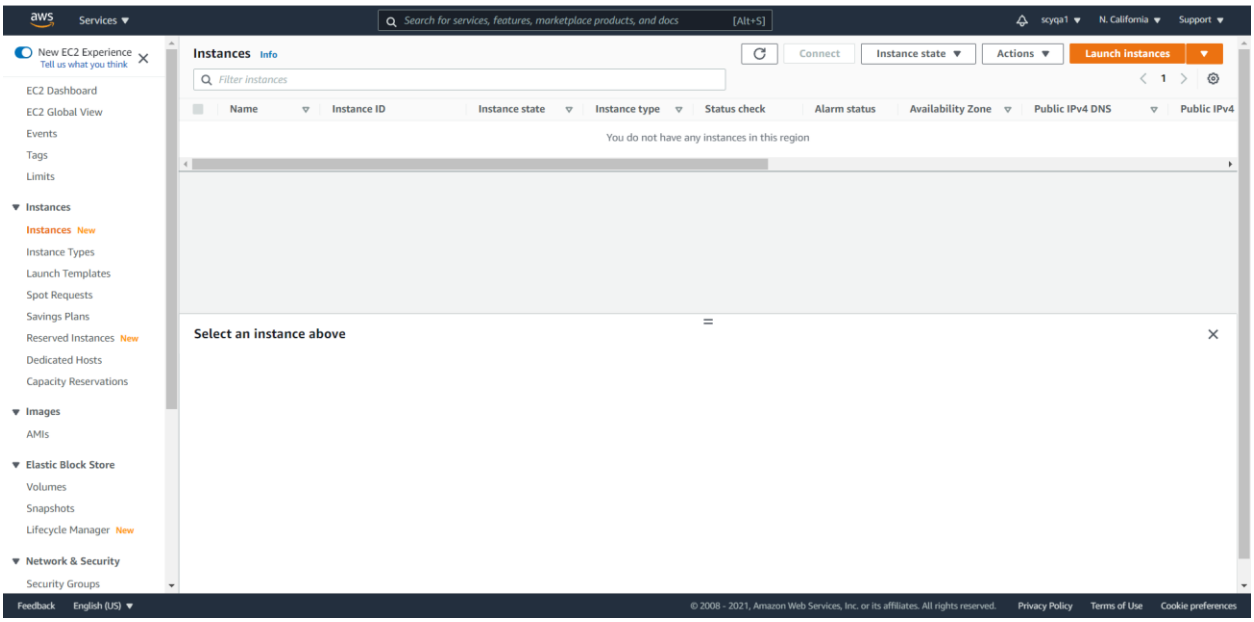
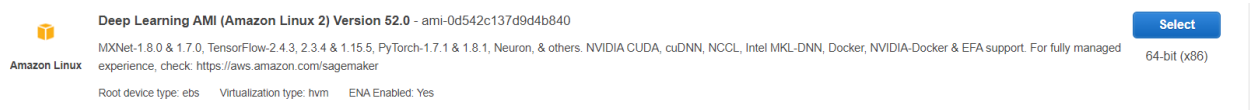
PART I: CI/CD With AWS ECS and Docker [100 points]

1) Docker Basics: Create EC2 Instance. Install Docker. Pull centos:centos6. Create index.html with your greeting. Create Dockerfile. Build a Docker image

