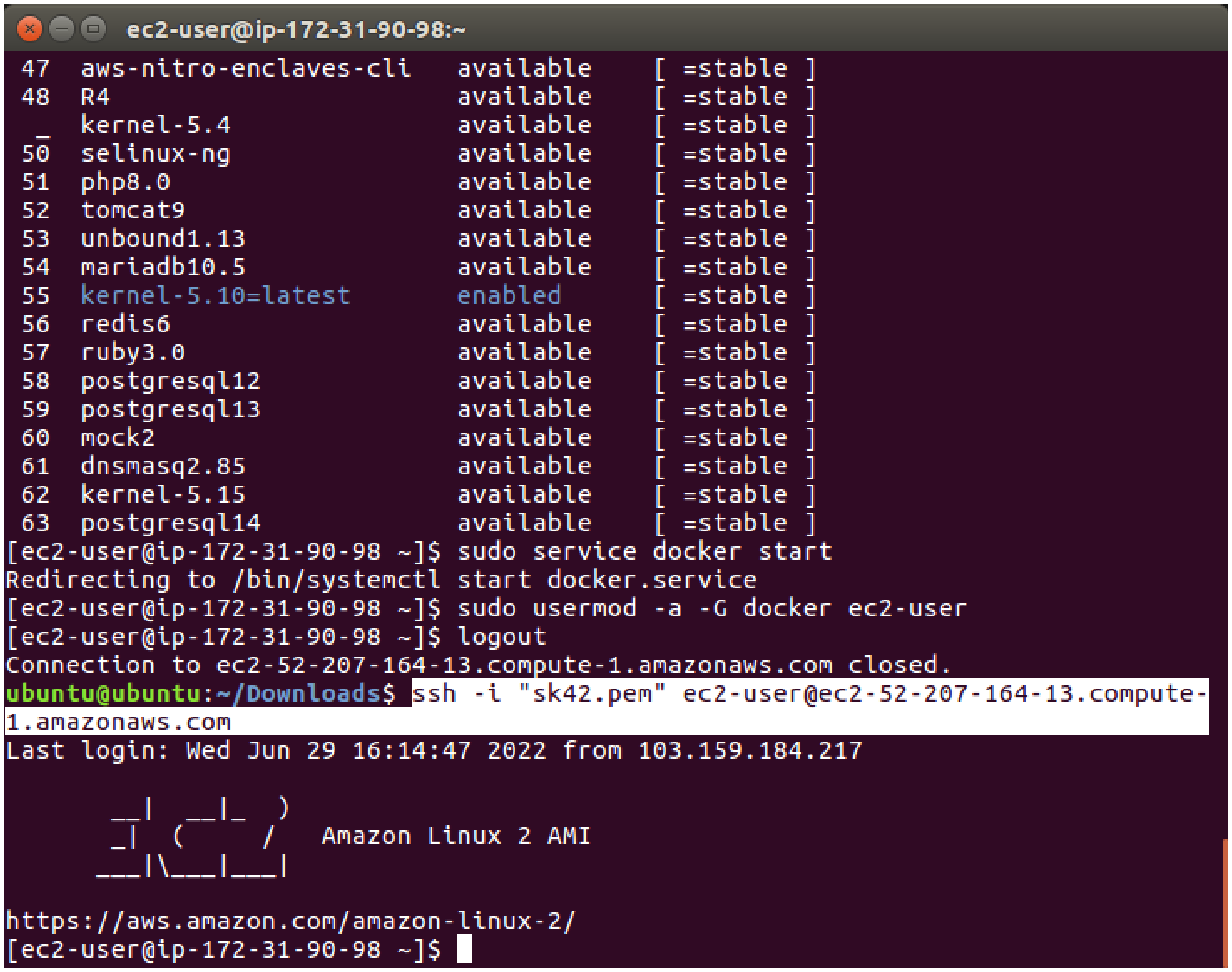
```
ec2-user@ip-172-31-90-98:~
 36  BCC                       available    [ =0.x  =stable ]
 37  mono                      available    [ =5.x  =stable ]
 38  nginx1                    available    [ =stable ]
 39  ruby2.6                   available    [ =2.6  =stable ]
 40  mock                      available    [ =stable ]
 41  postgresql11              available    [ =11  =stable ]
 42  php7.4                    available    [ =stable ]
 43  livepatch                 available    [ =stable ]
 44  python3.8                 available    [ =stable ]
 45  haproxy2                  available    [ =stable ]
 46  collectd                  available    [ =stable ]
 47  aws-nitro-enclaves-cli    available    [ =stable ]
 48  R4                        available    [ =stable ]
  _  kernel-5.4                available    [ =stable ]
 50  selinux-ng                available    [ =stable ]
 51  php8.0                    available    [ =stable ]
 52  tomcat9                   available    [ =stable ]
 53  unbound1.13               available    [ =stable ]
 54  mariadb10.5               available    [ =stable ]
 55  kernel-5.10=latest        enabled      [ =stable ]
 56  redis6                    available    [ =stable ]
 57  ruby3.0                   available    [ =stable ]
 58  postgresql12              available    [ =stable ]
 59  postgresql13              available    [ =stable ]
 60  mock2                     available    [ =stable ]
 61  dnsmasq2.85               available    [ =stable ]
 62  kernel-5.15               available    [ =stable ]
 63  postgresql14              available    [ =stable ]
[ec2-user@ip-172-31-90-98 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$
```

Fig 1.n Adding ec2-user to docker group(optional screenshot)



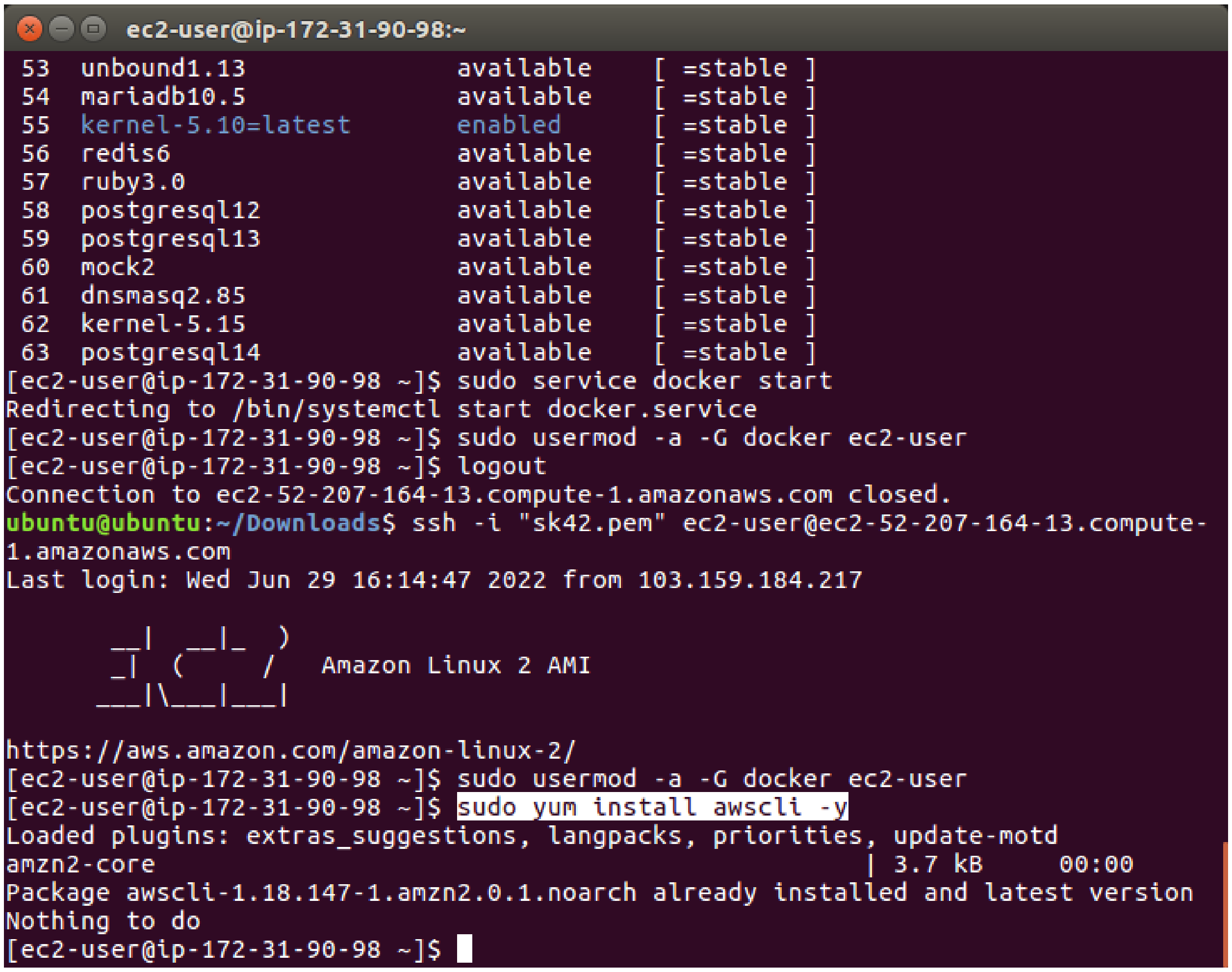
```
ubuntu@ubuntu: ~/Downloads
 38  nginx1                   available    [ =stable ]
 39  ruby2.6                  available    [ =2.6  =stable ]
 40  mock                     available    [ =stable ]
 41  postgresql11             available    [ =11  =stable ]
 42  php7.4                   available    [ =stable ]
 43  livepatch                available    [ =stable ]
 44  python3.8                available    [ =stable ]
 45  haproxy2                 available    [ =stable ]
 46  collectd                 available    [ =stable ]
 47  aws-nitro-enclaves-cli   available    [ =stable ]
 48  R4                       available    [ =stable ]
  _  kernel-5.4               available    [ =stable ]
 50  selinux-ng               available    [ =stable ]
 51  php8.0                   available    [ =stable ]
 52  tomcat9                  available    [ =stable ]
 53  unbound1.13              available    [ =stable ]
 54  mariadb10.5              available    [ =stable ]
 55  kernel-5.10=latest       enabled      [ =stable ]
 56  redis6                   available    [ =stable ]
 57  ruby3.0                  available    [ =stable ]
 58  postgresql12             available    [ =stable ]
 59  postgresql13             available    [ =stable ]
 60  mock2                    available    [ =stable ]
 61  dnsmasq2.85              available    [ =stable ]
 62  kernel-5.15              available    [ =stable ]
 63  postgresql14             available    [ =stable ]
[ec2-user@ip-172-31-90-98 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$ logout
Connection to ec2-52-207-164-13.compute-1.amazonaws.com closed.
ubuntu@ubuntu:~/Downloads$
```