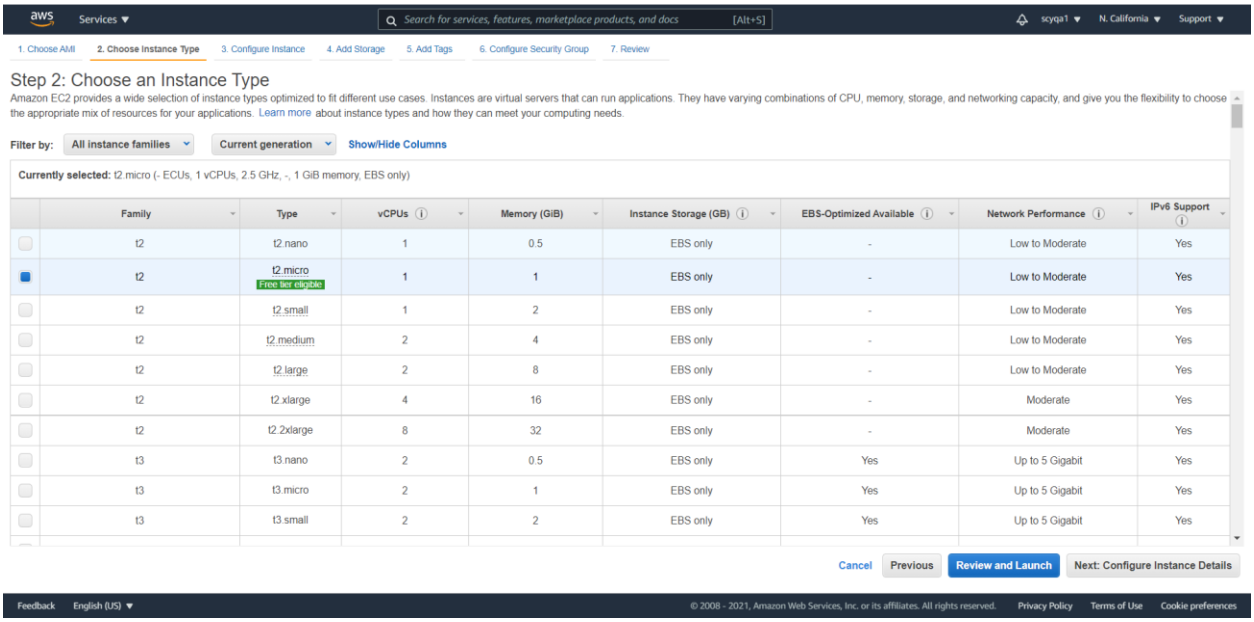
Launch an instance



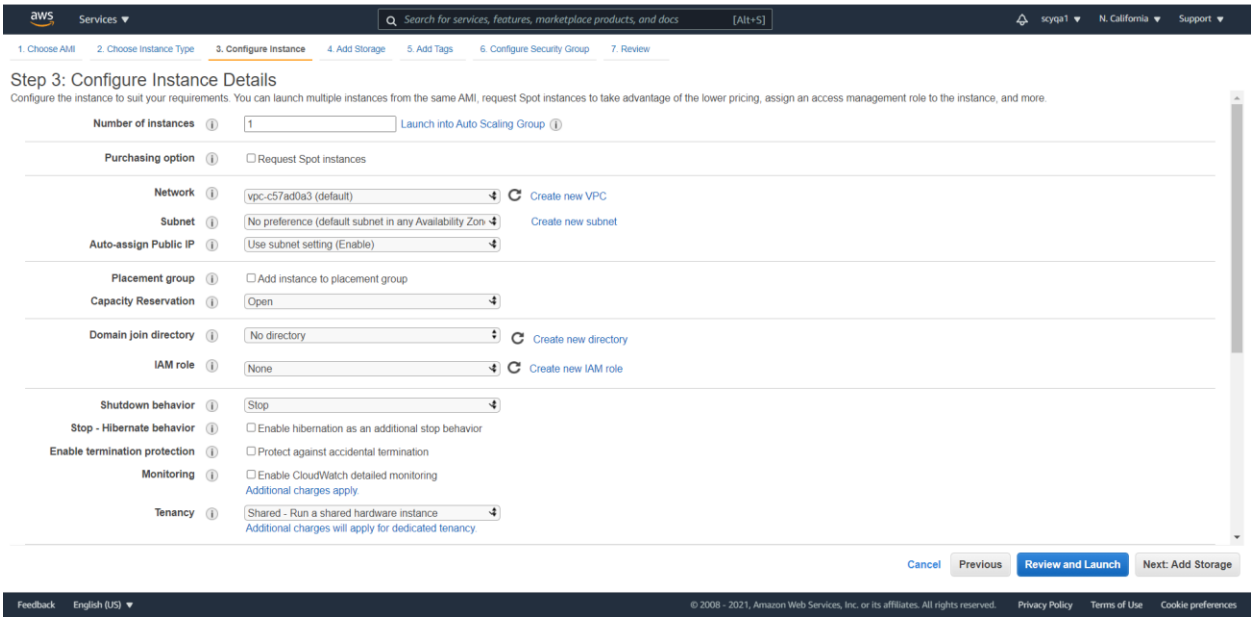
Choose AMI



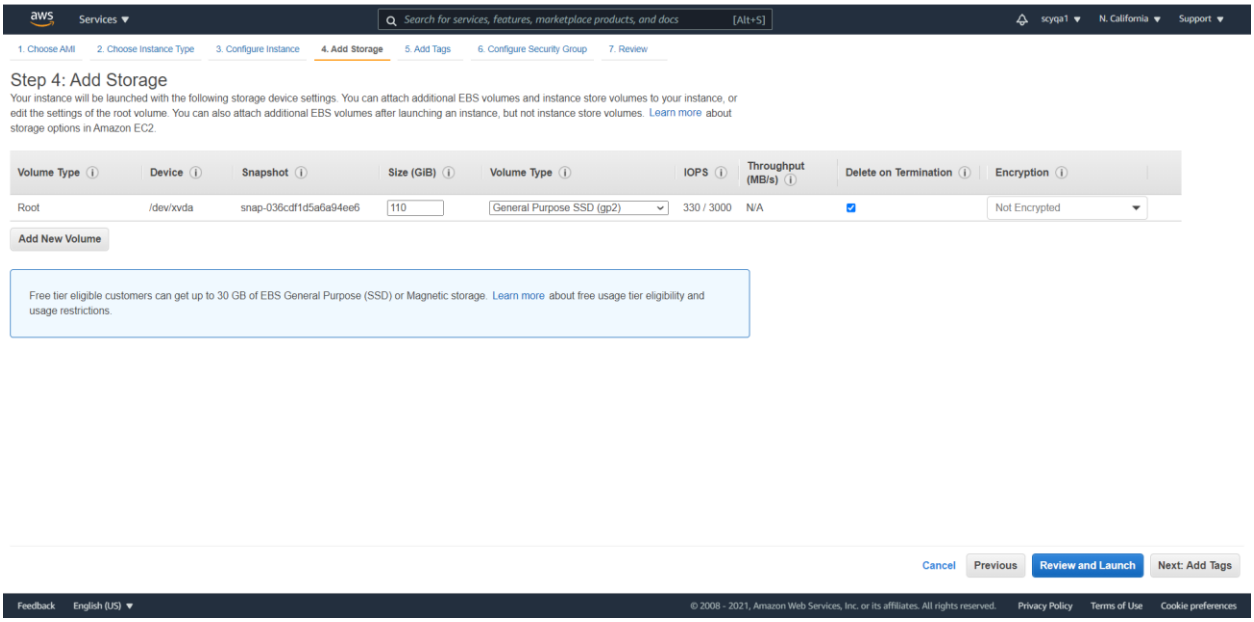
Choose an instance type



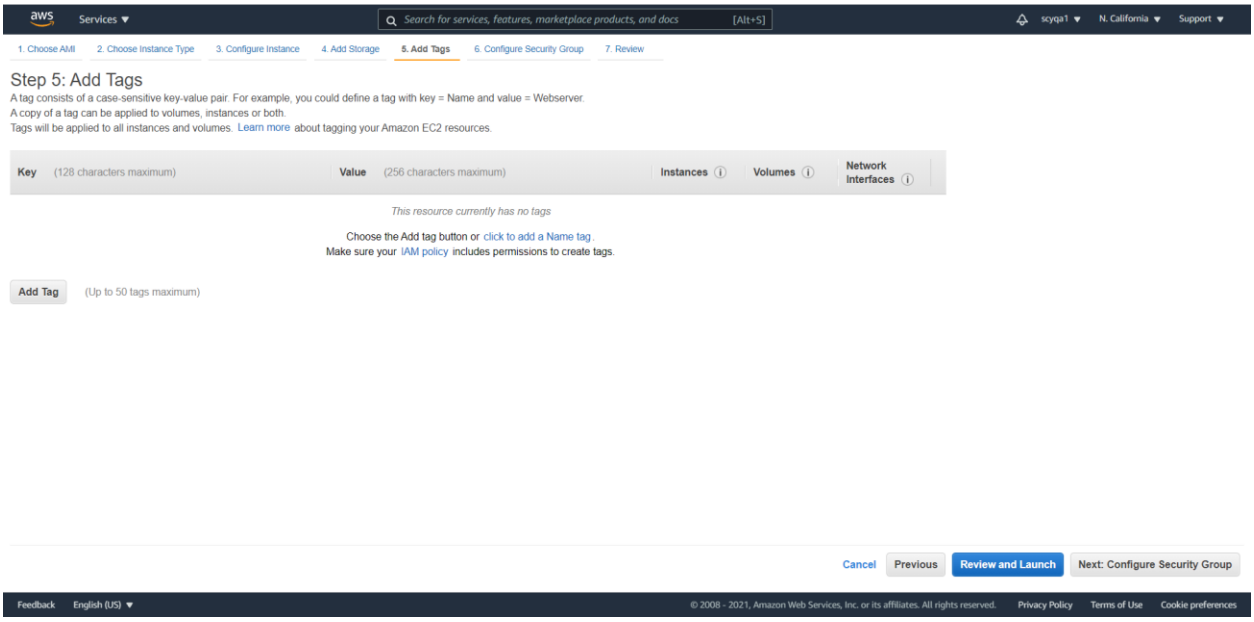
Configure instance details



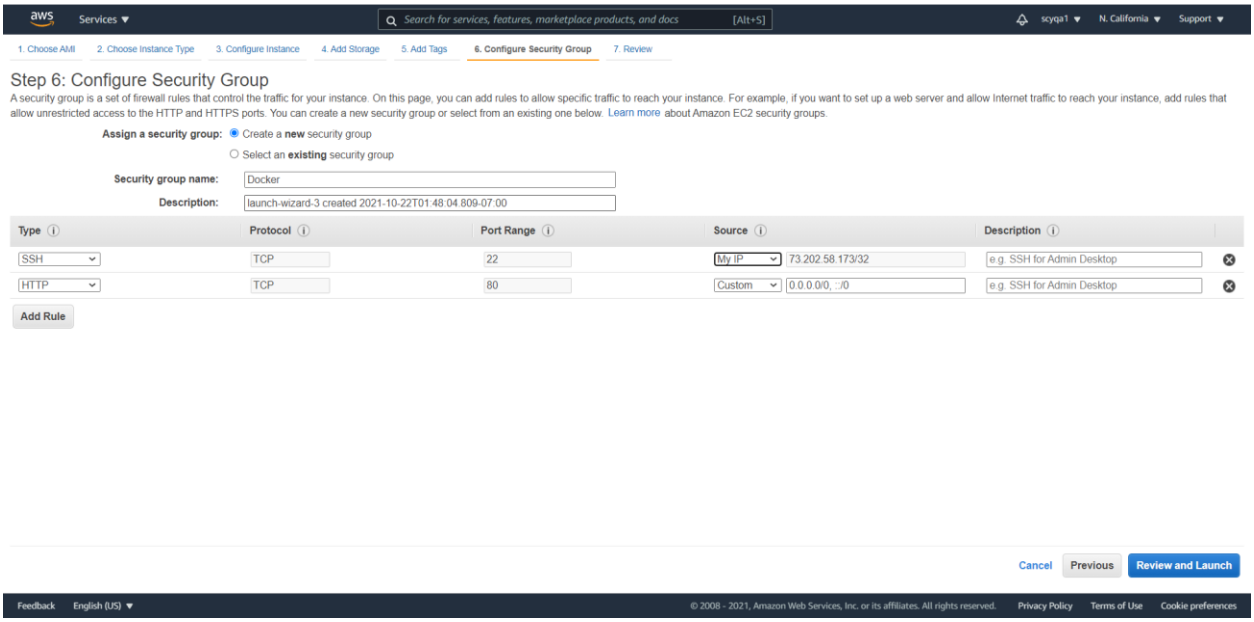
Add storage



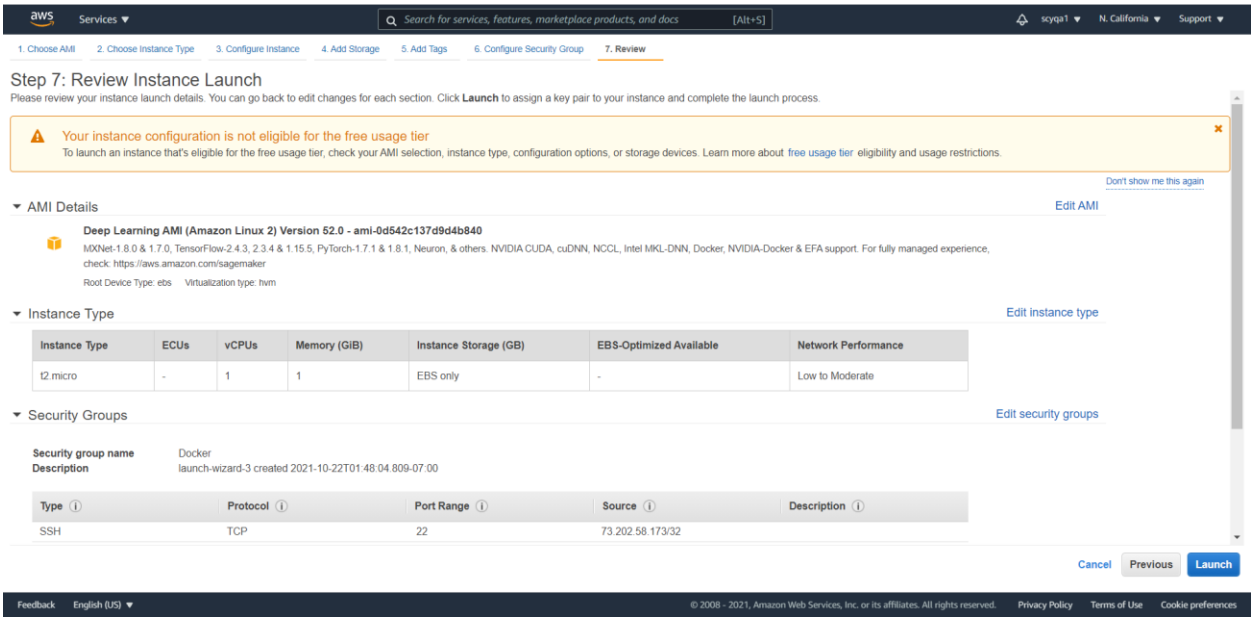
Add tags



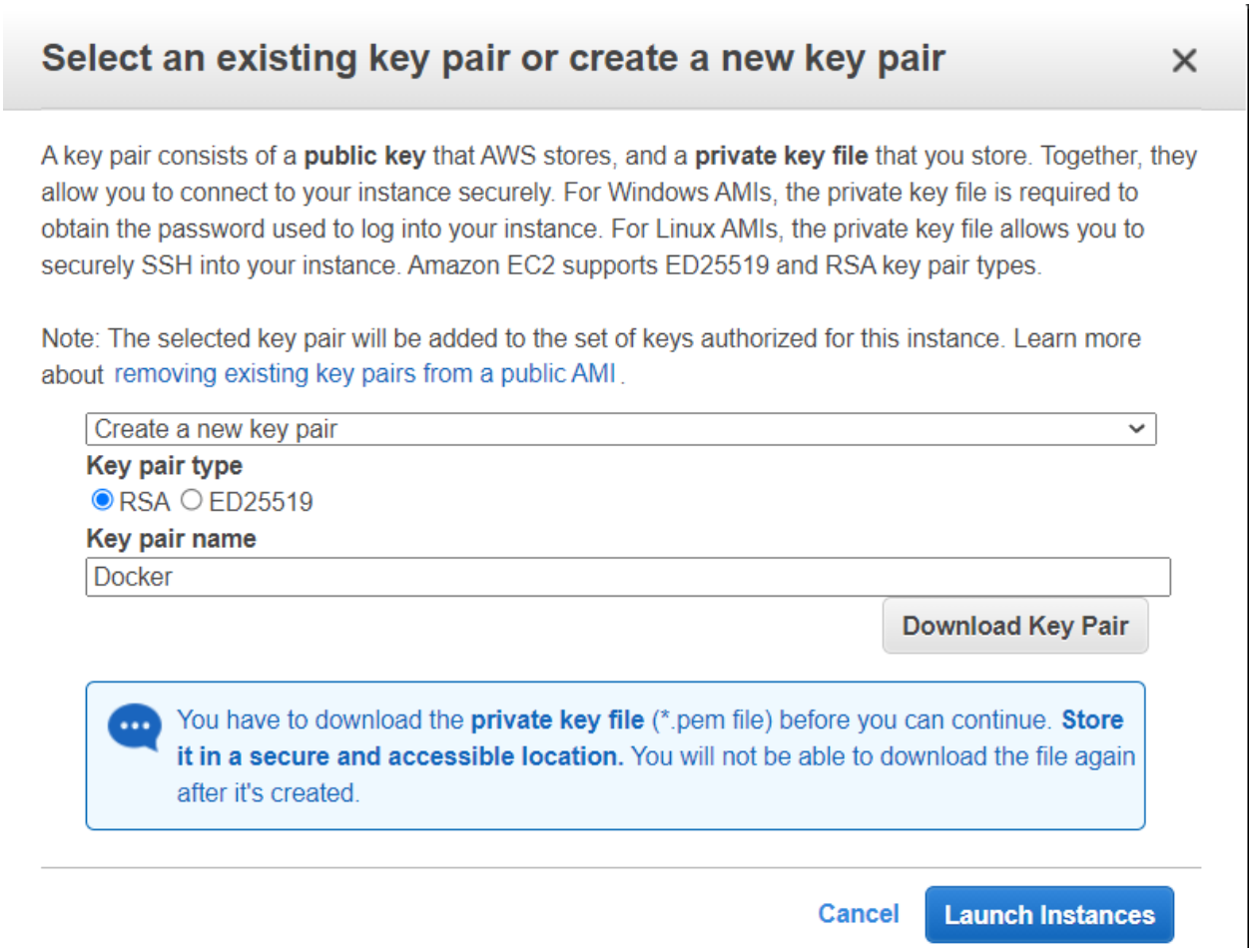
Configure security group



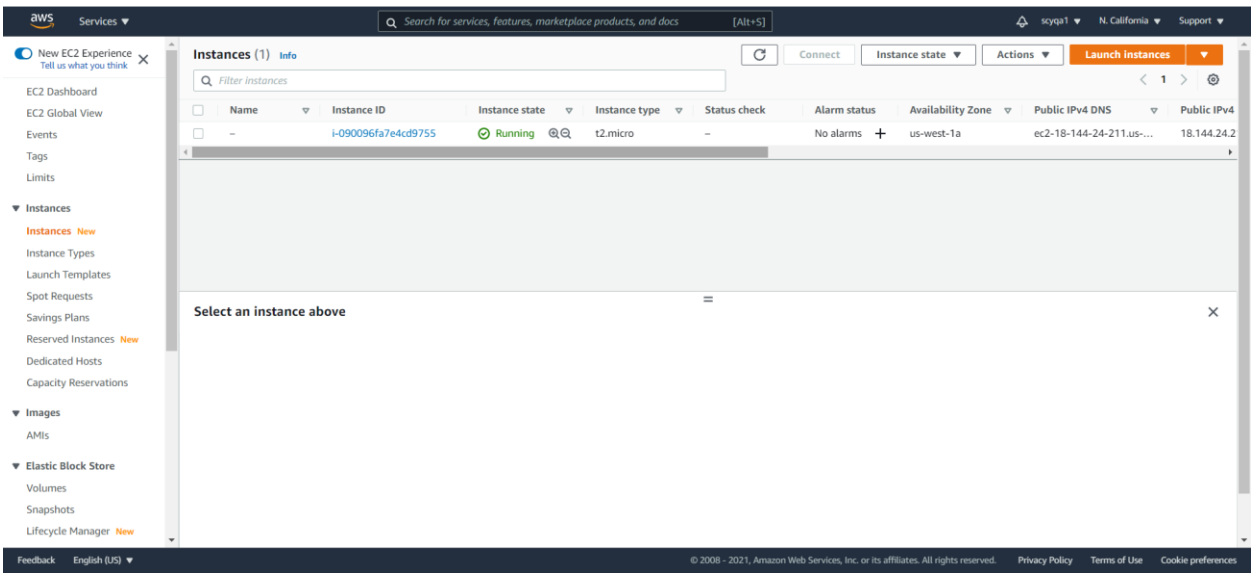
Review instance launch



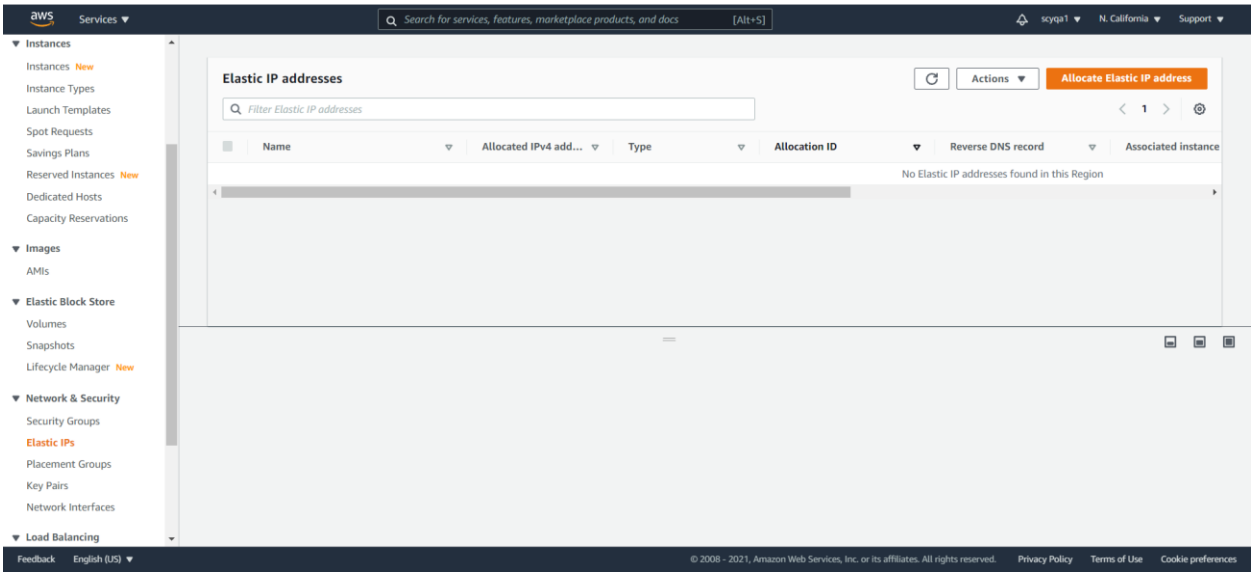
Create a new key pair

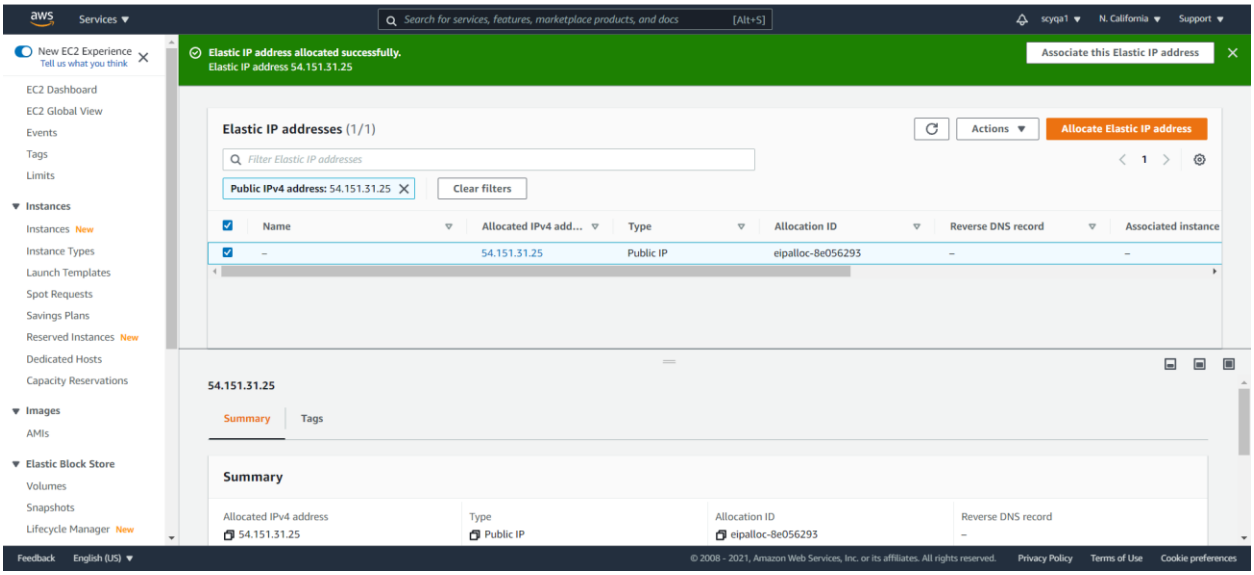


Now, the instance is created

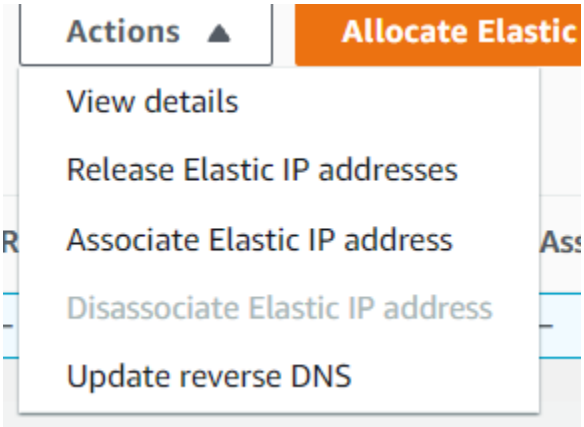


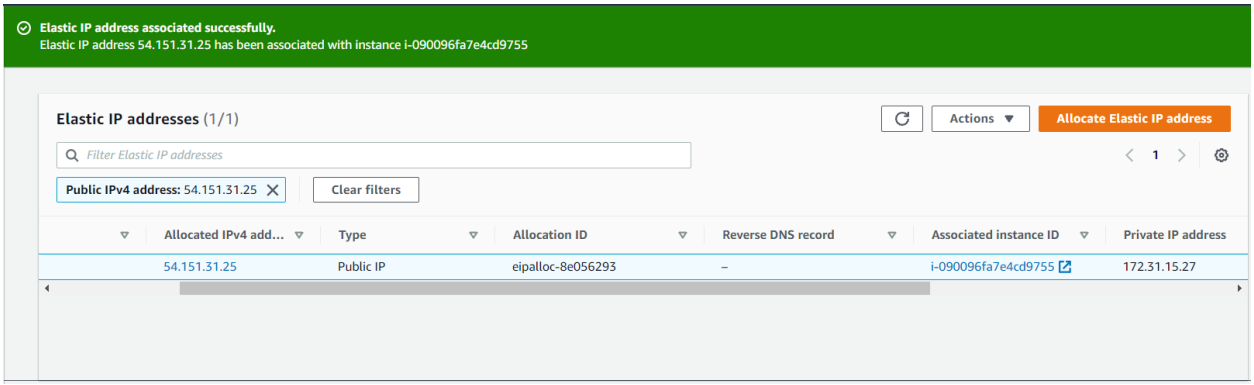
Allocate elastic IP address



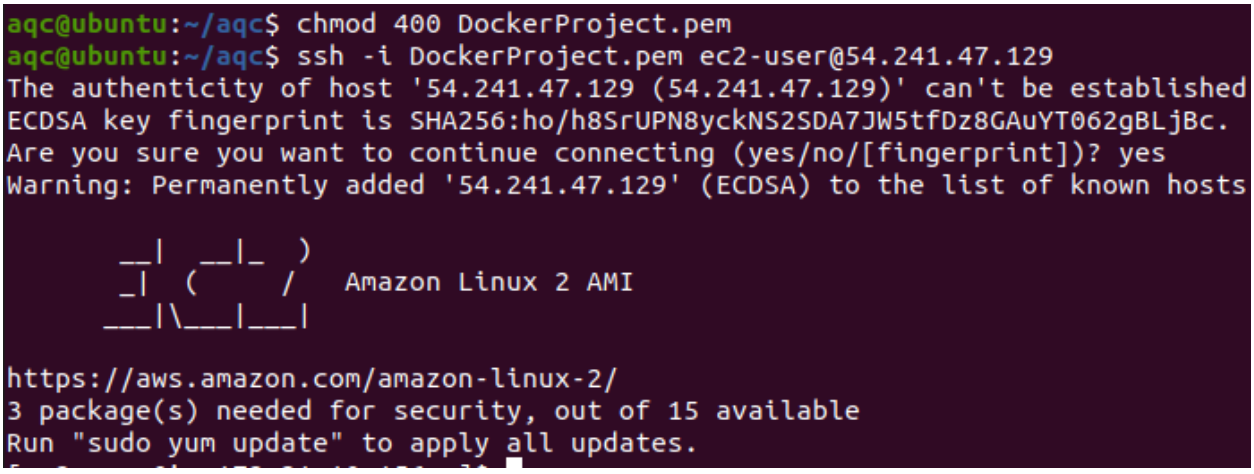


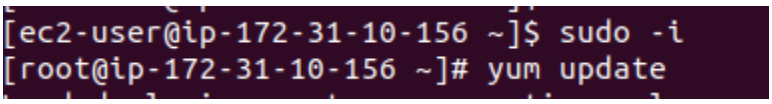
Associate elastic IP address





SSH connection to EC2





Install Docker

```
[root@ip-172-31-10-156 ~]# yum install docker -y
```

```
[root@ip-172-31-10-156 ~]# service docker start
```

Pull centos:centos6

```
[root@ip-172-31-10-156 ~]# docker pull centos:centos6
```

Docker images

```
[root@ip-172-31-10-156 ~]# docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
centos       centos6   5bf9684f4720   5 weeks ago   194MB
```

Docker run

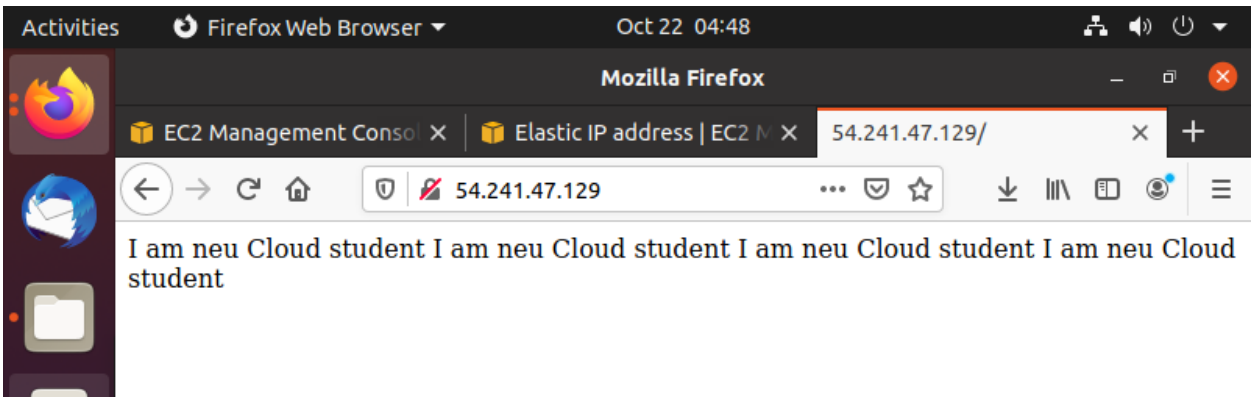
```
[root@ip-172-31-10-156 ~]# docker run -it -p 80:80 5bf9684f4720
[root@431a0adf7580 /]#
```

Install httpd

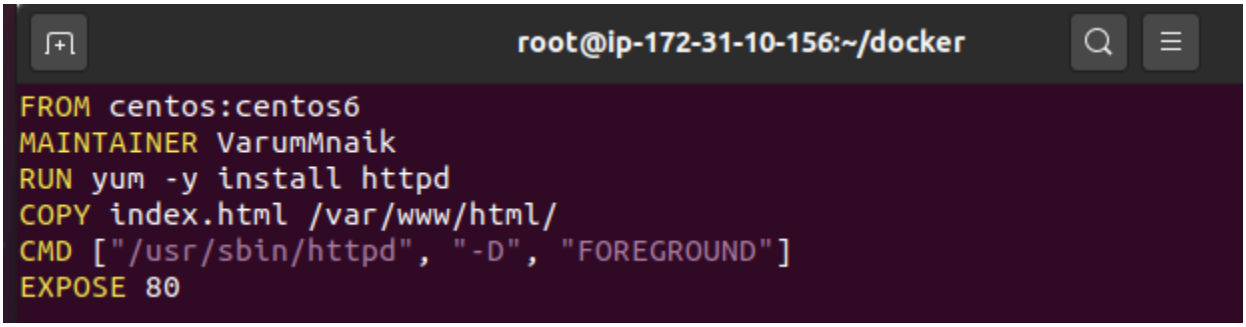
```
[root@431a0adf7580 /]# yum install httpd
```

Create index.html with my greeting

```
[root@431a0adf7580 /]# cd /var/www/html
[root@431a0adf7580 html]# echo "I am neu Cloud student" >> index.html
[root@431a0adf7580 html]# service httpd start
Starting httpd: httpd: Could not reliably determine the server's fully qualifie
d domain name, using 172.17.0.2 for ServerName
                                                           [  OK  ]
```

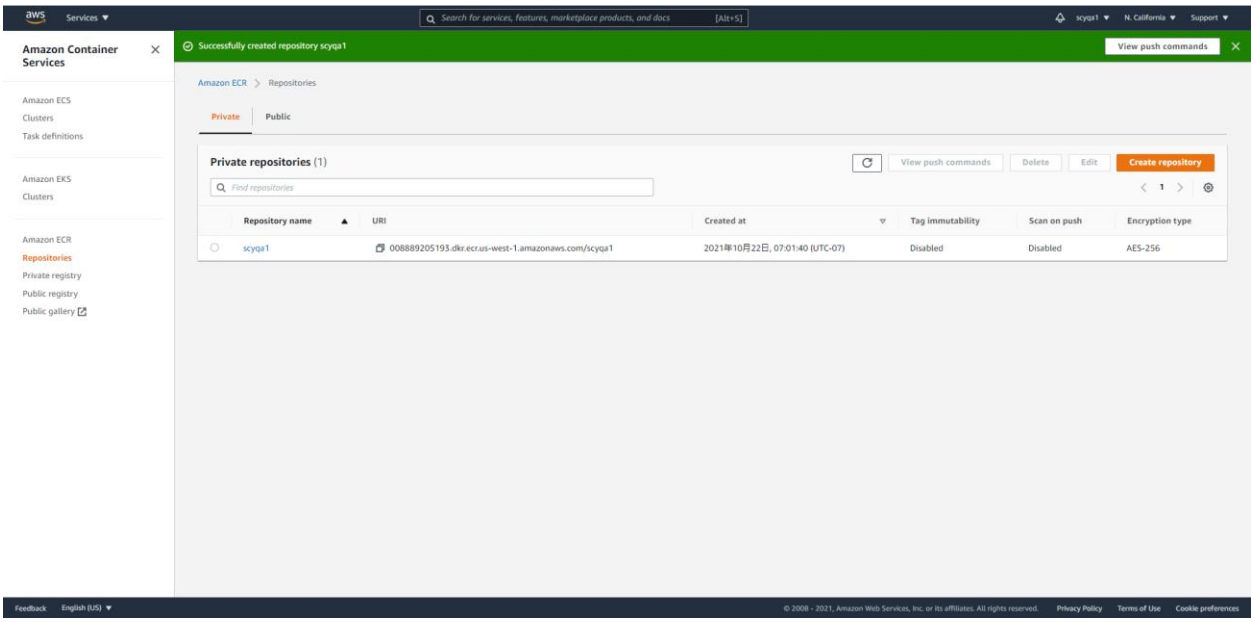


Create Dockerfile.

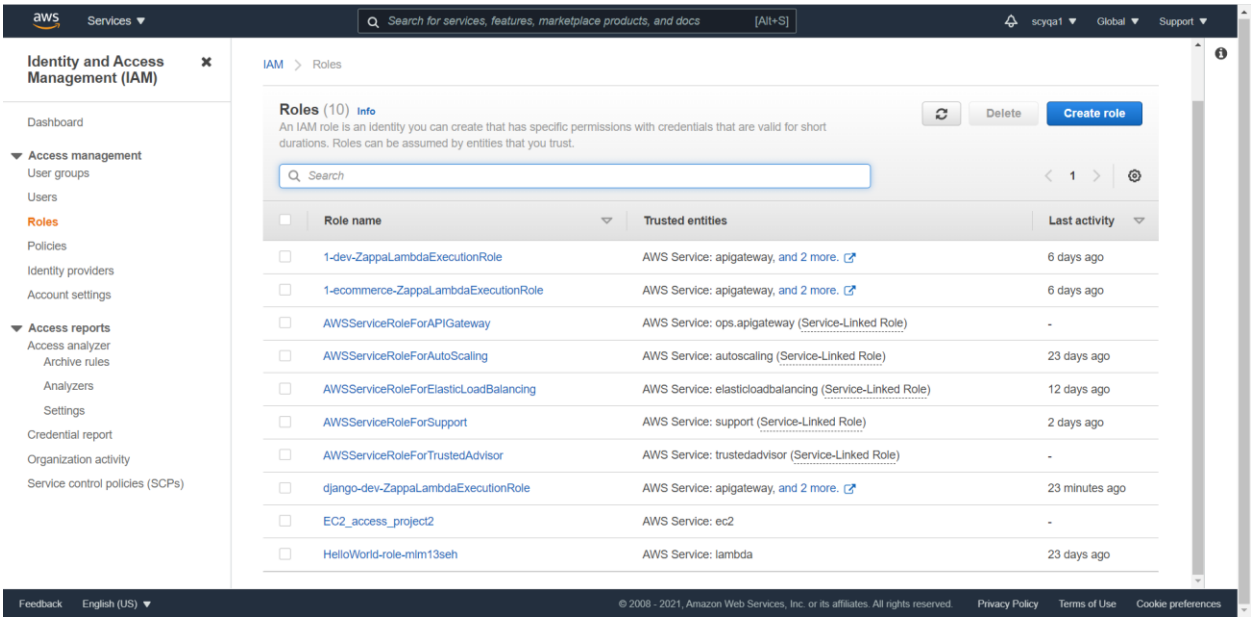


2) [17:57] AWS ECR(Elastics Container Registry) Creation: Create AWS ECR. Login to ECR. Tag existing image as AWS ECR repo. Push the image into the ECR

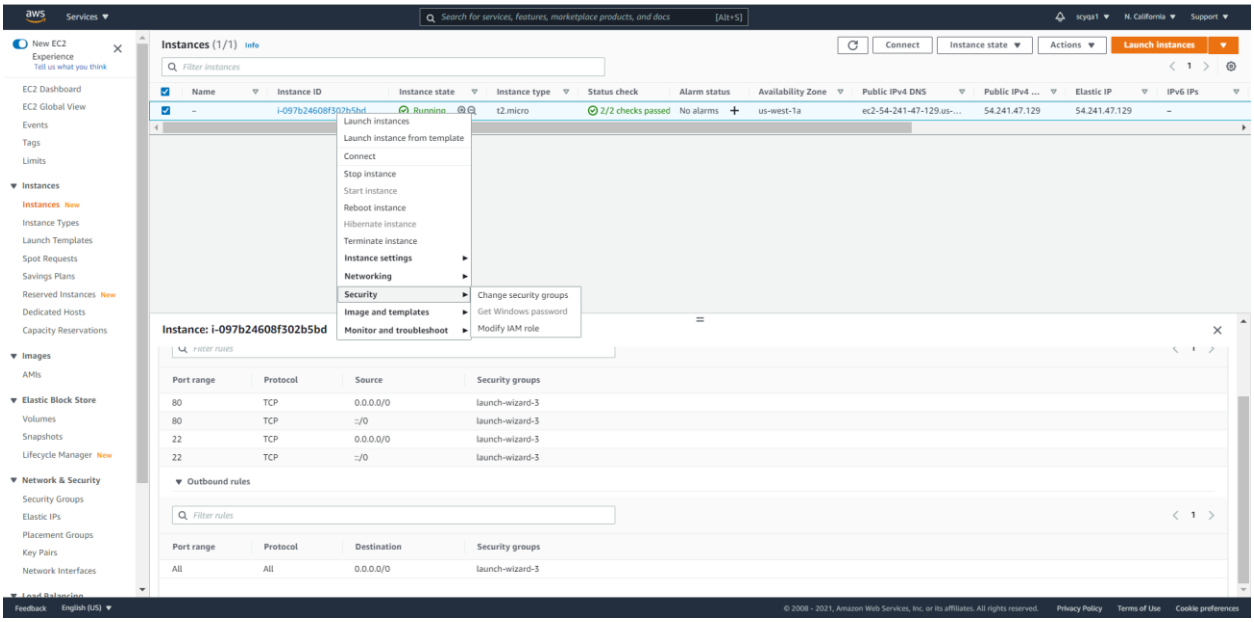
Create ECR



Create a role



Modify EC2's IAM role



Retrieve an authentication token and authenticate the Docker client to registry using the AWS CLI:

```
[root@ip-172-31-10-156 docker]# aws ecr get-login-password --region us-west-1 | docker login --user
name AWS --password-stdin 008889205193.dkr.ecr.us-west-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

After the build completes, tag the image

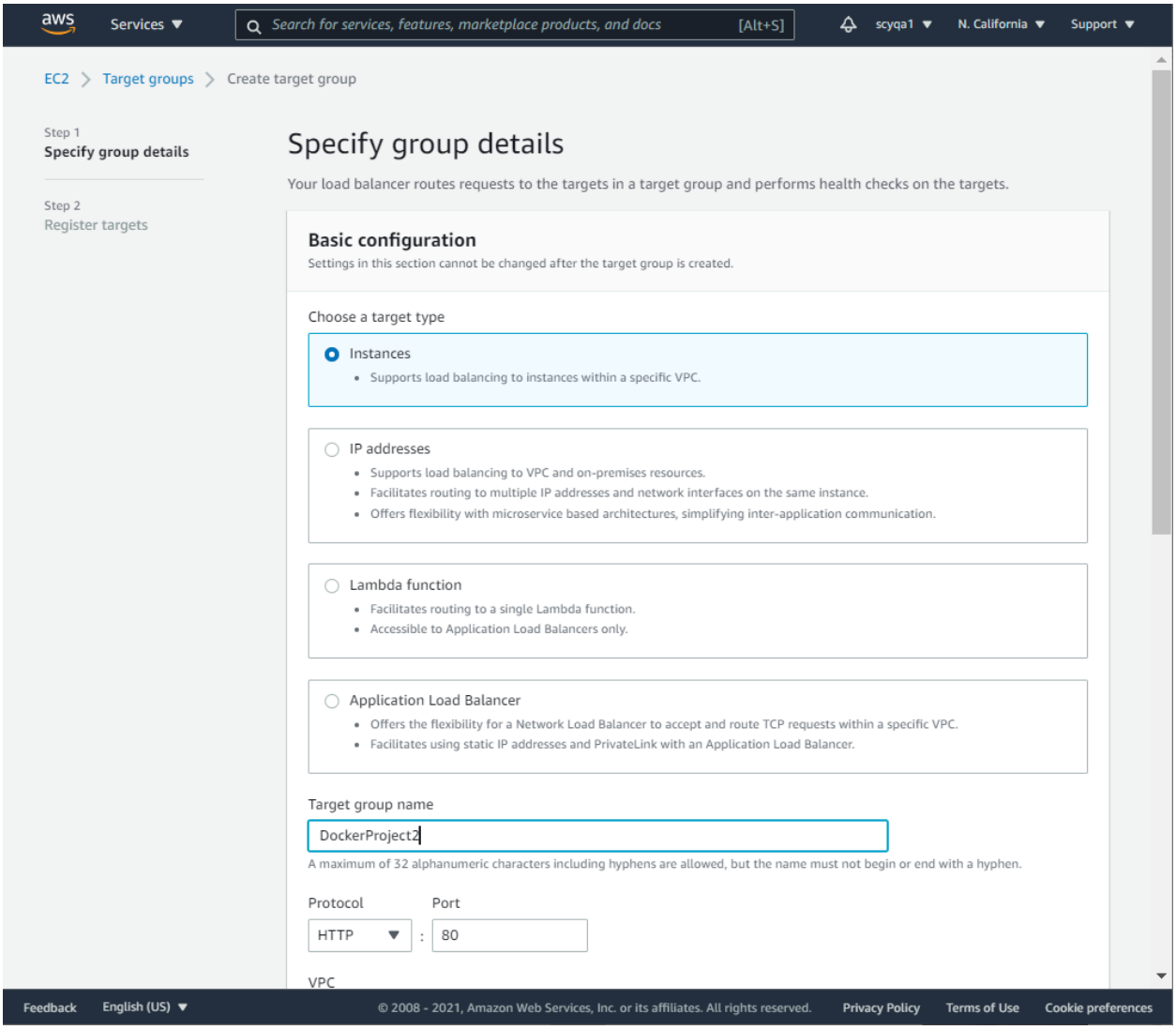
```
[root@ip-172-31-10-156 docker]# docker tag centos:centos6 008889205193.dkr.ecr.us-west-1.amazonaws.
com/scyqa1:latest
```

push the image to newly created AWS repository

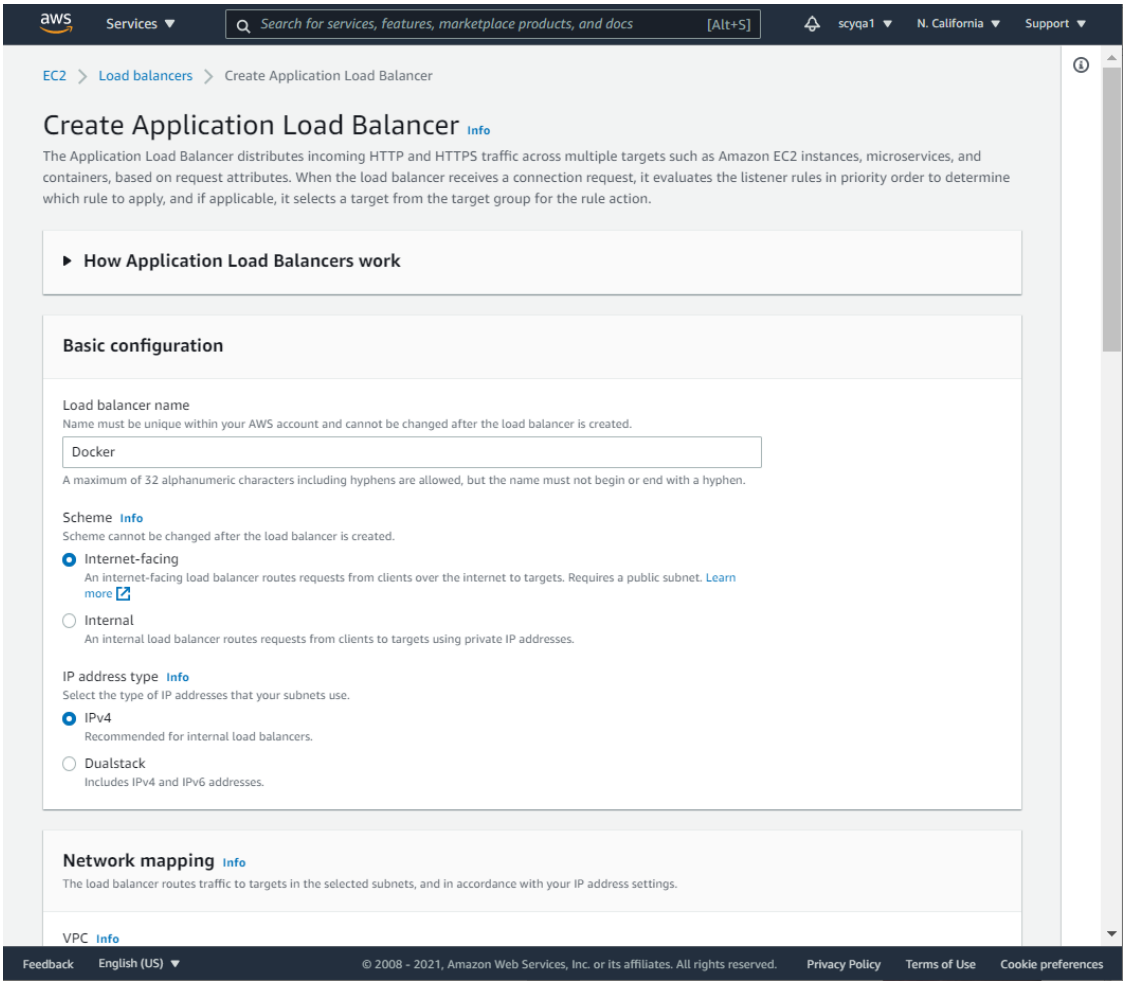
```
[root@ip-172-31-10-156 docker]# docker tag centos:cenos6 008889205193.dkr.ecr.us-west-1.amazonaws.c
om/scyqa1:latest
Error response from daemon: No such image: centos:cenos6
[root@ip-172-31-10-156 docker]# docker tag centos:centos6 008889205193.dkr.ecr.us-west-1.amazonaws.
com/scyqa1:latest
[root@ip-172-31-10-156 docker]# docker push 008889205193.dkr.ecr.us-west-1.amazonaws.com/scyqa1:lat
est
The push refers to repository [008889205193.dkr.ecr.us-west-1.amazonaws.com/scyqa1]
af6bf1987c2e: Pushed
latest: digest: sha256:3688aa867eb84332460e172b9250c9c198fdfd8d987605fd53f246f498c60bcf size: 529
```

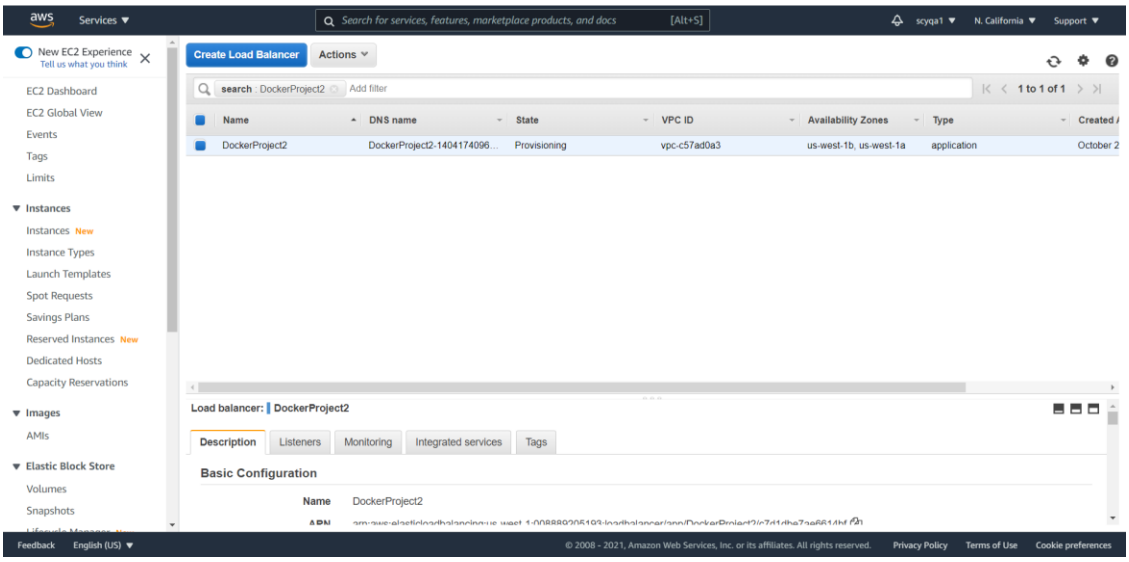
3) Create ALB(Application Load Balancer): Create Target group. Create ALB

Create target group



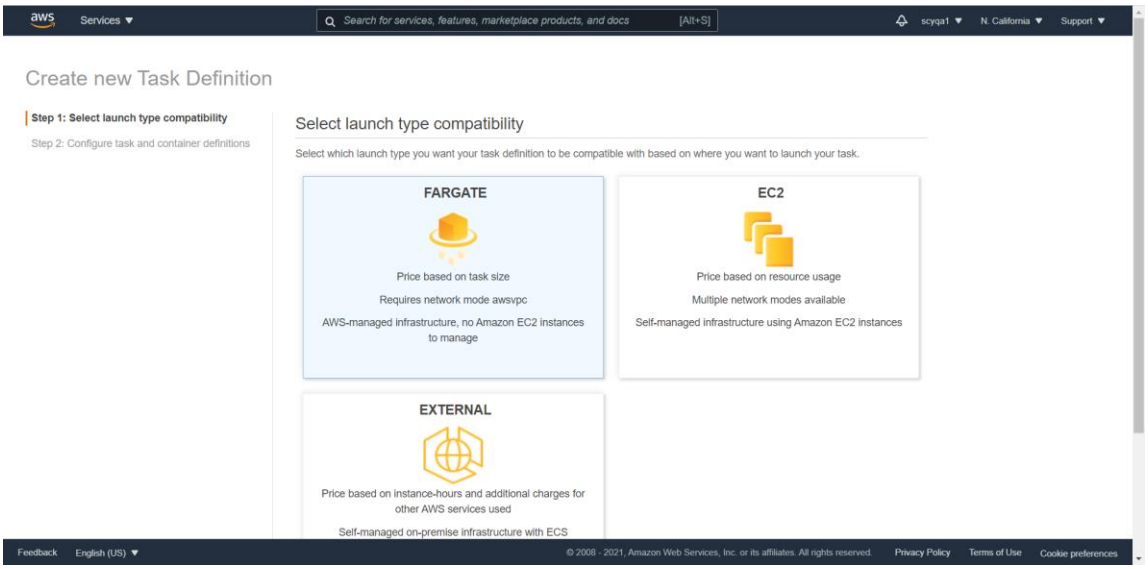
Create application load balancer



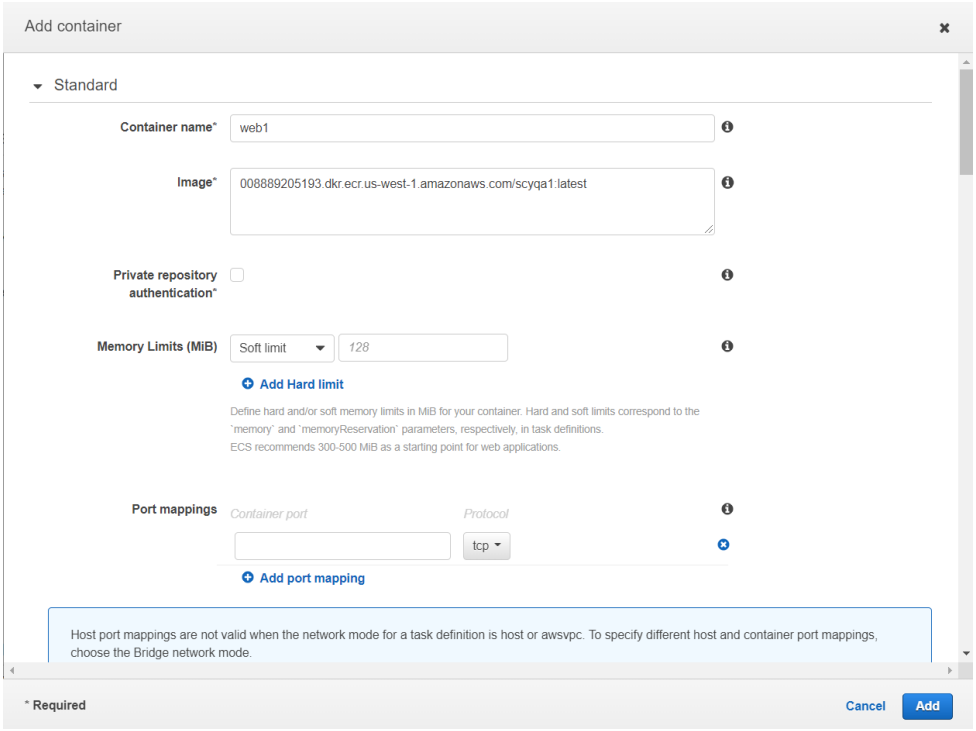


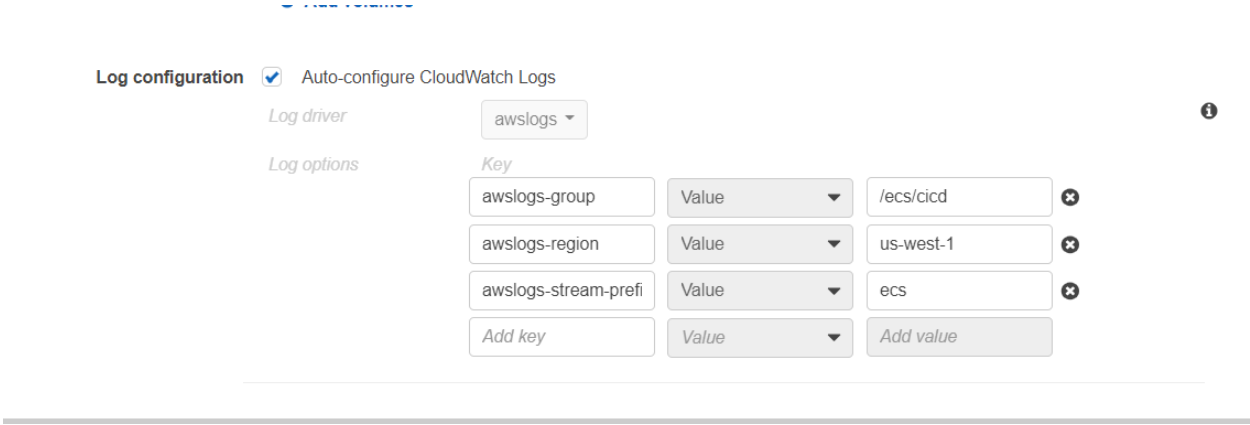
4) Create an AWS ECS(Elastics Container Service) Cluster: Create a Task with Fargate Computability. Create a Fargate Cluster. Create a Service with ALB enabled. Test ALB DNS, whether your site is running or not. Add your container with auto-config CloudWatch Logs

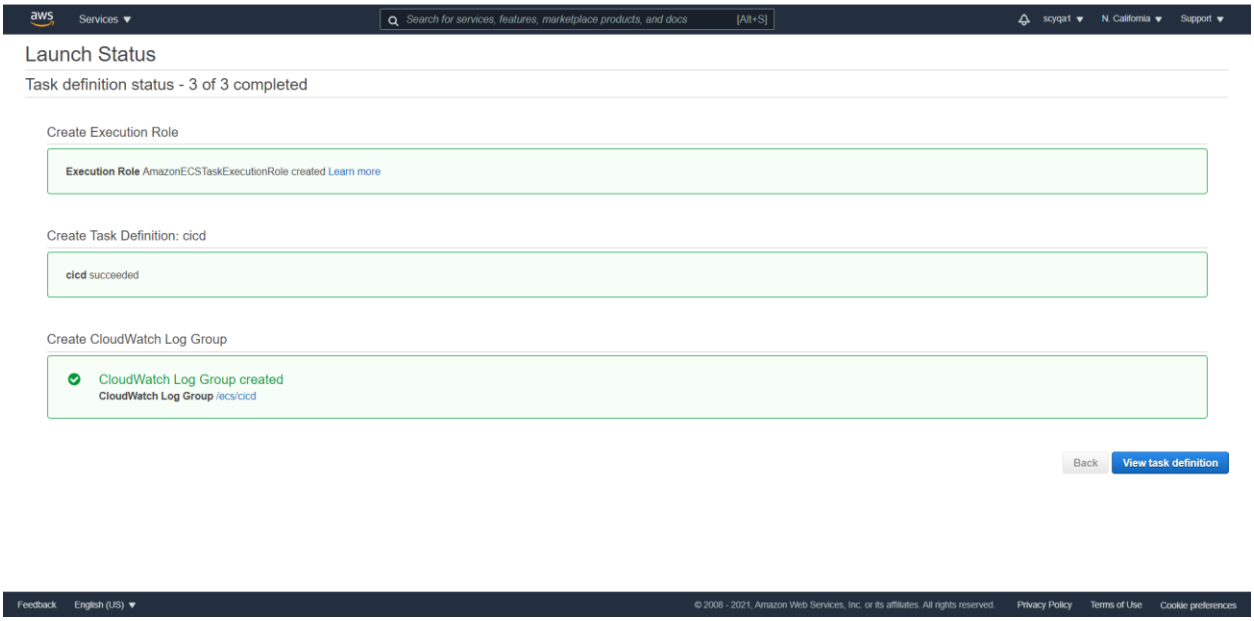
Create new Task definition



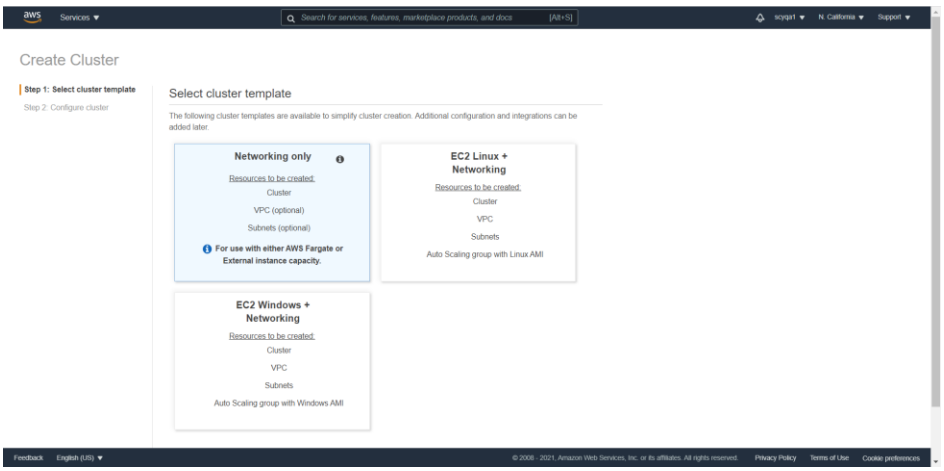
Add container



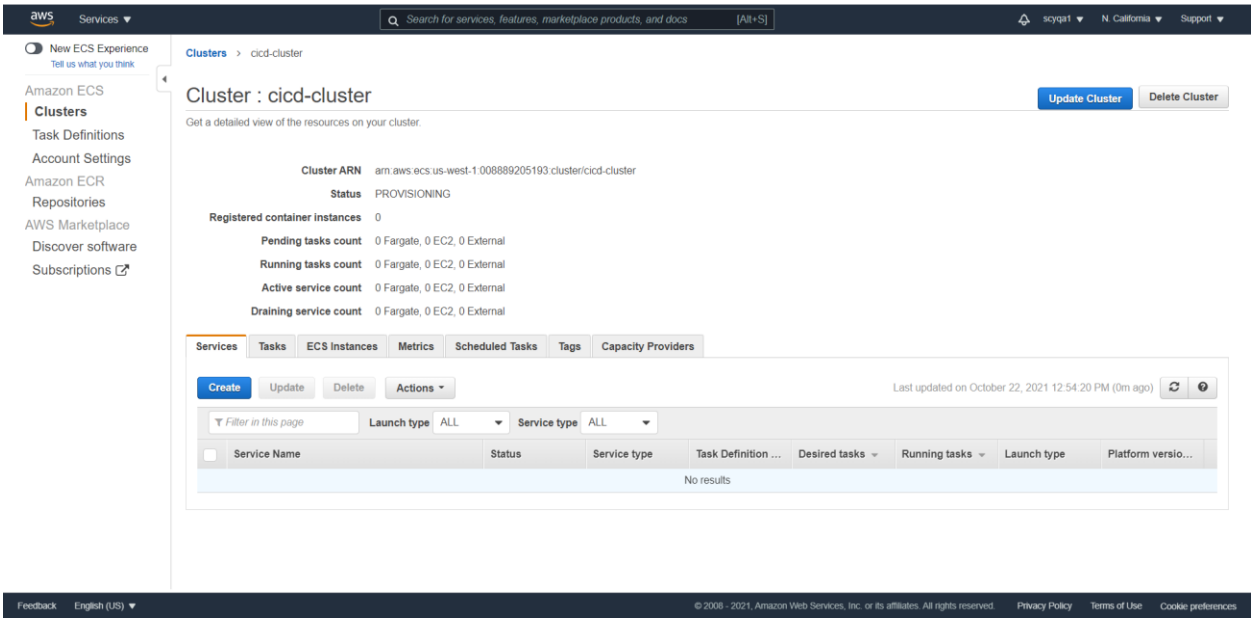


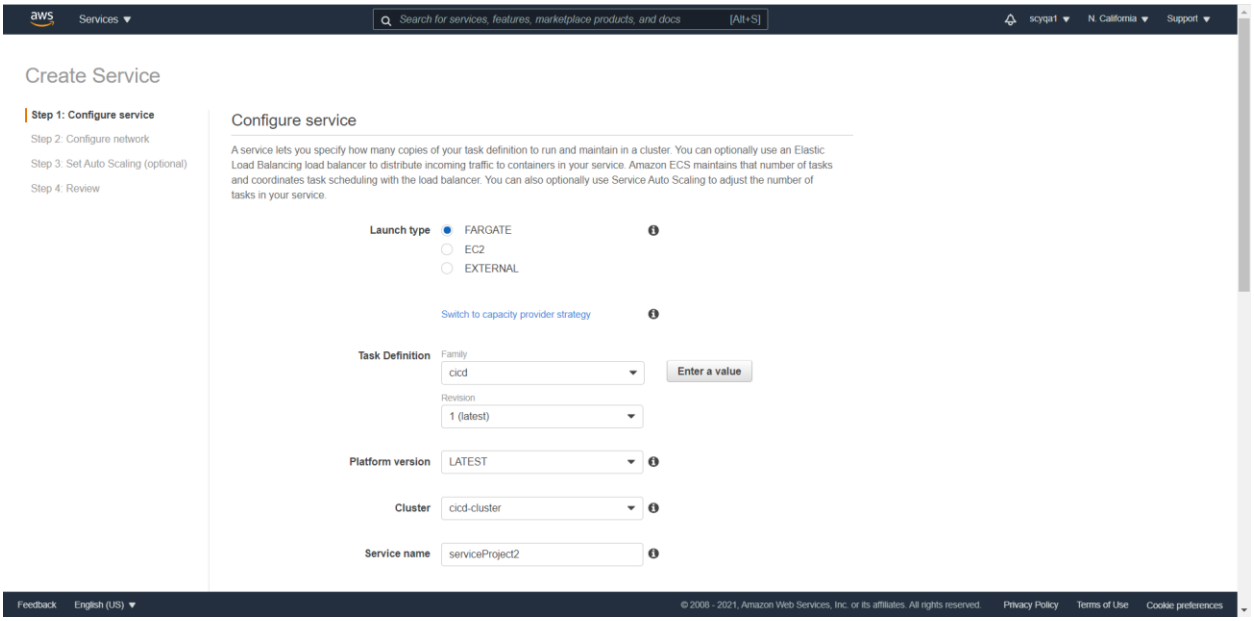


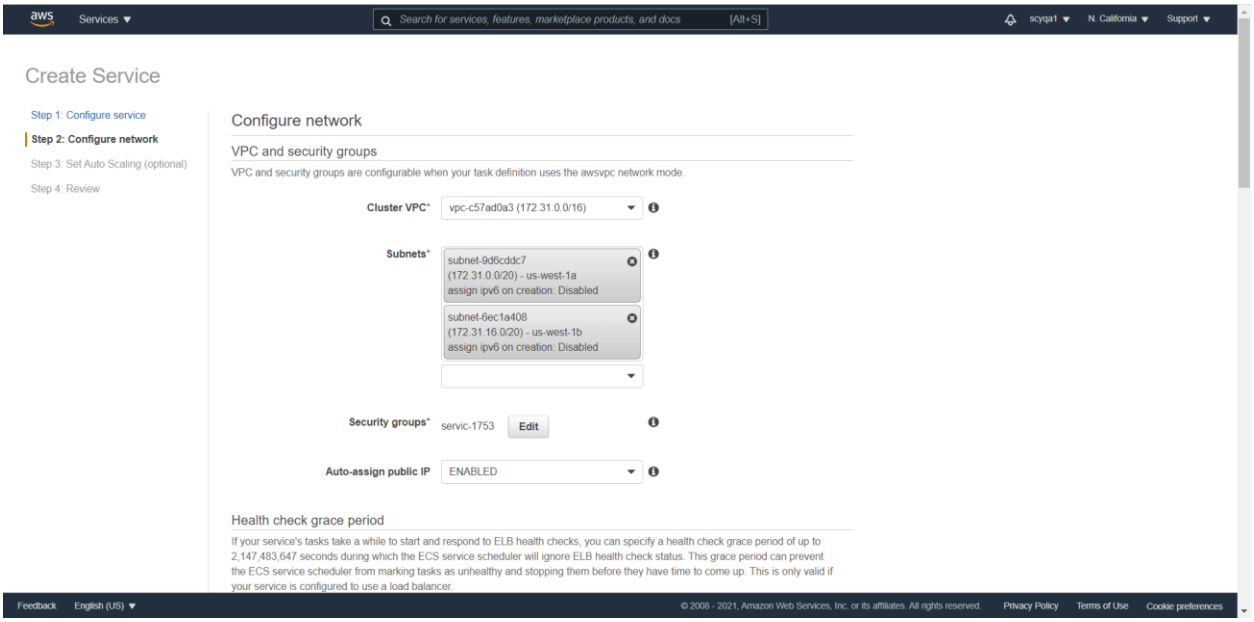
Create cluster

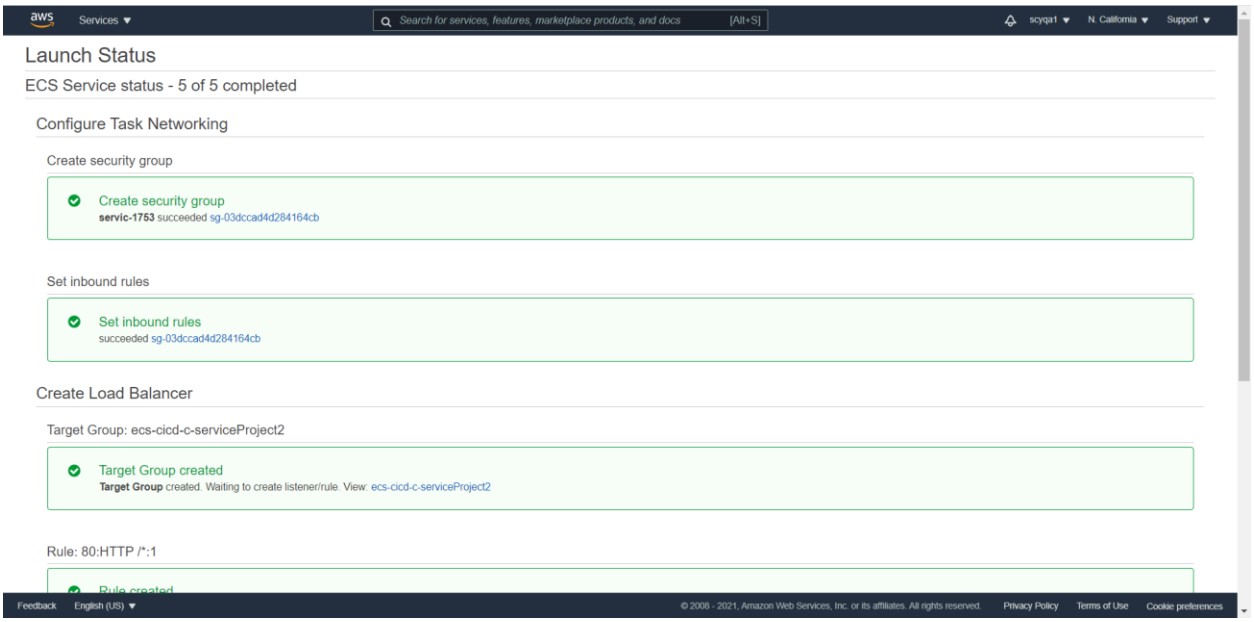


Create service

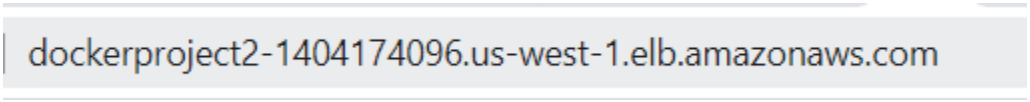





Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
N. California
Support
Create Service
Step 1: Configure service
Step 2: Configure network
Step 3: Set Auto Scaling (optional)
Step 4: Review
Configure network
VPC and security groups
VPC and security groups are configurable when your task definition uses the awsvpc network mode.
Cluster VPC*
vpc-c57ad0a3 (172.31.0.0/16)
Subnets*
subnet-9d6cddc7
(172.31.0.0/20) - us-west-1a
assign ipv6 on creation: Disabled
subnet-6ec1a408
(172.31.16.0/20) - us-west-1b
assign ipv6 on creation: Disabled
Security groups*
servic-1753
Edit
Auto-assign public IP
ENABLED
Health check grace period
If your service's tasks take a while to start and respond to ELB health checks, you can specify a health check grace period of up to 2,147,483,647 seconds during which the ECS service scheduler will ignore ELB health check status. This grace period can prevent the ECS service scheduler from marking tasks as unhealthy and stopping them before they have time to come up. This is only valid if your service is configured to use a load balancer.
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

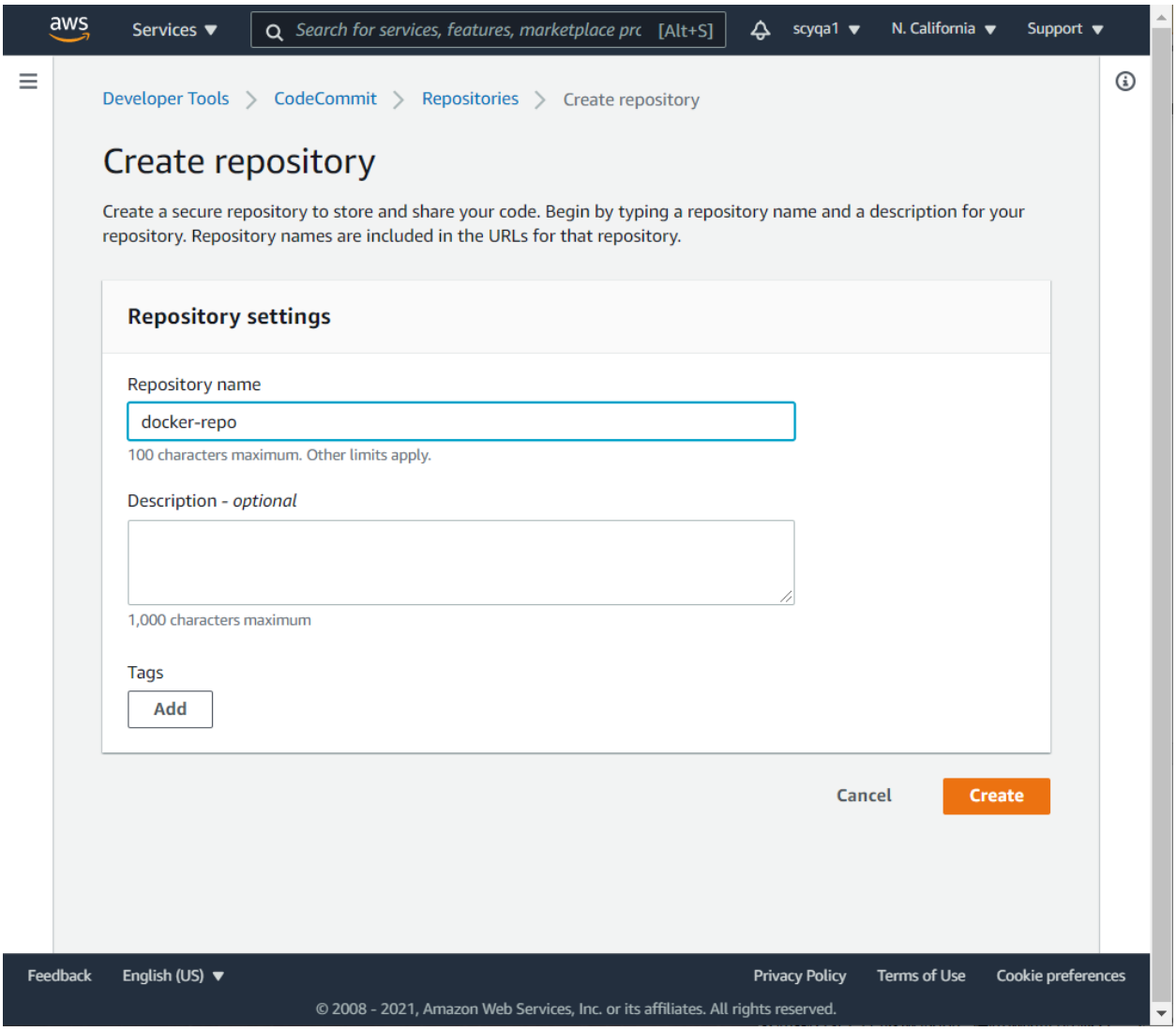

Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
N. California
Support
Launch Status
ECS Service status - 5 of 5 completed
Configure Task Networking
Create security group
Create security group
servic-1753 succeeded sg-03dccad4d284164cb
Set inbound rules
Set inbound rules
succeeded sg-03dccad4d284164cb
Create Load Balancer
Target Group: ecs-cicd-c-serviceProject2
Target Group created
Target Group created. Waiting to create listener/rule. View: ecs-cicd-c-serviceProject2
Rule: 80:HTTP /*:1
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

DNS Test

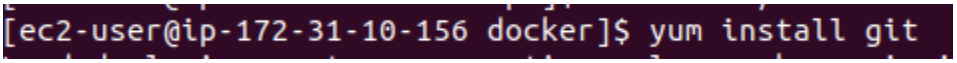

dockerproject2-1404174096.us-west-1.elb.amazonaws.com

5) [38:00] AWS Code commit: Create a Repo. Set ssh connectivity in your local machine. Push your code into the newly created repo

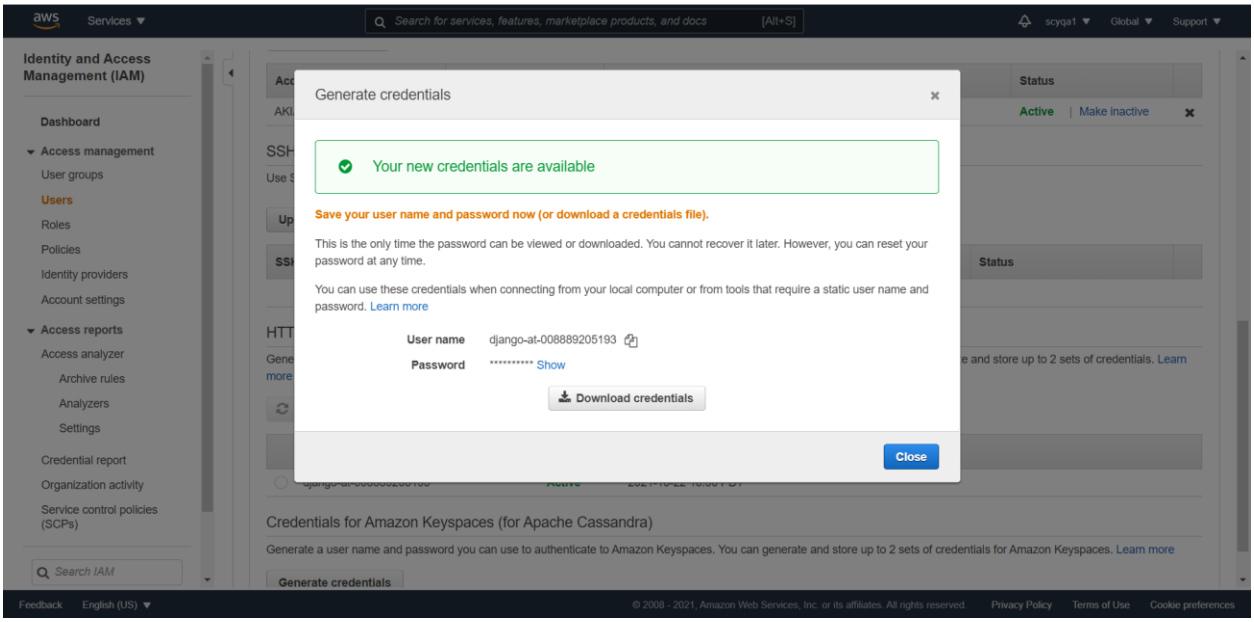
Create a repo



Install git



Generate HTTPS Git credentials for AWS CodeCommit



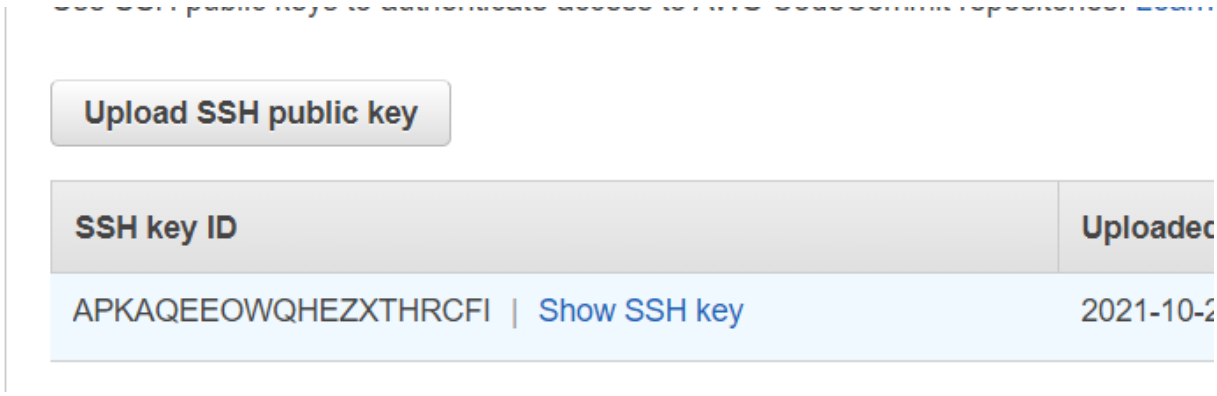
Clone