Fig 1.o logout from ec2 instance to check whether ec2 user is added to docker group or not (optional screenshot)



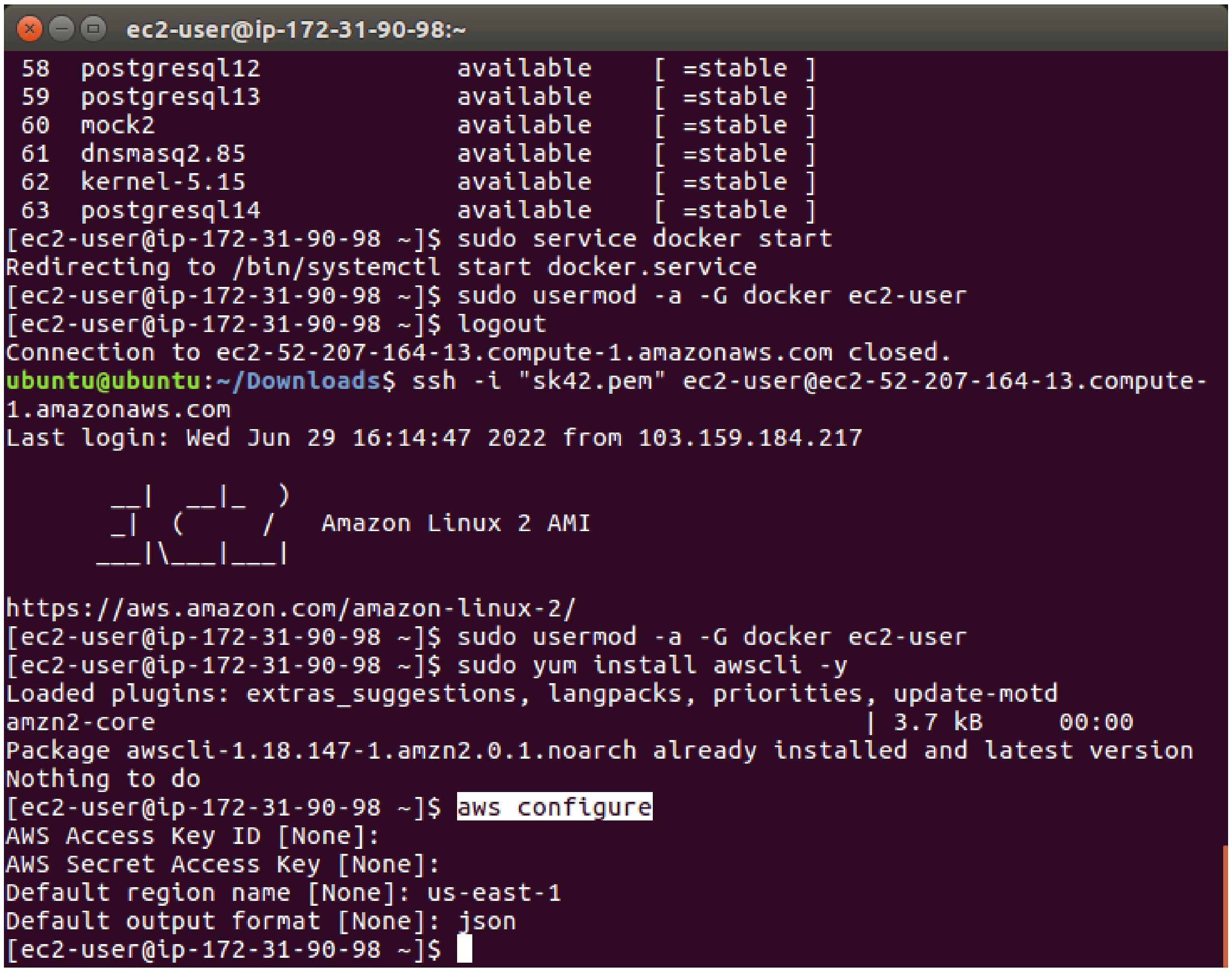
```
ec2-user@ip-172-31-90-98:~
 47  aws-nitro-enclaves-cli   available    [ =stable ]
 48  R4                       available    [ =stable ]
  _  kernel-5.4               available    [ =stable ]
 50  selinux-ng               available    [ =stable ]
 51  php8.0                   available    [ =stable ]
 52  tomcat9                  available    [ =stable ]
 53  unbound1.13              available    [ =stable ]
 54  mariadb10.5              available    [ =stable ]
 55  kernel-5.10=latest       enabled      [ =stable ]
 56  redis6                   available    [ =stable ]
 57  ruby3.0                  available    [ =stable ]
 58  postgresql12             available    [ =stable ]
 59  postgresql13             available    [ =stable ]
 60  mock2                    available    [ =stable ]
 61  dnsmasq2.85              available    [ =stable ]
 62  kernel-5.15              available    [ =stable ]
 63  postgresql14             available    [ =stable ]
[ec2-user@ip-172-31-90-98 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$ logout
Connection to ec2-52-207-164-13.compute-1.amazonaws.com closed.
ubuntu@ubuntu:~/Downloads$ ssh -i "sk42.pem" ec2-user@ec2-52-207-164-13.compute-
1.amazonaws.com
Last login: Wed Jun 29 16:14:47 2022 from 103.159.184.217

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-90-98 ~]$
```

Fig 1.p Login again to ec2 instance (optional screenshot)



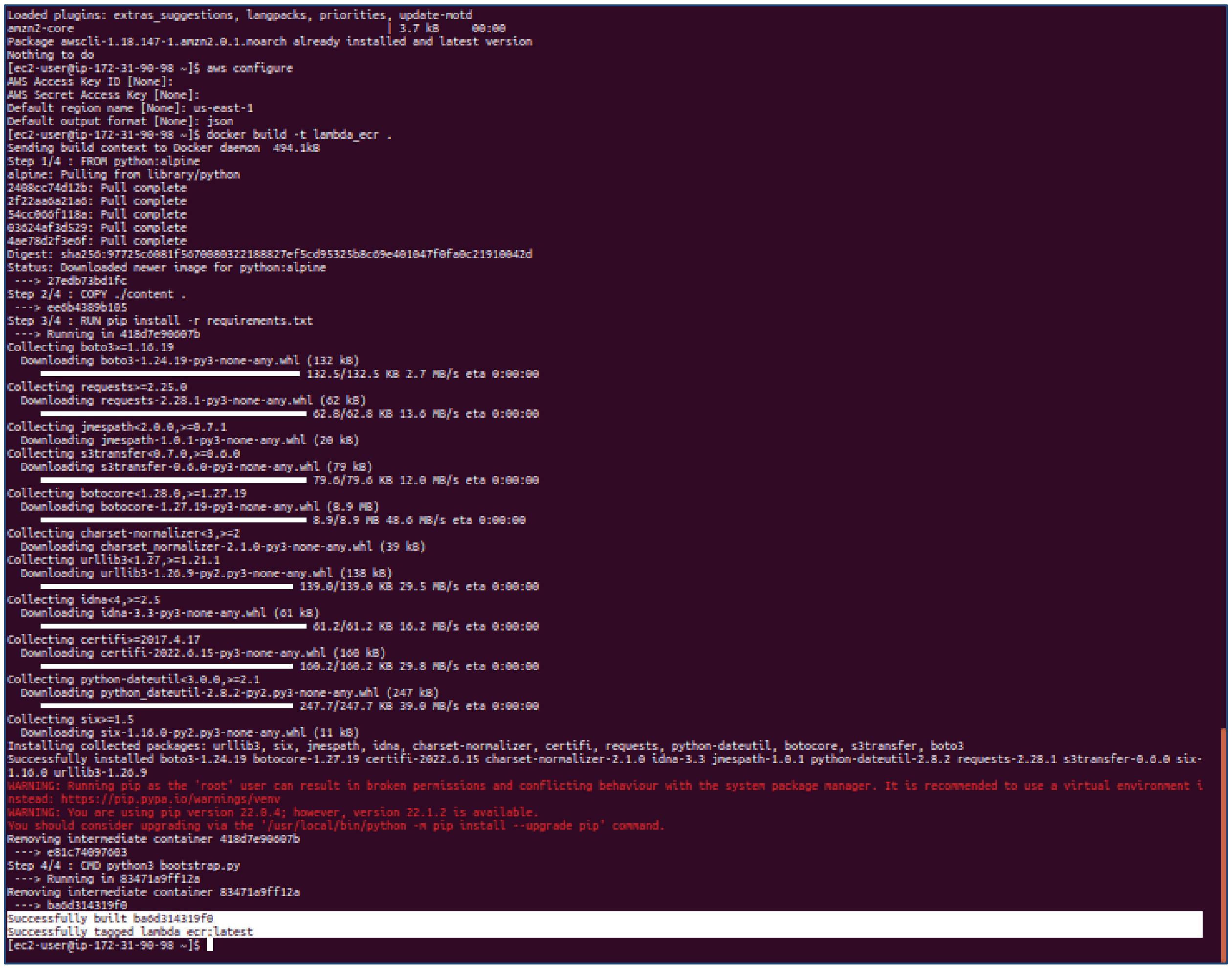
```
ec2-user@ip-172-31-90-98:~
 53  unbound1.13              available    [ =stable ]
 54  mariadb10.5              available    [ =stable ]
 55  kernel-5.10=latest       enabled      [ =stable ]
 56  redis6                   available    [ =stable ]
 57  ruby3.0                  available    [ =stable ]
 58  postgresql12             available    [ =stable ]
 59  postgresql13             available    [ =stable ]
 60  mock2                    available    [ =stable ]
 61  dnsmasq2.85              available    [ =stable ]
 62  kernel-5.15              available    [ =stable ]
 63  postgresql14             available    [ =stable ]
[ec2-user@ip-172-31-90-98 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$ logout
Connection to ec2-52-207-164-13.compute-1.amazonaws.com closed.
ubuntu@ubuntu:~/Downloads$ ssh -i "sk42.pem" ec2-user@ec2-52-207-164-13.compute-
1.amazonaws.com
Last login: Wed Jun 29 16:14:47 2022 from 103.159.184.217

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$ sudo yum install awscli -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                | 3.7 kB     00:00
Package awscli-1.18.147-1.amzn2.0.1.noarch already installed and latest version
Nothing to do
[ec2-user@ip-172-31-90-98 ~]$ 
```