```
[root@ip-172-31-10-156 docker]# git clone https://git-codecommit.us-west-1.amazo
naws.com/v1/repos/docker-repo
Cloning into 'docker-repo'...
Username for 'https://git-codecommit.us-west-1.amazonaws.com': django-at-0088892
05193
Password for 'https://django-at-008889205193@git-codecommit.us-west-1.amazonaws.
com':
warning: You appear to have cloned an empty repository.
```

Generate SSH key

```
[root@ip-172-31-10-156 docker]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:wCVWeA8py79YJ2CZKLhn8GhZkVA0joEsnWjY+ND03as root@ip-172-31-10-156.us-west
-1.compute.internal
The key's randomart image is:
+---[RSA 2048]----+
|+X==. ooo.       |
|BoO+.+o++        |
|o= .oo+*.o       |
|o o.. B. ..      |
| =o. . oS        |
|oo+    .+ .      |
|.o    Eo +       |
|                 |
```

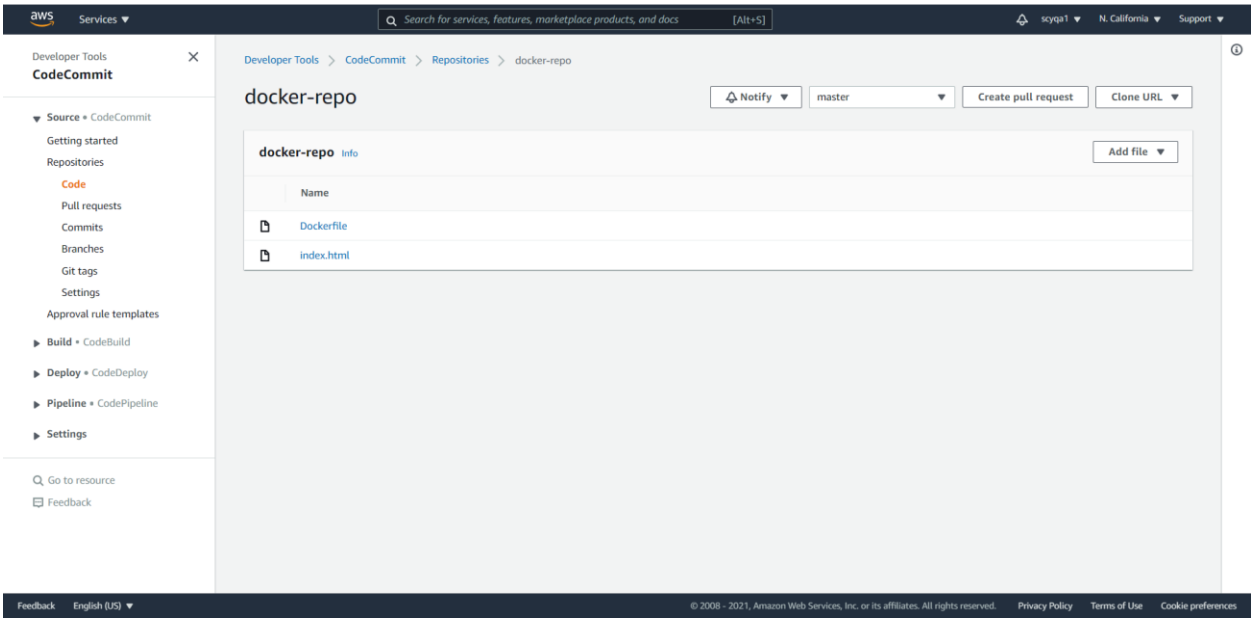
Then upload ssh key



Configure

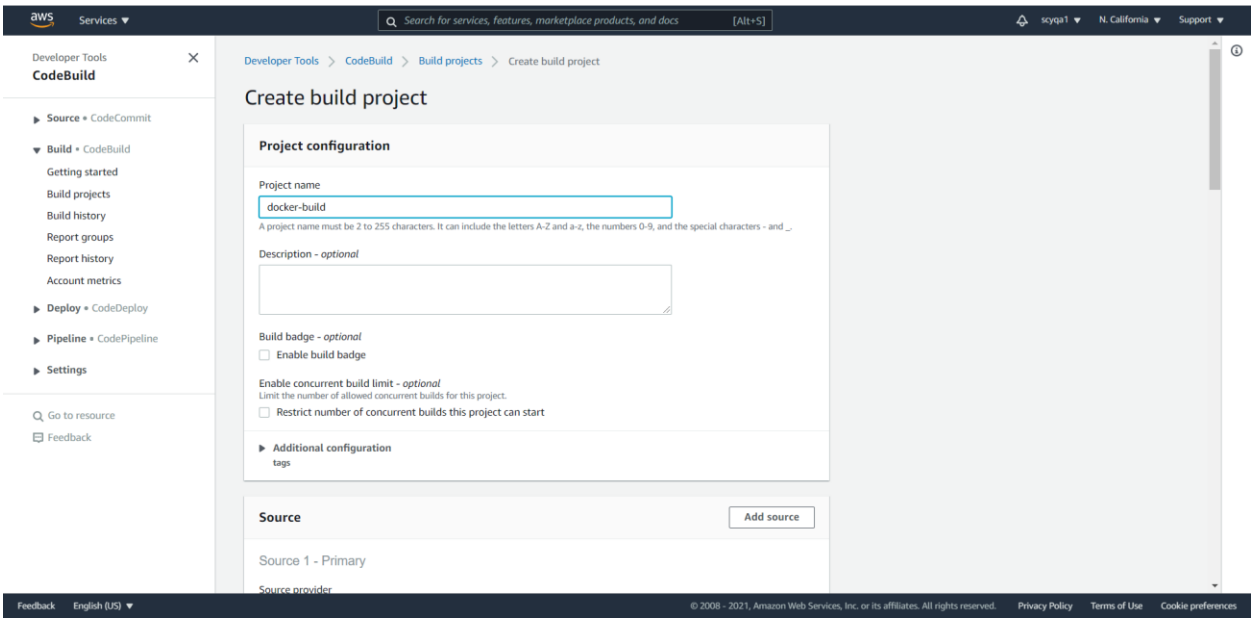
Git operations to push

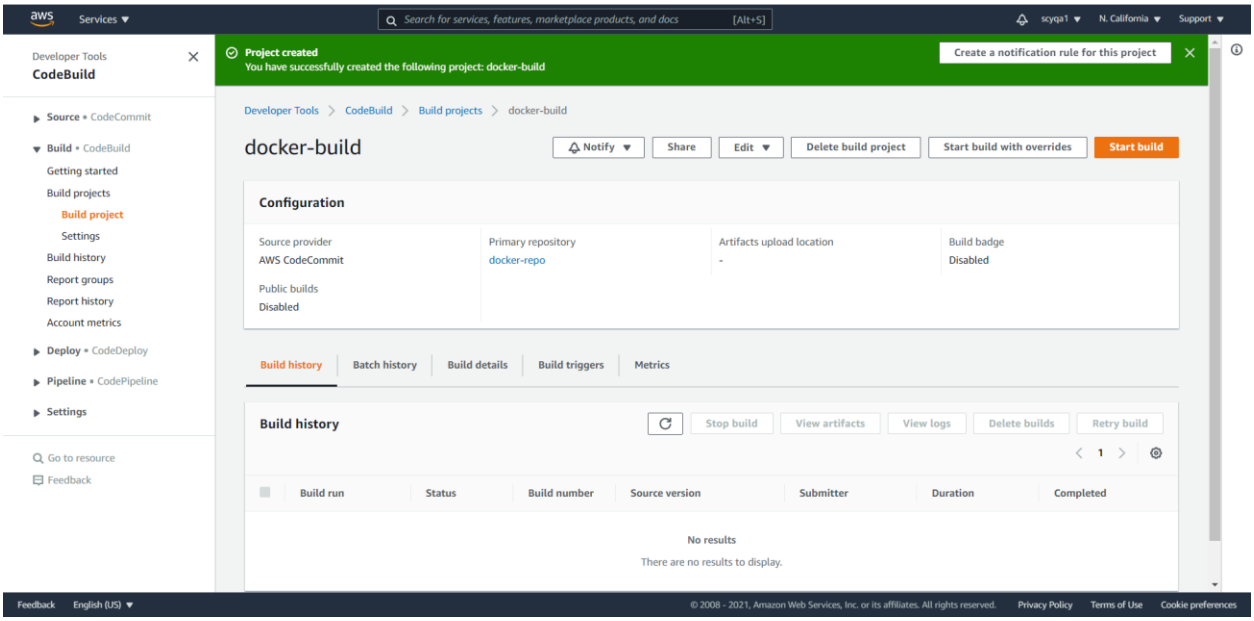
```
[root@ip-172-31-10-156 docker-repo]# git add .
[root@ip-172-31-10-156 docker-repo]# git commit -am"Primary Files"
[master (root-commit) 9b4ce6e] Primary Files
 Committer: root <root@ip-172-31-10-156.us-west-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 2 files changed, 9 insertions(+)
 create mode 100644 Dockerfile
 create mode 100644 index.html
[root@ip-172-31-10-156 docker-repo]# git push
```


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
N. California
Support
Developer Tools
CodeCommit
Source • CodeCommit
Getting started
Repositories
Code
Pull requests
Commits
Branches
Git tags
Settings
Approval rule templates
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Settings
Go to resource
Feedback
Developer Tools > CodeCommit > Repositories > docker-repo
docker-repo
Notify
master
Create pull request
Clone URL
docker-repo Info
Add file
Name
Dockerfile
index.html
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

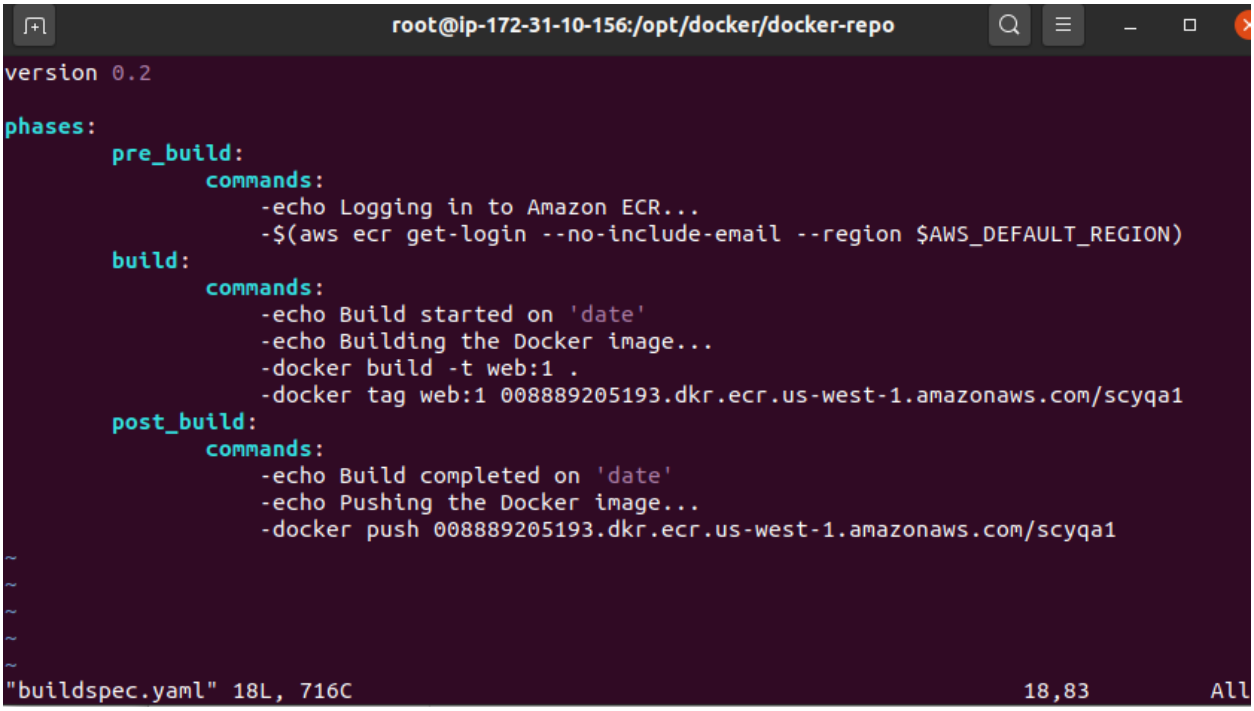
6) [54:00] AWS Code Build: Create a Code build Project. Select an Artifact for output. Build your Docker image & push to AWS ECR

Create build project



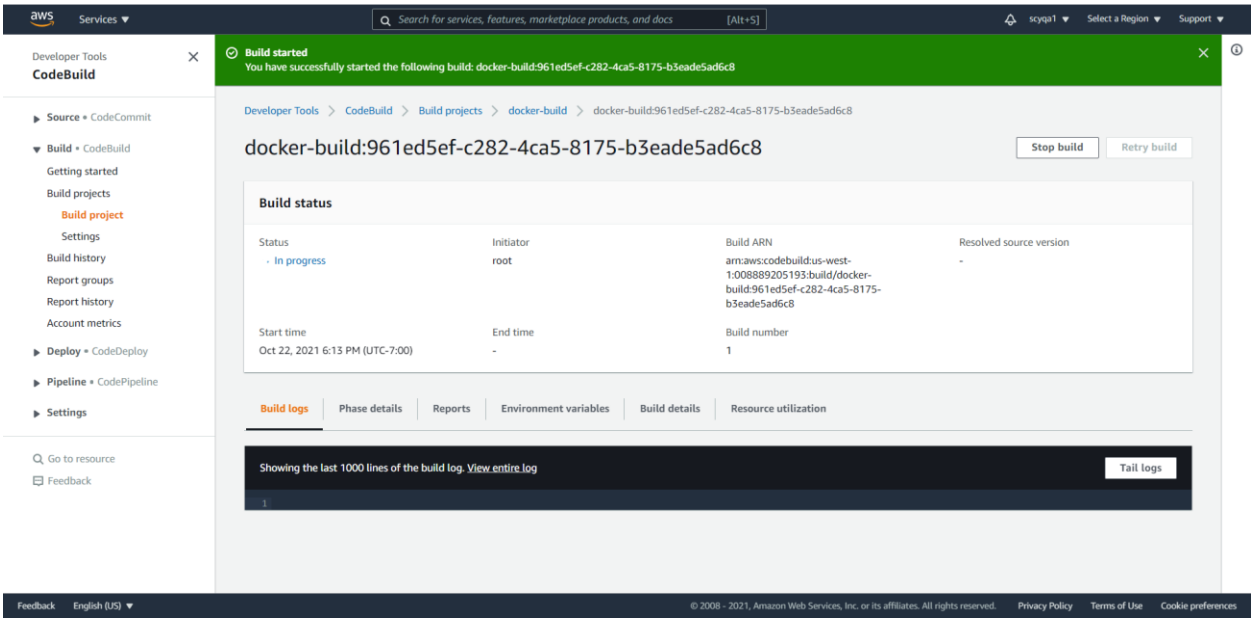


Configure

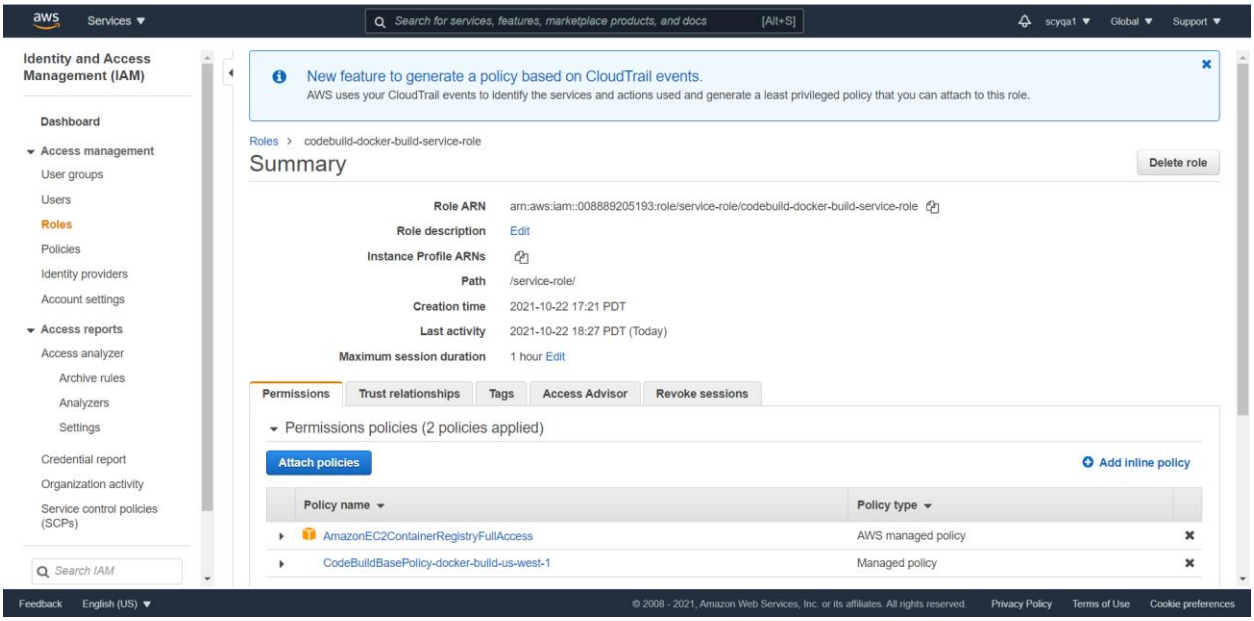