Fig 1.q Installing awscli for building docker image via CLI(optional screenshot)



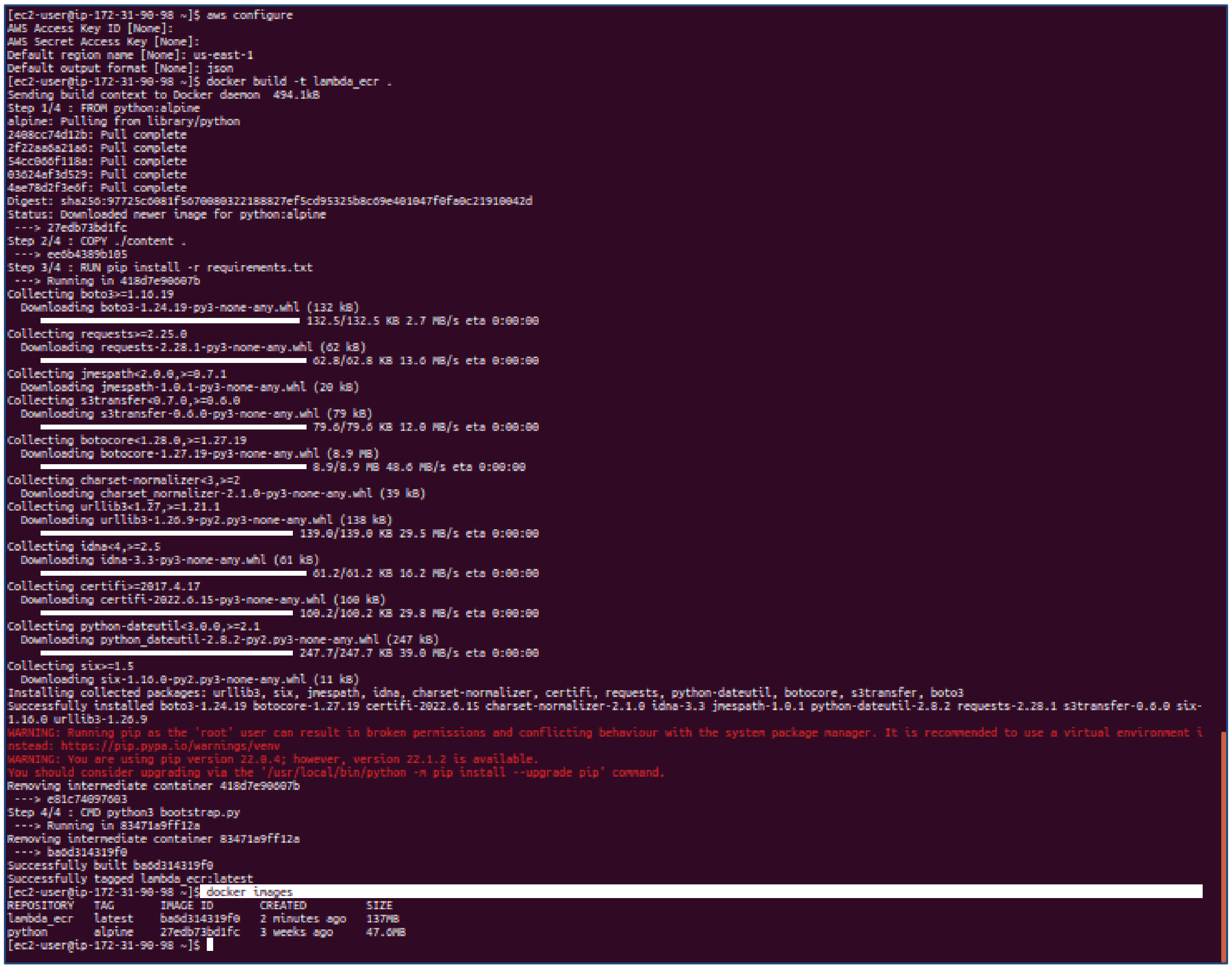
```
 58  postgresql12              available    [ =stable ]
 59  postgresql13              available    [ =stable ]
 60  mock2                     available    [ =stable ]
 61  dnsmasq2.85               available    [ =stable ]
 62  kernel-5.15               available    [ =stable ]
 63  postgresql14              available    [ =stable ]
[ec2-user@ip-172-31-90-98 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$ logout
Connection to ec2-52-207-164-13.compute-1.amazonaws.com closed.
ubuntu@ubuntu:~/Downloads$ ssh -i "sk42.pem" ec2-user@ec2-52-207-164-13.compute-
1.amazonaws.com
Last login: Wed Jun 29 16:14:47 2022 from 103.159.184.217

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-90-98 ~]$ sudo usermod -a -G docker ec2-user
[ec2-user@ip-172-31-90-98 ~]$ sudo yum install awscli -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                | 3.7 kB     00:00
Package awscli-1.18.147-1.amzn2.0.1.noarch already installed and latest version
Nothing to do
[ec2-user@ip-172-31-90-98 ~]$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-east-1
Default output format [None]: json
[ec2-user@ip-172-31-90-98 ~]$
```

Fig 1.r Configure AWS (fastest way to set up your AWS CLI installation) (optional screenshot)



```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                     | 3.7 kB     00:00
Package awscli-1.18.147-1.amzn2.0.1.noarch already installed and latest version
Nothing to do
[ec2-user@ip-172-31-90-98 ~]$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-east-1
Default output format [None]: json
[ec2-user@ip-172-31-90-98 ~]$ docker build -t lambda_ecr .
Sending build context to Docker daemon  494.1kB
Step 1/4 : FROM python:alpine
alpine: Pulling from library/python
2408cc74d12b: Pull complete
2f22aa0a21a0: Pull complete
54cc000f118a: Pull complete
03624af3d529: Pull complete
4ae78d2f3e6f: Pull complete
Digest: sha256:97725c0881f5670080322188827ef5cd95325b8c09e401047f0fa0c21910042d
Status: Downloaded newer image for python:alpine
 ---> 27edb73bd1fc
Step 2/4 : COPY ./content .
 ---> ee6b4389b105
Step 3/4 : RUN pip install -r requirements.txt
 ---> Running in 418d7e90007b
Collecting boto3>=1.16.19
  Downloading boto3-1.24.19-py3-none-any.whl (132 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 132.5/132.5 KB 2.7 MB/s eta 0:00:00
Collecting requests>=2.25.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.8/62.8 KB 13.6 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.6/79.6 KB 12.0 MB/s eta 0:00:00
Collecting botocore<1.28.0,>=1.27.19
  Downloading botocore-1.27.19-py3-none-any.whl (8.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.9/8.9 MB 48.6 MB/s eta 0:00:00
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.0-py3-none-any.whl (39 kB)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.0/139.0 KB 29.5 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.3-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.2/61.2 KB 16.2 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.6.15-py3-none-any.whl (160 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 160.2/160.2 KB 29.8 MB/s eta 0:00:00
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 247.7/247.7 KB 39.0 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-dateutil, botocore, s3transfer, boto3
Successfully installed boto3-1.24.19 botocore-1.27.19 certifi-2022.6.15 charset-normalizer-2.1.0 idna-3.3 jmespath-1.0.1 python-dateutil-2.8.2 requests-2.28.1 s3transfer-0.6.0 six-
1.16.0 urllib3-1.26.9
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment i
nstead: https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.1.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container 418d7e90007b
 ---> e81c74097003
Step 4/4 : CMD python3 bootstrap.py
 ---> Running in 83471a9ff12a
Removing intermediate container 83471a9ff12a
 ---> ba6d314319f0
Successfully built ba6d314319f0
Successfully tagged lambda_ecr:latest
[ec2-user@ip-172-31-90-98 ~]$ 
```

Fig 1.s Building docker image



```
[ec2-user@ip-172-31-98-98 ~]$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]: us-east-1
Default output format [None]: json
[ec2-user@ip-172-31-98-98 ~]$ docker build -t lambda_ecr .
Sending build context to Docker daemon  494.1kB
Step 1/4 : FROM python:alpine
alpine: Pulling from library/python
2408cc74d12b: Pull complete
2f22aa6a21a6: Pull complete
54cc066f118a: Pull complete
03624af3d529: Pull complete
4ae78d2f3e6f: Pull complete
Digest: sha256:97725c0081f5670080322188827ef5cd95325b8c09e401047f0fa0c21910042d
Status: Downloaded newer image for python:alpine
 ---> 27edb73bd1fc
Step 2/4 : COPY ./content .
 ---> ee6b4389b105
Step 3/4 : RUN pip install -r requirements.txt
 ---> Running in 418d7e90007b
Collecting boto3>=1.16.19
  Downloading boto3-1.24.19-py3-none-any.whl (132 kB)
                                           132.5/132.5 KB 2.7 MB/s eta 0:00:00
Collecting requests>=2.25.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
                                            62.8/62.8 KB 13.6 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
                                            79.6/79.6 KB 12.0 MB/s eta 0:00:00
Collecting botocore<1.28.0,>=1.27.19
  Downloading botocore-1.27.19-py3-none-any.whl (8.9 MB)
                                            8.9/8.9 MB 48.6 MB/s eta 0:00:00
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.0-py3-none-any.whl (39 kB)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
                                          139.0/139.0 KB 29.5 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.3-py3-none-any.whl (61 kB)
                                            61.2/61.2 KB 16.2 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.6.15-py3-none-any.whl (160 kB)
                                          160.2/160.2 KB 29.8 MB/s eta 0:00:00
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
                                          247.7/247.7 KB 39.0 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-dateutil, botocore, s3transfer, boto3
Successfully installed boto3-1.24.19 botocore-1.27.19 certifi-2022.6.15 charset-normalizer-2.1.0 idna-3.3 jmespath-1.0.1 python-dateutil-2.8.2 requests-2.28.1 s3transfer-0.6.0 six-
1.16.0 urllib3-1.26.9
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment i
nstead: https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.1.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container 418d7e90007b
 ---> e81c74097603
Step 4/4 : CMD python3 bootstrap.py
 ---> Running in 83471a9ff12a
Removing intermediate container 83471a9ff12a
 ---> ba6d314319f0
Successfully built ba6d314319f0
Successfully tagged lambda_ecr:latest
[ec2-user@ip-172-31-98-98 ~]$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
lambda_ecr   latest    ba6d314319f0   2 minutes ago   137MB
python       alpine    27edb73bd1fc   3 weeks ago     47.6MB
[ec2-user@ip-172-31-98-98 ~]$
```

Fig 1.t List of created images

Step 2: Create ECR repository and upload image to ECR

Step number a

Step name Creating the ECR repository

Instructions

1) Go to the ECR service on the AWS console
2) Select the Repositories from the left pane
3) Create a new private repository named **lambda_ecr** with the default settings

Step number b

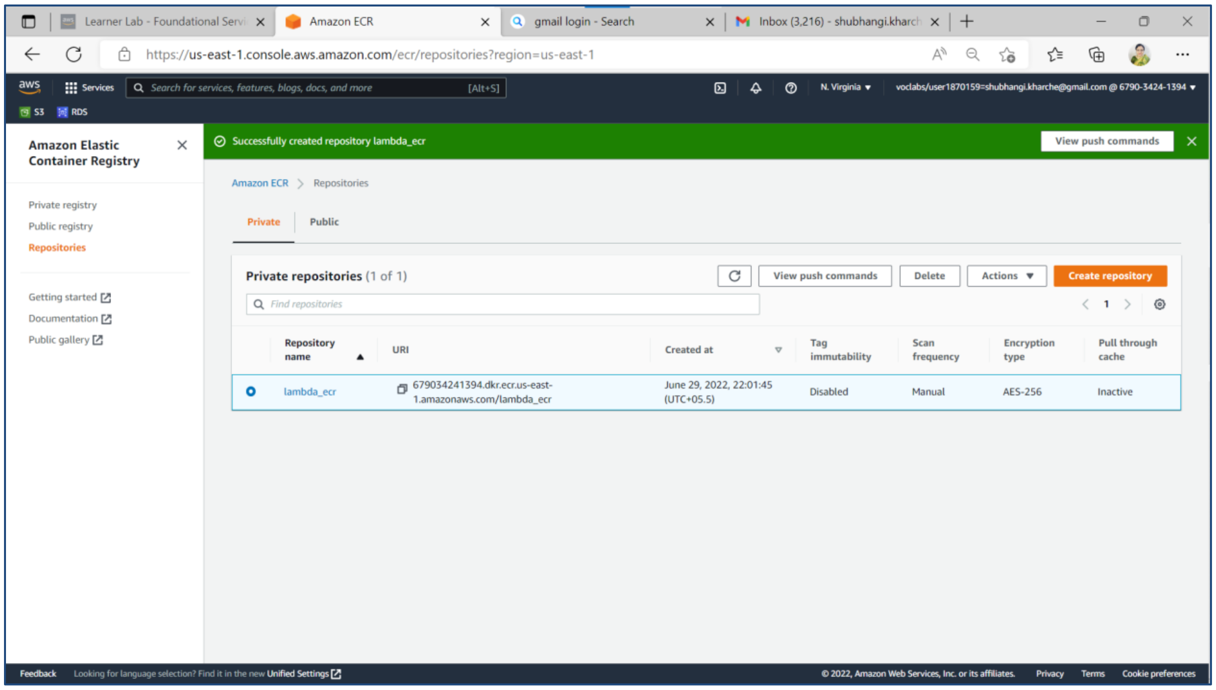
Step name Image upload to ECR

Instructions

1) Once the repository is created, select the repository and then click on “View push commands“ on the top right
2) From the pop up screen which appears, run commands 1, 3 and 4 after logging into the EC2 instance created above. Note that command 2 was already executed in the previous step when the image was created.
   For reference, the commands will be in the format shown below:

```
aws ecr get-login-password --region us-east-1 | docker login --username
AWS --password-stdin <xxxxxxx.dkr.ecr.us-east-1.amazonaws.com>

docker tag lambda_ecr_image:latest <xxxxxxx.dkr.ecr.us-east-
1.amazonaws.com/lambda_ecr>:latest

docker push <xxxxxxx.dkr.ecr.us-east-
1.amazonaws.com/lambda_ecr>:latest
```

Expected screenshots 1) Creation of Repository 2) View push commands 3) Login Succeeded 4) Tagging of the image 5) Pushing of image to ECR 6) Image uploaded on the ECR repo



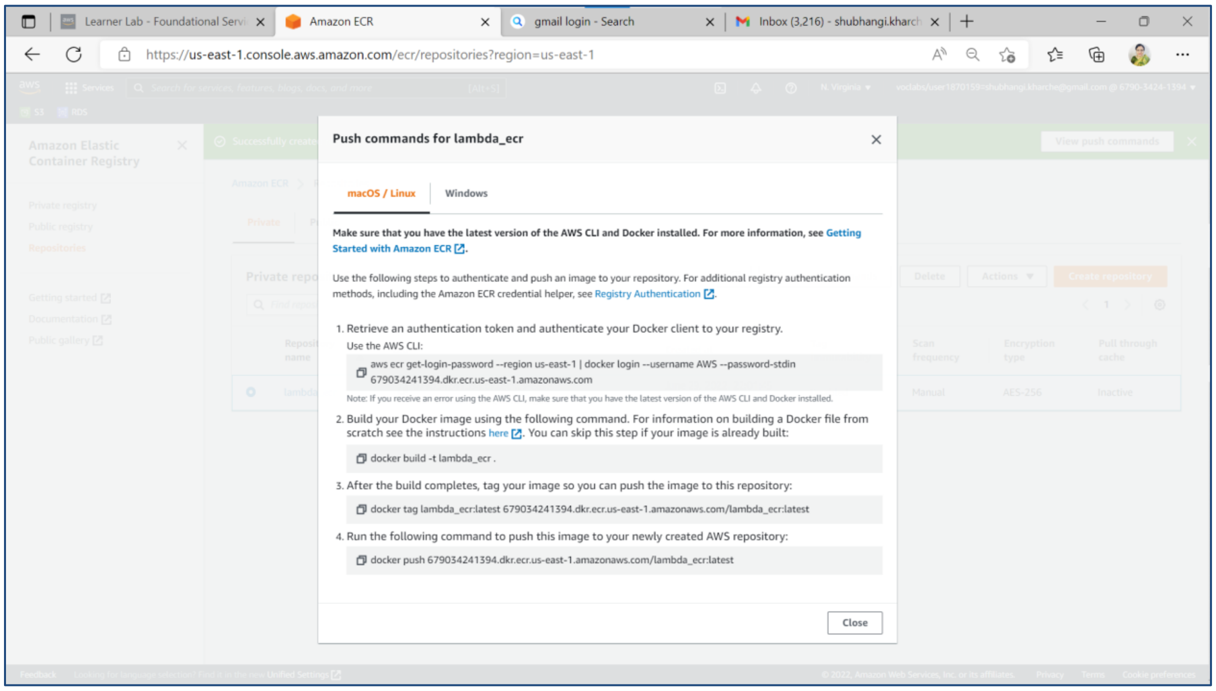
Successfully created repository lambda_ecr

Amazon ECR > Repositories

Private repositories (1 of 1)

	Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
●	lambda_ecr	679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr	June 29, 2022, 22:01:45 (UTC+05.5)	Disabled	Manual	AES-256	Inactive

Fig. 2 a) Creation of Repository on ECR (lambda_ecr)



Push commands for lambda_ecr

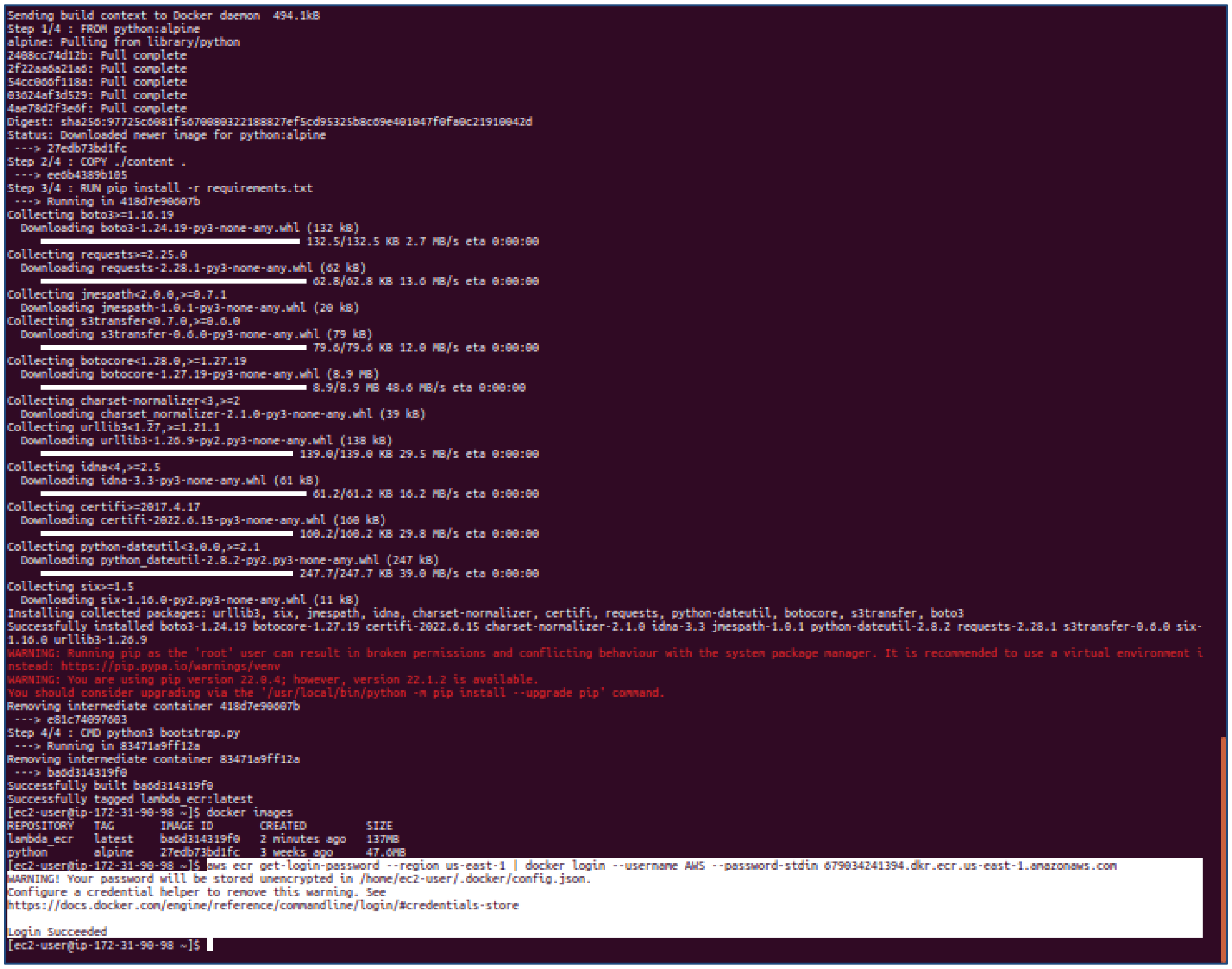
macOS / Linux | Windows

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.