```
[root@ip-172-31-10-156 docker-repo]# git add .
[root@ip-172-31-10-156 docker-repo]# git commit -am"Add buildspec.yaml"
[master 8babf19] Add buildspec.yaml
 Committer: root <root@ip-172-31-10-156.us-west-1.compute.internal>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 18 insertions(+)
 create mode 100644 buildspec.yaml
[root@ip-172-31-10-156 docker-repo]# git push
```

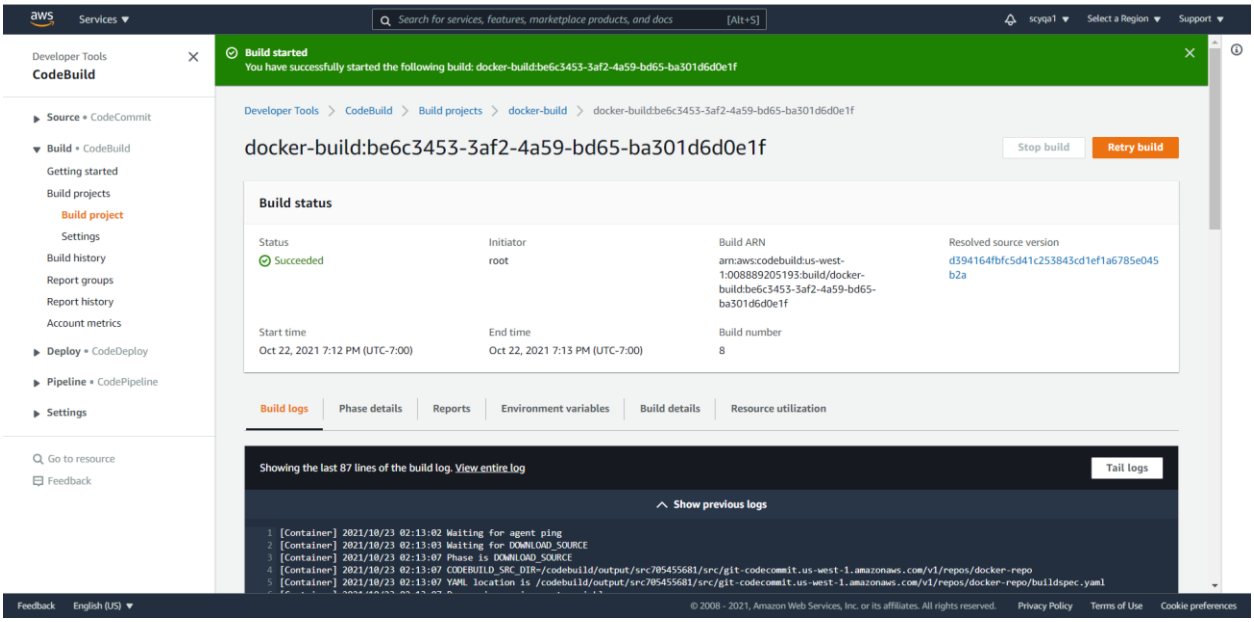
Start build

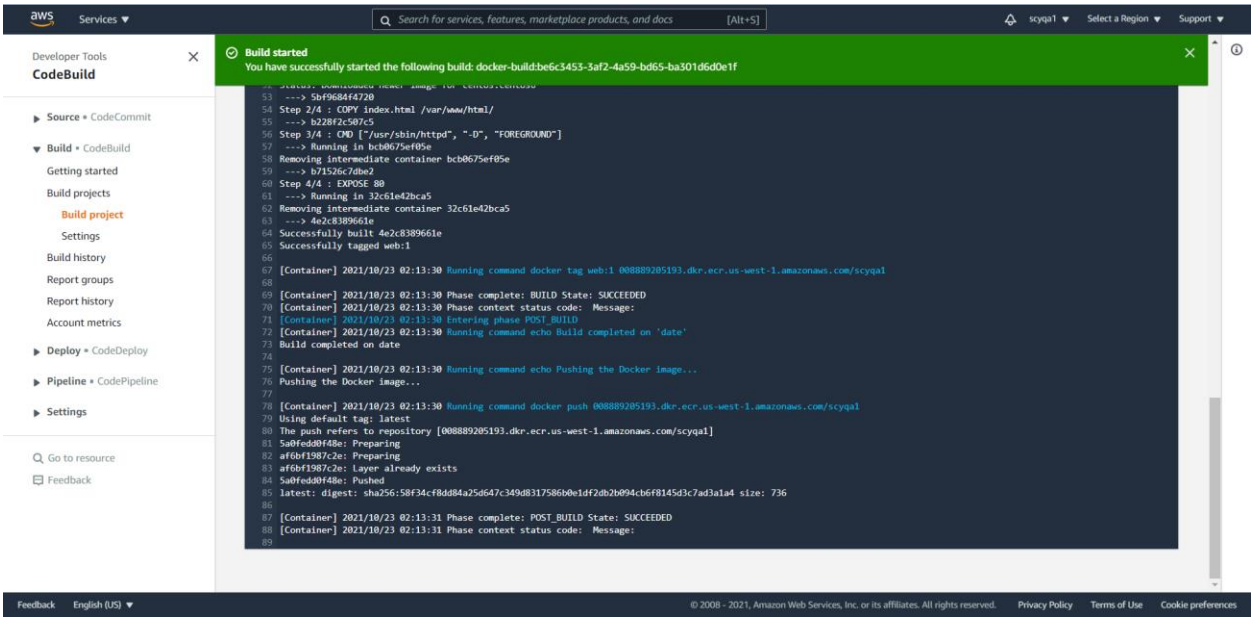


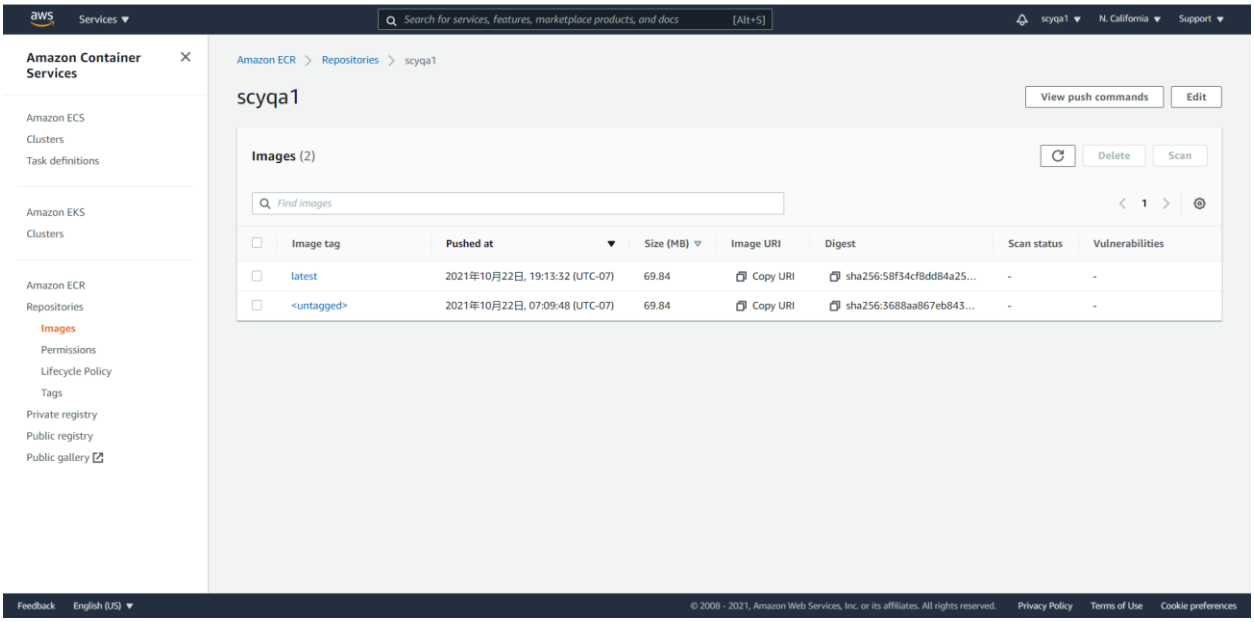
Add policy to roles



Success

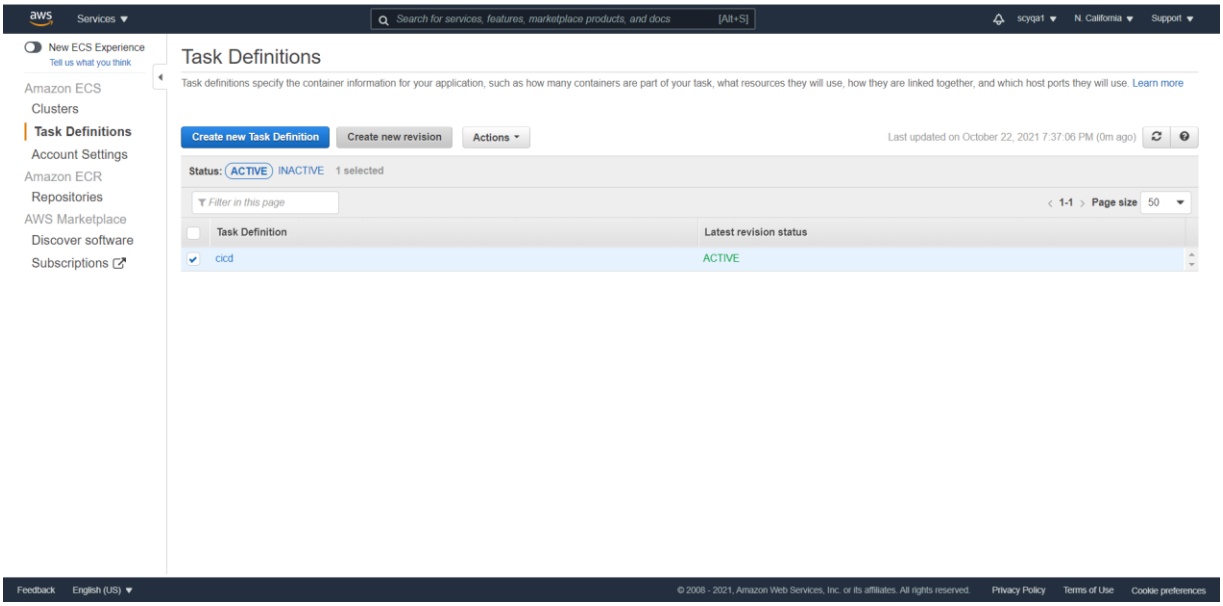

Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
Select a Region
Support
Developer Tools
CodeBuild
Source • CodeCommit
Build • CodeBuild
Getting started
Build projects
Build project
Settings
Build history
Report groups
Report history
Account metrics
Deploy • CodeDeploy
Pipeline • CodePipeline
Settings
Go to resource
Feedback
Build started
You have successfully started the following build: docker-build:be6c3453-3af2-4a59-bd65-ba301d6d0e1f
Developer Tools > CodeBuild > Build projects > docker-build > docker-build:be6c3453-3af2-4a59-bd65-ba301d6d0e1f
docker-build:be6c3453-3af2-4a59-bd65-ba301d6d0e1f
Stop build
Retry build
Build status
Status
Succeeded
Initiator
root
Build ARN
arn:aws:codebuild:us-west-1:008889205193:build/docker-build:be6c3453-3af2-4a59-bd65-ba301d6d0e1f
Resolved source version
d394164fbfc5d41c253843cd1ef1a6785e045b2a
Start time
Oct 22, 2021 7:12 PM (UTC-7:00)
End time
Oct 22, 2021 7:13 PM (UTC-7:00)
Build number
8
Build logs
Phase details
Reports
Environment variables
Build details
Resource utilization
Showing the last 87 lines of the build log. View entire log
Tail logs
Show previous logs
1 [Container] 2021/10/23 02:13:02 Waiting for agent ping
2 [Container] 2021/10/23 02:13:03 Waiting for DOWNLOAD_SOURCE
3 [Container] 2021/10/23 02:13:07 Phase is DOWNLOAD_SOURCE
4 [Container] 2021/10/23 02:13:07 CODEBUILD_SRC_DIR=/codebuild/output/src705455681/src/git-codecommit.us-west-1.amazonaws.com/v1/repos/docker-repo
5 [Container] 2021/10/23 02:13:07 YAML location is /codebuild/output/src705455681/src/git-codecommit.us-west-1.amazonaws.com/v1/repos/docker-repo/buildspec.yaml
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences


Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
Select a Region
Support
Developer Tools
CodeBuild
Source • CodeCommit
Build • CodeBuild
Getting started
Build projects
Build project
Settings
Build history
Report groups
Report history
Account metrics
Deploy • CodeDeploy
Pipeline • CodePipeline
Settings
Go to resource
Feedback
Build started
You have successfully started the following build: docker-build:be6c3453-3af2-4a59-bd65-ba301d6d0e1f
---> 5bf9684f4720
Step 2/4 : COPY index.html /var/www/html/
---> b228f2c507c5
Step 3/4 : CMD ["/usr/sbin/httpd", "-D", "FOREGROUND"]
---> Running in bcb0675ef05e
Removing intermediate container bcb0675ef05e
---> b71526c7dbe2
Step 4/4 : EXPOSE 80
---> Running in 32c61e42bca5
Removing intermediate container 32c61e42bca5
---> 4e2c8389661e
Successfully built 4e2c8389661e
Successfully tagged web:1
[Container] 2021/10/23 02:13:30 Running command docker tag web:1 008889205193.dkr.ecr.us-west-1.amazonaws.com/scyqa1
[Container] 2021/10/23 02:13:30 Phase complete: BUILD State: SUCCEEDED
[Container] 2021/10/23 02:13:30 Phase context status code: Message:
[Container] 2021/10/23 02:13:30 Entering phase POST_BUILD
[Container] 2021/10/23 02:13:30 Running command echo Build completed on 'date'
Build completed on date
[Container] 2021/10/23 02:13:30 Running command echo Pushing the Docker image...
Pushing the Docker image...
[Container] 2021/10/23 02:13:30 Running command docker push 008889205193.dkr.ecr.us-west-1.amazonaws.com/scyqa1
Using default tag: latest
The push refers to repository [008889205193.dkr.ecr.us-west-1.amazonaws.com/scyqa1]
5a0fedd0f48e: Preparing
af6bf1987c2e: Preparing
af6bf1987c2e: Layer already exists
5a0fedd0f48e: Pushed
latest: digest: sha256:58f34cf8dd84a25d647c349d8317586b0e1df2db2b094cb6f8145d3c7ad3a1a4 size: 736
[Container] 2021/10/23 02:13:31 Phase complete: POST_BUILD State: SUCCEEDED
[Container] 2021/10/23 02:13:31 Phase context status code: Message:
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

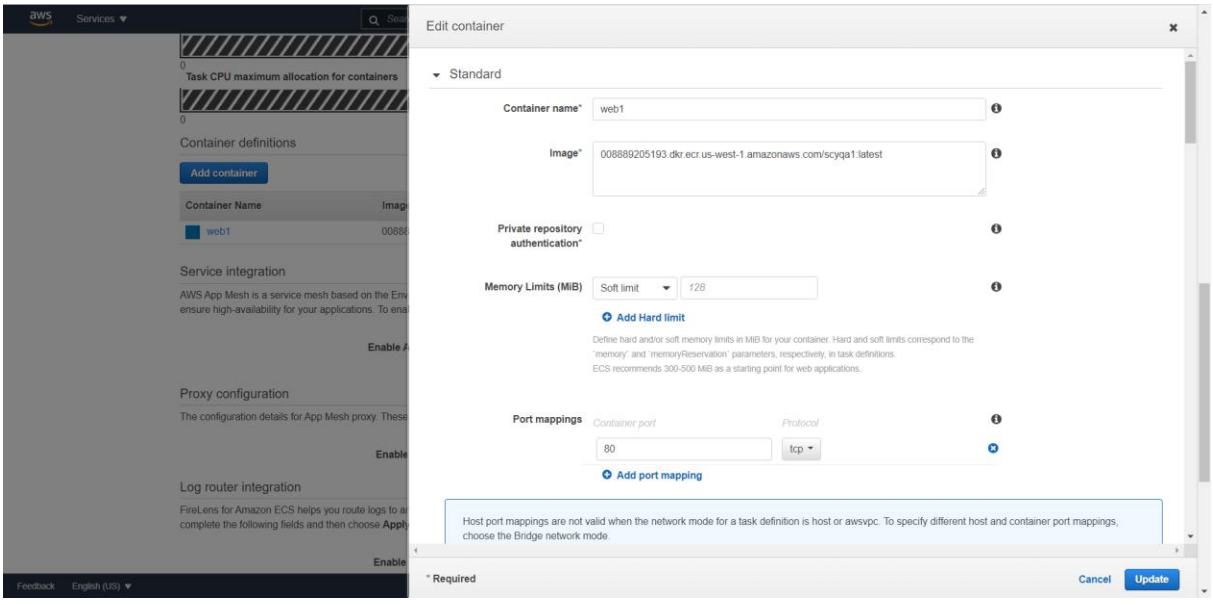

aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
N. California
Support
Amazon Container Services
Amazon ECS
Clusters
Task definitions
Amazon EKS
Clusters
Amazon ECR
Repositories
Images
Permissions
Lifecycle Policy
Tags
Private registry
Public registry
Public gallery
Amazon ECR > Repositories > scyqa1
scyqa1
View push commands
Edit
Images (2)
Delete
Scan
Find images
1
Image tag
Pushed at
Size (MB)
Image URI
Digest
Scan status
Vulnerabilities
latest
2021年10月22日, 19:13:32 (UTC-07)
69.84
Copy URI
sha256:58f34cf8dd84a25...
-
-
<untagged>
2021年10月22日, 07:09:48 (UTC-07)
69.84
Copy URI
sha256:3688aa867eb843...
-
-
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

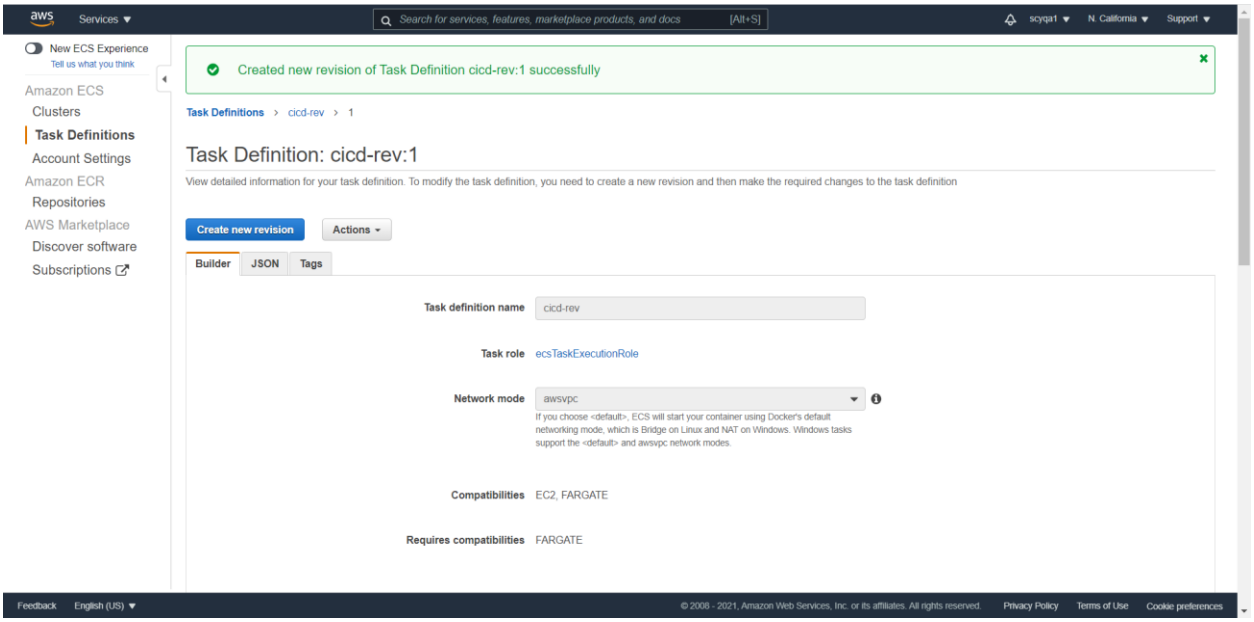
7) [1:06:55] AWS Code Pipeline (the manual one is optional and you can skip it but not harm to try, automated way starts at [1:10:05]): Create a pipeline Project. Select you Code commit repo as an input repo from step 5. Select Code build project from Step 6. Select Code Deploy for ECS. Select your cluster and Service name from step 3. Crete the Code pipeline