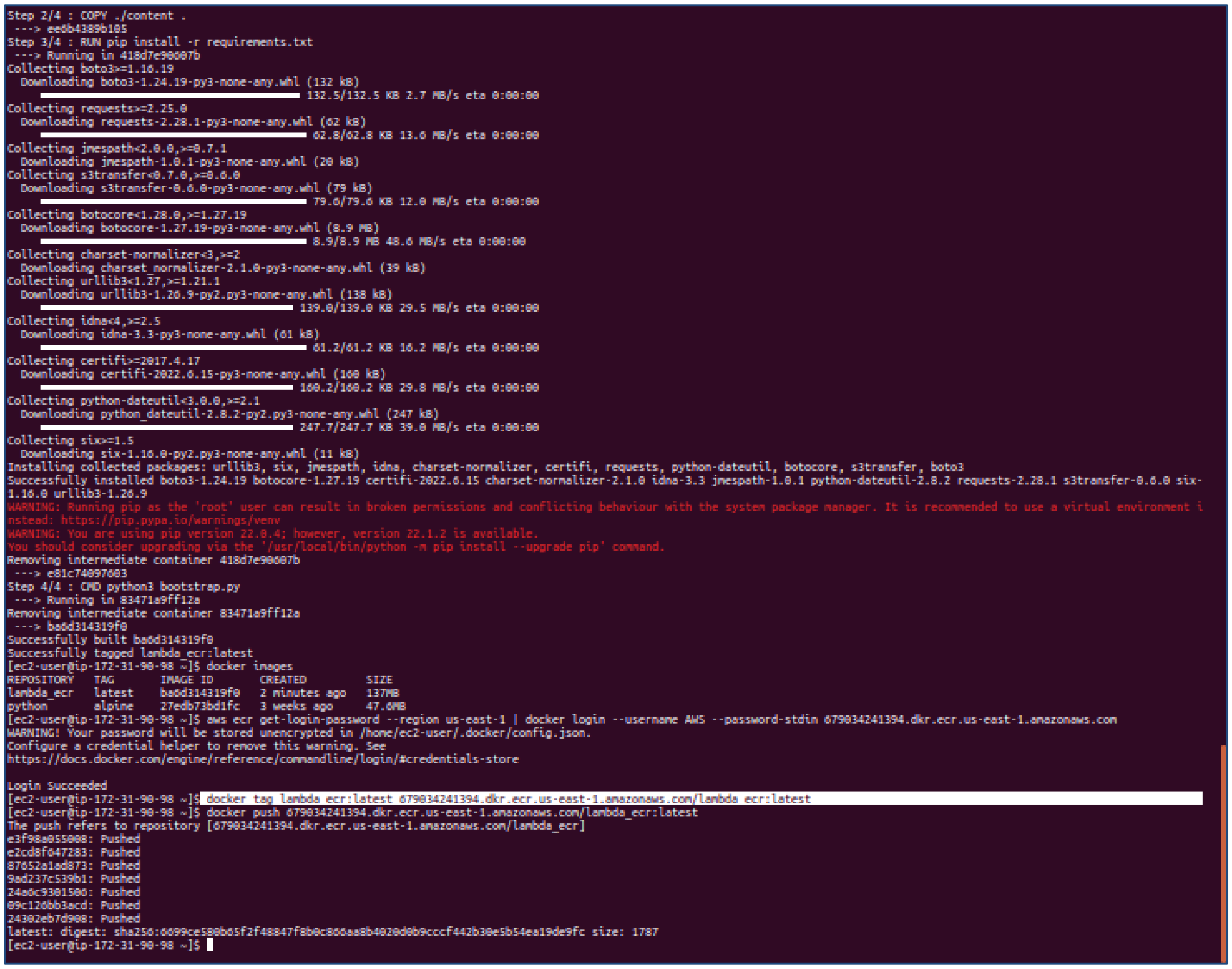
1. Retrieve an authentication token and authenticate your Docker client to your registry.
   Use the AWS CLI:
   ```
   aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 679034241394.dkr.ecr.us-east-1.amazonaws.com
   ```
   Note: If you receive an error using 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 lambda_ecr .
   ```
3. After the build completes, tag your image so you can push the image to this repository:
   ```
   docker tag lambda_ecr:latest 679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr:latest
   ```
4. Run the following command to push this image to your newly created AWS repository:
   ```
   docker push 679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr:latest
   ```

Fig. 2 b) View push commands



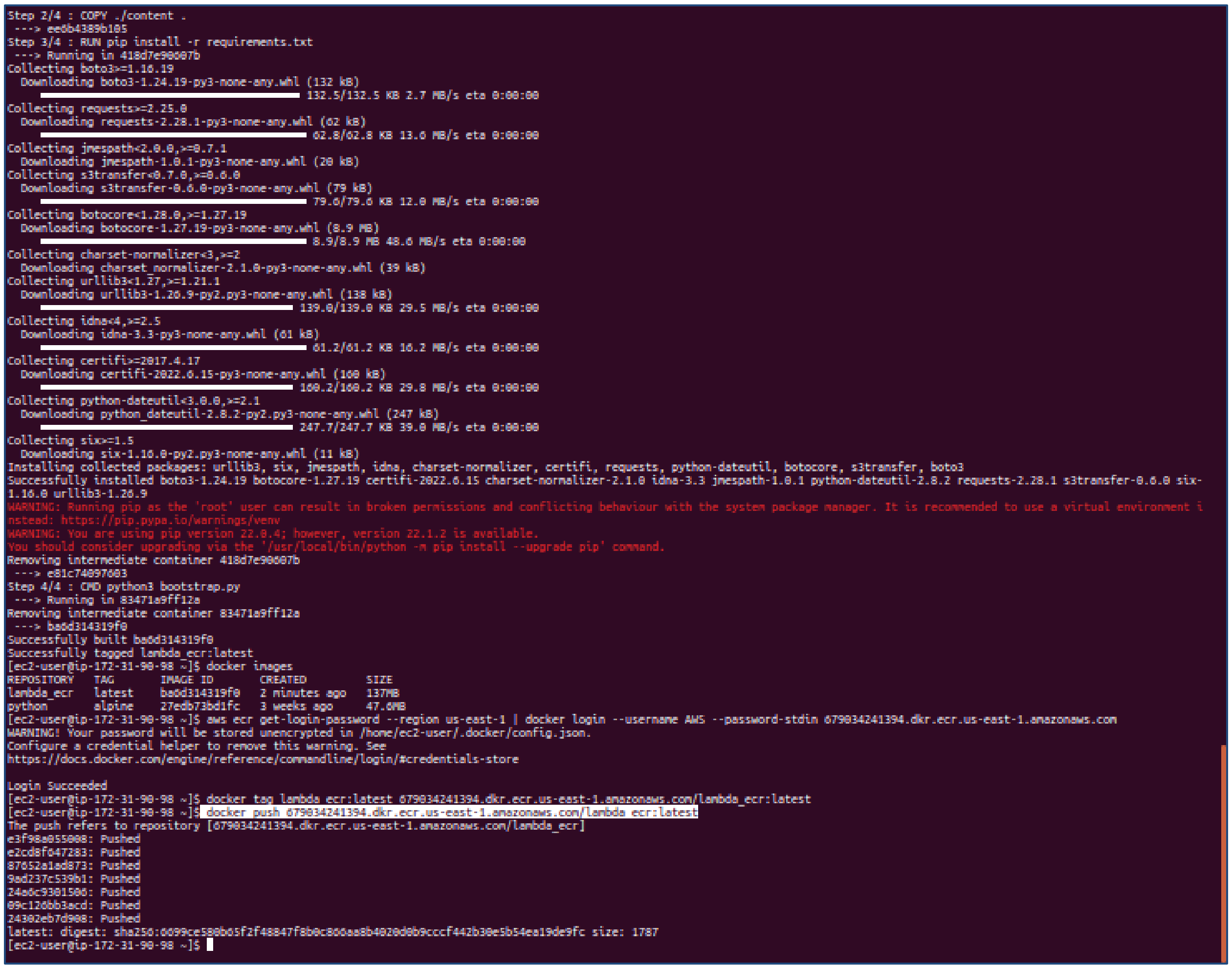
```
Sending build context to Docker daemon  494.1kB
Step 1/4 : FROM python:alpine
alpine: Pulling from library/python
2408cc74d12b: Pull complete
2f22aa6a21a8: Pull complete
54cc060f118a: Pull complete
03624af3d529: Pull complete
4ae78d2f3e0f: Pull complete
Digest: sha256:97725c0081f5670080322188827ef5cd95325b8c69e401047f0fa0c21910042d
Status: Downloaded newer image for python:alpine
 ---> 27edb73bd1fc
Step 2/4 : COPY ./content .
 ---> ee6b4389b105
Step 3/4 : RUN pip install -r requirements.txt
 ---> Running in 418d7e90007b
Collecting boto3>=1.16.19
  Downloading boto3-1.24.19-py3-none-any.whl (132 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 132.5/132.5 KB 2.7 MB/s eta 0:00:00
Collecting requests>=2.25.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.8/62.8 KB 13.6 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.6/79.6 KB 12.0 MB/s eta 0:00:00
Collecting botocore<1.28.0,>=1.27.19
  Downloading botocore-1.27.19-py3-none-any.whl (8.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.9/8.9 MB 48.6 MB/s eta 0:00:00
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.0-py3-none-any.whl (39 kB)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.0/139.0 KB 29.5 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.3-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.2/61.2 KB 16.2 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.6.15-py3-none-any.whl (160 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 160.2/160.2 KB 29.8 MB/s eta 0:00:00
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 247.7/247.7 KB 39.0 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-dateutil, botocore, s3transfer, boto3
Successfully installed boto3-1.24.19 botocore-1.27.19 certifi-2022.6.15 charset-normalizer-2.1.0 idna-3.3 jmespath-1.0.1 python-dateutil-2.8.2 requests-2.28.1 s3transfer-0.6.0 six-
1.16.0 urllib3-1.26.9
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment i
nstead: https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.1.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container 418d7e90007b
 ---> e81c74097603
Step 4/4 : CMD python3 bootstrap.py
 ---> Running in 83471a9ff12a
Removing intermediate container 83471a9ff12a
 ---> ba6d314319f0
Successfully built ba6d314319f0
Successfully tagged lambda_ecr:latest
[ec2-user@ip-172-31-90-98 ~]$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
lambda_ecr   latest    ba6d314319f0   2 minutes ago   137MB
python       alpine    27edb73bd1fc   3 weeks ago     47.6MB
[ec2-user@ip-172-31-90-98 ~]$ aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 679034241394.dkr.ecr.us-east-1.amazonaws.com
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
[ec2-user@ip-172-31-90-98 ~]$
```

Fig. 2 c) Login Succeeded



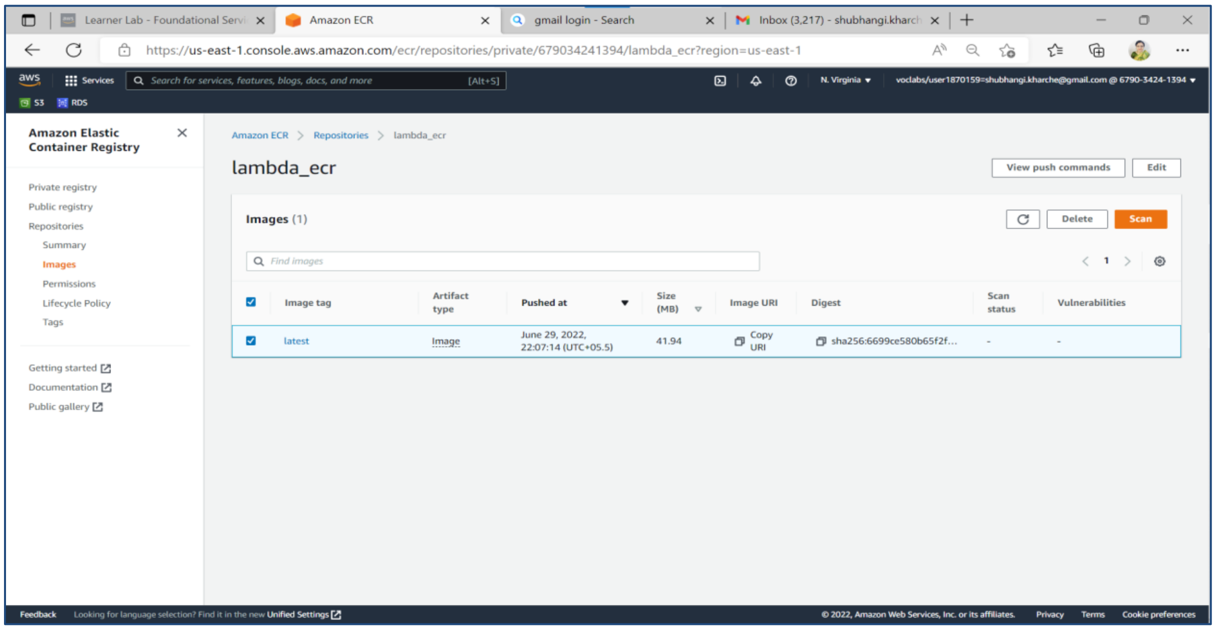
```
Step 2/4 : COPY ./content .
 ---> eedb4389b105
Step 3/4 : RUN pip install -r requirements.txt
 ---> Running in 418d7e98087b
Collecting boto3>=1.16.19
  Downloading boto3-1.24.19-py3-none-any.whl (132 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 132.5/132.5 KB 2.7 MB/s eta 0:00:00
Collecting requests>=2.25.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.8/62.8 KB 13.6 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.6/79.6 KB 12.0 MB/s eta 0:00:00
Collecting botocore<1.28.0,>=1.27.19
  Downloading botocore-1.27.19-py3-none-any.whl (8.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.9/8.9 MB 48.6 MB/s eta 0:00:00
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.0-py3-none-any.whl (39 kB)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.0/139.0 KB 29.5 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.3-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.2/61.2 KB 16.2 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.6.15-py3-none-any.whl (160 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 160.2/160.2 KB 29.8 MB/s eta 0:00:00
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 247.7/247.7 KB 39.0 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-dateutil, botocore, s3transfer, boto3
Successfully installed boto3-1.24.19 botocore-1.27.19 certifi-2022.6.15 charset-normalizer-2.1.0 idna-3.3 jmespath-1.0.1 python-dateutil-2.8.2 requests-2.28.1 s3transfer-0.6.0 six-
1.16.0 urllib3-1.26.9
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment i
nstead: https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.1.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container 418d7e98087b
 ---> e81c74097603
Step 4/4 : CMD python3 bootstrap.py
 ---> Running in 83471a9ff12a
Removing intermediate container 83471a9ff12a
 ---> ba6d314319f0
Successfully built ba6d314319f0
Successfully tagged lambda_ecr:latest
[ec2-user@ip-172-31-90-98 ~]$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
lambda_ecr   latest    ba6d314319f0   2 minutes ago   137MB
python       alpine    27edb73bd1fc   3 weeks ago     47.6MB
[ec2-user@ip-172-31-90-98 ~]$ aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 679034241394.dkr.ecr.us-east-1.amazonaws.com
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
[ec2-user@ip-172-31-90-98 ~]$ docker tag lambda_ecr:latest 679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr:latest
[ec2-user@ip-172-31-90-98 ~]$ docker push 679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr:latest
The push refers to repository [679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr]
e3f98a055008: Pushed
e2cd8f047283: Pushed
87652a1ad873: Pushed
9ad237c539b1: Pushed
24a6c9301506: Pushed
09c126bb3acd: Pushed
24302eb7d908: Pushed
latest: digest: sha256:6099ce580b05f2f48847f8b0c806aa8b4020d0b9cccf442b30e5b54ea19de9fc size: 1787
[ec2-user@ip-172-31-90-98 ~]$ █
```

Fig. 2 d) Tagging of the image



```
Step 2/4 : COPY ./content .
 ---> eedb4389b105
Step 3/4 : RUN pip install -r requirements.txt
 ---> Running in 418d7e90007b
Collecting boto3>=1.16.19
  Downloading boto3-1.24.19-py3-none-any.whl (132 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 132.5/132.5 KB 2.7 MB/s eta 0:00:00
Collecting requests>=2.25.0
  Downloading requests-2.28.1-py3-none-any.whl (62 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.8/62.8 KB 13.0 MB/s eta 0:00:00
Collecting jmespath<2.0.0,>=0.7.1
  Downloading jmespath-1.0.1-py3-none-any.whl (20 kB)
Collecting s3transfer<0.7.0,>=0.6.0
  Downloading s3transfer-0.6.0-py3-none-any.whl (79 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 79.6/79.6 KB 12.0 MB/s eta 0:00:00
Collecting botocore<1.28.0,>=1.27.19
  Downloading botocore-1.27.19-py3-none-any.whl (8.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.9/8.9 MB 48.6 MB/s eta 0:00:00
Collecting charset-normalizer<3,>=2
  Downloading charset_normalizer-2.1.0-py3-none-any.whl (39 kB)
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 139.0/139.0 KB 29.5 MB/s eta 0:00:00
Collecting idna<4,>=2.5
  Downloading idna-3.3-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.2/61.2 KB 16.2 MB/s eta 0:00:00
Collecting certifi>=2017.4.17
  Downloading certifi-2022.6.15-py3-none-any.whl (160 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 160.2/160.2 KB 29.8 MB/s eta 0:00:00
Collecting python-dateutil<3.0.0,>=2.1
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 247.7/247.7 KB 39.0 MB/s eta 0:00:00
Collecting six>=1.5
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: urllib3, six, jmespath, idna, charset-normalizer, certifi, requests, python-dateutil, botocore, s3transfer, boto3
Successfully installed boto3-1.24.19 botocore-1.27.19 certifi-2022.6.15 charset-normalizer-2.1.0 idna-3.3 jmespath-1.0.1 python-dateutil-2.8.2 requests-2.28.1 s3transfer-0.6.0 six-
1.16.0 urllib3-1.26.9
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment i
nstead: https://pip.pypa.io/warnings/venv
WARNING: You are using pip version 22.0.4; however, version 22.1.2 is available.
You should consider upgrading via the '/usr/local/bin/python -m pip install --upgrade pip' command.
Removing intermediate container 418d7e90007b
 ---> e81c74097603
Step 4/4 : CMD python3 bootstrap.py
 ---> Running in 83471a9ff12a
Removing intermediate container 83471a9ff12a
 ---> ba6d314319f0
Successfully built ba6d314319f0
Successfully tagged lambda_ecr:latest
[ec2-user@ip-172-31-90-98 ~]$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED         SIZE
lambda_ecr   latest    ba6d314319f0   2 minutes ago   137MB
python       alpine    27edb73bd1fc   3 weeks ago     47.6MB
[ec2-user@ip-172-31-90-98 ~]$ aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 679034241394.dkr.ecr.us-east-1.amazonaws.com
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
[ec2-user@ip-172-31-90-98 ~]$ docker tag lambda_ecr:latest 679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr:latest
[ec2-user@ip-172-31-90-98 ~]$ docker push 679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr:latest
The push refers to repository [679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr]
e3f98a055008: Pushed
e2cd8f047283: Pushed
87652a1ad873: Pushed
9ad237c539b1: Pushed
24a6c9381586: Pushed
09c126bb3acd: Pushed
24302eb7d908: Pushed
latest: digest: sha256:6699ce580b65f2f48847f8b0c866aa8b4020d0b9cccf442b30e5b54ea19de9fc size: 1787
[ec2-user@ip-172-31-90-98 ~]$
```

Fig. 2 e) Pushing of image to ECR



Learner Lab - Foundational Servi
Amazon ECR
gmail login - Search
Inbox (3,217) - shubhangi.kharch
https://us-east-1.console.aws.amazon.com/ecr/repositories/private/679034241394/lambda_ecr?region=us-east-1
Services
Search for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1870159=shubhangi.kharche@gmail.com @ 6790-3424-1394
S3
RDS
Amazon Elastic Container Registry
Private registry
Public registry
Repositories
Summary
Images
Permissions
Lifecycle Policy
Tags
Getting started
Documentation
Public gallery
Amazon ECR > Repositories > lambda_ecr
View push commands
Edit
Delete
Scan
Find images
Feedback
Looking for language selection? Find it in the new Unified Settings
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

lambda_ecr

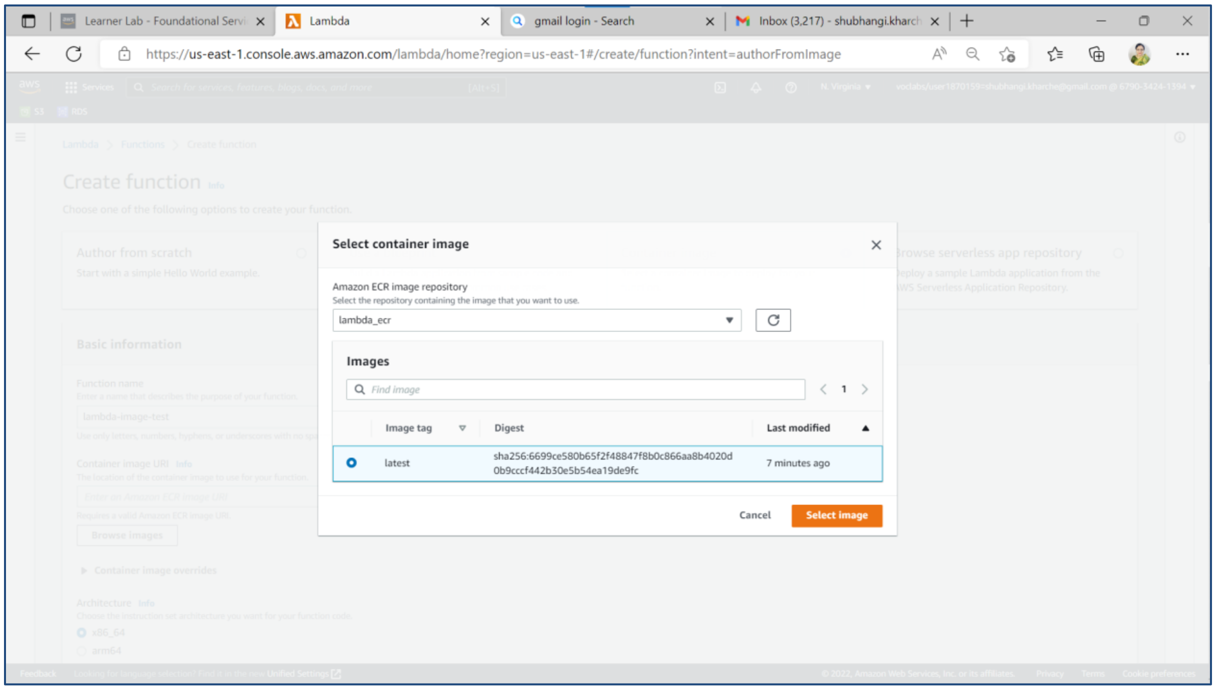
Images (1)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
latest	Image	June 29, 2022, 22:07:14 (UTC+05.5)	41.94	Copy URI	sha256:6699ce580b65f2f...	-	-

Fig. 2 f) Image uploaded on the ECR repo

Step 3: Creation of Lambda function to test the image

Step number	a
Step name	Create the Lambda function and test the image
Instructions	1) Navigate to the AWS Lambda service using the AWS Console 2) Click on **Create Function** 3) Under Create Function page select the 'Container image' option and enter a function name of your choice 4) For 'Container image URI' Click on "Browse Images" and select the repository and the image 5) Use the existing IAM role – LabRole. 6) Click on Create 7) Wait a few minutes for the function to be created 8) Test the function with the default "Hello World" test to see the result.
Expected screenshots	1) Container image selection 2) Execution role selection 3) Created function 4)Test result of function


https://us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/create/function?intent=authorFromImage

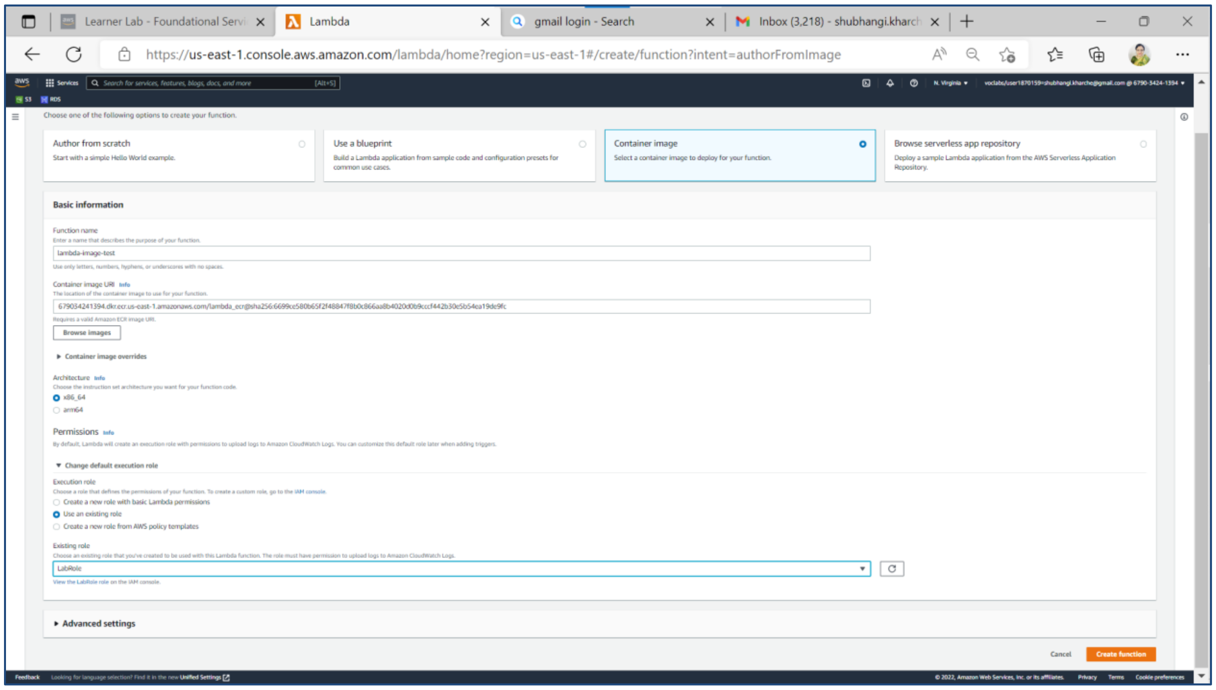
Select container image

Amazon ECR image repository
Select the repository containing the image that you want to use.
lambda_ecr

Images

	Image tag	Digest	Last modified
●	latest	sha256:6699ce580b65f2f48847f8b0c866aa8b4020d0b9cccf442b30e5b54ea19de9fc	7 minutes ago

Fig 3. a) Container image selection (latest)


https://us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/create/function?intent=authorFromImage

Choose one of the following options to create your function.

- Author from scratch — Start with a simple Hello World example.
- Use a blueprint — Build a Lambda application from sample code and configuration presets for common use cases.
- [✓ Container image] — Select a container image to deploy for your function.
- Browse serverless app repository — Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.
lambda-image-test
Use only letters, numbers, hyphens, or underscores with no spaces.

Container image URI
The location of the container image to use for your function.
679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr@sha256:6699ce580b65f2f48847f8b0c866aa8b4020d0b9cccf442b30e5b54ea19de9fc
Requires a valid Amazon ECR image URI.

Container image overrides

Architecture
Choose the instruction set architecture you want for your function code.
[✓ x86_64]
arm64

Permissions
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

Change default execution role

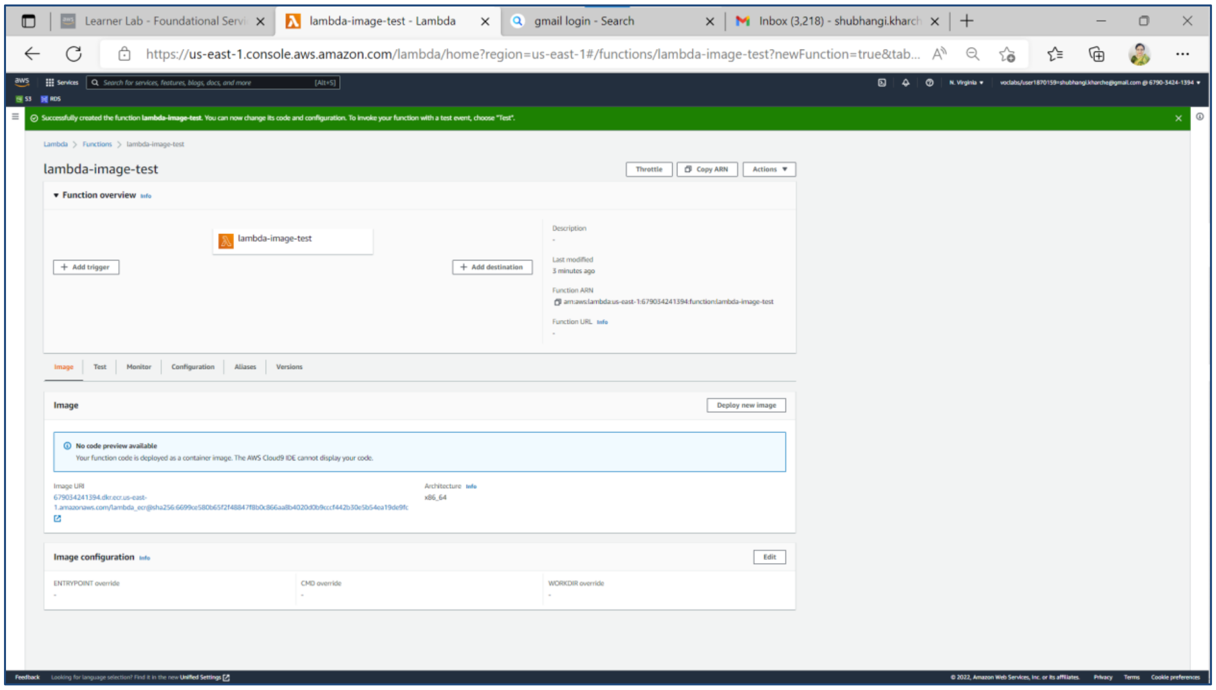
Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
Create a new role with basic Lambda permissions
[✓ Use an existing role]
Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
LabRole
View the LabRole role on the IAM console.

Advanced settings

Learner Lab - Foundational Servi
Lambda
gmail login - Search
Inbox (3,218) - shubhangi.kharch
Services
Browse images
Cancel
Create function
Feedback
© 2022, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Fig 3. b) Execution role selection (use an existing role)


https://us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/functions/lambda-image-test?newFunction=true&tab...

Successfully created the function **lambda-image-test**. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > lambda-image-test

lambda-image-test

Function overview

Description
-
Last modified
3 minutes ago
Function ARN
arn:aws:lambda:us-east-1:679034241394:function:lambda-image-test
Function URL
-

Image

No code preview available
Your function code is deployed as a container image. The AWS Cloud9 IDE cannot display your code.

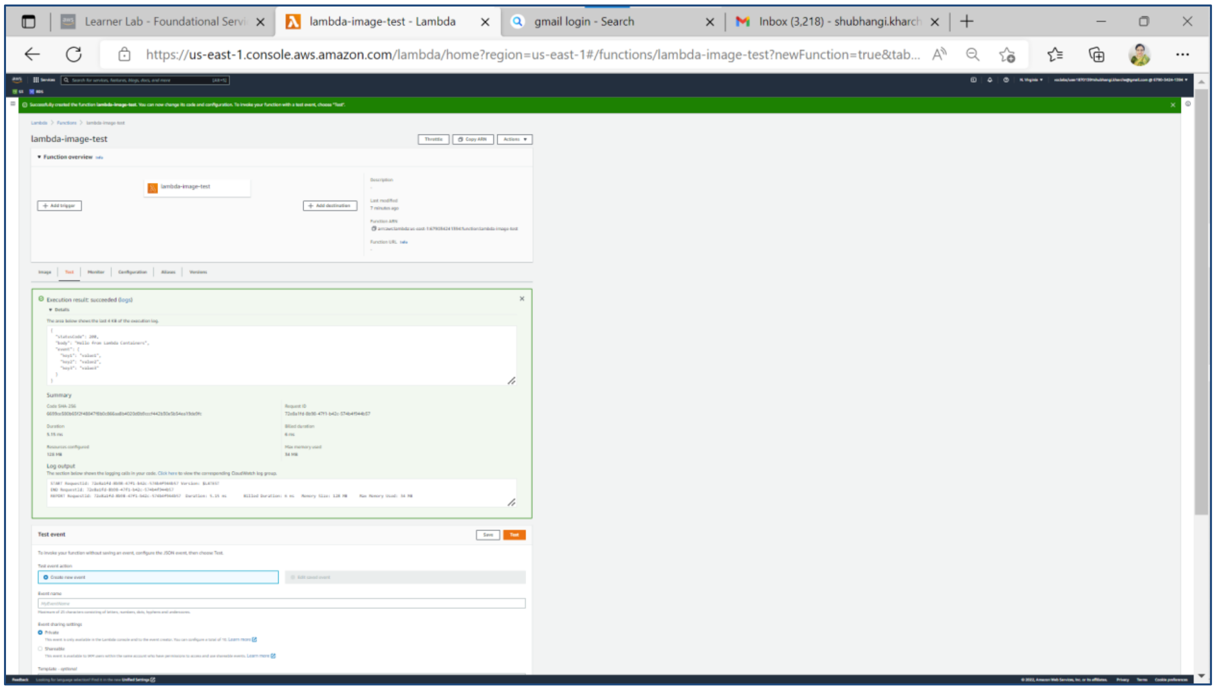
Image URI
679034241394.dkr.ecr.us-east-1.amazonaws.com/lambda_ecr@sha256:6699ce580b65f2f48847f8b0c866aa8b4020d0b9cccf442b30e5b54ea19de9fc

Architecture
x86_64

Image configuration

ENTRYPOINT override	CMD override	WORKDIR override
-	-	-

Fig 3. c) Created function (lambda-image-test)