Create new revision of task definitions

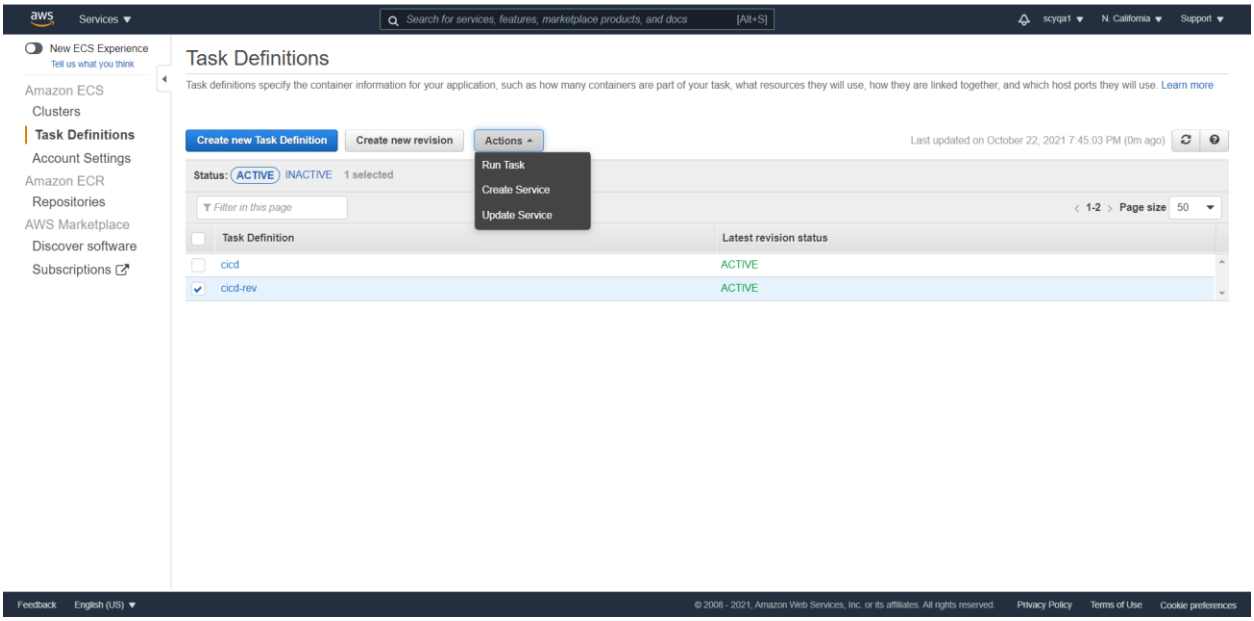


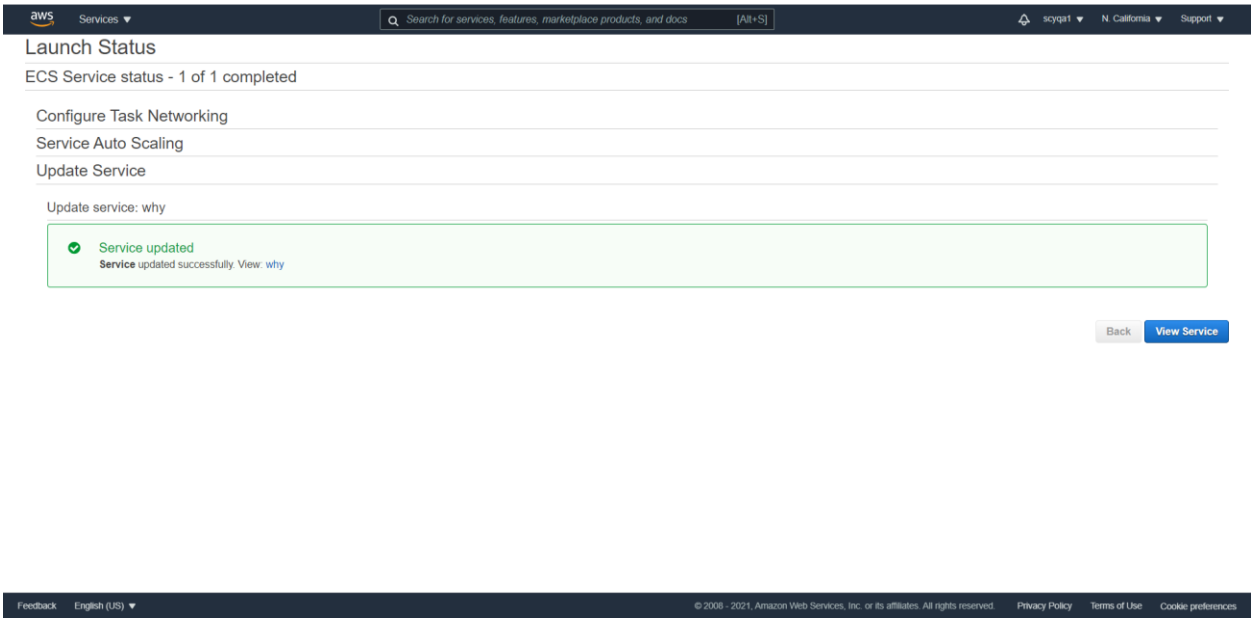
Update container





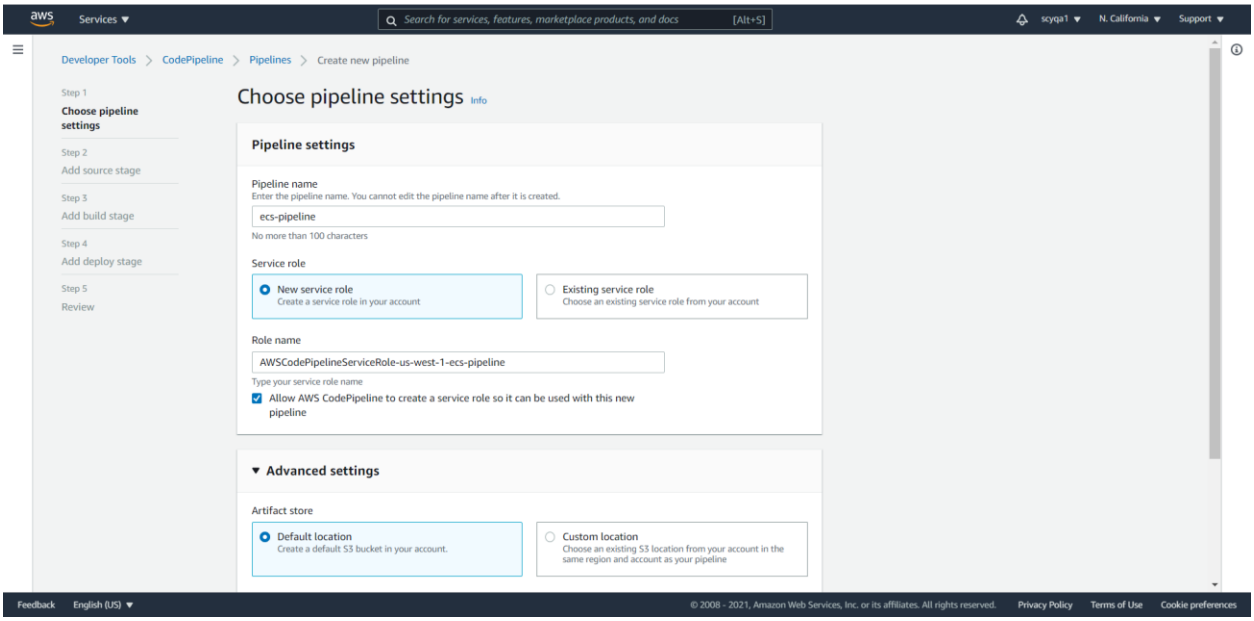
Update service



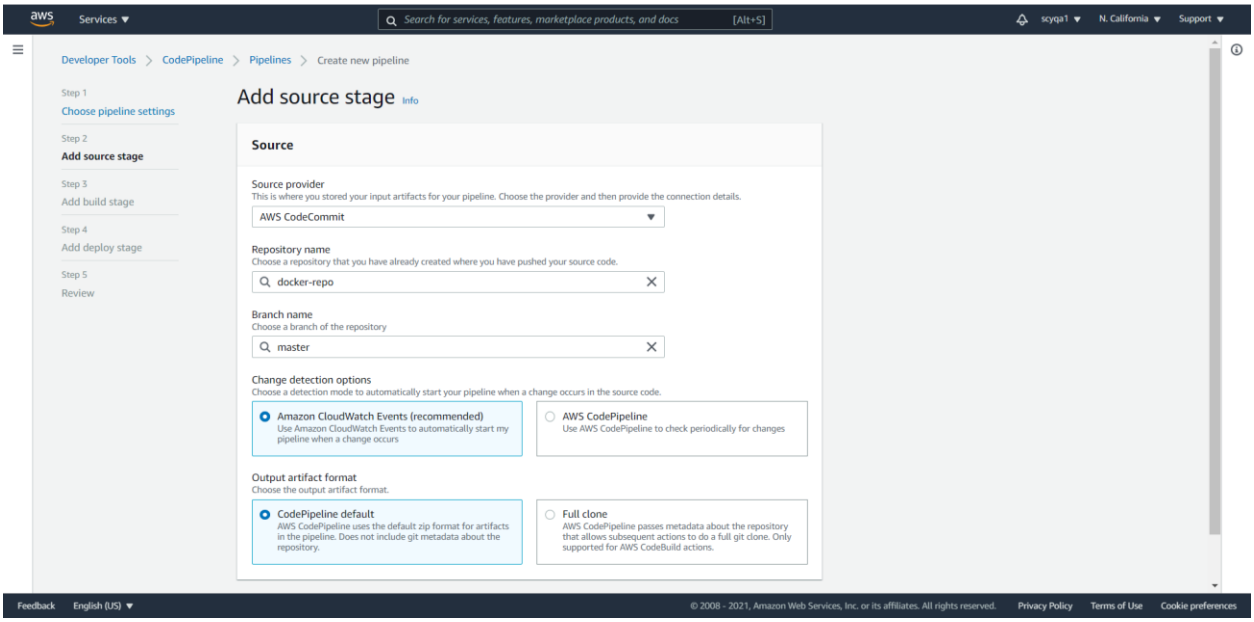


Create pipeline

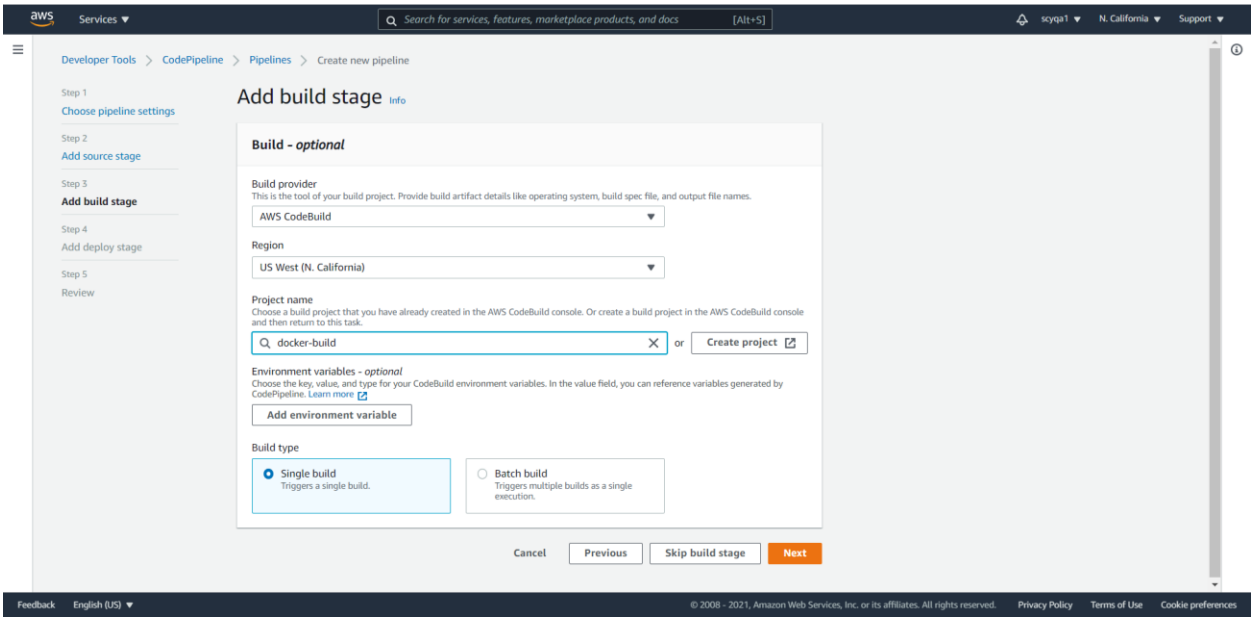
Choose pipeline settings



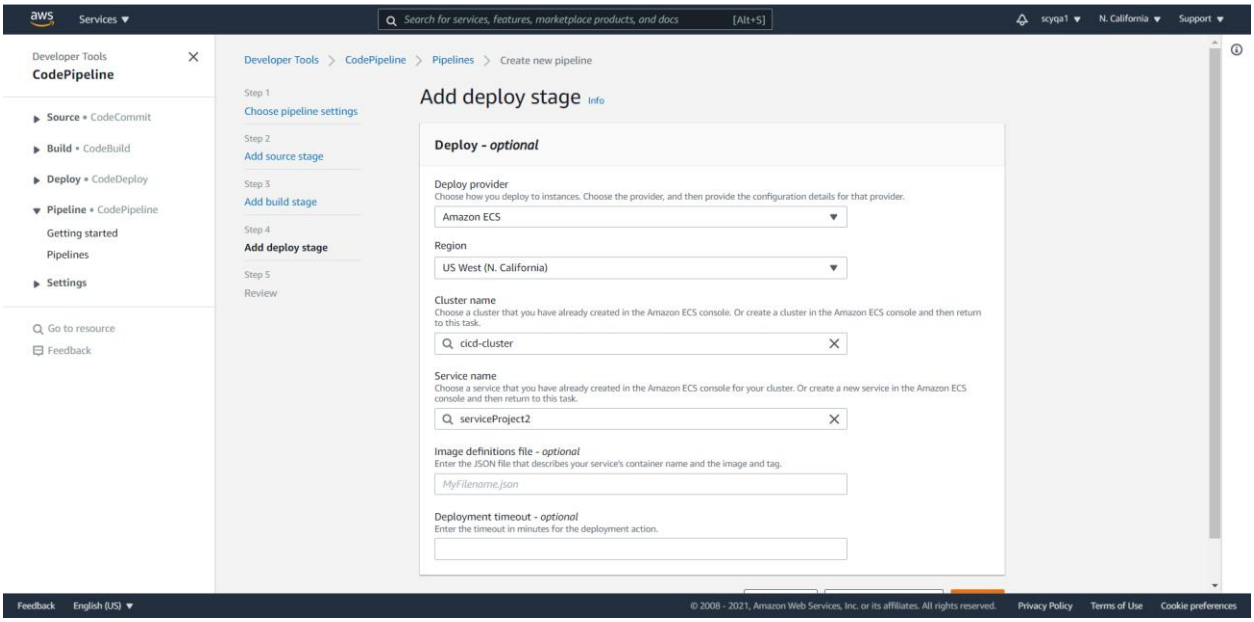
Add source stage



Add build stage

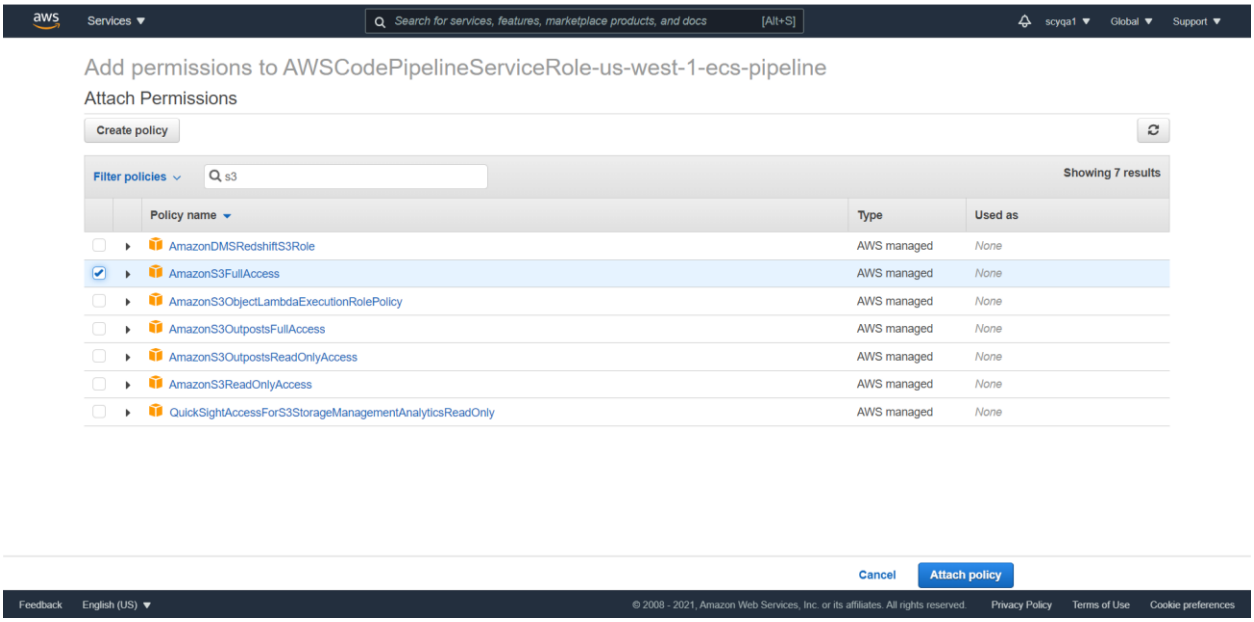


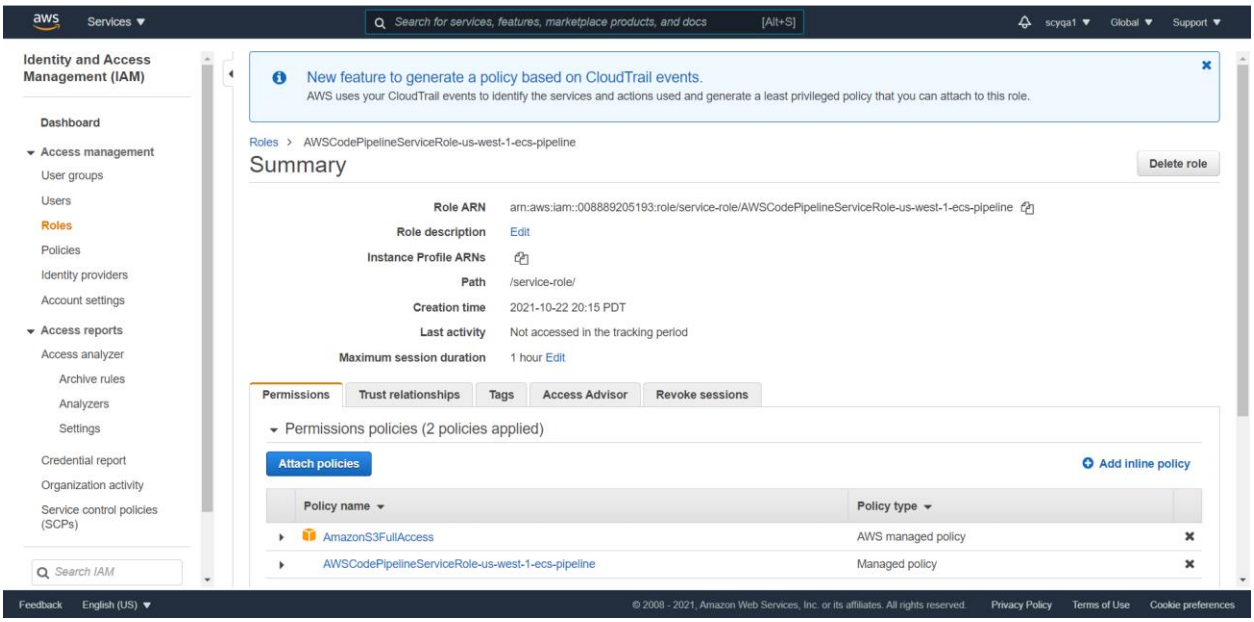
Add deploy stage



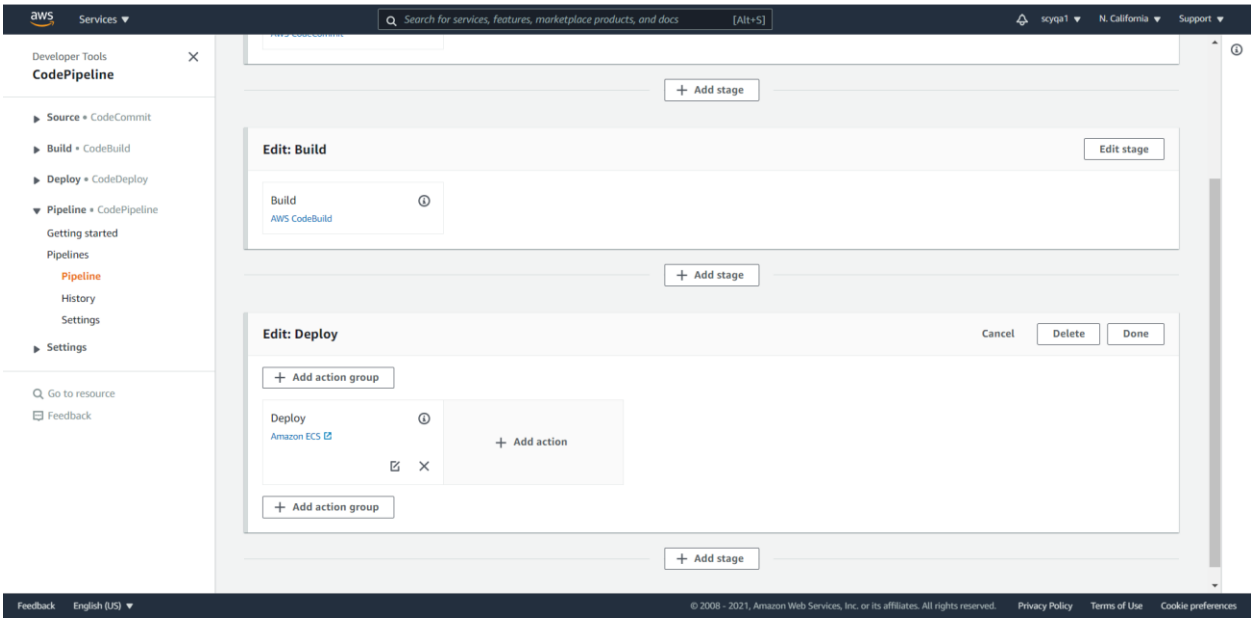
8) [1:19:45] Error & Resolution: In the above pipeline, first 2 steps will run successfully. Step 3 deploy will give you an error. For this, you need to Create one “imagedefinitions.json” file and push it to the code commit. The pipeline will run again and you will again get an error on step 3

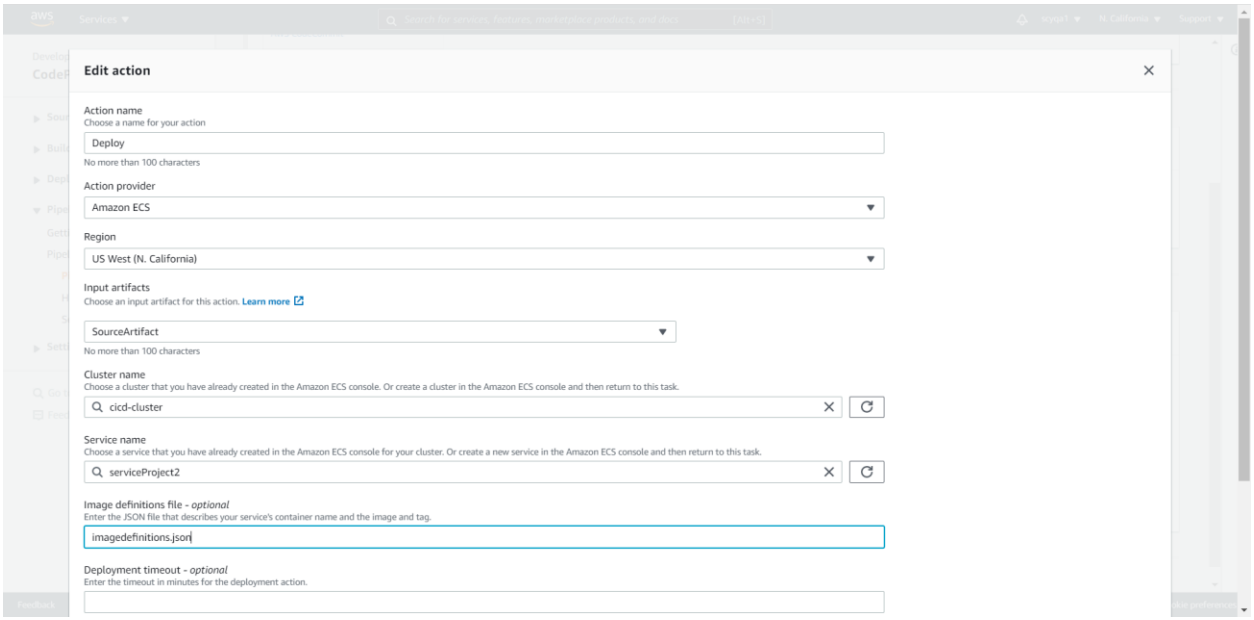
Roles attach policy



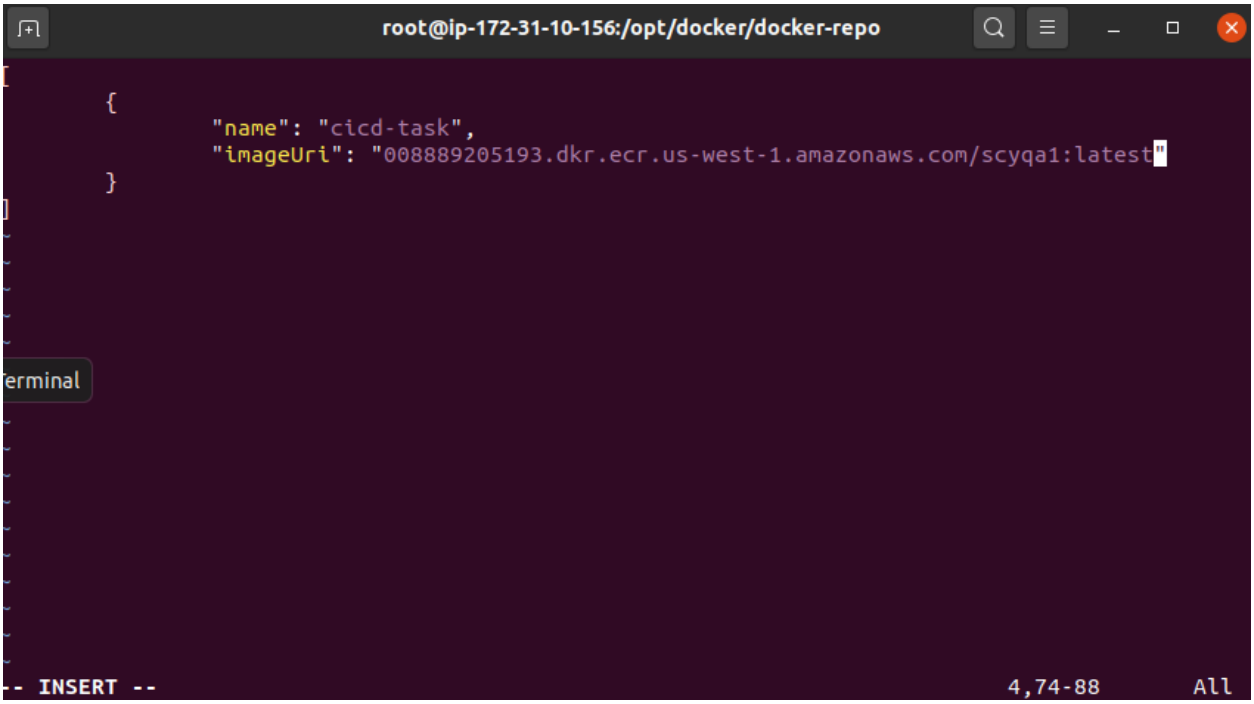


Edit: Deploy

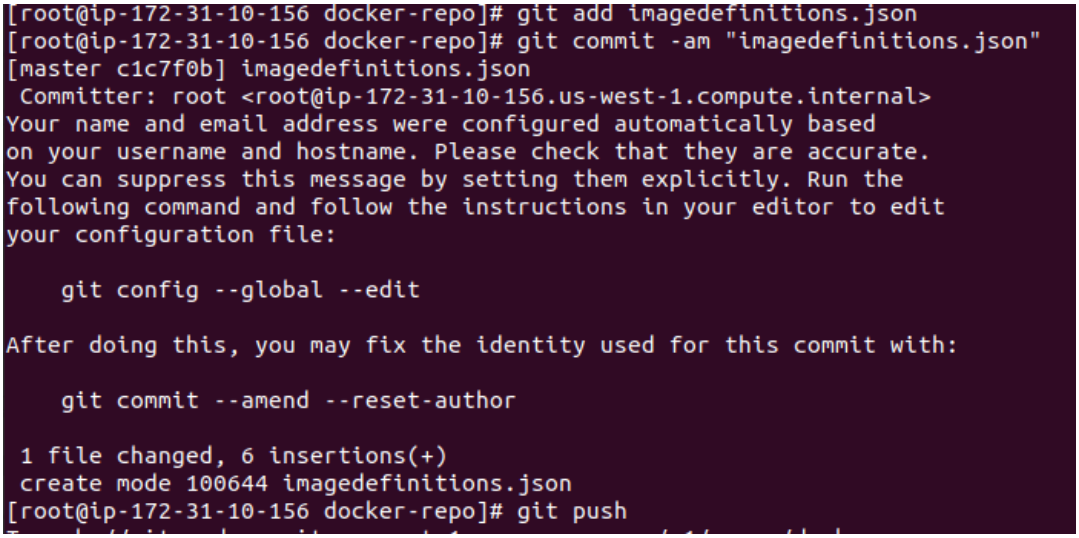

Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
N. California
Support
Developer Tools
CodePipeline
Source • CodeCommit
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
Add stage
Edit: Build
Edit stage
Build
AWS CodeBuild
Add stage
Edit: Deploy
Cancel
Delete
Done
Add action group
Deploy
Amazon ECS
Add action
Add action group
Add stage
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

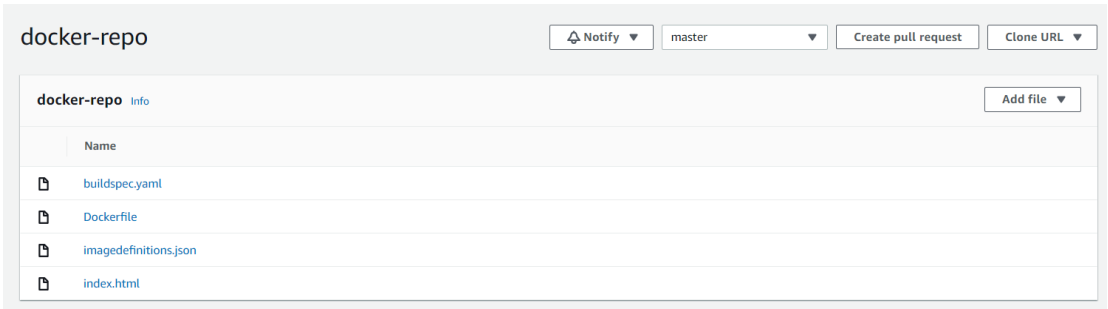

Edit action
Action name
Choose a name for your action
Deploy
No more than 100 characters
Action provider
Amazon ECS
Region
US West (N. California)
Input artifacts
Choose an input artifact for this action. Learn more
SourceArtifact
No more than 100 characters
Cluster name
Choose a cluster that you have already created in the Amazon ECS console. Or create a cluster in the Amazon ECS console and then return to this task.
cicd-cluster
Service name
Choose a service that you have already created in the Amazon ECS console for your cluster. Or create a new service in the Amazon ECS console and then return to this task.
serviceProject2
Image definitions file - optional
Enter the JSON file that describes your service's container name and the image and tag.
imagedefinitions.json
Deployment timeout - optional
Enter the timeout in minutes for the deployment action.

Imagedefinitions.json



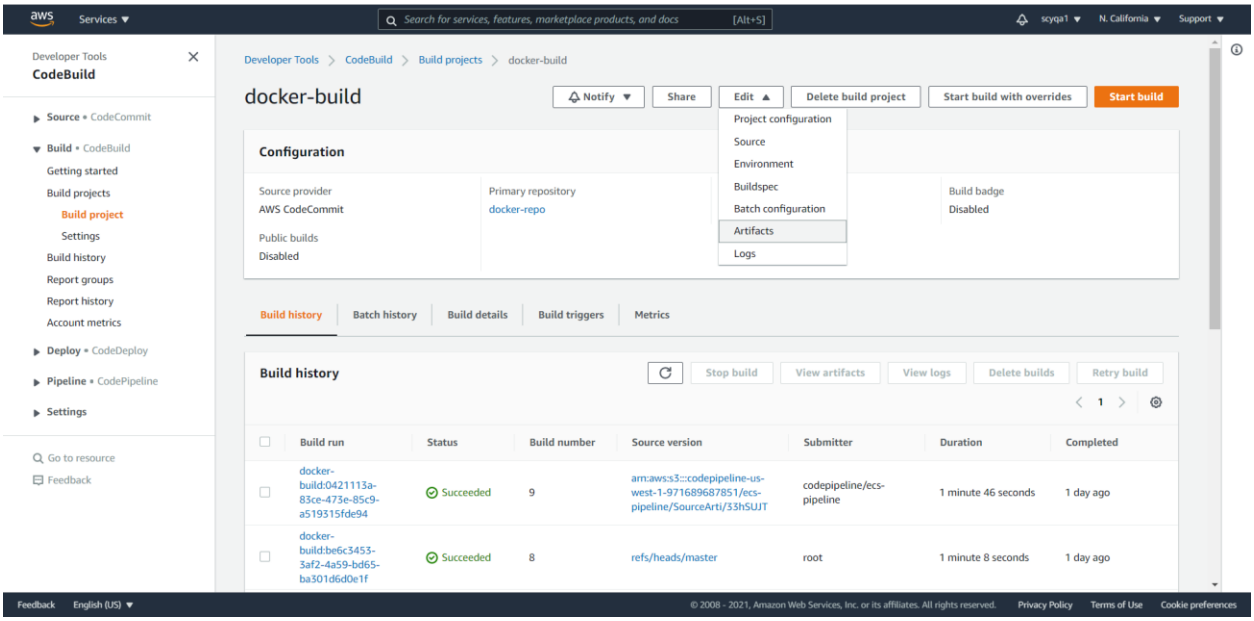
Push

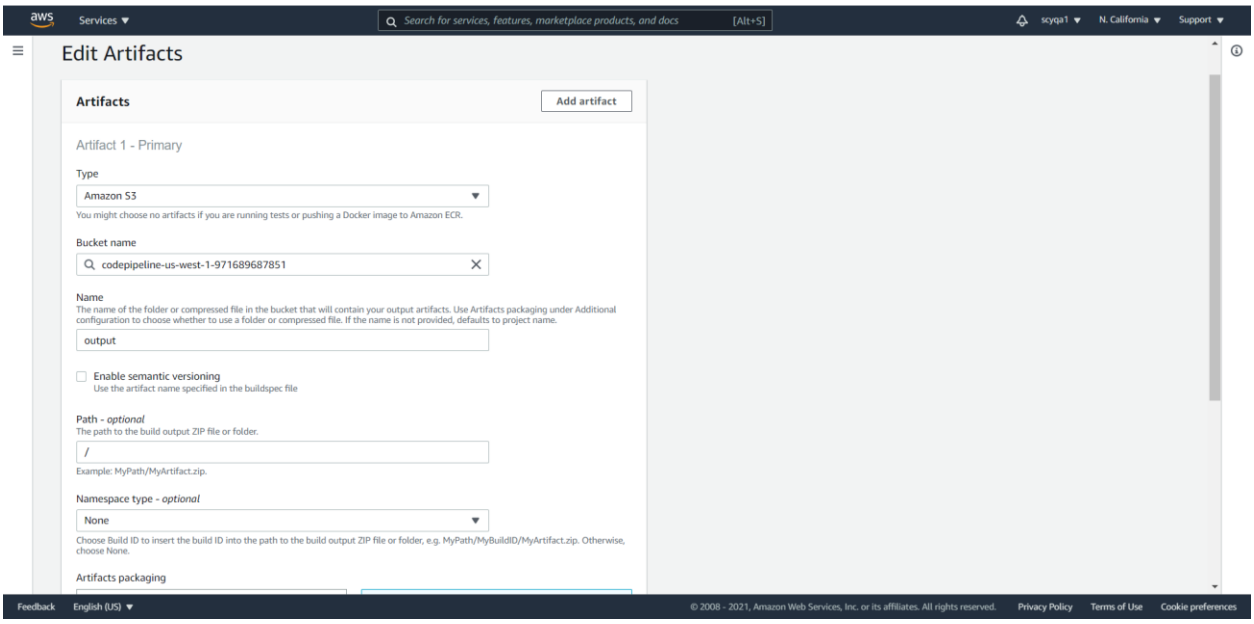




9) Modify Input Artifacts Settings for Stage: Change the input.

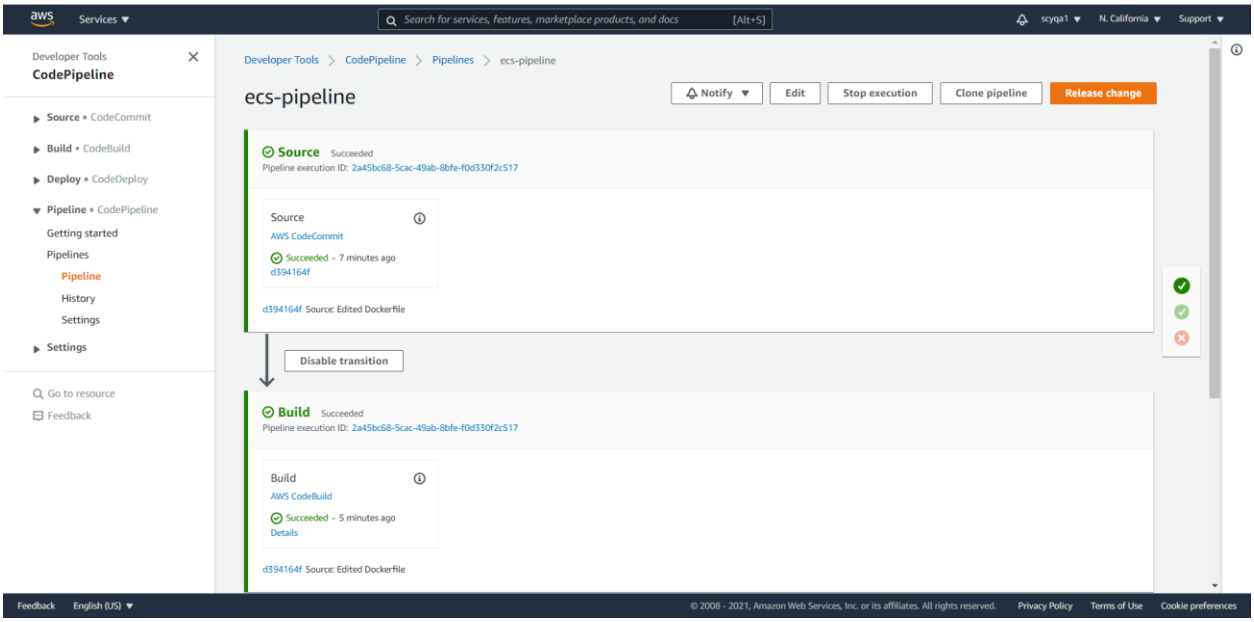
Artifacts

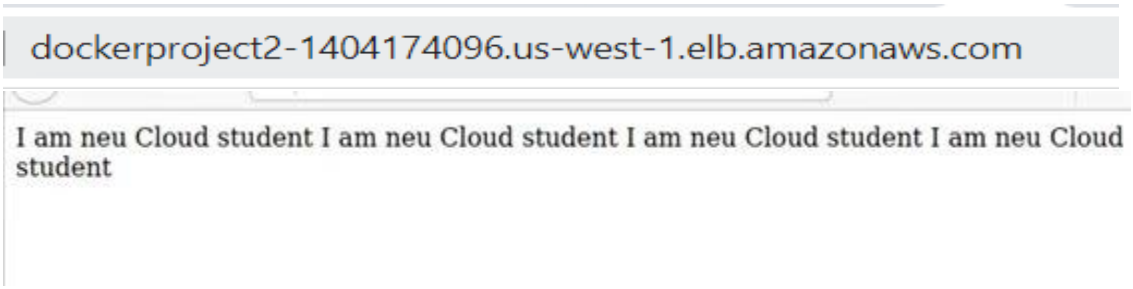




10) Final Deployment Test and Validation: Push the new version of code in a code commit. It will automatically deploy the new task with the new version. At last, you can run the DNS ALB on your browser

Success

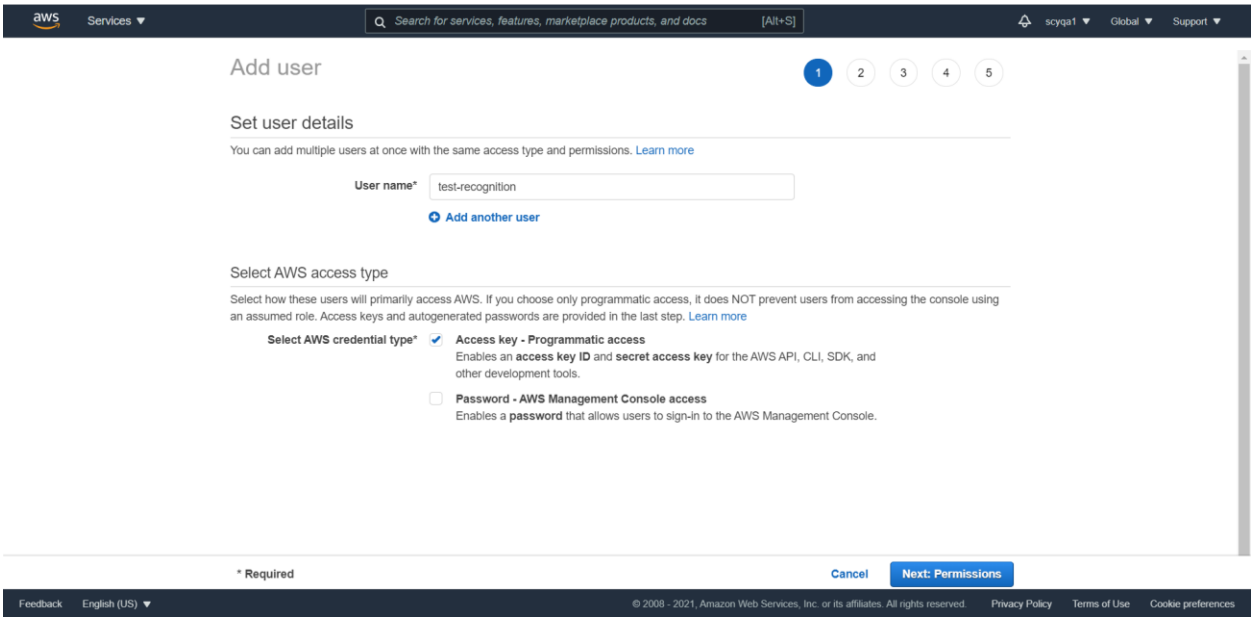


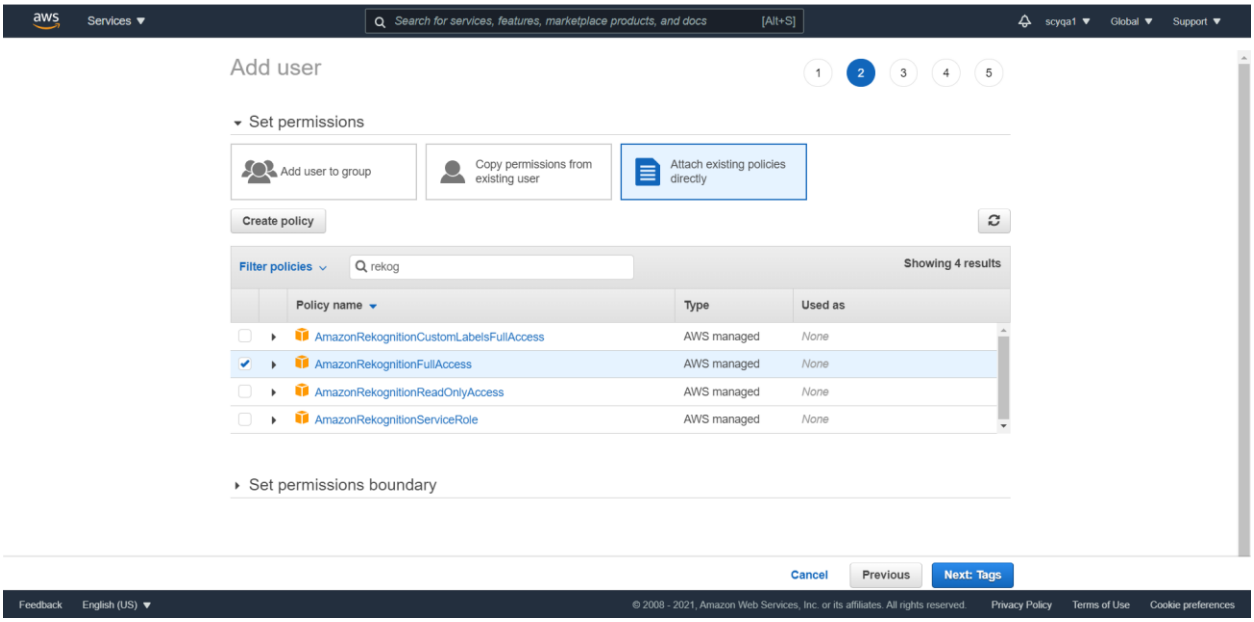


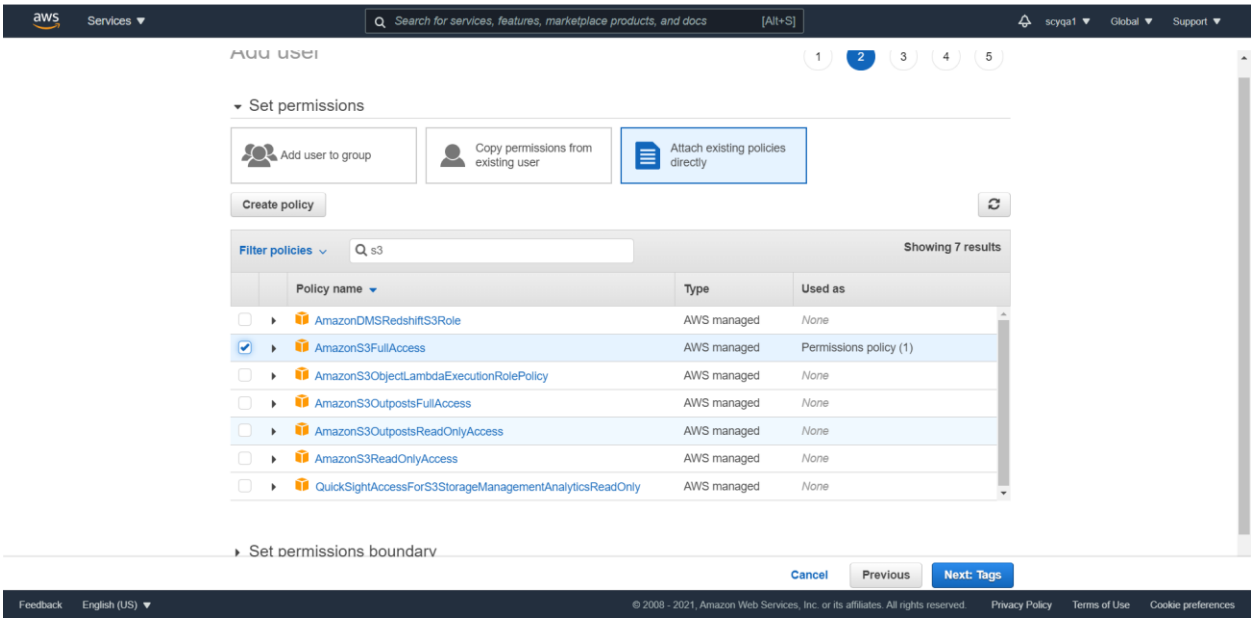
PART II: Design Character Recognition System using AWS Lambda Function [100 points]

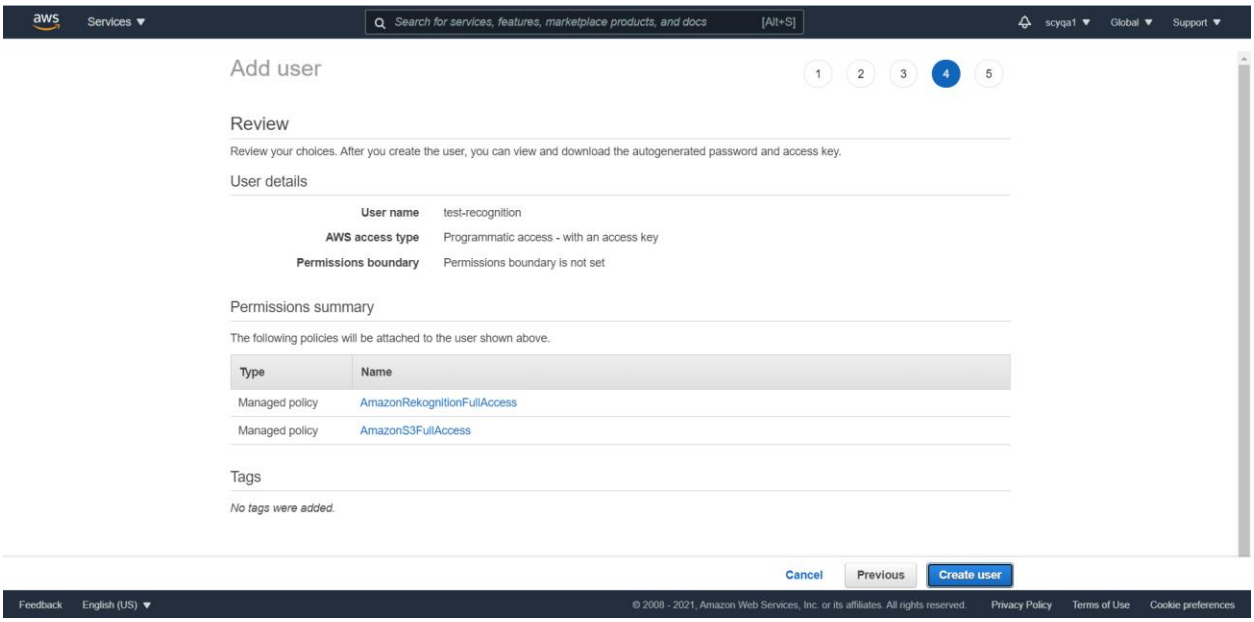
1) Docker Read the data from S3 (It is OK if you don't/can't use docker but use S3 to store your character image file)

Assigned role with S3 read/write permission and Amazon Rekognition API

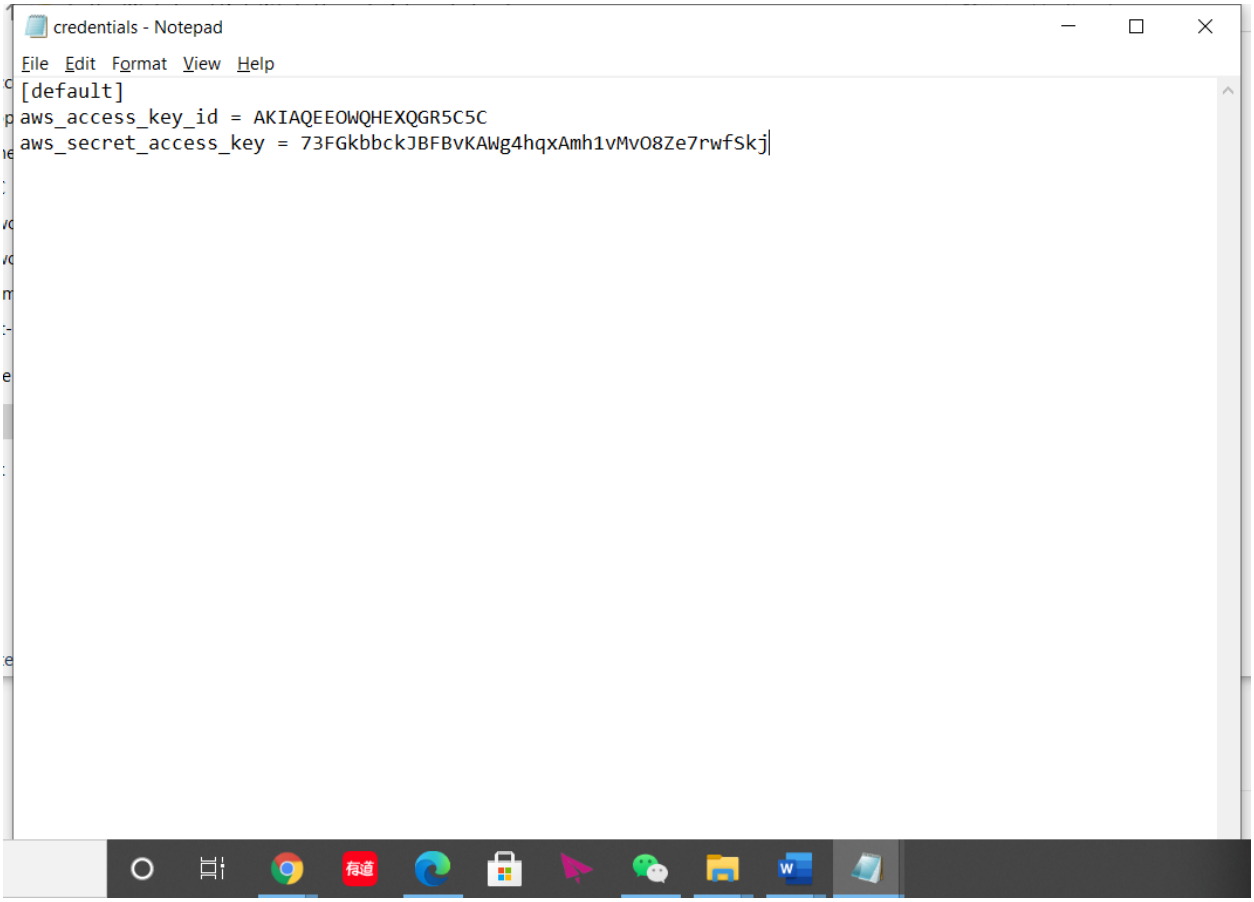


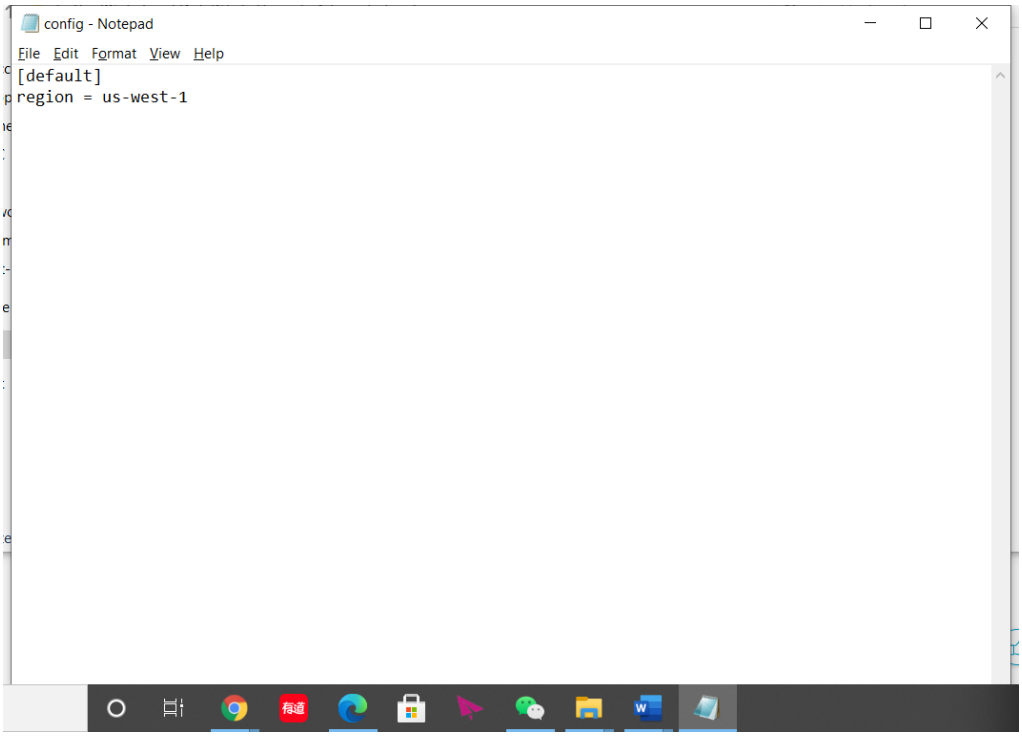


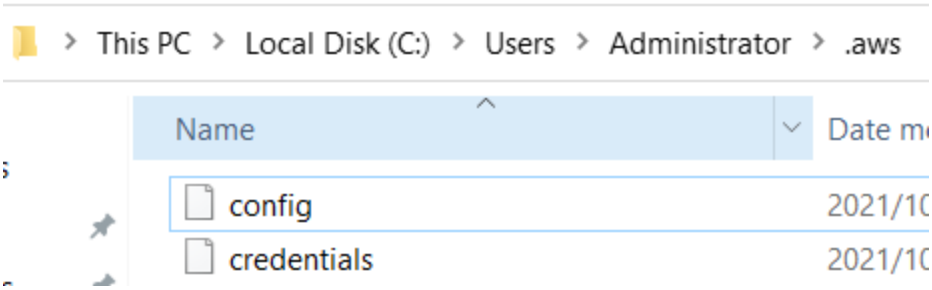

Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
Global
Support
1 2 3 4 5
Set permissions
Add user to group
Copy permissions from existing user
Attach existing policies directly
Create policy
Filter policies
s3
Showing 7 results
Policy name
Type
Used as
AmazonDMSRedshiftS3Role AWS managed None
AmazonS3FullAccess AWS managed Permissions policy (1)
AmazonS3ObjectLambdaExecutionRolePolicy AWS managed None
AmazonS3OutpostsFullAccess AWS managed None
AmazonS3OutpostsReadOnlyAccess AWS managed None
AmazonS3ReadOnlyAccess AWS managed None
QuickSightAccessForS3StorageManagementAnalyticsReadOnly AWS managed None
Set permissions boundary
Cancel
Previous
Next: Tags
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences


Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
Global
Support
Add user
1 2 3 4 5
Review
Review your choices. After you create the user, you can view and download the autogenerated password and access key.
User details
User name test-recognition
AWS access type Programmatic access - with an access key
Permissions boundary Permissions boundary is not set
Permissions summary
The following policies will be attached to the user shown above.
Type
Name
Managed policy AmazonRekognitionFullAccess
Managed policy AmazonS3FullAccess
Tags
No tags were added.
Cancel
Previous
Create user
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences

Configure

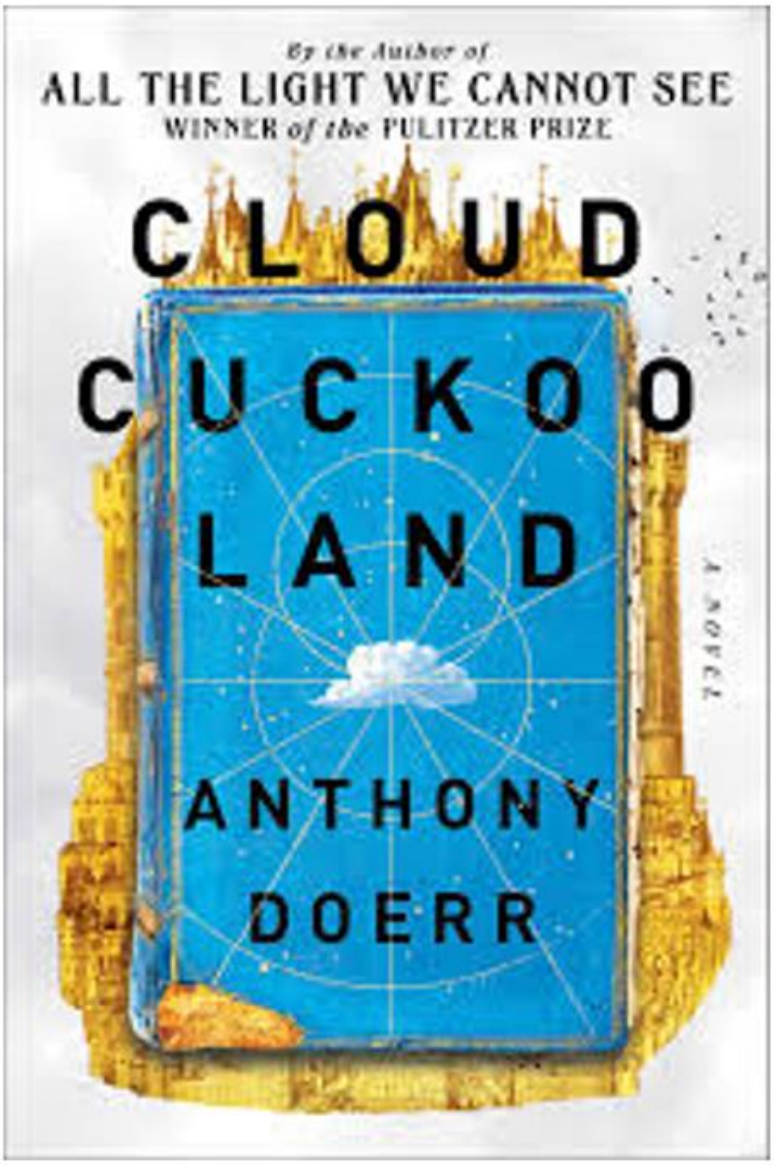

credentials - Notepad
File Edit Format View Help
[default]
aws_access_key_id = AKIAQEEOWQHEXQGR5C5C
aws_secret_access_key = 73FGkbbckJBFBvKAWg4hqxAmh1vMvO8Ze7rwfSkj


config - Notepad
File Edit Format View Help
[default]
region = us-west-1



Upload the image that contains text to your S3 bucket.

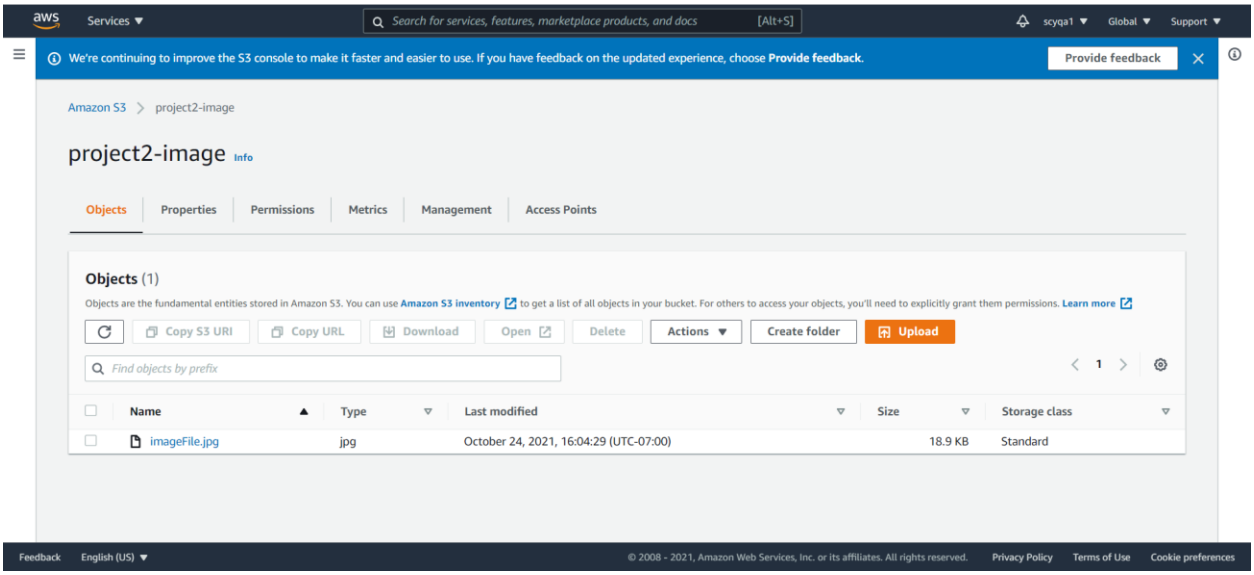
Used Image:



Upload:

```
[root@ip-172-31-10-156 s3]# aws s3 cp ./ s3://project2-image/ --recursive
upload: ./imageFile.jpg to s3://project2-image/imageFile.jpg
```

Result:

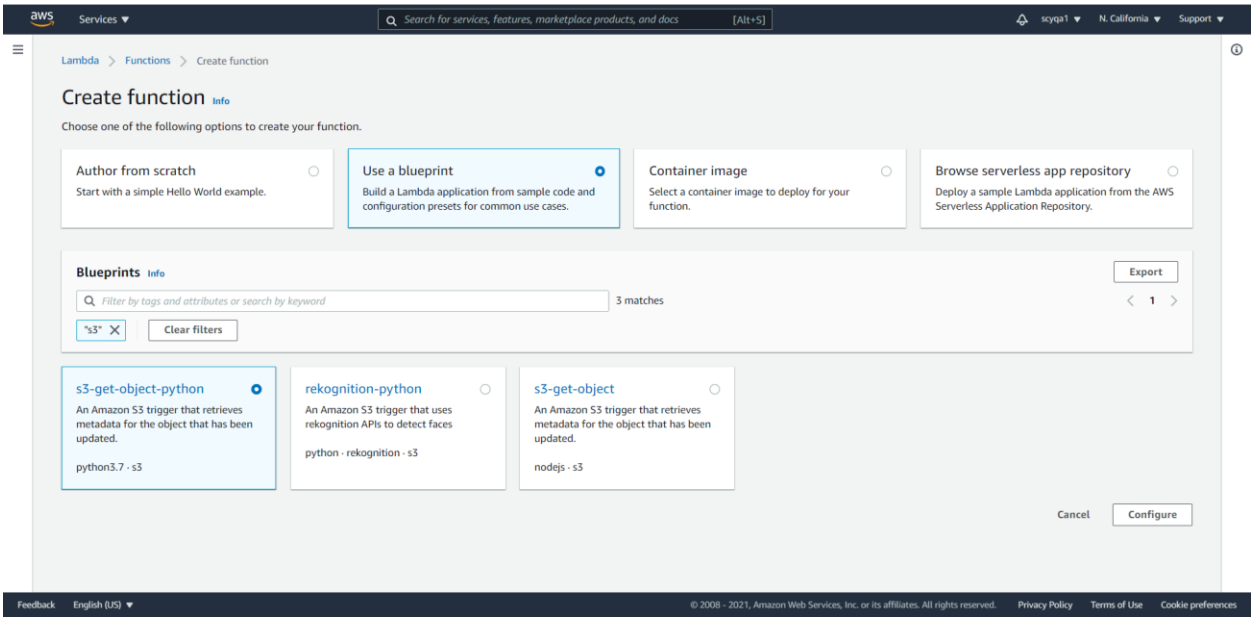


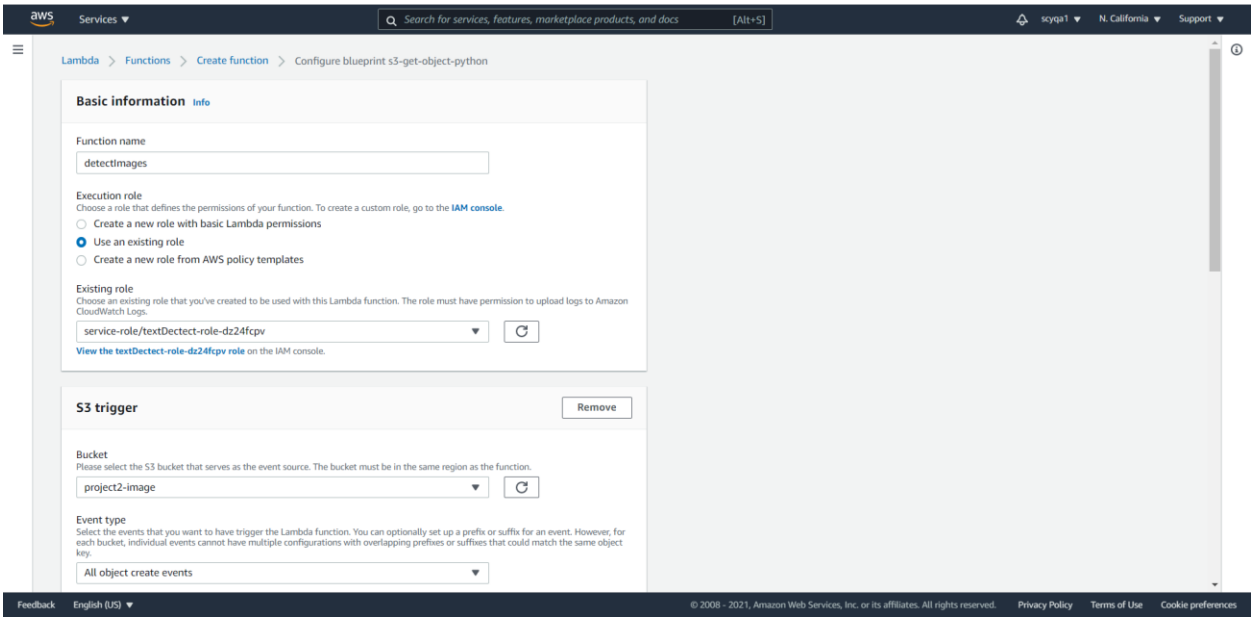
2) Recognize the character present in the text image you placed in the S3 bucket, using AWS API

For this question, I used two methods.

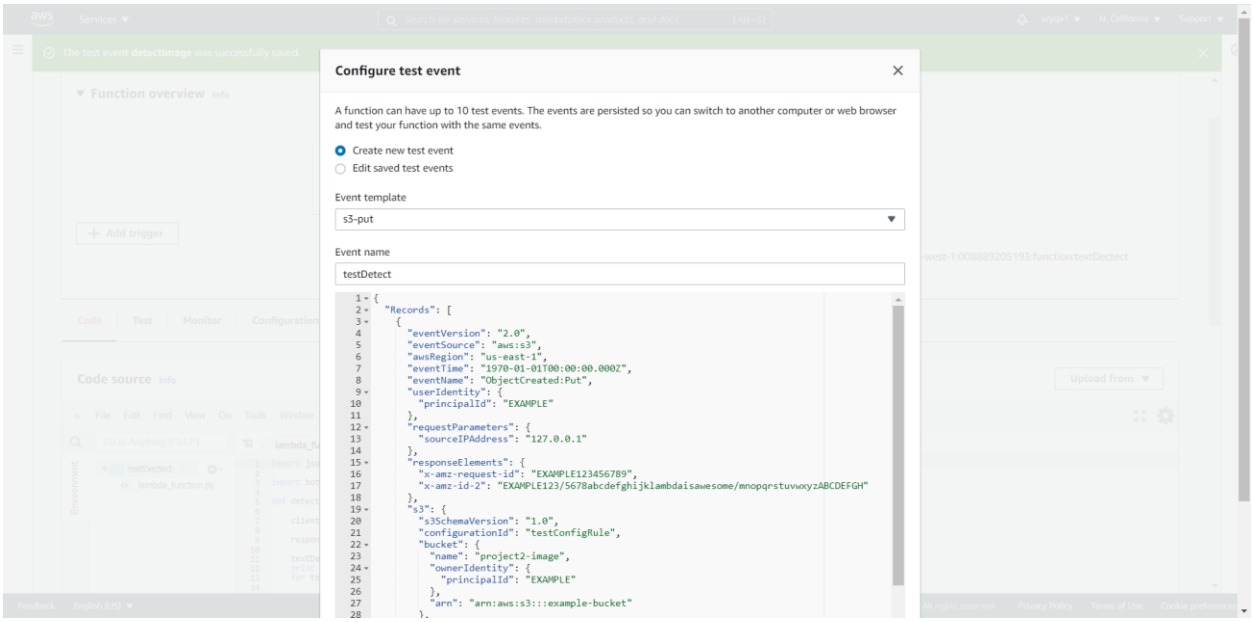
First method: console

Create lambda function



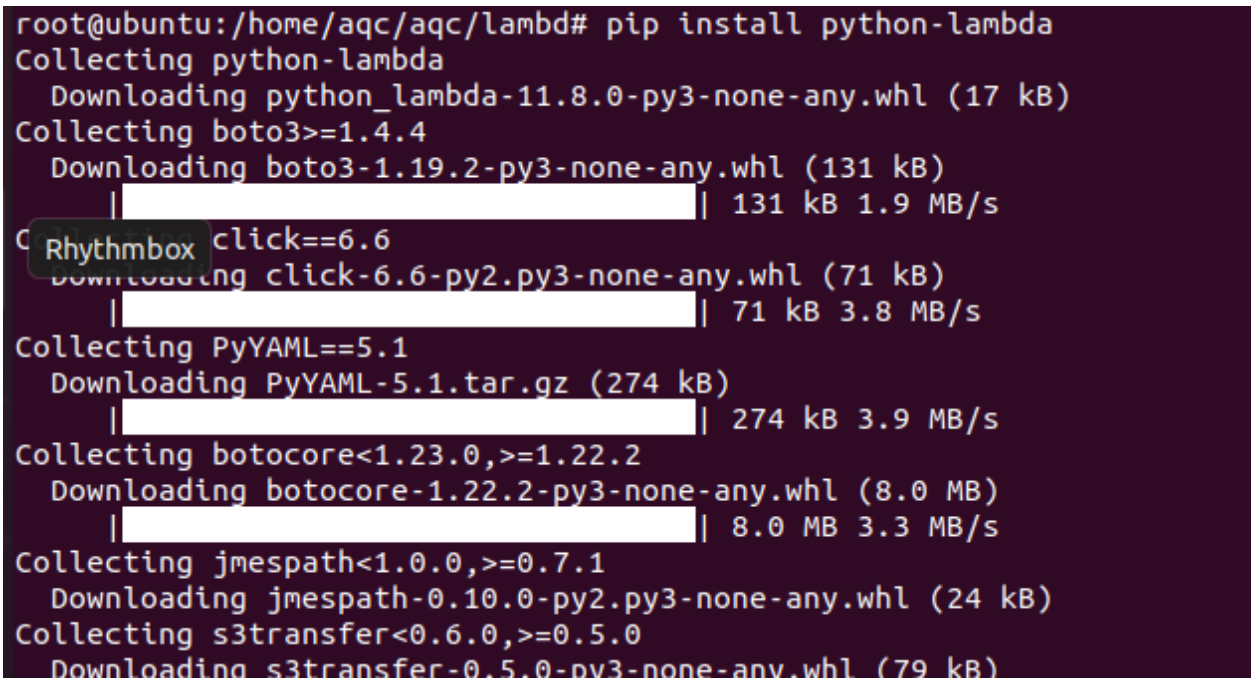


Configure test event



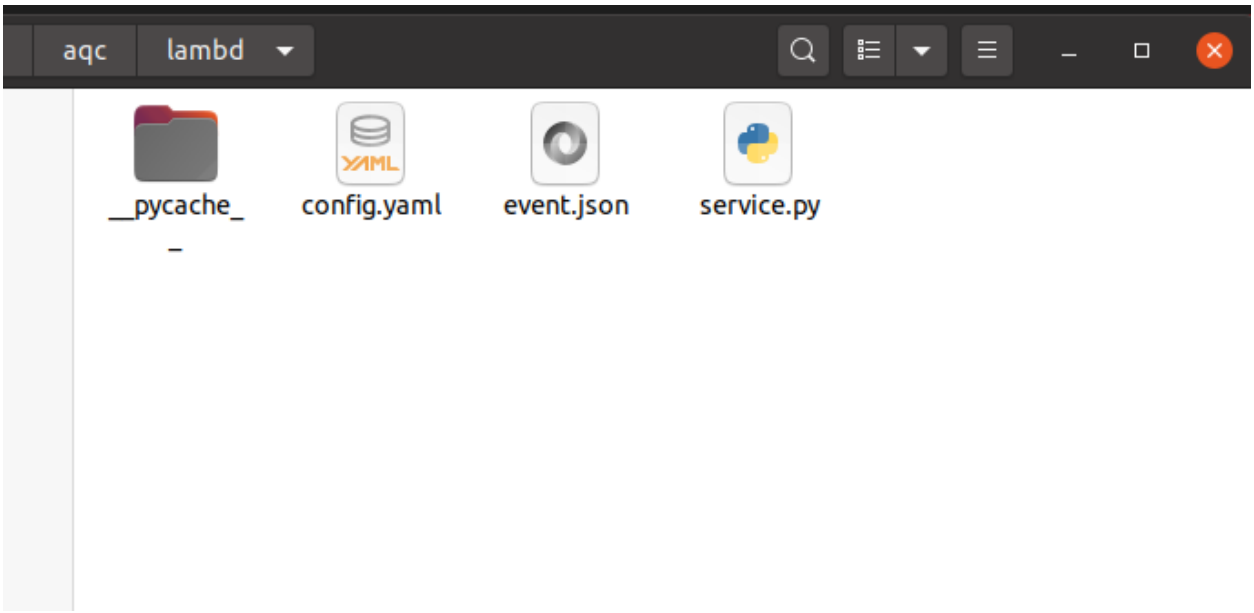
Second method: coding

Install lambda

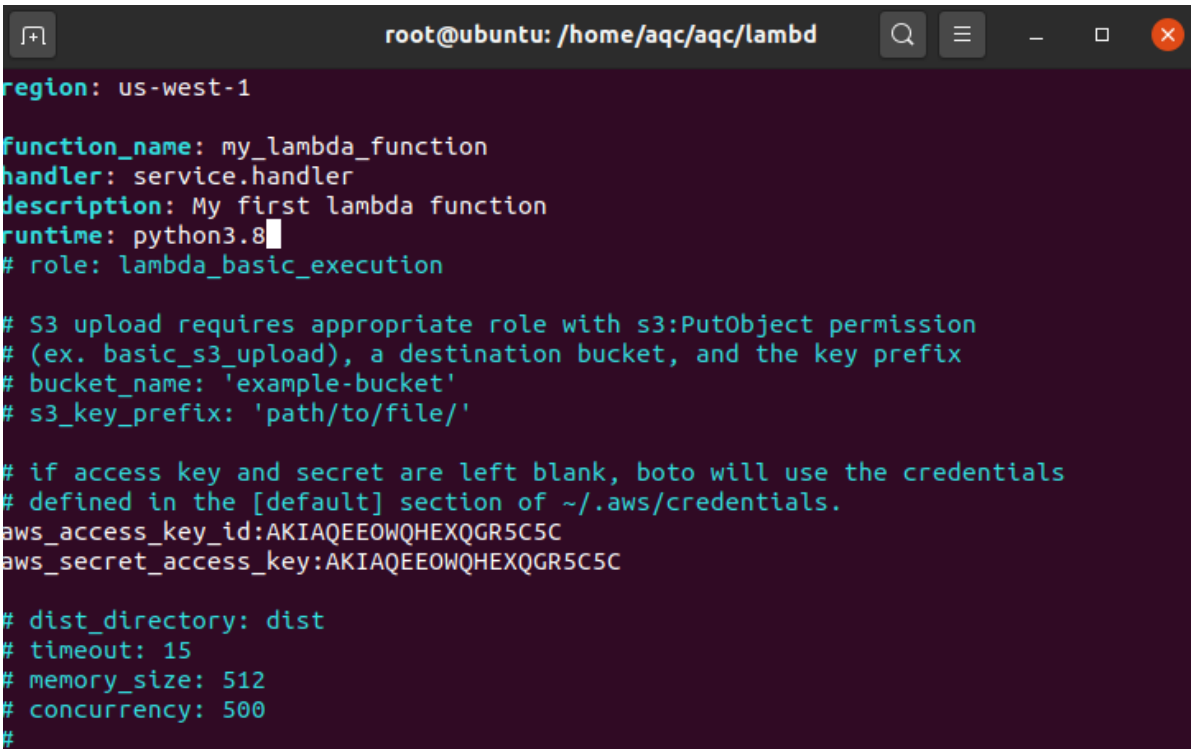


Initialize the project