```
{
  "statusCode": 200,
  "body": "Hello from Lambda Containers",
  "event": {
    "key1": "value1",
    "key2": "value2",
    "key3": "value3"
  }
}
```

Summary

Code SHA-256: 6699ce580b05f2f48047f6b0c866aa8b4020d0b6ccf442b50e5b54ea19de9fc

Request ID: 72e8a1fd-8b98-47f1-b42c-574b4f944b57

Duration: 5.15 ms

Billed duration: 6 ms

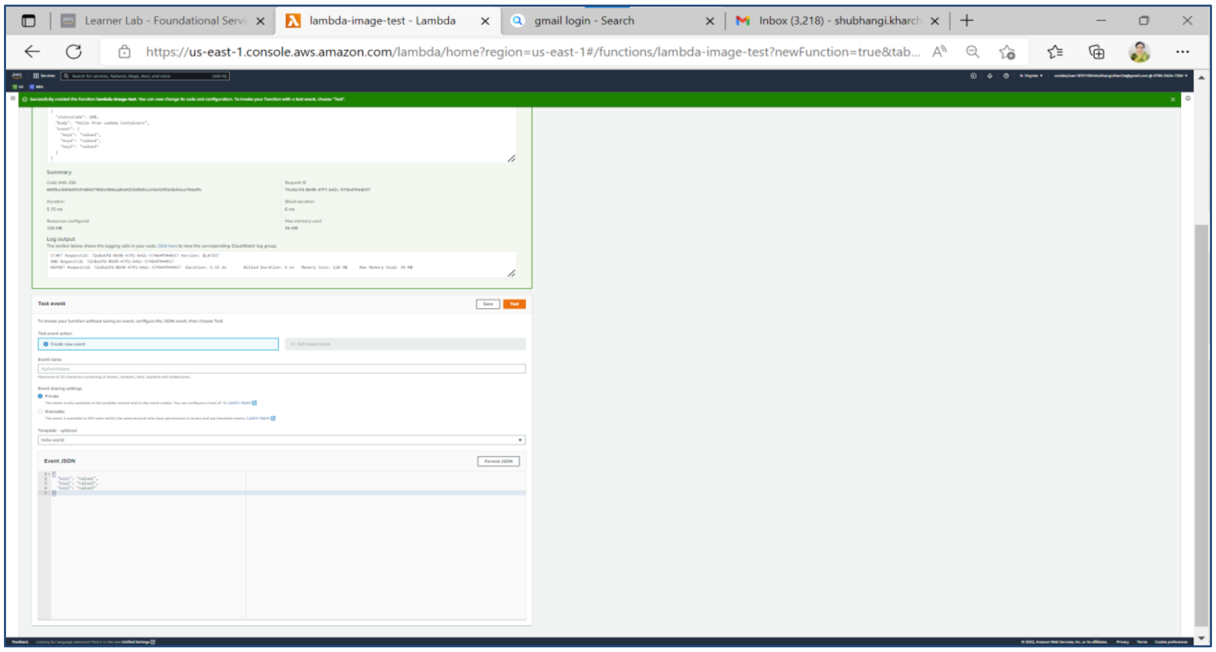
Resources configured: 128 MB

Max memory used: 34 MB

Log output

```
START RequestId: 72e8a1fd-8b98-47f1-b42c-574b4f944b57 Version: $LATEST
END RequestId: 72e8a1fd-8b98-47f1-b42c-574b4f944b57
REPORT RequestId: 72e8a1fd-8b98-47f1-b42c-574b4f944b57  Duration: 5.15 ms    Billed Duration: 6 ms   Memory Size: 128 MB    Max Memory Used: 34 MB
```

Fig 3. d)-1 Test result of function



```
{
  "key1": "value1",
  "key2": "value2",
  "key3": "value3"
}
```

Fig 3. d)-2 Test result of function

Answer the following questions		**Points**
Q1 How long does a container stay in the running state if it is not manually halted?		1
a) As long as the container’s PID 1 is running		
b) Has a set timeout after which it pauses		
c) Until its container is expunged		
d) Docker daemon process scheduler decides on load		
Enter your answer here	a) As long as the container’s PID 1 is running	
Q2 Which of the following best illustrates the relationship between an image and a container?		1
a) Executable and its hard link		
b) Executable and process		
c) Parent and child process		
d) Many to one		
Enter your answer here	b) Executable and process	
Q3 What is the maximum amount of RAM a container can consume if the memory flag is not used?		1
a) 8GiB		
b) 32GiB		
c) None of these		
d) As much as the host instance has free		
Enter your answer here	d) As much as the host instance has free	
Q4 Which of the following will happen in the same Docker image is pushed to Docker Hub multiple times with different tags		1
a) Dockerhub will refuse to upload the image		
b) The layers in the first image (if unchanged) will be reused in subsequent pushes		
c) Dockerhub will merge the images		
d) The same image cannot have multiple tags		
Enter your answer here	c) Dockerhub will merge the images	

Q5 Which of the following will run a Docker container in interactive mode? 1

a) -v

b) -it

c) -b

d) -u

Enter your answer here

b) -it

Q6 How would data persistence be handled in a container environment set up for autoscaling? 4

Data persistence in containers?

- A container can be in running state as long as its process is running.
- Data persists if it outlasts the process that created it.
- **By default, containers don't persist the data they produce.**

How to provide data persistence in containers?

- Containers are suitable for stateless applications because when they are terminated the data as well is destroyed.
- Data persistence can be handled in containerized applications with the storage backend that isn't destroyed when container terminates.
- Containerized autoscaling applications requiring stateful containers can be developed using AWS elastic container service (ECS) that can use the aws storage services to provide data persistence to inherently ephemeral containers.
- The suitable aws storage services to provide data persistence are EBS, EFS or FSx for windows file server.

Now, how aws ECS can provide data persistence?

- Data persistence in AWS is achieved by coupling compute and storage services.
- Like EC2, ECS can be used the decouple the lifecycle of containerized applications from the data they consume and produce.
- Using AWS storage services, ECS tasks can persist data even after tasks terminate.

Q7 Why is this statement false? "Docker is the only popular choice for microservices 4 deployment".

Containers are a means to deploy microservice architectures and Docker is a well-known containerization platform.
Though Docker is a popular technology to deploy microservices but it not the most popular or only one due to its following drawbacks:

- Containers don't run at bare-metal speeds. Containers consume resources more efficiently than virtual machines.
- The container ecosystem is fractured.
- Persistent data storage is complicated.
- Graphical applications don't work well.
- Not all applications benefit from containers.

Moreover, Microservices are about logical separation, not physical.

- it is not mandatory to use Docker in microservices architecture.
- We can design our system/Application and use microservices architecture and the final deployment can be pure hardware.

At the end, a microservice can be treated as a process that needs a host to run.
There are more other ways to deploy microservices:
REST is an architectural design pattern for building RESTful APIs. REST allows services to communicate directly via HTTP. Requests and responses are handled in standard formats like XML, HTML, or JSON. REST is a natural choice for most microservices, since many of them are Web Applications.

Redis, Prometheus and Consul with their associated set of advantages are some other tools to deploy microservices

---