```
root@ubuntu:/home/aqc/aqc/lambd# lambda init
```



Config.yaml



Service.py

root@ubuntu: /home/aqc/aqc/lambd

```
import json
import urllib.parse
import boto3

print('Loading function')

s3 = boto3.client('s3')

def detect_text(photo, bucket):

    client=boto3.client('rekognition')

    response=client.detect_text(Image={'S3Object':{'Bucket':project2-image,'Name
':imageFile.jpg}})
    textDetections=response['TextDetections']
    print ('Detected text\n----------')
    for text in textDetections:
            print ('Detected text:' + text['DetectedText'])
            print ('Confidence: ' + "{:.2f}".format(text['Confidence']) + "%")
            print ('Id: {}'.format(text['Id']))
            if 'ParentId' in text:
                print ('Parent Id: {}'.format(text['ParentId']))
                                                              19,1          11%
```

LibreOffice Writer

Running result

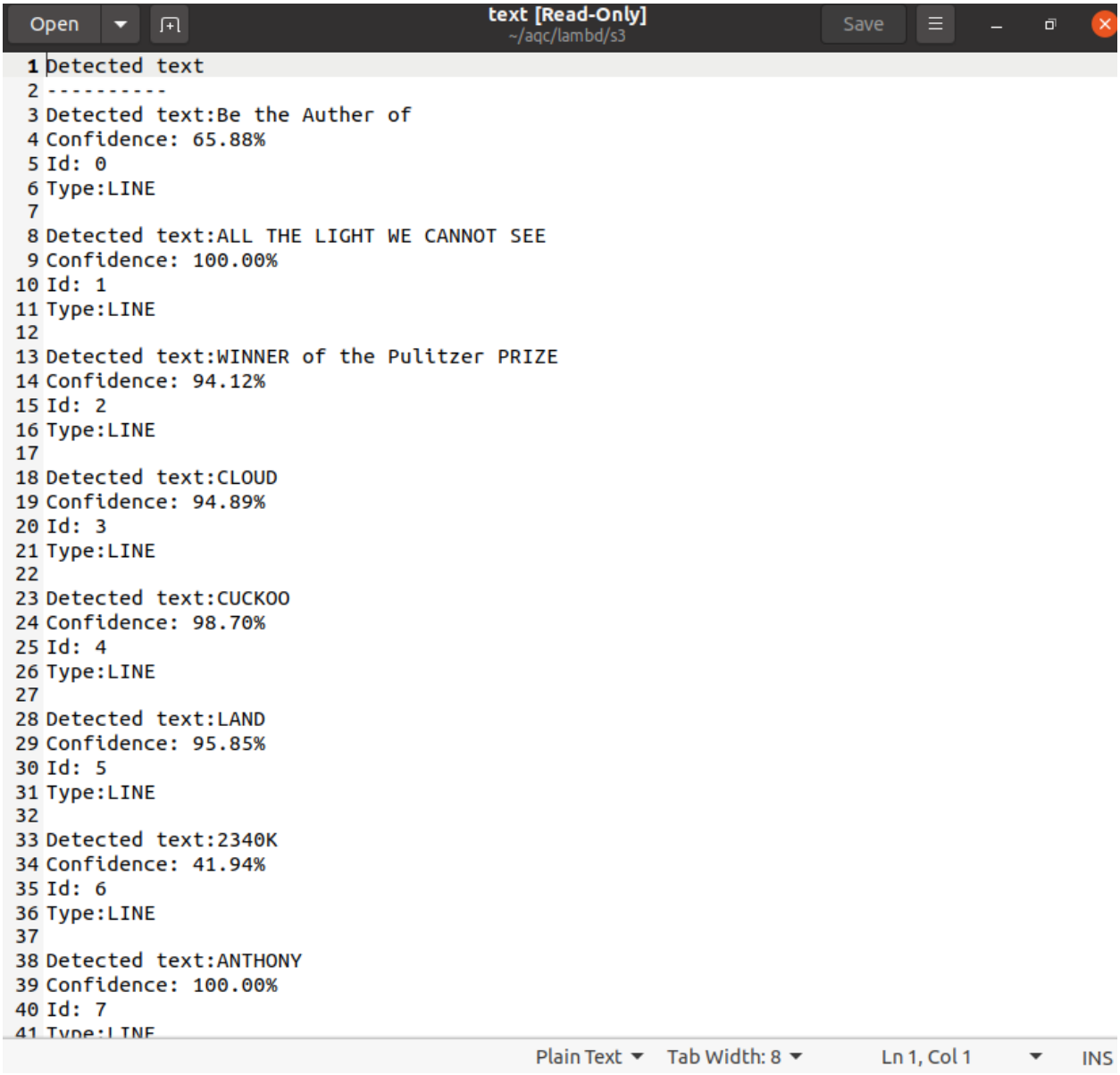
```
root@ubuntu:/home/aqc/aqc/lambd# lambda invoke -v
Loading function
Detected text
----------
Detected text:Be the Auther of
Confidence: 65.88%
Id: 0
Type:LINE

Detected text:ALL THE LIGHT WE CANNOT SEE
Confidence: 100.00%
Id: 1
Type:LINE

Detected text:WINNER of the Pulitzer PRIZE
Confidence: 94.12%
Id: 2
Type:LINE

Detected text:CLOUD
Confidence: 94.89%
Id: 3
Type:LINE
```

```
Detected text:2340K
Confidence: 41.94%
Id: 27
Parent Id: 6
Type:WORD

Detected text:ANTHONY
Confidence: 100.00%
Id: 28
Parent Id: 7
Type:WORD

Detected text:DOERR
Confidence: 100.00%
Id: 29
Parent Id: 8
Type:WORD

Text detected: 30
None

execution time: 1.25934410s
function execution timeout: 15s
root@ubuntu:/home/aqc/aqc/lambd#
```

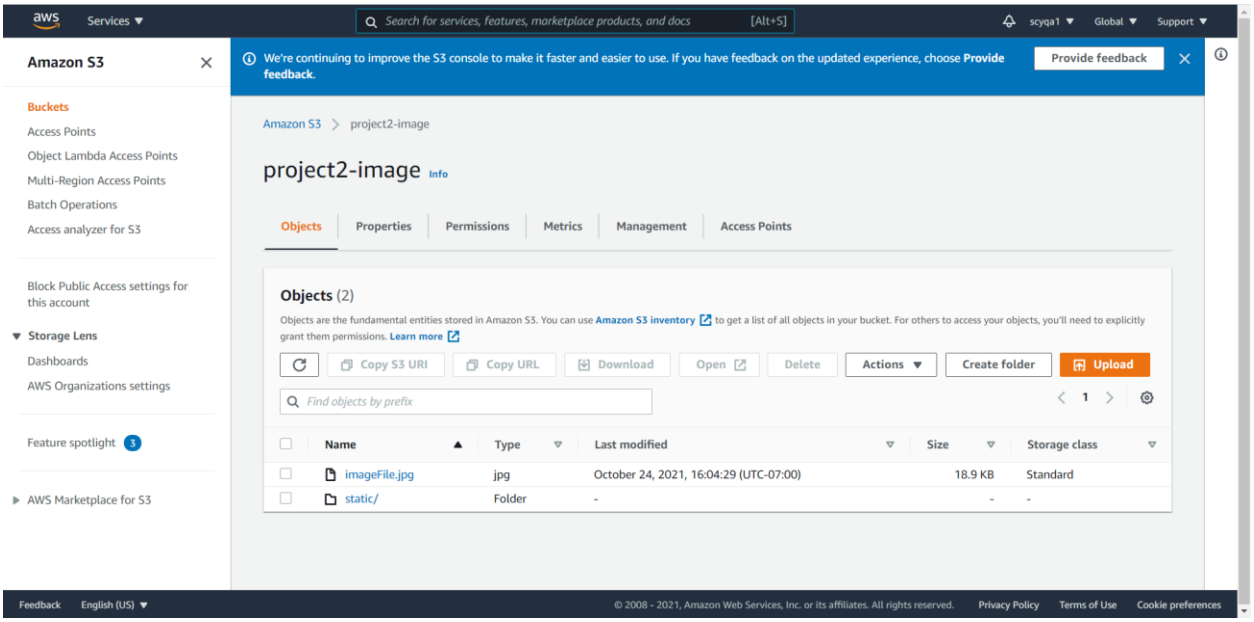
Deploy to website

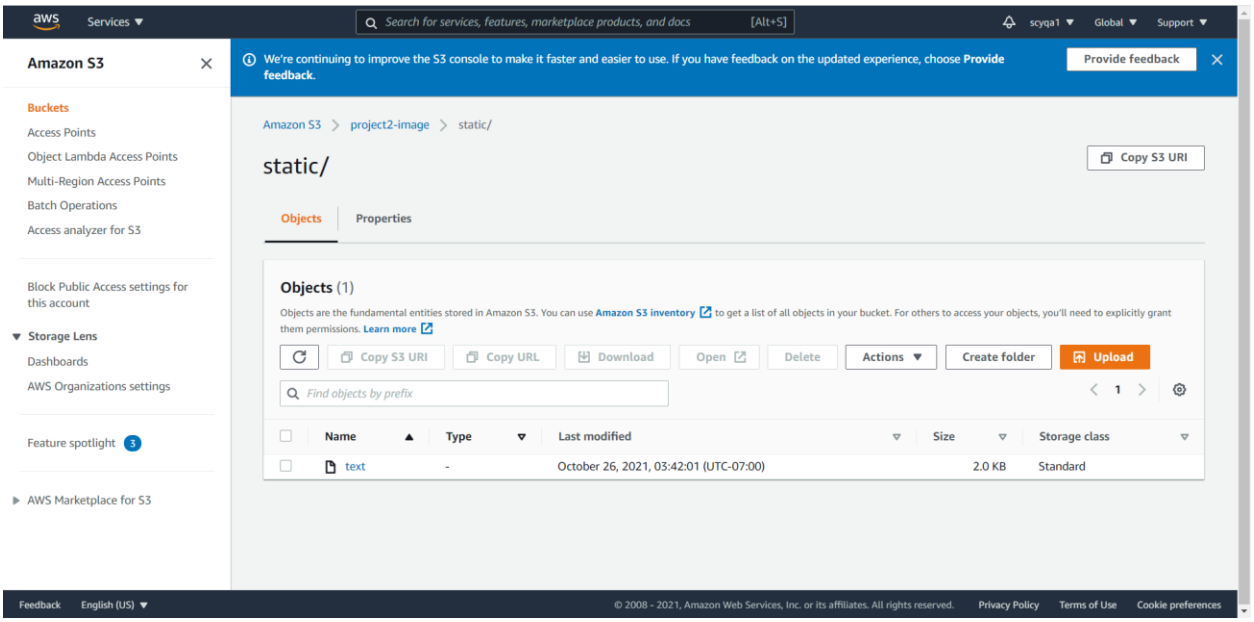
```
root@ubuntu:/home/aqc/aqc/lambd# lambda deploy
Gathering pip packages
```

3) Upload the extracted text from the above step back to S3 bucket. This is your Ouput (Result) text file.




```
[root@ip-172-31-10-156 s3]# aws s3 cp ./static s3://project2-image/static --recu
rsive
upload: static/text to s3://project2-image/static/text
[root@ip-172-31-10-156 s3]#
```


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
Global
Support
Amazon S3
We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose Provide feedback.
Provide feedback
Buckets
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
Access analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
AWS Organizations settings
Feature spotlight 3
AWS Marketplace for S3
Amazon S3 > project2-image
project2-image Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (2)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
imageFile.jpg
jpg
October 24, 2021, 16:04:29 (UTC-07:00)
18.9 KB
Standard
static/
Folder
-
-
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences


aws
Services
Search for services, features, marketplace products, and docs [Alt+S]
scyqa1
Global
Support
Amazon S3
We're continuing to improve the S3 console to make it faster and easier to use. If you have feedback on the updated experience, choose Provide feedback.
Provide feedback
Buckets
Access Points
Object Lambda Access Points
Multi-Region Access Points
Batch Operations
Access analyzer for S3
Block Public Access settings for this account
Storage Lens
Dashboards
AWS Organizations settings
Feature spotlight 3
AWS Marketplace for S3
Amazon S3 > project2-image > static/
static/
Copy S3 URI
Objects
Properties
Objects (1)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
text
-
October 26, 2021, 03:42:01 (UTC-07:00)
2.0 KB
Standard
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences